

IMPLEMENTATION OF PREDICTIVE MAINTENANCE BY PREDICTING THE REMAINING USEFUL LIFE OF MACHINE

UNDERGRADUATE THESIS

**Submitted to the Undergraduate Program in Industrial Engineering in Partial
Fulfilment of Requirement for the Degree of Sarjana Teknik
at the Faculty of Industrial Technology
Universitas Islam Indonesia**



Name : Salwa Nur Rahma
Student Number : 21522322

**UNDERGRADUATE PROGRAM IN INDUSTRIAL ENGINEERING
FACULTY OF INDUSTRIAL TECHNOLOGY
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA**

2025

AUTHENTICITY STATEMENT

For the sake of Allah SWT, I admit this work is the result of my own work, except for the excerpts and summaries from which I have explained the source. If in the future, it turns out that my confession is proven to be untrue and violates the legal regulations in the paper and intellectual property rights. In that case, I am willing to get a diploma that I have received to be withdrawn by Universitas Islam Indonesia.

Yogyakarta, August 5, 2025



(Salwa Nur Rahma)

21522322

RESEARCH COMPLETION LETTER



FAKULTAS
TEKNOLOGI INDUSTRI

Gedung KH. Mas Mansur
Kampus Terpadu Universitas Islam Indonesia
Jl. Kaliurang km 14,5 Yogyakarta 55584
T. (0274) 898444 ext. 4110, 4100
F. (0274) 895007
E. fti@uii.ac.id
W. fti.uii.ac.id

SURAT KETERANGAN PENELITIAN

Nomor : 131/Ka.lab SIMANTI/20/Lab.SIMANTI/VIII/2025

Assalamu'alaikum Warohmatullahi Wabarokaatuh

Dengan hormat,

Yang bertanda tangan dibawah ini, menerangkan bahwa:

Nama : Salwa Nur Rahma

Nim : 21522322

Jurusan : Teknik Industri

Dosen Pembimbing : Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM

Menyatakan bahwa mahasiswa tersebut diatas telah melaksanakan penelitian tugas akhir dengan judul **"IMPLEMENTATION OF PREDICTIVE MAINTENANCE BY PREDICTING THE REMAINING USEFUL LIFE OF MACHINE"**. Mulai pelaksanaan penelitian 3 Maret 2025 sampai 5 Agustus 2025.

Demikian surat keterangan penelitian ini kami buat. Atas perhatiannya dan kerja samanya kami mengucapkan terima kasih.

Wassalamu'alaikum Warohmatullahi Wabarokaatuh

Yogyakarta, 05 Agustus 2025

Kepala Laboratorium
Sistem Manufaktur Terintegrasi


Putri Dwi Annisa S.T., M.Sc.

SUPERVISOR APPROVAL SHEET

**IMPLEMENTATION OF PREDICTIVE MAINTENANCE BY PREDICTING THE
REMAINING USEFUL LIFE OF MACHINE**



Yogyakarta, August 5, 2025

Supervisor

(Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM)

EXAMINERS' APPROVAL PAGE

IMPLEMENTATION OF PREDICTIVE MAINTENANCE BY PREDICTING THE REMAINING USEFUL LIFE OF MACHINE

UNDERGRADUATE THESIS

Written by:

Name : Salwa Nur Rahma
Student Number : 21522322

Has been defended before the board of examiners in partial fulfillment of the requirement for Bachelor Degree in Industrial Engineering Department at the Faculty of Industrial Technology Universitas Islam Indonesia

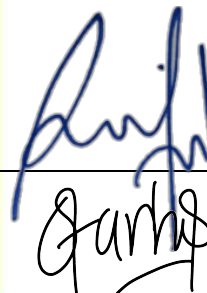
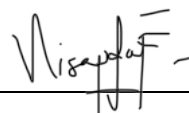
Yogyakarta, August 21, 2025

Board of Examiners

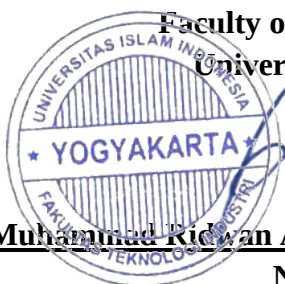
Ir. Muhammad Ridwan Andi Purnomo,
S.T., M.Sc., Ph.D., IPM
Chairman

Annisa Uswatun Khasanah, S.T., M.Sc.
Member I

Putri Dwi Annisa, S.T., M.Sc.
Member II

Acknowledged by,
Head of Undergraduate Program in Industrial Engineering
Faculty of Industrial Technology
Universitas Islam Indonesia



Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM
NIK. 015220101

DEDICATION PAGE

With gratitude to Allah SWT, this thesis is dedicated to the people who have supported me throughout my education, especially my family, and to myself, who kept going.

Thank you for making me see this adventure through to the end.

MOTTO

“My heart is at ease knowing that what was meant for me will never miss me, and that what misses me was never meant for me.”

Umar ibn Al-Khattab

PREFACE

Bismillahirrahmanirrahim

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillah, all praise and gratitude be to Allah SWT for His blessings, guidance, and mercy, through which the author was able to complete this final project report entitled "Implementation of Predictive Maintenance by Predicting Remaining Useful Life of Machine" properly and on time. This report is prepared as part of the academic requirement to obtain a Bachelor's degree in Industrial Engineering, Faculty of Industrial Technology, Universitas Islam Indonesia.

The completion of this report would not have been possible without the support and assistance of many individuals. Therefore, the author would like to express sincere gratitude to:

1. Professor Dr. Ir. Hari Purnomo, M.T., IPU., ASEAN.Eng., as the Dean of the Faculty of Industrial Technology, Universitas Islam Indonesia.
2. Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM, as the Head of the Industrial Engineering Undergraduate Program and my supervisor, who provided guidance and motivation throughout the preparation of this report.
3. The author's beloved family, Bapak Nurcholis, Ibu Tutik, and dek Amri for their endless prayers, encouragement, and unwavering support.
4. The author's dearest friends from, 3S, Djawara, Website, Shopee Tanam, and Chili's who have been a source of encouragement, stories, and support for the author.
5. My fellow students and peers who have provided support, shared knowledge, and offered encouragement, whether directly or indirectly.
6. All other parties who have provided support and assistance, but whose names cannot be mentioned individually.

The author realizes that this report is far from perfect. Therefore, the author eagerly welcomes constructive criticism and suggestions for future improvements. Hopefully, this report will be useful and serve as a reference for relevant parties.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Yogyakarta, August 5, 2025



Salwa Nur Rahma

NIM 21522322

ABSTRACT

Machine maintenance is one of the most important aspects for maintaining productivity. Effective maintenance not only prevents damage, but also extends the service life of equipment and reduces maintenance costs. Therefore, this study aims to determine the accuracy of Remaining Useful Life (RUL) predictions using a hybrid Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) model, as well as to determine the optimal time to perform maintenance in order to minimize total maintenance costs. The data used consists of machine condition parameters as input and actual RUL values as the target. The hybrid LSTM-GRU model was developed with hyperparameter tuning using Keras Tuner to obtain the best architecture combination. Evaluation was conducted using the Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2) metrics on the test data. After obtaining the RUL prediction results, maintenance cost optimization modeling was performed using the Weibull distribution approach and cost model. This model considers preventive costs and emergency breakdown repair costs based on the probability of failure at each point in time before the actual failure. The results of the study show that the hybrid LSTM-GRU model is capable of producing RUL predictions with excellent performance. In addition, the optimization model shows that performing maintenance before failure can significantly reduce total maintenance costs.

Keywords: Predictive Maintenance, RUL, LSTM, GRU, Cost Optimization

TABLE OF CONTENT

AUTHENCITY STATEMENT	ii
RESEARCH COMPLETION LETTER	iii
SUPERVISOR APPROVAL SHEET	iv
EXAMINERS' APPROVAL PAGE.....	v
DEDICATION PAGE.....	vi
MOTTO.....	vii
PREFACE	viii
ABSTRACT	ix
TABLE OF CONTENT	x
LIST OF TABLES	xii
LIST OF FIGURES.....	xiii
CHAPTER I INTRODUCTION	1
1.1 Background.....	1
1.2 Problem Formulation.....	4
1.3 Research Objective	4
1.4 Benefits of Research.....	4
1.5 Research Limitation.....	5
CHAPTER II LITERATURE REVIEW	6
2.1 Literature Review	6
2.2 Theoretical Basis	12
2.2.1 Maintenance.....	12
2.2.2 Predictive Maintenance	12
2.2.3 Machine Learning.....	13
2.2.4 Deep Learning	14
2.2.5 Long Short-Term Memory (LSTM).....	15
2.2.6 Gated Recurrent Unit (GRU).....	17
2.2.7 Internet of Things	18
2.2.8 Optimization Model.....	19
CHAPTER III RESEARCH METHODOLOGY	20
3.1 Research Object.....	20

3.2	Data Collection	20
3.3	Research Flow	20
CHAPTER IV DATA COLLECTING AND PROCESSING		25
4.1	Data Collection	25
4.2	Data Processing	29
4.2.1	Remaining Useful Life (RUL) Prediction	29
4.2.2	Maintenance Cost Calculation with Optimization Model	37
CHAPTER V DISCUSSION		43
5.1	Remaining Useful Life (RUL) Prediction Using the LSTM-GRU Hybrid Model....	43
5.2	Maintenance Time Optimization Using a Cost Optimization Model.....	45
5.3	Study Limitations	47
CHAPTER VI CLOSING		49
6.1	Conclusion.....	49
6.2	Sugesstion.....	49
BIBLIOGRAPHY		51

LIST OF TABLES

Table 2.1 State of Arts.....	11
Table 4.1 Batik Printer Machine Data Parameters	25
Table 4.2 General Maintenance Cost Data.....	27
Table 4.3 Preventive Maintenance Cost Data	28
Table 4.4 Predictive Maintenance Cost Data	28
Table 4.5 Corrective Maintenance Cost Data	28
Table 5. 1 Maintenance Strategy Comparison	46

LIST OF FIGURES

Figure 2.1 LSTM Architecture	15
Figure 2.2 GRU Architecture	17
Figure 3.1 Research Flow Chart.....	21
Figure 4.1 Import Library.....	29
Figure 4.2 Load Data.....	30
Figure 4.3 Pseudocode of Preprocessing Data	30
Figure 4.4 Preprocessing Data	31
Figure 4.5 Pseudocode of Define Model.....	31
Figure 4.6 Define Model	31
Figure 4.7 Pseudocode of Hyperparameter Optimization.....	32
Figure 4.8 Hyperparameter Optimization	32
Figure 4.9 Build the Best Model	33
Figure 4.10 Evaluate Model	33
Figure 4.11 Plot Actual vs Predicted RUL.....	33
Figure 4.12 Compute RMSE and R-squared.....	33
Figure 4.13 RUL Visualization	34
Figure 4.14 Actual vs Predicted RUL Graph	34
Figure 4.15 Plot Neural Network Architecture	35
Figure 4.16 Neural Network Architecture.....	36
Figure 4.17 Training Data	36
Figure 4.18 Testing Data	37
Figure 4.19 Import Library.....	37
Figure 4.20 Determining Weibull Parameters.....	38
Figure 4.21 Weibull Visualization	38
Figure 4.22 Weibull Visualization	38
Figure 4.23 Setting Parameters and Function	39
Figure 4.24 Cost Calculation.....	40
Figure 4.25 Finding Optimal Time	40
Figure 4.26 Optimal Time and Cost Result.....	41
Figure 4.27 Cost Visualization.....	42
Figure 4.28 Cost vs Time Graph	42
Figure 5.1 Hyperparameter Tuning Result.....	43
Figure 5.2 Evaluation Result.....	44
Figure 5.3 Maintenance Cost Comparison.....	46

CHAPTER I

INTRODUCTION

1.1 Background

In the manufacturing industry, machine maintenance is one of the most important aspects for maintaining productivity. It is because the capability and reliability of machines greatly affect the pace of production activities (Arisandhi, 2023). According to European Committee for Standardization (2017), maintenance is all technical, administrative, and managerial actions performed during the life cycle of a piece of equipment to keep it in or return it to a condition in which it can perform its required functions. These maintenance activities play an important role in supporting the smooth running of the company's operations at all times. Effective maintenance not only prevents damage, but also extends the service life of equipment and reduces maintenance costs, thereby contributing to greater operational efficiency (Joseph Nnaemeka Chukwunweike et al., 2024). Therefore, an appropriate maintenance strategy is needed to maintain the performance and reliability of the machine.

According to Carvalho et al. (2019), the maintenance strategies commonly used by industries are divided into three types, which are preventive maintenance, corrective maintenance, and predictive maintenance. Preventive maintenance is a maintenance method carried out on a scheduled basis to prevent unexpected damage (Pamungkas et al., 2021). As a result, machines that undergo this maintenance are guaranteed to operate smoothly and remain in good condition for use in production processes (Sitinjak & Silalahi, 2023). On the other hand, corrective maintenance is maintenance performed after a machine has experienced damage and can no longer function properly (Pamungkas et al., 2021).

In industry, machine breakdowns are still very common. These breakdowns occur suddenly and can disrupt the smooth running of production processes, resulting in significant losses. To reduce this risk, companies implement preventive maintenance strategies, which involve conducting regular maintenance before breakdowns occur. However, implementing preventive maintenance too frequently without considering the actual condition of the machinery can result in a high maintenance frequency. This leads to increased total maintenance costs, both in terms of maintenance expenses themselves and the potential loss of production time while the machinery is shut down for maintenance.

One of the companies implementing textile production processes in Pekalongan, PT. X, utilizes a batik printing machine as part of its production line. This machine is critical for

transferring intricate dye patterns onto fabric with precision, directly impacting product quality and production efficiency. To maintain its operation, the company conducts preventive maintenance every 30 days, resulting in approximately 160 maintenance activities per year.

Despite implementing preventive maintenance, the companies still faces challenges in its production process. These preventive activities can sometimes lead to unnecessary downtime and resource use, as functional components may be replaced even when not required. Moreover, there remains a possibility of machine failures that necessitate corrective maintenance during operations, resulting in unexpected downtime, higher repair costs, and the risk of more severe damage due to undetected component degradation.

Although preventive maintenance and corrective maintenance play an important role in machine maintenance, these methods have several drawbacks. Preventive maintenance, which is designed to reduce the risk of damage before it occurs, results in more frequent maintenance. This can lead to unnecessary costs. According to Hamasha et al. (2023), the main drawback of preventive maintenance is the increase in long-term maintenance costs if the machine continues to operate until it breaks down, as well as unexpected damage that can disrupt other maintenance work. Meanwhile, corrective maintenance, which is carried out to address machine breakdowns after they occur, often results in increased downtime, higher costs, and safety risks (Ewin & Oye, 2025).

To overcome the limitations of these two maintenance strategies, predictive maintenance offers a more effective alternative. Predictive maintenance is a maintenance method that uses data and analysis to predict when a machine will break down, so that maintenance can be performed just before the machine enters a critical phase. This maintenance can predict trends, behavioral patterns, and correlations using statistical models or machine learning to anticipate delayed failures earlier, thereby improving decision-making processes related to maintenance activities, particularly to avoid downtime (Zonta et al., 2020). Predictive maintenance can save up to 12% and 30% on repair and scheduled maintenance costs, respectively, and eliminate up to 70% of failures (Maher & Danouj, 2020). This demonstrates that predictive maintenance has great potential as a maintenance strategy for reducing costs and improving operational efficiency.

One of the main concepts in predictive maintenance is Remaining Useful Life (RUL). RUL is an estimate of the period during which a machine can operate before requiring repair or replacement (Achouch et al., 2022). This period can be expressed in terms of days, rounds,

cycles, or other quantities. Among the various approaches used to predict RUL, machine learning-based approaches are considered the most suitable because they can handle problems involving multiple variables, such as voltage, flow, and current (Kang et al., 2021). Given that machine parameters change with time, RUL prediction is naturally formulated as a time series problem. Time series data is defined as data collection points arranged in chronological order with consistent time intervals between observations, which helps to understand and predict the temporal dynamics of a particular phenomenon (Le, 2024).

Before the development of deep learning methods, RUL predictions were mostly done using traditional statistical and machine learning methods, such as ARIMA, SVM, Random Forest, or MLP. However, these methods have limitations in capturing long term temporal patterns such as time series data, so approaches based on Recurrent Neural Networks (RNN) were developed.

One of the most widely used methods is Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTM is a method that can be used to learn patterns in time series data (Ningrum et al., 2021). This method can handle long-term dependencies but often requires longer computation time. To address this limitation, the Gated Recurrent Unit (GRU) was developed as a variant of LSTM. Although similar to LSTM, GRU has a simpler architecture with fewer parameters, making it easier to train and more efficient, but it sometimes results in the loss of important information (Khairunisa & Hendikawati, 2024). Therefore, this study proposes the use of a hybrid LSTM-GRU model. According to Wang et al. (2023), the use of LSTM and its variants is highly recommended for estimating RUL and long-term degradation conditions. This approach combines the advantages of both models, LSTM is capable of capturing long-term dependencies in data, while GRU offers better computational efficiency. This enables more accurate and faster predictions compared to other models.

Although predictive maintenance offers many advantages, its implementation is not without challenges. One of the main challenges is determining the most appropriate maintenance timing based on RUL prediction results. This is because RUL predictions alone are not sufficient to provide practical decisions if they are not accompanied by an appropriate maintenance scheduling strategy. If maintenance is performed too early, it will increase the frequency and cost of maintenance that is not yet necessary. Conversely, if maintenance is performed too close to the end of the RUL, there is a risk of sudden failure, leading to downtime and high corrective maintenance costs. Therefore, an approach is needed that can optimize the

timing of maintenance based on RUL prediction results to minimize total costs. The optimal timing must be determined, which means there will be a trade off between scheduling maintenance earlier or later.

Upon completion of this study, the results obtained are expected to accurately predict the RUL of machines using the hybrid LSTM-GRU method. This RUL prediction serves as the basis for determining the optimal maintenance schedule, thereby optimizing maintenance strategies in terms of both time and cost. As a result, the optimal maintenance timing can be identified, enabling companies to reduce the risk of sudden failures, minimize unnecessary maintenance costs, and enhance overall operational efficiency and resource utilization.

1.2 Problem Formulation

Based on the background described above, the problem formulation that can be raised in this study is as follows:

1. What is the accuracy level of the LSTM-GRU hybrid model built to predict Remaining Useful Life (RUL)?
2. When is the best time to perform maintenance to minimize total cost?
3. How do the proposed methods compare with the previously used approach?

1.3 Research Objective

The following are the research objectives in accordance with the problem formulation that has been compiled:

1. To determine the accuracy level of the Remaining Useful Life (RUL) prediction of the hybrid LSTM-GRU model.
2. To determine the best time to perform maintenance in order to minimize total costs.
3. To compare the performance of the proposed methods against the previously applied maintenance approach.

1.4 Benefits of Research

The following are the expected benefits of this research:

1. For the author, this research was conducted as a requirement for obtaining a Bachelor's degree (S1) in Industrial Engineering, Faculty of Industrial Technology, Universitas Islam

Indonesia. Additionally, this research provides experience, insights, and knowledge in the implementation of Industrial Engineering principles.

2. For companies, the results of this research can serve as a reference for companies in implementing predictive maintenance to optimize machine maintenance.
3. For the academic world, this research can serve as a benchmark and reference for future studies in optimizing predictive maintenance models in industry.

1.5 Research Limitation

The limitations of this study are as follows:

1. The study uses data that was not collected directly (secondary data).
2. The study only considers cost aspects and does not consider reliability aspects.
3. The maintenance costs used are those at the time the data was collected and are considered to remain constant throughout the study period. This is because some maintenance fluids are imported, and there are fluctuations in the dollar exchange rate.
4. The study does not aim to compare the hybrid LSTM-GRU method with other methods.

CHAPTER II

LITERATURE REVIEW

2.1 Literature Review

Study by Joseph et al. (2022) shows how predictive maintenance can be used for robot cells using big data from an automotive company. They developed a Long Short-Term Memory (LSTM) model with hyperparameters. The results show that the model can predict the day of failure pretty accurately based on the severity of the alarm. This research aids in scheduling maintenance activities and provides a solid foundation for companies in implementing predictive maintenance.

Study by H. Wu et al. (2020) proposes a Long Short-Term Memory (LSTM)-based Recurrent Neural Network (RNN) model to detect manufacturing system damage and predict its future health condition for predictive maintenance. In this model, the time-sequential characteristics of historical sensor data are taken into account. To test the validity of the proposed model, it was applied to a bearing vibration dataset. The results obtained show that the model has good accuracy in critical conditions (warnings and failures) of the bearing.

Study by Krishna et al. (2024) discusses the limitations of previous research on Battery Management Systems (BMS) by proposing an Internet of Things (IoT) approach, a Long Short-Term Memory (LSTM) model, and a predictive model to analyze battery aging in BMS. This model was trained using a verified NASA Li-ion battery dataset and applied to real time data for Remaining Useful Life (RUL) prediction. Experimental results show that this model achieves higher accuracy than previous models, reducing the Root Mean Square Error (RMSE) from 0.0949 to 0.00665.

Study by de Pater & Mitici (2023) Rproposes a Long Short-Term Memory (LSTM) autoencoder to improve health indicators in aircraft systems. These health indicators are then used to predict the Remaining Useful Life (RUL) of aircraft systems using a similarity-based matching approach. This study illustrates the method on a turbofan engine. The results show that considering operational conditions improves the monotonicity of health indicators by 97%. Additionally, the approach yields accurate RUL estimates with a Root Mean Square Error (RMSE) of 2.67%, and a 19% reduction in RMSE compared to existing supervised learning models.

Study by Yousuf et al. (2022) proposes a Long Short-Term Memory Network (LSTM)-based regression model for predicting Remaining Useful Life (RUL) in Ring Oscillator (RO) circuits that utilize the most important electrical features of the device. The LSTM network is capable of capturing temporal dependencies in time-series data and eliminating the vanishing gradient problem encountered in conventional Recurrent Neural Networks (RNNs). This has shown that the decrease in RO frequency is fundamentally dependent on three main factors: operating temperature, voltage, and device aging parameters. The research results indicate that over 90% of RUL prediction cases for stages 13 and 21 are constrained by supply voltage variations ranging from 0.7 V to 0.9 V, with prediction deviations of 2 to 6 days.

Study by Wang et al. (2023) proposes a Self-Attention Long Short-Term Memory (SA-LSTM) prediction model for lithium-ion batteries. The proposed prediction model enables the estimated values of health indicators for the next cycle to be repeatedly estimated using the one-step-ahead method. Additionally, the Remaining Useful Life (RUL) value can be predicted after the predicted signal decreases. The results obtained demonstrate the effectiveness and advantages of the proposed prediction method.

Study by Yi et al. (2025) proposes a method for predicting the Remaining Useful Life (RUL) of proton exchange membrane fuel cells (PEMFC) based on the Matrix Long Short-Term Memory (M-LSTM) network. The results of the study indicate that, compared to the one-step LSTM prediction model, M-LSTM reduces MAE and MSE by 76.3% and 58.2%, respectively, while increasing R^2 by 2.3%. In multi-step predictions, M-LSTM reduces MAE and MSE by 79.4% and 60.4%, respectively, while increasing R^2 by 5.2%.

Study by Cahyani et al. (2023) discusses the prediction of national staple food prices. The study chose the Long Short-Term Memory (LSTM) method because the data used is non-linear and time series-based. Based on normality tests, four basic commodity prices were selected: cooking oil prices, bulk cooking oil prices, fresh chicken meat prices, and low-quality rice prices. Additionally, the Adaptive Gradient (ADAGRAD), Adaptive Moment Gradient (ADAM), and Root Mean Square Error (RMSProp) optimization methods were used to optimize prediction results as test scenarios. Based on the experiments, the best RMSE model evaluation values for cooking oil prices, bulk cooking oil prices, fresh chicken meat prices, and low-quality rice prices were obtained in the following order: 0.0313, 0.0531, 0.0937, and 0.0492.

Study by Ul Haque et al. (2025) proposes a cost-based maintenance approach for multi-component systems, with the aim of minimizing total production costs lost due to downtime. This research predicts Remaining Useful Life (RUL) using a CNN model based on C-MAPSS simulation data, then integrates the results into a developed mathematical cost optimization model. Furthermore, a Genetic Algorithm is used to determine the most efficient maintenance schedule sequence. Case study results show that this cost-based approach yields significant savings, with total costs decreasing from USD 10,038,505 to USD 9,994,368, and a 28% reduction in production loss costs. This study confirms that maintenance scheduling based on cost impact can enhance operational efficiency.

Study by Cai et al. (2022) proposes a Condition-Based Maintenance (CBM) method that utilizes Remaining Useful Life (RUL) predictions in systems with many components. This model considers imperfect maintenance using the concept of virtual age, as well as the integration of spare parts management strategies to reduce preparation and inventory costs. Optimization is performed using a Genetic Algorithm to determine maintenance thresholds and spare parts quantities. The results show that this approach can minimize total operational costs and downtime, while improving overall system efficiency.

Study by Li et al. (2020) discusses a proposed Gated Recurrent Unit (GRU)-based approach for predicting Remaining Useful Life (RUL). This study incorporates selected features into a GRU-based Recurrent Neural Network (RNN) to predict RUL. Subsequently, the effectiveness of the proposed method is demonstrated on NASA's Commercial Modular Aero Propulsion System Simulation (C-MAPSS) data. The research results validate that the proposed approach achieves better prediction performance than other methods.

Study by Purnomo (2022) discusses the optimization of capacitated planned maintenance using two deep analysis techniques. First, accurate prediction of binding machine failure based on Mean Time Between Failure (MTBF) data using deep analysis with Gated Recurrent Unit (GRU). Second, optimizing total maintenance costs within the available time. The results of this study indicate that the proposed deep learning data analysis technique can improve MTBF prediction accuracy by up to 66.12% and reduce total maintenance costs by up to 4% compared to the initial model.

Study by Soladoye et al. (2025) developed a stroke prediction system using Gated Recurrent Units (GRUs) and a variant of Recurrent Neural Networks (RNNs) using secondary data from the Afrocentric Stroke Investigative Research and Education Network (SIREN)

dataset. Comparisons were made with traditional machine learning algorithms (Logistic Regression, SVM, KNN) and Long Short-Term Memory (LSTM). The GRU-based system achieved a performance accuracy of 77.48%, an AUC of 0.84, and a prediction time of 0.43 seconds, outperforming all other models. Logistic Regression achieved 73.58%, while LSTM reached 74.88% but with a longer prediction time of 2.23 seconds. The GRU-based system demonstrated superior performance in stroke prediction, offering an efficient and measurable tool for healthcare.

Study by Chen et al. (2025) aims to predict the state of health (SOH) of lithium-ion batteries based on fragmented charging data and an enhanced Gated Recurrent Unit (GRU) neural network. The study developed a GRU model to achieve accurate lithium-ion battery SOH estimation with easier feature extraction and limited training data. The results showed that the built model could accurately estimate battery health conditions with truncated training data, yielding an average error of less than 1%. Additionally, the model was validated on various battery types, demonstrating better generalization capabilities.

Study by Zhi-Wen et al. (2024) performed carbon emission forecasting based on the Gated Recurrent Unit (GRU) model to empirically evaluate data from 16 coastal cities in China from 1997 to 2021. Carbon emissions in China's coastal cities were estimated using eight scenarios from 2022 to 2031 to investigate the combined effects of factors driving carbon emissions. The study results predict an increase in carbon emissions from 2022 to 2031, with growth significantly slower compared to carbon emission changes from 1997 to 2021. Additionally, changes in driving factors significantly influence carbon emissions, and economic size significantly influences carbon reduction.

Study by Yang et al. (2025) proposes a two-stage approach based on Gated Recurrent Unit (GRU) to improve real time flood depth forecasting by incorporating gutter height forecasting. For comparison, Long Short-Term Memory (LSTM) networks and Bidirectional LSTM (BiLSTM) networks were used as reference models alongside the GRU-based model. The results showed that the GRU-based model performed better than the other models in predicting channel levels and flood depth. The Nash-Sutcliffe efficiency (NSE) value for one-hour-ahead flood depth forecasting is 0.488. By integrating the results from the channel level forecasting model, the accuracy of the flood depth forecasting model improves to 0.746. This indicates that the proposed method has significance for advancing early warning systems.

Study by Xia et al. (2024) discusses a new prediction method that uses a Bidirectional Gated Recurrent Unit (BiGRU) network with multi-source feature fusion to predict the State of Health (SOH) and Remaining Useful Life (RUL) of lithium batteries. The uncertainty of RUL prediction results is analyzed using Monte Carlo (MC) simulation. This study utilizes two battery degradation datasets from the NASA Ames Prognostics Center and the University of Oxford. The proposed method can accurately predict the SOH and RUL of batteries. The Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for SOH prediction results can achieve values below 0.3% and 0.6%, respectively. The maximum absolute error of RUL for NASA and Oxford batteries is between 3 and 100, respectively.

Study by Tang et al. (2024) proposes the Gated Recurrent Unit (GRU) method for predicting total electron content (TEC) in the ionosphere at different latitudes and longitudes. This method aims to address the low accuracy issues of traditional neural networks and the challenges faced in training long-term time series. The GRU model was built using TEC data from the European Orbit Determination Center (CODE) from 2016 to 2018, geomagnetic indices, and solar activity indices. A comparative experiment was conducted with the International Reference Ionosphere (IRI) model and the Long Short-Term Memory (LSTM) model. The results show that the GRU model outperforms the other models in terms of accuracy, meaning that the GRU model has better performance than the IRI and LSTM models.

Study by Hong et al. (2024) aims to predict the state-of-charge (SOC) for vehicle battery systems using a hybrid Long Short-Term Memory and Gated Recurrent Unit (LSTM-GRU) neural network. In this study, the prediction results were validated using real-world vehicle data from spring, summer, autumn, and winter seasons, and the proposed method achieved minimum MAPE and MAE of 1.03% and 0.73, respectively, under summer conditions. Meanwhile, the minimum standard deviation of the prediction was 0.06% under experimental conditions. The research process demonstrates that this method achieves high accuracy when applied to large datasets and is expected to be applicable for multi-forward-step SOC prediction of vehicle battery systems in the future.

Study by Bayramova & Kazimov (2025) investigated the effectiveness of deep learning models in assessing software system reliability and the application of Recurrent Neural Network (RNN) algorithms in reliability prediction. A hybrid model combining Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models was proposed to predict software system reliability. Along with historical data collected during testing, several

environmental factors including the software lifecycle and software code complexity were taken as inputs. Combining LSTM and GRU is a promising approach to creating a more robust and flexible RNN. The selection of specific architectures and hyperparameters depends on the specific characteristics of the data and the requirements for the model. The selection of specific combination methods and the evaluation of their effectiveness require comprehensive experimental studies.

Table 2.1 State of Arts

No.	Author	Object	Method				<i>Cost Optimization</i>
			RUL	LSTM	GRU	IoT	
1.	Joseph et al. (2022)	Robot cell	✓	✓		✓	
2.	H. Wu et al. (2020)	Motor bearing	✓	✓		✓	
3.	Krishna et al. (2024)	Lithium battery	✓	✓		✓	
4.	de Pater & Mitici (2023)	Aircraft system	✓	✓		✓	
5.	Yousuf et al. (2022)	Ring Oscillator	✓	✓			
6.	Wang et al. (2023)	Lithium-ion battery	✓	✓		✓	
7.	Yi et al. (2025)	PEMFC	✓	✓		✓	
8.	Cahyani et al. (2023)	Commodity prices		✓			
9.	Ul Haque et al. (2025)	Turbofan engine	✓			✓	✓
10.	Cai et al. (2022)	Subsea Tree	✓			✓	✓
11.	Li et al. (2020)	C-MAPSS	✓		✓	✓	
12.	Purnomo (2022)	Manufacturing engine	✓		✓		✓
13.	Soladoye et al. (2025)	Stroke			✓		
14.	Chen et al. (2025)	Lithium-ion battery	✓		✓	✓	
15.	Zhi-Wen et al. (2024)	Carbon emissions			✓		
16.	Yang et al. (2025)	Waterlogging			✓		
17.	Xia et al. (2024)	Lithium battery	✓		✓	✓	
18.	Tang et al. (2024)	Ionospheric TEC			✓		
19.	Hong et al. (2024)	Vehicle battery	✓	✓	✓	✓	
20.	Bayramova & Kazimov (2025)	Software system		✓	✓	✓	
21.	This study	Batik printer engine	✓	✓	✓	✓	✓

Based on the literature review, RUL prediction has been implemented using various methods, including machine learning approaches such as SVM and SVR, as well as optimization based models focusing on cost efficiency. However, deep learning methods, particularly LSTM and GRU, have proven more effective in capturing temporal dependencies for accurate RUL estimation. Hybrid LSTM–GRU models further improve prediction robustness and accuracy. In addition, integrating IoT enables machine monitoring, and combining predictive models with cost optimization ensures economically efficient maintenance decisions. Therefore, this study employs a hybrid LSTM–GRU model with cost

optimization, leveraging both predictive accuracy and operational cost efficiency in predictive maintenance implementation.

2.2 Theoretical Basis

2.2.1 Maintenance

According to Pintelon & Parodi-Herz (2008), maintenance is an activity that encompasses all technical, administrative, and managerial actions carried out during the life cycle of an asset to maintain or restore it to a certain condition. The purpose of maintenance is to ensure that equipment continues to function optimally, avoid unwanted damage, and support the smooth running of production or operational processes.

Maintenance itself is divided into three type, which are preventive maintenance, corrective maintenance, and predictive maintenance (Carvalho et al., 2019). Preventive maintenance is a maintenance method carried out on a scheduled basis to prevent unexpected damage. Corrective maintenance is maintenance carried out after a machine has experienced damage so that it cannot function properly (Pamungkas et al., 2021). Finally, predictive maintenance is a maintenance method that uses data and analysis to predict when a machine will experience damage, so that maintenance can be carried out just before the machine enters a critical phase.

2.2.2 Predictive Maintenance

One of the modern maintenance strategies that is currently growing rapidly is predictive maintenance. This type of maintenance differs from corrective maintenance, which is carried out after damage has occurred, and preventive maintenance, which is carried out periodically without considering the actual condition of the machine. Predictive maintenance aims to estimate when a machine or component will experience damage, so that maintenance can be carried out just before the damage occurs. The implementation of predictive maintenance in overall industrial management will optimize the availability of process machines and significantly reduce maintenance costs (Mobley, 2002).

The principle of predictive maintenance is that regular monitoring of the actual mechanical condition of machines and the operational efficiency of process systems will ensure maximum intervals between repairs, minimize the number and cost of unscheduled maintenance caused by machine failures, and increase the overall availability of the operating plant. This maintenance uses data-driven strategies and technologies to anticipate when machinery will fail, enabling proactive scheduling of maintenance (Akbar Gozali et al., 2025).

Data such as temperature, vibration, pressure, rotational speed, or even operational data like operating hours are collected using sensors or digital monitoring systems, then analyzed to detect early signs of failure or performance degradation.

One important concept in predictive maintenance is Remaining Useful Life (RUL). RUL is an estimate of the remaining operational time or cycles of a machine before it reaches a failed state (Achouch et al., 2022). RUL is used as the basis for decision-making in determining the most optimal time to perform maintenance. By accurately predicting RUL, companies can avoid premature maintenance that causes cost waste or delayed maintenance that causes downtime or severe damage (Kang et al., 2021).

2.2.3 Machine Learning

Machine learning is a branch of Artificial Intelligence (AI) that designs algorithms to enable computers to learn from past data for model learning, thereby achieving optimal performance in extracting information from a data set (Darmawan Sidik et al., 2022). Essentially, the machine learning process involves three main components: data, model, and algorithm (Alpaydin, 2020). Data serves as the learning material, the model is a mathematical representation of the process being studied, and the algorithm is used to adjust the model to better recognize patterns in the data. The more data used and the higher its quality, the better the model's performance will be.

According to (Kusumaningrum, 2021), machine learning can be divided into three main types, which are supervised learning, unsupervised learning, and reinforcement learning.

1. Supervised Learning is a type of machine learning that uses labeled data as a reference in the training process. The model maps inputs to corresponding outputs based on the provided examples, such as classification and regression.
2. Unsupervised Learning is a type of machine learning used when data does not have labels. The model attempts to discover hidden structures in the data, such as clustering or associations between data points.
3. Reinforcement Learning is a type of machine learning that involves interaction with the environment, so it is not entirely unsupervised, but the data does not always have labels during machine training, so it is not entirely supervised either. The system will continuously interact with the environment in search of desired behavior and receive feedback from the environment.

Some machine learning methods that are often used for RUL prediction include:

1. Support Vector Machine (SVM) is a method that works by finding a regression function that can minimize prediction errors in high-dimensional space. SVR is widely used in RUL estimation due to its ability to handle non-linear data (Widodo & Yang, 2007).
2. Random Forest (RF) is a decision tree based ensemble method that can improve prediction accuracy by combining multiple trees simultaneously. RF is capable of handling highly complex and noisy data, making it a popular choice for predictive maintenance (Zhang et al., 2017).
3. K-Nearest Neighbor (KNN) is a method that predicts the remaining life of a machine based on the similarity of new data patterns to historical data. This method is simple but effective for datasets with clear distributions (Y. Wu et al., 2018).

Although these machine learning methods can provide good results, their weakness is that they are highly dependent on the quality of the extracted features. Therefore, for complex time series sensor data, deep learning methods are often superior because they are capable of automatic feature extraction.

2.2.4 Deep Learning

Deep learning is a further development of machine learning that uses artificial neural network architecture with many hidden layers. With its deep architecture, deep learning is capable of extracting data representations hierarchically and automatically, making it superior to traditional machine learning methods in processing complex and large data sets (Goodfellow et al., 2016).

The main structure of deep learning is an artificial neural network with many layers. Each layer performs a non-linear transformation on the input, allowing the network to gradually build a representation of the data from low to high levels. These layers are connected through weights that are updated using backpropagation algorithms and specific optimizers. The deeper the network architecture, the more complex the patterns that can be captured by the model.

Some commonly used deep learning methods for RUL estimation are as follows:

1. Convolutional Neural Network (CNN). CNN was initially widely used for image processing, but in RUL research, CNN has also proven effective in extracting features from sensor data (e.g., vibration, temperature, or pressure). CNN can recognize local patterns from time series related to machine degradation processes (Zhang et al., 2017).

2. Recurrent Neural Network (RNN). RNN is a deep learning architecture specifically designed to handle sequential or time series data. With a memory mechanism, RNN can remember previous information to predict future conditions. Therefore, RNN is widely used in machine RUL prediction research (Yin et al., 2014).

2.2.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of RNN used to prevent gradient accumulation or long-term dependency issues in time series data prediction. Therefore, LSTM is more suitable for time series data containing long-term patterns (Hochreiter & Schmidhuber, 1997).

The difference between LSTM and other RNN methods lies in the hidden state process, which consists of a forget gate, input gate, cell state, and output gate (Cahyani et al., 2023). Each of these gates functions to regulate the flow of information within the network so that the model can “remember” important information over a long period of time. This structure allows the model to store relevant information and discard unimportant information, resulting in more stable performance when processing long data sequences. To provide a visual representation of the LSTM structure, the basic architecture of this network is shown in Figure 2.1 (Dobilas, 2022).

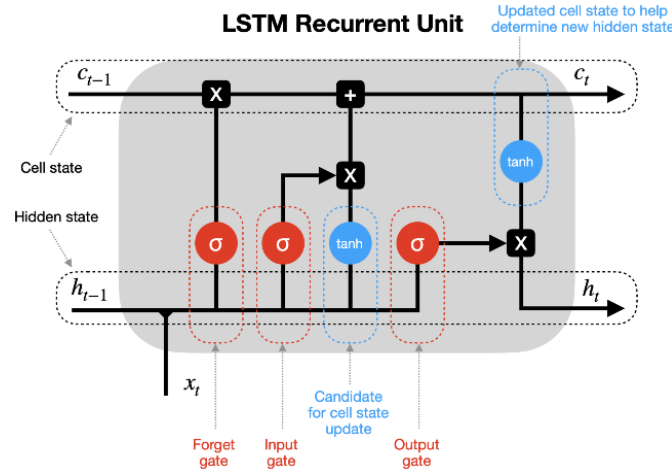


Figure 2.1 LSTM Architecture (Dobilas, 2022b)

The first step in LSTM is to decide what information to discard from the cell state. This decision is made by a sigmoid layer called the forget gate. The forget gate value is obtained using equation (2.1).

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.1)$$

Description:

f_t = forget gate

- σ = sigmoid activation function
 W_f = weight of forget gate
 h_{t-1} = hidden state at the previous timestep
 x_t = input
 b_f = bias value of forget gate

The next step is to decide what new information to store in the cell state. This step consists of two parts. First, a sigmoid layer called the “input gate” decides which values to update. Next, a tanh layer creates a new candidate cell state, which can be added to the state. The functions of both can be seen in equations (2.2) and (2.3).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (2.3)$$

Description:

- i_t = input gate
 W_i = weight of input gate
 b_i = bias value of input gate
 $\tilde{C}_t$ = new cell state candidate
 W_C = weight of cell state
 b_C = bias value of cell state

In the next step, the two results obtained will be combined to make updates to the state. This produces a new cell state value, which is scaled based on how much it was decided to update each state value. The function can be seen in equation (2.4).

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (2.4)$$

Description:

- C_t = cell state
 C_{t-1} = cell state at the previous timestep

After that, enter the output gate stage. Here, the sigmoid activation function will be used again. The output gate value is used to generate a new hidden state value together with the cell state value. Then, the output gate value obtained will be used as the hidden state output value using the tanh function. The output gate is obtained using equation (2.5), while the hidden state is obtained using equation (2.6).

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.5)$$

$$h_t = o_t * \tanh(C_t) \quad (2.6)$$

Description:

o_t = output gate

W_o = weight of output gate

b_o = bias value of output gate

h_t = hidden state

The latest cell state and hidden state results will be used as input in the next LSTM process. This loop will continue until it reaches the end of the sequence.

2.2.6 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is another development of RNN that has similar performance to LSTM. However, GRU has a simpler structure with only two main gates, namely the update gate and reset gate. Despite its simpler structure, GRU is still able to retain important information in the long term with higher efficiency (Khalis Sofi et al., 2021). To provide a visual representation of the GRU structure, the basic architecture of this network is shown in Figure 2.2 (Dobilas, 2022).

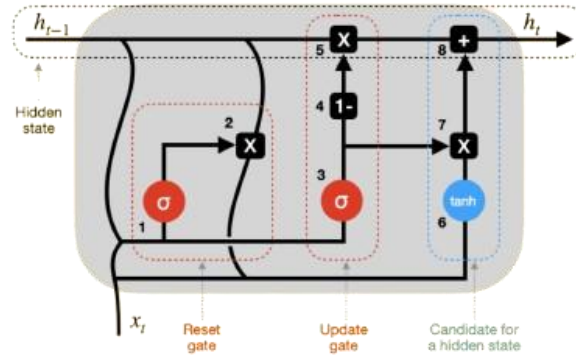


Figure 2.2 GRU Architecture (Dobilas, 2022a)

The first step in GRU is to receive two main inputs, namely the current input and the hidden state from the previous time. This hidden state functions as a memory of information that has been processed at the previous time. Then, GRU calculates the update gate that determines how much information from the previous hidden state will be passed on to the current time. The update gate value is obtained using equation (2.7).

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (2.7)$$

Description:

z_t = update gate

σ = sigmoid activation function

W_z = weight of update gate

h_{t-1} = hidden state at the previous timestep

x_t = input

The next step is to calculate the reset gate, which regulates how much the previous hidden state contributes to the new information to be processed. The reset gate value is obtained using equation (2.8).

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (2.8)$$

Description:

r_t = reset gate

W_r = weight of reset gate

After the two main gates are calculated, the next step is to process the information to form hidden state candidates. The results of these hidden state candidates do not immediately become the output, but are combined with the previous hidden state using the update gate to form a new hidden state. This process allows the network to retain some of the old information while updating it with new, more relevant information. Both functions can be seen in equations (2.9) and (2.10).

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t]) \quad (2.9)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \quad (2.10)$$

Description:

$\tilde{h}_t$ = hidden state candidate

W_h = weight of hidden state candidate

h_t = new hidden state

Just like LSTM, the final result of GRU, namely the latest hidden state, will be used as input in the next GRU process. This process occurs repeatedly (loop) until the entire data sequence has been processed.

2.2.7 Internet of Things

Internet of Things (IoT) is a concept in which physical objects are equipped with sensors, software, and other technologies to connect and exchange data via the internet without direct human involvement (Atzori et al., 2010). In the manufacturing industry, the application of IoT enables companies to monitor machine conditions in real time with a high degree of precision. Sensors on machines can record important parameters such as temperature, vibration, pressure,

current, and voltage, then send them to a central system for continuous analysis (Mohammed et al., 2023).

With data obtained through IoT-based systems, companies can implement more accurate and efficient predictive maintenance strategies. IoT enables the collection of historical and real time data, which forms the basis for developing RUL prediction models. In this study, IoT serves as a tool for collecting machine condition data, which is then used as input in machine learning-based RUL prediction models.

2.2.8 Optimization Model

Optimization model is a mathematical or analytical approach used to find the best solution from a number of alternatives based on specific criteria, such as minimum cost, minimum time, or maximum efficiency (Taha, 2017). In machine maintenance, optimization models are often used to support more rational and measurable decision-making, especially when there are resource constraints or a need to balance various operational factors.

One approach to maintenance optimization is to use an cost optimization model, which calculates the expected cost based on the probability of component failure within a certain period of time. The criteria of this model is to minimize the maintenance cost. This model generally utilizes probability distributions such as the Weibull distribution to model system failure behavior. The Weibull distribution is the most commonly used probability distribution in reliability engineering to represent the service life of components. This distribution is flexible and can describe various failure patterns, depending on the value of its shape parameter (Si et al., 2011).

The objective function of the model usually consists of two main components, which are expected predictive maintenance cost and expected emergency repair maintenance cost. Predictive costs occur when maintenance is performed before failure, while repair costs arise when maintenance is performed after failure. The probability of failure at a given time is used to weight each cost component, resulting in a total cost function that can be minimized. Through this approach, companies can determine the best time to perform maintenance relative to the predicted failure time, thereby balancing the risk of downtime with cost efficiency.

CHAPTER III

RESEARCH METHODOLOGY

3.1 Research Object

This study focuses on PT. X in Pekalongan, a Batik manufacturing company that uses technology in its production process. One of the main tools they use is a batik printing machine, which is used to print patterns on fabric in a mass and efficient way. The object of this research is that batik printing machine, where the machine data they get will be analyzed to figure out the best maintenance time for the batik company, so the production process can run more effectively and efficiently.

3.2 Data Collection

The type of data used in this study is secondary data, which is data that has been collected and made available previously from the company's internal system or other sources. This data was not obtained directly by researchers through experiments or interviews, but rather from existing historical documentation. The secondary data used includes historical information on machine conditions such as temperature, workload, last maintenance, and working hours during the machine 4800 days operating time.

3.3 Research Flow

Below is the research flow chart:

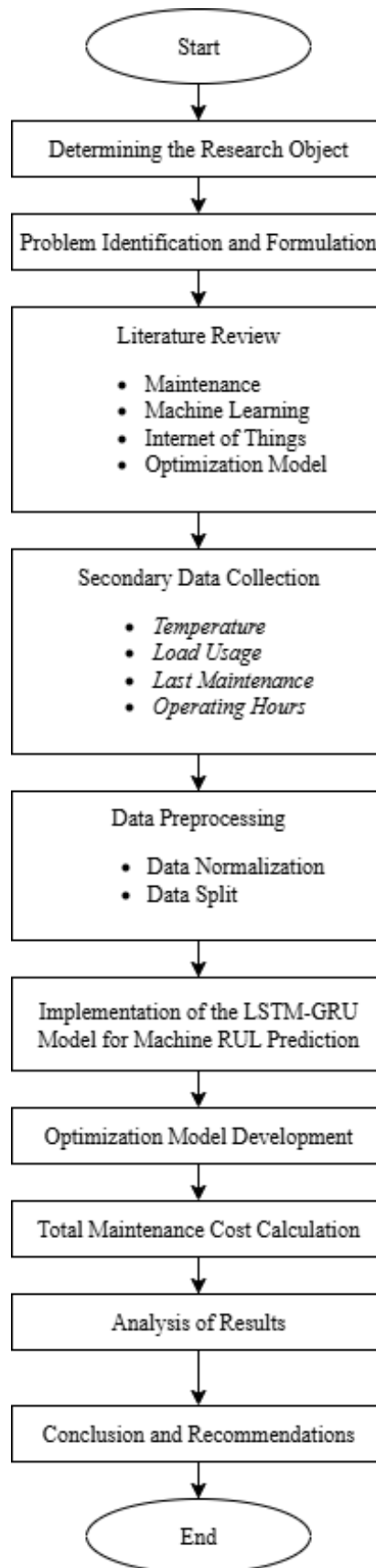


Figure 3.1 Research Flow Chart

1. Start

This is the beginning of the research.

2. Determination of Research Object

The research began with determining the object to be observed. The object of the study is a batik printer machine.

3. Problem Identification and Formulation

Identify machine downtime issues that can cause losses to the company. These issues are formulated into the focus of the research, which is how to predict Remaining Useful Life (RUL) and determine the optimal machine maintenance time to minimize costs.

4. Literature Review

Conduct a review of theories and previous research relevant to the research topic, such as the concept of maintenance, LSTM-GRU machine learning methods, the Internet of Things, and optimization models. This review forms the basis for selecting the methods and approaches used in the research.

5. Secondary Data Collection

Data was obtained from existing historical documentation containing machine condition sensor data. This data includes historical information on machine conditions such as temperature, workload, last maintenance, and operating hours. This data will then be used to build and test the RUL prediction model.

6. Data Pre-processing

The raw data that has been collected is processed so that it can be used for model training. This process includes normalizing feature values and separating data into training and test sets.

7. Implementation of the LSTM-GRU Model for Machine RUL Prediction

A hybrid LSTM-GRU model is used to predict machine RUL based on processed sensor data. The model is trained, validated, and evaluated using specific metrics such as MAE and RMSE to measure prediction accuracy. The trained model is used to predict RUL on test data. The prediction results are then analyzed to determine the model's performance in mapping machine lifespan before failure occurs.

8. Optimization Model Development

After obtaining the RUL prediction results, a cost optimization model is needed to determine the best time to perform maintenance in terms of cost. The maintenance

optimization model is developed using the RUL data, which follows a Weibull distribution. This model integrates the estimation of failure risk, predictive maintenance costs, and risk costs due to failures in the form of emergency breakdown repair costs.

The probability of machine failure before the predictive maintenance time t (measured as hours before RUL equals zero) is expressed as:

$$P_{fail}(t) = \begin{cases} 1, & t \leq 0 \\ 0, & t > \eta \\ 1 - F_{Weibull}(\eta - t; \beta, \eta), & \text{for } 0 < t \leq \eta \end{cases} \quad (2.11)$$

Description:

$P_{fail}(t)$ = probability of failure before time t

$F_{Weibull}$ = Weibull cumulative distribution function

η = shape parameter of the Weibull distribution

t = predictive maintenance time

β = shape parameter of the Weibull distribution

This probability quantifies the likelihood that a failure occurs before maintenance at time t , thus enabling risk based cost consideration.

Next step is finding the predictive maintenance cost per event. The cost for a single predictive maintenance event at time t consists of a fixed cost plus the cost associated with the risk of failure due to downtime, which is modeled as an emergency breakdown repair cost:

$$C_{event}(t) = C_{Pdm} + P_{fail}(t) \times C_{EBR} \quad (2.12)$$

Description:

$C_{event}(t)$ = cost per maintenance at time t

C_{Pdm} = cost per predictive maintenance event

C_{EBR} = cost of emergency breakdown repair if failure occurs

After that, calculate the expected interval between maintenance. This represents the average operational time before the next maintenance event once maintenance is performed at time. Assuming the interval between maintenances follows the residual life distribution of the Weibull, the expected interval for maintenance after performing it at time t is:

$$E[Interval_t] = \frac{\int_t^\eta (x - t) f_{Weibull}(x; \beta, \eta) dx}{S_{Weibull}(t; \beta, \eta)} \quad (2.13)$$

Description:

$E[Interval_t]$ = expected interval between maintenance after time t

$f_{Weibull}$ = Weibull probability density function

$S_{Weibull}$ = $1 - F_{Weibull}$ is the probability the machine operates beyond time t

The final step is multiplying the maintenance frequency by the cost per event to generate the total predictive maintenance cost. Below is the formulation:

$$f(t) = \frac{T}{E[Interval_t]} \quad (2.14)$$

$$C_{total}(t) = f(t) \times C_{event}(t) \quad (2.15)$$

Description:

$f(t)$ = frequency of predictive maintenance during T

T = total operational period

9. Total Cost Maintenance Calculation

The total maintenance cost for one machine period is calculated based on the model that has been developed. The calculation is performed with the help of Python programming.

10. Analysis of Results

Conduct an analysis of the accuracy of the prediction model and the cost efficiency of the maintenance strategy. This analysis will serve as the basis for concluding the effectiveness of using the LSTM-GRU model in machine maintenance decision-making.

11. Conclusion and Recommendations

Conclusions are drawn based on the analysis of the data that has been collected and processed. Recommendations are then formulated as suggestions that the company can use to improve its maintenance strategy.

12. End

The research has been completed.

CHAPTER IV

DATA COLLECTING AND PROCESSING

4.1 Data Collection

The data used in this study is secondary data obtained from an IoT-based machine monitoring system on batik printing machines. The data includes machine condition parameters, such as temperature and other parameters relevant to machine health. In addition, there is also data on Remaining Useful Life (RUL) and machine maintenance costs. The following is the machine data used in the study that have been collected during the machine 4800 days operating time:

Table 4.1 Batik Printer Machine Data Parameters

No.	Temperature (C)	Load Usage (Units)	Last Maintenance (Days)	Operating Hours (Hours)	RUL (Hours)
1	69,09	601	41	296	48
2	62,89	580	42	360	54
3	76,31	525	41	344	43
4	94,61	407	40	304	24
5	68,73	552	38	280	52
6	64,20	355	42	328	60
7	52,65	448	39	312	69
8	73,22	487	44	320	46
9	91,60	498	42	336	27
10	54,08	462	35	304	69
11	86,30	593	35	328	29
12	59,36	459	35	320	65
13	65,32	526	36	352	59
14	83,56	401	41	296	34
15	87,07	533	40	312	31
16	51,34	549	38	288	69
17	72,68	423	42	360	49
18	54,52	543	39	280	71
19	70,87	358	45	296	46
20	95,14	427	45	296	23
21	68,49	619	44	288	48
22	74,12	505	40	280	43
23	55,36	412	42	320	63
24	77,57	496	44	360	38
25	64,90	542	41	360	57
26	91,01	388	36	288	30
27	71,23	474	38	304	45
28	62,59	455	43	352	56
29	75,17	431	37	328	47

No.	Temperature (C)	Load Usage (Units)	Last Maintenance (Days)	Operating Hours (Hours)	RUL (Hours)
30	85,64	567	44	320	34
31	77,89	611	35	352	38
32	94,18	593	38	288	25
33	52,16	441	40	312	72
34	68,32	356	36	352	53
35	87,42	433	43	288	33
36	86,34	328	44	288	35
37	94,98	462	41	360	20
38	74,73	512	41	352	39
39	51,61	510	45	320	70
40	67,92	584	37	288	49
41	60,63	574	39	320	62
42	63,90	584	42	280	59
43	54,86	364	44	328	69
44	61,64	425	36	312	56
45	64,79	575	44	352	54
46	84,76	442	42	312	37
47	52,24	441	37	344	65
48	88,12	457	35	312	31
49	91,28	389	39	360	25
50	67,76	351	38	304	50
51	71,18	467	35	360	46
52	68,47	392	41	344	48
53	83,22	491	35	352	38
54	60,64	615	41	352	61
55	73,19	623	41	344	45
56	66,21	472	43	280	53
57	90,16	449	40	320	29
58	68,12	471	39	344	54
59	67,73	391	43	296	51
60	54,45	620	35	360	61
61	94,85	375	43	280	26
62	52,70	592	41	296	69
63	87,61	550	42	288	28
64	62,29	568	39	336	60
65	54,07	337	37	296	69
66	93,60	622	36	288	25
67	67,60	561	36	328	54
68	74,00	588	41	336	41
69	61,82	358	42	336	57
70	58,34	581	44	312	56
71	91,94	596	35	320	24
72	88,17	470	38	304	30
73	73,66	404	35	312	46

No.	Temperature (C)	Load Usage (Units)	Last Maintenance (Days)	Operating Hours (Hours)	RUL (Hours)
74	92,74	433	42	352	22
75	75,62	410	37	280	49
76	74,33	372	43	296	44
77	60,98	481	41	320	58
78	55,62	491	44	360	60
79	80,35	526	38	320	38
80	92,24	563	37	328	23
81	83,31	573	39	296	32
82	68,07	499	37	320	56
83	84,00	358	41	336	36
84	53,31	505	45	344	72
85	78,74	406	37	320	44
86	76,39	397	37	296	48
87	59,60	464	40	312	59
88	75,44	339	43	304	48
89	68,20	574	39	280	55
90	91,06	463	40	344	27
91	74,04	364	41	296	47
92	89,24	518	43	288	31
93	68,96	526	39	352	52
94	83,84	575	44	320	32
95	66,28	430	41	344	50
96	85,38	624	39	328	34
97	63,90	362	42	344	62
98	71,25	404	41	320	47
99	77,12	347	40	328	41
100	55,46	395	35	320	62

The next data is maintenance cost data. These costs consist of several activities, with each type of maintenance having different activities. The following are the details of machine maintenance costs:

Table 4.2 General Maintenance Cost Data

General Maintenance Activity	Cost (Rp)
Lubrication	50.000
Cleaning	75.000
Component Inspection	80.000
Minor Repair	25.000
Small Parts Replacement	100.000
Medium Parts Replacement	150.000
Major Overhaul	200.000
Emergency Breakdown Repair	200.000

This table shows the costs of various general machine maintenance activities. The activities recorded include lubrication, cleaning, component inspection, minor repairs, replacement of small to medium components, major overhauls, and emergency repairs due to sudden damage. This data is used as a basic reference for calculating the total cost of a particular maintenance strategy.

Table 4.3 Preventive Maintenance Cost Data

Preventive Maintenance Activity	Cost (Rp)
Lubrication	50.000
Cleaning	75.000
Component Inspection	80.000
Minor Repair	25.000
Total	230.000

The table above shows the costs for preventive maintenance, which are activities carried out on a scheduled basis to prevent damage. Preventive maintenance costs are relatively lower, totaling Rp230,000, consisting of routine activities such as lubrication, cleaning, component inspection, and minor repairs.

Table 4.4 Predictive Maintenance Cost Data

Predictive Maintenance Activity	Cost (Rp)
Lubrication	50.000
Cleaning	75.000
Component Inspection	80.000
Small Parts Replacement	100.000
Total	305.000

This table adds the replacement of small components to routine activities, increasing the total cost to Rp305,000. This strategy is based on the actual condition of the machine, so that replacement activities are carried out only when necessary, rather than simply according to schedule.

Table 4.5 Corrective Maintenance Cost Data

Corrective Maintenance Activity	Cost (Rp)
Component Inspection	80.000
Minor Repair	25.000
Medium Parts Replacement	150.000
Major Overhaul	200.000
Emergency Breakdown Repair	200.000
Total	655.000

Corrective maintenance incurs much higher costs, namely Rp655,000. This is because corrective maintenance is carried out after damage has occurred, so that the activities involved

include component inspection, minor repairs, moderate component replacement, major overhauls, and emergency repairs. These high costs reflect the consequences of only performing maintenance after the machine has failed to operate.

4.2 Data Processing

4.2.1 Remaining Useful Life (RUL) Prediction

The steps for performing RUL prediction are as follows, with the support of Google Colab serving as the Python tool:

1. Import Library

In the data processing for Remaining Useful Life (RUL) prediction, the first step is to import the necessary libraries. These libraries include data manipulation libraries such as pandas and numpy, deep learning development libraries such as TensorFlow and Keras, and visualization libraries such as matplotlib. Additionally, scikit-learn is used for preprocessing and model evaluation, and keras_tuner is employed for hyperparameter optimization. The use of these various libraries enables complex processes such as the creation of a hybrid neural network architecture (combining LSTM and GRU), data normalization, visualization of prediction results, and automatic hyperparameter tuning.

```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Dense, LSTM, GRU, Input, Dropout
from keras_tuner.tuners import RandomSearch
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import networkx as nx
import matplotlib.pyplot as plt
```

Figure 4.1 Import Library

2. Load Data

The next step is to load the dataset from the Excel file. This dataset contains important parameters measured from industrial machinery, such as temperature, load, duration since

last maintenance, and operating hours. This data is crucial as input for the machine learning model, because each parameter can have a significant relationship with the RUL value.

```
from google.colab import files
uploaded = files.upload()

# Load data
import pandas as pd
df = pd.read_excel("RUL_Mesin_Batik.xlsx")
```

Choose Files RUL_Mesin_Batik.xlsx

- **RUL_Mesin_Batik.xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet)

Saving RUL_Mesin_Batik.xlsx to RUL_Mesin_Batik.xlsx

Figure 4.2 Load Data

3. Preprocessing Data

Before the data is used to train the model, important preprocessing in the form of normalization and data division is carried out. In this stage, MinMaxScaler is used to scale feature and target values to a range of 0 to 1. The features used include temperature, load usage, last maintenance, and operating hours, while the target is Remaining Useful Life (RUL). The data is then divided into training data and test data with a ratio of 80:20. The data split of 80% for training and 20% for testing was chosen because it is generally considered to provide an optimal balance between the model's ability to learn patterns well and its ability to be evaluated on new data (Sivakumar et al., 2024).

1. Initialize a MinMaxScaler object
2. Define the list of feature columns to be used as input variables
3. Define the target variable (RUL)
4. Extract feature data from the dataframe based on the feature list
5. Extract target data from the dataframe based on the target variable
6. Normalize the feature data using the scaler
7. Normalize the target data using the scaler
(reshape target into a 2D array before scaling)
8. Split the dataset into training and testing sets
with 80% for training and 20% for testing
Use random state = 42 for reproducibility
9. Reshape the training feature data into 3D format
(number of samples, 1, number of features) for LSTM/GRU input
10. Reshape the testing feature data into 3D format
(number of samples, 1, number of features) for LSTM/GRU input

Figure 4.3 Pseudocode of Preprocessing Data

```

# Preprocessing
scaler = MinMaxScaler()
features = ["Temperature (C)", "Load Usage (Units)", "Last Maintenance (Days)", "Operating Hours (Hours)"]
target = "RUL (Hours)"

X = df[features]
y = df[target]

X_scaled = scaler.fit_transform(X)
y_scaled = scaler.fit_transform(y.values.reshape(-1, 1))

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size=0.2, random_state=42)

# Reshape for LSTM/GRU
X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

```

Figure 4.4 Preprocessing Data

4. Definition of Hybrid LSTM-GRU Model Architecture

This section defines the model to explore various combinations of hyperparameters. The model being built is a hybrid model consisting of sequential LSTM and GRU layers, equipped with a dropout layer to prevent overfitting, and a Dense output layer to generate RUL prediction values. The hyperparameters being optimized include the number of LSTM and GRU units, the dropout rate, and the Adam optimizer learning rate.

```

1. Define a function build_model with parameter hp (hyperparameters)
2. Create a Sequential model
3. Add an LSTM layer:
   - Number of units determined by hp.Int in the range 32 to 128, step size 32
   - return_sequences set to True
   - Input shape based on (time steps, number of features)
4. Add a GRU layer:
   - Number of units determined by hp.Int in the range 32 to 128, step size 32
5. Add a Dropout layer:
   - Dropout rate determined by hp.Float in the range 0.2 to 0.5, step size 0.1
6. Add a Dense layer with:
   - 1 output neuron
   - linear activation function
7. Compile the model using:
   - Adam optimizer with learning rate chosen from hp.Choice values [0.01, 0.001, 0.0001]
   - Mean Squared Error (MSE) as the loss function
   - Mean Absolute Error (MAE) as an evaluation metric
8. Return the compiled model

```

Figure 4.5 Pseudocode of Define Model

```

# Define model building function for hyperparameter tuning
def build_model(hp):
    model = Sequential()
    model.add(LSTM(hp.Int('lstm_units', min_value=32, max_value=128, step=32), return_sequences=True,
                    input_shape=(X_train.shape[1], X_train.shape[2])))
    model.add(GRU(hp.Int('gru_units', min_value=32, max_value=128, step=32)))
    model.add(Dropout(hp.Float('dropout', min_value=0.2, max_value=0.5, step=0.1)))
    model.add(Dense(1, activation='linear'))
    model.compile(optimizer=tf.keras.optimizers.Adam(hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4])),
                  loss='mse',
                  metrics=['mae'])
    return model

```

Figure 4.6 Define Model

5. Hyperparameter Optimization

In this section, the process of searching for the best hyperparameter combination is performed using the Random Search method. The tuner will run several model configurations based on the function that has been defined previously. This process aims to find the model configuration that produces optimal performance in predicting RUL.

1. Initialize a RandomSearch tuner with:
 - Model building function: `build_model`
 - Objective to minimize: validation loss (`val_loss`)
 - Maximum trials: 10
 - Executions per trial: 2
 - Save results in directory: `'hyperparameter_tuning'`
 - Project name: `'hybrid_lstm_gru'`
2. Run the hyperparameter search with:
 - Training data: `X_train, y_train`
 - Number of epochs: 50
 - Validation split: 20% of training data
 - Verbose output enabled
3. Retrieve the best hyperparameters found from the search

Figure 4.7 Pseudocode of Hyperparameter Optimization

```
# Hyperparameter optimization
tuner = RandomSearch(
    build_model,
    objective='val_loss',
    max_trials=10,
    executions_per_trial=2,
    directory='hyperparameter_tuning',
    project_name='hybrid_lstm_gru'
)

tuner.search(X_train, y_train, epochs=50, validation_split=0.2, verbose=1)
best_hps = tuner.get_best_hyperparameters(num_trials=1)[0]

Trial 10 Complete [00h 00m 29s]
val_loss: 0.0017838220810517669

Best val_loss So Far: 0.0016032429412007332
Total elapsed time: 00h 04m 43s
```

Figure 4.8 Hyperparameter Optimization

6. Build the Best Model

The best model from the tuning process was rebuilt and trained for 100 epochs to obtain optimal performance. This model is expected to be able to capture temporal patterns in the data to accurately predict RUL.

```
# Build the best model
best_model = tuner.hypermodel.build(best_hps)
history = best_model.fit(X_train, y_train, epochs=100, validation_split=0.2, verbose=1)
```

```
Epoch 72/100
2/2 ————— 0s 77ms/step - loss: 0.0041 - mae: 0.0510 - val_loss: 0.0026 - val_mae: 0.0446
Epoch 73/100
2/2 ————— 0s 70ms/step - loss: 0.0052 - mae: 0.0557 - val_loss: 0.0024 - val_mae: 0.0427
Epoch 74/100
2/2 ————— 0s 75ms/step - loss: 0.0042 - mae: 0.0512 - val_loss: 0.0020 - val_mae: 0.0399
Epoch 75/100
2/2 ————— 0s 80ms/step - loss: 0.0036 - mae: 0.0472 - val_loss: 0.0019 - val_mae: 0.0382
Epoch 76/100
2/2 ————— 0s 75ms/step - loss: 0.0033 - mae: 0.0482 - val_loss: 0.0018 - val_mae: 0.0383
Epoch 77/100
2/2 ————— 0s 74ms/step - loss: 0.0044 - mae: 0.0523 - val_loss: 0.0020 - val_mae: 0.0399
```

Figure 4.9 Build the Best Model

7. Evaluate Model

The model was evaluated by calculating the loss value and mean absolute error (MAE) in the test data. MAE is the main measure used in this study to measure how far the RUL prediction results deviate from the actual values on average.

```
# Evaluate model
train_loss, train_mae = best_model.evaluate(X_train, y_train, verbose=0)
print(f"Train Loss: {train_loss}, Train MAE: {train_mae}")
test_loss, test_mae = best_model.evaluate(X_test, y_test, verbose=0)
print(f"Test Loss: {test_loss}, Test MAE: {test_mae}")
```

Train Loss: 0.003069145604968071, Train MAE: 0.046513017266988754
Test Loss: 0.003453754587098956, Test MAE: 0.0478239543735981

Figure 4.10 Evaluate Model

8. RUL Prediction on Test Data and Return to Original Scale

Predictions are made on test data using the best model resulting from tuning. The predicted RUL values and actual targets are then returned to their original scale (in time units).

```
# Plot actual vs. predicted RUL
y_pred = best_model.predict(X_test)
y_test_unscaled = scaler.inverse_transform(y_test)
y_pred_unscaled = scaler.inverse_transform(y_pred)
```

1/1 ————— 0s 374ms/step

Figure 4.11 Plot Actual vs Predicted RUL

9. Calculation of RMSE and R-squared Values

To evaluate the model's performance in more detail, the Root Mean Square Error (RMSE) and coefficient of determination (R^2) values were calculated. RMSE describes the magnitude of prediction errors in hours, while R^2 indicates how much of the variation in actual values can be explained by the model's predictions. An R^2 value close to 1 indicates that the model has excellent predictive performance.

```
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, r2_score

# Compute RMSE
rmse = np.sqrt(mean_squared_error(y_test_unscaled, y_pred_unscaled))
print(f"RMSE: {rmse:.4f}")

# Compute R-squared
r2 = r2_score(y_test_unscaled, y_pred_unscaled)
print(f"R-squared: {r2:.4f}")
```

RMSE: 3.1624
R-squared: 0.9499

Figure 4.12 Compute RMSE and R-squared

10. Actual RUL vs. Predicted RUL Visualization

Line graphs are used to compare actual and predicted RUL values for each test data sample. The closer the two curves are to each other, the better the model's accuracy in mapping the actual condition of the machine.

```
plt.figure(figsize=(10, 6))
plt.plot(y_test_unscaled, label='Actual RUL', marker='o')
plt.plot(y_pred_unscaled, label='Predicted RUL', marker='x')
plt.xlabel('Test Samples')
plt.ylabel('RUL (Unscaled)')
plt.legend()
plt.title('Actual vs. Predicted RUL')
plt.show()
```

Figure 4.13 RUL Visualization

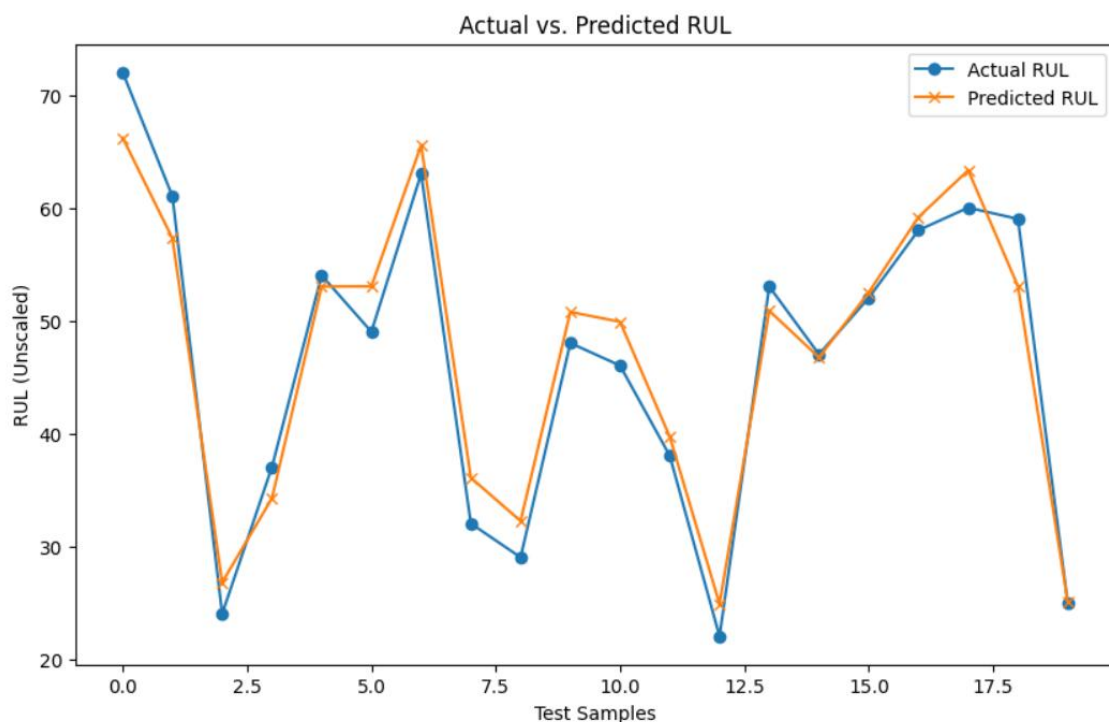


Figure 4.14 Actual vs Predicted RUL Graph

11. Visualization of Artificial Neural Network Architecture

To improve understanding of the internal structure of the model, the architecture of the artificial neural network was visualized. This visualization depicts the number of neurons in the input layer, hidden layers (LSTM and GRU), and output layer, as well as the connections between neurons. The colors of each layer are differentiated to indicate the function of each layer, which are blue for input, green for hidden layers, and red for output.

```

# Plot neural network architecture
def plot_neural_network_custom(input_neurons, hidden_layers, output_neurons):
    """
    Plots a neural network with labeled neurons and customized layer colors.
    """
    # Initialize graph
    G = nx.DiGraph()
    pos = {}
    layer_gap = 2
    neuron_gap = 1

    # Input layer
    for i, name in enumerate(input_neurons):
        G.add_node(f"Input {i+1}", layer="Input", label=name)
        pos[f"Input {i+1}"] = (0, -i * neuron_gap)

    # Hidden layers
    prev_layer = [f"Input {i+1}" for i in range(len(input_neurons))]
    layer_x = layer_gap
    for l_idx, num_neurons in enumerate(hidden_layers):
        curr_layer = [f"Hidden{l_idx+1} Neuron {i+1}" for i in range(num_neurons)]
        for j, neuron in enumerate(curr_layer):
            G.add_node(neuron, layer=f"Hidden {l_idx+1}", label=f"H{l_idx+1} N{j+1}")
            pos[neuron] = (layer_x, -j * neuron_gap)
        for p_node in prev_layer:
            for c_node in curr_layer:
                G.add_edge(p_node, c_node)
        prev_layer = curr_layer
        layer_x += layer_gap

    # Output layer
    output_layer = [f"Output {i+1}" for i in range(len(output_neurons))]
    for i, name in enumerate(output_layer):
        G.add_node(name, layer="Output", label=output_neurons[i])
        pos[name] = (layer_x, -i * neuron_gap)
    for p_node in prev_layer:
        for o_node in output_layer:
            G.add_edge(p_node, o_node)

    # Draw graph
    plt.figure(figsize=(12, 8))
    nx.draw(
        G, pos,
        with_labels=True,
        labels=nx.get_node_attributes(G, 'label'),
        node_size=5000,
        node_color=[
            'skyblue' if G.nodes[node]['layer'] == "Input" else
            'lightgreen' if "Hidden" in G.nodes[node]['layer'] else
            'salmon'
        ],
        for node in G.nodes
    ),
    font_size=10,
    font_color='black',
    edge_color='gray',
    arrowstyle='->',
    arrowsize=15,
    connectionstyle='arc3,rad=0.1'
    )
    plt.title("Neural Network Architecture", fontsize=16)
    plt.show()

input_features = features
hidden_layers_neurons = [best_hps.get('lstm_units'), best_hps.get('gru_units')]
output_labels = [target]

plot_neural_network_custom(input_features, hidden_layers_neurons, output_labels)

```

Figure 4.15 Plot Neural Network Architecture

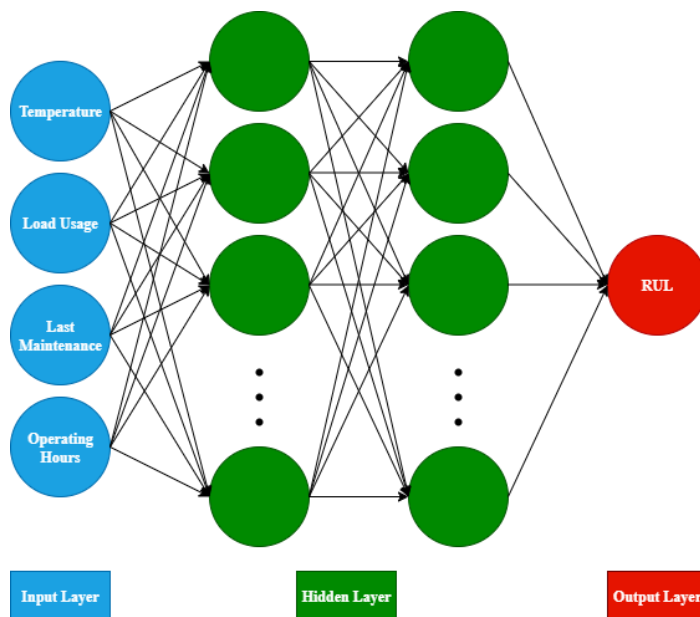


Figure 4.16 Neural Network Architecture

12. Training Data Table

This section displays the training data that has been processed in DataFrame format. The last column shows the target value (RUL) used during model training.

```
# Display training data
train_data = pd.DataFrame(X_train.reshape(X_train.shape[0], -1), columns=[f"Feature_{i+1}" for i
                                                                    in range(X_train.shape[2])])

train_data['Target (RUL)'] = y_train
print("Training Data:")
print(train_data)
```

```
Training Data:
   Feature_1  Feature_2  Feature_3  Feature_4  Target (RUL)
0    0.339498    0.486486         0.8         0.0         0.634615
1    0.384932    0.831081         0.4         0.0         0.673077
2    0.454110    0.493243         0.3         0.3         0.480769
3    0.080365    0.121622         0.9         0.6         0.942308
4    0.159817    0.854730         0.9         0.4         0.692308
..         ...         ...         ...         ...         ...
75   0.993379    0.158784         0.8         0.0         0.115385
76   0.840868    0.479730         0.3         0.3         0.192308
77   0.815753    0.692568         0.5         0.4         0.211538
78   0.402283    0.668919         0.4         0.9         0.615385
79   0.391096    0.216216         0.6         0.8         0.538462

[80 rows x 5 columns]
```

Figure 4.17 Training Data

13. Test Data Table and Prediction Results

The test data is presented in a table containing feature values and two additional columns showing the actual RUL values and the model's prediction results.

```
# Display testing data and predicted RUL
test_data = pd.DataFrame(X_test.reshape(X_test.shape[0], -1), columns=[f"Feature_{i+1}" for i
                                                                    in range(X_test.shape[2])])
test_data['Actual RUL'] = y_test_unscaled
test_data['Predicted RUL'] = y_pred_unscaled
print("\nTesting Data with Predicted RUL:")
print(test_data)
```

Testing Data with Predicted RUL:

	Feature_1	Feature_2	Feature_3	Feature_4	Actual RUL	Predicted RUL
0	0.044977	0.597973	1.0	0.8	72.0	67.113182
1	0.212329	0.969595	0.6	0.9	61.0	57.991867
2	0.926941	0.905405	0.0	0.5	24.0	26.228159
3	0.763014	0.385135	0.7	0.4	37.0	33.774109
4	0.307078	0.834459	0.9	0.9	54.0	53.555920
5	0.378539	0.864865	0.2	0.1	49.0	53.395840
6	0.091781	0.283784	0.7	0.5	63.0	66.000183
7	0.729909	0.827703	0.4	0.2	32.0	35.905731
8	0.798174	0.895270	0.0	0.6	29.0	31.847973
9	0.405251	0.922297	0.6	0.2	48.0	51.173485
10	0.445890	0.101351	1.0	0.2	46.0	49.863228
11	0.606164	0.956081	0.0	0.9	38.0	39.583477
12	0.945205	0.354730	0.7	0.9	22.0	23.927486
13	0.387671	0.094595	0.1	0.9	53.0	51.055977
14	0.518265	0.121622	0.6	0.2	47.0	46.603565
15	0.397032	0.756757	0.3	0.0	52.0	52.797333
16	0.220091	0.516892	0.6	0.5	58.0	59.558395
17	0.097717	0.550676	0.9	1.0	60.0	64.112434
18	0.319178	0.668919	0.1	0.9	59.0	53.370678
19	0.978082	0.895270	0.3	0.1	25.0	24.643023

Figure 4.18 Testing Data

4.2.2 Maintenance Cost Calculation with Optimization Model

The steps for performing optimization model are as follows, with the support of Google Colab serving as the Python tool:

1. Import Library

The first step in building a maintenance cost optimization model is to import the necessary libraries. The numpy and pandas libraries are used for data manipulation, scipy.stats for fitting the Weibull distribution, and matplotlib.pyplot for visualizing the analysis results.

```
import numpy as np
import pandas as pd
from scipy.stats import weibull_min
import matplotlib.pyplot as plt
from scipy import stats
```

Figure 4.19 Import Library

2. Determination of Weibull Distribution Parameters

This step is to model the distribution of the Remaining Useful Life (RUL) of the machine using the Weibull distribution. At this stage, fitting is performed using the Maximum

Likelihood Estimation (MLE) method. The shape (β) and scale (λ) values are estimated based on actual RUL data from test data.

```
# Choose RUL data for Weibull fitting
rul_data = test_data['Actual RUL'].values

# Ensure that the data is positive and valid (Weibull is only for data > 0)
rul_data = rul_data[rul_data > 0]

# Weibull distribution fitting with Maximum Likelihood Estimation (fix loc=0 assumption)
shape, loc, scale = weibull_min.fit(rul_data, floc=0)

print(f"Parameter Weibull hasil fitting:")
print(f"  Shape (k): {shape:.4f}")
print(f"  Scale (lambda): {scale:.4f}")
```

Parameter Weibull hasil fitting:
 Shape (k): 3.8314
 Scale (lambda): 51.5411

Figure 4.20 Determining Weibull Parameters

3. Visualization of Weibull Distribution Fitting

To evaluate the suitability of the Weibull distribution for RUL data, a Weibull Probability Plot is used. This plot compares the theoretical quantiles of the Weibull distribution with the actual quantiles of the data.

```
# Visualization of Weibull Probability Plot for fitting evaluation
plt.figure(figsize=(8,6))
stats.probplot(rul_data, dist=weibull_min, sparams=(shape, loc, scale), plot=plt)
plt.title('Weibull Probability Plot dari RUL Data')
plt.xlabel('Teoritis Weibull Quantiles')
plt.ylabel('Data RUL Observasi')
plt.grid(True)
plt.show()
```

Figure 4.21 Weibull Visualization

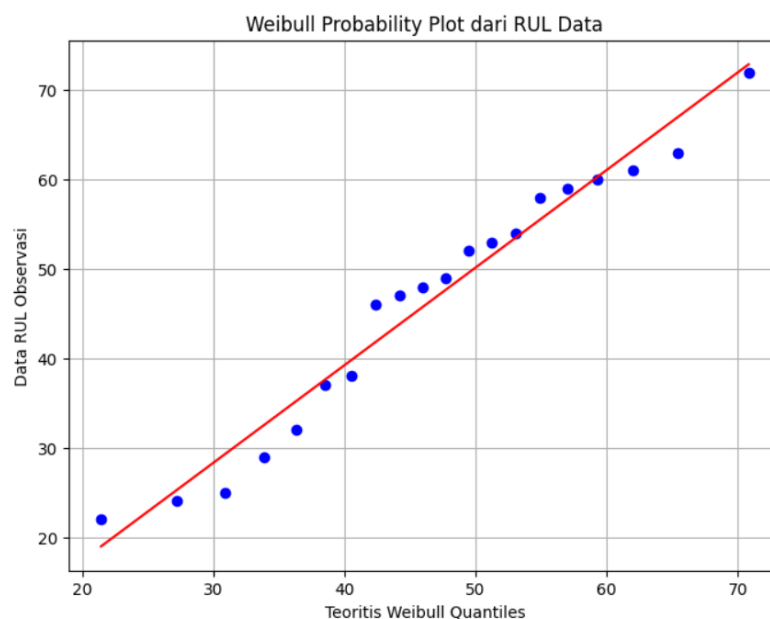


Figure 4.22 Weibull Visualization

4. Setting Cost Parameters and Probability Functions

After the RUL distribution is modeled, the next step is to define the cost parameters and probability functions. Costs consist of the fixed costs of predictive maintenance and additional costs in the event of a sudden failure (emergency breakdown repair). In addition, define the probability function of failure before maintenance time, the probability density function and survival function of the Weibull distribution, and the expected remaining life function of the machine.

```
# Weibull Parameter
shape = 3.8314
scale = 51.5411 # dalam jam

# Cost Parameter
cost_PdM_fixed = 305000      # Predictive maintenance cost per event (fixed)
emergency_breakdown_repair = 200000 # emergency breakdown repair cost

# Total machine operating period
T_operational = 4800

# Probability function of failure before maintenance time t (hours before RUL=0)
def P_fail(t, scale, shape):
    if t <= 0:
        return 1.0
    if t > scale:
        return 0.0
    return 1 - weibull_min.cdf(scale - t, shape, scale=scale)

# Weibull PDF function
def weibull_pdf(x):
    return weibull_min.pdf(x, shape, scale=scale)

# Weibull survival function (probability still functions more than t)
def weibull_sf(x):
    return weibull_min.sf(x, shape, scale=scale)

# Interval expectation function between maintenance  $E[RUL - t \mid RUL > t]$ 
def expected_interval(t):
    if t >= scale:
        return 0.0
    survival_t = weibull_sf(t)
    if survival_t == 0:
        return 0.0
    integrand = lambda x: (x - t) * weibull_pdf(x)
    integral_result, _ = quad(integrand, t, scale)
    return integral_result / survival_t
```

Figure 4.23 Setting Parameters and Function

5. Calculation of Maintenance Costs per Event and Total Costs

After calculating the probability of failure and expected interval, the next step is to calculate maintenance costs. The total cost per event is obtained by adding the cost of preventive maintenance and the possible cost of breakdown if failure occurs before

maintenance time. The total cost over the operational cycle depends on the expected number of hours the machine will last. The more frequently maintenance is performed, the higher the frequency but each event is cheaper. On the other hand, if maintenance is too infrequent, the risk of failure increases, leading to high emergency costs.

```
# Predictive maintenance cost per event function (fixed + emergency breakdown repair)
def cost_per_event(t):
    return cost_PdM_fixed + P_fail(t, scale, shape) * emergency_breakdown_repair

# Total predictive maintenance cost function during the operating period with frequency
def total_cost_with_frequency(t):
    cost_event = cost_per_event(t)
    interval = expected_interval(t)
    if interval == 0:
        return float('inf'), cost_event, 0 # hindari pembagian nol
    freq = T_operational / interval
    total_cost = cost_event * freq
    return total_cost, cost_event, freq

# Maintenance time range
t_values = np.arange(0, int(scale), 1)

# Calculate the cost per event and total cost by frequency
results = []
for t in t_values:
    total_c, cost_evt, freq = total_cost_with_frequency(t)
    results.append({
        "Jam sebelum RUL=0": t,
        "Biaya per Event (Rp)": cost_evt,
        "Frekuensi Maintenance (kali)": freq,
        "Total Biaya (Rp)": total_c
    })

df = pd.DataFrame(results)
```

Figure 4.24 Cost Calculation

6. Identify the Optimal Maintenance Time

To determine the optimal time, a search is conducted for the t value that results in the lowest total cost. This value is used as a recommendation for the maintenance time in order to minimize operational costs.

```
# Find the optimal time based on minimum total cost
optimal_idx = df['Total Biaya (Rp)'].idxmin()
optimal = df.loc[optimal_idx]

# Print optimal results
print(f"Waktu optimal maintenance (t*): {optimal['Jam sebelum RUL=0']} jam sebelum RUL=0")
print(f"Biaya predictive maintenance per event: Rp {optimal['Biaya per Event (Rp)']:,.0f}")
print(f"Frekuensi maintenance selama {period_days} hari: {optimal['Frekuensi Maintenance (kali)']:,.2f} kali")
print(f"Total biaya predictive maintenance minimal: Rp {optimal['Total Biaya (Rp)']:,.0f}")

print("\nTabel biaya predictive maintenance per waktu maintenance:")
print(df.head(15))
```

Figure 4.25 Finding Optimal Time

Waktu optimal maintenance (t^*): 1.0 jam sebelum RUL=0
 Biaya predictive maintenance per event: Rp 384,094
 Frekuensi maintenance: 92.90 kali
 Total biaya predictive maintenance minimal: Rp 35,683,892

Tabel biaya predictive maintenance per waktu maintenance:

Jam sebelum RUL=0	Biaya per Event (Rp)	Frekuensi Maintenance (kali)	\
0	505000.000000	90.488602	
1	384093.951558	92.904071	
2	389691.522328	95.451704	
3	395341.438233	98.141908	
4	401016.290166	100.985832	
5	406688.804864	103.995490	
6	412332.210174	107.183878	
7	417920.575738	110.565113	
8	423429.123069	114.154582	
9	428834.500236	117.969120	
10	434115.017708	122.027209	
11	439250.843167	126.349214	
12	444224.154316	130.957651	
13	449019.249871	135.877512	
14	453622.619886	141.136635	

	Total Biaya (Rp)
0	4.569674e+07
1	3.568389e+07
2	3.719672e+07
3	3.879956e+07
4	4.049696e+07
5	4.229380e+07
6	4.419537e+07
7	4.620744e+07
8	4.833637e+07
9	5.058923e+07
10	5.297384e+07
11	5.549900e+07
12	5.817455e+07
13	6.101162e+07
14	6.402277e+07

Figure 4.26 Optimal Time and Cost Result

7. Visualization of Total Cost vs Maintenance Time

Once all cost values have been calculated for each t value, the graph can be used to identify the minimum cost point. The minimum point on this curve is the optimal time to perform maintenance in order to minimize total costs during the operating period.

```
# Graph Visualization

fig, axs = plt.subplots(2, 1, figsize=(12, 10), sharex=True)

# Graph 1: Predictive cost per event (varies)
axs[0].plot(df["Jam sebelum RUL=0"], df["Biaya per Event (Rp)"])
axs[0].axvline(optimal["Jam sebelum RUL=0"], color='red', linestyle='--',
               label=f'Waktu Optimal: {optimal["Jam sebelum RUL=0"]} jam')
axs[0].axhline(optimal["Biaya per Event (Rp)"], color='green', linestyle='--',
               label=f'Biaya Optimal Rp {int(optimal["Biaya per Event (Rp)"]):,}')
axs[0].set_title("Biaya Predictive Maintenance per Event (Fixed + Lost Production)")
axs[0].set_ylabel("Biaya (Rp)")
axs[0].legend()
axs[0].grid(True)

# Graph 2: Total predictive maintenance costs by frequency
axs[1].plot(df["Jam sebelum RUL=0"], df["Total Biaya (Rp)"])
axs[1].axvline(optimal["Jam sebelum RUL=0"], color='red', linestyle='--',
               label=f'Waktu Optimal: {optimal["Jam sebelum RUL=0"]} jam')
axs[1].axhline(optimal["Total Biaya (Rp)"], color='green', linestyle='--',
               label=f'Total Biaya Optimal Rp {int(optimal["Total Biaya (Rp)"]):,}')
axs[1].set_title("Total Biaya Predictive Maintenance (Biaya per Event x Frekuensi)")
axs[1].set_xlabel("Jam sebelum RUL=0")
axs[1].set_ylabel("Total Biaya (Rp)")
axs[1].legend()
axs[1].grid(True)

plt.tight_layout()
plt.show()
```

Figure 4.27 Cost Visualization

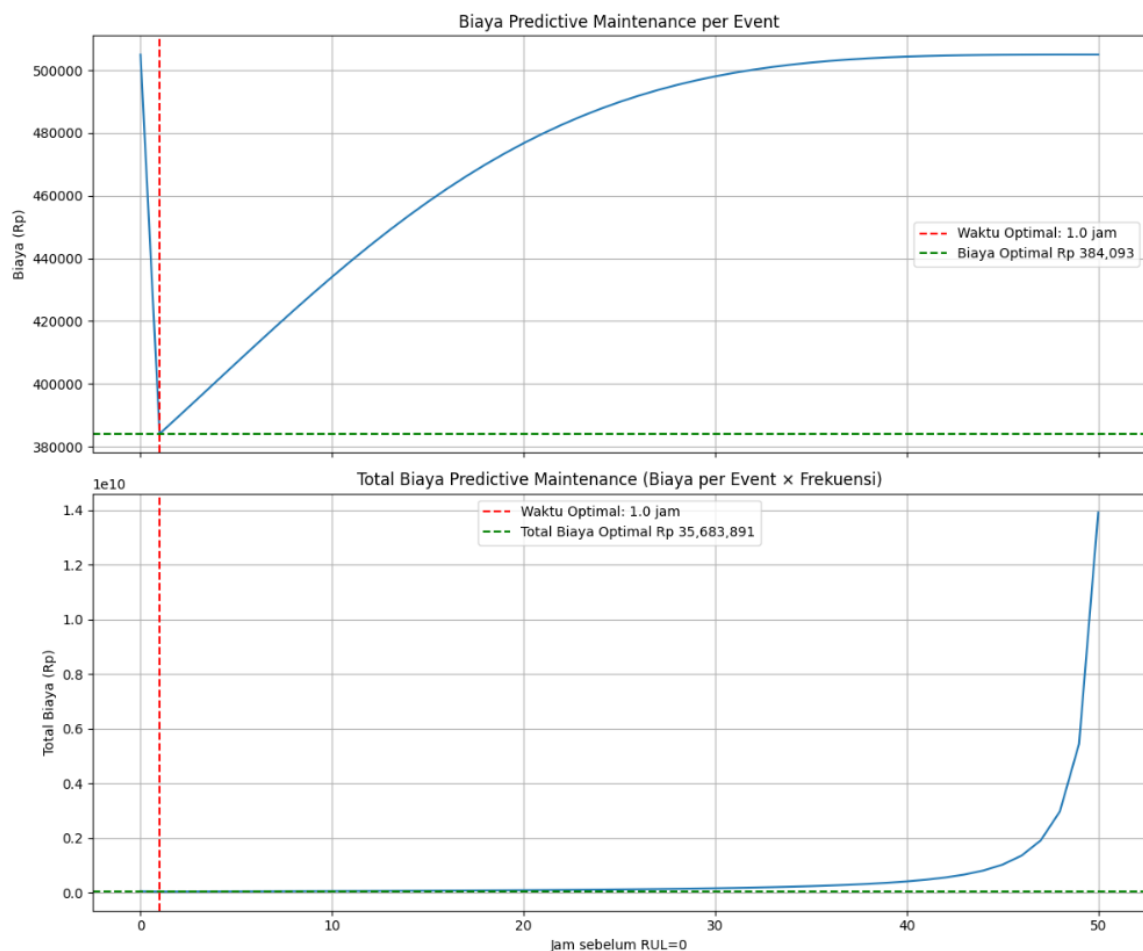


Figure 4.28 Cost vs Time Graph

CHAPTER V

DISCUSSION

5.1 Remaining Useful Life (RUL) Prediction Using the LSTM-GRU Hybrid Model

Predicting Remaining Useful Life (RUL) plays a crucial role in predictive maintenance systems in the industrial sector, particularly in the context of operational sustainability of machinery and maintenance cost control. In this study, a hybrid model combining Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) was used to predict RUL values for machinery based on time series data. The combination of two neural network architectures in a single model is intended to leverage the strengths of each, where LSTM excels at maintaining long-term dependencies, and GRU is lighter and more computationally efficient yet still reliable in recognizing temporal patterns.

The hybrid LSTM-GRU model was trained using historical sensor data from the machine, with a preprocessing process that included MinMaxScaler normalization, sequence formation to capture temporal context, and separation of training and test data. In addition, this model was optimized using the hyperparameter method. The hyperparameter tuning process was carried out using the Random Search method with the help of the Keras Tuner library. A total of 10 experiments were conducted with 2 executions per experiment to improve the reliability of the results. The best model was selected based on the smallest validation loss value achieved, which was 0.0016. This model was then used as the final model in the training and evaluation process for the test data.

```
Trial 10 Complete [00h 00m 29s]
val_loss: 0.0017838220810517669

Best val_loss So Far: 0.0016032429412007332
Total elapsed time: 00h 04m 43s
```

Figure 5.1 Hyperparameter Tuning Result

After training, the model was tested against test data to predict the actual RUL value. The prediction results were then evaluated using regression accuracy metrics, which are Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R-squared (R^2). Based on the evaluation results, the train MSE value was 0.00306, the train MAE value was 0.0465, the test MSE value was 0.00345, the train MAE value was 0.0478, the RMSE value was 3.1624, and the R^2 value was 0.9499. These values indicate that the model has good predictive capabilities, with relatively small errors and high accuracy. An R^2 value close to 1

indicates that the model can explain nearly all of the variability in the target data, so it can be concluded that the hybrid LSTM-GRU model used is sufficiently reliable and suitable for application in a predictive maintenance system.

Train Loss: 0.003069145604968071, Train MAE: 0.046513017266988754
 Test Loss: 0.003453754587098956, Test MAE: 0.0478239543735981
 RMSE: 3.1624
 R-squared: 0.9499

Figure 5.2 Evaluation Result

The model does not exhibit overfitting, as indicated by the small difference between training MAE (0.0465) and test MAE (0.0478), with only about a 2.7% gap. Similarly, the test MSE shows only a slight increase compared to the training MSE, which reflects a good generalization ability. According to Goodfellow et al. (2016), overfitting occurs when the training error is much smaller than the test error, while no universal threshold exists because the degree of acceptable error difference depends heavily on the dataset characteristics.

The visualization of the prediction results also shows good accuracy between the actual and predicted values. The graph comparing actual RUL with prediction results shows that the model is able to follow the trend of remaining useful life decline well, although there are some deviations at certain points. These deviations are likely caused by sensor signal fluctuations that do not fully reflect the real time condition of the machine, or the presence of noise in the input data that has not been completely filtered.

The proposed LSTM-GRU model in this study achieved an MAE of 0.0478, which means that the average deviation between the predicted and actual values is relatively small. When compared to the study by Yuan et al. (2024) which applied Support Vector Regression (SVR) to wind turbine components with an MAE result of 0.075, the model proposed in this study provides better accuracy. Meanwhile, the study by Xia et al. (2024) which performed a Bidirectional Gated Recurrent Unit (BiGRU) to predict the State of Health (SOH) and RUL of lithium batteries, achieved MAE values as low as 0.3% and 0.6% for SOH prediction. Although the proposed model in this study does not reach those exceptionally low error rates, its performance is still competitive. This comparison highlights that while advanced architectures such as BiGRU can deliver extremely high accuracy in specific domains, the hybrid LSTM-GRU approach remains a robust and effective method for predictive maintenance applications in industrial settings.

Overall, the application of the hybrid LSTM-GRU model in this study provides very good RUL prediction results. This model not only accurately captures the patterns of machine lifespan degradation but also has the potential to be integrated into real time industrial maintenance systems. With these RUL prediction results, more efficient maintenance strategies can be designed by utilizing them to determine the most optimal maintenance timing based cost considerations.

5.2 Maintenance Time Optimization Using a Cost Optimization Model

After obtaining the Remaining Useful Life (RUL) prediction results using the LSTM-GRU hybrid model, the next step is to determine the optimal time for predictive maintenance to minimize the total maintenance costs during the machine's operational period. For this, a Weibull distribution based approach is used as a probabilistic failure model, as well as a cost estimation model that takes into account the fixed costs per maintenance and potential additional costs due to machine failure.

First, Weibull distribution fitting was performed on the actual RUL data in the test data, with parameter estimation using the Maximum Likelihood Estimation (MLE) method. Based on this fitting, the shape parameter (k) was obtained as 3.8314 and the scale parameter (λ) as 51.5411 hours. These values indicate that the RUL data follows a Weibull distribution with an increasing failure rate, which is common in industrial equipment that experiences wear and tear over time.

The optimization model considers two main cost components, which are the fixed cost of predictive maintenance amounting to Rp305,000, and the emergency breakdown repair amounting to Rp200,000, calculated probabilistically based on the probability of failure before maintenance is performed. Additionally, the expected interval between two maintenance activities is calculated based on the actual RUL distribution. By considering the required maintenance frequency during the machine's operational period, an estimate of the total maintenance cost is obtained for various possible maintenance times.

In this study, it was found that the optimal time for predictive maintenance is 1 hour before RUL=0, with a predictive maintenance cost per event of Rp384,094 and a maintenance frequency during the operational period of 92.90 times. The minimum total predictive maintenance cost obtained from this configuration is Rp35,683,892. The graph result clarifies the relationship between maintenance time and cost, if maintenance is performed too far from the end of the service life, the frequency increases, leading to high total costs. Conversely, if

performed too close to $RUL=0$, the likelihood of failure increases, driving up the cost per event. Therefore, the optimal time is achieved at the balance point between maintenance frequency and failure risk.

After obtaining the cost optimization results, the next step is to compare them with the maintenance strategy previously used by the company. In comparison, preventive maintenance carried out routinely every 30 days during the 4800 operating time results in a frequency of around 160 times with a total cost of Rp36,800,000. Corrective maintenance shows a much higher cost, namely Rp65,500,000, with a frequency of 100 times because maintenance is only carried out after damage occurs. Meanwhile, predictive maintenance with the proposed cost optimization approach results in a frequency of 92.2 times with a cost of Rp35,683,892.

Table 5. 1 Maintenance Strategy Comparison

Strategy	Frequency	Total Cost
Preventive Maintenance	160	Rp36,800,000
Corrective Maintenance	100	Rp65,500,000
Predictive Maintenance	92.2	Rp35,683,892

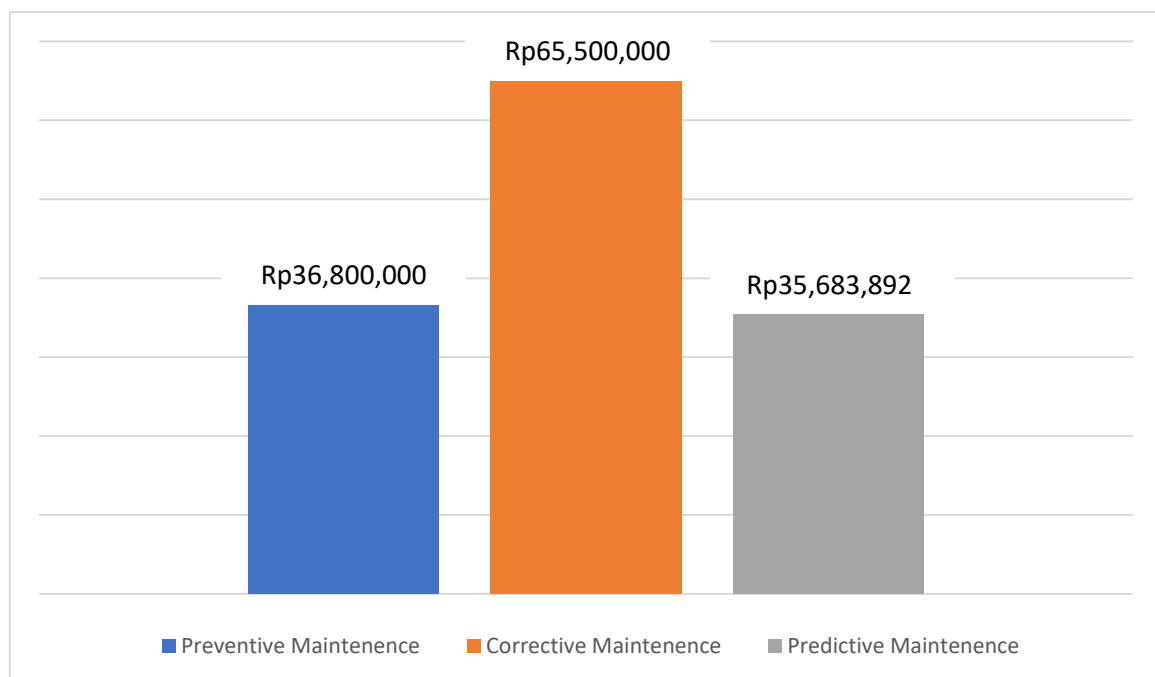


Figure 5. 3 Maintenance Cost Comparison

From the table and visualization above, it can be seen that corrective maintenance incurs the highest costs compared to other strategies. Preventive maintenance has lower costs than corrective maintenance, but is still slightly higher than predictive maintenance due to its more frequent maintenance schedule. However, predictive maintenance remains more efficient

because it can reduce the frequency of maintenance while maintaining machine reliability at a relatively lower cost. These results confirm that the implementation of predictive maintenance is not only relevant from a technical standpoint, but also provides significant financial benefits for companies.

Thus, this approach allows for more accurate maintenance timing based on predictions of actual machine conditions and failure distribution characteristics. The optimal timing obtained can be used as a basis for maintenance decisions in the implementation of predictive maintenance in real production systems, with the hope of reducing maintenance costs and minimizing the risk of downtime due to sudden failures.

5.3 Study Limitations

This study has several limitations that need to be considered as a basis for future development. The first limitation is the assumptions regarding maintenance cost parameters used in the optimization model. In this study, all cost parameters are assumed to be constant throughout the analysis period. In reality, some maintenance cost components, particularly maintenance fluids, are obtained through imports. Therefore, costs are greatly affected by fluctuations in the dollar exchange rate. Exchange rate changes can cause significant price increases or decreases, so the total cost calculations used to determine the optimal maintenance timing may not fully reflect actual field conditions. A more realistic approach should consider these price dynamics to ensure that optimization results are more accurate and relevant to market conditions.

Second, this study uses historical time series data and does not utilize real time data from sensors directly on the machines. This limitation may affect the accuracy of RUL predictions and maintenance timing recommendations, as the model relies solely on past data, which may not fully reflect the actual conditions at the time of decision-making. With the integration of real time data in the future, it is expected that the model will provide more accurate predictions and respond more effectively to changes in machine conditions in real time.

Lastly, this study analysis is focused only on the cost aspects of different maintenance strategies, without considering reliability aspects. While cost efficiency is an important factor in maintenance decision making, it does not provide a complete picture of machine performance in the long term. Reliability factors such as how consistently a machine can operate without failure, or how maintenance actions affect the durability of components, are not incorporated into the model. As a result, the conclusions of this study primarily emphasize financial

efficiency, and may overlook potential risks associated with machine downtime or reduced operational stability. For future research, it would be valuable to integrate both cost and reliability perspectives to provide a more holistic understanding of maintenance optimization.

CHAPTER VI

CLOSING

6.1 Conclusion

Based on the results of the research that has been conducted, the following conclusions can be drawn:

1. The hybrid LSTM-GRU model successfully provided Remaining Useful Life (RUL) predictions with a high degree of accuracy. Based on the evaluation results on the test data, the Mean Squared Error (MSE) value was 0.00369, the Mean Absolute Error (MAE) was 0.0524, the Root Mean Squared Error (RMSE) was 3.1624, and the coefficient of determination (R-squared) was 0.9499. This indicates that the model is able to effectively capture temporal patterns in machine data and provide very good RUL predictions.
2. Based on the RUL prediction results obtained, the timing of predictive maintenance was optimized using an cost optimization model based on the Weibull distribution. The optimization results showed that the optimal time to perform maintenance was 1 hours before RUL=0, with an estimated minimum total maintenance cost of Rp35,683,892 during the machine's operating period. This model balances the costs of preventive maintenance and the risk of failure, resulting in cost efficient maintenance decisions.
3. The comparison between the proposed predictive maintenance model and the previously applied approaches shows that the predictive strategy provides a more cost-efficient solution. The corrective maintenance strategy resulted in the highest total cost, reaching approximately Rp65,500,000 with 100 occurrences. Preventive maintenance was able to reduce the total cost to around Rp36,800,000 with 160 scheduled interventions. In contrast, the proposed predictive maintenance model achieved the lowest cost of about Rp35,683,892 with only 92.2 interventions. These results demonstrate that predictive maintenance not only minimizes costs compared to corrective and preventive strategies but also reduces the frequency of interventions, making it the most effective and efficient approach for maintaining batik printing machines.

6.2 Sugestion

Based on the results of the research that has been conducted, there are several suggestions that can be considered for future research and future improvements:

1. The optimization model used in this study is based on fixed cost parameters. In the future, research can develop models that consider more dynamic cost factors that takes into account changes in the prices of components and maintenance materials, particularly maintenance fluids that are imported and affected by fluctuations in foreign exchange rates. This approach would provide more realistic and accurate cost estimates for determining the optimal timing of maintenance.
2. This study uses time series data that is already available. For further development, it is recommended to collect real time data from direct sensors so that prediction and optimization results are closer to actual conditions. This will enable the model to adjust predictions and maintenance recommendations based on the actual condition of the machine in the field.
3. Future research is suggested to not only focus on cost aspects but also incorporate reliability considerations. By integrating both perspectives, the analysis will provide a more comprehensive evaluation of maintenance strategies, ensuring not only cost efficiency but also sustained machine performance and operational stability.
4. For companies, the results of this research can serve as an initial reference when considering the implementation of predictive maintenance approaches. By leveraging RUL predictions and optimal maintenance timing calculations, companies can gain insights into the potential cost efficiency of future machine maintenance.

BIBLIOGRAPHY

- Achouch, M., Dimitrova, M., Ziane, K., Sattarpanah Karganroudi, S., Dhouib, R., Ibrahim, H., & Adda, M. (2022). On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. *Applied Sciences (Switzerland)*, 12(16). <https://doi.org/10.3390/app12168081>
- Akbar Gozali, A., Faris Ruriawan, M., Alamsyah, A., Purwanto, Y., Romadhony, A., Pandu Wijaya, F., Nugroho, F., Nala Husna, D., Kridanto, A., Fakhrudin, A., Itqon, M., & Widiyanesti, S. (2025). Smart train control and monitoring system with predictive maintenance and secure communications features. *Transportation Research Interdisciplinary Perspectives*, 31. <https://doi.org/10.1016/j.trip.2025.101409>
- Alpaydm, E. (2020). *Introduction to Machine Learning* (4th ed.). The MIT Press.
- Arisandhi, I. N. (2023). Perawatan Preventif Pada Mesin Press Batang Tebu Demi Meningkatkan Kehandalan Mesin Dengan Menggunakan Metode Reliability Centered Maintenance (Rcm) PT. Madu Baru Ps/Pg Madukismo Yogyakarta. *SEMINAR NASIONAL PENELITIAN MAHASISWA TEKNIK (SINLIMATEK)*, 195–200.
- Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15), 2787–2805. <https://doi.org/10.1016/j.comnet.2010.05.010>
- Bayramova, T. A., & Kazimov, T. H. (2025). Application of LSTM-GRU combined model to calculate the reliability of software systems. *Procedia Computer Science*, 252, 127–135. <https://doi.org/10.1016/j.procs.2024.12.014>
- Cahyani, J., Mujahidin, S., & Fiqar, T. P. (2023). Implementasi Metode Long Short Term Memory (LSTM) untuk Memprediksi Harga Bahan Pokok Nasional. *Jurnal Sistem Dan Teknologi Informasi (JustIN)*, 11(2), 346. <https://doi.org/10.26418/justin.v11i2.57395>
- Cai, B., Wang, Y., Zhang, Y., Liu, Y., Ge, W., Li, R., Liu, Y., & Liu, G. (2022). Condition-based maintenance method for multi-component system based on RUL prediction: Subsea tree system as a case study. *Computers and Industrial Engineering*, 173. <https://doi.org/10.1016/j.cie.2022.108650>
- Carvalho, T. P., Soares, F. A. A. M. N., Vita, R., Francisco, R. da P., Basto, J. P., & Alcalá, S. G. S. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. *Computers and Industrial Engineering*, 137. <https://doi.org/10.1016/j.cie.2019.106024>
- Chen, Z., Peng, Y., Shen, J., Zhang, Q., Liu, Y., Zhang, Y., Xia, X., & Liu, Y. (2025). State of health estimation for lithium-ion batteries based on fragmented charging data and improved gated recurrent unit neural network. *Journal of Energy Storage*, 115, 115952. <https://doi.org/10.1016/j.est.2025.115952>
- Darmawan Sidik, A., Ansawarman, A., Kunci, K., Kendaraan Bermotor, J., Regresi, M., & Jalan, F. (2022). Prediksi Jumlah Kendaraan Bermotor Menggunakan Machine Learning.

- Formosa Journal of Multidisciplinary Research (FJMR)*, 1(3), 559–568.
<https://doi.org/10.55927>
- de Pater, I., & Mitici, M. (2023). Developing health indicators and RUL prognostics for systems with few failure instances and varying operating conditions using a LSTM autoencoder. *Engineering Applications of Artificial Intelligence*, 117.
<https://doi.org/10.1016/j.engappai.2022.105582>
- Dobilas, S. (2022a). *GRU Recurrent Neural Networks – A Smart Way to Predict Sequences in Python*. <https://towardsdatascience.com/gru-recurrent-neural-networks-a-smart-way-to-predict-sequences-in-python-80864e4fe9f6/>
- Dobilas, S. (2022b). *LSTM Recurrent Neural Networks – How to Teach a Network to Remember the Past*. <https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e/>
- European Committee for Standardization. (2017). *Maintenance - Maintenance Terminology (EN 13306:2017)*.
- Ewin, G., & Oye, E. (2025). *Transitioning from Corrective to Preventive Maintenance Strategies: Enhancing Equipment Reliability and Efficiency*.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- Hamasha, M. M., Bani-Irshid, A. H., Al Mashaqbeh, S., Shwaheen, G., Al Qadri, L., Shbool, M., Muathen, D., Ababneh, M., Harfoush, S., Albedoor, Q., & Al-Bashir, A. (2023). Strategical selection of maintenance type under different conditions. *Scientific Reports*, 13(1). <https://doi.org/10.1038/s41598-023-42751-5>
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Hong, J., Liang, F., Yang, H., Zhang, C., Zhang, X., Zhang, H., Wang, W., Li, K., & Yang, J. (2024). Multi- forward-step state of charge prediction for real-world electric vehicles battery systems using a novel LSTM-GRU hybrid neural network. *ETransportation*, 20.
<https://doi.org/10.1016/j.etrans.2024.100322>
- Joseph, D., Gallege, T., Bekar, E. T., Dudas, C., & Skoogh, A. (2022). A Predictive Maintenance Application for A Robot Cell using LSTM Model. *IFAC-PapersOnLine*, 55(19), 115–120. <https://doi.org/10.1016/j.ifacol.2022.09.193>
- Joseph Nnaemeka Chukwunweike, Andrew Nii Anang, Adewale Abayomi Adeniran, & Jude Dike. (2024). Enhancing manufacturing efficiency and quality through automation and deep learning: addressing redundancy, defects, vibration analysis, and material strength optimization. *World Journal of Advanced Research and Reviews*, 23(3), 1272–1295.
<https://doi.org/10.30574/wjarr.2024.23.3.2800>
- Kang, Z., Catal, C., & Tekinerdogan, B. (2021). Remaining Useful Life (RUL) prediction of equipment in production lines using artificial neural networks. *Sensors (Switzerland)*, 21(3), 1–20. <https://doi.org/10.3390/s21030932>

- Khairunisa, N. K., & Hendikawati, P. (2024). Long Short-Term Memory and Gated Recurrent Unit Modeling for Stock Price Forecasting. *Jurnal Matematika, Statistika Dan Komputasi*, 21(1), 321–333. <https://doi.org/10.20956/j.v21i1.35930>
- Khalis Sofi, Aswan Supriyadi Sunge, Sasmitoh Rahmad Riady, & Antika Zahrotul Kamalia. (2021). Perbandingan Algoritma Linear Regression, LSTM, dan GRU dalam Memprediksi Harga Saham dengan Model Time Series. *SEMINASTIKA*, 3(1), 39–46. <https://doi.org/10.47002/seminastika.v3i1.275>
- Krishna, G., Singh, R., Gehlot, A., & Akram, S. V. (2024). An IoT-based predictive model for improved battery management system using advanced LSTM model. *Journal of Energy Storage*, 101. <https://doi.org/10.1016/j.est.2024.113694>
- Kusumaningrum, D. (2021). *Penggunaan Machine Learning Untuk Predictive Maintenance (Studi Kasus: PT.X)*. Institut Teknologi Sepuluh Nopember.
- Le, L.-H. (2024). Time series analysis and applications in data analysis, forecasting and prediction. *HPU2 Journal of Science: Natural Sciences and Technology*, 3(1), 20–29. <https://doi.org/10.56764/hpu2.jos.2024.3.1.20-29>
- Li, L., Zhao, Z., Zhao, X., & Lin, K. Y. (2020). Gated Recurrent Unit Networks for Remaining Useful Life Prediction. *IFAC-PapersOnLine*, 53, 10498–10504. <https://doi.org/10.1016/j.ifacol.2020.12.2795>
- Maher, Y., & Danouj, B. (2020). Survey on deep learning applied to predictive maintenance. *International Journal of Electrical and Computer Engineering*, 10(6), 5592–5598. <https://doi.org/10.11591/ijece.v10i6.pp5592-5598>
- Mobley, R. K. (2002). *An Introduction to Predictive Maintenance* (2nd ed.). Butterworth-Heinemann.
- Mohammed, N. A., Abdulateef, O. F., & Hamad, A. H. (2023). An IoT and Machine Learning-Based Predictive Maintenance System for Electrical Motors. *Journal Europeen Des Systemes Automatises*, 56(4), 651–656. <https://doi.org/10.18280/jesa.560414>
- Ningrum, A. A., Syarif, I., Gunawan, A. I., Satriyanto, E., & Muchtar, R. (2021). Algoritma Deep Learning-LSTM untuk Memprediksi Umur Transformator. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 8(3), 539–548. <https://doi.org/10.25126/jtiik.2021834587>
- Pamungkas, I., Irawan, H. T., & Pandria, T. M. A. (2021). IMPLEMENTASI PREVENTIVE MAINTENANCE UNTUK MENINGKATKAN KEANDALAN PADA KOMPONEN KRITIS BOILER DI PEMBANGKIT LISTRIK TENAGA UAP. *VOCATECH: Vocational Education and Technology Journal*, 2(2), 73–78. <https://doi.org/10.38038/vocatech.v2i2.53>
- Pintelon, L., & Parodi-Herz, A. (2008). Maintenance: An Evolutionary Perspective. In *Springer Series in Reliability Engineering* (Vol. 8, pp. 21–48). Springer London. https://doi.org/10.1007/978-1-84800-011-7_2

- Purnomo, M. R. A. (2022). Incorporating deep learning data analytics techniques in the optimisation of capacitated planned maintenance. *Jurnal Sistem Dan Manajemen Industri*, 6(2), 167–175. <https://doi.org/10.30656/jsmi.v6i2.5076>
- Si, X. S., Wang, W., Hu, C. H., & Zhou, D. H. (2011). Remaining useful life estimation - A review on the statistical data driven approaches. *European Journal of Operational Research*, 213(1), 1–14. <https://doi.org/10.1016/j.ejor.2010.11.018>
- Sitinjak, F. R., & Silalahi, F. T. R. (2023). Analisis Strategi Pemeliharaan Preventive Maintenance Excavator Menggunakan Pendekatan Analytical Hierarchy Process (AHP) dan Analisis Sensitivitas. *Journal of Integrated System*, 6(2), 226–242. <https://doi.org/10.28932/jis.v6i2.7633>
- Sivakumar, M., Parthasarathy, S., & Padmapriya, T. (2024). Trade-off between training and testing ratio in machine learning for medical image processing. *PeerJ Computer Science*, 10. <https://doi.org/10.7717/PEERJ-CS.2245>
- Soladoye, A. A., Olawade, D. B., Adeyanju, I. A., Akpa, O. M., Aderinto, N., & Owolabi, M. O. (2025). Optimizing stroke prediction using gated recurrent unit and feature selection in Sub-Saharan Africa. *Clinical Neurology and Neurosurgery*, 249. <https://doi.org/10.1016/j.clineuro.2025.108761>
- Taha, H. A. (2017). *Operations Research An Introduction* (10th ed.). Pearson .
- Tang, J., Liu, C., Yang, D., & Ding, M. (2024). Prediction of ionospheric TEC using a GRU mechanism method. *Advances in Space Research*, 74(1), 260–270. <https://doi.org/10.1016/j.asr.2024.03.050>
- Ul Haque, F., Haque Raza, A., & Mosharraf Hossain, Dr. Md. (2025). Optimizing Maintenance Cost in A Multi Component Environment. *International Journal of Scientific and Management Research*, 08(01), 99–120. <https://doi.org/10.37502/IJSMR.2025.8105>
- Wang, Z., Liu, N., Chen, C., & Guo, Y. (2023). Adaptive self-attention LSTM for RUL prediction of lithium-ion batteries. *Information Sciences*, 635, 398–413. <https://doi.org/10.1016/j.ins.2023.01.100>
- Widodo, A., & Yang, B.-S. (2007). Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical Systems and Signal Processing*, 21(6).
- Wu, H., Huang, A., & Sutherland, J. W. (2020). Avoiding Environmental Consequences of Equipment Failure via an LSTM-Based Model for Predictive Maintenance. *Procedia Manufacturing*, 43, 666–673. <https://doi.org/10.1016/j.promfg.2020.02.131>
- Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, 275.
- Xia, F., Yu, Y., & Chen, J. (2024). SOH and RUL prediction of lithium batteries based on fusions of RLOESS filtered electrochemical and thermal features by bidirectional gated recurrent unit network. *Journal of Energy Storage*, 102. <https://doi.org/10.1016/j.est.2024.114134>

- Yang, S.-Y., Jhong, Y.-D., Jhong, B.-C., Fan-Chiang, L., & Tsai, M.-C. (2025). A two-stage multi-step-ahead Gated-Recurrent-Unit-based method with the use of sewer level data for improving real-time forecasting of inundation depth. *Journal of Hydrology*, 133230. <https://doi.org/10.1016/j.jhydrol.2025.133230>
- Yi, F., Shu, X., Zhou, J., Zhang, J., Feng, C., Gong, H., Zhang, C., & Yu, W. (2025). Remaining useful life prediction of PEMFC based on matrix long short-term memory. *International Journal of Hydrogen Energy*, 111, 228–237. <https://doi.org/10.1016/j.ijhydene.2025.02.302>
- Yin, S., Ding, S. X., Xie, X., & Luo, H. (2014). A Review on Basic Data-Driven Approaches for Industrial Process Monitoring. *IEEE Transactions on Industrial Electronics*, 61(11).
- Yousuf, S., Khan, S. A., & Khursheed, S. (2022). Remaining useful life (RUL) regression using Long–Short Term Memory (LSTM) networks. *Microelectronics Reliability*, 139. <https://doi.org/10.1016/j.microrel.2022.114772>
- Yuan, K., Ma, C., Guo, G., & Wang, P. (2024). Slope Failure of Shilu Metal Mine Transition from Open-Pit to Underground Mining under Excavation Disturbance. *Applied Sciences (Switzerland)*, 14(3). <https://doi.org/10.3390/app14031055>
- Zhang, C., Lim, P., Qin, A. K., & Tan, K. C. (2017). Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics. *IEEE Trans Neural Netw Learn Syst*, 28(10).
- Zhi-Wen, H., Qian-Bin, D., & Xiao-Long, C. (2024). Forecasting carbon emissions in Chinese coastal cities based on a gated recurrent unit model. *Energy Reports*, 12, 5747–5756. <https://doi.org/10.1016/j.egyr.2024.11.048>
- Zonta, T., da Costa, C. A., da Rosa Righi, R., de Lima, M. J., da Trindade, E. S., & Li, G. P. (2020). Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers and Industrial Engineering*, 150. <https://doi.org/10.1016/j.cie.2020.106889>