CHALMERS
UNIVERSITY OF TECHNOLOGY

Machine Learning Modeling of Thermal System Components For BEVs

Master's thesis in Systems, Control and Mechatronics Programme

Abinaya Rajanarayanan
Sri Sai Saranya Sridhar

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Machine Learning Modeling of Thermal System Components For BEVs

A Data-Driven Approach for Health Monitoring and Remaining Useful Life Estimation

Abinaya Rajanarayanan
Sri Sai Saranya Sridhar

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of System and Control
Automation Research Group
Chalmers University of Technology
Gothenburg, Sweden 2026

Machine Learning Modeling of Thermal System Components For BEVs
A Data-Driven Approach for Health Monitoring and Remaining Useful Life Estimation
Abinaya Rajanarayanan
Sri Sai Saranya Sridhar

University Examiner and Supervisor: Martin Fabian, Chalmers
Industry Supervisor: El Houcine El Mchichi, Volvo Car Corporation

Master's Thesis 2026
Department of Electrical Engineering
Division of System and Control
Automation Research Group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: High-definition visualization of the integrated Transformer Autoencoder and Temporal Convolutional Network (TCN) hybrid model used in this study. The central neural network structure is connected to key thermal system component data points, forming a data-driven pipeline for accurate Remaining Useful Life (RUL) estimation and health monitoring.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Machine Learning Modeling of Thermal System Components For BEVs
A Data-Driven Approach for Health Monitoring and Remaining Useful Life Estimation
Abinaya Rajanarayanan
Sri Sai Saranya Sridhar
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Thermal-management components such as valves, fans and pumps are essential for maintaining the safety, efficiency, and reliability of Battery Electric Vehicles (BEVs). Their operation varies across a wide range of operating conditions, while progressive degradation can reduce thermal performance and increase the risk of unexpected maintenance. This thesis investigates a machine-learning-based framework for modelling the behaviour of selected thermal-system components and estimating their Remaining Useful Life (RUL) using condition-monitoring data. The methodology includes data analysis, generation and preparation of representative operating data, extraction of degradation-relevant features, construction of component health indicators, and RUL estimation. The developed framework demonstrates how data-driven models can support component health assessment when real run-to-failure data are limited. The work provides a basis for future validation using measured degradation data and for the development of condition-based maintenance strategies for BEV thermal systems.

Keywords: Battery Electric Vehicle, Thermal Management System, Machine Learning, Remaining Useful Life, Health Indicator, Predictive Maintenance, Component Degradation, Multivariate Time-Series Data.

Acknowledgements

We would like to express our sincere gratitude to El Houcine EL MCHICHI, our supervisor at Volvo Cars Corporation, for his expert advice and unwavering support throughout this project. We would also like to sincerely thank Yashasvi Nandivada at Volvo Cars for his valuable technical guidance, insightful discussions, and continuous support throughout the thesis. We are also deeply thankful to Martin Fabian, our supervisor and examiner at Chalmers University of Technology, for his valuable feedback during our meetings and for guiding us through the challenges of this work.

Our heartfelt thanks go to Mounica Johansson, our manager at Volvo Cars, for her flexibility in accommodating our requirements and for her continuous support during the project. We are immensely grateful for all the academic support, resources, and opportunities provided by Chalmers University of Technology and Volvo Cars Corporation, which were instrumental in completing this thesis.

Finally, we extend special thanks to our families for their love and encouragement, which have been invaluable throughout this journey. This project would not have been the same without the guidance and support we received from all those involved.

Abinaya Rajanarayanan & Sri Sai Saranya Sridhar, Gothenburg, August 2026

Declaration of Generative AI

Generative AI tools were used to improve grammar, vocabulary, and clarity in the written text. They were not used to generate the original academic content or conclusions of this thesis. The tools were also used to suggest code improvements, support documentation, and assist in generating synthetic datasets used in the methodologies developed in this thesis. All generated outputs were reviewed, verified, and adapted by the authors, who take full responsibility for the final content of the thesis.

Abinaya Rajanarayanan and Sri Sai Saranya Sridhar, Gothenburg, August 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

BEV	Battery Electric Vehicles
RUL	Reamining Useful Life
HI	Health Indicator
TCN	Temporal Convolutional Network
MHSA	Multi-Head Self-Attention
FFN	Feed-Forward Network
HV	High Voltage
ED	Electric Drive
HVP	Battery Voltage Pump

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

n	Index of an input window within a training batch
t	Index of a time step within an input window
f	Index of an input feature
j	Index of a convolution filter element

Parameters

N	Number of input windows in a training batch
T	Number of time steps in each input window
F	Number of input features
C	Number of learned temporal feature channels
D_z	Dimension of the latent representation
K	Convolution kernel size
d	Dilation factor
w_j	Convolution filter weight corresponding to index j
b	Bias term of the convolution operation
$\mathbf{W}_e$	Trainable weight matrix used to project the pooled feature representation into the latent space
$\mathbf{b}_e$	Trainable bias vector used in the latent-space projection
$\boldsymbol{\theta}_e$	Trainable parameters of the encoder
$\boldsymbol{\theta}_p$	Trainable parameters of the decoder projection function
$\boldsymbol{\theta}_d$	Trainable parameters of the decoder

μ_x	Mean of variable x, calculated from the healthy training data
σ_x	Standard deviation of variable x, calculated from the healthy training data
x'	Standardised value of variable x, expressed relative to its healthy-training mean and standard deviation
d_k	Dimension of the Query and Key vectors
$\mathbf{W}_Q$	Learnable Query projection matrix
$\mathbf{W}_K$	Learnable Key projection matrix
$\mathbf{W}_V$	Learnable Value projection matrix
R_{base}	Base observation covariance
$\mathbf{b}_r$	Reconstruction layer bias vector
d_k	Dimension of the Query and Key vectors
d_{model}	Embedding dimension
ϵ	Small positive constant to prevent division by zero
R_h	Hydraulic resistance
w_t	Process noise
v_t	Measurement noise

Variables

x	Original value of the time-series signal or contextual variable before normalisation
x'	Normalised value obtained after standardisation
$\mathbf{X}$	Original multivariate input sequence
$X_{n,t,f}$	Original value of feature f at time step t in input window n
$\hat{\mathbf{X}}$	Reconstructed multivariate input sequence produced by the decoder
$\hat{X}_{n,t,f}$	Reconstructed value of feature f at time step t in input window n
x_t	Input value at time step t in a causal convolution
h_t	Output of the causal convolution at time step t
$\mathbf{H}$	Temporal feature map produced by the stacked TCN blocks
$\mathbf{H}_{:,t}$	Feature-channel vector at time step t
$\bar{\mathbf{h}}$	Temporally pooled feature representation obtained by global average pooling
$\mathbf{z}$	Compressed latent representation of the input sequence
$\mathbf{c}$	Vector of contextual operating variables

$\mathbf{u}$	Combined latent-context representation, defined as $[\mathbf{z}; \mathbf{c}]$
$\mathbf{U}$	Intermediate decoder representation used as input to the temporal reconstruction layers
$\mathcal{L}_{\text{rec}}$	Mean Squared Error reconstruction loss calculated over a training batch
HI_n	Reconstruction-based Health Indicator for input window n
$\mathbf{Q}$	Query matrix
$\mathbf{K}$	Key matrix
$\mathbf{V}$	Value matrix
$\mathbf{Z}$	Position-aware input embedding
$\mathbf{H}_1$	Output of the first residual connection and layer normalization
$\mathbf{H}_2$	Output of the feed-forward network after residual connection and layer normalization
x_t	Estimated signal state at time t
y_t	Measured observation at time t
ΔP	Pressure drop
Q	Coolant flow rate
L_{MSE}	Mean squared error loss
$R_{j,t}$	Observation covariance for signal j at time t
S_t^{system}	System-support score
$S_{j,t}^{\text{peer}}$	Peer-support score
$\hat{S}_{j,t}^{\text{peer}}$	Smoothed peer-support score
$\bar{p}_{j,t}$	Supported percentile score
$p_{j,t}$	Local percentile score

Functions

$f_{\text{TCN}}(\cdot)$	Stacked temporal convolutional blocks used to extract temporal features
$f_{\boldsymbol{\theta}_e}(\cdot)$	Encoder function parameterised by $\boldsymbol{\theta}_e$
$p_{\boldsymbol{\theta}_p}(\cdot)$	Projection function that transforms the latent-context representation into the decoder representation
$g_{\boldsymbol{\theta}_d}(\cdot)$	Decoder function parameterised by $\boldsymbol{\theta}_d$
$f_{\text{enc}}(\cdot)$	Encoder function
$f_{\text{dec}}(\cdot)$	Decoder function
$f_{\text{TAE}}(\cdot)$	Transformer Autoencoder function

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background

Battery Electric Vehicles (BEVs) are increasingly regarded as an important pathway towards more energy-efficient and lower-emission road transportation [1]. Their performance depends on the reliable operation of temperature-sensitive subsystems such as the high-voltage battery, electric drive, power electronics, charging system, and passenger cabin. These subsystems continuously generate and exchange heat under driving, charging, and changing environmental conditions. Effective thermal management is therefore essential for maintaining safety, energy efficiency, charging capability, driving range, passenger comfort, and component durability [2].

The Thermal Management System regulates heat flow between the battery, electric drive, cabin, and surrounding environment through actively controlled components such as coolant pumps, radiator fans, and coolant-control valves [2]. Their operating behaviour varies with ambient temperature, vehicle speed, thermal load, battery condition, charging power, and the selected thermal mode.

Over time, the performance of these components may change because of mechanical, hydraulic, electrical, or sensor-related degradation. Pump bearing wear may increase friction and power demand, while seal leakage or impeller wear may reduce pressure and coolant flow. Fan degradation can influence rotational response and cooling effectiveness, whereas valves may develop delayed movement, sticking, or position-sensor drift.

In prognostics, Remaining Useful Life (RUL) refers to the estimated duration for which a component can continue to perform its intended function before reaching a defined end-of-life condition. A Health Indicator (HI) is often used to summarize degradation-related changes in measurable variables such as temperature, speed, flow, pressure, power consumption, or tracking error. As the HI approaches a predefined threshold, the remaining distance to that threshold is used to estimate RUL.

The end-of-life condition may represent complete failure, unacceptable performance loss, excessive energy demand, insufficient cooling capability, or violation of a functional limit. Since future operating conditions and degradation behaviour are uncertain, RUL should be interpreted as a condition-based estimate that supports maintenance planning rather than as an exact prediction of failure [3]. This creates a need for data-driven methods that can model normal component behaviour, detect meaningful deviations over time, and convert those deviations into reliable HI for RUL estimation.

1.2 Motivation

Maintenance strategies are generally divided into reactive, preventive, and predictive approaches.

Reactive maintenance is performed after a component has failed. Although it avoids unnecessary early replacement, it may result in unexpected vehicle downtime, secondary damage, and higher repair costs.

Preventive maintenance replaces or services components at predetermined time or usage intervals. It is easier to schedule than reactive maintenance, but it may lead to healthy components being replaced too early and may not prevent failures that occur unexpectedly between service intervals.

Predictive maintenance uses measured operating behaviour to assess component condition and determine when intervention is actually needed. Its purpose is to identify developing degradation, estimate remaining life, reduce unnecessary replacement, and support maintenance planning before functional failure occurs.

For BEV thermal systems, predictive maintenance is especially valuable because degradation in one component can influence several connected subsystems. Reduced pump performance can decrease coolant circulation, impaired fan operation can limit radiator heat rejection, and valve faults can disturb coolant routing. These effects may influence battery temperature, charging capability, propulsion performance, energy consumption, and passenger comfort.

Machine-learning-based monitoring provides a way to analyse several interacting signals simultaneously. Autoencoders can learn relationships that are difficult to express using individual thresholds alone. The reconstruction error produced by these models can be converted into a Health Indicator, providing a continuous representation of component condition rather than only a binary healthy-or-faulty classification.

Therefore, this thesis aims to determine whether thermal-system measurements can be transformed into meaningful Health Indicators for detecting degradation and estimating the RUL of selected BEV thermal-management components.

1.3 Aim and Objectives

1.3.1 Aim

The aim of this thesis is to develop and evaluate machine-learning-based methods for assessing the health and estimating the Remaining Useful Life of selected thermal-management components in Battery Electric Vehicles.

The work focuses on the Electric Drive coolant pump, High Voltage battery coolant pump, radiator fan, and coolant-control valves.

1.3.2 Objectives

- To analyse the behaviour of selected thermal-system components under different driving, charging, environmental, and thermal operating conditions.
- To identify signals and derived features that describe component operation and degradation-sensitive behaviour.
- To develop independent Transformer Autoencoder and Temporal Convolutional Network Autoencoder methodologies for learning multivariate temporal behaviour.
- To calculate reconstruction error and transform it into component Health Indicators.
- To evaluate whether the Health Indicators can distinguish normal behaviour from increasing degradation levels.
- To relate Health Indicator progression to the remaining-life fraction and estimate the Remaining Useful Life of the monitored components.

1.4 Limitations

RUL estimation is inherently uncertain because future component life depends on operating conditions, load history, environmental exposure, degradation mechanism, and the selected end-of-life criterion. The predicted RUL should therefore be interpreted as a condition-based estimate rather than an exact failure time. Previous studies also identify uncertainty, changing operating conditions, and model generalisation as major challenges in data-driven prognostics [10].

The Health Indicator used in this study is derived from model reconstruction error and therefore represents deviation from learned component behaviour rather than direct physical degradation. An increased reconstruction error may result from component degradation, but it may also be influenced by unfamiliar operating conditions, or variations not sufficiently represented during model training. The reliability of the estimated RUL consequently depends on signal selection, preprocessing, model performance, and threshold definition.

The study considers selected degradation mechanisms for coolant pumps, radiator fans, and coolant-control valves. Real components may experience combined faults, nonlinear ageing, manufacturing variation, and interactions with other thermal-system components. In addition, the Transformer Autoencoder and TCN Autoencoder were developed using different data-processing and modelling pipelines; therefore, their results are evaluated independently and are not treated as a direct model-to-model comparison.

1.5 Literature Review

1.5.1 Transformer Autoencoder for Anomaly Detection

Lu et al. [**?**] proposed a Transformer Autoencoder (TAE) for unsupervised temporal anomaly detection by combining the self-attention mechanism of the Transformer architecture with sequence reconstruction and bidirectional sequence prediction. The model employs a Transformer encoder–decoder architecture to learn the normal temporal behaviour of multivariate time-series data. During inference, deviations from the learned healthy behaviour produce higher reconstruction errors, enabling anomalies to be identified without requiring labelled fault data. The use of self-attention allows the model to capture both short-term and long-term temporal dependencies that are difficult to model using conventional recurrent neural networks (RNNs).

The proposed work adopts the core Transformer Autoencoder framework introduced by Lu et al., particularly the encoder–decoder architecture and the reconstruction-based anomaly detection strategy. Similar to their approach, the model is trained exclusively using healthy operating data, allowing it to learn the normal operating characteristics of thermal management components. During testing, degraded operating conditions generate higher reconstruction errors, which are used to detect deviations from healthy behaviour.

Although Lu et al. combine sequence reconstruction with forward and backward prediction tasks, the proposed work employs only the reconstruction branch of the Transformer Autoencoder. The objective of this work is to accurately characterize the normal operating behaviour of thermal system components through reconstruction rather than jointly optimizing forecasting and reconstruction. By focusing on a single reconstruction objective, the proposed model maintains a simpler architecture while directly supporting reconstruction-error-based anomaly detection.

However, anomaly detection alone is insufficient for predictive maintenance, as it only indicates whether abnormal behaviour exists without describing how degradation evolves over time or how much useful life remains. Therefore, the reconstruction error generated by the Transformer Autoencoder serves as the starting point for the subsequent prognostic analysis rather than the final output of the proposed framework.

To transform the anomaly information into a degradation representation suitable for prognostics, a Health Indicator (HI) is constructed from the reconstruction error. The Health Indicator provides a continuous representation of component degradation and forms the basis for Remaining Useful Life (RUL) estimation. The concept of using Health Indicators as intermediate degradation representations is widely adopted in Prognostics and Health Management (PHM), where condition-monitoring information is converted into a degradation trajectory before performing RUL prediction.

Since the Health Indicator is derived from reconstruction errors obtained under varying operating conditions, it may exhibit short-term variations while still reflecting the overall degradation trend of the component. Directly estimating RUL from these

variations could lead to unstable life predictions. Therefore, the proposed framework adopts a Kalman Filter-based degradation tracking approach to estimate the underlying degradation trend from the Health Indicator. Rather than responding to every short-term variation, the Kalman Filter recursively combines current and historical degradation information to produce a smoother and more consistent degradation trajectory. This estimated degradation trend is subsequently used for Remaining Useful Life estimation through an exponential degradation model.

Consequently, the proposed framework extends the Transformer Autoencoder-based anomaly detection concept presented by Lu et al. by integrating Health Indicator construction and Kalman Filter-based degradation tracking into a complete prognostic pipeline capable of estimating the Remaining Useful Life of thermal management components.

2

Theory

2.1 Unsupervised Learning for Condition Monitoring

In many industrial condition-monitoring applications, measurements from healthy operation are widely available, whereas labelled degradation and failure data are limited. Component failures occur relatively infrequently, and collecting representative degradation data under different operating conditions can be costly and time-consuming. Consequently, supervised learning methods that require labelled examples of different fault or degradation states may not always be applicable.

Unsupervised learning provides an alternative approach in which the model learns the characteristics of normal system operation without requiring predefined fault labels. In reconstruction-based anomaly detection, a model is trained using healthy operating data and learns the relationships among the measured variables. During inference, the model attempts to reconstruct previously unseen observations based on the normal patterns learned during training.

When an input sequence follows the learned healthy behaviour, the model is generally able to reconstruct it with a relatively low error. When the relationships among the signals differ from those observed during training, the reconstruction error may increase. Such deviations can indicate abnormal operation or possible degradation.

However, the reconstruction error is not a direct measurement of physical degradation. It may also be affected by operating conditions, transient behaviour, measurement noise, and regions of the operating space that are insufficiently represented in the training data. Therefore, reconstruction error is commonly interpreted as an indicator of deviation from the learned healthy behaviour and may require additional processing before it is used for health assessment or Remaining Useful Life estimation.

2.2 Model-1 Transformer AutoEncoders

2.2.1 Autoencoders

An autoencoder is a neural network trained to reconstruct its input. It consists of two main parts: an encoder and a decoder. The encoder transforms the input data

into an internal learned representation, while the decoder uses this representation to reconstruct the original input [5].

For an input sequence $\mathbf{X}$, the reconstructed output can be expressed as

$$\hat{\mathbf{X}} = f_{\text{dec}}\left(f_{\text{enc}}(\mathbf{X})\right), \tag{2.1}$$

where $f_{\text{enc}}(\cdot)$ denotes the encoder, $f_{\text{dec}}(\cdot)$ denotes the decoder, and $\hat{\mathbf{X}}$ is the reconstructed sequence.

The parameters of the autoencoder are optimized by minimizing a reconstruction loss that measures the difference between the original input and its reconstruction. A commonly used reconstruction loss is the mean squared error (MSE). The reconstruction capability of an autoencoder can also be exploited for anomaly detection. When the model is trained predominantly on normal observations, it learns the characteristics of the normal data distribution. Consequently, observations that are similar to the learned normal patterns are generally reconstructed with relatively low error, whereas observations that differ from these patterns may result in larger reconstruction errors. This principle has been investigated for neural-network-based anomaly detection using reconstruction error [14].

For multivariate time-series data, however, conventional fully connected autoencoders do not explicitly account for the temporal relationships between observations. Since temporal dependencies can contain important information about the dynamic behaviour of a system, an architecture capable of modelling relationships across different time steps is required. This motivates the use of a Transformer-based autoencoder, which combines reconstruction-based learning with attention-based temporal modelling.

2.2.2 Transformer Architecture

The Transformer is a neural network architecture designed to process sequential data using attention mechanisms. Unlike recurrent neural networks, which process observations sequentially, the Transformer can process all time steps of an input sequence simultaneously. This enables efficient training and allows relationships between distant observations in a sequence to be modelled directly.

The central component of the Transformer is the self-attention mechanism. Self-attention allows the representation of each time step to be updated using information from every other time step in the sequence. As a result, the model can identify which observations are most relevant when representing a particular point in time.

Before a sequence is processed by the attention mechanism, the input features are transformed into a common embedding space. Since self-attention alone does not preserve the order of observations, positional information is also added to the embedded sequence [13].

2.2.2.1 Input Embedding and Positional Information

Let the input sequence be represented as

$$\mathbf{X} \in \mathbb{R}^{L \times d}, \tag{2.2}$$

where L is the number of time steps and d is the number of input features. An embedding layer projects the input features into a representation of dimension d_{model},

$$\mathbf{E} = \mathbf{X}\mathbf{W}_e + \mathbf{b}_e, \tag{2.3}$$

where $\mathbf{W}_e$ and $\mathbf{b}_e$ are trainable parameters.

The embedding operation creates a common representation for all time steps and allows the model to learn combinations of the original input features that are useful for sequence reconstruction.

Since the Transformer processes all observations simultaneously, it does not inherently know the temporal order of the sequence. Positional information is therefore added to the embedded representation,

$$\mathbf{Z} = \mathbf{E} + \mathbf{P}, \tag{2.4}$$

where $\mathbf{P}$ denotes the positional representation and $\mathbf{Z}$ is the position-aware sequence. Positional representations may be fixed or learned during training.

2.2.2.2 Self-Attention

For the position-aware input $\mathbf{Z}$, three representations are generated through learnable linear transformations,

$$\mathbf{Q} = \mathbf{Z}\mathbf{W}_Q, \tag{2.5}$$

$$\mathbf{K} = \mathbf{Z}\mathbf{W}_K, \tag{2.6}$$

$$\mathbf{V} = \mathbf{Z}\mathbf{W}_V, \tag{2.7}$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ denote the Query, Key, and Value matrices, respectively.

The Query representation describes the information sought by each time step, while the Key representation describes the information available at every time step. Their similarity determines how strongly different observations should influence one another. The Value representation contains the information that is combined to form the updated sequence representation.

The scaled dot-product attention mechanism introduced by Vaswani et al. [13] is defined as:

$$\text{Attention}\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{2.8}$$

where d_k is the dimension of the Query and Key vectors.

The product $\mathbf{Q}\mathbf{K}^T$ measures the similarity between all pairs of time steps. The scaling factor $\sqrt{d_k}$ limits the magnitude of the similarity scores and improves numerical stability. The Softmax operation converts the scores into normalized attention weights. These weights determine how information from different time steps is combined when constructing the updated representation.

2.2.2.3 Multi-Head Attention

Multi-Head Self-Attention performs several attention operations in parallel. Each attention head uses separate Query, Key, and Value projections and can therefore learn a different representation of the same input sequence.

The output of the attention heads is concatenated and projected into the model dimension,

$$\text{MultiHead}\left(\mathbf{Z}\right) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_h\right)\mathbf{W}_O, \tag{2.9}$$

where h denotes the number of attention heads and $\mathbf{W}_O$ is a trainable output projection.

Using multiple heads allows the model to capture different types of temporal relationships simultaneously. For example, one head may focus on short-term transitions, while another may represent relationships occurring over a longer interval. The interpretation of individual attention heads is not predetermined, but their independent projections increase the representational capability of the model.

2.2.2.4 Feed Forward Network, Residual Connections, and Normalization

The attention output is processed by a position-wise Feed Forward Network. The same network is applied independently to each time step and is commonly expressed as

$$\text{FFN}(\mathbf{x}) = \mathbf{W}_2\sigma\left(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1\right) + \mathbf{b}_2, \tag{2.10}$$

where $\sigma(\cdot)$ is a nonlinear activation function.

The Feed Forward Network introduces nonlinear transformations that allow the model to learn more complex feature relationships after the temporal information has been combined by the attention mechanism.

Residual connections are applied around the attention and Feed Forward Network sub-layers. These connections preserve information from the input of each sub-layer and improve gradient propagation during training. Layer Normalization is also applied to stabilize the distribution of the learned representations.

A Transformer encoder block can therefore be summarized as

$$\mathbf{H}_1 = \text{LayerNorm}\left(\mathbf{Z} + \text{MHSA}(\mathbf{Z})\right), \tag{2.11}$$

followed by

$$\mathbf{H}_2 = \text{LayerNorm}\left(\mathbf{H}_1 + \text{FFN}(\mathbf{H}_1)\right). \tag{2.12}$$

2.2.3 Transformer Autoencoder

A Transformer Autoencoder combines the reconstruction-based learning principle of an autoencoder with the temporal modelling capability of the Transformer. The encoder applies self-attention and nonlinear transformations to learn a sequence representation that captures relationships among observations at different time steps. The decoder then transforms this representation into a reconstruction of the original input sequence [9, 11].

For multivariate time-series condition monitoring, this architecture can learn both temporal dependencies and interactions among multiple measured variables. When trained using healthy sequences, the Transformer Autoencoder learns signal relationships associated with normal operation. During inference, changes in these relationships may lead to increased reconstruction errors [9]

The complete reconstruction can be expressed as

$$\hat{\mathbf{X}} = f_{\text{TAE}}\left(\mathbf{X}; \boldsymbol{\theta}\right), \tag{2.13}$$

where f_{TAE} denotes the Transformer Autoencoder and $\boldsymbol{\theta}$ represents its trainable parameters.

The reconstruction error may then be calculated by comparing $\mathbf{X}$ and $\hat{\mathbf{X}}$.Reconstruction error can be used for anomaly detection in multivariate time-series data [9, 12] also this error provides the basis for subsequent anomaly detection, health assessment, and degradation state estimation.

2.3 Temporal Convolutional Autoencoders

A Temporal Convolutional Autoencoder combines the temporal-feature extraction capability of a Temporal Convolutional Network (TCN) with the representation-learning structure of an autoencoder. The model receives a multivariate time-series

sequence, extracts temporal relationships through convolutional operations, compresses the resulting information into a latent representation, and reconstructs the original sequence through a decoder.

The TCN component enables the model to represent both short-duration variations and longer temporal dependencies. The autoencoder introduces a bottleneck between the encoder and decoder, encouraging the network to preserve the information most relevant for reconstruction. When trained using normal operating data, the model is expected to reconstruct familiar temporal behaviour more accurately than behaviour that differs from the learned patterns [4, 5, 6].

The general information flow is expressed as

$$\mathbf{X} \longrightarrow f_{\boldsymbol{\theta}_e}(\mathbf{X}) \longrightarrow \mathbf{z} \longrightarrow g_{\boldsymbol{\theta}_d}(\mathbf{z}, \mathbf{c}) \longrightarrow \hat{\mathbf{X}}, \tag{2.14}$$

where $\mathbf{X}$ is the multivariate input sequence, $f_{\boldsymbol{\theta}_e}(\cdot)$ is the encoder, $\mathbf{z}$ is the latent representation, $\mathbf{c}$ is a vector of contextual variables, $g_{\boldsymbol{\theta}_d}(\cdot)$ is the decoder, and $\hat{\mathbf{X}}$ is the reconstructed sequence.

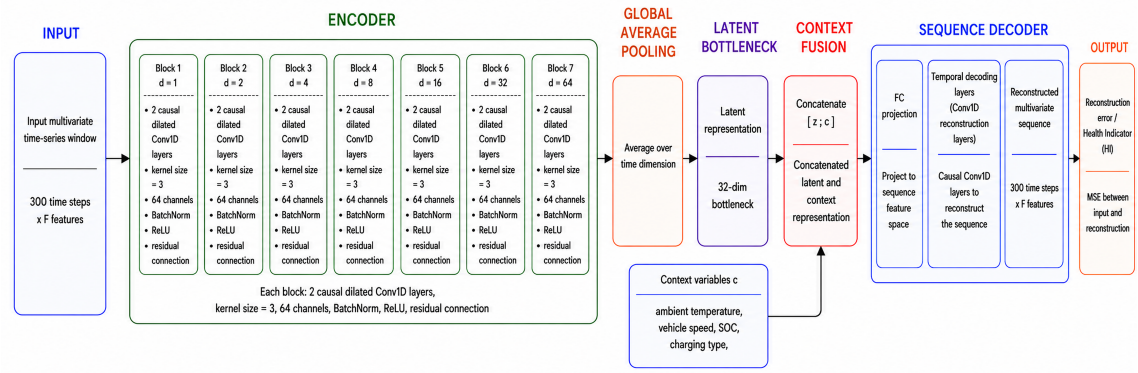

Figure 2.1: Architecture of the baseline TCN Autoencoder.

2.3.1 Temporal Feature Extraction

A TCN applies one-dimensional convolutions along the time axis to extract patterns from sequential measurements. Unlike recurrent neural networks, which process observations successively, temporal convolutions can process several sequence positions in parallel while retaining the temporal structure of the data [4].

A causal convolution ensures that the representation at time step t depends only on the current and preceding observations. Consequently, future observations are not used when calculating the representation of an earlier time step.

To capture dependencies across a wider temporal interval, TCNs use dilated convolutions. A dilated causal convolution can be written as

$$h_t = \sum_{j=0}^{K-1} w_j x_{t-dj} + b, \tag{2.15}$$

where h_t is the convolution output at time step t, x_{t-dj} denotes the input feature value selected from the sequence at time index $t - dj$, K is the kernel size, d is the dilation factor, w_j is the convolution-filter weight corresponding to kernel position j, and b is the bias term.

When $d = 1$, the convolution uses neighbouring observations. Increasing d introduces spacing between the selected samples, allowing a small kernel to cover a wider temporal interval. Progressively increasing dilation factors therefore enable early layers to identify local variations and deeper layers to represent longer-duration temporal dependencies [4, 7].

The convolutional layers are commonly arranged in residual blocks. A residual connection combines the transformed output of a block with its original input. This provides a direct path for information and gradients, supporting the training of deeper networks.

As residual blocks with increasing dilation are stacked, the portion of the sequence that can influence an output increases. This temporal coverage is referred to as the receptive field. A sufficiently large receptive field allows the network to represent relationships occurring across the complete input sequence.

2.3.2 Encoder and Latent Representation

The encoder transforms the original multivariate sequence into a learned temporal feature map:

$$\mathbf{H} = f_{\mathrm{TCN}}(\mathbf{X}), \tag{2.16}$$

where $\mathbf{X} \in \mathbb{R}^{T \times F}$ is the input sequence, T is the number of time steps, F is the number of input features, and $\mathbf{H} \in \mathbb{R}^{C \times T}$ is the temporal feature map containing C learned feature channels.

The feature map preserves information across both the temporal and channel dimensions. To obtain a compact description of the complete sequence, global average pooling may be applied:

$$\bar{\mathbf{h}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{H}_{:,t}, \tag{2.17}$$

where $\bar{\mathbf{h}} \in \mathbb{R}^{C}$ represents the average activation of each feature channel over the sequence.

The pooled representation is then projected into a lower-dimensional latent space:

$$\mathbf{z} = \mathbf{W}_e \bar{\mathbf{h}} + \mathbf{b}_e, \tag{2.18}$$

where $\mathbf{z} \in \mathbb{R}^{D_z}$ is the latent representation, D_z is the latent dimension, and $\mathbf{W}_e$ and $\mathbf{b}_e$ are trainable projection parameters.

The encoder therefore converts a high-dimensional temporal sequence into a compact numerical representation that summarises the dominant patterns required for reconstruction.

2.3.3 Bottleneck and Contextual Conditioning

The latent representation forms the bottleneck between the encoder and decoder. In a strict-bottleneck architecture, the decoder does not receive the complete high-dimensional encoder feature map. Instead, the information required for reconstruction must pass through the compressed latent representation [5].

The reconstruction path can be represented as

$$\mathbf{X} \longrightarrow \mathbf{z} \longrightarrow [\mathbf{z}; \mathbf{c}] \longrightarrow \hat{\mathbf{X}}, \tag{2.19}$$

where $\mathbf{c}$ denotes contextual operating variables and $[\mathbf{z}; \mathbf{c}]$ denotes the concatenation of the latent and contextual representations.

The combined representation is defined as

$$\mathbf{u} = [\mathbf{z}; \mathbf{c}], \tag{2.20}$$

where $\mathbf{u}$ contains both the compressed temporal information and the associated operating context.

Contextual conditioning allows reconstruction to account for systematic variations caused by operating conditions. The same component response may be normal under one operating condition but unusual under another. Including contextual information therefore helps the decoder distinguish operating-condition effects from deviations in the learned temporal behaviour.

The bottleneck capacity must be selected carefully. An excessively restrictive bottleneck may remove information required to reconstruct normal sequences, whereas an overly large bottleneck may allow the model to reproduce inputs without learning a sufficiently selective representation.

2.3.4 Sequence Decoder

The decoder transforms the latent and contextual information into a reconstructed multivariate sequence. The combined latent-context representation is first projected into an intermediate decoder representation:

$$\mathbf{U} = p_{\boldsymbol{\theta}_p}(\mathbf{u}), \tag{2.21}$$

where $p_{\boldsymbol{\theta}_p}(\cdot)$ is a trainable projection function, $\boldsymbol{\theta}_p$ denotes its parameters, and $\mathbf{U}$ is the intermediate representation supplied to the temporal reconstruction layers.

The reconstructed sequence is then obtained as

$$\hat{\mathbf{X}} = g_{\boldsymbol{\theta}_d}(\mathbf{U}), \tag{2.22}$$

where $g_{\boldsymbol{\theta}_d}(\cdot)$ is the decoder, $\boldsymbol{\theta}_d$ represents its trainable parameters, and $\hat{\mathbf{X}} \in \mathbb{R}^{T \times F}$ is the reconstructed version of the original input sequence.

The decoder is not required to mathematically reverse every encoder operation. Instead, it learns a nonlinear mapping that reconstructs the temporal patterns and inter-feature relationships preserved in the latent representation.

2.3.5 Reconstruction-Based Condition Monitoring

An autoencoder trained using normal operating sequences learns a representation of the temporal patterns present in the training data. Inputs that are consistent with this learned behaviour are expected to be reconstructed with relatively low error, whereas unfamiliar or deviating inputs may produce a larger reconstruction error [6].

The reconstruction error can therefore be used as a data-driven indicator of deviation from learned normal behaviour. However, a high reconstruction error does not by itself confirm physical degradation. It may also result from previously unseen operating conditions, sensor disturbances, preprocessing inconsistencies, or insufficient model capacity. A health-related interpretation consequently requires validation using system knowledge and consistency across consecutive temporal observations.

2.3.6 Health Indicator-Based RUL Estimation

A reconstruction-based Health Indicator quantifies how strongly an observed sequence deviates from the normal behaviour learned by an autoencoder. A small Health Indicator indicates that the observed behaviour is similar to the learned healthy condition, whereas an increasing Health Indicator indicates a greater deviation from normal operation. However, the Health Indicator alone does not directly provide the Remaining Useful Life. A relationship must therefore be established between the Health Indicator and the component life fraction.

The remaining-life fraction is denoted by α, where

$$0 \leq \alpha \leq 1. \tag{2.23}$$

Here, $\alpha = 1$ represents a healthy component with its full reference life remaining, while $\alpha = 0$ represents the selected end-of-life condition. The corresponding consumed-life fraction is defined as

$$d = 1 - \alpha. \tag{2.24}$$

A power-law function can be used to describe how degradation progresses as the component life is consumed:

$$q = d^{\beta} = (1 - \alpha)^{\beta}, \tag{2.25}$$

where q is the normalised degradation level and β controls the shape of the degradation progression. When $\beta = 1$, degradation progresses linearly with the consumed-life fraction. Values of $\beta > 1$ represent a progression that becomes more pronounced toward the end of life, whereas $0 < \beta < 1$ represents stronger degradation during the earlier life stages.

The relationship between the Health Indicator and the remaining-life fraction may be expressed as

$$HI(\alpha) = HI_0 + a(1 - \alpha)^{\beta}, \tag{2.26}$$

where HI_0 is the representative Health Indicator under healthy operation, a is a scaling parameter, and β is the degradation-curve shape parameter.

Once this relationship has been established, the Health Indicator obtained for an evaluated window can be mapped to an estimated remaining-life fraction $\hat{\alpha}$. The estimated Remaining Useful Life is then calculated as

$$\widehat{RUL} = \hat{\alpha} T_{\text{ref}}, \tag{2.27}$$

where T_{ref} is the selected reference lifetime of the component. The resulting RUL is therefore a condition-based estimate whose reliability depends on the validity of the assumed degradation progression and the relationship between the Health Indicator and the component condition.

3

Methods

3.1 Methodological Overview

This thesis investigates two separate data-driven approaches for monitoring the health of thermal-management components in Battery Electric Vehicles and estimating their Remaining Useful Life. The first approach is based on a Transformer Autoencoder, while the second uses a Temporal Convolutional Network Autoencoder.

Although both approaches are intended to identify changes in component behaviour and support health assessment, they were developed using different datasets, preprocessing methods, input features, operating conditions, model structures, and degradation scenarios. The two models were therefore evaluated independently within their respective application settings. The purpose of this work is to assess how each approach performs under the conditions for which it was developed and how it can be used to estimate the Remaining Useful Life of the monitored components.

3.2 Overall Framework

This thesis presents a data-driven framework for monitoring the health of Battery Electric Vehicle (BEV) thermal system components and estimating their Remaining Useful Life (RUL) using a Transformer Autoencoder. The proposed framework begins with collecting operational data from the vehicle's thermal management system and concludes with estimating the health condition and remaining useful life of the selected components.

The data used in this work are collected during vehicle testing under different operating conditions. Sensors installed on the thermal system continuously record the behaviour of the components along with other relevant system variables throughout the tests. The resulting dataset consists of time-series measurements that capture the operational characteristics of the system. A detailed description of the data acquisition process, testing conditions, and dataset preparation is provided in Section 3.3.

Before model development, the collected data are preprocessed to ensure they are suitable for training the machine learning model. The preprocessing stage includes organizing the recorded signals into a consistent format, normalizing the input fea-

tures, and dividing the continuous time-series data into fixed-length sliding windows. These steps enable the model to effectively learn the temporal relationships present in the sensor measurements.

The preprocessed data are then used to train a Transformer Autoencoder using data representing normal operating conditions. During training, the model learns the underlying patterns of healthy system behaviour by reconstructing the input sequences. Once trained, the model is applied to unseen data, where the difference between the original input and the reconstructed output, referred to as the *reconstruction error*, is calculated. This reconstruction error provides an indication of how much the current operating behaviour deviates from the learned healthy behaviour.

The reconstruction error is subsequently used to generate a Health Indicator (HI), which represents the condition of the monitored component over time. As the operating behaviour changes, the Health Indicator reflects the progression of degradation and serves as the basis for estimating the Remaining Useful Life (RUL).

3.3 Dataset

The dataset used in this thesis consists of multivariate time-series measurements collected from the Battery Electric Vehicle (BEV) thermal management system during normal vehicle operation under various real-world driving and environmental conditions. The measurements were sampled at a frequency of 1 Hz, providing one observation per second for each recorded variable.

The original dataset contains 138 recorded signals representing the operating conditions of the thermal management system, including temperatures, coolant circulation, actuator responses, control commands, and vehicle operating conditions. Since not all recorded variables contribute equally to the health monitoring task, feature selection and preprocessing were performed to identify the most informative signals for model development.

The available data represent healthy operating behaviour and do not contain progressive component degradation. Consequently, the dataset was used for data analysis, feature engineering, however for model training and RUL prediciton Synthethic dataset has been generated by immitating the behaviour of real data with degradation scenarios.

3.3.1 Data Collection

The dataset contains both sensor measurements and controller-generated variables describing the operating state of the BEV thermal management system.

Temperature measurements were recorded at several locations within the coolant circuit, including the ambient temperature, battery coolant inlet and outlet, radiator outlet, battery coolant pump module inlet, electric drive pump outlet, and average battery temperature. These measurements characterize the thermal behaviour of the cooling circuit and battery system.

The operating conditions of the High Voltage (HV) battery coolant pump and Electric Drive (ED) coolant pump are represented by their requested and actual coolant flow rates together with their requested and actual rotational speeds. Similarly, the radiator fan is described by its requested and actual rotational speeds, while the coolant control valves are represented by their requested and actual valve positions obtained from Valve A and Valve B.

Together, these measurements provide a comprehensive representation of the thermal management system and form the basis for the subsequent feature engineering and preprocessing steps.

3.4 Data Preparation

The collected measurements were stored as multiple CSV files containing multivariate time-series data. Before model development, a data-preparation pipeline was implemented to organize the data, retain the relevant thermal management signals, identify unreliable measurements, and construct improved signal representations suitable for machine learning.

The objective of the preprocessing stage was to improve data quality while preserving genuine system dynamics. Since thermal management signals naturally vary with operating conditions such as ambient temperature, vehicle speed, thermal operating mode, coolant circulation, and actuator operation, abrupt signal changes cannot always be considered measurement noise.

To distinguish measurement anomalies from normal system behaviour, each signal was first analyzed locally to identify rapid changes. These changes were then compared with the behaviour of the remaining signals within a short time interval to estimate a signal-specific trust level. The trust level was subsequently used to combine the original measurement with an adaptive Kalman-filter estimate, reducing unreliable fluctuations while preserving physically meaningful system dynamics.

The complete preprocessing pipeline included signal selection, numerical conversion, temporal sorting, local derivative analysis, percentile-based change scoring, cross-signal support estimation, adaptive Kalman filtering, signal reconstruction, feature engineering, and categorical encoding. The processed datasets were stored separately while preserving their temporal order and contextual information.

3.4.1 Signal Selection

The original dataset contained 138 recorded signals representing different vehicle subsystems. Since this thesis focuses on the health monitoring of the Battery Electric Vehicle (BEV) thermal management system, only the variables directly related to thermal control and coolant circulation were retained for feature engineering. Table 3.1 summarizes the selected signal categories and their purpose in the proposed methodology.

Table 3.1: Selected signal categories used for feature engineering.

Category	Selected Variables	Purpose
Ambient Conditions	Ambient temperature, vehicle speed	Describe the external operating conditions affecting thermal performance and cooling demand.
Temperature Measurements	Battery inlet/outlet, radiator outlet, HV pump inlet, ED pump outlet, chiller inlet/outlet, water condenser inlet/outlet, average battery temperature	Characterize heat transfer and the thermal state of the cooling circuit and battery system.
Coolant Pumps	Requested/actual coolant flow, requested/actual pump speed (HV battery and ED pumps)	Monitor coolant circulation and evaluate pump response to controller commands.
Radiator Fan	Requested/actual fan speed	Evaluate radiator cooling performance and fan response.
Coolant Control Valves	Requested/actual valve positions (Valve A, Valve B, Valve C)	Describe coolant routing and valve operation under different thermal management modes.
Controller	Thermal operating mode	Provide operational context for interpreting component behaviour and control actions.

3.4.2 Numerical Conversion and Time-Order Validation

Each CSV file was processed independently as a multivariate time-series sequence. A `time` column was required to preserve the chronological order of the measurements. Files without this column were excluded from further processing.

All variables, except the time column, were converted to numerical data types, with invalid entries replaced by missing values. The data were then sorted according to the time column and the row indices were reset to ensure a consistent temporal sequence.

Maintaining the correct chronological order is essential for subsequent preprocessing operations, including derivative calculation, rolling-window analysis, and adaptive Kalman filtering. Missing or invalid values were retained at this stage, as they were handled during the later signal reconstruction process.

3.4.3 Local Change Detection

The first signal-quality assessment was performed independently for each numerical variable. The objective was to identify measurements that changed unusually rapidly compared with their local temporal neighbourhood.

A global statistical measure was not used because the vehicle data were collected under different operating conditions. For example, the normal value of a coolant temperature during cold ambient operation may differ considerably from its normal value during high-load or warm-weather operation. A global mean and standard deviation could therefore incorrectly identify valid operating-region changes as abnormal.

Instead, a robust local score was calculated using a centred rolling window. For a signal x_t, the local median at time t was calculated as

$$\tilde{x}_t = \text{median}\left(x_{t-k}, \ldots, x_t, \ldots, x_{t+k}\right), \tag{3.1}$$

where the complete rolling window contains $2k + 1$ observations.

The local variability was estimated using the median absolute deviation, given by

$$\text{MAD}_t = \text{median}\left(|x_i - \tilde{x}_t|\right), \qquad i \in [t-k, t+k]. \tag{3.2}$$

The median absolute deviation is less sensitive to isolated extreme values than the conventional standard deviation. This makes it suitable for vehicle measurements that may contain spikes or short disturbances.

A robust local score can be expressed as

$$z_t = \frac{|x_t - \tilde{x}_t|}{1.4826\,\text{MAD}_t}. \tag{3.3}$$

The factor 1.4826 is used to scale the MAD to a value comparable to the standard deviation for normally distributed data [15]. This scaling does not require the vehicle measurements to be normally distributed; it is used here to obtain a standardised robust measure of local variability.If the local variability was zero or the score could not be calculated, the resulting invalid value was replaced by zero.

In the implemented pipeline, this robust score was not applied directly to the signal values. It was applied to the first difference of each signal,

$$\Delta x_t = x_t - x_{t-1}. \tag{3.4}$$

Analysing Δx_t makes the method sensitive to sudden changes rather than to the absolute level of the signal. This is important because a high coolant temperature is not necessarily erroneous, whereas an unrealistically abrupt change in coolant temperature may indicate a measurement disturbance.

For every signal, the derivative-based score was converted into a percentile rank. Let $z^{\Delta}_{j,t}$ denote the local derivative score of signal j. Its percentile score was calculated as

$$p_{j,t} = \text{PercentileRank}\left(z^{\Delta}_{j,t}\right). \tag{3.5}$$

The resulting value lies between 0 and 1. A value close to 0 indicates that the change is small relative to most other changes in the same signal. A value close to 1 means that the local change is among the most extreme changes observed within that signal sequence.

The percentile transformation allows signals with different physical units and numerical ranges to be compared. For example, temperature, pump speed, valve position, and coolant flow may have very different scales. Their percentile scores nevertheless share the same interval between zero and one.

A rolling window of 11 samples was used in the current implementation. Since the data were sampled at 1 Hz, this corresponds to an 11-second local window. The centered window allows each point to be compared with observations immediately before and after it.

3.4.4 Temporal and Cross-Signal Support Analysis

An extreme local change in a single signal is not sufficient to determine whether the observation represents sensor noise. Genuine thermal-system events may also produce rapid changes. For example, a change in thermal operating mode may affect pump speed, coolant flow, valve position, and temperature-related variables within a short period.

The preprocessing method therefore evaluates whether a detected change is supported by the behaviour of other measurements. Before signals were compared, a small temporal tolerance was introduced. This was necessary because related components do not always react at exactly the same sample. A controller request may change first, followed by the actuator response, coolant-flow change, and temperature response.

For each signal, the supported percentile score was calculated as the maximum percentile score within a local time interval:

$$\bar{p}_{j,t} = \max_{\tau \in [t-h, t+h]} p_{j,\tau}, \tag{3.6}$$

where h is the temporal support-window size.

The implemented support window was set to one sample. At a sampling rate of 1 Hz, the analysis therefore considered the interval from one second before to one second after the current observation:

$$[t-1, t, t+1]. \tag{3.7}$$

Using the maximum value within this interval allows related changes to support one another even when their responses are shifted slightly in time.

An overall system-support score was calculated by averaging the supported percentile values across all numerical signals:

$$S_t^{\text{system}} = \frac{1}{M} \sum_{j=1}^{M} \bar{p}_{j,t}, \tag{3.8}$$

where M denotes the number of numerical signals.

A high value of S_t^{system} indicates that several measurements show notable local changes at approximately the same time. This is more consistent with a genuine system-level event than with isolated sensor noise.

However, the system-support score includes the signal currently being evaluated. This could allow a highly unusual signal to contribute to its own support. To avoid this, a leave-one-signal-out approach was used when calculating the trust level of each individual variable.

For signal j, the peer-support score was calculated using all other signals:

$$S_{j,t}^{\text{peer}} = \frac{1}{M-1} \sum_{\substack{m=1 \\ m \neq j}}^{M} \bar{p}_{m,t}. \tag{3.9}$$

This means that a signal cannot classify its own change as a genuine event. Instead, the change must be supported by the remaining variables. An isolated spike in one temperature sensor, for example, will receive weak peer support if no corresponding change occurs in the pump, fan, valve, flow, or other temperature measurements.

The peer-support score was further smoothed using a centred three-sample rolling mean:

$$\hat{S}_{j,t}^{\text{peer}} = \frac{1}{3} \sum_{\tau=t-1}^{t+1} S_{j,\tau}^{\text{peer}}. \tag{3.10}$$

At 1 Hz, this represents approximately three seconds. This additional operation reduces rapid fluctuations in the support estimate and prevents the trust level from changing sharply because of a single sample.

It is important to note that cross-signal support does not prove that every jointly observed change is physically correct. Instead, it provides a data-driven indication of whether a change is isolated or is part of a wider system response. The method is based on the assumption that genuine operating transitions often influence several related thermal-system variables, whereas random sensor disturbances are more likely to appear independently.

3.4.5 Adaptive Kalman Smoothing

After the peer-support score had been calculated, an adaptive Kalman filter was used to create a smoothed estimate of each signal. The Kalman filter combines a predicted state with the current observation while accounting for uncertainty in both the system evolution and the measurement.

A simple one-dimensional state model was used for each signal:

$$x_t = x_{t-1} + w_t, \tag{3.11}$$

$$y_t = x_t + v_t, \tag{3.12}$$

where x_t is the estimated underlying signal state, y_t is the recorded observation, w_t represents process uncertainty, and v_t represents measurement uncertainty.

The transition and observation matrices were both set to one. Therefore, the state model assumes that the next signal value is related to the previous value, while still allowing gradual changes through the process covariance.

A fixed process covariance of 0.05 was used. In contrast, the observation covariance was varied over time according to the peer-support score Adaptive Kalman filtering can adjust the measurement-noise covariance when the measurement uncertainty varies over time [16]. In this work, the peer-support score was used to determine the measurement covariance for each signal. The dynamic observation covariance for signal j was defined as

$$R_{j,t} = \frac{R_{\text{base}}}{\hat{S}_{j,t}^{\text{peer}} + \epsilon}, \tag{3.13}$$

where $R_{\text{base}} = 5$ and ϵ is a small constant used to prevent division by zero.

The observation covariance represents the uncertainty associated with the measured value. When the peer-support score is high, $R_{j,t}$ becomes smaller, and the Kalman smoother places greater confidence in the recorded measurement. When peer support is low, $R_{j,t}$ becomes larger, and the smoother relies more strongly on the temporal state estimate.

This behaviour is suitable for the intended application. A rapid change supported by several other signals is more likely to be a meaningful system event and should therefore remain visible in the processed data. In contrast, an isolated change with little support from the rest of the system is treated as less reliable and is smoothed more strongly.

The initial state was set to the first finite observation of the sequence. If the first value was invalid, the mean of the available observations was used. The initial state covariance was set to one. If Kalman smoothing could not be completed for a particular signal, the original signal was retained as a fallback. This prevented the failure of one signal-processing operation from stopping the complete data-preparation pipeline.

3.4.6 Trust-Based Signal Reconstruction

The Kalman-smoothed estimate was not used to replace the complete recorded signal. Replacing all observations with a smoothed version could reduce genuine transient behaviour, including controller actions, actuator responses, mode transitions, and component dynamics. Instead, a soft reconstruction method was used to combine the raw measurement and the smoothed estimate.

For each signal, the peer-support score was transformed into a trust factor:

$$\alpha_{j,t} = \text{clip}\left(\frac{\hat{S}_{j,t}^{\text{peer}} - 0.6}{0.1}, 0, 1\right). \tag{3.14}$$

According to this formulation:

- when the peer-support score is less than or equal to 0.6, the trust factor is zero;
- when the peer-support score is between 0.6 and 0.7, the trust factor gradually increases from zero to one; and
- when the peer-support score is greater than or equal to 0.7, the trust factor is one.

The reconstructed signal was then calculated as

$$x_{j,t}^{\text{recon}} = \alpha_{j,t} x_{j,t}^{\text{raw}} + (1 - \alpha_{j,t})\, x_{j,t}^{\text{smooth}}. \tag{3.15}$$

When the trust factor is close to one, the reconstructed signal remains close to the original measurement. This occurs when the behaviour of the other variables supports the observed change. When the trust factor is close to zero, the reconstructed signal is primarily determined by the Kalman-smoothed estimate. Intermediate trust values produce a weighted combination of the two.

This continuous blending avoids an abrupt binary decision in which a sample is classified as either completely valid or completely invalid. In real vehicle data, measurement reliability is not always clearly separated into these two categories. A gradual weighting strategy therefore provides a more flexible treatment of uncertain observations.

The complete signal preprocessing procedure, including the calculation of temporal variation, local deviation, signal confidence, temporal and peer support, adaptive Kalman filtering, and trust factor based reconstruction, is summarized in Algorithm[1].

3.5 Feature Engineering

Following data preparation, additional features were derived from the processed signals to provide a more informative representation of the thermal management

Algorithm 1 Signal Preprocessing

Require: Available signals $\mathcal{S}_k$
Ensure: Reconstructed signals $\mathcal{S}_{\text{recon}}$
for each signal $s \in \mathcal{S}_k$ **do**
    Compute temporal variation Δs_t
    Estimate local deviation score $z_s(t)$
    Compute signal confidence $p_s(t)$
    Compute temporal support $\bar{p}_s(t)$
    Identify peer signals $\mathcal{S}_{-s}$
    if $|\mathcal{S}_{-s}| > 0$ **then**
        Compute peer support $p_{\text{peer}}(t)$
    else
        Set $p_{\text{peer}}(t) \leftarrow \bar{p}_s(t)$
    end if
    Compute trust factor $\alpha_s(t)$
    if $\alpha_s(t) > 0.6$ **then**
        Apply adaptive Kalman filtering
    else
        Retain original signal $s(t)$
    end if
    Reconstruct signal $s_{\text{recon}}(t)$
end for
return $\mathcal{S}_{\text{recon}}$

system. While the original sensor measurements describe the instantaneous operating state of individual components, they do not explicitly capture the relationships between controller commands, actuator responses, coolant circulation, or thermal behaviour. Feature engineering was therefore performed to extract variables that better represent the dynamic characteristics of the system and improve the ability of the machine learning model to identify abnormal operating behaviour.

The engineered features were developed based on the physical operation of the Battery Electric Vehicle (BEV) thermal management system and were grouped into five categories: request–actual difference features, valve behaviour features, pump–flow interaction features, flow–valve interaction features, and temperature difference features.

3.5.1 Request–Actual Difference Features

To characterize actuator performance, request–actual feature pairs were generated for the coolant pumps, radiator fan, and coolant control valves. The selected request–actual pairs are listed in Table 3.2.

For each request–actual signal pair, additional features were generated to capture both the steady-state and dynamic behaviour of the actuators. These features include the signed error, absolute error, response ratio, first-order temporal difference, one- and five-step lag features, and rolling mean and rolling standard deviation

Table 3.2: Request–actual signal pairs used for feature engineering.

Component	Requested Variable	Actual Variable
HV Battery Pump	Pump speed request	Pump speed
ED Pump	Pump speed request	Pump speed
Radiator Fan	Fan speed request	Fan speed
Valve A	Valve position request	Valve position sensor
Valve B	Valve position request	Valve position sensor
Valve C	Valve position request	Valve position sensor

computed over a five-sample window.

These engineered features provide information about actuator tracking performance, temporal behaviour, and short-term variability, enabling the machine learning model to better distinguish normal operating behaviour from potential degradation.

3.5.2 Valve Behaviour Features

In addition to the request–actual valve position features, additional features were generated from the measured valve positions to describe the operating behaviour of the coolant control valves. While the request–actual features indicate whether the valve follows the controller command, they do not provide information about how the valve behaves during operation.

To capture the valve behaviour, features were generated to identify whether the valve was moving, whether it was operating in a partially open position, and how much the valve moved over a short period of time. Rolling statistics were also calculated to determine how frequently the valve changed its position within a five-sample window.

These features provide additional information about the dynamic behaviour of the valves and help distinguish normal valve operation from abnormal movement patterns. For example, excessive valve movement, frequent oscillations, or prolonged operation in intermediate positions may indicate increased mechanical resistance or the early stages of valve degradation.

3.5.3 Pump–Flow Interaction Features

The coolant flow generated by a centrifugal pump is closely related to its rotational speed. According to the pump affinity law, the coolant flow rate is proportional to the pump speed under similar hydraulic conditions,

$$Q \propto N, \tag{3.16}$$

where Q is the measured coolant flow rate and N is the pump rotational speed.

Under normal operating conditions, a given pump speed is expected to produce a corresponding coolant flow rate. However, component degradation, such as impeller wear, internal leakage, or increased hydraulic resistance, may reduce the coolant flow even when the pump continues to operate at the same rotational speed.

To capture this relationship, a pump–flow interaction feature was generated as

$$F_{\text{pump-flow}} = \frac{Q}{N}, \tag{3.17}$$

where $F_{\text{pump-flow}}$ represents the coolant flow generated per unit pump speed.

This feature provides a proxy measure of pump effectiveness. Under similar operating conditions, the ratio is expected to remain relatively constant. A reduction in the ratio indicates that less coolant flow is being produced for the same pump speed, which may suggest reduced pump performance or the onset of component degradation.

3.5.4 Flow–Valve Interaction Features

The coolant flow in the thermal management system is influenced by both the coolant pump and the hydraulic resistance of the cooling circuit, which is regulated by the coolant control valves. Consequently, the measured coolant flow must be interpreted together with the corresponding valve position.

The relationship between pressure drop and coolant flow is given by

$$\Delta P = R_h Q^2, \tag{3.18}$$

where ΔP is the pressure drop, Q is the coolant flow rate, and R_h is the hydraulic resistance. Changes in valve position modify the hydraulic resistance, thereby affecting the coolant flow under the same pump operating condition.

To account for this dependency, flow–valve interaction features were generated by combining the measured coolant flow with the corresponding valve positions. These features provide additional information about the hydraulic state of the cooling circuit, enabling the machine learning model to better distinguish normal operating variations from potential component degradation.

3.5.5 Temperature Difference Features

Temperature difference features were generated between key locations within the thermal management system, including the radiator, battery coolant circuit, and ambient environment. These features provide information about heat transfer and cooling performance that cannot be obtained from individual temperature measurements alone.

The temperature difference between two measurement locations is calculated as

$$\Delta T = T_1 - T_2, \tag{3.19}$$

where T_1 and T_2 are the temperatures measured at two different locations.

To capture temporal thermal behaviour, the first-order temperature difference and a five-sample rolling average were also computed,

$$\Delta T'(t) = T(t) - T(t-1), \tag{3.20}$$

$$\bar{T}(t) = \frac{1}{5}\sum_{i=0}^{4} T(t-i). \tag{3.21}$$

These features describe both the thermal gradients and their temporal evolution, providing additional information for degradation monitoring.

3.6 Sliding Window Generation

Following feature engineering, the multivariate time-series data were segmented into fixed-length sequences using a sliding window approach. Since the Transformer Autoencoder learns temporal dependencies from sequential data, each input sample consists of a sequence of consecutive measurements rather than an individual time step.

Given a multivariate time series

$$X = \{x_1, x_2, \ldots, x_T\}, \tag{3.22}$$

where $x_t \in \mathbb{R}^d$ denotes the feature vector at time step t, each sliding window is constructed as

$$W_i = \{x_i, x_{i+1}, \ldots, x_{i+L-1}\}, \tag{3.23}$$

where L is the window length.

In this work, a window length of $L = 60$ samples was adopted. Since the sensor data were recorded at a sampling frequency of 1 Hz, each window represents 60 seconds of vehicle operation. Consecutive windows were generated using a stride of 10 samples, resulting in overlapping sequences that preserve temporal continuity while increasing the number of training samples.

Consequently, each input sample provided to the Transformer Autoencoder has a dimension of $60 \times d$, where d denotes the total number of engineered features.

3.7 Transformer Autoencoder Framework

A Transformer Autoencoder was used to learn the healthy temporal and multivariate behaviour represented by the prepared input sequences. The model was selected because the thermal management signals contain interactions across both features and time. For example, a controller request may be followed by a delayed response in pump speed, coolant flow, valve position, or temperature. The attention mechanism allows such temporal relationships to be learned without defining them explicitly through physical equations.

The model was trained only using healthy operating sequences. Each sequence was used as both the input and the reconstruction target. The objective was therefore not to predict a future signal value or classify a fault type, but to reconstruct the complete healthy input window.

3.7.1 Input Representation

The preprocessed and engineered features were divided into sliding windows containing 60 consecutive time steps. Each model input is represented as

$$\mathbf{X} \in \mathbb{R}^{60 \times d}, \tag{3.24}$$

where d denotes the number of features after preprocessing and feature engineering.

Each time step contains the simultaneous measurements and derived variables describing the thermal system operating condition. The fixed sequence length provides the model with temporal context while maintaining a uniform input size for training.

3.7.2 Embedding and Positional Encoding

The original feature vectors were projected into a 64-dimensional model space using a fully connected embedding layer,

$$\mathbf{E} = \mathbf{X}\mathbf{W}_e + \mathbf{b}_e, \tag{3.25}$$

where

$$\mathbf{W}_e \in \mathbb{R}^{d \times 64} \tag{3.26}$$

and

$$\mathbf{b}_e \in \mathbb{R}^{64}. \tag{3.27}$$

The resulting embedded sequence has the dimensions

$$\mathbf{E} \in \mathbb{R}^{60 \times 64}. \tag{3.28}$$

This transformation allows the model to learn combinations of the original features that are useful for reconstructing healthy operating sequences. The same embedding transformation is applied to every time step.

Learnable positional embeddings were added to preserve temporal order,

$$\mathbf{Z} = \mathbf{E} + \mathbf{P}, \tag{3.29}$$

where

$$\mathbf{P} \in \mathbb{R}^{60 \times 64}. \tag{3.30}$$

The positional vectors were optimized together with the remaining model parameters. Consequently, the model could learn position-dependent information suitable for the selected window length.

3.7.3 Encoder Architecture

The position-aware embedded sequence was passed through Transformer encoder blocks. Each block consisted of Multi-Head Self-Attention, a position-wise Feed Forward Network, residual connections, and Layer Normalization.

The model used four attention heads. Since the embedding dimension was 64, the representation dimension of each attention head was

$$d_k = \frac{d_{\text{model}}}{h} = \frac{64}{4} = 16, \tag{3.31}$$

where $h = 4$ is the number of attention heads.

Each attention head independently projected the complete 64-dimensional input representation into a 16-dimensional Query, Key, and Value space. Therefore, the heads did not receive fixed, manually divided groups of input features. Instead, every head learned its own transformation of the complete embedded representation.

The attention mechanism allowed each time step to incorporate information from all other time steps in the window. This is relevant for the thermal system because component responses may depend on earlier controller commands, thermal states, and actuator dynamics.

The attention output was combined with the block input through a residual connection and was subsequently normalized. A Feed Forward Network then applied a nonlinear transformation independently to each time step. A second residual connection and Layer Normalization completed the encoder block.

The output of the final encoder block was treated as the learned sequence representation. The term learned representation is used instead of compressed representation because the encoder output is not necessarily smaller than the original sequence in every dimension.

3.7.4 Decoder and Reconstruction Layer

The learned sequence representation was passed to Transformer decoder blocks that reconstructed the complete sequence. The decoder followed a structure similar to the encoder and contained Multi-Head Self-Attention, Feed Forward Networks, residual connections, and Layer Normalization.

The decoder did not use causal masking. Causal masking is required when a model must prevent a time step from accessing future observations, such as in autoregressive forecasting. In the present work, the complete input sequence was available and the objective was to reconstruct all time steps simultaneously. Access to both earlier and later observations within the window was therefore permitted.

The output of the decoder remained in the 64-dimensional model space. A final fully connected reconstruction layer mapped the decoder output back to the original number of input features,

$$\hat{\mathbf{X}} = \mathbf{H}_{\text{dec}}\mathbf{W}_r + \mathbf{b}_r, \tag{3.32}$$

where

$$\hat{\mathbf{X}} \in \mathbb{R}^{60 \times d}. \tag{3.33}$$

The reconstructed sequence had the same dimensions as the input sequence, enabling element-wise comparison between the measured and reconstructed values.

3.7.5 Training Procedure

The Transformer Autoencoder was trained exclusively using healthy operating sequences. The training objective was to minimize the difference between the input sequence $\mathbf{X}$ and its reconstruction $\hat{\mathbf{X}}$.

The Mean Squared Error loss was used,

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} \left(X_i - \hat{X}_i \right)^2, \tag{3.34}$$

where N denotes the total number of reconstructed values included in the loss calculation.

Mean Squared Error penalizes larger reconstruction deviations more strongly because the errors are squared. This makes the loss suitable for training the model to accurately reproduce the continuous-valued thermal system signals.

The Adam optimizer was employed with an initial learning rate of 0.001. Training was performed using mini-batches of 32 sequences for a maximum of 15 epochs. The use of mini-batches reduced memory requirements and provided repeated parameter updates within each epoch.

The validation loss was evaluated after every epoch. Early Stopping with a patience of five epochs was applied, meaning that training was terminated when the validation loss failed to improve for five consecutive epochs. The model parameters corresponding to the lowest validation loss were retained. This procedure reduced unnecessary training and limited overfitting to the healthy training sequences.

3.7.6 Degradation State Estimation

The reconstruction error obtained from the Transformer Autoencoder quantifies the deviation between the observed system behaviour and the healthy operating patterns learned during training. Since the model is trained exclusively using healthy operating data, larger reconstruction errors generally indicate increasing deviations from normal system behaviour. However, the reconstruction error cannot be directly interpreted as the degradation state of the component. Although the Transformer learns healthy behaviour under different operating conditions, the magnitude of the reconstruction error may still vary across operating points because the same level of degradation can produce different observable signal deviations under different pump speeds, thermal loads, or operating modes. Furthermore, measurement noise and transient operating conditions may introduce additional variations in the reconstruction error. Consequently, a post-processing framework is employed to transform the

reconstruction error into a robust and temporally consistent estimate of the underlying degradation state.

3.7.6.1 Reconstruction Error Normalization

The reconstruction error is first normalized using the statistical characteristics of the healthy validation dataset. The median reconstruction error is adopted as the reference healthy operating condition, while the corresponding standard deviation represents the natural variability of the healthy system. By expressing the reconstruction error relative to this healthy baseline, the influence of the absolute reconstruction error magnitude is reduced and the anomaly level becomes comparable across different operating conditions. Negative normalized values are clipped to zero to ensure that reconstruction errors within the expected healthy operating range do not contribute to the degradation estimation process.

3.7.6.2 Hidden Degradation State Estimation

The degradation evidence is subsequently treated as an observation of the underlying degradation process and processed using a Kalman Filter to estimate the hidden degradation state. The Kalman Filter recursively combines the current degradation observation with the previously estimated degradation state, thereby reducing the influence of short-term fluctuations while preserving the long-term degradation trend. This approach is motivated by the physical characteristic that component degradation evolves gradually over time, whereas reconstruction-error observations may vary due to operating-condition changes and measurement uncertainty.

Following the Kalman Filter update, a monotonic constraint is applied to ensure that the estimated degradation state does not decrease over time. This constraint reflects the assumption that component degradation is irreversible during the considered operating period and prevents temporary reductions in reconstruction error from being interpreted as physical recovery of the component. The resulting degradation state provides a temporally consistent representation of component health and serves as the basis for the subsequent Health Indicator (HI) calculation and Remaining Useful Life (RUL) estimation.

3.7.7 Health Indicator Estimation

The estimated hidden degradation state obtained from the degradation state estimation framework is converted into a Health Indicator (HI) to provide an intuitive representation of the component's health throughout its operational lifetime. The Health Indicator is defined as the complement of the estimated hidden degradation state,

$$HI(t) = 1 - \hat{x}(t), \tag{3.35}$$

where $HI(t)$ denotes the Health Indicator at time t, and $\hat{x}(t)$ represents the estimated hidden degradation state obtained from the Kalman Filter.

Under this definition, an HI value close to one corresponds to healthy operating conditions, whereas progressively lower values indicate increasing component degra-

dation. Since the Health Indicator is derived from the normalized reconstruction error, degradation evidence generation, Kalman filtering, and monotonic state estimation, it provides a smooth and physically consistent representation of the component degradation progression rather than reflecting instantaneous fluctuations in the reconstruction error.

The resulting Health Indicator serves as the degradation trajectory for continuously monitoring component health and forms the basis for the subsequent Remaining Useful Life (RUL) estimation.

3.7.8 Remaining Useful Life Estimation

The Remaining Useful Life (RUL) represents the estimated operating time remaining before the component reaches a predefined failure condition. Following the estimation of the hidden degradation state, the future degradation progression is predicted to determine the expected time at which the component reaches its end-of-life criterion.

Since the degradation state is continuously updated throughout operation, the degradation rate may vary over time. Consequently, using the complete degradation history for prediction may not accurately represent the current degradation behaviour. In this work, the linear trend is estimated using the last 30% of the degradation trajectory. This selection provides a better representation of the current degradation rate while reducing the influence of earlier degradation stages that no longer reflect the present degradation behaviour.

A first-order linear model is fitted to the selected degradation trajectory and is expressed as

$$\hat{x}(t) = mt + c, \tag{3.36}$$

where $\hat{x}(t)$ denotes the estimated degradation state at time t, m represents the degradation rate (slope), and c is the intercept.

A predefined degradation threshold, $\hat{x}_{\text{fail}}$, is adopted to represent the end-of-life condition of the component. The predicted failure time is determined by extrapolating the fitted degradation trend until it intersects the failure threshold,

$$t_{\text{fail}} = \frac{\hat{x}_{\text{fail}} - c}{m}, \tag{3.37}$$

where t_{fail} denotes the predicted failure time.

The Remaining Useful Life is then calculated as

$$RUL = t_{\text{fail}} - t_{\text{current}}, \tag{3.38}$$

where t_{current} represents the current operating time.

As new degradation observations become available, the degradation trajectory is continuously updated, allowing the degradation trend and the corresponding RUL prediction to be refined throughout the operational lifetime of the component.

3.8 Temporal Convolutional Network

3.8.1 Overall Framework

The Temporal Convolutional Network Autoencoder was developed as an independent workflow for monitoring the condition of selected BEV thermal-management components and estimating their Remaining Useful Life. The methodology focuses on learning the normal temporal behaviour of the ED coolant pump, HV battery coolant pump, radiator fan, and coolant-control valves from multivariate time-series data.

For each component, relevant command, response, thermal, and contextual variables were identified and prepared through component-specific preprocessing and feature engineering. The processed signals were divided into fixed-length temporal windows and used to train separate TCN Autoencoder models. Separate models were adopted because the monitored components differ in their operating characteristics, signal relationships, and degradation-sensitive behaviour.

During training, each TCN Autoencoder learned to reconstruct input sequences representing normal component operation. After training, the reconstruction error between the original and reconstructed sequences was used to quantify the deviation from the learned normal behaviour. This error was transformed into a Health Indicator, where lower values represented behaviour closer to the learned reference condition and higher values indicated a greater deviation.

The progression of the Health Indicator was then related to component condition and remaining-life fraction. For each evaluated input sequence, the current Health Indicator was mapped to the fitted degradation relationship to estimate the remaining-life fraction. The estimated Remaining Useful Life was subsequently obtained using the predicted life fraction and the selected reference lifetime of the component.

The complete TCN workflow therefore comprises component-specific signal preparation, feature engineering, temporal window construction, TCN Autoencoder training, Health Indicator construction and RUL calculation.

3.8.2 Data Source

The TCN-based methodology used the Volvo Cars expedition dataset described in Subsections 3.3 and 3.3.1, respectively. The selected measurements were used to characterise the operating behaviour of the ED coolant pump, HV battery coolant pump, radiator fan, and coolant-control valves under representative vehicle and thermal conditions.

For each component, the relevant command, response, thermal, and contextual variables were identified and grouped according to their functional relationship. The resulting multivariate time-series data formed the basis for preprocessing, feature preparation and TCN Autoencoder training.

3.8.3 Data Preprocessing

The raw expedition measurements were preprocessed before feature engineering and TCN Autoencoder training to ensure that the resulting temporal sequences were consistently ordered, physically plausible, and suitable for component-level modelling. The preprocessing procedure was applied independently to each expedition file and included signal selection, time-axis verification, physical-range validation, abrupt-event assessment, treatment of short invalid segments, and component-specific data extraction.

Session identifiers, vehicle information, and operating-scenario metadata were preserved throughout the procedure. This ensured that the source and operating context of each processed time series remained traceable during the later stages of model development and evaluation.

3.8.3.1 Signal Selection and Component Grouping

The original expedition files contained a large number of vehicle and thermal-system variables. Only signals relevant to the monitored components and their operating context were retained. These included:

- requested and actual pump speeds;
- requested and actual radiator-fan speeds;
- requested and measured valve positions;
- coolant-flow estimates;
- measured and model-estimated coolant temperatures;
- ambient temperature;
- vehicle speed;
- battery state of charge;
- charging type;
- thermal modes.

The selected variables were organised into four component-level datasets:

- Electric Drive coolant pump;
- High Voltage battery coolant pump;
- radiator fan;
- coolant-control valves.

This component-wise organisation reduced the dimensionality of the data and ensured that each dataset contained variables with a direct functional relationship to the monitored actuator. It also limited the influence of unrelated vehicle signals on the reconstruction task.

3.8.3.2 Time Ordering and Sampling-Interval Verification

Each expedition file was arranged in ascending order according to the recorded time variable. The interval between two consecutive observations was calculated as

$$\Delta t_i = t_i - t_{i-1}, \tag{3.39}$$

where t_i denotes the timestamp of the i-th observation.

The expedition measurements were expected to be sampled at 1 Hz. A sampling irregularity was therefore recorded when

$$|\Delta t_i - 1| > \varepsilon, \tag{3.40}$$

where ε is a small numerical tolerance introduced to account for floating-point representation.

This verification identified missing samples, irregular intervals, and temporal discontinuities within each measurement session. Correct temporal ordering is important for the TCN Autoencoder because temporal convolutions learn relationships from the sequence and spacing of observations. Incorrect ordering or unaccounted time gaps could introduce artificial dependencies into the input data.

Non-1-second intervals were recorded as session-level quality indicators. The data were not automatically resampled at this stage, allowing sessions with substantial timing irregularities to be reviewed before temporal-window construction.

3.8.3.3 Physical-Range Validation

The retained numerical signals were checked against predefined engineering bounds. An observation x_i was considered invalid when

$$x_i < x_{\min} \quad \text{or} \quad x_i > x_{\max}, \tag{3.41}$$

where $x_{\min}$ and $x_{\max}$ are the lower and upper admissible limits of the corresponding signal.

Table 3.3: Physical validation ranges used during preprocessing.

Signal group	**Admissible range**
Pump, fan, and valve request or response	0–100%
Coolant temperature	−40–70 °C
Ambient temperature	−50–70 °C
Coolant-flow estimate	0–100 recorded flow units

Values outside these limits were replaced with missing values rather than clipped to the nearest boundary. This avoided forcing invalid measurements into apparently admissible values. Short invalid intervals were subsequently treated using the interpolation procedure described in Subsection 3.8.3.7.

3.8.3.4 Rate-of-Change Analysis

The first-order difference of selected signals was calculated to identify abrupt temporal variations:

$$\Delta x_i = x_i - x_{i-1}, \tag{3.42}$$

where x_i and x_{i-1} are the current and preceding signal values.

A rate-of-change flag was assigned when

$$|\Delta x_i| > \tau_x, \tag{3.43}$$

where τ_x is the threshold associated with signal x.

The applied thresholds were:

- 20 percentage points per second for pump-speed, fan-speed, and valve-position request and actual signals;
- 3 °C per second for temperature signals;
- 20 recorded flow units per second for coolant-flow estimates.

These values were used as empirical screening thresholds rather than physical fault limits. The actuator threshold identified substantial changes relative to the 0–100% signal range, whereas the lower temperature threshold reflected the slower dynamics associated with thermal inertia.

Threshold exceedance was used only to identify candidate events. Flagged observations were subsequently evaluated using the request signal, tracking error, and neighbouring temporal behaviour before any correction was applied.

3.8.3.5 Request-to-Response Tracking Error

For each monitored actuator, the tracking error was calculated as

$$e_i = y_i - r_i, \tag{3.44}$$

where r_i is the requested value, y_i is the measured response, and e_i is the tracking error at time step i.

Tracking error was calculated for the ED coolant pump, HV battery coolant pump, radiator fan, and coolant-control valves Valve A, Valve B, and Valve C.

A tracking error close to zero indicates close agreement between the requested and measured values, whereas a positive or negative error indicates that the response is above or below the command, respectively. The tracking error was retained because component-performance deviations may appear as increasing request–response disagreement even when the individual signals remain within their admissible ranges.

3.8.3.6 Classification of Abrupt Events

Abrupt events identified through the rate-of-change analysis were classified as isolated spikes, actuator transitions, persistent mismatches, or unclear events. The purpose of this classification was to distinguish likely short-duration measurement anomalies from physically meaningful actuator behaviour using the requested value, measured response, tracking error, and neighbouring observations.

An isolated spike was defined as a single-sample excursion in the measured response that was not supported by a corresponding request change and returned close to its preceding level in the following sample. An actuator transition represented a substantial change in the requested value followed by movement of the measured response towards the new command. A persistent mismatch represented a command–response deviation that remained substantial across multiple consecutive samples. Events that did not satisfy any of these conditions were classified as unclear.

Only isolated spikes were treated as likely measurement irregularities and replaced with missing values for subsequent interpolation. Actuator transitions, persistent mismatches, and unclear events were retained because they could represent valid control-system behaviour, operating constraints, delayed response, or potential component-performance deviations.

The complete mathematical classification rules, threshold values, and observation intervals are provided in Appendix A.1.

3.8.3.7 Treatment of Invalid and Missing Measurements

Values identified as physically invalid or classified as isolated spikes were replaced with missing values. Short missing intervals were reconstructed using linear interpolation. For two valid observations x_a and x_b, the interpolated value at time t was calculated as

$$\hat{x}(t) = x_a + \frac{t - t_a}{t_b - t_a} \left(x_b - x_a \right), \tag{3.45}$$

where t_a and t_b are the time points corresponding to x_a and x_b, respectively.

For physical-range violations, interpolation was limited to a maximum of three consecutive samples. At the 1 Hz sampling rate, this represented a maximum interval of three seconds. Longer gaps were retained as missing values to avoid reconstructing intervals that could contain genuine changes in actuator or thermal behaviour.

Samples classified as isolated spikes were treated more restrictively. Since an isolated spike was defined as a single-sample excursion, interpolation was applied only to the identified sample. Actuator transitions, persistent mismatches, and unclear events were retained without automatic correction.

Missing values at the beginning or end of a session were not extrapolated because valid observations were not available on both sides. Windows containing unresolved missing values were reviewed before model training.

3.8.3.8 Component-Specific Output Preparation

Following preprocessing, each expedition session was saved in complete and component-specific forms. The component files retained the relevant request and response signals, tracking errors, first-order differences, event classifications, thermal and vehicle-context variables, diagnostic counters, and metadata required for traceability.

Event classifications and supporting indicators were retained primarily for data-quality review and were not necessarily used as numerical model inputs. The component-specific files were subsequently used to prepare independent datasets for the ED coolant pump, HV battery coolant pump, radiator fan, and coolant-control valves.

3.8.4 Feature Engineering

Following preprocessing, the component-specific measurements were divided into fixed-duration windows and transformed into features describing actuator response, thermal and hydraulic behaviour, diagnostic activity, and operating context. The generated feature tables supported dataset partitioning and provided selected contextual variables to the TCN Autoencoder.

3.8.4.1 Window Generation and Activity Filtering

Each measurement session was divided into non-overlapping windows of 300 samples. At the sampling rate of 1 Hz, each window represented five minutes of operation. This duration was selected to capture both short-term actuator responses and slower thermal variations, while non-overlapping segmentation avoided repeated use of the same measurements across adjacent windows.

Only complete windows were retained. Pump and fan windows were included when the corresponding component was active for at least 50% of the samples. For the valve dataset, a window was retained when at least one valve was commanded above zero for at least 10% of the samples. Windows associated with an uninitialised thermal mode were excluded.

These criteria ensured that the retained windows contained sufficient component activity for meaningful feature extraction.

3.8.4.2 Component and Context Features

For each retained window, command–response features were calculated from the tracking-error signal. These included the mean, standard deviation, Root Mean Square (RMS), maximum absolute value, normalised error, and rolling statistics over 60- and 300-sample intervals.

The tracking-error RMS was calculated as

$$e_{\mathrm{RMS}} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} e_t^2}, \tag{3.46}$$

where e_t is the tracking error at time step t, and T is the number of samples in the window.

Component-specific features included sensor-to-model temperature differences, inlet-to-outlet temperature differences, coolant-flow statistics, valve hold stability, fault-counter statistics, and counts of actuator transitions, persistent mismatches, and isolated spikes.

The operating-context features included mean ambient temperature, seasonal operating condition, mean vehicle speed, battery state of charge, thermal mode, charging type, and drivetrain configuration.

These engineered features complemented the raw time-series signals by expressing relationships that were not directly represented by individual measurements. For example, tracking error described request–response agreement, while temperature differences and flow statistics represented thermal and hydraulic behaviour. Contextual variables helped distinguish changes caused by normal operating conditions from possible component-related deviations. The complete component-specific feature definitions are provided in Appendix A.2.

3.8.4.3 Dataset Partitioning

The generated windows were divided into training, validation, and test sets using a 70%/15%/15% ratio. The split was performed at the session level to prevent windows from the same expedition session from appearing in more than one subset.

3.8.4.4 Input Normalisation

Before training, all time-series signals and contextual variables were normalised using z-score standardisation. To reduce seasonal operating differences,the time-series signals were first normalised within ambient-temperature groups representing cold, mild, warm, and hot conditions. Contextual variables were standardised separately using their healthy-training statistics.

3.8.5 TCN Autoencoder Architecture

A separate TCN Autoencoder was developed for each monitored component. The encoder consisted of seven residual TCN blocks, with two causal dilated convolutional layers in each block. The dilation factors increased from 1 to 64, while a kernel size of 3 and 64 feature channels were used throughout the encoder.

The output of the final TCN block was reduced using global average pooling and projected into a 32-dimensional latent representation. This latent vector was concatenated with the selected contextual variables before being passed to the decoder.

The decoder reconstructed the original multivariate input sequence without receiving the complete encoder feature map, thereby maintaining a strict bottleneck.

3.8.5.1 Reconstruction Error and Health Indicator

The encoder and decoder were trained jointly by minimising the Mean Squared Error between the original and reconstructed sequences:

$$\mathcal{L}_{\text{rec}} = \frac{1}{NTF} \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{f=1}^{F} \left(X_{n,t,f} - \hat{X}_{n,t,f} \right)^2, \tag{3.47}$$

where $\mathcal{L}_{\text{rec}}$ denotes the batch-averaged reconstruction loss minimised during model training, N is the number of windows in the training batch, T is the number of time steps, and F is the number of input features.

For an individual window n, the reconstruction error was used as the Health Indicator:

$$HI_n = \frac{1}{TF} \sum_{t=1}^{T} \sum_{f=1}^{F} \left(X_{n,t,f} - \hat{X}_{n,t,f} \right)^2. \tag{3.48}$$

A lower HI_n indicates close agreement between the input and its reconstruction, whereas a higher value indicates greater deviation from the temporal behaviour learned from the healthy training data. The Health Indicator was interpreted as a measure of deviation from learned normal behaviour rather than direct evidence of physical degradation.

3.8.5.2 Model Configuration and Training Procedure

A separate TCN Autoencoder was trained for the EDrive coolant pump, HV battery coolant pump, radiator fan, and coolant-control valves. The same general architecture and training settings were used for all components, while the number of input signals and contextual variables depended on the corresponding component dataset.

The encoder consisted of seven residual TCN blocks. The dilation factor of block l was defined as

$$d_l = 2^l, \qquad l \in \{0, 1, \ldots, 6\}, \tag{3.49}$$

resulting in dilation values of $1, 2, 4, 8, 16, 32$, and 64. Each block contained two causal convolutional layers with a kernel size of 3 and 64 feature channels.

For two convolutional layers per residual block, the receptive field was calculated as

$$R = 1 + 2(K - 1) \sum_{l=0}^{L-1} d_l, \tag{3.50}$$

where K is the kernel size, L is the number of residual blocks, and d_l is the dilation factor of block l. Using $K = 3$ and $L = 7$ resulted in a receptive field of 509 samples. Since this exceeded the 300-sample input-window length, the encoder could access information across the complete five-minute sequence.

Batch normalisation and Rectified Linear Unit activation were applied within the residual blocks. The final temporal feature map was reduced using global average pooling and projected into a 32-dimensional latent representation, which was combined with the selected contextual variables before decoding.

The models were trained using the Adam optimiser with a learning rate of 10^{-3}, a batch size of 32, and mean squared reconstruction error as the loss function. Training was limited to 100 epochs. Early stopping with a patience of 10 epochs was applied based on the validation reconstruction loss, and the model state with the lowest validation loss was retained.

The selected settings were treated as dataset-specific design choices intended to balance temporal coverage, representation capacity, training stability, and computational cost.

3.8.6 Remaining Useful Life Estimation

Since the expedition dataset primarily represented healthy operation, controlled degraded data were generated solely to evaluate the response of the TCN Autoencoder and the reconstruction-based Health Indicator. The degradation severity was defined using the power-law relationship $q = (1 - \alpha)^{\beta}$. In the present implementation, $\beta = 1$ was selected, resulting in a linear progression of the imposed degradation severity across the evaluated stages. The calculated degradation level was then used to introduce controlled and physically plausible changes into the healthy component signals. These degraded conditions were used only for methodological validation and were not intended to reproduce the exact physical degradation behaviour of the components.

The reconstruction error calculated for each temporal window was used as the Health Indicator (HI). The controlled degradation stages were introduced to evaluate whether the HI increased consistently as the component condition worsened.

Four remaining-life fractions were evaluated:

$$\alpha \in \{1.00,\ 0.75,\ 0.50,\ 0.25\}, \tag{3.51}$$

where $\alpha = 1.00$ represents the healthy condition, $\alpha = 0.75$ represents an early degradation stage, $\alpha = 0.50$ represents an intermediate degradation stage, and $\alpha = 0.25$ represents a severe degradation stage. The value $\alpha = 0$ denotes the theoretical end-of-life condition but was not included as one of the evaluated stages. The corresponding consumed-life fraction was calculated as

$$d = 1 - \alpha. \tag{3.52}$$

The normalised degradation severity applied at each stage was determined using

$$q = (1 - \alpha)^{\beta}, \tag{3.53}$$

where q denotes the normalised degradation level and β controls the shape of the degradation progression. Since $\beta = 1$ was used, the imposed degradation severity increased linearly with the consumed-life fraction.

The healthy and controlled degraded windows were subsequently passed through the trained TCN Autoencoder. For each remaining-life fraction, the reconstruction errors were summarised to obtain a representative stage-wise HI. This resulted in known pairs of remaining-life fraction and Health Indicator:

$$(\alpha_i, HI_i), \qquad i = 1, \ldots, 4. \tag{3.54}$$

These stage-wise values were used to construct the relationship between the HI and the remaining-life fraction. For an evaluated window, the reconstruction-based Health Indicator HI_{new} was calculated and mapped to the fitted degradation relationship to estimate the corresponding remaining-life fraction $\hat{\alpha}$. The estimated RUL was then obtained using the relationship introduced in Section 2.3.6.

Figure 3.1 provides a schematic representation of the HI-to-remaining-life mapping procedure. It is included only to support conceptual interpretation of the methodology and does not present the actual numerical results of any evaluated component.

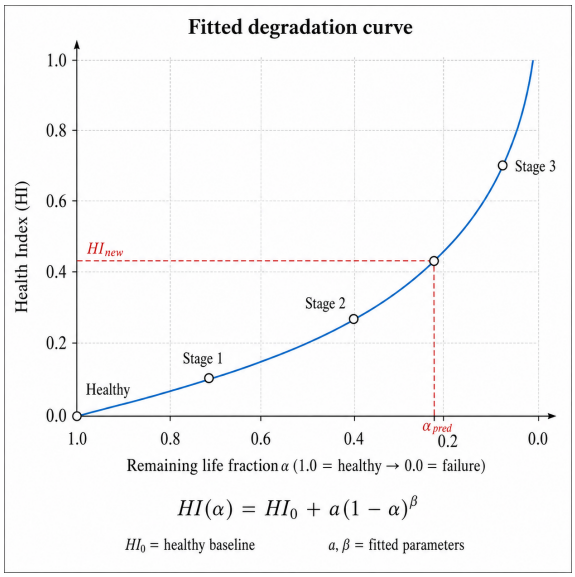

Figure 3.1: Conceptual illustration of the HI-to-remaining-life mapping used for RUL estimation. For a new Health Indicator, HI_{new}, the corresponding remaining-life fraction $\hat{\alpha}$ is obtained from the fitted degradation relationship and subsequently converted into RUL. The figure is schematic and does not represent the actual component-specific numerical results.

Since the assigned remaining-life fractions were known, they were used as reference

values to evaluate the estimated remaining-life fractions and the resulting RUL values. The component-wise comparison results are presented in Section 4.2.1.

The overall procedure was used as a proof-of-concept under an assumed linear degradation progression. Since real component degradation may be nonlinear and component-specific, the estimated RUL values should be interpreted as a methodological validation of the proposed framework rather than as direct estimates of the physical remaining lifetime of the evaluated components.

4

Results

4.1 Transformer Autoencoder

4.1.1 Pre-Processing:

Figure 4.1 illustrates the output of the proposed preprocessing framework applied to a real measurement signal from the BEV thermal management system. The first subplot compares the original measurement (Raw), the adaptive Kalman-filtered signal (Smooth), and the reconstructed signal (Recon). Under normal operating conditions, the reconstructed signal closely follows the overall thermal behaviour while reducing abrupt variations that are inconsistent with the surrounding measurements.

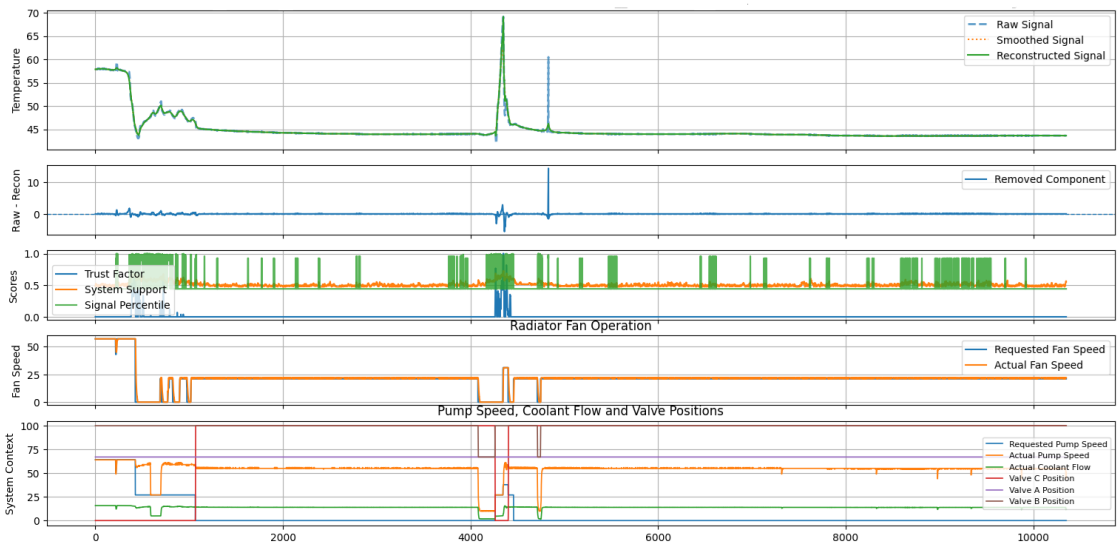

Figure 4.1: Pre-Processing of Raw Dataset.

To evaluate the effectiveness of the preprocessing framework, two artificial measurement errors were manually introduced at approximately sample 4500. The disturbance near sample 4500 was inserted during a period in which several related thermal management signals were also changing. This scenario was designed to assess whether the proposed method could distinguish an artificial measurement anomaly from a genuine system transition occurring simultaneously.

The second subplot shows the difference between the original and reconstructed signals. The difference remains close to zero for most of the measurement sequence and increases primarily around the manually introduced disturbances. This indicates that the preprocessing framework preserves the original measurements when they are consistent with the overall system behaviour, while selectively correcting abnormal measurements.

The third subplot presents the intermediate quantities used during the reconstruction process, including the percentile score, peer-support score, and trust factor. The peer-support score evaluates whether a local signal variation is supported by the behaviour of other related thermal management variables. This information is used to determine the trust factor, which controls the weighting between the original measurement and the adaptive Kalman-filter estimate during signal reconstruction. As a result, the reconstructed signal remains smooth and physically consistent while adapting to the operating conditions of the system.

The final subplot shows several contextual signals from the thermal management system, including coolant flow requests, pump speeds, coolant flow measurements, and valve positions. These signals provide system-level information that supports the evaluation of whether an observed signal variation represents a genuine operating event or an isolated measurement anomaly. Around sample 4500, multiple contextual signals exhibit simultaneous changes, creating a mixed operating scenario containing both normal system dynamics and an introduced measurement error. Here, the reconstructed signal successfully suppresses the introduced anomalies while preserving the underlying thermal behaviour.

Overall, these results demonstrate that the proposed preprocessing framework effectively combines local signal analysis with system-level contextual information to distinguish genuine operating changes from unreliable measurements. Consequently, the reconstructed signals provide a cleaner and more reliable dataset for the subsequent feature engineering and Transformer Autoencoder model development.

4.1.2 RUL Estimation:

Figure 4.3 presents the Remaining Useful Life (RUL) estimation for the aging dataset. The 4.2 Plot shows the hidden degradation state estimated by the Kalman Filter together with the fitted degradation trend and the predefined failure threshold, while the 4.3 plot illustrates the corresponding RUL estimate over time.

The hidden degradation state increases progressively as the simulated component degrades. A linear trend was fitted to the final 30% of the degradation trajectory and extrapolated to the predefined failure threshold. The predicted intersection between the degradation trend and the threshold was then used to estimate the remaining useful life.

At the beginning of the degradation process, the estimated RUL is approximately 7.3 hours and gradually decreases as the degradation state approaches the predefined threshold. Once the predicted failure point is reached, the estimated RUL becomes zero, indicating that the defined degradation limit has been reached.

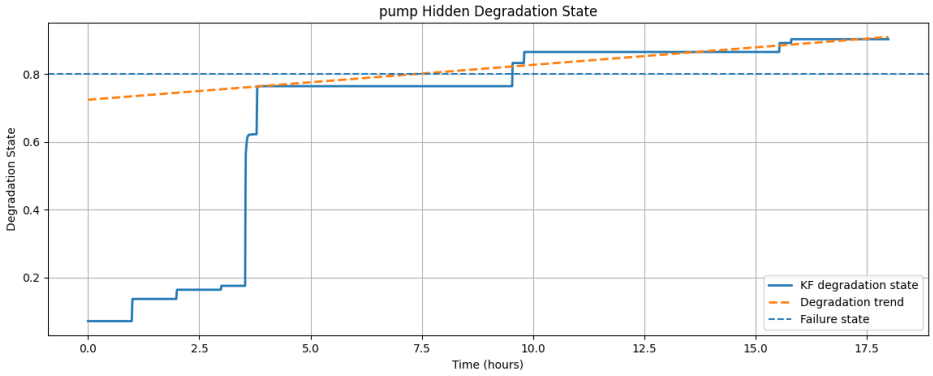

Figure 4.2: Degradation State of Pump .

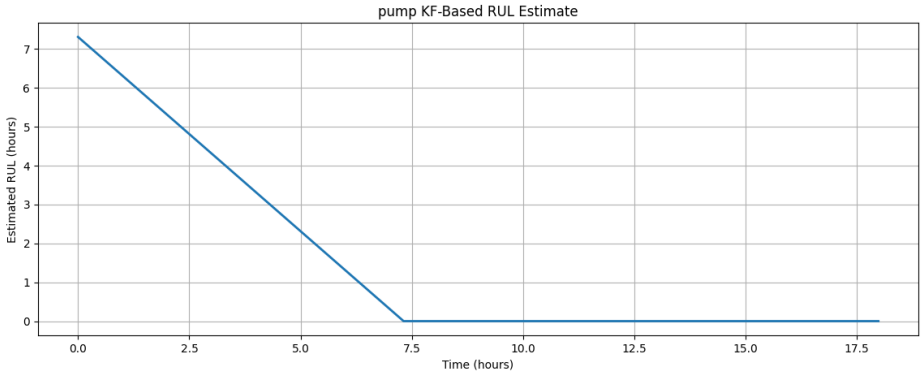

Figure 4.3: RUL estimation of Pump.

It should be noted that the failure threshold of 0.8 was selected as a fixed reference value to evaluate the proposed prognostic framework. This threshold does not represent a manufacturer-defined end-of-life criterion or an experimentally validated failure limit. Since the degradation data were synthetically generated to emulate the behaviour of Battery Electric Vehicle (BEV) thermal management components, the reported RUL values demonstrate the capability of the proposed methodology rather than the actual remaining lifetime of a real component.

4.2 TCN Autoencoder

4.2.1 Component-wise Health Indicator and RUL Estimation

The component-wise Health Indicator (HI) and Remaining Useful Life (RUL) estimates were obtained using the procedure described in Section 3.8.6. The resulting degradation curves for the Electric Drive (ED) coolant pump, High Voltage (HV) battery coolant pump, radiator fan, and coolant-control valves (Valve A and Valve B) are presented in Figures 4.4, 4.5, 4.6, and 4.7. The consolidated performance metrics are summarised in Table 4.1, while the complete stage-wise HI, true and

predicted α, and RUL values are provided in Appendix B. The mathematical definitions and interpretation of the evaluation metrics (R^2, MAE, and RMSE) are provided in Appendix C.

4.2.1.1 Electric Drive (ED) Coolant Pump

The ED coolant pump exhibited a clear monotonic increase in HI with increasing degradation severity, indicating that the reconstruction-based Health Indicator consistently reflected the imposed degradation trend. The fitted degradation curve achieved an R^2 of 0.893. At Stage 3, the predicted remaining-life fraction (α) was 0.300 compared with the true value of 0.250, resulting in an estimated RUL of 17,266 hours.

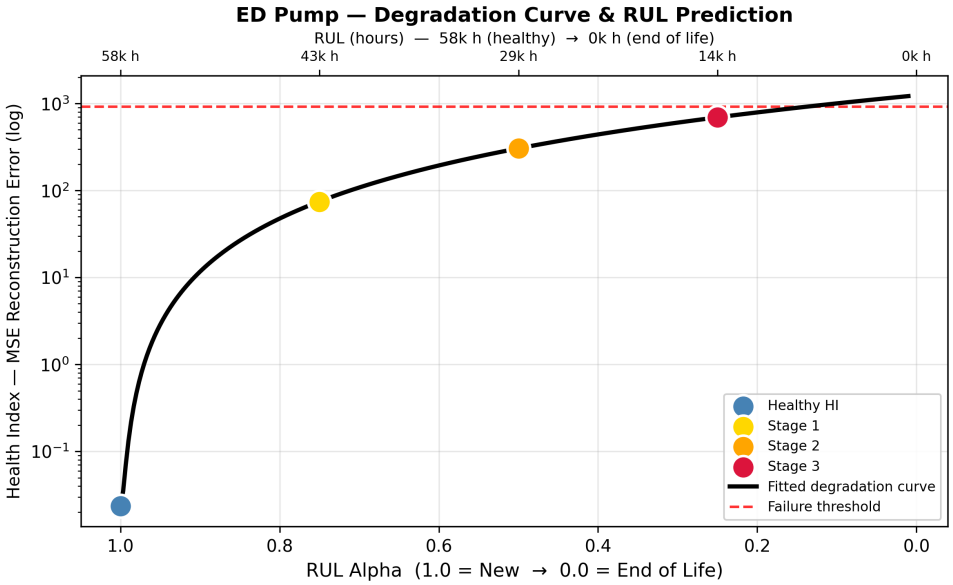

Figure 4.4: Degradation curve and RUL prediction for the Electric Drive coolant pump.

4.2.1.2 High Voltage (HV) Battery Coolant Pump

The HV battery coolant pump also showed a consistent increase in HI with degradation severity. The fitted degradation curve achieved an R^2 of 0.707, indicating a moderate agreement between the fitted model and the stage-wise Health Indicator values. At Stage 3, the predicted α was 0.275 compared with the true value of 0.250, corresponding to an estimated RUL of 15,836 hours.

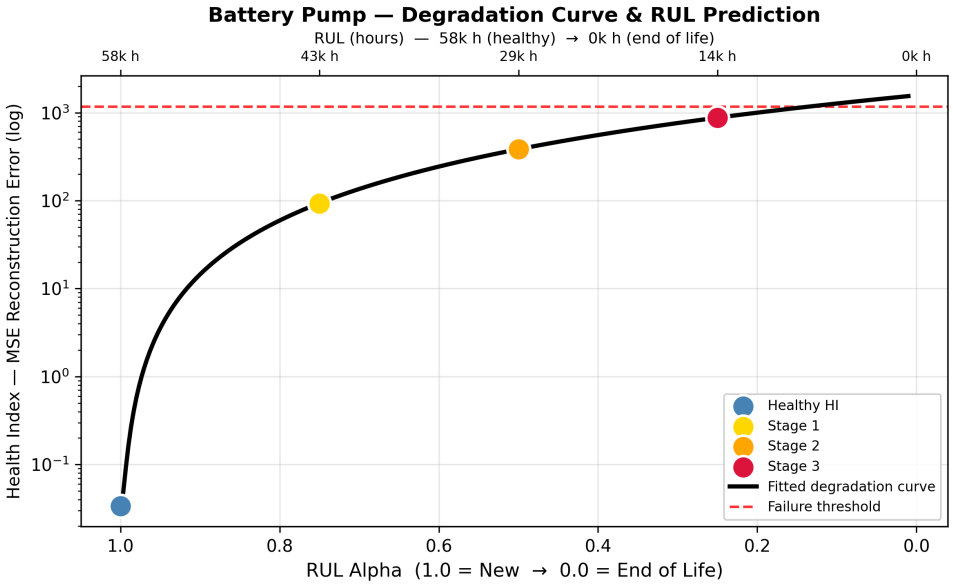

Figure 4.5: Degradation curve and RUL prediction for the High Voltage battery coolant pump.

4.2.1.3 Radiator Fan

The radiator fan produced the most consistent degradation trend and achieved the best overall performance, with an R^2 of 0.954 and the lowest prediction errors among the four components. At Stage 3, the predicted α of 0.247 closely matched the true value of 0.250, resulting in an estimated RUL of 14,203 hours.

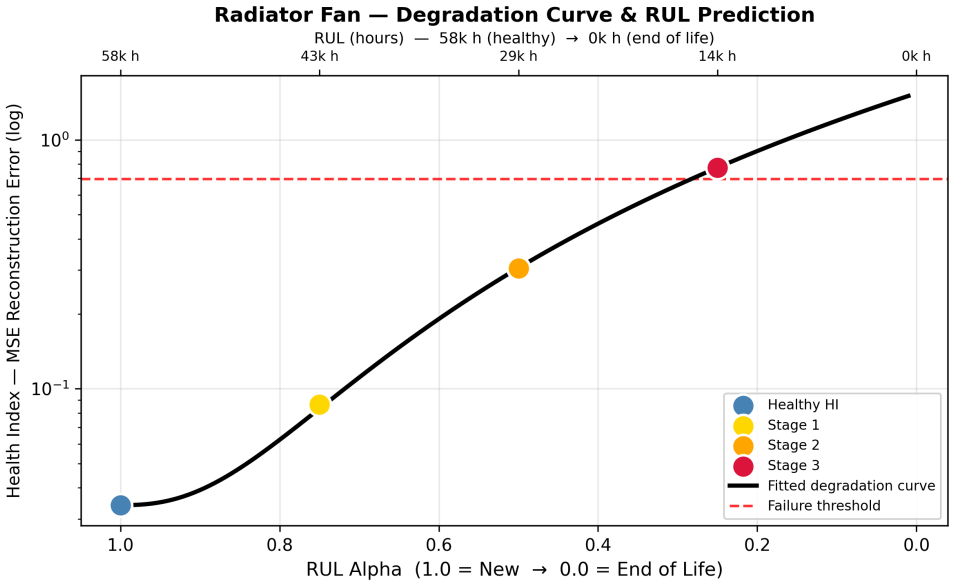

Figure 4.6: Degradation curve and RUL prediction for the radiator fan.

4.2.1.4 Coolant-Control Valves (Valve A and Valve B)

The coolant-control valves exhibited a monotonic increase in HI but achieved the lowest goodness of fit, with an R^2 of 0.601, indicating greater variability in the fitted degradation relationship. At Stage 3, the predicted α of 0.382 overestimated the true value of 0.250, resulting in an estimated RUL of 21,960 hours.

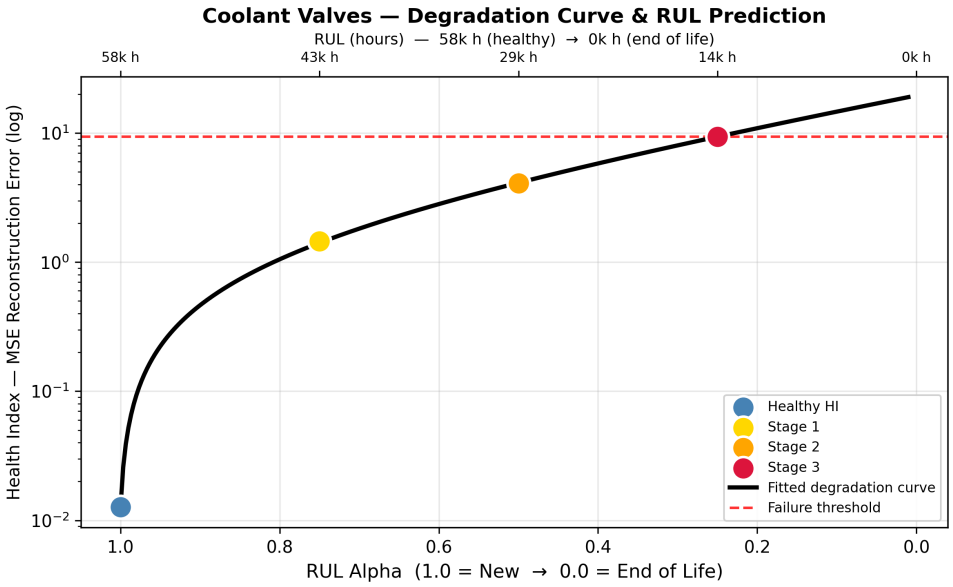

Figure 4.7: Degradation curve and RUL prediction for the coolant-control valves.

4.2.1.5 Overall Comparison

All four components exhibited an increasing HI as the degradation severity progressed, confirming that the TCN Autoencoder was sensitive to deviations from the learned healthy behaviour. However, the absolute HI values differed considerably between components because each model reconstructed a different set of component-specific signals. Therefore, the raw HI magnitudes should not be directly compared across components. Instead, the goodness of fit of the degradation curve and the errors in the predicted remaining-life fraction provide more meaningful measures for comparison.

Among the four components, the radiator fan achieved the strongest overall performance, with the highest R^2 and the lowest prediction errors. The ED coolant pump also demonstrated a strong and well-defined degradation trend. The HV battery coolant pump captured the overall progression but exhibited moderate variability, whereas the coolant-control valves showed the greatest uncertainty and consistently overestimated the remaining life.

Overall, the results demonstrate the feasibility of combining reconstruction-based Health Indicators with power-law degradation modelling for Remaining Useful Life estimation. At the same time, they indicate that prediction accuracy depends on the operating characteristics, signal behaviour, and degradation complexity of the individual component. A quantitative interpretation of the reported R^2, MAE, and

RMSE values is provided in Appendix C.

Table 4.1: Consolidated performance of the component-wise degradation models.

Component	**Healthy HI**	**Failure HI**	R^2	$\mathbf{MAE}_\alpha$	$\mathbf{RMSE}_\alpha$
ED Coolant Pump	0.023607	924.759842	0.893	0.1114	0.1517
HV Battery Coolant Pump	0.033734	1168.226187	0.707	0.0873	0.1219
Radiator Fan	0.034081	0.694730	0.954	0.0635	0.0834
Coolant-Control Valves	0.012680	9.380123	0.601	0.1751	0.2230

5

Conclusion

This thesis investigated data-driven health monitoring and Remaining Useful Life (RUL) estimation of Battery Electric Vehicle (BEV) thermal management components using multivariate operational data. Two independent reconstruction-based approaches were developed and evaluated: a Transformer Autoencoder-based framework and a Temporal Convolutional Network (TCN) Autoencoder-based framework.

5.1 Transformer Autoencoder

A complete workflow was developed, including data preprocessing, feature engineering, sliding window generation, Transformer Autoencoder modelling, Health Indicator (HI) estimation, and RUL prediction.

The proposed preprocessing pipeline improved the quality of the operational data while preserving the dynamic behaviour of the thermal management system. The Transformer Autoencoder successfully learned the normal operating behaviour through sequence reconstruction, and the resulting reconstruction error was used to estimate a smooth Health Indicator using a Kalman Filter. The degradation trajectory was subsequently extrapolated to estimate the Remaining Useful Life, demonstrating how reconstruction-based anomaly detection can be extended towards prognostic health monitoring.

However, the available dataset contained only healthy operational data and did not include real degradation or failure information. Therefore, synthetic degradation scenarios were generated to evaluate the proposed HI and RUL estimation framework. Consequently, the obtained Health Indicator and RUL predictions should be considered as a proof of concept demonstrating the proposed methodology rather than representing the actual lifetime of production components.

Overall, the proposed framework demonstrates the potential of combining Transformer-based unsupervised learning with degradation state estimation for predictive maintenance of BEV thermal management systems. Future work should focus on validating the framework using real degradation data and manufacturer-defined end-of-life criteria to improve its applicability in industrial environments.

5.2 Temporal Convolutional Network (TCN) Autoencoder

The proposed framework comprised data preprocessing, feature engineering, input normalisation, TCN Autoencoder modelling, reconstruction-based Health Indicator (HI) generation, and power-law-based Remaining Useful Life (RUL) estimation.

The TCN Autoencoder successfully learned the normal temporal behaviour of the thermal management components, and the resulting reconstruction error provided an effective Health Indicator that increased consistently with the imposed degradation stages. By relating the Health Indicator to controlled degradation levels through a power-law progression, the proposed framework demonstrated the feasibility of estimating the remaining-life fraction and the corresponding Remaining Useful Life.

Since the available dataset represented predominantly healthy operation, synthetic degradation stages were introduced solely for methodological validation. Consequently, the reported Health Indicator and RUL estimates should be interpreted as a proof of concept demonstrating the capability of the proposed framework rather than as direct predictions of the physical remaining lifetime of the components.

Overall, the proposed TCN Autoencoder framework demonstrates the potential of reconstruction-based temporal modelling for predictive maintenance of BEV thermal management systems.

5.2.1 Future Work

Future work should focus on validating the proposed framework using real run-to-failure data collected from thermal management components operating under practical vehicle conditions. Such datasets would enable the Health Indicator and Remaining Useful Life estimates to be evaluated against actual degradation behaviour rather than controlled synthetic degradation stages.

The current work employed a linear power-law progression as a proof-of-concept to evaluate the proposed methodology. Future studies should investigate component-specific degradation models that more closely represent the physical ageing mechanisms of coolant pumps, radiator fans, and coolant-control valves under different operating environments.

Further improvements may also be achieved by exploring alternative Temporal Convolutional Network architectures, optimising model hyperparameters, and integrating uncertainty quantification into the Remaining Useful Life estimation process. In addition, defining application-specific end-of-life criteria based on industrial requirements would improve the practical applicability of the framework in condition-based maintenance and prognostic health management of Battery Electric Vehicle thermal management systems.

Bibliography

[1] X. Xia and P. Li, "A Review of the Life Cycle Assessment of Electric Vehicles: Considering the Influence of Batteries," *Science of the Total Environment*, vol. 814, p. 152870, Mar. 2022. [Online]. Available: https://doi.org/10.1016/j.scitotenv.2021.152870. [Accessed: 12-Feb-2026]

[2] D. Dan, Y. Zhao, M. Wei, and X. Wang, "Review of Thermal Management Technology for Electric Vehicles," *Energies*, vol. 16, no. 12, p. 4693, Jun. 2023. [Online]. Available: https://doi.org/10.3390/en16124693. [Accessed: 12-Feb-2026]

[3] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery Health Prognostics: A Systematic Review from Data Acquisition to RUL Prediction," *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–834, May 2018. [Online]. Available: https://doi.org/10.1016/j.ymssp.2017.11.016. [Accessed: 21-Jul-2026]

[4] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *arXiv preprint arXiv:1803.01271*, Mar. 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1803.01271. [Accessed: 24-Jul-2026]

[5] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006. [Online]. Available: https://doi.org/10.1126/science.1127647. [Accessed: 24-Jul-2026]

[6] M. Thill, W. Konen, H. Wang, and T. Bäck, "Temporal Convolutional Autoencoder for Unsupervised Anomaly Detection in Time Series," *Applied Soft Computing*, vol. 112, p. 107751, Nov. 2021. [Online]. Available: https://doi.org/10.1016/j.asoc.2021.107751. [Accessed: 24-Jul-2026]

[7] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," *arXiv preprint arXiv:1609.03499*, Sep. 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1609.03499. [Accessed: 24-Jul-2026]

[8] K. Sohn, H. Lee, and X. Yan, "Learning Structured Output Representation Using Deep Conditional Generative Models," in *Advances in*

Neural Information Processing Systems, vol. 28, pp. 3483–3491, 2015. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/8d55a249e6baa5c06772297520da2051-Abstract.html. [Accessed: 24-Jul-2026]

[9] J. Lu, X. Yang, Y. Liu, H. Zuo, F. Zhou, T. Yu, D. Liu, T. Deng, and L. Luo, "Transformer-Autoencoder-Based Unsupervised Temporal Anomaly Detection for Network Traffic with Dual Prediction and Reconstruction," *Applied Sciences*, vol. 16, no. 4, p. 2143, 2026. doi: https://doi.org/10.3390/app16042143.

[10] Y. Pan, S. Kang, L. Kong, J. Wu, Y. Yang, and H. Zuo, "Remaining Useful Life Prediction Methods of Equipment Components Based on Deep Learning for Sustainable Manufacturing: A Literature Review," *AI EDAM*, vol. 39, p. e4, Feb. 2025. [Online]. Available: https://doi.org/10.1017/S0890060424000271. [Accessed: 29-Jul-2026]

[11] S. Fu, X. Gao, B. Li, F. Zhai, J. Lu, B. Xue, J. Yu, and C. Xiao, "Multivariate time series anomaly detection via separation, decomposition, and dual transformer-based autoencoder," *Applied Soft Computing*, vol. 159, p. 111671, 2024. doi: https://doi.org/10.1016/j.asoc.2024.111671.

[12] S. Tuli, G. Casale, and N. R. Jennings, "TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data," *Proceedings of the VLDB Endowment*, vol. 15, no. 6, pp. 1201–1214, 2022. doi: https://doi.org/10.14778/3514061.3514067.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017. https://arxiv.org/abs/1706.03762.

[14] M. Sakurada and T. Yairi, "A Neural Network-Based Anomaly Detector Using a Reconstruction Error for Deep Learning," in *Proceedings of the 2nd International Conference on Machine Learning and Applications Workshops (ICMLA-W)*, 2014, pp. 13–17. doi: https://doi.org/10.1109/ICMLA.2014.110.

[15] H. Shiri and M. Belal, "A real-time framework for mapping subsea cable burial state using Poincaré spectral coherence of DAS measurements," *Scientific Reports*, vol. 16, Art. no. 23647, 2026. doi: https://doi.org/10.1038/s41598-026-59846-4.https://www.nature.com/articles/s41598-026-59846-4

[16] A. Chhabra, J. R. Venepally, and D. Kim, "Measurement Noise Covariance-Adapting Kalman Filters for Varying Sensor Noise Situations," *Sensors*, vol. 21, no. 24, p. 8304, 2021. doi: https://doi.org/10.3390/s21248304.

A

Supplementary Methodological Details and Results of TCN

A.1 Detailed Abrupt-Event Classification Rules

A.1.1 Isolated-Spike Classification

An event at time step i was classified as an isolated spike when

$$|y_i - y_{i-1}| > \tau_y, \tag{A.1}$$

$$|r_i - r_{i-1}| < \delta_r, \tag{A.2}$$

$$|y_{i+1} - y_{i-1}| < \delta_y, \tag{A.3}$$

and

$$|y_i - r_i| > \delta_e. \tag{A.4}$$

Here, r_i denotes the requested actuator value, y_i denotes the measured response, and y_{i-1} and y_{i+1} denote the neighbouring measured values. The selected thresholds were

$$\tau_y = 20, \qquad \delta_r = 5, \qquad \delta_y = 5, \qquad \delta_e = 10 \tag{A.5}$$

percentage points.

These conditions identify a substantial single-sample excursion that is unsupported by the request signal, produces a large instantaneous tracking error, and returns close to the pre-event response level in the following sample.

A.1.2 Actuator-Transition Classification

An event was classified as an actuator transition when

$$|r_i - r_{i-1}| > \tau_r, \tag{A.6}$$

and

$$\min_{k \in \{i+1, i+2, i+3\}} |e_k| < |e_i|\,, \tag{A.7}$$

where

$$e_i = y_i - r_i. \tag{A.8}$$

The request threshold was set to

$$\tau_r = 20 \tag{A.9}$$

percentage points per second. The second condition required the measured response to move closer to the new request during at least one of the following three samples.

A.1.3 Persistent-Mismatch Classification

An event was classified as a persistent mismatch when

$$\sum_{k=i+1}^{i+3} \mathbb{1}\left(|e_k| > \delta_p\right) \geq 2, \tag{A.10}$$

where $\mathbb{1}(\cdot)$ is the indicator function and

$$e_k = y_k - r_k. \tag{A.11}$$

The persistent-mismatch threshold was set to

$$\delta_p = 15 \tag{A.12}$$

percentage points. The rule therefore required the tracking error to exceed the threshold in at least two of the following three samples.

A.1.4 Unclear-Event Classification

Events that did not satisfy the isolated-spike, actuator-transition, or persistent-mismatch criteria were classified as unclear. These events were retained because the available temporal information was insufficient to determine their cause reliably.

The thresholds and observation intervals were used as empirical screening parameters for the present dataset and should not be interpreted as universal component fault limits. The classification identifies temporal patterns consistent with the defined rules but does not establish the physical cause of an event.

A.2 Component-Specific Engineered Features

The engineered features were grouped into command–response, thermal, hydraulic, diagnostic, and operating-context categories. The main feature groups used for the evaluated components are summarised in Table A.1.

Table A.1: Summary of engineered feature groups.

Feature group	Examples	Purpose
Command–response features	Tracking-error mean, standard deviation, RMS, maximum absolute error, and normalised error	Describe the agreement between requested and actual actuator behaviour.
Temporal-variability features	Rolling mean and rolling standard deviation over 60- and 300-sample intervals	Capture short-term and window-level variation.
Thermal features	Sensor-to-model temperature difference and inlet-to-outlet temperature difference	Describe thermal response and temperature-model disagreement.
Hydraulic features	Coolant-flow mean, coolant-flow standard deviation, and flow balance	Represent coolant-circuit operating behaviour.
Valve-specific features	Valve hold stability and active fraction	Describe the ability of a valve to maintain its commanded position and its activity level.
Event features	Counts of actuator transitions, persistent mismatches, and isolated spikes	Summarise dynamic and irregular behaviour within each window.
Diagnostic features	Fault-counter mean and variance	Represent diagnostic activity during the window.
Operating-context features	Ambient temperature, vehicle speed, state of charge, charging type, and drivetrain configuration	Account for changes in normal behaviour caused by operating conditions.

B

Detailed Component-wise Degradation Results

This section presents the detailed stage-wise Health Indicator, true and predicted remaining-life fraction, and estimated Remaining Useful Life values used in the component-wise analysis presented in Section 4.2.1.

B.0.1 ED Coolant Pump Stage-wise Results

Table B.1: Stage-wise degradation and RUL results for CPED.

Stage	HI (MSE)	True α	Predicted α	RUL (hours)
Healthy	0.0236	1.000	1.000	57,500
Stage 1	74.2322	0.750	0.738	42,440
Stage 2	305.6109	0.500	0.526	30,268
Stage 3	693.8735	0.250	0.300	17,266

B.0.2 HV Battery Coolant Pump Stage-wise Results

Table B.2: Stage-wise degradation and RUL results for BCPM.

Stage	HI (MSE)	True α	Predicted α	RUL (hours)
Healthy	0.0337	1.000	1.000	57,500
Stage 1	92.4784	0.750	0.749	43,092
Stage 2	384.9041	0.500	0.529	30,443
Stage 3	876.6600	0.250	0.275	15,836

B.0.3 Radiator Fan Stage-wise Results

Table B.3: Stage-wise degradation and RUL results for the radiator fan.

Stage	HI (MSE)	True α	Predicted α	RUL (hours)
Healthy	0.0341	1.000	1.000	57,500
Stage 1	0.0864	0.750	0.758	43,602
Stage 2	0.3042	0.500	0.500	28,740
Stage 3	0.7734	0.250	0.247	14,203

B.0.4 Coolant-Control Valve Stage-wise Results

Table B.4: Stage-wise degradation and RUL results for the coolant-control valve.

Stage	HI (MSE)	True α	Predicted α	RUL (hours)
Healthy	0.0127	1.000	1.000	57,500
Stage 1	1.4486	0.750	0.838	48,199
Stage 2	4.0976	0.500	0.589	33,847
Stage 3	9.3801	0.250	0.382	21,960

C

Evaluation Metrics

The degradation modelling and Remaining Useful Life (RUL) estimation presented in this thesis were evaluated using the coefficient of determination (R^2), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). These metrics quantify the agreement between the predicted values and their corresponding reference values.

C.0.1 Coefficient of Determination (R^2)

The coefficient of determination measures how well the predicted values explain the variation in the reference values and is calculated as

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

where y_i is the reference value, $\hat{y}_i$ is the predicted value, $\bar{y}$ is the mean of the reference values, and N is the number of observations.

An R^2 value of 1 indicates perfect agreement between the predicted and reference values, whereas values closer to 0 indicate weaker agreement. Therefore, higher R^2 values correspond to better model performance.

C.0.2 Mean Absolute Error (MAE)

The Mean Absolute Error represents the average absolute difference between the predicted and reference values and is calculated as

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right|. \tag{C.2}$$

The MAE is expressed in the same units as the predicted quantity. A value of zero indicates perfect prediction, while smaller values indicate greater prediction accuracy.

C.0.3 Root Mean Square Error (RMSE)

The Root Mean Square Error measures the average prediction error while assigning greater weight to larger deviations. It is calculated as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}. \tag{C.3}$$

Like the MAE, the RMSE is expressed in the same units as the predicted quantity. A value of zero indicates perfect prediction, while lower RMSE values indicate better predictive performance.

C.0.4 Interpretation in This Thesis

The three evaluation metrics complement one another and should therefore be interpreted together. In this thesis,

- a higher R^2 indicates that the predicted degradation trend more closely follows the reference trend;
- a lower MAE indicates a smaller average error in the estimated remaining-life fraction;
- a lower RMSE indicates fewer large prediction errors and greater consistency in the estimated remaining-life fraction.

Consequently, the best-performing component models are characterised by high R^2 together with low MAE and RMSE values.

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY