MASTER EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MASTER

COMPARATIVE ANALYSIS OF MODEL-BASED SOH ESTIMATION METHODS FOR LI-ION BEV BATTERIES USING REAL-WORLD DATA SETS

Autor: Paul Aaron Ulmer

Director: Kilian Sagner

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

"Comparative analysis of model-based SoH estimation methods for Li-ion BEV batteries using real-world data sets"

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2024/2025 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Paul Aaron Ulmer Fecha: 18/08/2025

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Kilian Sagner Fecha: 26/08/2025

COMPARATIVE ANALYSIS OF MODEL-BASED SOH ESTIMATION METHODS FOR LI-ION BEV BATTERIES USING REAL-WORLD DATA SETS

Author: Ulmer, Paul Aaron.
Supervisor: Cossent Arín, Rafael.
Collaborating Entity: FEV Consulting

ABSTRACT

This study develops a multi-criteria evaluation framework for model-based State of Health (SoH) estimation methods for lithium-ion batteries (LiBs), enabling transparent and use-case specific method selection. More than twenty methods were qualitatively assessed, with three chosen for validation on real-world datasets. Results confirmed the framework's initial qualitative ratings with only minor adjustments.

Keywords: SoH, Li-ion battery, Electric vehicle, Model-based estimation, Real world data

1. Introduction

The global shift towards decarbonization has placed the electrification of transportation at the center of climate mitigation strategies [1]. This has rapidly increased demand for high-performance battery systems, driven by the widespread adoption of battery electric vehicles (BEVs) [2]. To meet this demand, LiBs have emerged as the preferred solution, due to their high energy density, efficiency and long cycle life [3]. However, LiBs degrade due to a series of irreversible internal chemical and physical processes [4]. Beyond reducing performance, this can impair safety, increase operational cost and reduce system reliability [4].

To mitigate these risks, Battery Management Systems (BMS) are employed [4]. As the central control system of a battery, they are responsible for monitoring battery parameters; protecting the battery against unsafe operating conditions; balancing of cell voltages; and estimating internal battery states (e.g., SoH) [5]. Among these, accurate SoH estimation is critical to ensure safe and reliable operation [6]. Since SoH is not directly measurable, it must be inferred from indirect observations, such as voltage, current and temperature [5]. To address this challenge, researchers have developed a wide range of techniques, which can be categorized into experimental, model-based and hybrid approaches [5]. In particular, model-based methods, have gained popularity due to real-time, non-invasive estimation [6].

2. Project definition

The growing variety of model-based methods has led to inconsistent evaluation, often focusing on a single performance dimension, typically accuracy [6]. This overlooks critical dimensions for practical deployment such as computational efficiency or robustness to noise. In addition, most methods are validated on laboratory datasets, which fail to capture the complexity of real-world degradation [7]. Consequently, methods trained on such data often exhibit significant generalization gaps when applied to real-world scenarios [7].

This study addresses these limitations by developing a replicable, multi-criteria evaluation framework that allows model-based SoH estimation methods to be compared across multiple dimensions relevant to real-world use. In addition, it validates initial qualitative method assessments using real-world datasets. The detailed approach is shown in **Figure. 1**.

Figure. 1: Five-step approach to develop and validate a multi-criteria evaluation framework for use-case specific SoH method selection

3. Evaluation framework development and qualitative assessment

The framework is derived via an extensive literature review, in which five suitable studies are identified. They are selected based on their use of multi-criteria evaluation frameworks, which are applied to multiple model-based SoH estimation approaches. A comparative analysis of these studies reveals six key evaluation criteria: accuracy, computational efficiency, interpretability, data requirements, reliability and scalability. Each criterion is formally defined and a corresponding five-level scoring scale is introduced, ranging from "Very Low" to "Very High", to facilitate a consistent and transparent assessment process.

Then, over twenty model-based SoH estimation methods are qualitatively evaluated against these six criteria. Ratings are assigned based on empirical comparisons from literature, methodological strengths and weaknesses and expert insights from comparative reviews.

The outcome of this evaluation highlights ANNs and ensemble learning as the best methods across the six criteria, due to high levels of accuracy, reliability and scalability. However, these strengths are accompanied by increased computational demands, significant data requirements and limited interpretability. In the context of this study, interpretability and data availability are no limiting factors, as public datasets are accessible and sufficient methodological understanding has been developed to construct such methods. Given these boundary conditions, RF (Random Forest), XGBoost and CNN (Convolutional neural network) are selected for empirical validation.

4. Quantitative validation of evaluation framework

To validate the selected methods, a real-world dataset, consisting of three smaller datasets, is prepared. After preprocessing the data is labeled, forming the basis to train the selected methods. The labeling process starts with the grouping of data points based on vehicle ID and mileage, before concatenating them into charging segments occurring at the same mileage. Each charging segment is used to estimate the battery's current capacity. By dividing the current capacity with the battery's nominal capacity, its SoH is calculated. Finally, the SoH is assigned to all charging snippets within that charging segment.

To apply the three selected SoH estimation methods to the labeled dataset and validate the initial qualitative ratings, each method is implemented in Python.

5. Results

As accuracy, computational efficiency, reliability and scalability were used to guide the selection of SoH estimation methods, for empirical validation for the qualitative ratings, they are assigned a corresponding quantitative metric. Accuracy is measured using the Mean Absolute Percentage Error (MAPE), computational efficiency via the training and inference time, reliability by comparing the MAPE across multiple datasets and scalability by comparing the MAPE and training time on several proportions of one dataset.

The results for each of the four criteria is shown in **Figure. 2**. They largely validate the qualitative assessments, requiring only minor adjustments.

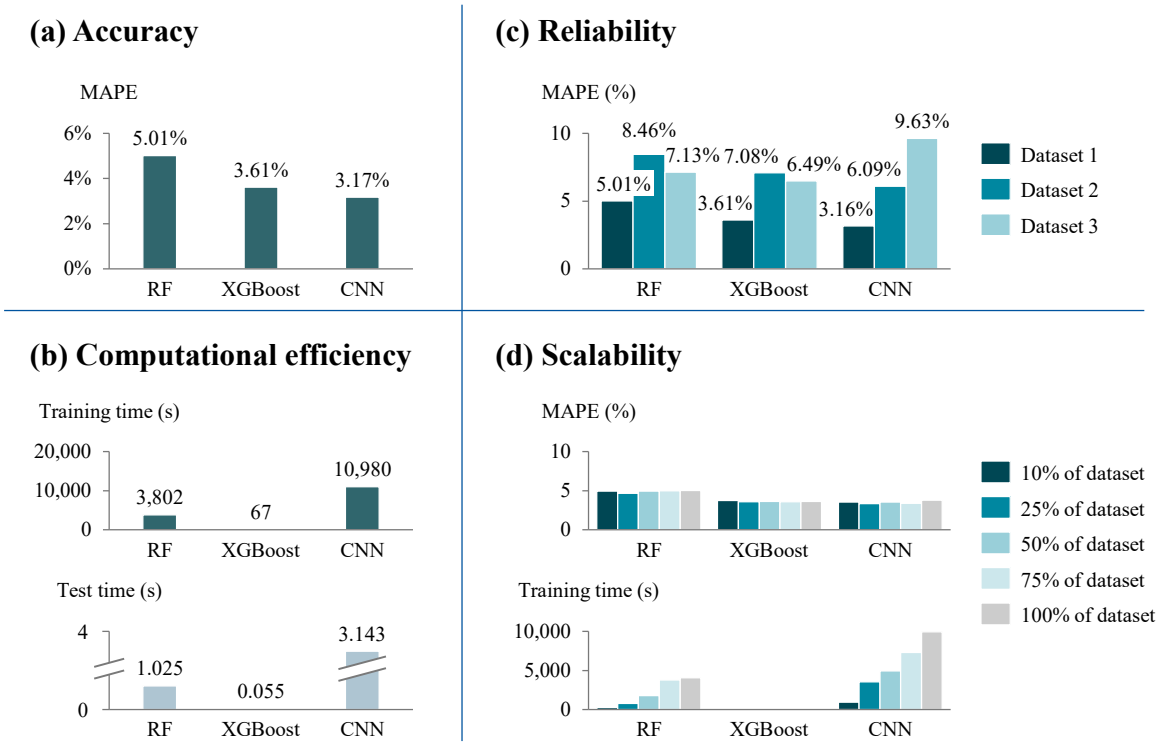


Figure. 2: Quantitative results across the four selection criteria: accuracy, computational efficiency, reliability and scalability

These findings can be used for recommendations for method selection in practice, e.g., RF for onboard BMS in BEVs, where method stability and robustness under varying operational conditions are essential, or XGBoost for diagnostic tools and low-resource environments, combining speed, accuracy and reliability. At the same time, the results highlight the limitations of the selected quantitative metrics. While they offer meaningful insights, certain aspects of the initial criteria definitions are not covered and require further refinement.

6. Conclusion

This study successfully developed a multi-criteria evaluation framework that allows users to select SoH estimation methods for their specific use-case. In addition it assessed a total of over twenty methods qualitatively using empirical comparisons and comparative studies found in literature. Afterwards, the qualitative ratings of a subset of the evaluated methods was validated using real-world data, which largely confirmed the qualitative ratings, with only a few adjustments necessary.

To further improve the framework, future research should focus on three key directions. First, expand the current quantitative metrics to enable a more complete assessment of the evaluation criteria. Second, apply additional model-based SoH estimation methods to the same dataset to develop quantitative thresholds for the framework's rating scale and transition the qualitative assessment to a hybrid qualitative-quantitative framework, allowing for absolute, rather than relative, performance comparison across methods. Third, greater effort should be placed on obtaining usable real-world datasets for method development, as such datasets are currently limited yet essential for testing practical applicability.

7. References

[1] P. Eleftheriadis, M. Gangi, S. Leva, A. V. Rey, E. Groppo und L. Grande, „Comparative study of machine learning techniques for the state of health estimation of Li-Ion batteries“, in Electric Power Systems Research, DOI: 10.1016/j.epsr.2024.110889, 2024

[2] N. Noura, L. Boulon und S. Jemeï, „A Review of Battery State of Health Estimation Methods: Hybrid Electric Vehicle Challenges“, in World Electric Vehicle Journal, DOI: 10.3390/wevj11040066, 2020

[3] X. Sui, S. He, S. B. Vilsen, J. Meng, R. Teodorescu und D.-I. Stroe, „A review of non-probabilistic machine learning-based state of health estimation techniques for Lithium-ion battery“, in Applied Energy, DOI: 10.1016/j.apenergy.2021.117346, 2021

[4] K. Yang, L. Zhang, Z. Zhang, H. Yu, W. Wang, M. Ouyang, C. Zhang, Q. Sun, X. Yan, S. Yang und X. Liu, „Battery State of Health Estimate Strategies: From Data Analysis to End-Cloud Collaborative Framework“, in Batteries, DOI: 10.3390/batteries9070351, 2023

[5] S. K. Pradhan und B. Chakraborty, „Battery management strategies: An essential review for battery state of health monitoring techniques“, in Journal of Energy Storage, DOI: 10.1016/j.est.2022.104427, 2022

[6] J. Tao, S. Wang, W. Cao, P. Takyi-Aninakwa, C. Fernandez und J. M. Guerrero, „A comprehensive review of state-of-charge and state-of-health estimation for lithium-ion battery energy storage systems“, in Ionics, DOI: 10.1007/s11581-024-05686-z, 2024

[7] Y. Lu, D. Guo, G. Xiong, Y. Wei, J. Zhang, Y. Wang und M. Ouyang, „Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data“, in eTransportation, DOI: 10.1016/j.etran.2024.100361, 2024

ANÁLISIS COMPARATIVO DE MÉTODOS DE ESTIMACIÓN DEL SHO BASADOS EN MODELOS PARA BATERÍAS LI-ION DE BEV UTILIZANDO CONJUNTOS DE DATOS DEL MUNDO REAL

Autor: Ulmer, Paul Aaron.
Director: Cossent Arín, Rafael.
Entidad Colaboradora: FEV Consulting

RESUMEN DEL PROYECTO

El presente estudio tiene como objetivo desarrollar un marco de evaluación multicriterio para métodos de estimación del estado de salud (SoH) basados en modelos para baterías de iones de litio (LiB). Este marco permite una selección de métodos transparente y específica para cada caso de uso. Se evaluaron cualitativamente más de veinte métodos, de los cuales se seleccionaron tres para su validación en conjuntos de datos del mundo real. Los resultados de la investigación confirmaron las calificaciones cualitativas iniciales del marco, con solo ligeras modificaciones.

Palabras clave: Estado de salud, batería de Li-ion, vehículo eléctrico, métodos de estimación basados en modelos, conjunto de datos del mundo real

1. Introducción

El cambio global hacia la descarbonización ha situado la electrificación del transporte en el centro de las estrategias de mitigación del cambio climático [1]. Esto ha provocado un rápido aumento de la demanda de sistemas de baterías de alto rendimiento, impulsado por la adopción generalizada de los vehículos eléctricos de batería (BEV) [2]. Para satisfacer esta demanda, las LiB se han convertido en la solución preferida, debido a su alta densidad energética, eficiencia y larga vida útil [3]. Sin embargo, las LiB se degradan debido a una serie de procesos químicos y físicos internos irreversibles [4]. Además de reducir el rendimiento, esto puede afectar a la seguridad, aumentar los costes operativos y reducir la fiabilidad del sistema [4].

Para mitigar estos riesgos, se emplean sistemas de gestión de baterías (BMS) [4]. Como sistema de control central de una batería, se encargan de supervisar los parámetros de la batería, protegerla contra condiciones de funcionamiento inseguras, equilibrar los voltajes de las celdas y estimar los estados internos de la batería (p. ej., SoH) [5]. Entre ellos, la estimación precisa del SoH es fundamental para garantizar un funcionamiento seguro y fiable [6]. Dado que el SoH no se puede medir directamente, debe inferirse a partir de observaciones indirectas, como el voltaje, la corriente y la temperatura [5]. Para abordar este reto, los investigadores han desarrollado una amplia gama de técnicas, que pueden clasificarse en enfoques experimentales, basados en modelos e híbridos [5]. En particular, los métodos basados en modelos han ganado popularidad debido a su estimación en tiempo real y no invasiva [6].

2. Definicíon del proyecto

La creciente variedad de métodos basados en modelos ha dado lugar a evaluaciones inconsistentes, que a menudo se centran en una única dimensión del rendimiento, normalmente la precisión [6]. Esto pasa por alto dimensiones críticas para la implementación práctica, como la eficiencia computacional o la robustez frente al ruido. Además, la mayoría de los métodos se validan con conjuntos de datos de laboratorio, que no logran captar la

complejidad de la degradación en el mundo real [7]. En consecuencia, los métodos entrenados con estos datos suelen presentar importantes lagunas de generalización cuando se aplican a escenarios del mundo real [7].

Este estudio aborda estas limitaciones mediante el desarrollo de un marco de evaluación multicriterio replicable que permite comparar los métodos de estimación del estado de salud basados en modelos en múltiples dimensiones relevantes para el uso en el mundo real. Además, valida las evaluaciones cualitativas iniciales de los métodos utilizando conjuntos de datos del mundo real. El enfoque detallado se muestra en la **Figura. 1**.

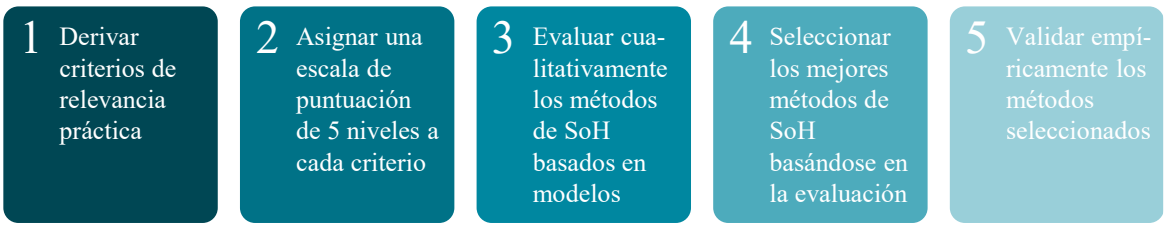


Figura. 1: Enfoque de cinco pasos para desarrollar y validar un marco de evaluación multicriterio para la selección de métodos de estado de salud específicos para cada caso de uso

3. Desarrollo del marco de evaluación y evaluación cualitativa

El marco se deriva de una amplia revisión bibliográfica, en la que se identifican cinco estudios adecuados. Estos se seleccionan en función de su uso de marcos de evaluación multicriterio, que se aplican a múltiples enfoques de estimación del SoH basados en modelos. Un análisis comparativo de estos estudios revela seis criterios clave: precisión, eficiencia computacional, interpretabilidad, requisitos de datos, fiabilidad y escalabilidad. Cada criterio se define formalmente y se introduce una escala de puntuación de cinco niveles, que va de "muy bajo" a "muy alto", para facilitar un proceso de evaluación coherente y transparente.

A continuación, se evalúan cualitativamente más de veinte métodos de estimación del SoH basados en modelos en función de estos seis criterios. Las calificaciones se asignan basándose en comparaciones empíricas de la bibliografía, los puntos fuertes y débiles de la metodología y las opiniones de expertos procedentes de revisiones comparativas.

El resultado de esta evaluación destaca las redes neuronales artificiales (ANN) y el aprendizaje conjunto como los mejores métodos en los seis criterios, debido a sus altos niveles de precisión, fiabilidad y escalabilidad. Sin embargo, estas ventajas van acompañadas de mayores exigencias computacionales, requisitos de datos significativos y una interpretabilidad limitada. En el contexto de este estudio, la interpretabilidad y la disponibilidad de datos no son factores limitantes, ya que se puede acceder a conjuntos de datos públicos y se ha desarrollado una comprensión metodológica suficiente para construir dichos métodos. Dadas estas condiciones límite, se seleccionan RF (Random Forest), XGBoost y CNN (red neuronal convolucional) para la validación empírica.

4. Validación cuantitativa del marco de evaluación

Para validar los métodos seleccionados, se prepara un conjunto de datos del mundo real, compuesto por tres conjuntos de datos más pequeños. Tras el preprocesamiento, los datos se etiquetan, lo que constituye la base para entrenar los métodos seleccionados. El proceso de etiquetado comienza con la agrupación de los puntos de datos en función del ID del vehículo y el kilometraje, antes de concatenarlos en segmentos de carga que se producen en el mismo

kilometraje. Cada segmento de carga se utiliza para estimar la capacidad actual de la batería. Dividiendo la capacidad actual por la capacidad nominal de la batería, se calcula su SoH. Por último, el SoH se asigna a todos los fragmentos de carga dentro de ese segmento de carga.

Para aplicar los tres métodos de estimación del SoH seleccionados al conjunto de datos etiquetado y validar las calificaciones cualitativas iniciales, cada método se implementa en Python.

5. Resultados

Dado que la precisión, la eficiencia computacional, la fiabilidad y la escalabilidad se utilizaron para guiar la selección de los métodos de estimación del SoH, para la validación empírica de las calificaciones cualitativas se les asignó una métrica cuantitativa correspondiente. La precisión se mide utilizando el error porcentual absoluto medio (MAPE), la eficiencia computacional a través del tiempo de entrenamiento e inferencia, la fiabilidad comparando el MAPE en múltiples conjuntos de datos y la escalabilidad comparando el MAPE y el tiempo de entrenamiento en varias proporciones de un conjunto de datos.

Los resultados para cada uno de los cuatro criterios se muestran en la **Figura. 2**. En gran medida, validan las evaluaciones cualitativas, requiriendo solo ajustes menores.

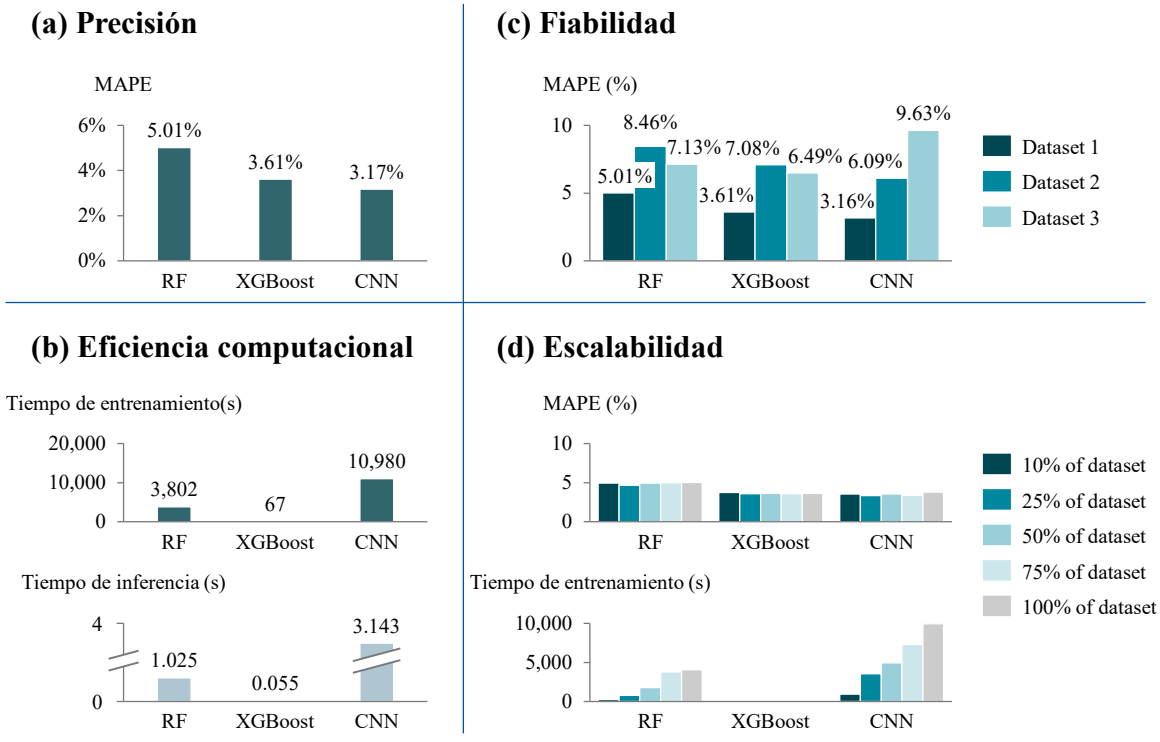


Figura. 2: Resultados cuantitativos de los cuatro criterios de selección: precisión, eficiencia computacional, fiabilidad y escalabilidad

Estos resultados pueden utilizarse para recomendar la selección de métodos en la práctica, por ejemplo, RF para BMS a bordo en BEV, donde la estabilidad y la robustez del método en condiciones operativas variables son esenciales, o XGBoost para herramientas de diagnóstico y entornos con pocos recursos, que combina velocidad, precisión y fiabilidad. Al mismo tiempo, los resultados ponen de relieve las limitaciones de las métricas

cuantitativas seleccionadas. Si bien ofrecen información significativa, algunos aspectos de las definiciones iniciales de los criterios no están cubiertos y requieren un mayor refinamiento.

6. Conclusión

Este estudio desarrolló con éxito un marco de evaluación multicriterio que permite a los usuarios seleccionar métodos de estimación del SoH para su caso de uso específico. Además, evaluó un total de más de veinte métodos de forma cualitativa utilizando comparaciones empíricas y estudios comparativos encontrados en la literatura. Posteriormente, las calificaciones cualitativas de un subconjunto de los métodos evaluados se validaron utilizando datos del mundo real, lo que confirmó en gran medida las calificaciones cualitativas, con solo unos pocos ajustes necesarios.

Para mejorar aún más el marco, las investigaciones futuras deberían centrarse en tres direcciones clave. En primer lugar, ampliar las métricas cuantitativas actuales para permitir una evaluación más completa de los criterios de evaluación. En segundo lugar, aplicar métodos adicionales de estimación del SoH basados en modelos al mismo conjunto de datos para desarrollar umbrales cuantitativos para la escala de calificación del marco y pasar de la evaluación cualitativa a un marco híbrido cualitativo-cuantitativo, lo que permitiría una comparación absoluta, en lugar de relativa, del rendimiento entre los distintos métodos. En tercer lugar, se debe hacer un mayor esfuerzo por obtener conjuntos de datos del mundo real que sean útiles para el desarrollo de métodos, ya que dichos conjuntos de datos son actualmente limitados, pero esenciales para probar la aplicabilidad práctica.

7. Referencias

[1] P. Eleftheriadis, M. Gangi, S. Leva, A. V. Rey, E. Groppo und L. Grande, „Comparative study of machine learning techniques for the state of health estimation of Li-Ion batteries“, in Electric Power Systems Research, DOI: 10.1016/j.epsr.2024.110889, 2024

[2] N. Noura, L. Boulon und S. Jemeï, „A Review of Battery State of Health Estimation Methods: Hybrid Electric Vehicle Challenges“, in World Electric Vehicle Journal, DOI: 10.3390/wevj11040066, 2020

[3] X. Sui, S. He, S. B. Vilsen, J. Meng, R. Teodorescu und D.-I. Stroe, „A review of non-probabilistic machine learning-based state of health estimation techniques for Lithium-ion battery“, in Applied Energy, DOI: 10.1016/j.apenergy.2021.117346, 2021

[4] K. Yang, L. Zhang, Z. Zhang, H. Yu, W. Wang, M. Ouyang, C. Zhang, Q. Sun, X. Yan, S. Yang und X. Liu, „Battery State of Health Estimate Strategies: From Data Analysis to End-Cloud Collaborative Framework“, in Batteries, DOI: 10.3390/batteries9070351, 2023

[5] S. K. Pradhan und B. Chakraborty, „Battery management strategies: An essential review for battery state of health monitoring techniques“, in Journal of Energy Storage, DOI: 10.1016/j.est.2022.104427, 2022

[6] J. Tao, S. Wang, W. Cao, P. Takyi-Aninakwa, C. Fernandez und J. M. Guerrero, „A comprehensive review of state-of-charge and state-of-health estimation for lithium-ion battery energy storage systems“, in Ionics, DOI: 10.1007/s11581-024-05686-z, 2024

[7] Y. Lu, D. Guo, G. Xiong, Y. Wei, J. Zhang, Y. Wang und M. Ouyang, „Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data“, in eTransportation, DOI: 10.1016/j.etran.2024.100361, 2024

Table of contents

List of illustrations

List of tables

Abbreviations

Abbreviation	*Explanation*
ANN	Artificial neural network
API	Application programming interface
BDU	Battery disconnect unit
BEV	Battery electric vehicle
BMS	Battery management system
CALCE	Center for Advanced Life Cycle Engineering
CART	Classification and regression tree algorithm
CEI	Cathode-electrolyte interphase
CNN	Convolutional neural network
CPU	Central processing unit
DEC	Diethyl carbonate
DMC	Dimethyl carbonate
DT	Decision tree
ECM	Equivalent circuit model
E/E	Electrical/electronic
EKF	Extended Kalman filter
EM	Electrochemical model
FFNN	Feedforward neural network
FL	Fuzzy logic

GBDT	Gradient boosting decision tree
GPR	Gaussian process regression
GPU	Graphics processing unit
GRU	Gated recurrent unit
IR	Increased internal resistance
KF	Kalman filter
kNN	k-nearest neighbors
LAM	Loss of active electrode material
LiB	Lithium-ion battery
Li-ion	Lithium-ion
LLI	Loss of lithium inventory
LSTM	Long-short term memory
MAPE	Mean absolute percentage error
MSE	Mean square error
ML	Machine learning
NMC-C	Nickel-manganese-cobalt carbon battery
NN	Neural network
P2D	Pseudo-two-dimensional model
OCV	Open circuit voltage
OOB	Out-of-bag samples
PF	Particle filter

RAM	Random-access memory
ReLU	Rectified linear unit
RF	Random forest
RMSE	Root mean square error
RNN	Recurrent neural network
RVM	Relevance vector machine
SEI	Solid electrolyte interphase
SoC	State of charge
SoH	State of health
SoP	State of power
SPM	Single particle model
SVM	Support vector machine
SVR	Support vector regression
TMD	Transition metal dissolution
UKF	Unscented Kalman filter
WP	Wiener process

Symbols

Symbol	*Description*
$B(t)$	Standard Brownian motion
C_P	Charge transfer impedance
f	Filter in CNN
f_m	Decision tree function
g_i	First order derivative of the loss function in XGBoost
$h(x)$	RF model function
h_i	Second order derivative of the loss function in XGBoost
I	Current
I_j	Sample set assigned to leaf node j in XGBoost
K	Total number of samples (MAPE)
k	Number of selected features for node partition in RF
L	Objective function of XGBoost
L_{split}	Candidate split gain in XGBoost
$l(y_i, \hat{y}_i)$	Loss function of the objective function of XGBoost
M	Number of features in RF
m	Number of DTs in XGBoost
N	Amount of particles in PF
$o_{cov\,p}$	Value at position p in feature map after convolutional layer in CNN
$o_{pool\,p}$	Value at position p in feature map after pooling layer in CNN

P	Number of leaf nodes in DT of XGBoost
Q	Current lithium capacity
Q_0	Initial lithium capacity
Q_m	Current maximum available capacity
Q_r	Initial maximum available capacity
$q(x)$	Mapping function of leaf nodes in XGBoost
R	Current internal resistance
R_0	Ohmic Resistor
R_C	Charge transfer impedance
R_e	End-of-life resistance
R_n	Initial resistance
R_t	Decision tree in RF
S	Original dataset
S_T	Labels of sample sets in RF
SoC_{end}	SoC at end time
SoC_{start}	SoC at start time
s	Number of OOB samples
T	Number of sample sets in RF
T_{max}	Maximum cell temperature
T_{min}	Minimum cell temperature
t	Timestamp

t_{end}	End time
t_{start}	Start time
U_{0C}	Ideal voltage source
V	Cell voltage
V_{max}	Maximum cell voltage
V_{min}	Minimum cell voltage
W	Functional space of all possible DTs
W_0	Uniform initial weight of particles in PF
w	Leaf node weight in RF
w_j	Leaf node weight in XGBoost
$w_j{}^*$	Optimal leaf node weight in XGBoost
x_0	Initial capacity of battery in WP
y	True SoH value
$\hat{y}$	Predicted SoH value
γ	Complexity parameter in XGBoost
ϵ	Small positive constant to prevent division by zero (MAPE)
η	Penalty parameter of the leaf weights in XGBoost
λ	Drift coefficient in WP
σ	Diffusion coefficient in WP
$\boldsymbol{\Omega(f_m)}$	Regularization term of the objective function of XGBoost

Chapter 1. INTRODUCTION

As the opening chapter of this thesis, the introduction aims to establish a foundational understanding of the key topics and provide orientation for the reader. It begins by outlining the motivation behind the research (**Chapter 1.1**), followed by a discussion of the research context and the central guiding question (**Chapter 1.2**). Finally, the chapter presents the structure of the thesis, offering an overview of its organization and content (**Chapter 1.3**).

1.1 MOTIVATION AND OBJECTIVES

The global shift towards decarbonization has placed the electrification of transportation at the center of climate mitigation strategies [1]. This transition has led to a rapid increase in demand for high-performance battery systems, driven by the widespread adoption of battery electric vehicles (BEVs) [2–9]. To meet this demand, lithium-ion batteries (LiBs) have emerged as the preferred solution, due to their high energy density, efficiency and long cycle life, among other factors [3–6, 9–15]. However, LiBs degrade over time due to a series of irreversible internal chemical and physical processes, causing a gradual loss in power output and capacity [1, 3–6, 8–10, 13, 16–18]. Beyond reducing performance, it can impair safety, increase operational cost and reduce system reliability, potentially leading to premature failure or safety-critical incidents [4].

To mitigate these risks and extend battery life, Battery Management Systems (BMS) have become an essential component in LiB-based applications [4, 11, 19]. As the central control system of a battery, a BMS is responsible for monitoring battery parameters; protecting the battery against unsafe operating conditions; balancing of cell voltages; and estimating internal battery states, such as State of Charge (SoC) and State of Health (SoH) [2, 4, 5, 7, 9, 17–22]. Among these, accurate SoH estimation is critical to ensure safe and reliable operation [9, 23]. The SoH reflects the battery's ageing state

and helps predict its remaining useful life, enabling failure anticipation, predictive maintenance and battery replacement planning [2–4, 13, 19].

Since SoH is not directly measurable, it must be inferred from indirect observations, such as voltage, current and temperature profiles under various load conditions [2, 4, 5, 8, 13]. To address this challenge, researchers have developed a wide range of estimation techniques, which can be categorized into experimental, model-based and hybrid approaches [1–5, 8, 19, 21, 23, 24]. Among these, model-based methods have become particularly attractive as they can estimate SoH in real-time without disrupting battery operation [12].

Despite this progress, the evaluation and application of model-based SoH estimation methods remain challenging due to inconsistent assessment practices and a lack of consideration for real-world variability. This thesis aims to address these gaps by developing a comprehensive evaluation framework that supports the selection of appropriate SoH estimation methods for practical deployment.

1.2 CONTEXT OF RESEARCH

The increasing relevance of battery longevity, environmental impact, predictive maintenance and second-life planning has increased the importance of estimating a battery's SoH in both academia and industry [25, 26]. In particular, model-based SoH estimation techniques, ranging from physics-informed approaches like equivalent circuit models (ECMs), to data-driven methods based on machine learning (ML) and neural networks (NNs), have become popular due to their potential for real-time, non-invasive estimation [3, 8, 9].

However, the growing variety of approaches has resulted in fragmentation and inconsistency in how SoH estimation methods are evaluated. Most studies assess approaches primarily on a single performance dimension, typically accuracy [4, 5, 9, 14, 23, 24, 27]. This overlooks critical dimensions for practical deployment such as

computational efficiency, robustness to input noise and adaptability to different operating conditions. In addition, most methods are validated on laboratory datasets, which fail to capture the variability and complexity of real-world degradation in operational environments [8, 24]. Consequently, models trained on laboratory data often exhibit significant generalization gaps when applied to real-world scenarios [8].

This thesis addresses these limitations by developing and validating a replicable, multi-criteria evaluation framework that allows model-based SoH estimation methods to be compared across multiple dimensions relevant to real-world use. The central research question guiding this thesis is:

How can model-based SoH estimation methods be systematically evaluated across multiple practically relevant criteria?

To answer the overarching research question, this thesis identifies the core performance dimensions relevant for real-world deployment of SoH estimation methods, formalizes them into a structured evaluation framework and applies this framework to compare selected approaches under identical conditions using real-world data. In doing so, the work bridges the gap between method development and operational deployment, offering both a theoretical contribution and a practical tool for evaluating and selecting SoH models across diverse use cases.

1.3 STRUCTURE

This thesis is structured into six chapters, developing, implementing and validating a comprehensive evaluation framework for model-based SoH estimation methods. After **Chapter 1**, introducing the research motivation, outlining current challenges in battery health diagnostics and defining the central objectives addressed in this work, **Chapter 2** provides the theoretical foundation. It begins with an overview of LiB operation and ageing mechanisms, followed by a formal definition of SoH and its relevance to degradation assessment. It then gives an overview of different SoH

estimation methods, before focusing on model-based SoH estimation. The specific methods selected for further analysis in this thesis are introduced and briefly characterized. After that, **Chapter 3** develops the proposed evaluation framework. First, a set of practically relevant evaluation criteria is derived from the literature and thoroughly defined. Each criterion is operationalized using a five-level scoring scale ranging from "Very Low" to "Very High". In the second part of the chapter, the presented model-based SoH estimation methods are qualitatively assessed and ranked according to these criteria. Based on the resulting evaluation profiles, three high-potential methods are selected for implementation and further empirical validation. In **Chapter 4**, the practical application of the selected methods is described. A real-world dataset is introduced, preprocessed and labelled with SoH values. In addition, the chapter outlines the implementation details of each of the selected SoH estimation methods in Python. **Chapter 5** presents the comparative analysis of the implemented methods. Evaluation criteria that were qualitatively assessed in Chapter 3 are now translated into quantitative metrics and their values are obtained from the application on the real-world dataset. The results are analyzed across the defined criteria and used to refine and validate the initial qualitative assessment. The chapter concludes with a discussion on method feasibility in different use cases and identifies the most suitable approach. Lastly, **Chapter 6** summarizes the main findings, reflects on the contributions of the thesis and outlines directions for future research topics.

Chapter 2. TECHNICAL FUNDAMENTALS

This chapter aims to establish the theoretical framework of this thesis and highlight the need for a comprehensive selection framework that compares existing model-based SoH estimation methods for LiBs. First, **Chapter 2.1** provides a comprehensive overview of LiBs. It details the internal components and operating principles at cell level, before discussing the different cell types and their integration into battery system architectures. The section concludes with an examination of BMS, which plays a pivotal role in monitoring battery degradation and ensuring safe and reliable operation. **Chapter 2.2** focuses on the degradation mechanisms that lead to the deterioration of LiB performance. It explores processes occurring at both electrodes and within the electrolyte, highlighting the complex interactions between them. Following this, **Chapter 2.3** defines the concept of SoH, which numerically specifies the battery degradation. It then introduces the principal methodologies used for its estimation. Among these, model-based techniques have gained relevance due to their non-invasive nature. **Chapter 2.4** presents an in-depth review of commonly employed model-based SoH estimation methods. This section outlines a range of approaches, each with specific assumptions, advantages and limitations. **Chapter 2.5** concludes by arguing for the necessity of a structured framework to compare the presented model-based SoH estimation methods. Such a framework would support the identification of optimal techniques for specific use cases, ultimately guiding the development of more robust and application-oriented BMS solutions.

2.1 LITHIUM-ION BATTERY (LIB)

LiBs are electrochemical storage systems that enable efficient, reversible energy conversion through the movement of Li-ions [28, 29]. Their widespread adoption across

applications is driven by their high energy density, long cycle life and favorable performance characteristics under varying operating conditions [14, 30].

2.1.1 BATTERY COMPONENTS AND FUNCTIONALITY

Figure 1 illustrates the basic structure and operating principle of a LiB. The battery operates through the reversible movement of Li-ions between the anode and the cathode, which allows LiBs to store and release electrical energy [28, 31, 32]. Additional components include the electrolyte, separator and current collectors [31].

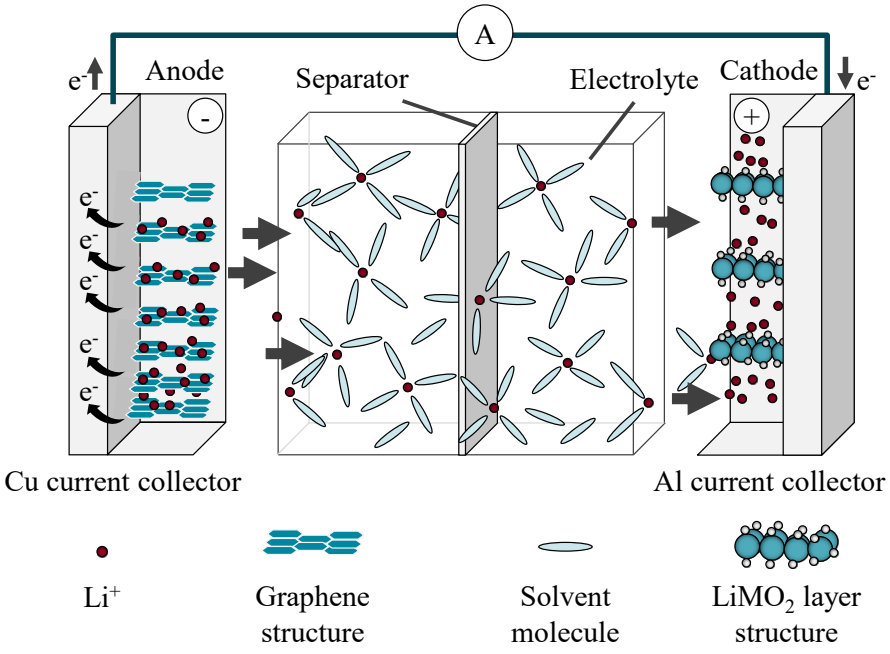


Figure 1: Schematic illustration of the discharging process of a LiB [15]

The anode is typically made of graphite (e.g., LiC_6), a material that can intercalate Li-ions between its carbon layers [28, 32]. These ions are electrostatically balanced by electrons within the graphite structure, resulting in neutral lithium. During discharge, the lithium atoms in the anode are oxidized, releasing electrons and Li-ions (**Eq. 2.1**).

$$Li_xC_6 \rightarrow xLi^+ + xe^- + C_6 \qquad \textit{Eq. 2.1}$$

The released Li-ions migrate through the electrolyte, a commonly liquid medium, containing a dissolved conductive lithium salt (e.g., $LiPF_6$) [28, 31, 32]. The electrolyte allows ionic conduction while blocking electrons [33]. As Li-ions move through the electrolyte, they pass through the separator, a porous polymer membrane that physically separates the electrodes while remaining permeable to ions [28, 31, 33]. It prevents electrical short circuits by avoiding direct contact between the anode and the cathode.

At the same time, the released electrons travel from the anode to the cathode through the external circuit. This electron flow generates the usable electric current. Once the electrons reach the cathode, which is typically composed of a lithium oxide (e.g., $LiCoO_2$), they reduce the transition metal ions (**Eq. 2.2**). The arriving Li-ions intercalate into the cathode structure, maintaining charge balance. [28, 31, 33]

$$Li_{1-x}CoO_2 + xLi^+ + xe^- \rightarrow LiCoO_2 \qquad \textit{Eq. 2.2}$$

Each electrode is coated onto a metal foil that serves as the current collector. Copper is used for the anode and aluminum for the cathode. These foils do not participate in the electrochemical reactions but ensure efficient electron transfer to and from the external circuit. [33]

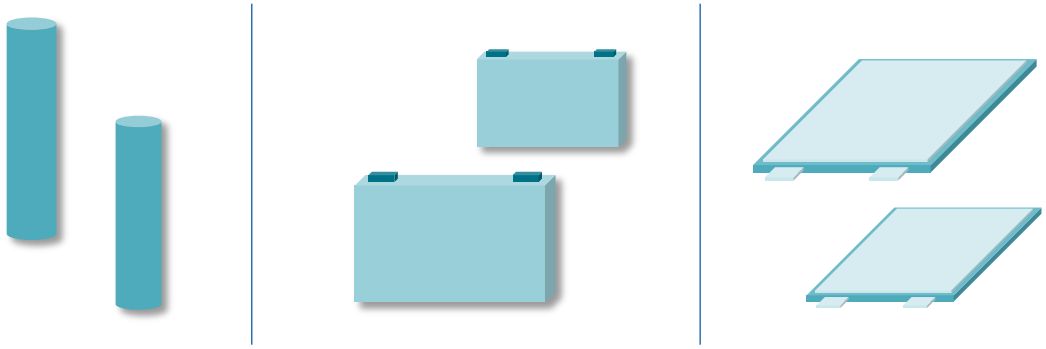
When the battery is charged, this process is reversed [28, 31]. An external power source forces Li-ions to deintercalate from the cathode and migrate back to the anode [15]. Electrons flow in the opposite direction through the circuit, reducing Li-ions at the anode surface and allowing them to re-enter the graphite structure [33].

2.1.2 BATTERY STRUCTURE

The practical performance and reliability of LiBs are contingent not only on their internal chemistry but also on their physical arrangement. This encompasses the design of individual cells as well as their integration into modules and packs.

2.1.2.1 Cell types

LiB cells are manufactured in three principal formats: cylindrical, prismatic and pouch (see **Figure 2**) [34, 35]. Each cell type offers distinct structural characteristics and trade-offs that influence its suitability for specific applications.



(a) Cylindrical cells (b) Prismatic cells (c) Pouch cells

Figure 2: Standard cell formats: (a) cylindrical, (b) prismatic and (c) pouch cells [35]

The primary differences among the three cell types lie in the configuration of the housing and the internal arrangement of the electrodes and separator. Cylindrical cells contain a spirally wound electrode array (jelly roll) that is surrounded by a rigid metal casing. This casing is typically made of stainless steel or aluminum. Prismatic cells arrange electrodes in either a flattened jelly roll configuration or in stacked layers. The casing material is like that of cylindrical cells. Finally, pouch cells utilize a stacked electrode configuration, which is enclosed in multiple layers of aluminum composite foils. [35]

2.1.2.2 Battery system architecture

To meet the voltage and capacity demands of BEVs, individual cells are not sufficient. They need to be assembled into battery packs [36]. Traditionally, multiple cells are consolidated into modules, which are then combined to form a battery pack [36]. However, an emerging approach is to directly integrate individual cells into the battery pack (cell-to-pack approach) [37]. Focusing on the modular approach, electrical

interconnections within the battery pack are configured in one of two ways. Either series-connected battery modules are arranged in parallel or parallel-connected battery modules are stacked in series (see **Figure 3**) [36].

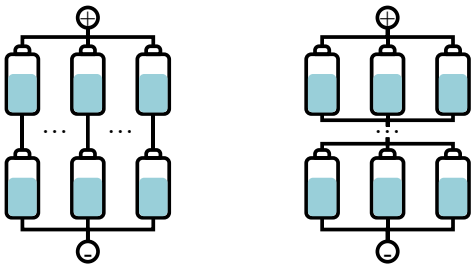


Figure 3: Battery pack structures. (a) Battery pack with series modules in parallel; (b) battery pack with parallel modules in series [36]

Parallel connection enhances the capacity, while series connection increases the voltage of the battery pack. The type of connection affects the control effort. It is common practice to connect cells in parallel first and then in series, as connecting large series of battery modules in parallel requires increased voltage balancing effort. [28]

The battery pack is embedded into the E/E-architecture of the vehicle, which enables communication, sensing and control. A crucial component of this architecture is the Battery Disconnect Unit (BDU) that manages high-voltage connections and isolates the pack if needed. Overseeing both the BDU and the broader battery systems is the BMS, which serves as the central control unit, ensuring safe operation, performance optimization and system protection. [28]

2.1.3 BATTERY MANAGEMENT SYSTEM (BMS)

As the central control unit, the BMS performs several critical functions. It regulates charging and discharging processes, balances individual cells and manages thermal conditions [23]. In addition, it protects the battery against operational risks such as overheating and deep discharge [23, 38]. Finally, it estimates several internal battery states, including SoH, state of charge (SoC) and state of power (SoP) [22, 28, 31].

Despite advanced control strategies, the BMS cannot prevent the gradual degradation of LiBs. Over time, complex chemical and physical changes alter the battery's behavior, leading to reduced performance and complicating the estimation of internal states such as SoH [38]. While the BMS provides continuous SoH estimates to anticipate failure and optimize operation, the concept itself reflects a complex interplay of multiple degradation mechanisms. A precise understanding of these underlying processes is therefore essential to clearly define SoH. Both aspects are examined in the following.

2.2 BATTERY DEGRADATION

Capacity and power fade are the primary observable effects of LiB degradation [39, 40]. Both effects are caused by two fundamental degradation modes: Loss of Lithium Inventory (LLI) and Loss of Active Material (LAM) at both the cathode and the anode [30, 41]. LLI refers to the irreversible consumption of cyclable lithium due to side reactions such as solid electrolyte interphase (SEI) formation, lithium plating and electrolyte decomposition [30, 40]. These reactions sequester Li-ions, preventing them from participating in further charge-discharge cycles [39]. LAM, in contrast, results from structural or chemical degradation of active electrode materials, including particle cracking, current collector corrosion and transition metal dissolution (TMD) [30]. These effects reduce the number of electrochemically active sites available for lithium intercalation and deintercalation [39]. **Figure 4** graphically illustrates the degradation mechanisms leading to the two primary degradation modes described above.

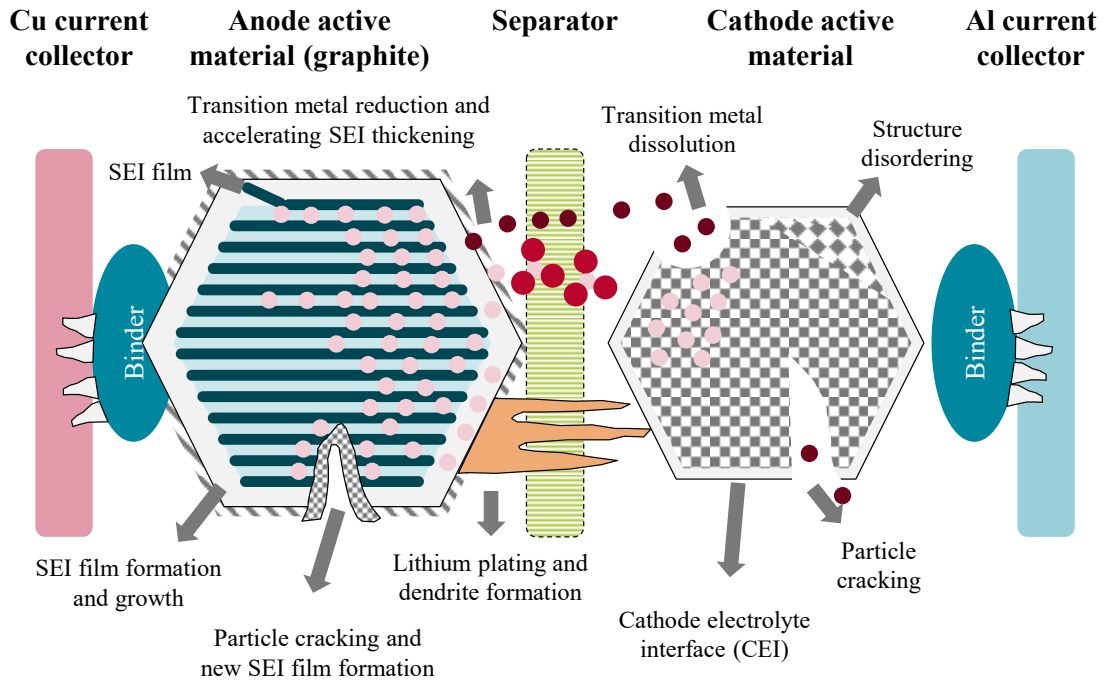


Figure 4: Overview of battery degradation mechanisms in Li-ion cells [42]

2.2.1 ANODE DEGRADATION

As shown in **Figure 4**, the anode is subject to several side reactions that significantly contribute to battery degradation. These processes occur primarily due to interactions with the electrolyte and repeated lithium intercalation and deintercalation, leading to the formation and growth of the SEI, lithium plating and particle fracture [15].

2.2.1.1 Solid Electrolyte Interphase (SEI) formation and growth

The SEI forms when the liquid electrolyte comes into contact with the conductive anode surface [15, 30, 41]. The anode surface typically operates below the electrochemical stability window of the electrolyte, which accelerates redox processes that break down the electrolyte and release Li-ions [39, 42]. Together with graphite, they form a passivation layer (SEI) that can be composed of various compounds (e.g., Li_2CO_3) [15, 39]. This reaction particularly occurs during the initial charge cycles and leads to irreversible lithium consumption, contributing to a substantial loss in capacity [30, 39].

The SEI layer is semi-permeable. That means, it allows Li-ion transport while blocking further electron flow, preventing continuous electrolyte decomposition [39]. However, its formation and evolution are highly dependent on factors such as graphite microstructure or electrolyte composition [41]. In addition, the SEI can crack due to volume changes during cycling, exposing fresh electrode surface leading to renewed SEI growth [15, 41]. This repetitive formation consumes active lithium, increases impedance and ultimately reduces the battery's efficiency and lifespan [15, 39].

Although SEI formation is essential for initial cell stabilization, it remains a dynamic structure. Elevated temperatures, high charging currents and prolonged cycling can accelerate its growth and deterioration, thereby exacerbating degradation and performance fade over time [15, 39].

2.2.1.2 Lithium Plating

Lithium plating refers to the undesirable deposition of metallic lithium on the anode surface instead of Li-ion intercalation into the graphite. This phenomenon occurs when the anode potential relative to the lithium falls below $0V$ which is reached more likely under high current rates, low temperatures or high SoCs [29]. Once deposited, metallic lithium may form dendritic structures that grow over successive cycles [29]. These dendrites can penetrate the separator, potentially leading to internal short circuits and cell failure. In addition, lithium plating contributes to capacity loss by forming electronically isolated "dead lithium", reducing the amount of active lithium in the system. To mitigate lithium plating, cells are designed with excess anode capacity. However, overcharging, uneven current distribution or localized defects can still result in its occurrence. [15, 39, 41]

2.2.1.3 Particle Fracture

Lastly, mechanical degradation of the anode is crucial. During cycling, the repeated lithiation and delithiation of graphite particles induce significant volumetric changes. This expansion and contraction generate internal mechanical stresses that can lead to

particle fracture [39]. Crack formation exposes fresh surface area, promoting further SEI growth and potentially detaching conductive additives. This leads to an increase in internal resistance and decrease in electronic conductivity. [15]

2.2.2 ELECTROLYTE DEGRADATION

The electrolyte is located between and within both the anode and the cathode, serving as a critical medium for Li-ion transport [15]. It is composed of organic solvents (e.g., DMC or DEC), lithium salts (e.g., $LiPF_6$) and additives if necessary [15, 41]. Its decomposition is initiated through both chemical and electrochemical pathways, particularly under high temperatures, overcharging and the presence of trace impurities such as moisture [15, 41]. A key degradation mechanism involves the breakdown of $LiPF_6$, especially in the presence of water. This leads to the formation of decomposition products such as di-fluorophosphate and phosphorus oxyfluoride [41]. These species are known to reduce ionic conductivity and increase the cell's ohmic resistance, thereby deteriorating overall battery performance [41]. Overall, the degradation of the electrolyte leads to increased gas formation, reduced lithium inventory and diminished ionic transport, impairing both battery capacity and safety [30, 41].

2.2.3 CATHODE DEGRADATION

Cathode degradation in LiBs is material-dependent, making it less predictable than anode side reactions. A key aspect is the formation of the cathode-electrolyte interphase (CEI). The CEI is a protective, yet resistive layer formed by oxidative decomposition of the electrolyte at the cathode surface. While the CEI suppresses further electrolyte breakdown and gas evolution, its poor Li-ion conductivity increases cell impedance and reduces energy efficiency. [29, 41]

A second contributor to cathode aging is TMD. This includes the dissolution of transition metal oxides based on nickel, manganese or cobalt into the electrolyte. This process is accelerated by elevated temperature, high voltage and HF formation

following $LiPF_6$ hydrolysis. The dissolved metal ions migrate to the anode, where they catalyze unwanted SEI thickening and contribute to lithium dendrite growth. [29, 41]

Furthermore, mechanical structural degradation is critical. Analogous to the anode, continuous lithiation and delithiation induce volume changes and mechanical stress in the cathode, which can lead to particle cracking and microcracking of the active material [29]. This exposes fresh active surfaces to the electrolyte, further promoting CEI growth and TMD. [41]

Finally, cathode materials such as NMC are also prone to phase transitions, including Jahn-Teller distortions. This phenomenon occurs particularly at low states of charge and decreases lithium capacity and structural stability [29]. Acidic degradation can worsen these effects, especially under high temperatures ($> 150°C$), leading to oxygen and gas release, active material deactivation and potential safety hazards. [41]

2.2.4 DEGRADATION MECHANISMS INTERACTIONS

The previously described individual degradation mechanisms generally do not act in isolation [15]. Instead, LiB degradation is driven by a complex interplay of electrochemical, chemical and mechanical processes [15]. As shown in **Figure 5**, various causes can simultaneously trigger multiple degradation pathways, which interact and reinforce each other.

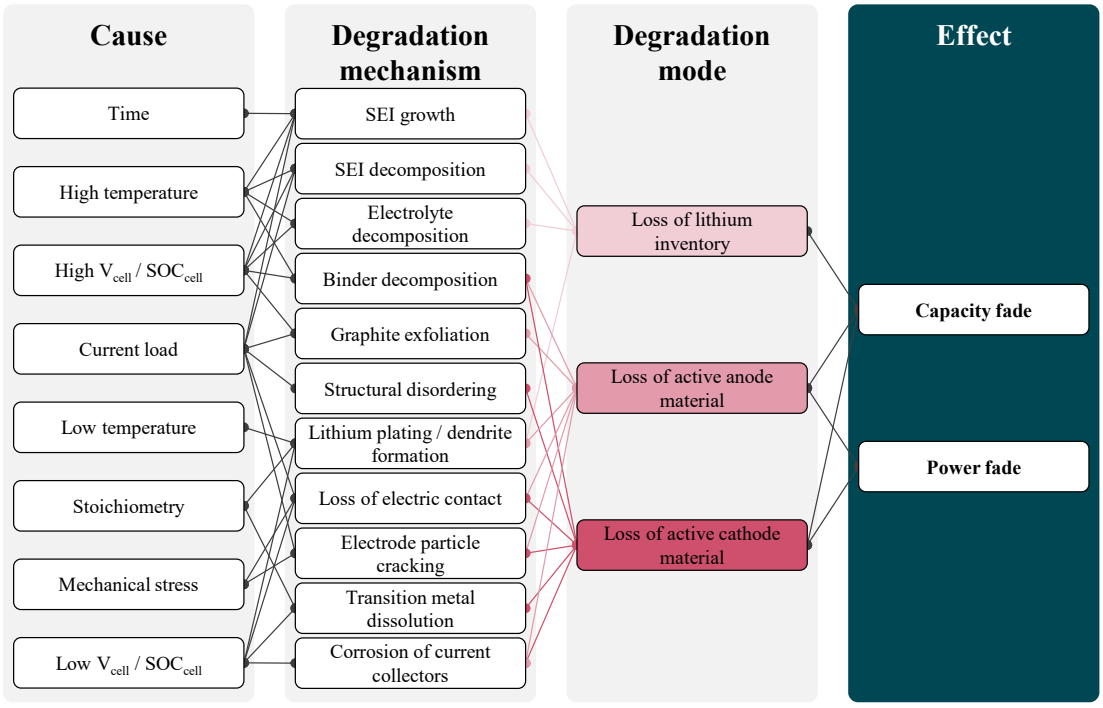


Figure 5: Mapping between degradation causes, mechanisms, modes and effects [15]

For example, particle cracking in the electrodes exposes fresh surfaces to the electrolyte, accelerating side reactions such as SEI or CEI growth. TMD from the cathode not only depletes active material but also leads to metal ion deposition on the anode, where it catalyzes further SEI thickening and lithium plating. These interactions amplify both LLI and LAM, ultimately resulting in measurable declines in capacity and power. In addition, the physical degradation of cell components can change internal resistance, alter lithium transport pathways and shift the electrode stoichiometry. As a result, the observed performance decline is often a combined outcome of multiple, interrelated degradation mechanisms. [15]

These interdependencies complicate the modeling and prediction of battery aging but are essential for understanding system-level performance loss [15]. To capture the functional consequences of internal degradation, the concept of SoH is essential.

2.3 BATTERY STATE OF HEALTH (SOH)

Battery SoH quantifies the measurable outcomes of degradation, specifically capacity and power fade, which arise from the complex interplay of degradation modes and mechanisms. This chapter first introduces standard SoH definitions, followed by an overview of commonly employed estimation approaches, including experimental, model-based and hybrid methods.

2.3.1 SOH DEFINITION

The SoH of a LiB describes its current performance relative to its original condition [9, 38]. As seen in **Chapter 2.2**, battery degradation results in a capacity and a power fade [21]. Central to those effects are a decrease in capacity and an increase in internal resistance, which is why both parameters are typically used to define SoH [14]. The most commonly used metric is based on the reduction in charge capacity over time and is defined as follows [9, 14, 43]:

$$SoH = \frac{Q_m}{Q_r} * 100\% \qquad \textit{Eq. 2.3}$$

Q_m is the current maximum available capacity and Q_r is the initial maximum capacity [9, 14, 43]. This formulation reflects the loss of usable charge storage due to LLI and LAM. A second approach defines SoH based on the change in internal resistance [9, 14, 43]:

$$SoH = \frac{R_e - R}{R_e - R_n} * 100\% \qquad \textit{Eq. 2.4}$$

Here, R is the current internal resistance, R_n is the initial resistance and R_e is the resistance at the end-of-life [9, 14, 43]. This measure describes the increase in internal resistance that affects the transmission of energy by the battery [23]. A third definition is based on the lithium content within the anode, representing the material-level degradation. It is described by the following formula [9, 14, 43]:

$$SoH = \frac{Q}{Q_0} * 100\% \qquad \textit{Eq. 2.5}$$

Q denotes the current lithium capacity within the anode and Q_0 the initial lithium content [9, 14, 43]. This reflects the depletion of cyclable lithium and structural changes within the electrode material. Together, these definitions provide complementary views on battery aging and are selected based on application requirements, available measurements and modeling needs.

2.3.2 SoH ESTIMATION METHODS OVERVIEW

While SoH formulas define how to calculate the state of health of a battery, the parameters required for these calculations are not directly observable. Instead, they must be determined by experimental procedures or inferred from operating data. A variety of estimation methods are employed for this purpose, which are classified into three main categories: experimental, model-based and hybrid methods (see **Figure 6**).

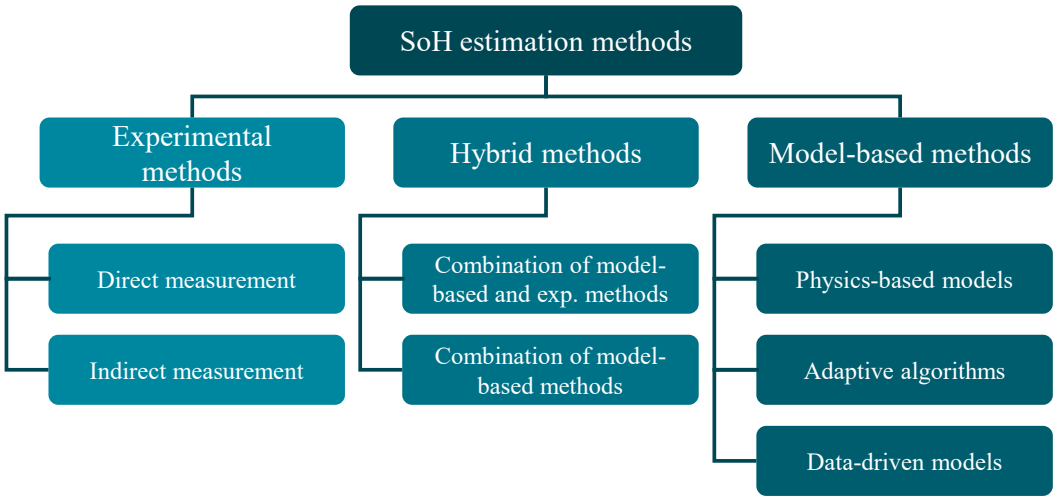


Figure 6: Overview of the three types of battery SoH estimation methods: Experimental, model-based and hybrid

Experimental methods are based on physical measurements, either by direct or indirect observation. Direct measurement methods are generally employed to determine the SoH under controlled laboratory conditions. Typical methodologies include capacity tests,

internal resistance measurements and impedance spectroscopy. These methods are reliable but can be time-consuming and disruptive and making them unsuitable for continuous monitoring. Indirect measurements draw conclusions about the SoH based on voltage and capacity characteristics using techniques such as incremental capacitance analysis or differential voltage analysis. Although these methods are less invasive, their accuracy depends on noise, measurement resolution and operational variability. [23]

Model-based methods are increasingly preferred as they allow for continuous online estimation of SoH. They can be categorized into three categories: physics-based models, adaptive algorithms and data-driven models. Physics-based models simulate the electrochemical behavior of batteries using mathematical formulas. Prominent examples are ECMs and electrochemical models (EMs). They provide information about the internal degradation mechanisms but often require detailed parameterization and considerable computing resources. Adaptive algorithms, including Kalman filters (KF) and particle filters (PF), continuously update the internal model parameters based on new data. This allows them to track changes in battery behavior in real-time and makes them robust against operational fluctuations [23]. Data-driven models rely on statistical learning techniques and are trained on historical or real-time battery data. These models can adapt to different application scenarios, but require large, high-quality data sets to ensure generalizability. [14, 44]

Finally, hybrid methods aim to combine the strengths of experimental and model-based approaches. Commonly, they integrate physical measurements with adaptive or ML models to improve accuracy and robustness. Another form of hybridization combines multiple model-based methods. These combinations can improve both interpretability and predictive power, especially under changing operating conditions. [45]

With the increasing use of LiBs in BEVs, the focus of SoH estimation has shifted to model-based methods. These approaches combine physical accuracy with computational efficiency and real-time applicability. Unlike purely experimental

techniques, model-based methods can work in real-time without interrupting battery operation. In addition, the flexibility to incorporate adaptive behavior or leverage historical data sets allows for improved accuracy under different load conditions. [12]

2.4 MODEL-BASED SOH ESTIMATION METHODS

Due to their wide application and growing importance, model-based SoH estimation methods are the focus of this chapter. In the following, all three types are introduced in detail. The chapter starts with physics-based models, including ECM, EM and empirical models. It then discusses adaptive algorithms, which include Kalman and particle filters. In the final part, the chapter deals with data-driven methods. This last part is divided into statistical data models, rule-based models, classical ML models and artificial neural networks (ANN).

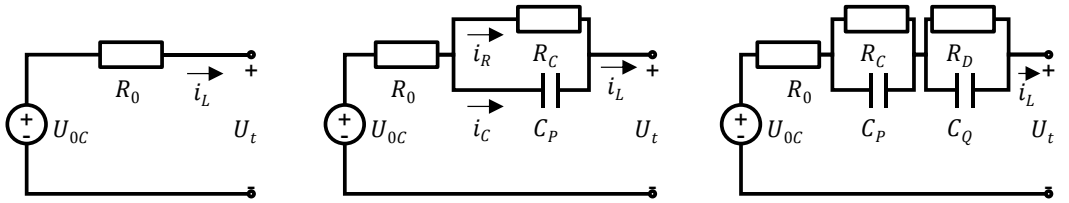
2.4.1 PHYSICS-BASED MODELS

The first category of model-based SoH estimation methods covers physics-based approaches. These approaches construct a mathematical model of the battery to estimate its SoH. The mathematical model includes multiple formulas that describe the electrochemical and physical processes within the battery driving its degradation. Standard methods include ECMs, EMs and empirical models. [9, 45, 46]

2.4.1.1 Equivalent circuit model (ECM)

The first physics-based model is the ECM. It is widely used in BMS due to its ability to simulate key battery behaviors with high computational efficiency [15, 23]. Rather than modeling electrochemical reactions in detail, ECMs abstract them into an electrical circuit consisting of resistors, capacitors and voltage sources [23, 46]. This captures essential characteristics such as open circuit voltage (OCV), internal resistance and transient responses during charge and discharge cycles [15].

Depending on the desired balance between accuracy and computational efficiency, various ECM topologies are employed in literature and industry [9]. Common topologies include the Rint model, Thevenin model and dual polarization model (see **Figure 7**). [15, 38, 47]



(a) Rint model (b) Thevenin model c) Dual polarization model

Figure 7: Common ECM topologies: (a) Rint model, (b) Thevenin model and (c) Dual polarization model [46, 47]

The Rint model is the simplest form of ECMs. It consists of an ohmic resistor (R_0) and an ideal voltage source (U_{0C}), which are connected in series to depict the internal resistance ageing of the battery. The Thevenin model extends the Rint model, adding a parallel RC network that includes the charge transfer impedance (R_C) and capacitance (C_p) to account for polarization characteristics. To further increase the accuracy, the dual polarization model adds an RC series to the Thevenin model, which simulates concentration and electrochemical polarization characteristics. [47]

In ECMs, aging is typically reflected by tracking changes in model parameters over time, e.g., increasing internal resistance or reducing available capacity. [32, 47]. These parameters are identified from voltage-current responses during standard operation using algorithms like recursive least squares or KFs [46]. By mapping these parameters over the battery lifetime, the ECM is capable of inferring the SoH.

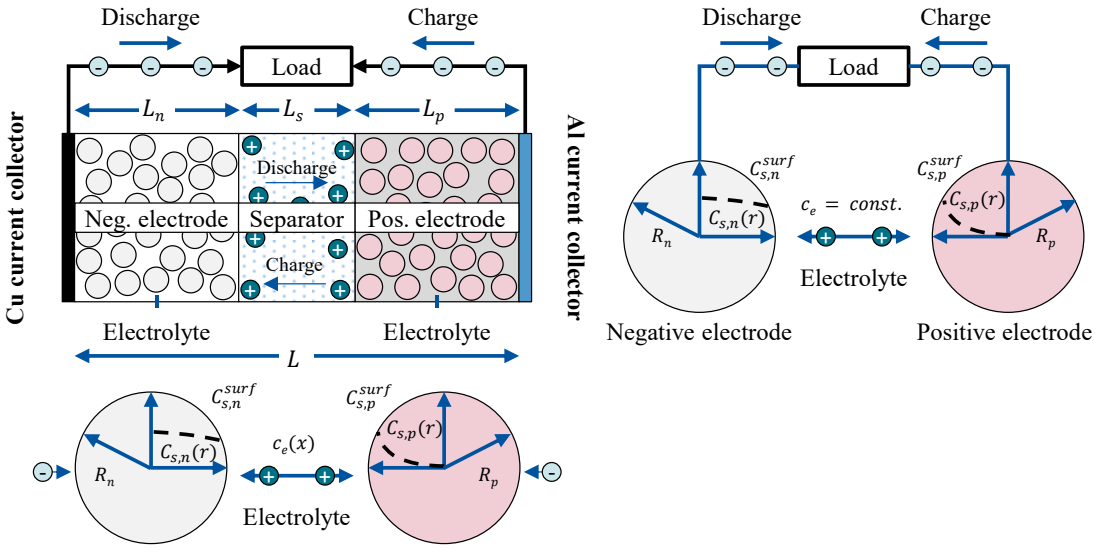
Overall, ECMs are widely applied to estimate SoH due to their physical interpretability, computational efficiency and real-time capability [23, 32]. However, their applicability in high current and low temperature environments is limited [32]. In addition, ECMs

lack robustness and generalization across different battery chemistries and designs, requiring extensive development for each new application [32].

2.4.1.2 Electrochemical model (EM)

EMs are the second category of physics-based approaches. These models simulate the internal chemical, physical and transport processes during battery operation [23, 46]. EMs are rooted in porous electrode theory and reaction kinetics [14]. They use differential equations to describe ion diffusion, charge transfer, heat generation and aging phenomena like SEI formation or lithium plating [9, 32].

Among existing EMs, the pseudo-two-dimensional (P2D) model and the Single Particle Model (SPM) are the most common [32, 38, 47]. Both are shown in **Figure 8**.



(a) Pseudo-two-dimensional model (P2D) (b) Single particle model (SPM)

Figure 8: Common EMs: (a) Pseudo-two-dimensional model (P2D) and (b) single particle model (SPM) [48]

The P2D model provides the most detailed representation of both physical and chemical phenomena. It divides the cell into a negative electrode, separator and a positive electrode [49]. By solving partial differential equations, the P2D model can simulate ion diffusion and migration, electrochemical reactions on the surface of active particles

and several other phenomena [49]. Achieving such accuracy requires solving a large number of differential equations, making P2Ds computationally intensive and therefore impractical for real-time use. To reduce complexity, the SPM simplifies the battery model by treating both electrodes as active particles, assuming a uniform potential of solid and electrolyte and ignoring electrolyte concentration gradients. This makes them more suitable for technical applications, as the calculation costs are reduced. However, accuracy decreases considerably, especially at high C-rates and dynamic loads. [47, 48]

In EMs, ageing is generally captured by internal mechanisms such as LLI and LAM, which evolve the parameters of the model over time [32]. To calibrate the parameters and estimate the internal states of the battery, techniques like genetic algorithms, pseudo-spectral methods and KFs are used [15, 46].

Overall, EMs provide high physical interpretability and precisely depict actual degradation processes. This enables accurate tracking of SoH across diverse conditions. However, EMs are complex, slow to compute and require detailed knowledge of battery chemistry and structure [15, 23, 46]. This limits their use in real-time applications but makes them valuable for in-depth analysis, design optimization and offline diagnostics. [9, 32]

2.4.1.3 Empirical model

Empirical models are the third type of physics-based models. They describe degradation behavior using mathematical functions derived from experimental observations and do not model electrochemical mechanisms in detail [43, 46, 47]. Empirical models rather approximate capacity fade or internal resistance growth as a function of stress factors such as temperature, depth of discharge and cycle number [46]. Two empirical aging models are commonly used: capacity degradation models and internal resistance models [46].

On the one hand, capacity degradation models use the Arrhenius equation, which models temperature-dependent degradation using exponential terms. Extensions of this

model incorporate additional variables such as charge cutoff voltage and discharge rate. On the other hand, internal resistance models describe the growth of internal resistance. Common formulas include single or double exponential functions to reflect nonlinear growth over time. Advanced versions couple empirical resistance models with thermal or electrical sub-models to improve performance under varying environmental conditions. [36, 46]

Overall, empirical models offer structural simplicity, low computational cost and real-time capability. They are particularly effective in stable environments where degradation trends follow known patterns. However, external disturbances and complex environments substantially decrease the models accuracy, making it overall less reliable. [43, 46]

2.4.2 ADAPTIVE ALGORITHMS

The second category of model-based SoH estimation methods are adaptive algorithms. These algorithms optimize the estimation of battery parameters such as SoH based on real-time and empirical data [23, 31]. By continuously adapting the estimation based on the battery's operational performance, they allow for precise SoH estimates [23]. The basis of adaptive algorithms is always a battery model, such as an ECM or EM [10]. Commonly employed methods include KFs, PFs and their extensions [31].

2.4.2.1 Kalman filter (KF)

KFs provide accurate, real-time estimates of the SoH of LiBs even under noisy and uncertain conditions [44, 50]. That is why they are commonly applied in BMS. To achieve these results, the algorithm combines a system prediction with actual measurements in a two-step process (see **Figure 9**) [44].

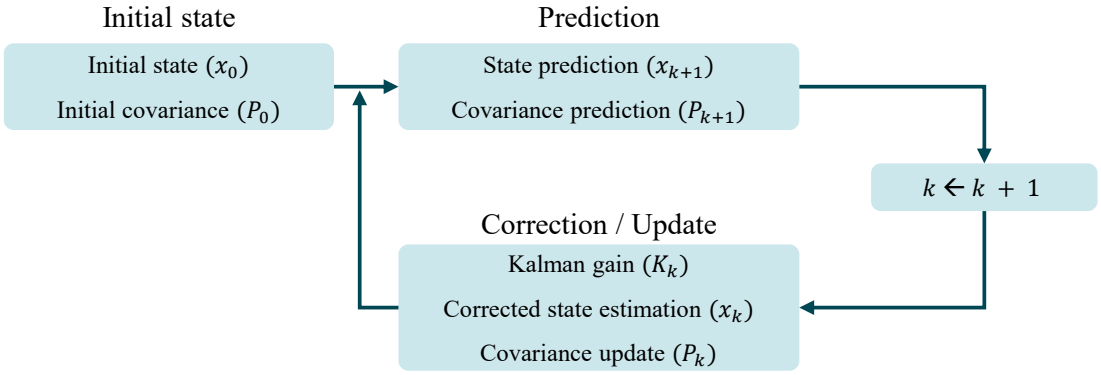


Figure 9: Schematic illustration of the KF recursive steps [44]

As illustrated in **Figure 9,** a KF consists of a prediction and an update step. First, the current state of the battery, including SoH, is predicted based on the initial state of an underlying battery model, which may be of mathematical, electrical or electrochemical nature [10]. In a second step, this state estimate is updated using real-time measurements, to achieve a more accurate result [10].

While the traditional KF assumes linear system dynamics with Gaussian noise, most battery systems are inherently nonlinear, requiring advanced variants such as the Extended Kalman Filter (EKF) or the Unscented Kalman Filter (UKF) [31, 44].

EKFs are the nonlinear version of KFs. Before the prediction step, EKFs linearize the model around the current state estimate using a first-order Taylor expansion [44, 51]. This allows them to estimate battery parameters like capacity and internal resistance with high precision in real-time, even under changing operating conditions [50]. However, the EKF's reliance on linearization can introduce errors in highly nonlinear scenarios and its performance is sensitive to initial conditions and battery model accuracy [50]. Computational cost and tuning effort can also limit its applicability in resource-constrained embedded systems [44, 50].

To address these issues, the UKF offers a more robust alternative [32, 44]. It avoids linearization by using deterministic sampling (sigma points) to better capture the nonlinearities of the system [52]. This results in improved estimation accuracy and

stability compared to EKFs, especially in systems with strong nonlinear behavior [44]. Nevertheless, UKFs are more computationally intensive, making real-time implementation on low-power platforms challenging [44].

Overall, Kalman Filter-based models offer a good balance between accuracy and adaptability, making them well-suited for real-time SoH monitoring in BEV applications. However, their dependence on detailed battery models and significant computational resources presents key limitations. [44, 51]

2.4.2.2 Particle Filter (PF)

PFs are a different approach for dealing with nonlinear systems [38]. Also known as sequential Monte Carlo approaches, PFs are advanced probabilistic techniques used to estimate the SoH of LiBs by representing the system state with a set of particles [50]. These particles collectively approximate the posterior probability distribution of the system's state [23, 31, 32].

The PF operates through a recursive four-step process: initialization, prediction, weight update and resampling. First, a set of N particles is drawn from the prior probability distribution of the battery's state. Each particle represents a possible state of the system and is assigned a uniform initial weight ($w_0 = \frac{1}{N}$). In the prediction step, the next state of each particle is predicted using an underlying battery degradation model. After that, the weight of each particle is adjusted according to the battery's actual state measurement. The weight reflects how well the particle predicts the actual state. Finally, in the resampling step, new particles are randomly selected from the set of weighted particles. The probability of selection is proportional to a particle's weight. Therefore, particles with higher weights are more likely to be selected, while those with lower weights are more likely to be discarded. After the resampling step, the weight of the newly selected particles is reset to a uniform value [53]

This cycle allows the filter to effectively handle both measurement noise and model uncertainties [50]. In addition, PFs are adaptable to different battery chemistries and

models and have demonstrated strong performance in accurately capturing complex battery dynamics and parameter degradation behavior in real-time [50]. However, this robustness comes at the cost of significant computational requirements [32, 50]. Many particles are often required to maintain estimation accuracy, especially in high-dimensional state-space models [38, 50]. This increases the computational burden, making PF implementation challenging for resource-constrained embedded systems [50]. In addition, PFs are susceptible to particle degeneracy, where many particles carry negligible weight over time, reducing diversity and leading to reduced estimation accuracy [32, 50].

2.4.3 DATA-DRIVEN MODELS

Due to their complex internal structure and unpredictable conditions, building an accurate battery model that captures the internal electrochemical and physical processes is highly challenging [9, 14, 23, 32]. Unlike model-based approaches, data-driven models do not require a deep understanding of battery working principles and degradation mechanisms [15, 23]. They are trained on historical battery aging data to accurately predict the battery's SoH [9, 14, 15, 23]. Commonly employed approaches include statistical data models, rule-based models, classical ML models and artificial neural networks (ANNs).

2.4.3.1 Statistical data models

Statistical data models represent the first category of data-driven models. By using probability distribution functions, they cannot only provide an SoH estimate, but also confidence intervals that take into account the uncertainty of the battery degradation process [54]. The most prominent methods are the Gaussian Process Regression (GPR) and the Wiener Process (WP) [14, 54].

2.4.3.1.1 Gaussian process regression (GPR)

GPR is a non-parametric, probabilistic approach that is used to model nonlinear relationships in battery SoH estimation [15, 31, 46]. It is based on Bayesian theory and used under the assumption that the target function follows a multivariate Gaussian distribution, where correlations between data points are encoded by a kernel (covariance) function [15, 55]. This enables GPR to predict SoH values and provide uncertainty estimates in the form of confidence intervals [15, 31, 46].

Initially, a suitable kernel function that captures similarities between data points by mapping nonlinear data into a higher-dimensional space is selected. This transformation allows complex relationships to be more easily analyzed. Next, the initial values of the kernel's hyperparameters are initialized and the prior model is created in the form of a probability distribution, comprising a mean function and a kernel function. The mean function is defined as the expected value of the function for a given input. In the prior model, the mean is often assumed to be zero for simplicity, as the posterior mean naturally becomes non-zero after observing data. After that, the prior model is trained on datasets collected in battery tests or real-world operation to obtain the optimal hyperparameters of the kernel function. The resulting regression prediction model (posterior distribution) is then applied to the test samples to make predictions for new, unseen data points. The final step is the output of these predictions, containing both the mean and variance to express uncertainty. [14, 55]

Overall, the GPR's strength lies in its ability to capture complex aging dynamics without requiring explicit physical modeling. In addition, it provides uncertainty quantification and is suitable for sparse or noisy data scenarios. Nevertheless, GPR is computationally intensive, making it less scalable for real-time or large-scale applications. Its accuracy also highly depends on the choice of kernel function and tuning of associated hyperparameters. [14, 56]

2.4.3.1.2 Wiener process (WP)

Another widely used statistical model for SoH estimation is the WP, which is particularly effective due to its ability to represent uncertainty and irregularities in degradation behavior [14, 54]. Unlike deterministic models, the WP is well-suited to effectively capture nonlinear and non-monotonic patterns of battery degradation [54]. The basic WP uses **Eq. 2.6** to describe battery degradation:

$$X(t) = x_0 + \lambda t + \sigma B(t) \qquad \textit{Eq. 2.6}$$

Here, x_0 represents the initial capacity of the battery. λ is the drift coefficient representing the average degradation rate [57]. σ is the diffusion coefficient that captures random variations and $B(t)$ represents standard Brownian motion, which represents the stochastic dynamics of the degradation process [57]. The WP uses this formula to model both systematic trends in capacity fade and stochastic deviations caused by operational noise or internal battery phenomena. [58]

A key strength of the WP is its ability to represent degradation as a continuous-time stochastic process with a probabilistic structure [59]. This allows it to capture gradual trends and abrupt deviations in battery behavior [54]. Moreover, WPs are able to explicitly model uncertainties, which makes them suitable for real-world applications where measurement noise, operational variability and incomplete observations are common [60]. However, parameter estimation remains a key challenge. The drift and diffusion coefficients must be accurately determined via methods such as maximum likelihood estimation or expectation maximization. Both approaches can be computationally intensive, especially if nonlinear behavior or measurement errors are taken into account. [61]

2.4.3.2 Rule-based models

Rule-based models are a second distinct category of data-driven models. In contrast to statistical or ML approaches, rule-based methods encode domain knowledge directly

into their structure through expert-defined *if-then* rules [62]. Among these methods, Fuzzy logic (FL) is the most prominent.

FL is particularly suitable for systems with nonlinear, imprecise or incomplete data [63]. It does not require an underlying model of the battery system [38]. Instead, it uses linguistic rules and membership functions to map inputs such as temperature, current and voltage into SoH estimates [38]. This enables the abstraction of system equations and implementation based on empirical observations or expert knowledge. **Figure 10** illustrates the structure of an FL system.

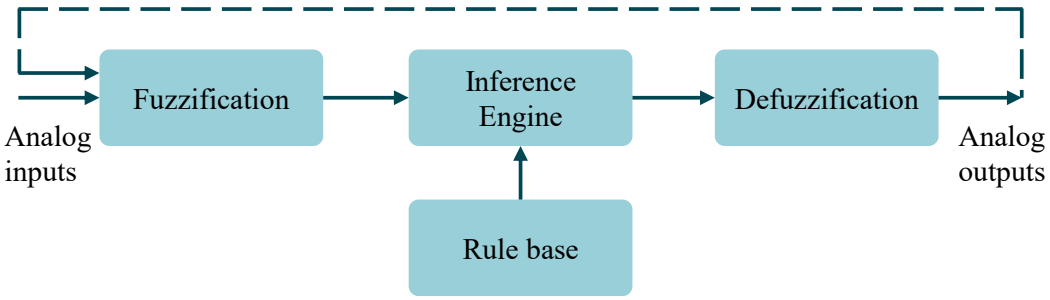


Figure 10: Components of an FL system [63]

A FL system consists of four main components: fuzzification, rule base, inference engine and defuzzification [38]. During fuzzification, crisp numerical inputs are converted into fuzzy sets, consisting of multiple fuzzy variables that are assigned a degree of membership (a value between 0 and 1) by a membership function. The degree of membership describes the affiliation of the variable to a linguistic term (e.g., the temperature is 'warm'). Examples of membership functions include triangular, trapezoidal and Gaussian functions. The rule base comprises a set of expert-derived or data-driven *if-then* rules that link inputs to outputs (e.g., "*if temperature is warm and discharge rate is high then output is low*" [64]). The inference engine applies fuzzy reasoning, commonly using either the Mamdani or Sugeno method, to evaluate these rules. Finally, defuzzification converts the fuzzy output into a single crisp value using output membership functions. [63, 64]

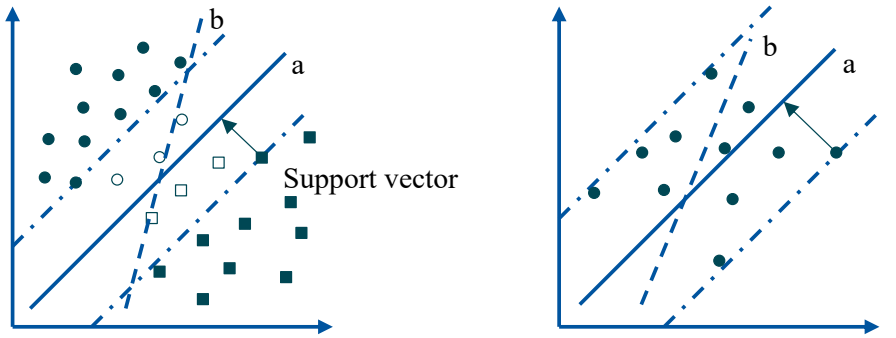
Overall, FL offers high flexibility in modeling complex, nonlinear battery behavior without reliance on mathematical models or ECMs [38, 63]. In addition, it provides good estimation accuracy, especially with well-designed *if-then* rules and membership functions [43]. However, its performance strongly depends on the quality of these rules and functions [43]. Finally, FL models are computationally intensive due to parallel rule evaluation and require expert input for initial rule design or tuning [7, 38].

2.4.3.3 Classical machine learning models

Classic ML models are trained to recognize patterns in data, regardless of the underlying physical and electrochemical processes [38]. By analyzing historical battery data, they learn relationships between measurable characteristics such as voltage, current or temperature and the SoH of the battery [38]. To extract relevant indicators from the raw data that capture battery degradation, manual feature engineering is usually required [65]. Commonly used algorithms include support vector machines (SVM), k-nearest neighbors (kNN), decision trees (DT) and ensemble learning methods.

2.4.3.3.1 Support Vector Machine (SVM)

SVMs are widely used for battery SoH estimation because of their ability to handle nonlinear and high-dimensional data [9, 50]. SVMs are supervised learning algorithms which are primarily used for classification problems. Regression problems are handled by support vector regression (SVR), a variant of the conventional SVM [66]. The fundamental principles are shown in **Figure 11**.



(a) Support vector machine (b) Support vector regression

Figure 11: Geometric representation of SVM and SVR [38]

The core concept of an SVM involves determining an optimal hyperplane that differentiates between data from different classes (see **Figure 11a**) [31, 50, 66]. The optimal hyperplane is selected by maximizing the distance between the hyperplane and the closest data point of any class [44, 50]. In the example, hyperplane "*a*" outperforms hyperplane "*b*" because it effectively separates the two classes. To construct a hyperplane, a support vector is required. It represents the distance between the nearest sample point and the data plane. SVR follows the same principle (see **Figure 11b**), however, it aims to minimize the distance of the farthest sample point from the hyperplane. [38]

To estimate a battery's SoH the SVM model is trained using historical monitoring data and offline test results [23]. During training, the model builds an optimal hyperplane, which enables it to estimate SoH in real-time using parameters such as internal resistance and capacity. A key component of this approach is kernel functions. They map the input features in a higher-dimensional feature space, which allows the SVM to capture complex, nonlinear relationships by treating them as linear. [31, 38, 50]

Although SVMs are very accurate and resistant to overfitting, they are sensitive to parameter settings and kernel selection [44, 50]. On the other hand, their ability to handle large datasets is limited because memory usage increases with the number of support vectors [43, 44]. Furthermore, the resulting models often lack interpretability,

limiting their ability to provide deeper insight into battery degradation mechanisms [50].

Relevance Vector Machines (RVMs) have emerged as a response to these limitations [31, 43]. RVMs are less reliant on the kernel function and require only a few correlation vectors as they naturally create sparse solutions based on Bayesian inference [31, 43]. This significantly simplifies the model and reduces computational cost [31]. However, RVM performance can decrease if the input data is sparse or noisy, potentially affecting consistency and repeatability [43].

In conclusion, SVM-based approaches are robust tools for estimating the SoH of LiBs, providing strong generalization abilities and effectively handling nonlinear dynamics [44, 50]. However, model performance depends heavily on kernel selection, feature engineering and hyperparameter tuning [9, 50]. Additionally, for high-dimensional feature spaces and large datasets, SVMs are computationally demanding [50].

2.4.3.3.2 K-nearest neighbors (kNN)

The kNN algorithm is an instance-based learning method used to estimate the SoH of LiBs [56]. It operates based on the principle that the battery's SoH can be inferred from the most similar past instances in a historical dataset [38]. A geometric representation of this approach is shown in **Figure 12**.

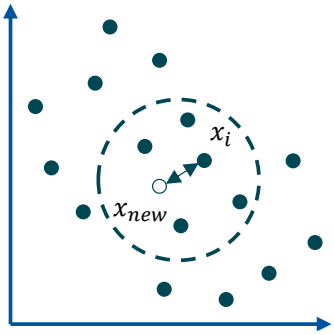


Figure 12: Geometric representation of a kNN [38]

In this method, the SoH is estimated by identifying the 'k' most similar instances (neighbors) based on selected features such as voltage, current, temperature and capacity [3, 38]. This requires a dataset containing labeled instances, where the features are paired with known SoH values from batteries tested under similar operating conditions. The neighbors are identified by calculating the distance between the current sample and instances in the historical dataset [56]. Typical metrics include the Euclidean and Manhattan distance [56]. The SoH values of the nearest neighbors are then aggregated, typically through averaging, to predict the SoH of the current battery instance [38, 56].

kNN models offer several advantages. They are conceptually simple and able to capture complex, nonlinear relationships without requiring an explicit model of the underlying battery degradation processes [3, 56]. This makes them suitable for applications where the degradation mechanisms are not fully understood or are too complex to model analytically.

However, the performance of kNN models strongly depends on the quality and representativeness of the historical dataset, as accurate SoH estimation requires the availability of sufficient similar samples [3, 38]. This dependency increases computational demands, as the algorithm requires calculating distances between the query instance and all instances in the dataset [38, 56]. Finally, kNN models are sensitive to noise and outliers, which requires thorough data preparation and careful selection of features [56].

2.4.3.3.3 Decision tree (DT)

DTs are another class of supervised learning methods. They are a nonparametric technique with a hierarchical structure that contains the following elements: root node, branches, internal nodes and leaf nodes (see **Figure 13**). [67]

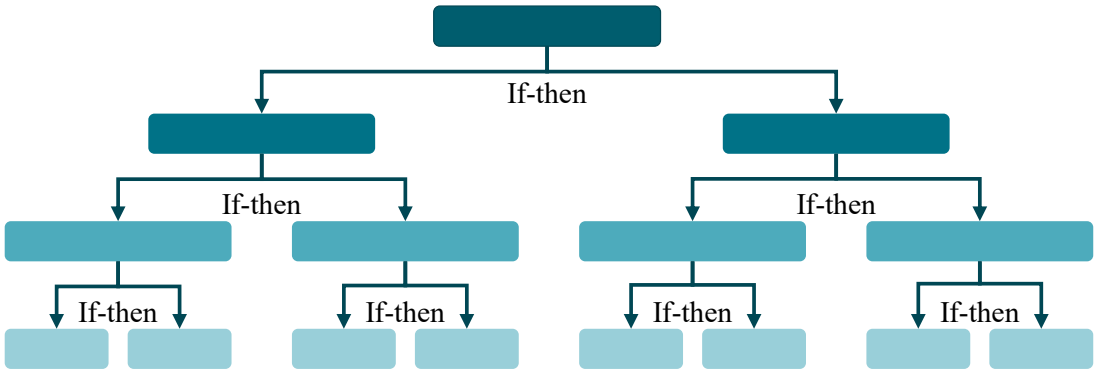


Figure 13: Representative structure of a decision tree [67]

The tree begins at the root node, which represents the starting point of the decision-making process. From the root node, branches guide through the DT, representing the different decisions. Internal nodes correspond to the various decision rules, while leaf nodes exhibit the different outcomes from the dataset. [67]

In the context of battery SoH estimation, relevant features from battery operational data have to be determined, such as voltage, current, temperature and capacity measurements [56]. These features serve as input, enabling decision-making within the DT. Finally, each leaf node provides a different SoH estimate of the battery, allowing for an accurate health assessment.

DTs offer several advantages, including simplicity, interpretability and ease of implementation. Furthermore, DTs are adaptable and perform well under varying conditions, effectively modeling complex, nonlinear and multivariate relationships. However, they are prone to overfitting, especially when the tree becomes too deep and captures noise in the training data. This can lead to poor generalization on unseen data. Moreover, model performance is sensitive to the initial parameter selection and often requires extensive experiments and tuning. As the dataset grows and relationships become more complex, the computational cost and training time also increase. [56]

2.4.3.3.4 Ensemble Learning Methods

Ensemble learning methods are increasingly used for SoH estimation due to their ability to model complex degradation behaviors with high accuracy and generalization capability. They combine multiple base models, such as DTs, to produce a more robust and accurate output. Two commonly used ensemble methods are Random Forests (RF) and Gradient Boosting Machines, such as XGBoost and AdaBoost. [68, 69]

RF is a bagging algorithm that constructs numerous DTs on randomly sampled subsets of the training data and averages their outputs (see **Figure 14**) [69]. It can directly utilize raw input features such as voltage, current and time, eliminating the need for complex preprocessing, which reduces computational cost. [14, 32]

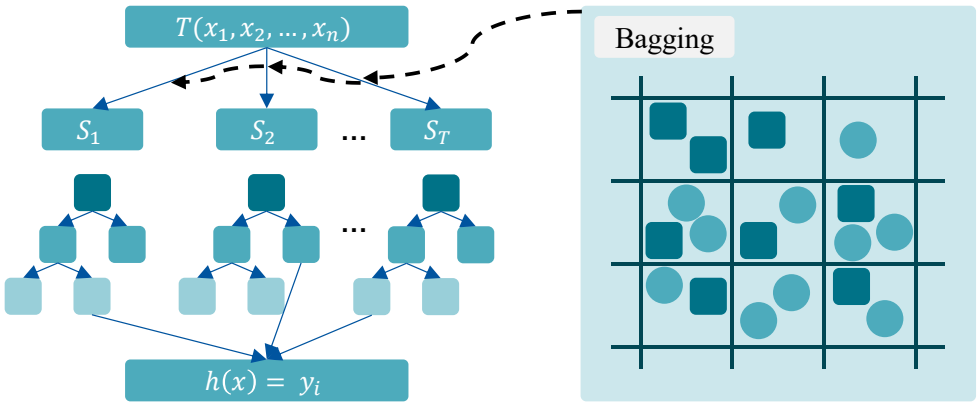


Figure 14: Schematic diagram of the RF [69]

Developing an RF includes four steps. First, the original data is resampled and split into T sample sets with dedicated labels $\{S_1, S_2, \ldots, S_T\}$. Second, each sample is used to construct a corresponding DT. To assign attributes to the decision nodes, k attributes are sampled from M attributes. This introduces diversity among the trees, preventing strong correlation. The attributes are then split via the CART algorithm to partition each node optimally. The outcome is the RF model $h(x) = y$. Third, each regression tree is calculated, determining a multitude of predicted values. Finally, all of these values are averaged, resulting in the overall outcome of the RF model. [69]

RFs offer several important advantages in battery SoH estimation. They exhibit high accuracy even without a large number of parameters and little risk of overfitting [69]. Also, they do not require feature selection and provide strong generalization ability [32, 69]. However, compared to other methods RFs, have longer operation times. [14]

In contrast, boosting algorithms like XGBoost and AdaBoost build trees sequentially, where each tree corrects the errors of its predecessors [68]. XGBoost introduces regularization and second-order optimization to improve prediction accuracy and prevent overfitting, making it suitable for large datasets [14, 32]. AdaBoost adjusts weights on training samples based on previous errors, which improves sensitivity to challenging patterns but can make it vulnerable to noise [6]. Both methods offer accurate SoH estimation but require careful parameter tuning and are computationally less efficient compared to simpler models [32].

In summary, ensemble methods are practical tools for battery SoH estimation, offering high prediction accuracy and generalization ability. However, they are dependent on large datasets and require extensive computational resources. This limits their use in resource-constrained BMS environments.

2.4.3.4 Artificial neural networks (ANN)

ANNs are the fourth category of data-driven models. They present a specific type of ML that is inspired by the structure and function of the human brain [32, 50, 66]. ANNs consist of layers of interconnected nodes (neurons) that process input features such as voltage, current, temperature and charge cycles to estimate the battery's SoH [50]. Compared to classical ML methods, ANNs are capable of automatically learning features from raw data. They are trained using large historical datasets to learn to predict a battery's SoH for unseen operating conditions. The most prominent models include feedforward neural network (FFNN), convolutional neural network (CNN) and recurrent neural network (RNN).

2.4.3.4.1 Feedforward neural network (FFNN)

FFNNs represent the simplest architecture of ANNs. In their structure, information flows unidirectionally from the input layer via all hidden layers to the output layer, without forming cycles or feedback loops (see **Figure 15**). Due to their ability to model complex nonlinear relationships, FFNNs are widely applied in BMS for tasks such as SoH estimation. [70]

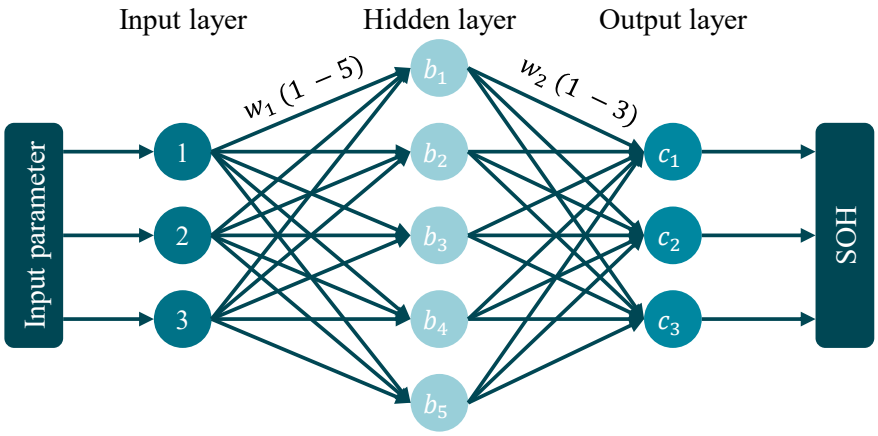


Figure 15: Structure of a three-layer FFNN [14]

As shown in **Figure 15**, an FFNN in its simplest version consists of an input layer, a hidden layer and an output layer. Initially, a set of input variables (e.g., voltage, current, temperature) is selected and fed into the input layer [45]. These inputs are then passed to the hidden layer, where each neuron computes a weighted sum of its inputs, followed by the application of an activation function to introduce nonlinearity. During training, the network adjusts the utilized weights through backpropagation and gradient descent to minimize the prediction error. Finally, the resulting values are transferred from the hidden layer to the output layer, providing the overall SoH estimate. [71]

Its simple architecture is a key advantage of FFNNs, as it enables straightforward implementation and training. They are particularly effective at predicting the state of a battery based on static input data. However, FFNNs have several limitations. Most notably, they struggle to capture time-dependent or sequential patterns in data, which

may reduce their effectiveness in managing the dynamic nature of battery degradation. In addition, training FFNNs often involves computationally expensive iterative optimization and hyperparameter tuning, hindering rapid deployment. Additionally, FFNNs cannot automatically extract and learn from raw sensor data, often requiring manual feature engineering. [70]

2.4.3.4.2 Convolutional neural network (CNN)

CNNs represent a second class of ANNs. They are designed to automatically extract hierarchical features from structured grid-like data by applying convolutional layers (see **Figure 16**). CNNs have become increasingly popular for SoH estimation due to their ability to process time-series data sampled at regular intervals. This allows them to detect subtle degradation trends without the need for manual feature engineering. [26]

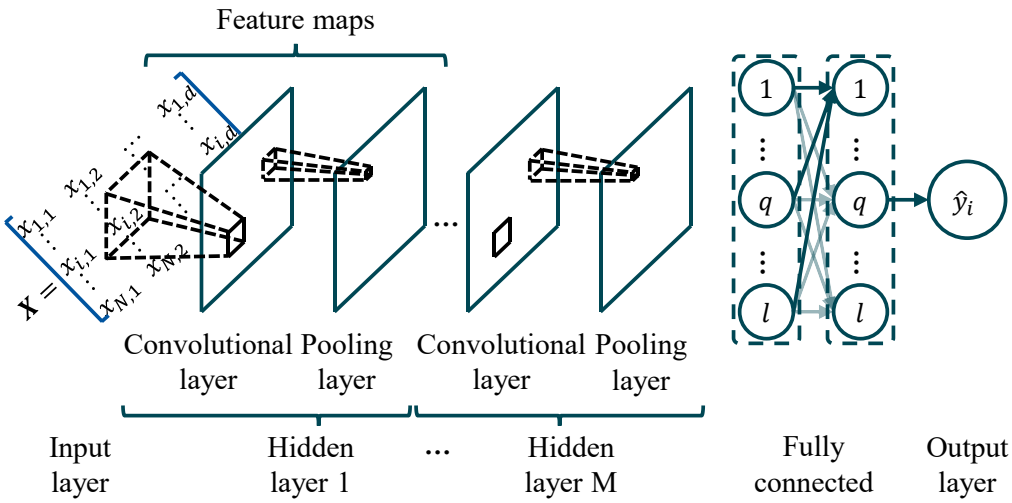


Figure 16: Structure of a CNN [3]

As shown in **Figure 16**, CNNs comprise several components. The convolutional layer applies kernels that scan the input data, producing feature maps that capture local patterns relevant to battery degradation. Within the convolutional layer, these maps are passed through an activation function. This function introduces nonlinearity, enabling the model to learn complex relationships. Subsequently, a pooling layer reduces the

dimensionality of the feature maps while preserving essential information. Common pooling operations include max pooling and average pooling. This sequence of convolution and pooling is repeated across multiple hidden layers, depending on the depth of the architecture. The output of the last hidden layer is passed to the fully connected layer. This layer integrates the extracted features and performs regression to estimate the battery's SoH. The final output layer then provides the SoH prediction, including a probability distribution. [3, 4, 26, 72]

CNNs offer several advantages. They provide both high prediction accuracy and fast inference times [38]. In addition, their end-to-end learning capability eliminates the need for manual feature engineering and their local perception mechanism and parameter sharing mitigate overfitting. CNNs can also directly learn from raw data, which reduces reliance on domain expertise and enables consistent model performance across different battery types and usage scenarios. However, their performance depends on careful architecture design, including the number of layers, kernel sizes and pooling strategies. In addition, substantial amounts of training data are required to achieve optimal accuracy and generalization [70]. Finally, CNNs are less effective at capturing long-term temporal dependencies, which leads to suboptimal performance when analyzing extended time series data. [26, 32]

2.4.3.4.3 Recurrent neural network (RNN)

RNNs represent a third class of ANNs. Derived from FFNNs, they are specifically designed for modeling sequential data, which makes them well-suited for applications involving time-dependent patterns [3, 46, 70]. They distinguish themselves from FFNNs by incorporating feedback connections, which enable them to maintain a memory of previous inputs through internal hidden states (see **Figure 17**) [70]. At each time step, the RNN's output is estimated using both the current input and previous hidden state [3, 70]. This ability is crucial for battery diagnostics, since historical usage has a significant impact on current performance. [26]

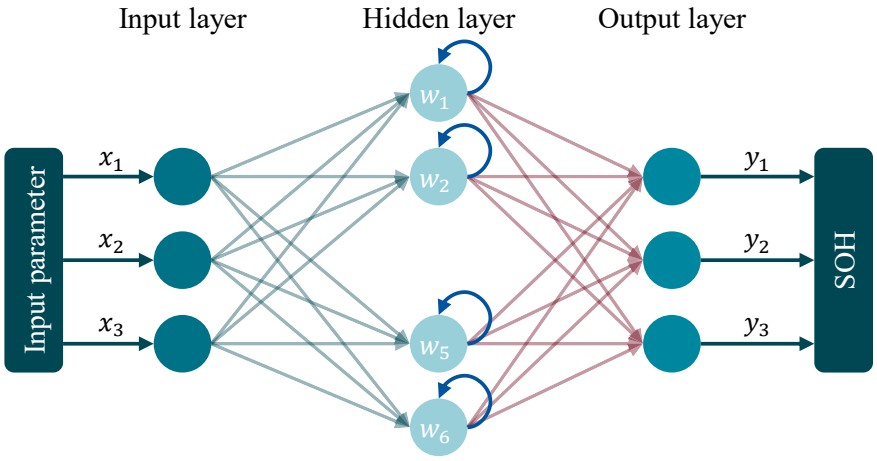


Figure 17: Structure of an RNN [32]

As shown in **Figure 17**, an RNN has a similar architecture to FFNNs. It consists of an input layer, a hidden layer and an output layer. However, RNNs incorporate recurrent connections in the hidden layer, which enables the network to carry information across time steps. This allows RNNs to utilize the previous hidden state when estimating the current state. [3, 4, 26]

A key advantage of RNNs is their ability to process and model time series data due to their feedback connection. Additionally, the weights of each connection stay constant across all time steps. This significantly reduces the number of model parameters and thereby the overall computational resources required. However, standard RNNs are limited in their ability to model long-range dependencies due to the vanishing and exploding gradient problems during training [46, 70]. These challenges reduce the network's capacity to retain information across long periods, which is critical for capturing the gradual degradation behaviors of battery systems. To address these challenges, more advanced RNN variants have been developed, including Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs). [3, 26]

LSTMs address the vanishing gradient problem. Their architecture consists of a memory cell that is regulated by three gating mechanisms: the input gate, forget gate and output gate [38, 70]. These gates control the flow of information into, within and

out of the cell, enabling the selective retention or discarding of temporal information [38]. This enables the model to efficiently store, update and manage information across sequences, which allows for robust modeling of temporal dependencies. [26]

In the context of battery SoH estimation, LSTMs are particularly valuable due to their ability to model time-varying and nonlinear behaviors [38, 45]. In addition, LSTM-based models have demonstrated high accuracy in predicting battery degradation [44]. However, the computational complexity increases with sequence length and hidden layer size [44]. Additionally, overfitting can occur when the model architecture is overly complex relative to the available training data [44].

GRUs offer a simplified structure compared to LSTMs, which reduces the computational complexity significantly. They consist of only two components: a reset and an update gate. GRUs merge the input and forget gates of a classic LSTM into a single update gate and combine the hidden states and cell states. This results in fewer parameters and reduced computational requirements. Despite their simpler architecture, GRUs often achieve performance comparable to LSTMs, which makes them attractive for SoH estimation, particularly in BMS where, computational resources are limited. [26]

In summary, RNN-based architectures are effective tools for estimating battery SoH. This is attributable to the fact that they can model the sequential nature of battery degradation by incorporating feedback loops. However, the complexity of the model, the required training data and the necessary computational resources must be carefully considered to use them effectively. [44]

2.5 INTERIM SUMMARY

LiBs are widely used in BEVs due to their high energy density, long life and favorable performance characteristics under diverse operating conditions. However, LiBs are subject to degradation over time. This degradation is caused by a complex interplay of

several degradation mechanisms, mainly resulting in two modes: LLI and LAM. These phenomena contribute to a capacity and a power fade, the two primary effects that reduce battery performance over time.

To extend battery life, ensure safe and reliable operation and maintain performance, BMSs are used. A key function of the BMS is to estimate the SoH of the battery, which indicates the current state of health and estimated remaining useful life. The SoH is typically characterized by parameters such as the remaining capacity and the internal resistance in relation to their initial values. However, these parameters are not directly observable; therefore, different methods are employed to determine them.

A variety of methods are used to estimate these parameters, which can be divided into three categories: experimental, model-based and hybrid methods. Among them, model-based approaches have gained prominence due to highly accurate, non-invasive SoH estimation during regular operation, which makes them ideal for real-time integration in BMS. Model-based approaches include physics-based models, adaptive algorithms and data-driven methods.

Given the diversity of model-based approaches and their respective strengths and weaknesses, a comprehensive selection framework is crucial. Such a framework enables a systematic comparison of these methods and thereby facilitates the identification of the most appropriate approach for specific application requirements and operational constraints. The following chapter develops this framework by applying key evaluation criteria to support application-oriented decision making in the context of battery SoH estimation.

Chapter 3. SOH METHOD SELECTION

To systematically assess and compare the in **Chapter 2.4** presented model-based SoH estimation methods, this chapter develops a comprehensive evaluation framework. It is structured into two main parts.

The first part focuses on the development of the mentioned framework. In **Chapter 3.1.1** a structured literature review identifies five relevant studies that utilize multiple criteria to assess model-based SoH estimation methods. From these studies, six comprehensive criteria are derived that capture key performance and implementation aspects of said methods. **Chapter 3.1.2** then defines the selected criteria and introduces corresponding 5-level scoring scales. **Chapter 3.1.3** describes the structural implementation of the defined framework, including the evaluation process and the interpretation of results.

In the second part, the proposed evaluation framework is applied across the SoH estimation methods described in **Chapter 2.4**. Subchapters **3.2.1** to **3.2.6** each address one of the six evaluation criteria, systematically rating the methods based on the defined 5-level scale. **Chapter 3.2.7** consolidates the results across all criteria and identifies the three most promising approaches for further analysis based on the boundary conditions given in this thesis. The selected methods are carried forward to **Chapter 4** and **Chapter 5**, where they are applied to real-world datasets. The section concludes with **Chapter 3.3**, summarizing the key findings of the overall chapter and setting the stage for the subsequent practical implementation.

3.1 EVALUATION FRAMEWORK CONSTRUCTION

This section presents the structured development of the comprehensive evaluation framework introduced at the beginning of the chapter. It consists of three sequential

steps. First, relevant evaluation criteria are identified through an extensive literature review. Then, each criterion is formally defined and assigned a 5-step scoring scale, before the structural implementation of the framework is described.

3.1.1 EVALUATION CRITERIA DERIVATION

To define a comprehensive set of evaluation criteria for model-based SoH estimation methods, a structured literature review was conducted. From this review, five representative studies were selected for further analysis and comparison. These studies were chosen because they employ multi-criteria assessment frameworks rather than focusing on one single performance metric. This offers a strong foundation for identifying a set of practically relevant criteria for selecting an appropriate SoH estimation method for a given application. While several other relevant studies exist, most of them repeat the criteria covered in the selected works. Therefore, only these five studies were included in the analysis. They are presented in the paragraphs below.

In [3], *Sui et al.* conducted a comparative analysis of multiple ML algorithms using five evaluation criteria: estimation accuracy, implementation easiness, computational complexity, training data size requirements and overfitting. They are defined as follows:

- **Estimation accuracy**: Considers both the training error on known data and the generalization error on unseen data.
- **Ease of implementation**: Refers to required computational resources and the extent of manual feature extraction.
- **Computational complexity**: Evaluates the approach's overall memory usage.
- **Data requirements**: Measures the volume of data needed for development and training, independent of input dimensionality.
- **Overfitting**: Examines the model's tendency to closely fit limited training data, while estimation results on unseen data are poor.

In [13], *Oji et al.* proposed a broader set of seven evaluation criteria, which they apply to multiple model-based SoH estimation methods. These criteria are defined as follows:

- **Accuracy**: Assesses the degree to which the models predicted SoH values align with the true values.
- **Confidence Interval**: Evaluates whether the method provides a probabilistic output in the form of a confidence interval, which provides a range in which the true SoH value is expected to fall.
- **Ability to deal with nonlinearity**: Examines the method's capacity to model the strongly nonlinear battery degradation process.
- **Robustness**: Measures the model's performance and accuracy on data with a certain amount of noise.
- **Computation complexity**: Assesses the inference latency and memory requirements during method deployment.
- **Capability to deal with sparse data**: Evaluates the model's performance, accuracy and computation time on incomplete datasets.
- **Generalization**: Determines the ability of an approach to deal with unseen datasets or different battery types without extensive retraining.

In [1], *Eleftheriadis et al.* applied five criteria in their assessment of model-based SoH estimation methods, including accuracy, time to tune, interpretability, flexibility and simplicity. They are defined as follows:

- **Accuracy**: Examines how closely the predicted SoH values align with measured values.
- **Time to tune**: Assesses the model's total processing time, consisting of training and test time.
- **Interpretability**: Assesses how easily experts can understand the model structure and explain its outputs.

- **Flexibility**: Evaluates the model's adaptability to new data, including its ability to incorporate updates and process big data.
- **Simplicity**: Measures the algorithm's structural complexity based on the number of parameters and their respective value range.

In [20], *Guo and Ma* applied five evaluation criteria to assess ANN-based SoH estimation methods: accuracy at a fixed temperature, algorithm generalization, robustness against noise, generalization to different battery materials and computing burden. These are described as follows:

- **Accuracy at a fixed temperature**: Evaluates the prediction accuracy of the models when trained and tested on data collected at 25 °C.
- **Algorithm generalization**: Assesses the predictive performance under varying temperature conditions.
- **Robustness against noise**: Examines model performance in the presence of random sensor noise, simulating real-world conditions.
- **Generalization to different battery materials**: Assesses the model's transferability by applying the same network structure to a different battery type and retraining it using new datasets collected from the new chemistry.
- **Computing burden**: Evaluates the calculating time that each method takes to perform the SoH estimation.

Finally, in [50], *Dar and Singh* analyzed model-based SoH estimation methods using five criteria: accuracy and precision, computational complexity, real-time capability, robustness to environmental factors and model requirements. These are defined as follows:

- **Accuracy and precision**: Measures how accurately and consistently the predicted SoH values reflect true values.
- **Computational complexity**: Assesses resource demands during both model training and deployment.

- **Real-time capability**: Evaluates whether the model is suitable for real-time use, considering computational speed and complexity.
- **Robustness to environmental factors**: Tests model performance under variable conditions, including noise and environmental changes.
- **Model requirements**: Examines the prerequisites needed for model implementation, such as the necessity of a physical battery model, extensive training or measurement data or the use of optimization functions.

Based on the analysis of the studies presented above, six evaluation criteria are identified that comprehensively capture the essential dimensions for assessing model-based SoH estimation methods (see **Figure 21**). These criteria are accuracy, computational efficiency, interpretability, data requirements, reliability and scalability. They were derived by prioritizing aspects that are both frequently cited and methodologically significant across all sources.

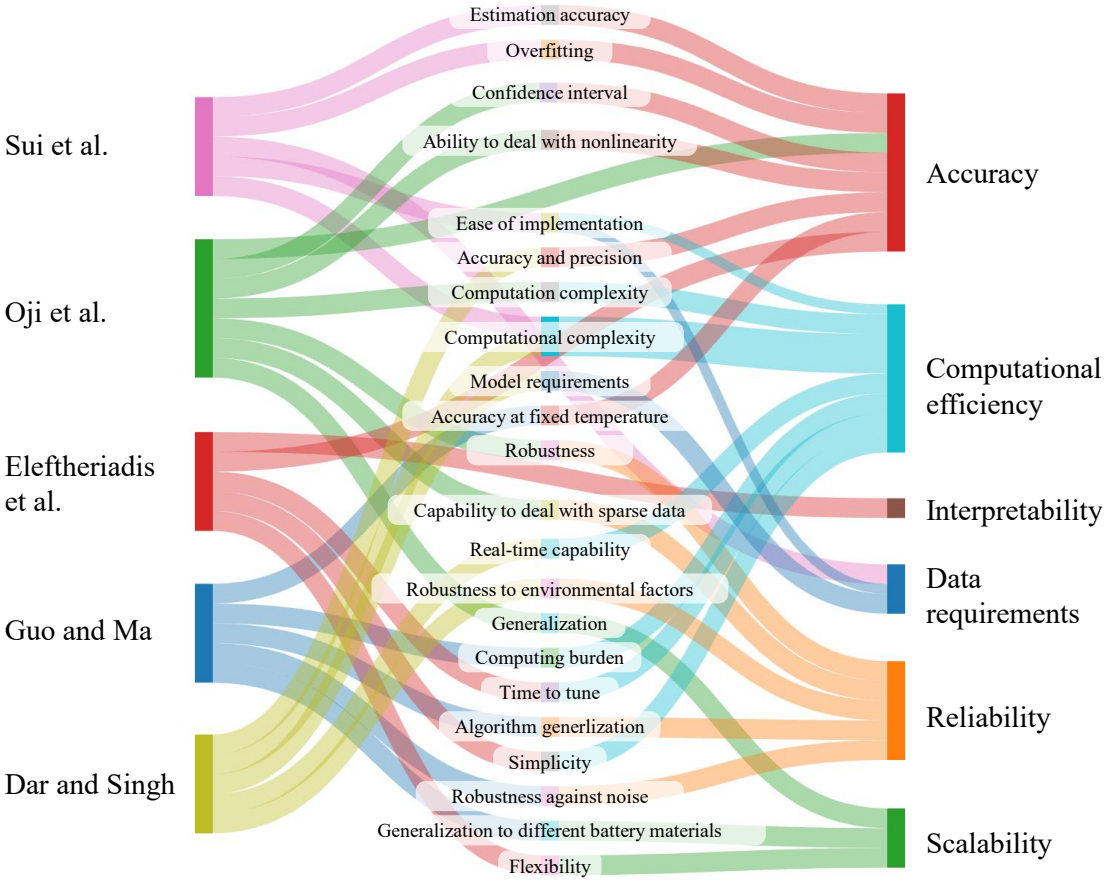


Figure 18: Derivation of six evaluation criteria from five selected approaches found in the literature that apply evaluation frameworks to model-based SoH estimation methods

Among the reviewed studies, accuracy and computational efficiency are the most consistently emphasized evaluation categories. The **accuracy** criterion integrates not only the predictive precision of the model on both training and test data, but also covers its capacity to manage overfitting, provide confidence intervals and handle nonlinearity. These aspects are consolidated under a single criterion, as they collectively contribute to the overall accuracy of the model. Similarly, **computational efficiency** encompasses several subdimensions, including computational resource demands, structural complexity, training and tuning time, inference latency, real-time applicability and memory requirements.

While referenced less consistently, additional criteria such as data requirements, reliability and scalability are also frequently cited and thus incorporated into the framework. **Data requirements** address the volume and quality of data necessary for model development and training, including the dependence on manual feature engineering or a physical battery model. **Reliability** combines resilience to sensor noise, performance under varying environmental conditions and the ability to operate effectively with incomplete or sparse datasets. **Scalability** captures the model's generalization capability to unseen data and different battery chemistries. While not explicitly mentioned, the adaptability to increasing dataset size and system complexity is added to this criterion, as it is an essential characteristic in the context of large-scale battery systems.

Finally, although mentioned only in one of the reviewed studies, **interpretability** is included as a sixth criterion due to its critical relevance in practical applications. It reflects the extent to which the model's structure and outputs can be understood and explained by humans, directly impacting the level of expert knowledge required for development, validation and deployment.

Collectively, these six criteria form a comprehensive framework for the evaluation of model-based SoH estimation methods, which is further detailed in the following chapter.

3.1.2 EVALUATION CRITERIA DEFINITION AND SCORING

Within this chapter, the six previously identified evaluation criteria are formally defined and accompanied by a corresponding five-point scale. These scales later serve to evaluate each of the presented SoH estimation approaches in comparatively in **Chapter 2.4**.

3.1.2.1 Accuracy

Accuracy assesses how closely the SoH values predicted by the model match the actual or reference SoH values [27, 70]. This reflects the main predictive capability of the model and is essential for ensuring safe and reliable battery management [1]. It includes the model's ability to deal with high-dimensional input spaces and nonlinear relationships in battery behavior, as models that can capture these characteristics typically achieve higher prediction accuracy [51]. In addition, it evaluates a model's predictive performance on unseen test data drawn from the same distribution as training data, reflecting the model's ability to generalize and avoid overfitting. [13, 50]

Based on this definition, the following scale is defined to categorize the different model-based SoH estimation methods. It is presented in **Table 1**.

Scale	*Description*
Very High	Excellent accuracy on unseen test datasets; robust handling of nonlinear, high-dimensional data; minimal to no overfitting
High	Strong accuracy across unseen test datasets; effectively manages nonlinearity and high-dimensionality, rare overfitting
Moderate	Acceptable accuracy on training datasets; partial ability to deal with nonlinear, high-dimensional data; some overfitting
Low	Limited accuracy on training datasets; struggles with nonlinearity and high-dimensionality; prone to overfitting
Very low	Predictions consistently deviate from true SoH; no ability to manage nonlinearity/high-dimensionality; frequent overfitting

Table 1: Scoring of the accuracy criterion

3.1.2.2 Computational efficiency

Computational efficiency measures the resource demand of a method during both training and inference, directly affecting its feasibility for real-time applications [30,

51]. It considers training time and inference latency as well as memory consumption. In addition, it covers the model's structural complexity, including the number of parameters, functions and equations involved. Finally, computational efficiency reflects whether the method requires extensive iterations or optimization routines. [1, 13, 50]

To assess computational efficiency, a corresponding scale is introduced. It is shown in **Table 2**.

Scale	*Description*
Very High	Computationally lightweight and real-time capable; minimal parameters and resource usage; negligible latency
High	Efficient training and fast inference; suitable for embedded or real-time systems; minimal reliance on optimization routines
Moderate	Moderate training and inference time; real-time application possible; moderate memory demands; manageable model complexity
Low	Slow execution; moderate to high memory consumption; model includes multiple parameters/functions; may require optimization
Very low	Extensive training and inference time; very high memory usage; complex architecture with many parameters/functions; impractical

Table 2: Scoring of the computational efficiency criterion

3.1.2.3 Interpretability

Interpretability refers to the degree of expert knowledge that is needed to understand the model [1]. This includes the transparency of the decision logic, ranging from opaque "black-box" models to fully transparent "white-box" models [36]. While some models provide insights into physical degradation mechanisms, it is treated separately as an optional attribute and does not directly influence the interpretability scoring.

A classification scale focused on interpretability is constructed based on this definition to differentiate among the model-based SoH methods. It is outlined in **Table 3**.

Scale	*Description*
Very High	Fully transparent (white-box) model; every step in the decision process is explicitly understandable
High	Clear and traceable logic; input-output relationships are explainable; most internal operations are understandable with reasonable expert effort
Moderate	Partially transparent model structure; provides limited but meaningful insights into decision logic
Low	Mostly opaque model; internal workings are difficult to access or explain
Very low	Fully opaque (black-box) model; decision logic is inaccessible or incomprehensible

Table 3: Scoring of the interpretability criterion

3.1.2.4 Data requirements

The data requirements criterion assesses the quantity, diversity and preparation complexity of the data needed for the method to function effectively [50, 51]. This includes the type of measurements required, the necessity for high-resolution or long-term datasets, the availability of ground-truth labels for SoH and the complexity of preprocessing steps such as signal filtering or transformation [50]. It also addresses whether extensive manual feature engineering is needed [70].

A scale is defined based on this definition to evaluate and compare the data requirements of different approaches. It is presented in **Table 4**.

Scale	*Description*
Very Low	Learns effectively from scarce or noisy data; robust to input variability and missing values; minimal preprocessing and feature engineering required
Low	Performs well with relatively small or partial datasets; minimal preprocessing; low reliance on labeled data
Moderate	Requires moderate data volume; basic preprocessing; limited need for manual feature engineering
High	Needs large and diverse datasets; moderate preprocessing and labeling effort required
Very High	Demands extensive, high-resolution, labeled datasets; extensive preprocessing and manual feature engineering

Table 4: Scoring of the data requirements criterion

3.1.2.5 Reliability

Reliability assesses how well the method maintains performance under noisy or incomplete data and across different operational scenarios [13, 36]. It focuses on robustness to changes in conditions that deviate from the training distribution, e.g., atypical operating temperatures or degraded sensor inputs. [50]

A comparative scale is introduced based on this definition to capture the reliability of each model. It is outlined in **Table 5**.

Scale	*Description*
Very High	Consistently reliable and adaptive; manages diverse conditions with minimal performance loss; self-adjusts to new data or environments without retraining
High	Performs well across different conditions and operational profiles; resilient to noise and incomplete data
Moderate	Generally stable in common use cases; tolerates minor variability in input or environment
Low	Operates reliably only under narrowly defined conditions; limited resilience to data disturbances or changing conditions
Very low	Highly sensitive to environmental factors (e.g., temperature); lacks resilience to noise or incomplete data

Table 5: Scoring of the reliability criterion

3.1.2.6 Scalability

Scalability assesses the model's ability to maintain performance and feasibility as the scope of application increases, whether in terms of data size, model input complexity or system scale (from cell to pack level) [13, 73]. This criterion evaluates how training and inference time, memory demand and accuracy evolve with growing dataset size or system complexity [1]. Furthermore, it encompasses the model's resilience to high-dimensional or multivariate inputs, ensuring that increased data complexity does not compromise stability or generalization.

A scale for assessing scalability is designed based on this definition to evaluate how each method performs under varying conditions. It is presented in **Table 6**.

Scale	*Description*
Very High	Performance remains robust as data volume, dimensionality or system complexity grow, without requiring structural changes, tuning or significant resource scaling
High	Maintains strong performance with large datasets, multivariate inputs and complex system configurations; may require model tuning or moderate computational upgrades
Moderate	Handles modest increases in data volume or complexity; may require some tuning or resource scaling
Low	Struggles with increasing dataset size or system scale; performance deteriorates noticeably
Very low	Fails to maintain performance with growing datasets, system complexity or input dimensionality; retraining required

Table 6: Scoring of the scalability criterion

3.1.3 STRUCTURAL IMPLEMENTATION OF THE EVALUATION FRAMEWORK

The evaluation of model-based SoH estimation methods across the previously defined criteria follows a qualitative approach. Each method presented in **Chapter 2.4** (see **Figure 19** for a condensed overview) is assigned a rating from "Very Low" to "Very High" for each of the six criteria, based on the defined indicator thresholds (as presented in **Chapter 3.1.2**). These thresholds are informed by literature benchmarks, comparative studies and performance indicators, e.g., estimation error metrics, computational benchmarks or structural model characteristics. Where available, existing multi-method comparison studies are used to calibrate the scoring scale, ensuring internal consistency and alignment with established assessments.

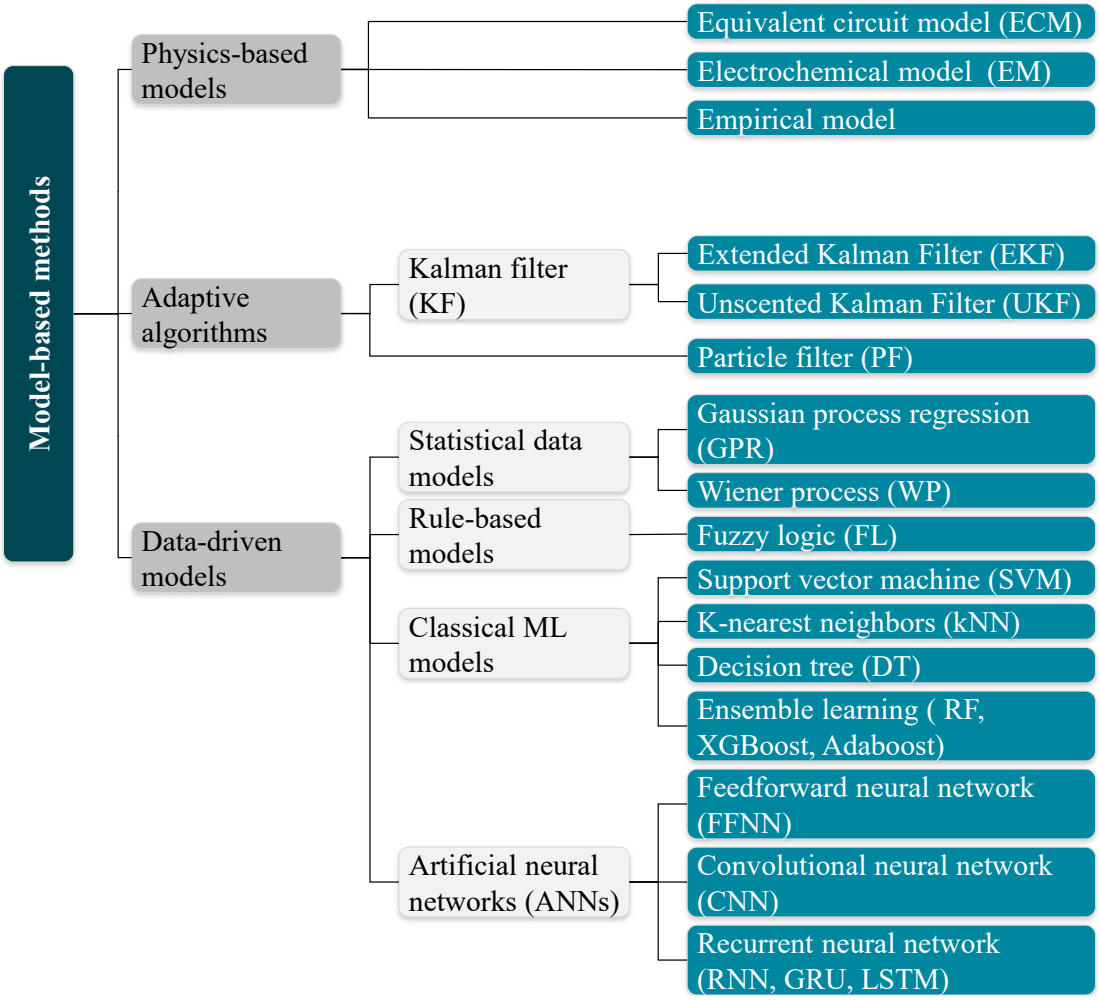


Figure 19: Overview of the model-based SoH estimation methods that are compared by the qualitative evaluation framework

To promote consistency and reproducibility, the scoring process is predominantly based on published comparisons in which two or more SoH estimation methods are evaluated under common experimental or simulation conditions. Furthermore, qualitative insights from studies that review multiple SoH estimation methods are systematically interpreted to support the relative positioning of methods along the scoring scale. In cases where methods are not directly compared, their reported advantages and limitations are evaluated in the context of the criterion definitions.

To enhance transparency and enable traceability of the scoring decisions, summary comparison tables are used to document the scoring rationale for each method–criterion pair. They can be found in **APPENDIX I** to **APPENDIX VI**.

The ratings across all six criteria are aggregated in a matrix structure (see **Figure 20**) to provide a cumulative performance profile for each method. On the one hand, these profiles enable a structured, evidence-based comparison that supports application-oriented selection of estimation methods. On the other hand, they guide the selection of candidates for further empirical analysis.

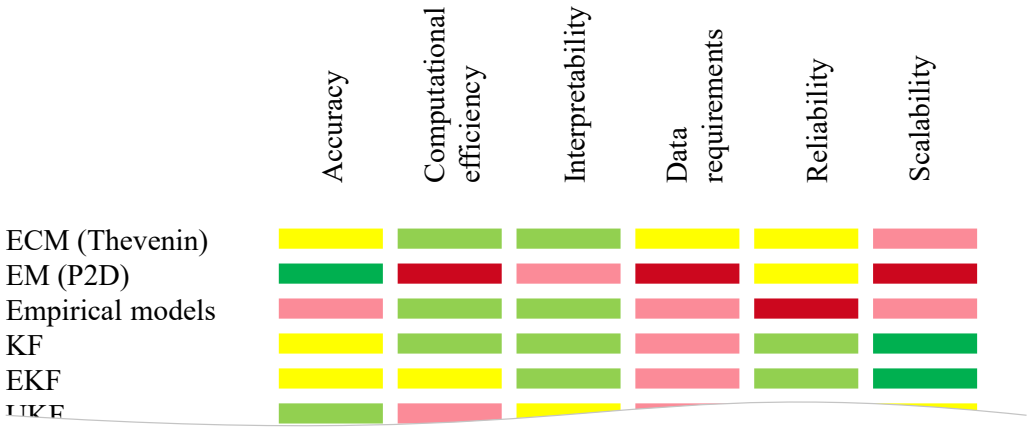


Figure 20: Matrix structure used to compare the model-based SoH estimation methods across the derived evaluation criteria

Based on the boundary conditions present in this thesis, a subset of four criteria is chosen. Across this subset, the three highest-scoring methods are carried forward to **Chapter 4** and **Chapter 5**, where they are implemented and evaluated on real-world battery datasets. This follow-up analysis serves as a quantitative validation of the framework's effectiveness in identifying high-performing methods. It is used to confirm or refine the qualitative assessments, to close the loop between conceptual evaluation and applied validation.

3.2 EVALUATION FRAMEWORK APPLICATION

Following the definition and construction of the evaluation framework in the previous chapter, this section applies it systematically to the model-based SoH estimation methods introduced in **Chapter 2.4**. Each of the subsequent subchapters evaluates these methods across one of the six defined criteria.

For ECM and EM, a representative model type was selected for consistent and meaningful comparison, as there are significant differences between the existing topologies. The Thevenin model is used to represent ECMs, while the P2D model is selected for EMs. Both models are detailed in **Chapter 2.4** and were chosen due to being commonly employed in literature and industry. In contrast, the remaining methods are evaluated as described, without the need for further model specification.

3.2.1 ACCURACY

The accuracy criterion is assessed based on the method's ability to match predicted values with actual or reference SoH data on both training and test datasets. In addition, it covers the model's ability to deal with high-dimensional, nonlinear battery behavior as well as its tendency to overfit. The results are shown in **Table 7**.

Model-based SoH estimation method	*Accuracy*
ECM (Thevenin)	Moderate
EM (P2D)	Very High
Empirical model	Low
KF	Moderate
EKF	Moderate
UKF	High
PF	High
GPR	Very High

WP	Moderate
FL	Moderate
SVM/SVR	High
RVM	High
kNN	Low
DT	Low
RF	High
XGBoost	Very High
AdaBoost	High
FFNN	High
CNN	Very High
RNN	High
LSTM	High
GRU	High

Table 7: Qualitative comparison of model-based SoH estimation methods for the accuracy criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX I.**

Very High Accuracy

Methods rated "Very High" on the accuracy scale demonstrate robust performance on both training and test datasets as well as minimal tendency to overfit. In addition, they effectively capture nonlinear and high-dimensional battery dynamics. The methods belonging to this category are EM (P2D), GPR, XGBoost and CNN.

EMs achieve very high accuracy through detailed physical modeling that inherently captures degradation processes and nonlinearities [15, 32, 46, 74]. **GPRs**, on the other hand, excel in handling high-dimensional and nonlinear data [29–31, 75]. They also

provide probabilistic outputs through confidence intervals, offering insight into predictive uncertainty [29–31, 75]. Despite moderate susceptibility to overfitting, it yields one of the highest accuracies, outperforming several other model-based SoH estimation methods across multiple studies [1, 17, 66, 76]. **XGBoost** algorithms achieve similar accuracies as GPR by effectively managing nonlinearity through ensemble learning and regularization techniques like sub-sampling [24, 32, 47, 77]. Its performance on smaller datasets, however, can suffer from overfitting [32]. **CNNs** excel by autonomously learning hierarchical features from raw input data, increasing their effectiveness in high-dimensional input spaces [29]. In addition, they consistently outperform most ML and ANN models in battery SoH estimation [1, 22, 29, 70].

High Accuracy

Methods rated "High" demonstrate robust performance on training and test data with less accuracy than approaches in the previous category. Moreover, they are robust against overfitting, while also effectively managing nonlinear battery behavior. This category contains adaptive algorithms (UKF, PF), SVM, SVR and RVM, ensemble learning methods (RF, AdaBoost) and multiple ANNs (FFNN, RNN, LSTM and GRU).

UKFs achieve high accuracy without requiring linearization techniques, which enables strong performance even under high-nonlinearity [2]. **PFs** yield comparable results to UKFs [27]. In addition, they can manage nonlinear and non-Gaussian systems while also providing confidence intervals [23, 27, 50]. **SVMs** and **SVRs** combine high predictive power with structural risk minimization to reduce overfitting, though they may degrade on unseen data [13, 27, 29, 31, 50, 56]. Their prediction accuracy rivals that of ANNs and RFs due to their capability to handle nonlinearity and high-dimensionality [3, 19]. **RVMs** achieve similar accuracy to SVMs but offer the added advantage of probabilistic outputs and better control over underfitting and overfitting [14, 17, 75]. Among ensemble learning methods, **RFs** perform excellently on both training and test data [1]. Furthermore, they capture complex and nonlinear relationships with low overfitting risk [24, 27]. Similarly, **AdaBoost** achieves high

accuracy through model combination and careful tuning of weak learners, consistently outperforming weaker individual models (e.g., SVRs or DTs) [6, 14, 30, 68]. ANNs such as **FFNN**, **RNN**, **LSTM** and **GRU** also receive high accuracy ratings. Their predictive precision across multiple studies rivals that of CNNs [1, 3, 13, 16, 20, 22, 27, 32, 46, 70]. However, while all are capable of modeling nonlinear dynamics, they tend to overfit more easily compared to CNNs [3, 47, 50].

Moderate Accuracy

Methods rated "Moderate" exhibit reasonable accuracy on training data and partial ability to model nonlinear, high-dimensional phenomena. However, they suffer from limitations in generalization to unseen test data and overfitting. This category includes ECM, KF, EKF, WP and FL.

ECMs offer moderate accuracy due to their simplified representation of battery dynamics and struggle to capture complex nonlinearities [9, 24]. **KFs** rely heavily on the underlying battery model and do not inherently account for nonlinearities [2, 10, 13, 31, 43]. While **EKFs** improve KFs using linearization techniques that allow partial handling of nonlinearities, these approximations introduce errors and limit robustness under strong nonlinearity [2, 50, 51]. The **WP** offers good estimation accuracy due to its probabilistic structure, which allows it to capture gradual trends and abrupt deviations in battery behavior [17, 54, 59]. However, it is restricted to linear trends, limiting applicability in more complex systems [78]. Finally, **FL** performs moderately in both comparative and absolute terms, though capable of handling nonlinear relationships [2, 17, 22, 27, 43].

Low Accuracy

The "Low" accuracy rating is assigned to models that show limited predictive ability and high overfitting tendencies, especially when exposed to unseen test datasets or complex dynamics. The empirical model, kNN and DT are part of this category.

Empirical models offer moderate accuracy, but lack robustness across datasets and perform poorly outside of training conditions [15, 36, 73]. **kNNs** demonstrate both low accuracy and a strong tendency toward overfitting [3]. Lastly, **DTs** are capable of modeling nonlinear relationships but often overfit and underperform relative to other methods, leading to their placement in this category [1, 30, 56].

No assessed method falls into the "Very Low" category of the accuracy criterion, as none consistently demonstrates both very inaccurate predictions and a complete inability to model complex battery behavior.

3.2.2 COMPUTATIONAL EFFICIENCY

Computational efficiency is evaluated based on the computation burden during both model development and operation. This covers the model's structural complexity, training time, inference latency and memory consumption. Additionally, it reflects the model's suitability for real-time applications. The results are shown in **Table 8**.

Model-based SoH estimation method	*Computational efficiency*
ECM (Thevenin)	High
EM (P2D)	Very Low
Empirical model	High
KF	High
EKF	Moderate
UKF	Low
PF	Low
GPR	Very Low
WP	Moderate
FL	Low
SVM/SVR	Moderate

RVM	Low
kNN	Moderate
DT	Moderate
RF	Moderate
XGBoost	Moderate
AdaBoost	Moderate
FFNN	Moderate
CNN	Very Low
RNN	Very Low
LSTM	Very Low
GRU	Low

Table 8: Qualitative comparison of model-based SoH estimation methods for the computational efficiency criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX II**.

None of the evaluated methods have received a "Very High" computational efficiency rating. This can be attributed to the inherent structural complexity in all methods, which requires either prolonged training durations or slower inference times.

High Computational Efficiency

Methods rated as having "High" computational efficiency are characterized by efficient training and test times, low structural complexity and suitability for real-time deployment. This category includes the ECM, empirical model and KF.

First, the **ECM** demonstrates high computational efficiency due to its simplicity and minimal parameter set, which enables fast computation and real-time deployment [5, 9, 14, 23, 32, 47]. However, model training remains rather time-consuming due to the

required parameter initialization [15]. **Empirical models** are similarly efficient as they rely on simple equations and calculations, which makes them suitable for real-time applications [15, 30, 43, 46, 47]. Nonetheless, they require aging tests for parameter identification, which increases training time [15, 73]. **KFs** offer fast processing times, rapid model updates and simple calculations, making them highly efficient and real-time capable [2, 31]. However, their development phase is often prolonged by the required model pre-validation [2, 10].

Moderate Computational Efficiency

Methods within the "Moderate" computational efficiency category balance manageable training complexity and real-time feasibility, although with higher computational demands during development and, in some cases, during inference. This category includes the EKF, WP, multiple classical machine learning models (SVM/SVR, kNN, DT, RF, XGBoost, AdaBoost) and FFNN.

EKFs support real-time operation through recursive estimation, but their ability to deal with nonlinearities requires additional computations, which elevate computational complexity [13, 50, 51]. While inference is efficient, training remains extensive due to careful parameter tuning [2, 44]. **WP** models are conceptually simple and parameter-efficient post-training [14, 17, 57]. However, both parameter estimation and inference require mathematically intensive formulations, contributing to increased computational demand [17, 30, 57]. Similarly, **SVM/SVR** require extensive training due to complex mathematical operations, cross-validation and parameter tuning [3, 17, 27, 44, 50, 56]. In addition, their memory footprint is typically high, scaling with the number of support vectors [13]. However, after the model is trained, it offers fast inference suitable for real-time applications [17, 31, 50]. In contrast, **kNN** avoids a training phase entirely but utilizes significant computational resources during inference due to the distance calculations between the input sample and every point in the dataset [3, 56]. While execution is fast in small datasets, the method suffers from high memory consumption and increasing inference latency in high-dimensional spaces [79]. **DTs** and **RFs** offer

fast inference capabilities but suffer from computationally intensive training and hyperparameter tuning [1, 30, 56]. In addition, RFs demand more memory than methods such as SVM and kNN but benefit from the ability to process raw input data without preprocessing [3, 32]. The other ensemble models, **XGBoost** and **AdaBoost**, achieve moderate efficiency by leveraging parallelism and weak learner ensembles [32]. Both exhibit fast inference but also long training times, especially in large-scale applications [56, 68, 80, 81]. Lastly, **FFNNs**, while structurally simple, require intensive training phases [1, 50, 70, 72]. Additionally, inference is generally fast, making them suitable for real-time applications [1, 20, 50, 70, 82].

Low Computational Efficiency

Methods categorized as having "Low" computational efficiency exhibit substantial memory usage and structural complexity. In addition, these methods pose slow execution and extended training times. This group includes the UKF, PF, FL, RVM and GRU.

UKFs introduce increased computational burden through sigma point propagation, which elevates their complexity relative to KFs and EKFs [44, 52]. However, using a high-performance controller, UKFs are capable of real-time estimation [2]. Similarly, **PFs** are computationally demanding, with their efficiency being highly dependent on the number of particles used [13, 27]. Repetitive sampling and resampling further increase processing time and memory demands [14, 50]. **FLs** involve complex rule bases and operations such as exponentials and divisions, resulting in high memory consumption and often requiring specialized hardware for real-time use [10, 17, 27, 83]. **RVMs** benefit from sparse solutions through Bayesian inference, which simplifies the model compared to SVMs and reduces inference cost [31]. However, they suffer from long, iterative training and memory-intensive operations, which limit their suitability for real-time applications [17, 29, 31, 67]. Finally, **GRUs** improve on LSTMs regarding computational efficiency by simplifying their architecture, making them faster and

more efficient [22, 26]. However, they remain demanding, especially in terms of training and hardware requirements [22, 26].

Very Low Computational Efficiency

The "Very Low" computational efficiency category includes methods characterized by high training complexity, large parameter sets, significant memory requirements and long inference times. This category includes the EM, GPR and multiple ANNs (CNN, RNN and LSTM).

EMs are computationally demanding due to multiple partial differential equations and extensive parameterization [6, 30, 77]. They require long training and update times and are not suitable for real-time applications [9, 23, 24, 46, 74]. **GPRs** exhibit similarly high computational burdens. Their inference latency is high due to matrix operations and exponential functions, which also lead to high memory usage [1, 16, 29–31]. Additionally, tuning hyperparameters and kernels significantly extends training duration [1, 17]. **CNNs** are among the most computationally demanding models, with extensive training time, massive datasets and high parameter count leading to long processing times [1, 14, 16, 70]. **RNNs** also require extensive computational resources due to their recursive structure [3, 38, 46, 70]. They exhibit long training times and elevated memory consumption exceeding that of models such as RF, SVM or kNN [3]. Their deployment on embedded systems is limited [38]. Lastly, **LSTMs** surpass RNNs and GRUs in both time and memory needed for training and inference, which restricts them in their applicability in real-time settings [1, 20, 22, 44, 75].

3.2.3 INTERPRETABILITY

The interpretability criterion is assessed via the transparency of the model's decision logic. This ranges from opaque "black-box" models to fully transparent "white-box" models. The results of the evaluation are shown in **Table 9**.

Model-based SoH estimation method	*Interpretability*
ECM (Thevenin)	High
EM (P2D)	Low
Empirical model	High
KF	High
EKF	High
UKF	Moderate
PF	Low
GPR	Low
WP	Moderate
FL	Very High
SVM/SVR	Low
RVM	Moderate
kNN	High
DT	Very High
RF	Low
XGBoost	Low
AdaBoost	Low
FFNN	Low
CNN	Very Low
RNN	Very Low
LSTM	Very Low
GRU	Very Low

Table 9: Qualitative comparison of model-based SoH estimation methods for the interpretability criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX III**.

Very High Interpretability

Methods rated "Very High" on the interpretability scale are fully transparent. They exhibit easily understandable decision processes to the external observer without requiring in-depth technical expertise. FL and DT belong to this category.

FL systems are based on explicit *if-then* rules containing linguistic variables and fuzzy sets, which make the internal reasoning process visible and intuitive [7]. This structure allows for easy traceability of the system's decisions from inputs to outputs without requiring extensive analysis of complex mathematical models. Similarly, **DTs** provide clear, hierarchical and rule-based structures which are easily traceable from root to leaf [30, 56, 67].

High Interpretability

Models with "High" interpretability demonstrate clear and traceable logic with explainable input-output relationships. While not as inherently transparent as the previous category, these methods allow for meaningful insight into their internal operations with reasonable effort. This category features the ECM, empirical model, KF, EKF and kNN.

Both the **ECM** and **empirical model** are conceptually simple [24, 43]. They are based on a limited set of parameters linked to degradation mechanisms and easy to interpret after expert parameter initialization [15, 23, 32, 73]. **KF** and **EKF** operate via transparent recursive updates based on known models and measurements, which enable full traceability of computations [44]. However, they require expertise to implement their equations, which are based on a detailed understanding of the battery parameters [44]. Lastly, the **kNN** algorithm achieves high interpretability by averaging the outcome

of the k most similar training instances, though this interpretability tends to decrease in high-dimensional spaces [79].

Moderate Interpretability

Methods rated as having "Moderate" interpretability are partially transparent. They provide limited but meaningful insight into the decision logic. A degree of expert knowledge is required for understanding. This category includes UKF, WP and RVM.

UKFs retain the Kalman structure but introduce less intuitive sigma points, reducing interpretability compared to KF and EKF [52]. In contrast, the **WP** is mathematically transparent with parameters linked to degradation trends [57, 58]. Nevertheless, its probabilistic output can be difficult to interpret for non-specialists. **RVMs** improve interpretability in comparison to SVM/SVRs by producing sparser solutions through fewer relevance vectors [29]. This simplifies the decision function and enhances the model's transparency compared to standard SVMs [29].

Low Interpretability

Methods with "Low" interpretability exhibit mostly opaque structures with limited insights into their decision logic. Their outputs are difficult to trace back to individual inputs without advanced expert knowledge. Methods included in this category are EM, PF, GPR, SVM/SVR, ensemble methods (RF, XGBoost, AdaBoost) and FFNN.

EMs simulate complex coupled partial differential equations that capture detailed physical insights, but diminish the traceability of the internal decision logic [2, 14, 24, 30]. Similarly, **PFs** estimate system states through a cloud of weighted particles, a process that lacks transparent rules and makes state estimation hard to interpret [23, 31, 32]. **GPRs** offer some interpretability as they allow for inspection of kernel functions and provide explicit uncertainty quantification [1]. However, it relies on complex covariance calculations and lacks simple analytical forms. **SVM/SVR** models generate decision boundaries from complex support vector combinations, which creates an opaque internal structure [23, 24, 50]. Ensemble methods (**RF**, **XGBoost** and

AdaBoost) combine multiple base learners to enhance predictive performance. Although each base learner is individually interpretable, their combination forms complex, less transparent models [24, 47, 68]. Finally, **FFNNs** are typically considered black-box models but benefit from a simpler structure than CNNs and RNNs, placing them at the lower end of this category [32, 50].

Very Low Interpretability

Methods rated "Very Low" function as opaque black-box models. In such models, the decision logic is inaccessible. This category features exclusively ANNs (CNN, RNN, LSTM and GRU).

CNN exhibits highly abstract internal operations and parameter-heavy architectures that limit its interpretability [32, 47, 50]. **RNN**, **LSTM** and **GRU** networks introduce complex, nonlinear hidden states that evolve over time [38, 70]. While these structures capture long-term dependencies effectively, they hinder traceability and understanding [50].

3.2.4 DATA REQUIREMENTS

The data requirements are evaluated in terms of the volume, heterogeneity and preparation complexity of the input data that are necessary for effective model performance. It also addresses the necessity for preprocessing or manual feature engineering before deployment. The results are shown in **Table 10**.

Model-based SoH estimation method	*Data requirements*
ECM (Thevenin)	Moderate
EM (P2D)	Very High
Empirical model	High
KF	High
EKF	High

UKF	High
PF	High
GPR	Moderate
WP	Low
FL	High
SVM/SVR	High
RVM	Moderate
kNN	High
DT	High
RF	Moderate
XGBoost	High
AdaBoost	Moderate
FFNN	Very High
CNN	Very High
RNN	Very High
LSTM	Very High
GRU	Very High

Table 10: Qualitative comparison of model-based SoH estimation methods for the data requirements criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX IV**.

No method was assigned a "Very Low" rating, as none are capable of learning effectively from scarce or noisy data while maintaining robustness in combination with minimal preprocessing and feature engineering.

Low Data Requirements

Methods rated as having "Low" data requirements perform reliably with small or partially incomplete datasets. They require minimal preprocessing and have limited reliance on labeled data. Solely WPs are classified under this category.

WP models typically require only a small amount of degradation data for offline parameter computation compared to other data-driven methods. Their structure does not depend on high-dimensional inputs or extensive labeling and their stochastic formulation allows for effective modelling under data sparsity [17, 30].

Moderate Data Requirements

Models within "Moderate" data requirements need a reasonable amount of diverse data, basic preprocessing and limited manual feature engineering. The category includes ECM, GPR, RVM, RF and AdaBoost.

ECMs have moderate data requirements, mainly due to identification and recalibration of the model's parameters [9, 36]. **GPRs** operate effectively with small datasets, but they are sensitive to input feature quality and kernel selection, which requires careful tuning [24, 56, 75]. **RVMs** reduce the reliance on kernel and penalty factor selection found in SVMs, resulting in less manual configuration [13, 14]. However, to effectively initialize the model, diverse and labeled input data are beneficial [75]. **RFs** require larger datasets than simpler methods such as SVM or kNN, yet they compensate for that by offering automatic feature selection, which reduces preprocessing needs [3, 14, 32]. Similarly, **AdaBoost** exhibits moderate data requirements, even though high-quality input data is beneficial. In addition, they are less reliant on complex hyperparameter tuning and require only modest labeling [14, 24, 80].

High Data Requirements

Methods with "High" data requirements rely on large, diverse and high-quality datasets. They often need moderate preprocessing, labeled data and significant effort in feature

selection. This group includes empirical models, adaptive algorithms (KF, EKF, UKF and PF) and multiple data-driven methods (FL, SVM/SVR, kNN, DT and XGBoost).

Empirical models require extensive experimental testing to derive parameters and ensure accurate model calibration [5, 15]. In contrast, Kalman filters (**KF**, **EKF** and **UKF**) do not require training data per se but depend heavily on the development of precise underlying battery models which require experimental testing and datasets [43, 44, 50, 74]. **PFs** are particularly data-intensive requiring large volumes of high-quality experimental measurements to accurately estimate system states and capture system dynamics under varying conditions [23, 74]. Furthermore, all data-driven methods of this category (**FL**, **SVM/SVR**, **kNN**, **DT** and **XGBoost**) rely on high-quality and diverse input data for optimal performance [2, 7, 10, 24, 32, 56, 74, 83, 84]. FL further depends on expert-defined rule sets, increasing the data burden [85]. In contrast, SVM/SVR and XGBoost involve extensive preprocessing, including kernel tuning and manual feature engineering [16, 30, 31, 50, 77, 80].

Very High Data Requirements

Methods rated as having "Very High" data requirements rely on extensive, high-resolution and labeled datasets. Furthermore, they need significant preprocessing and manual feature engineering. EMs and multiple ANNs (FFNN, CNN, RNN, LSTM and GRU) belong to this category.

EMs demand extensive experimental data and expert knowledge for accurate parameter identification and model calibration [15, 23, 50, 74]. In contrast, ANNs require large volumes of diverse and high-quality datasets for effective training [3, 16, 22, 27, 38, 50, 74]. Their performance depends on the completeness and stability of the input data, including relevant features [30, 38]. Moreover, **FFNNs** as well as **RNNs** and their extensions (**LSTM** and **GRU**) depend on careful hyperparameter tuning [3, 47]. **CNNs**, on the other hand, are capable of automatic feature extraction [3, 16]. While this eliminates the manual feature engineering process, for effective automated feature identification, it further increases the need for diverse and high-quality data [3, 16].

Finally, temporal architectures such as **RNN**, **LSTM** and **GRU** are especially demanding in terms of datasets. They require high-resolution, long-term sequences with accurate SoH labels to capture the dynamics of battery degradation over time [3, 22, 50].

3.2.5 RELIABILITY

The reliability criterion is assessed by how well the model performs under changing conditions, noisy or incomplete data and across different operational scenarios. The comparative results are summarized in **Table 11**.

Model-based SoH estimation method	*Reliability*
ECM (Thevenin)	Moderate
EM (P2D)	Moderate
Empirical model	Very Low
KF	High
EKF	High
UKF	Very High
PF	High
GPR	Very High
WP	High
FL	High
SVM/SVR	High
RVM	High
kNN	Low
DT	Moderate
RF	Very High
XGBoost	High

AdaBoost	Moderate
FFNN	High
CNN	Very High
RNN	Very High
LSTM	High
GRU	High

Table 11: Qualitative comparison of model-based SoH estimation methods for the reliability criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX V**.

Very High Reliability

Models in this category consistently exhibit high performance across diverse and dynamic operating conditions, demonstrating robustness to noise, incomplete data and environmental variability. This category includes the UKF, GPR, RF, CNN and RNN.

The **UKF** offers improved robustness compared to both KF and EKF by incorporating nonlinear transformations through sigma points, enhancing accuracy under uncertain conditions [22, 27, 50]. While it remains sensitive to variations in current profiles or ambient temperature, it demonstrates strong adaptability when properly configured [54]. **GPR** provides a probabilistic framework with inherent uncertainty quantification, which enables handling of nonlinearities and sparse or noisy data [45, 56]. In addition, its flexibility in modeling complex, nonlinear relationships enhances reliability across different operating scenarios [56]. **RF** models are robust to outliers and noise, which allows raw sensor data to be directly fed into the trained model [3, 11, 30, 47]. This enhances the model's resilience when handling heterogeneous data or incomplete datasets [68]. In addition, they are adaptable across diverse battery usage scenarios, supporting their classification in this group [68]. Finally, **CNNs** and **RNNs** are both adaptable to different operating conditions and exhibit superior generalization ability to

other ANNs [22, 38, 50]. For **CNNs**, these abilities are further supported through automatic feature extraction [50].

High Reliability

Methods rated "High" on the reliability scale consistently perform well across a range of operational conditions and are resilient to moderate levels of noise and incomplete data. However, their robustness may depend on the accuracy of an underlying model or be affected by specific operating conditions. This category includes adaptive filters (KF, EKF and PF), WP, FL, classical machine learning approaches (SVM/SVR, RVM and XGBoost) and selected ANNs (FFNN, LSTM and GRU).

The **KF** and **EKF** are recursive filters that perform well on noisy and sparse data [13, 51, 86]. However, their predictive performance depends on the underlying battery model [50]. **PFs** and **WPs** both utilize probabilistic frameworks to effectively manage uncertainties in system behavior, which enables them to remain robust under operational variability and incomplete data [13, 23, 31, 50, 60]. While the WP demonstrates robustness towards measurement noise, PFs are rather sensitive to such conditions [23, 32, 60]. Similarly, **FL** systems manage uncertainty and imprecision well, outperforming methods such as SVM and FFNN in fluctuating conditions [27, 83]. **SVM/SVRs** offer strong generalization across a wide range of operating conditions and an inherent robustness to noise [2, 13, 23, 27, 50]. They are effective with small datasets and maintain performance under complex and varying conditions [14, 56]. However, their sensitivity to kernel and hyperparameter selections can affect both their adaptability and tolerance to noise and data sparsity [3, 44, 56]. **RVMs** enhance noise resistance and adaptability beyond SVMs through probabilistic modeling [2, 13, 14, 75]. Yet, this reliability is limited when the training data are overly sparse [14, 43]. Finally, ensemble and deep learning methods such as **XGBoost**, **FFNN** and **LSTM/GRU** networks maintain stable performance under noisy or incomplete inputs, though some susceptibility to temperature effects and data outliers exists [14, 20, 22, 27, 32, 47, 50, 56, 80].

Moderate Reliability

Methods with "Moderate" reliability typically perform well under nominal conditions but have limited resilience to noisy or highly variable inputs. The ECM, EM, DT and AdaBoost are part of this category.

ECMs are effective under steady-state conditions and struggle with extreme temperature or high-current scenarios [9, 24, 32]. Its noise resilience is often enhanced through integration with filtering techniques such as KFs [46]. **EMs** require comprehensive datasets covering multiple operating conditions to maintain robustness and performance for changing conditions [9, 32]. **DTs** and **AdaBoost** algorithms provide stable performance under standard operating conditions and can adapt to different operating profiles [6, 47, 56]. However, both methods are sensitive to noise and outliers [14, 56, 80]. Moreover, DTs heavily depend on the initial parameter selection, which limits their robustness [56].

Low Reliability

Methods with "Low" reliability are only reliable under specific, controlled conditions. Their performance degrades significantly in the presence of noise, outliers or missing data. The kNN algorithm is representative of this category.

kNN generally performs well under controlled conditions. To mitigate sensitivity to noise and outliers as well as increase its reliability in unseen operational conditions, it requires extensive data preprocessing and careful feature selection. [3, 56]

Very Low Reliability

Methods rated as having "Very Low" reliability are sensitive to multiple environmental factors and lack resilience to both noise and incomplete data. Empirical models are the sole models belonging to this category.

Empirical models are calibrated on specific experimental datasets and tend to generalize poorly beyond the conditions represented in the training data [6].

Furthermore, they lack robustness under changing conditions, further decreasing their overall performance [15, 36, 43, 46, 47, 69, 73].

3.2.6 SCALABILITY

Scalability is evaluated based on each model's ability to maintain predictive performance and computational feasibility as the complexity or scale of the application increases. This includes assessing how the model responds to larger datasets, more complex inputs and broader system-level deployment (from cell to pack). Key indicators include training time, inference speed, memory demand and accuracy. The results of the evaluation are presented in **Table 12**.

Model-based SoH estimation method	*Scalability*
ECM (Thevenin)	Low
EM (P2D)	Very low
Empirical model	Low
KF	Very High
EKF	Very High
UKF	Moderate
PF	Low
GPR	Moderate
WP	Moderate
FL	Low
SVM/SVR	Moderate
RVM	Low
kNN	Low
DT	Moderate
RF	High
XGBoost	Very High

AdaBoost	Moderate
FFNN	High
CNN	Very High
RNN	High
LSTM	High
GRU	High

Table 12: Qualitative comparison of model-based SoH estimation methods for the scalability criterion

The rationale for each rating is presented in the following paragraphs. For clarity and coherence, methods are grouped and discussed according to their assigned rating category. Additional explanation details can be found in **APPENDIX VI**.

Very High Scalability

Methods rated "Very High" on the scalability scale maintain robust performance across increasing data volumes, high input dimensionality or large-scale systems without requiring fundamental redesigns, extensive tuning or significantly increased computational resources. KF, EKF, XGBoost and CNN are included in this category.

KF and **EKF** achieve excellent scalability due to their low computational complexity, which allows for efficient operation in larger battery systems [13]. In addition, both filters can accommodate multivariate inputs due to their use of state vectors and covariance matrices [44]. **XGBoost** shows very high scalability, being one of its key factors for its widespread success [80]. Combined with support for parallel and distributed training, it enables the model to learn fast on large and complex datasets while maintaining predictive accuracy [87]. **CNNs** also manage large scale datasets effectively [1, 3, 16]. Additionally, they can handle high-dimensional input via automatic feature extraction and scalable architecture. However, their larger parameter counts in deeper networks may increase computational demands with scale [26].

High Scalability

Methods with "High" scalability maintain strong performance when scaling to large datasets or complex system configurations, although they may require moderate tuning or computational upgrades. This category includes RFs and several ANNs (FFNN, RNN, LSTM and GRUs)

RFs are well-suited for large-scale applications, managing large and high-dimensional datasets flexibly with fewer tuning parameters than SVR and several ANNs [1, 3, 69]. In contrast, the **FFNNs'** scalability stems from effectively accommodating multivariate inputs and complex system configurations [50]. Compared to CNNs, their flexibility in model updating and adaptation to big data environments is slightly lower [1]. Nevertheless, they maintain stable performance on both large datasets and multivariate inputs [50]. Lastly, **RNNs**, along with their variants **LTSM** and **GRU**, capture temporal dependencies, making them suitable for applications involving sequential inputs [26]. These architectures scale well to larger datasets and complex system configurations [13, 26]. However, the increase in model parameters with growing data volumes imposes greater computational and memory demands [13, 26]. LSTMs improve scalability over basic RNNs by mitigating vanishing gradient issues, while GRUs offer a more computationally efficient alternative with comparable performance [13, 16, 26].

Moderate Scalability

Methods exhibiting "Moderate" scalability can accommodate some increase in data volume or input complexity but face computational or tuning challenges as system scale expands. They often require additional resources or adjustments to maintain performance. UKF, statistical data models (GPR, WP), FL and multiple classical machine learning approaches (SVM/SVR, DT and AdaBoost) fall in this category.

UKFs scales moderately. On the one hand, they can handle multivariate inputs effectively and operate in parallel architectures [44]. On the other hand, its computational structure involves nonlinear transformations using sigma points, which

increases memory demand and inference time compared to KF and EKF and limits overall efficiency in larger systems [44, 50]. **GPR** is highly capable of managing large datasets but its computational cost increases cubically with dataset size, making scaling expensive despite approximation techniques and sparse representations [1, 29, 56]. The **WP** has a mathematically lightweight structure. However, larger battery systems and increased input dimensionality increase system complexity and require computationally demanding parameter identification and model tuning [88]. Furthermore, **SVM/SVR** exhibit strong generalization abilities, enabling effective performance on unseen data without extensive retraining [13]. However, their computational costs increase with large or high-dimensional datasets [22, 50, 75]. While **DTs** can handle moderately large datasets and multivariate inputs, their performance and efficiency degrade as the data volume or input dimensionality increases, which leads to longer training times and increased memory usage [56]. Finally, **AdaBoost's** sequential training process slows scaling compared to parallelized methods like XGBoost, limiting scalability [68].

Low Scalability

Methods with "Low" scalability struggle to maintain performance and computational efficiency as dataset size or system complexity increases, often requiring substantial resource increases or retraining. The ECM, empirical model, PF, RVM and kNN are all part of this category.

The **ECM** is typically developed for individual cells and must be redesigned for modules or packs [36]. This redesign process increases complexity and complicates parameter identification, thus limiting scalability to large scale systems [36]. Similar to ECMs, **empirical models** require a dedicated model per cell, which increases complexity and computational demands when scaling to modules or packs [36]. While some studies have proposed aggregate empirical models for modules, these approaches have shown limited accuracy and have not yet been extended to full battery packs [36]. **PFs** suffer from scalability issues due to the exponential growth in the number of particles needed as system dimensionality increases, leading to substantial

computational demands [50]. **FL** also struggles with complex systems and high input dimensionality, requiring an exponential increase in rules and parameters, which leads to substantial design and tuning efforts [89]. **RVMs**, although sparser than SVMs, have longer training times and higher computational demands when working with large datasets or complex input structures [13, 29]. Lastly, **kNN** requires storing and comparing all training samples during inference [38]. As the dataset or input dimensionality grows, both memory and computational cost grow significantly, limiting scalability [3, 56].

Very Low Scalability

Methods rated as having "Very Low" scalability fail to maintain performance or feasibility when scaling, often becoming impractical due to large computational demands or model complexity. The EM is the only method belonging to this category.

Even for single cells, **EMs** simulate detailed electrochemical processes with significant computational complexity [13]. Since each cell ages differently, applying EMs to battery packs increases the complexity and computational effort significantly, making it impractical for large systems without major simplifications. [13]

3.2.7 SELECTION OF SOH ESTIMATION METHODS FOR QUANTITATIVE ANALYSIS

After finalizing the evaluation across the six criteria of the developed framework, the consolidated results are presented in **Figure 21**. To enhance interpretability, the individual ratings are color-coded according to the legend provided at the bottom of the figure.

Figure 21: Summary of the evaluation of each model-based SoH estimation method across six criteria

As illustrated in **Figure 21**, data-driven methods, specifically ANNs and ensemble learning methods, pose the best overall performance across the six criteria. Their primary strength lies in their combination of high accuracy, strong reliability and scalability. However, these advantages come at the cost of significant data requirements and reduced computational efficiency. Furthermore, their inherent black-box

characteristics limit interpretability, which may hinder adoption in applications requiring transparency.

Physics-based methods show highly variable performance depending on the complexity of the selected model. Advanced EMs and detailed ECMs offer high predictive accuracy but are constrained by heavy computational demands and substantial data requirements. Simpler alternatives, such as empirical models and basic ECMs, provide more efficient and accessible options, but their limited robustness and poor scalability undermine their overall suitability.

Adaptive filters demonstrate a favorable trade-off between accuracy and reliability, with simpler implementations offering excellent interpretability and scalability. Nonetheless, their reliance on pre-established battery models (ECMs, EMs, or empirical) limits their applicability, as extensive model development is generally required beforehand.

Within the spectrum of data-driven approaches, each subgroup presents distinct advantages. Statistical models like GPR provide probabilistic outputs that enhance both accuracy and robustness. However, their computational burden is high, limiting real-time applicability. Rule-based methods such as FL exhibit moderate accuracy but are highly interpretable and perform reliably in noisy environments. Classical ML methods require less data than ANNs and offer simpler models, but they typically fall short in scalability and accuracy. However, ensemble learning techniques, RF and XGBoost, distinguish themselves by maintaining said accuracy and reliability with lower data and computational demands compared to deep learning approaches.

To select a subset of methods for implementation and testing in **Chapter 4** and **Chapter 5**, four key criteria are prioritized equally based on the boundary conditions given in this thesis: accuracy, computational efficiency, reliability and scalability. Interpretability is de-emphasized, as sufficient understanding of the model mechanisms

has been established. Importantly, the required dataset for training and validation is available, thus eliminating a common barrier to deploying data-driven approaches.

Under these considerations, **RF**, **XGBoost** and **CNNs** are selected for further implementation. They offer the most balanced and robust performance across the prioritized criteria. CNNs demonstrate superior capabilities in handling complex, large-scale data with high accuracy and reliability. RF and XGBoost, while showing slightly lower performance in these areas, offer a more favorable tradeoff between predictive power and computational efficiency.

Although alternative methods such as GPR, SVM/SVR and adaptive filters also show considerable potential, their inclusion would extend the scope of the thesis beyond practical limits. Moreover, the dependence of adaptive filters on external battery models further complicates their deployment and limits their generalizability.

3.3 INTERIM SUMMARY

This chapter developed a comprehensive evaluation framework for the systematic assessment of model-based SoH estimation methods. The framework was derived through an extensive literature review, during which five suitable studies were identified. They were selected based on their use of multi-criteria evaluation frameworks applied to a range of model-based SoH estimation approaches.

From a comparative analysis of these studies, six evaluation criteria were derived: accuracy, computational efficiency, interpretability, data requirements, reliability and scalability. Each criterion was formally defined and a corresponding five-level scoring scale was introduced, ranging from “Very Low” to “Very High”, to facilitate a consistent and transparent assessment process.

In the second part of the chapter, the SoH estimation methods introduced in **Chapter 2.4** were qualitatively evaluated against these six criteria. Ratings were

assigned based on empirical comparisons from literature, methodological strengths and weaknesses and expert insights from comparative reviews. The results were consolidated into a color-coded evaluation matrix to enhance interpretability and support high-level comparison.

The outcome of this evaluation highlights ANNs and ensemble learning methods as the most promising candidates under the given boundary conditions, because they offer high levels of accuracy, reliability and scalability. However, these strengths are accompanied by increased computational demands, significant data requirements and limited interpretability due to their black-box nature. In the context of this thesis, interpretability and data availability are not limiting factors, as public datasets are accessible and sufficient methodological understanding has been developed to construct and deploy such models.

Out of the ANNs and ensemble learning methods, RF, XGBoost and CNNs have been selected for further empirical validation in **Chapter 4** and **Chapter 5**. These methods offer the most promising combination of accuracy, reliability and scalability and are subject to quantitative performance analysis in the following chapters to verify the qualitative findings established in this chapter.

Chapter 4. IMPLEMENTATION OF SOH METHODS

To apply and evaluate the in **Chapter 3** selected model-based SoH estimation methods, this chapter presents the complete implementation process. It is structured into two main parts.

In the first part, the selected dataset is introduced. **Chapter 4.1.1** provides a brief description of the dataset's technical specifications and describes the collected raw data. As the dataset originates from real-world operations, several preprocessing and preparation steps are required before applying the SoH estimation methods. **Chapter 4.1.2** outlines these preprocessing techniques, including data cleaning, interpolation, sliding and normalization. It then details the SoH labeling process required to generate the ground truth data for subsequent model training and evaluation. This includes data sorting, concatenation and the calculation of capacity and SoH values.

The second part of the chapter details the implementation of the three selected estimation methods: RF, XGBoost and CNN. **Chapters 4.2** to **4.4** each address one method, beginning with a theoretical background including mathematical formulations, followed by its implementation in Python.

This section concludes with **Chapter 4.5**, providing an interim summary of the key implementation steps and forming the transition to **Chapter 5**, in which the performance of each method is evaluated based on the simulation results.

4.1 DATASET

This study uses real-world datasets for SoH estimation, as they accurately reflect operational conditions and battery aging behaviors encountered in practical battery use. Unlike laboratory datasets, real world data captures a broad spectrum of influencing

factors such as ambient temperature, current rates, SoC variations and user-specific driving or charging behavior [8, 24]. These variabilities are essential for training robust and generalizable data-driven models. Still, laboratory datasets remain prevalent in the literature as they are more readily available. Prominent examples include datasets from NASA, CALCE (Center for Advanced Life Cycle Engineering) and several universities [90].

Among the limited number of real-world datasets identified, a dataset compiled by Tsinghua University stands out for its scale and diversity. It contains a total of three datasets covering 464 vehicles across three different BEV types. In total, it includes over 1.2 million charging events ("snippets") collected over timespans ranging from one to 2.5 years per vehicle [91]. This extensive and heterogeneous dataset forms the basis for implementing and testing the SoH estimation methods selected in **Chapter 3**.

4.1.1 DESCRIPTION

As described in the introductory paragraph of this chapter, the dataset contains three distinct subsets that collectively capture operating records of 464 vehicles across three different BEV types. The raw data consists of roughly 1.2 million charging snippets, each representing a discrete charging event with two main components: time- series measurements and associated meta information. The time-series data includes eight key variables: voltage (V), current (I), SoC, maximum and minimum cell voltage (V_{max}, V_{min}), highest and lowest cell temperature (T_{max}, T_{min}) and a timestamp (t). These variables provide a comprehensive view of the battery's electrical and thermal state during charging. Complementing the time-series data, the meta information includes identifiers such as vehicle number and mileage. An overview of the technical specifications and statistics of each dataset is shown in **Table 13**. [8]

Parameters	*Dataset* 1	*Dataset* 2	*Dataset* 3
BEV type	A-commercial	B-commercial	C-private
Fleet size	217	198	49
Battery material	NMC-C	NMC-C	NMC-C
Rated battery capacity	145 Ah	155 Ah	153 Ah
Sampling interval	30s	10s	10s
Collecting period	2.2 years	1.0 year	2.5 years
Charging snippets	629,121	472,829	176,327

Table 13: Statistics of BEV battery systems and datasets [8]

Datasets 1 and 2 represent commercial vehicle fleets, whereas Dataset 3 contains private vehicles. All vehicles are equipped with batteries of the same NMC-C (Nickel Manganese Cobalt Carbon) chemistry, with rated capacities ranging from 145 Ah to 155 Ah. The datasets differ notably in both fleet size and data collection period. Dataset 1 encompasses 217 vehicles, Dataset 2 includes 198 vehicles and Dataset 3 contains data from 49 vehicles. Regarding their collection period, Datasets 1 and 3 span over 2 years, while Dataset 2 covers only a one-year period. These differences, along with differing sampling intervals, result in an uneven distribution of charging snippets across the three datasets, totaling ~630,000, ~470,000 and ~175,000, respectively.

While discharge behavior is often irregular and random due to inconsistent driving patterns, charging events determined by the vehicle BMS and input conditions exhibit more consistency. In addition, ignoring the discharging data does not eliminate information needed to calculate the battery's SoH, as it is solely based on the remaining battery capacity. Consequently, only charging segments were retained for further analysis in preparation of this dataset. Both slow and fast charging events are covered as illustrated in **Figure 22** and **Figure 23**.

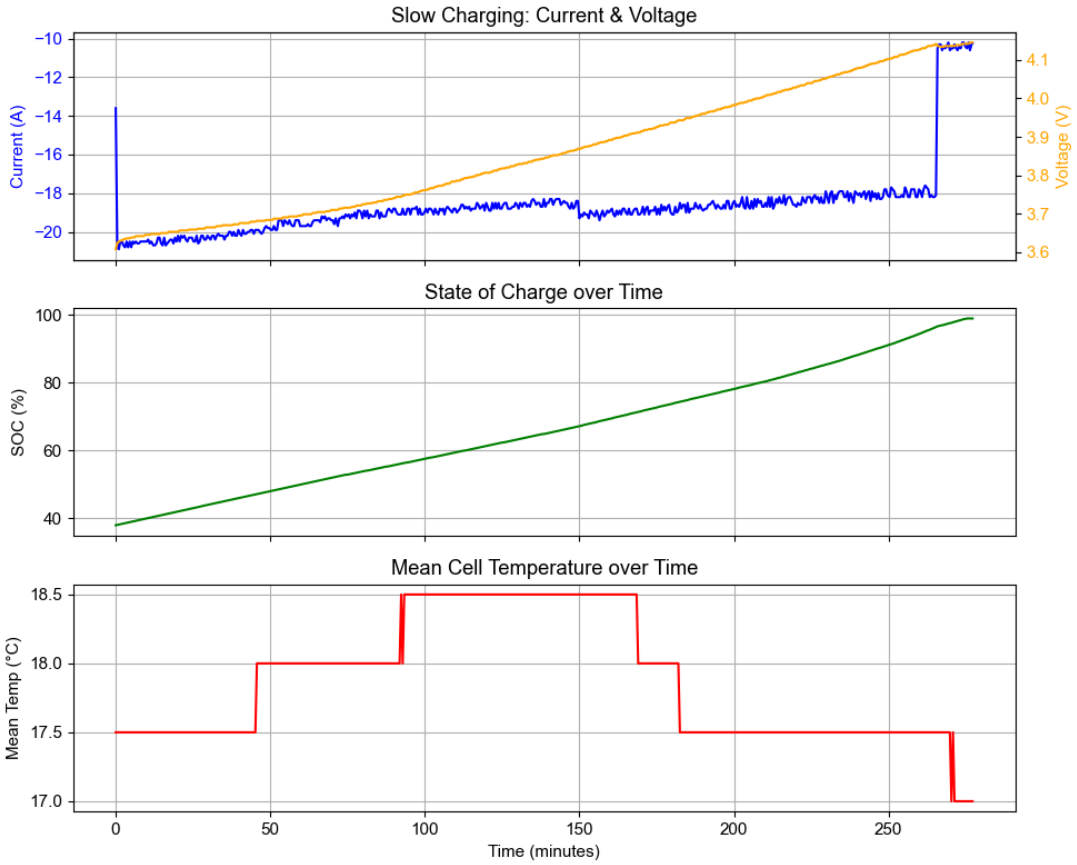


Figure 22: Visualization of an exemplary slow charging event within the dataset

Slow charging events are characterized by a consistent average current between 10 A and 20 A. In this dataset, a negative sign in front of the current value suggests that the battery is charging. For a full charge, the overall duration can exceed 9 hours. In the example depicted in **Figure 22**, the vehicle charges from 40% to 100% over 4.5 hours. During this time, cell temperature remains relatively stable, ranging between $17°C$ and $18°C$.

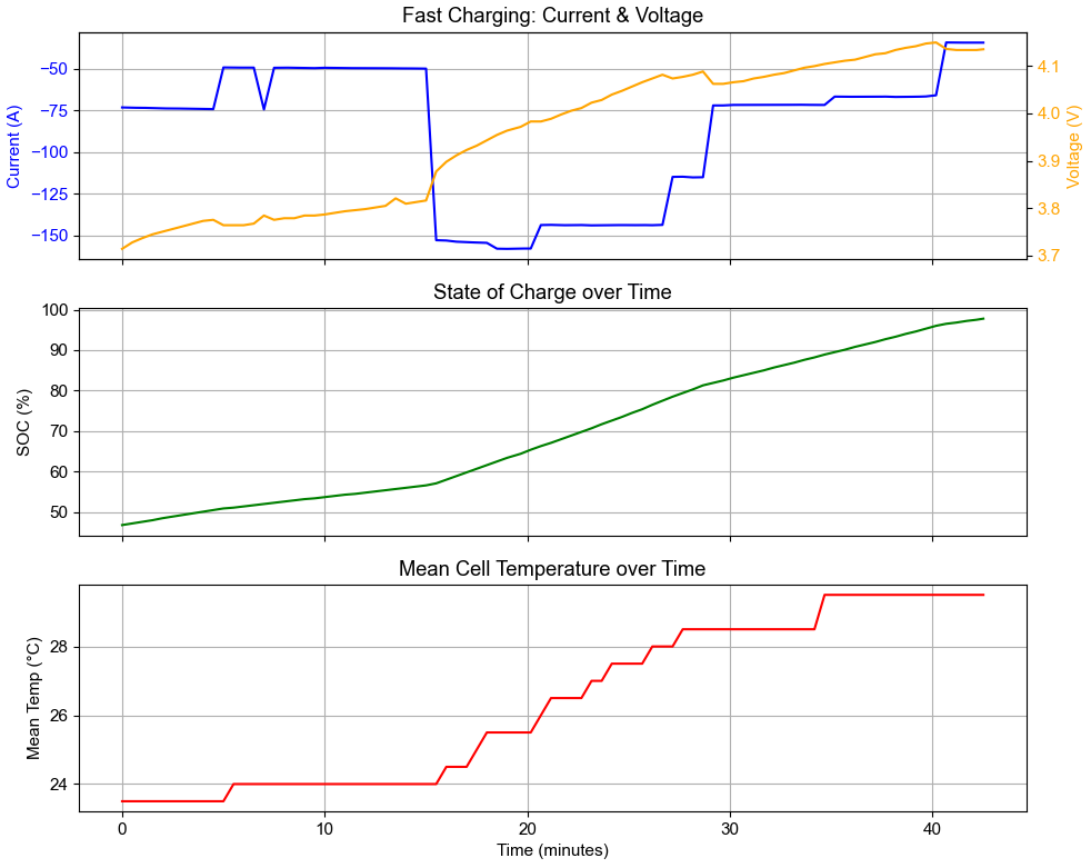


Figure 23: Visualization of an exemplary fast charging event within the dataset

Higher and more variable current levels distinguish fast charging events. A full charging cycle requires approximately one hour. In the illustrated example, a charge from 50% to 100% takes roughly 40 minutes. As expected, the mean cell temperature varies more during fast charging than during slow charging, reflecting the greater thermal load.

These visualizations are representative examples selected from different vehicles within the dataset. For a comprehensive description of the full dataset and its characteristics, the reader is referred to Lu et al., "*Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data*" [8].

4.1.2 PREPARATION

To address the incompleteness, variability and uncertainty inherent in real-world BEV charging data, the raw segments described in the previous chapter were preprocessed following the methodology outlined by Lu et al. in "*Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data*". This preprocessing is essential to ensure the quality and consistency of the input data for model training and evaluation. Once these segments are cleaned and normalized, each charging snippet is assigned a corresponding SoH label. This enables data-driven SoH estimation methods to infer patterns between observed input features and the battery's health status, which is the basis for the subsequent training and evaluation.

4.1.2.1 Preprocessing

Lu et al. proposed a four step preprocessing procedure to address non-uniformity, noise, missing values and outliers, which are common challenges in large-scale, field-collected battery datasets. The main steps include data cleaning, interpolation, sliding and normalization [8].

In the initial stage, the charging segments are extracted from the raw data. Then, noise filtering techniques, typically based on smoothing filters, are used and outliers that deviate significantly from the statistical distribution of the data are identified and removed. To address the non-uniform time interval characteristic of Dataset 1, the second stage of preprocessing applies linear interpolation to the time series data. Originally collected with a timestamp interval of 30s, it is resampled to a uniform 10s to align with the temporal resolution of Datasets 2 and 3. In the third step, a sliding window approach extracts fixed length charging snippets consisting of 128 sampling points out of the collected charging segments. This results in a structured input matrix structure which is defined as follows [8]:

$$M_{snippet}=\begin{bmatrix} t_1 & I_1 & V_1 & V_1^{max} & V_1^{min} & T_1^{max} & T_1^{min} & SoC_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_i & I_i & V_i & V_i^{max} & V_i^{min} & T_i^{max} & T_i^{min} & SoC_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_{128} & I_{128} & V_{128} & V_{128}^{max} & V_{128}^{min} & T_{128}^{max} & T_{128}^{min} & SoC_{128} \end{bmatrix} \quad \textit{Eq. 4.1}$$

Finally, to ensure user privacy, the datasets underwent a desensitization and anonymization process. Key variables such as charging current, average cell voltage and cell temperature were perturbed and interpolated to mask exact values while maintaining structural characteristics. Additionally, metadata such as timestamps and vehicle mileage data were randomly shifted and scaled, eliminating reidentification of individual users or driving behaviors. Despite these transformations, Lu et al. assert that the usability of the dataset for meaningful analysis and modeling is preserved [8].

4.1.2.2 SoH labeling

Following the preprocessing steps described in the previous chapter, the next essential phase involves labeling the charging snippets with corresponding SoH values. This labeling is a critical enabler for the learning process of data-driven SoH estimation methods, as it provides the ground truth required for training and testing.

Lu et al. propose a four step methodology for SoH labeling in their study "*Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data*" [8]. However, the successful application of this method relies on the availability of accurate vehicle mileage data, which, as described in the previous chapter, has been randomly shifted and scaled for anonymization. As a result, the original labeling procedure cannot be used and has to be adjusted to accommodate this limitation while preserving analytical validity.

In a first step, all charging snippets are sorted by vehicle ID and then, within each vehicle ID, by mileage. Despite the perturbation applied to mileage data, empirical inspection reveals that charging snippets with identical mileage values are unaffected

by the relative shifts and scaling. Therefore, snippets that share the same mileage values can still be reliably grouped. The groups are then sorted by their respective SoC values to determine whether several charging snippets together constitute a complete charging segment. In most instances, this assumption holds true and multiple consecutive snippets at a given mileage collectively represent a full charging session. These grouped charging segments, ordered from lowest to highest SoC, represent the foundation for subsequent SoH calculation.

Once full charging segments are identified, the battery's SoH is computed using the capacity-based approach, which is the most widely used method in both academic and industrial contexts (see **Chapter 2.3.1**). The formula is defined as follows [9, 14, 43]:

$$SoH = \frac{Q_m}{Q_r} * 100\% \qquad \textit{Eq. 4.2}$$

where Q_m is the current capacity of the battery and Q_r is the nominal rated capacity, which, depending on the dataset, has the following values: 145 Ah (Dataset 1), 155 Ah (Dataset 2) or 153 Ah (Dataset 3).

The battery's current capacity is calculated as the total charge delivered during a charging segment, divided by the corresponding change in SoC [8]:

$$Q_m = \frac{\int_{t_{start}}^{t_{end}} I(t)\, dt}{SoC_{end} - SoC_{start}} \qquad \textit{Eq. 4.3}$$

The total charge is determined by integrating the current $I(t)$ over time. The start and end time of that integral determine the corresponding SoCs by whose difference the charge is divided to calculate the battery's current capacity.

To ensure the reliability and accuracy of the computed SoH values, several filtering criteria are applied throughout the calculation to exclude segments that are physically implausible or statistically problematic.

The first criterion is a voltage range constraint. Only the data points within a charging segment that depict voltages between 3.75V and 4.10V are retained, as this range corresponds to the near-linear region of the OCV curve of NMC-C batteries, which is a critical factor for accurate capacity estimation [8]:

$$V \in [3.75V,\ 4.10V] \qquad \textit{Eq. 4.4}$$

The second criterion tackles SoC and voltage stability within the charging segments. Those who exhibit sudden jumps in SOC or voltage are eliminated, as they are indicators of sensor noise, anomalies or data incompleteness and can lead to inaccuracies in SoH calculation:

$$SoC\ jumps \geq 2.5\%\ or\ V\ jumps \geq\ 100\,mV \qquad \textit{Eq. 4.5}$$

The third criterion addresses the SoC windows of the charging segments. With this filtering step, charging segments with insufficient SoC variation are removed, because they do not provide enough information to compute meaningful capacity:

$$\Delta SoC\ \geq 20\% \qquad \textit{Eq. 4.6}$$

The final criterion addresses the SoH range of calculated values. It retains only these segments, which show realistic SoH values:

$$SoH \in [0.7,\ 1.1] \qquad \textit{Eq. 4.7}$$

The lower bound of 70% reflects the typical end-of-life threshold for BEV batteries, while the upper bound of 110% accounts for early-life deviations above 100% due to initial conditioning and manufacturing variability *[8]*.

Once the filtering steps are completed, the computed SoH values are assigned to each charging snippet that belongs to the associated charging segment. This results in a total of **888, 500 labeled charging snippets** that can be used for the training and testing of the selected SoH estimation methods.

After the dataset preparation is completed, the selected SoH estimation methods, determined in **Chapter 3**, are implemented. The underlying methodology, as well as their implementation, is depicted in the following three subchapters.

4.2 RANDOM FOREST

RF is a bagging-based ensemble learning method that constructs multiple randomized DTs and aggregates their predictions, resulting in high prediction accuracy and robustness [80]. The algorithm uses bootstrap sampling to create diverse training datasets by resampling the original data. Each of these bootstrapped datasets is used to train an individual DT using the classification and regression tree (CART) algorithm. Once all trees have been constructed, their outputs are combined, typically through averaging, to provide the final model prediction. [69]

4.2.1 METHODOLOGY

The RF algorithm begins by generating T bootstrap samples $\{S_1, S_2, \dots, S_T\}$ from the original training dataset S. Each sample S_t is then used to train a decision tree R_t using the CART algorithm. At each node split within a tree, rather than considering the full set of M features, a random subset of k features is selected, among which the optimal split is chosen [80]. This randomized feature selection introduces decorrelation among the trees, reducing variance without significantly increasing bias. [69]

After all T trees have been trained, the RF prediction for a new input x is obtained by averaging the predictions from each individual tree, as defined below [80]:

$$\hat{y} = h(x) = \frac{1}{T}\sum_{t=1}^{T} R_t(x) \qquad \textit{Eq. 4.8}$$

An important aspect of RF training is the use of out-of-bag (OOB) samples. Due to the bootstrap sampling process, each DT is trained on roughly two-thirds of the original dataset. The remaining unused samples form the OOB samples. They serve as an

internal validation set and enable an unbiased estimate of the model's generalization error. The OOB mean square error (MSE) is computed as follows:

$$MSE_{OOB} = \frac{1}{s}\sum_{i=1}^{s}(\hat{y}(x_i) - y_i)^2 \qquad \textit{Eq. 4.9}$$

where s is the number of OOB samples and $\hat{y}(x_i)$ and $y_i(x_i)$ are the predicted and true values of x_i. [69]

Two key hyperparameters that significantly influence the RF's performance are the maximum tree depth and the number of features k considered at each split. The tree depth controls the model complexity and overfitting, while the number of features k balances diversity and accuracy of the individual trees. [69]

4.2.2 IMPLEMENTATION DETAILS

To implement the methodology described in the previous chapter in Python, the "*RandomForestRegressor*" class from the "*scikit-learn*" library is used. The parameter configuration is based on a combination of literature recommendations and best practices to balance predictive accuracy and computational efficiency. **Table 14** summarizes the hyperparameter choices used for this implementation.

Parameter	*Value*	*Description*
n_estimators	200	Number of decision trees in the ensemble
max_depth	4	Maximum tree depth to control model complexity
max_features	n_features	Number of features considered at the split of each node
min_samples_split	3	Minimum number of samples (data points) required to split an internal node
random_state	0	Ensures reproducibility by fixing the random seed
n_jobs	-1	Enables parallelization, utilizing all available CPU cores for training and inference

Table 14: RF selected hyperparameters

The choice of "$n\_estimators = 200$" and "$min\_samples\_split = 3$" is informed by findings from Nguyen et al., which found that increasing the number of trees beyond 200 yields negligible accuracy improvements while increasing computational cost [92]. To not limit the model's accuracy, "*max_features*" is set to the default value for regression problems, which considers all existing features ("*n_features"*). Limiting the tree depth to 4 intends to balance model accuracy and overfitting control [69]. Finally, setting "$n\_jobs = 1$" ensures that both training and inference are fully parallelized, which improves computational efficiency.

4.3 XGBOOST

XGBoost is a boosting-based ensemble learning method built upon the gradient boosting decision tree (GBDT) framework [77, 80]. In contrast to RF, which builds DTs in parallel, XGBoost generates them sequentially, where each successive tree is trained to correct the residual error of its predecessor (see **Figure 24**) [68].

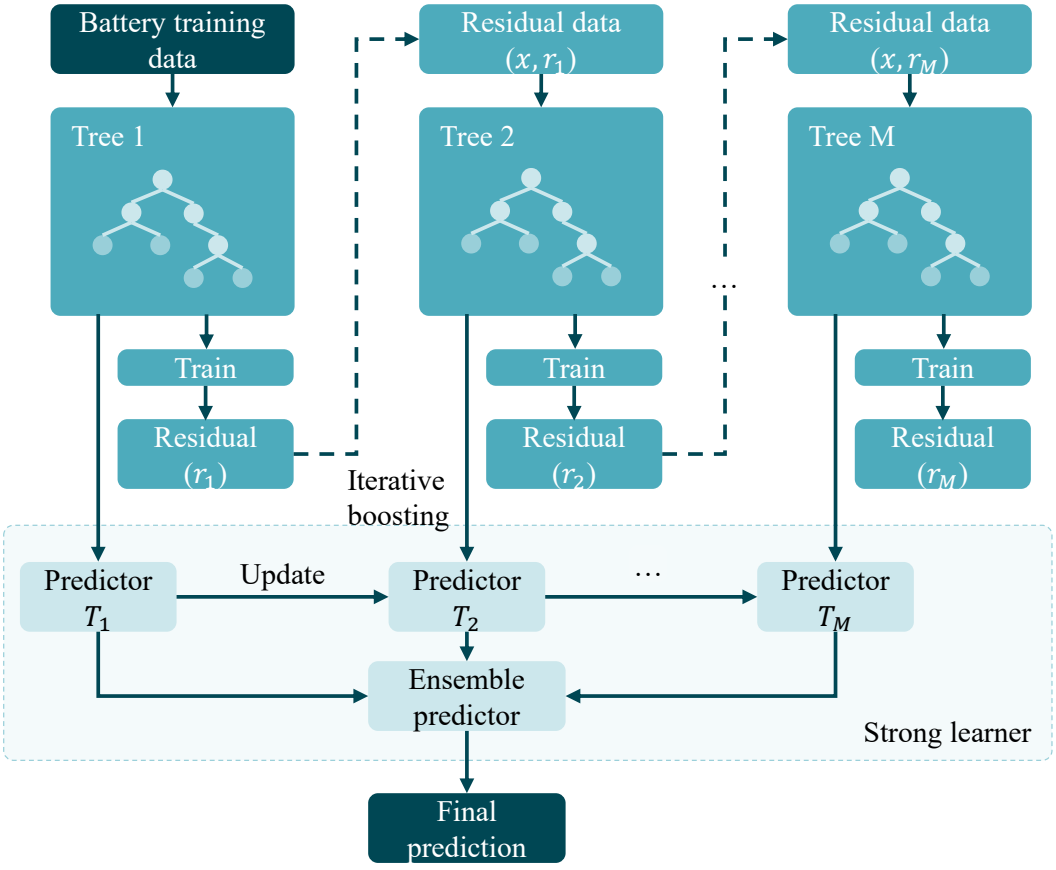


Figure 24: Schematic diagram of the XGBoost [68]

The fundamental modeling concept of XGBoost involves defining a generalized objective function. In each iteration, a new regression tree is fitted to the residuals from the previous iteration. This approach gradually minimizes the objective function, refining the model's predictions and bringing them closer to the real values. [77]

4.3.1 METHODOLOGY

The XGBoost methodology was developed by Chen and Guestrin in 2016 and is presented in their paper "*XGBoost: A Scalable Tree Boosting System*" [87]. Additional sources that explain the method and have been utilized for the following paragraphs are "*State of Health Estimation for Lithium-Ion Batteries Using an Explainable XGBoost Model with Parameter Optimization*" by Xiao et al. and "*A Survey of Ensemble*

Learning: Concepts, Algorithms, Applications, and Prospects" by Mienye and Sun [77, 80].

Let m be the number of DTs and W the functional space of all possible DTs. Then the predictive model of XGBoost can be expressed as:

$$\hat{y}_i = \sum_{i=1}^{m} f_m(x_i),\ f_m \in W \qquad \textit{Eq. 4.10}$$

Each function $f_m \in W$ represents an individual DT, defined as $f(x) = w_{q(x)}$, where $q(x)$ is a mapping function that assigns the input x to one of P leaf nodes and w is the corresponding weight at that leaf.

The objective function to be minimized consists of two parts, a loss function $l(y_i,\ \hat{y}_i)$, which measures the error between the predicted value $\hat{y}_i$ and the real value y_i, and a regularization term $\Omega(f_m)$, which reduces the model's variance and prevents overfitting by penalizing model complexity. The overall function is shown in **Eq. 4.11**, while the regularization term is further specified in **Eq. 4.12**:

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{i=1}^{m} \Omega(f_m) \qquad \textit{Eq. 4.11}$$

$$\Omega(f_m) = \gamma P + \frac{1}{2}\eta\, ||w||^2 \qquad \textit{Eq. 4.12}$$

Within the regularization term, f_m represents the function of the $m-$ th tree. P denotes the number of leaf nodes in a tree, while γ is a complexity parameter that controls the minimum loss reduction required for splitting an internal node. Lastly, η is a penalty parameter on the leaf weights w.

XGBoost adopts an additive training strategy. Given the predictions from the first $m-\ 1$ trees, ${\hat{y}_i}^{(m-1)}$, the prediction after the $m-$ th iteration is:

$$\hat{y}_i^{(m)} = \hat{y}_i^{(m-1)} + f_m(x_i) \qquad \textit{Eq. 4.13}$$

This results in the following objective function:

$$L^{(m)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(m-1)} + f_m(x_i)\right) + \Omega(f_m) \qquad \textit{Eq. 4.14}$$

To efficiently optimize this function, a second-order Taylor expansion around the current prediction $\hat{y}_i^{(m-1)}$ is employed:

$$L^{(m)} \approx \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(m-1)} + g_i f_m(x_i) + \frac{1}{2} h_i {f_m}^2(x_i)\right) + \Omega(f_m) \qquad \textit{Eq. 4.15}$$

where the first and second order derivatives of the loss function are defined as follows:

$$g_i = \partial \hat{y}_i^{(m-1)} l\left(y_i, \hat{y}_i^{(m-1)}\right), h_i = \partial^2 \hat{y}_i^{(m-1)} l\left(y_i, \hat{y}_i^{(m-1)}\right) \qquad \textit{Eq. 4.16}$$

Let $I_j = \{i | q(x_i) = j\}$ be the sample set assigned to leaf node j. Then the objective function can be reformulated in terms of the weights w_j of the leaf nodes:

$$L^{(m)} = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \eta \right) {w_j}^2 \right] + \gamma P \qquad \textit{Eq. 4.17}$$

To find the optimal weight ${w_j}^*$ for each leaf node, the first derivative is set to zero, yielding:

$${w_j}^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \eta} \qquad \textit{Eq. 4.18}$$

Substituting ${w_j}^*$ back into the objective function provides the optimal value for a given tree structure $q(x)$:

$$L^{(m)}(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \eta} + \gamma P \qquad \textit{Eq. 4.19}$$

This score is utilized to evaluate the quality of a given tree structure. However, enumerating all possible tree structures is computationally expensive. XGBoost addresses this by employing a greedy algorithm that iteratively adds branches. Suppose a candidate split divides the instance set I into I_L and I_R, the gain from this split is computed as:

$$L_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \eta} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \eta} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \eta}\right] - \gamma \qquad \textit{Eq. 4.20}$$

The algorithm selects the split point that maximizes L_{split}, which optimizes the tree structure. This process continues for each node until one of the following stopping criteria is met: the maximum tree depth is reached or the sum of sample weights falls below a predefined threshold.

4.3.2 IMPLEMENTATION DETAILS

To implement the XGBoost methodology in Python, the "*xgb.train()*" API from the "*xgboost"* library is used. Similar to the RF model, hyperparameters are selected based on standard practices for regression tasks and finetuned through prior benchmarking studies. **Table 15** summarizes the exact parameters used in the final implementation.

Parameter	*Value*	*Description*
objective	Reg:squarederror	Standard regression loss using squared error
eta	0.1	Learning rate controlling the contribution of each tree to the ensemble
max_depth	3	Maximum depth of individual trees
eval_metric	rmse	Root Mean Square Error used to monitor model performance
num_boost_round	100	Total number of boosting rounds (number of trees)

Table 15: XGBoost selected hyperparameters

The XGBoost model is trained using the "*train()*" function, with performance evaluated on the test set at each boosting round. The value for the maximum number of boosting rounds "*num_boost_round*" is selected based on the findings in [92]. As described in the methodology chapter, each boosting round introduces a new DT that corrects the residual error of the ensemble up to that point. More boosting rounds can improve the model fit, but they also increase the risk of overfitting and computational cost. To mitigate this, early stopping is implemented by tracking the RMSE (Root Mean Square Error) evaluation metric at each boosting round [93]. The iteration yielding the lowest RMSE is identified and then used for the final prediction. The learning rate "eta = 0.1" ensures gradual convergence, reducing the likelihood of overshooting the optimal solution [93]. Finally, the tree depth ("*max_depth*") is selected based on the findings in [92], representing a balanced trade-off between accuracy and overfitting control.

4.4 CONVOLUTIONAL NEURAL NETWORK

CNNs are a class of ANNs designed to automatically extract hierarchical features from structured, grid-like input data using convolutional layers. Their effectiveness in SoH estimation stems from their ability to process time-series data and identify subtle degradation patterns without requiring manual feature engineering. By sequentially applying convolution and pooling operations, CNNs generate feature maps that capture local dependencies and spatial correlations. These learned features are then combined through fully connected layers to produce accurate SoH estimates. This end-to-end learning approach provides both high accuracy and robustness. [3, 26, 38]

4.4.1 METHODOLOGY

As introduced in **Chapter 2.4**, a typical CNN consists of at least one stack of convolutional and pooling layers, followed by a fully connected layer and an output layer. In contrast to fully connected layers, where each output neuron is connected to every input neuron, convolutional layers are characterized by sparse connectivity and parameter sharing. [3, 94, 95]

Sparse connectivity implies that each neuron in a convolutional layer is connected only to a subset of the input, determined by a sliding filter, rather than to the entire input space [3]. The calculation process between the filter and the subset of the input is called "*convolution*". On the other hand, parameter sharing means that the same set of filter weights is applied across different regions of the input. This allows the CNN to detect features regardless of their position in the input space. Both characteristics are key to reduce and control overfitting. [94]

The first operation of a CNN is the convolution. Given a 1D input, such as the time-series data in the dataset of this study, and a filter $f \in R^F$, the convolution output at position p is defined as [3]:

$$o_{cov\,p} = \sum_{i=1}^{F} f_i * x_{i+p-1} \qquad \textit{Eq. 4.21}$$

Here, x represents the input matrix and $o_{cov\,p}$ is the resulting value at the corresponding location of the feature map. This process is repeated across the entire input using the same filter until the entire feature map is created. The convolutional layer, therefore, maps the raw input data to a learned feature space. [3, 94]

Following the convolution, a nonlinear activation function is applied to the feature map. The most commonly used activation function is the Rectified Linear Unit (ReLU) [3]:

$$ReLU(x) = max\ (0, x) \qquad \textit{Eq. 4.22}$$

After that, a pooling layer is applied to further downsample the size of the feature map, reducing dimensionality while retaining essential features. The pooling operation over a window of size l is defined as [3]:

$$o_{pool\,p} = pool\left((o_{p+i-1})_{i\in 1,\,2,\,\ldots,\,l}\right) \qquad \textit{Eq. 4.23}$$

where $\mathrm{pool}(\ldots)$ is the pooling function, which is typically either average (computes the average value within the window) or max pooling (takes the maximum value within the window). [3, 94]

At the end of the feature extraction process, the fully connected layer aggregates and integrates the extracted features. In these layers, each neuron is connected to every neuron of the previous layer, as in traditional ANNs. These layers integrate the learned features into a final representation used for prediction. [94, 95]

4.4.2 IMPLEMENTATION DETAILS

The CNN is implemented using the PyTorch deep learning framework in Python. The architecture follows a two layer design, similar to that proposed in [1], which offers a balanced trade-off between predictive accuracy and training time. In addition, the selected parameter values are based on multiple studies that implemented CNNs for SoH estimation [8, 96, 97]. **Table 16** provides a summary of the architecture and training parameters used in this implementation.

Parameter	*Value*	*Description*
Input shape	(8, 128)	Multivariate input with 8 features and 128 time steps
Convolutional layer 1	32 filters, kernel size 3	Extracts local patterns from the input
Pooling layer 1	Max pooling, kernel size 2	Reduces the feature maps by a factor of 2
Dropout layer 1	Dropout rate 0.25	Regularizes the model to prevent overfitting
Convolutional layer 2	64 filters, kernel size 3	Learns higher level abstractions
Pooling layer 2	Max pooling, kernel size 2	Further temporal dimensionality reduction
Dropout layer 2	Dropout rate 0.25	Additional regularization
Fully connected 1	100 units, ReLU activation	Dense representation of extracted features

Fully connected 2	1 unit, linear activation	Final output layer for regression (SoH)

Table 16: CNN selected hyperparameters

During data loading, the input tensors are reshaped from (128, 8) to PyTorch's expected format (8, 128). The model begins with a 1D convolutional layer comprising 32 filters with a kernel size of 3. This is followed by a max-pooling layer with kernel size 2, halving the temporal resolution, and a dropout layer with a rate of 0.25 to reduce overfitting. The second convolutional layer increases the number of filters to 64, maintaining the kernel site of 3. It is again followed by max pooling and dropout with identical parameters as in layer 1. This improves the model's capacity to capture more complex temporal structures. After the convolutional layers, the output is flattened and passed through a fully connected layer with 100 hidden units and ReLU activation, before the final layer, a single linear output unit, produces the regression prediction of the SoH target variable.

The configuration is intentionally kept simple and quite standard to ensure stable convergence and computational efficiency while still enabling the CNN to learn complex temporal patterns, resulting in good predictive accuracy.

4.5 INTERIM SUMMARY

This chapter outlined the methodological steps undertaken to implement the three selected SoH estimation methods, RF, XGBoost and CNN, based on the qualitative framework established in **Chapter 3**, providing the foundation for the quantitative comparison that follows in **Chapter 5**.

First, to ensure practical relevance, a real-world dataset was selected as the foundation for model development. Real-world datasets more accurately reflect operational variabilities that are often missing in laboratory datasets, which are predominantely employed when it comes to comparing SoH estimation methods. Before SoH labeling,

several preprocessing steps were applied, including data cleaning, interpolation, sliding and normalization. After that, data snippets were grouped based on vehicle ID and mileage and concatenated into charging segments occurring at the same mileage. Then, each charging segment was used to estimate the battery's current capacity, from which the SoH was calculated by dividing the current capacity by the battery's nominal capacity, as introduced in **Chapter 2.3.1**. The calculated SoH was then assigned to all corresponding charging snippets within that charging segment.

Subsequently, the chapter introduced the core principles, architectures and mathematical formulas of the three selected SoH estimation methods. In addition, it described the implementation in Python, including key hyperparameter selection based on literature recommendations and best practices.

All simulations based on that implementation, which are evaluated in the following chapter, are conducted using Python 3.13 on a local laptop with an Intel Core i7-1365U CPU (10 cores @ 1.80 GHz), 64 GB RAM and no discrete GPU (CPU-only training). The CNN was implemented using PyTorch 2.7.1, while RF and XGBoost utilized Scikit-learn 1.7.0 and XGBoost 3.0.2, respectively.

Chapter 5. TESTING RESULTS OF SOH METHODS

After the data preparation and method implementation outlined in **Chapter 4**, this chapter presents the quantitative evaluation of the three selected SoH estimation methods: RF, XGBoost and CNN.

First, **Chapter 5.1** introduces the quantitative metrics employed to evaluate the qualitative criteria that were used to select the three SoH estimation methods: accuracy, computational efficiency, reliability and scalability. These metrics are used to translate the previously defined qualitative assessments into measurable results.

In **Chapter 5.2**, the performance of each method is analyzed across the four criteria using the defined metrics. The results are compared both between methods and against the original qualitative ratings. This enables a critical validation of the qualitative framework and provides insight into the extent to which initial ratings align with real-world performance.

Finally, in **Chapter 5.3,** the findings are interpreted in the context of practical deployment. Conclusions are drawn regarding the suitability of each method for specific use cases and stakeholder needs, including implications for industrial, research and embedded system applications.

5.1 SELECTED TESTING CRITERIA

Accuracy, computational efficiency, reliability and scalability were introduced in **Chapter 3** to guide the selection of SoH estimation methods for empirical validation. In this chapter, each of these criteria is assigned a corresponding quantitative metric to enable systematic and comparable evaluation across methods. While these metrics may not capture the entirety of the original qualitative definitions, they are selected to reflect the core aspects to support meaningful analysis and comparison.

5.1.1 ACCURACY

As defined in **Chapter 3.1.2.1**, accuracy refers to the ability of a method to produce SoH predictions that closely match the actual SoH values. The criterion captures key aspects such as generalization to unseen test data, robustness to overfitting and the ability to model nonlinear and high-dimensional relationships in battery behavior.

To quantitatively evaluate accuracy, the Mean Absolute Percentage Error (MAPE) is employed. MAPE offers a normalized measure of prediction error and is expressed as a percentage of the true SoH values, making it suitable for direct comparison across different models. It is defined as [1, 27]:

$$MAPE = \frac{1}{K}\sum_{i=1}^{K}\frac{|y_i - \hat{y}_i|}{max(\epsilon, |y_i|)} \qquad \textit{Eq. 5.1}$$

where y_i and $\hat{y}_i$ represent the actual and predicted SoH value, K is the total number of samples and ϵ is a small positive constant introduced to prevent division by zero. A lower MAPE value indicates higher accuracy.

5.1.2 COMPUTATIONAL EFFICIENCY

Computational efficiency refers to the resource demands of a SoH estimation method during both training and inference (see **Chapter 3.1.2.2**). This directly influences the method's feasibility for real-time applications. Relevant aspects include training time, inference latency, memory usage and structural complexity.

To quantitatively assess this criterion, both training and inference runtimes are measured for each method [1, 20]. These measurements are recorded using Python's built-in "*time*" module, which captures the elapsed time required to complete the training and inference processes.

While runtime represents only one dimension of computational efficiency, it serves as a direct and interpretable proxy, particularly when methods are executed under

consistent hardware and software conditions. In this context, shorter runtimes indicate that a method is more efficient.

5.1.3 RELIABILITY

Chapter 3.1.2.5 defines reliability as the ability of a method to maintain consistent performance across varied input conditions. It focuses on robustness to noise, missing data and changes in operational context (e.g., temperature or usage profiles).

To evaluate this criterion, each of the selected SoH estimation methods is trained and tested independently on the three real-world datasets introduced in **Chapter 4.1**. These datasets represent varying operational conditions, enabling a systematic comparison of how each method performs under different boundary conditions.

For each dataset, the MAPE is computed on the test portion. By analyzing its variation across the three datasets, this study assesses the method's reliability. A method is considered more reliable if it achieves low and stable error across all datasets.

5.1.4 SCALABILITY

As outlined in **Chapter 3.1.2.6**, scalability evaluates a method's ability to maintain predictive performance and computational feasibility as dataset size, input dimensionality and system scale increase.

To assess scalability, the training dataset is increased incrementally at proportions of 10%, 25%, 50%, 75% and 100%. Each method is trained and evaluated independently on these subsets. Two metrics are recoded: MAPE, to observe changes in predictive accuracy, and training time, to monitor computational demand as the dataset size grows.

This enables a detailed analysis of each method's response to increasing data volume. A method is considered scalable if accuracy is improved or maintained with more data while showing manageable growth in training time. Additionally, this analysis identifies diminishing returns in performance. For instance, if a method's accuracy

plateaus after 50% of the data, further training may be unnecessary. Such insights are particularly relevant for real-world deployment, where computational resources or data availability may be limited.

5.2 RESULTS AND DISCUSSION

After defining the quantitative metrics for each evaluation criterion, this chapter presents and discusses the empirical results for the selected SoH estimation methods. To ensure a fair comparison, all methods were implemented using simple configurations without hyperparameter optimization or architectural tuning. This enables a controlled assessment of each method's performance metrics under identical conditions. However, it is crucial to note that this may lead to non-optimal performance outcomes. As such, the reported results should be interpreted as representatives of baseline capability and not peak potential.

In addition to the direct performance comparison, this chapter revisits the qualitative assessment framework introduced in **Chapter 3**, which enables a critical reflection on how well the initial qualitative ratings align with the empirical evidence obtained from real-world data. It is important to note that the analysis conducted only allows for relative comparison among the three selected methods, rather than an absolute evaluation against the full scoring scale. For absolute placement, a larger sample of SoH estimation methods is required to define statistically meaningful boundaries.

5.2.1 ACCURACY

The quantitative results of the accuracy criterion are presented in **Figure 25**. These results were obtained using Dataset 1, which is the largest of the three datasets introduced in **Chapter 4.1**, offering the most robust basis for reliable evaluation.

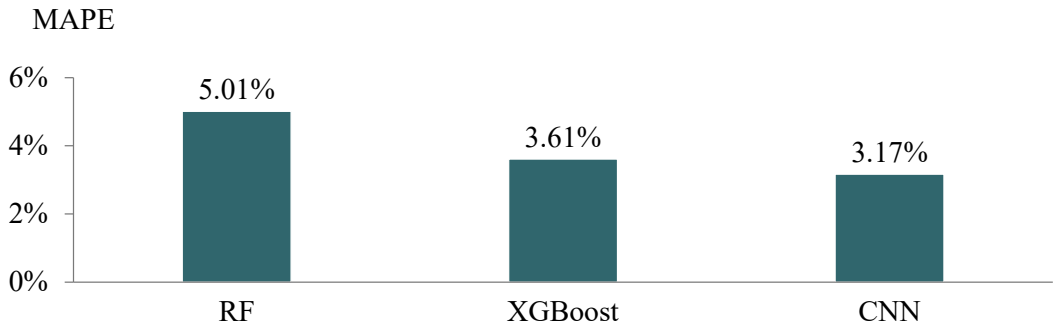


Figure 25: Results of MAPE (%) quantitatively measuring the accuracy of RF, XGBoost and CNN using dataset 1

The findings indicate that CNN achieved the lowest MAPE at 3.17%, closely followed by XGBoost with 3.61%. RF exhibits slightly worse accuracy with a MAPE of 5.01%. This suggests that CNN and XGBoost provide similar levels of predictive precision, whereas RF's performance is inferior under the current evaluation conditions.

This performance confirms the qualitative assessments of the accuracy criterion outlined in **Chapter 3**. CNN, initially rated "Very High" due to its ability to automatically extract hierarchical features and subtle patterns in high-dimensional inputs, achieves the highest accuracy, thereby validating its rating. Similarly, XGBoost confirms its qualitative rating of "Very High", demonstrating comparable accuracy with only marginally higher prediction error. Its strong predictive accuracy is based on its GBDT framework, which models nonlinearities effectively. Finally, RF, which was rated as "High", also aligns with its expectation. Although it yields less accurate results than CNN and XGBoost, it still provides reasonably precise SoH estimates on the used dataset.

While MAPE provides a normalized measure of average prediction error magnitude relative to true SoH values, it does not fully capture all facets of the accuracy criterion as defined in **Chapter 3.1.2.1**. For example, it summarizes average deviation but may not reflect how prediction errors evolve over time or vary across different stages of battery aging. To address this, complementary quantitative assessments are beneficial. Metrics such as RMSE can provide additional perspectives on the model's overall fit

and ability to approximate the variance in SoH data. In addition, temporal analysis of prediction errors, such as evaluating error trends over cycles or calendar time, can reveal whether models maintain consistent accuracy or show systematic deviation during specific aging phases.

In summary, the MAPE results quantitatively validate the qualitative assessment that CNN and XGBoost exhibit comparable and strong predictive accuracy, while RF achieves slightly reduced accuracy.

5.2.2 COMPUTATIONAL EFFICIENCY

The quantitative results for the computational efficiency criterion are summarized in **Figure 26**. Similar to the accuracy criterion, Dataset 1 is chosen to ensure comparability and consistency.

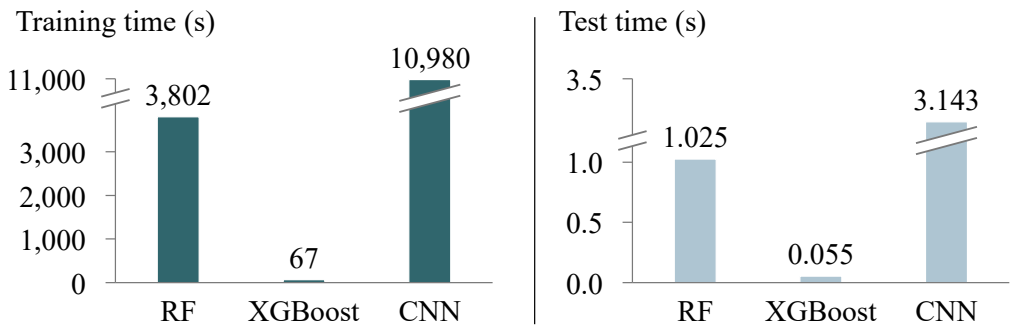


Figure 26: Results of training and test time (in seconds) quantitatively measuring the computational efficiency of RF, XGBoost and CNN using dataset 1

The findings show that XGBoost has the shortest training time (67s) compared to RF (3,802s) and CNN (10,980s). In addition, it also achieved the fastest inference time at 0.055s. This indicates that XGBoost is the most computationally efficient among the three methods, with significantly lower resource demands for both training and inference. RF exhibits a notably higher computational load, particularly during training, while CNN shows the highest overall demand, with runtimes tripling those of RF.

These results correspond largely to the qualitative assessment of **Chapter 3**. CNN was categorized as "Very Low" due to its structural complexity and substantial training demands. The empirical results confirm this classification. RF was rated as "Moderate", which is consistent with its comparatively low inference and long training time. XGBoost, however, previously also rated as "Moderate", demonstrates substantially better performance than RF in terms of both inference and training time. This suggests that its qualitative rating may need to be reconsidered, potentially as "High".

It is important to note that runtime does not fully capture all aspects of the computational efficiency criterion. Additional dimensions, such as the number of model parameters, memory usage and reliance optimization routines, are also central to computational demand but are not reflected in the runtime metric. Furthermore, structural complexity, such as the depth of a CNN or the number of boosting rounds in XGBoost, can significantly affect hardware requirements. To provide a more comprehensive analysis, future evaluations should incorporate additional metrics such as peak memory usage or number of trainable parameters, which better capture structural complexity, particularly in embedded or real-time systems.

In summary, XGBoost emerges as the most computationally efficient method based on runtime, followed by RF and CNN. This ordering results in minor adjustments to the rating of XGBoost in the qualitative categorization but overall supports the validity of the qualitative evaluation framework for this criterion.

5.2.3 RELIABILITY

The quantitative results of the reliability criterion are presented in **Table 17**. They were obtained by independently training and evaluating the selected SoH estimation methods on each of the three real-world datasets introduced in **Chapter 4.1**. These datasets differ in sampling frequency, fleet size and usage context, offering a meaningful basis for evaluating reliability under varied operational conditions.

Dataset	*RF*	*XGBoost*	*CNN*
1	5.01%	3.61%	3.17%
2	8.46%	7.08%	6.09%
3	7.13%	6.49%	9.63%

Table 17: Results of MAPE (%) to quantitatively measure the reliability of RF, XGBoost and CNN across datasets

The results indicate that XGBoost and RF exhibit similar performance (~3.5 p.p. MAPE variation) across the datasets, showing robust generalization. Although their absolute MAPE values differ slightly, their ability to maintain a comparably consistent error profile across operational contexts indicates strong reliability. On the other hand, CNN, while achieving the lowest error on Dataset 1, displays the largest variation across datasets, with MAPE rising from 3.17% to 9.64%. This fluctuation implies increased sensitivity to changing data characteristics, particularly in Dataset 3, which includes fewer vehicles and more heterogeneous charging behavior typical of private users.

The observed reliability aligns partially with the qualitative assessments discussed in **Chapter 3**. CNN was rated “Very High” based on its adaptability and robustness to noise during training. However, the quantitative results suggest that while CNN performs well on complex data, its generalization across operational profiles, particularly in smaller or more heterogeneous datasets, may be limited. Therefore, its reliability rating may require reclassification from "Very High" to "Moderate" or "High". XGBoost and RF, initially rated as “High” and “Very High”, maintain more stability across all datasets and should both be rated as “High” after validation.

While MAPE is effective for capturing average deviation, it does not reflect error volatility or sensitivity to specific operating conditions. Therefore, additional metrics should be considered for a more comprehensive reliability analysis. These include the standard deviation of MAPE across individual charging cycles, RMSE to capture

vulnerability to outliers or noise and controlled input perturbation experiments to test robustness under missing data scenarios.

In summary, the reliability assessment reveals that RF and XGBoost maintain stable performance across varied datasets and fulfill their qualitative classification as "High" in reliability. CNN, despite its accuracy in controlled conditions, shows reduced stability under operational variation, indicating a need to revise its qualitative rating.

5.2.4 SCALABILITY

The quantitative results of the scalability criterion are shown in **Table 18**. As with the assessments of accuracy and computational efficiency, Dataset 1 was selected as the basis for analysis due to its size and comprehensiveness. To assess scalability, the training dataset size was incrementally increased from 10% to 100% and each method was trained independently on the subsets.

	RF		*XGBoost*		*CNN*	
Fraction	MAPE	Training time	MAPE	Training time	MAPE	Training time
0.1	4.95%	248.8s	3.76%	16.6s	3.56%	933.5s
0.25	4.68%	802.8s	3.59%	29.9s	3.33%	3,556.8s
0.5	4.95%	1,775.3	3.61%	57.5s	3.56%	4,934.9s
0.75	5.00%	3,801.5	3.60%	51.3s	3.38%	7,308.6s
1.0	5.01%	4,064.1	3.61%	76.5s	3.79%	9,962.4s

Table 18: Results of MAPE (%) and training time (s) to quantitatively measure the scalability of RF, XGBoost and CNN using dataset 1

XGBoost demonstrates the best scalability among the evaluated methods. Its training time increases linearly with dataset size but remains comparatively fast (< 80s). In addition, MAPE remains stable across all training sizes, varying only marginally

between 3.59% and 3.76%. The early convergence (after 25%) reflects robust generalization and computational efficiency at scale. RF shows steeper increases in training time as the dataset grows, surpassing 4,000s at full dataset size. While its accuracy remains stable, the best result is achieved already at the 25% training fraction. This suggests early saturation in learning capacity. This plateau does not indicate performance degradation, but the high training times compared to marginal accuracy improvements signal limited efficiency at scale. Similarly to RFs, CNNs show a sharp increase in training time with dataset size, reaching nearly 10,000s at full scale. In contrast to RF and XGBoost, CNN shows non-monotonic trends in accuracy. MAPE improves slightly at the 25% level (3.33%) and again at 75% (3.38%) but worsens at full scale to 3.79%. This suggests sensitivity to optimization dynamics and potential overfitting or instability as the dataset size increases. While CNNs are structurally capable of handling large, high-dimensional data, these findings indicate that scalability in practice depends on proper model configuration and sufficient computational resources.

The observed scalability results partially validate the qualitative assessments of **Chapter 3**. XGBoost, previously rated "Very High" in scalability, demonstrates consistent accuracy and low training times across dataset sizes, confirming its classification. Similarly, RF supports its previous rating ("High") by scaling functionally, but with slightly diminishing accuracy and increasing computational demand. In contrast, the "Very High" rating of CNNs requires revision. While CNNs offer competitive accuracy, their rapidly increasing training time and nonlinear accuracy progression indicate practical limitations. A revised classification of "High" is more appropriate based on the findings.

5.3 USE CASE IDENTIFICATION BASED ON EMPIRICAL RESULTS

The validation of the qualitative assessment of RF, XGBoost and CNN across accuracy, computational efficiency, reliability and scalability provides a strong foundation for aligning each method's performance with potential real-world deployment scenarios.

RF demonstrates consistent reliability across diverse datasets, maintaining stable accuracy, though with slightly higher prediction errors compared to XGBoost and CNN. Its inference efficiency is moderate, while training times increase notably with larger datasets. These results suggest that RF is suitable for applications where robustness and stability across varying operational conditions are required and inference has to be efficient, while training can be conducted offline. An example application is onboard BMS in BEVs, where predictable and reliable SoH estimation is critical under limited computational resources.

XGBoost consistently achieves high accuracy with the shortest training and inference times among the evaluated methods. It scales well with dataset size and maintains stable predictive performance across different operational contexts. This balance between the criteria makes XGBoost the best option for scenarios requiring rapid inference, while having low resource demands. Representative use cases include third-party diagnostic tools or workshop-based SoH assessment devices.

CNN achieves the lowest prediction errors, demonstrating strong capabilities in modeling complex battery degradation patterns. However, CNN training demands are significantly higher, and its accuracy varies more substantially across datasets. These findings imply CNNs are best suited for environments with access to large, high-quality datasets and sufficient computational resources, such as centralized fleet analytics platforms, predictive maintenance systems or academic research.

While this analysis supports informed recommendations for method selection tailored to specific application contexts, it is important to recognize that the selected quantitative

metrics are limited in scope. They do not capture other crucial factors such as memory usage, noise sensitivity or time-dependency of prediction errors, elements that heavily influence deployment feasibility. Consequently, further refinement and expansion of the metrics are needed to incorporate these additional dimensions. Enhancing these metrics will enable more comprehensive decision support for selecting SoH estimation methods that optimally balance predictive performance with operational constraints across real-world scenarios.

Chapter 6. CONCLUSION AND OUTLOOK

Accurate estimation of a battery's SoH is critical to prevent failure and ensure reliable operation throughout its lifecycle [2–4, 13, 19]. Since SoH cannot be directly measured, a wide range of estimation techniques has been proposed in literature, which can be categorized into experimental, model-based, and hybrid approaches [1–5, 8, 19, 21, 23, 24]. Among these, model-based SoH estimation methods have gained importance due to their non-invasive nature [12].

However, the increasing variety of model-based SoH estimation approaches has led to inconsistent and fragmented evaluation practices. Most studies focus on individual performance metrics like accuracy, which do not sufficiently capture the multi-dimensional requirements for practical deployment [4, 5, 9, 14, 23, 24, 27]. In addition, the predominant reliance on laboratory datasets for model development often fails to reflect the operational variability and complexity of real-world environments, resulting in poor model generalization when deployed in practice [8, 24].

This thesis addressed these challenges by developing a multi-criteria evaluation framework that enables use-case specific selection of model-based SoH estimation methods. Afterwards, it validated the qualitative method assessments for selected estimation methods through empirical evaluation on a large, real-world battery dataset.

Following the theoretical foundation presented in **Chapter 2**, which introduced the fundamental concepts of battery ageing and SoH estimation as well as characterized relevant model-based approaches, **Chapter 3** developed the evaluation framework. Based on insights from five selected studies, six practically significant evaluation criteria were derived: accuracy, computational efficiency, interpretability, data requirements, reliability and scalability. Each model-based method introduced earlier was then qualitatively assessed across these six dimensions using literature benchmarks, comparative studies and performance indicators. To select candidates for

implementation, four of the six criteria (accuracy, computational efficiency, reliability and scalability) were prioritized given the availability of a large real-world dataset and the knowledge to implement the methods. Based on this assessment, RF, XGBoost and CNN were identified as the most suitable candidates.

Chapter 4 detailed the empirical implementation, including preprocessing and SoH labeling of the dataset and the practical deployment of each method in Python. **Chapter 5** then presented the quantitative performance analysis, using defined quantitative metrics aligned with the four prioritized criteria. The empirical results largely validated the qualitative assessments from **Chapter 3**, requiring only minor adjustments. CNN achieved the highest accuracy, followed closely by XGBoost, while RF showed comparatively lower but acceptable predictive precision. XGBoost significantly outperformed expectations for computational efficiency, exhibiting fast training and inference times. RF showed moderate resource demands, while CNN was confirmed as the most computationally intensive model. In terms of reliability, both RF and XGBoost performed consistently across varying operational conditions, whereas CNN underperformed its qualitative assessment. Regarding scalability, XGBoost maintained high accuracy and efficiency across different dataset sizes, RF showed stable accuracy but increased training time on larger datasets and CNN's scalability proved limited in both accuracy and runtime.

These findings support differentiated recommendations for method selection in practice. RF is well-suited for embedded applications like onboard BMS in BEVs, where method stability and robustness under varying operational conditions are essential. XGBoost offers an efficient option for diagnostic tools and low-resource environments, combining speed, accuracy and reliability. CNNs are best deployed in data-rich, computationally flexible settings like fleet analytics platforms.

At the same time, the results highlight the limitations of the selected quantitative metrics. While they offer meaningful insights, certain aspects of the initial criteria definitions are not covered. Deployment decisions in real-world systems also depend

on additional factors such as memory usage or noise sensitivity, which are currently not considered. Further expansion and refinement of the quantitative metrics across the four criteria are needed to comprehensively represent the initial criteria definition.

Building on this limitation, further research should focus on three key directions. First, expand the current quantitative metrics by incorporating the described dimensions to enable a more complete and accurate assessment of the evaluation criteria. Second, apply more of the described model-based SoH estimation methods to the same dataset. This will allow for the development of quantitative thresholds for the framework's rating scales, to transition the purely qualitative assessment to a hybrid qualitative-quantitative framework, allowing for absolute, rather than relative, performance comparison across methods. And third, greater effort should be placed on obtaining usable real-world datasets for SoH estimation method development, as such datasets are currently limited yet essential for testing practical applicability.

Chapter 7. BIBLIOGRAPHY

[1] P. Eleftheriadis, M. Gangi, S. Leva, A. V. Rey, E. Groppo und L. Grande, „Comparative study of machine learning techniques for the state of health estimation of Li-Ion batteries“, in *Electric Power Systems Research*, DOI: 10.1016/j.epsr.2024.110889, 2024

[2] N. Noura, L. Boulon und S. Jemeï, „A Review of Battery State of Health Estimation Methods: Hybrid Electric Vehicle Challenges“, in *World Electric Vehicle Journal*, DOI: 10.3390/wevj11040066, 2020

[3] X. Sui, S. He, S. B. Vilsen, J. Meng, R. Teodorescu und D.-I. Stroe, „A review of non-probabilistic machine learning-based state of health estimation techniques for Lithium-ion battery“, in *Applied Energy*, DOI: 10.1016/j.apenergy.2021.117346, 2021

[4] K. Yang, L. Zhang, Z. Zhang, H. Yu, W. Wang, M. Ouyang, C. Zhang, Q. Sun, X. Yan, S. Yang und X. Liu, „Battery State of Health Estimate Strategies: From Data Analysis to End-Cloud Collaborative Framework“, in *Batteries*, DOI: 10.3390/batteries9070351, 2023

[5] S. K. Pradhan und B. Chakraborty, „Battery management strategies: An essential review for battery state of health monitoring techniques“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2022.104427, 2022

[6] J. Wu, J. Chen, X. Feng, H. Xiang und Q. Zhu, „State of health estimation of lithium-ion batteries using Autoencoders and Ensemble Learning“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2022.105708, 2022

[7] H. M. Hussein, A. Aghmadi, M. S. Abdelrahman, S. M. S. H. Rafin und O. Mohammed, „A review of battery state of charge estimation and management systems: Models and future prospective“, in *Wiley Interdisciplinary Reviews: Energy and Environment*, DOI: 10.1002/wene.507, 2024

[8] Y. Lu, D. Guo, G. Xiong, Y. Wei, J. Zhang, Y. Wang und M. Ouyang, „Towards real-world state of health estimation: Part 2, system level method using electric vehicle field data“, in *eTransportation*, DOI: 10.1016/j.etran.2024.100361, 2024

[9] J. Tao, S. Wang, W. Cao, P. Takyi-Aninakwa, C. Fernandez und J. M. Guerrero, „A comprehensive review of state-of-charge and state-of-health estimation for lithium-ion battery energy storage systems“, in *Ionics*, DOI: 10.1007/s11581-024-05686-z, 2024

[10] M. Berecibar, I. Gandiaga, I. Villarreal, N. Omar, J. van Mierlo und P. van den Bossche, „Critical review of state of health estimation methods of Li-ion batteries

for real applications“, in *Renewable and Sustainable Energy Reviews*, DOI: 10.1016/j.rser.2015.11.042, 2016

[11] M.-F. Ng, J. Zhao, Q. Yan, G. J. Conduit und Z. W. Seh, „Predicting the state of charge and health of batteries using data-driven machine learning“, in *Nature Machine Intelligence*, DOI: 10.1038/s42256-020-0156-7, 2020

[12] F. Liu, Y. Liu, W. Su, C. Jiao und Y. Liu, „Online estimation of lithium-ion batteries state of health during discharge“, in *International Journal of Energy Research*, DOI: 10.1002/er.6502, 2021

[13] T. Oji, Y. Zhou, S. Ci, F. Kang, X. Chen und X. Liu, „Data-Driven Methods for Battery SOH Estimation: Survey and a Critical Analysis“, in *IEEE Access*, DOI: 10.1109/access.2021.3111927, 2021

[14] L. Yao, S. Xu, A. Tang, F. Zhou, J. Hou, Y. Xiao und Z. Fu, „A Review of Lithium-Ion Battery State of Health Estimation and Prediction Methods“, in *World Electric Vehicle Journal*, DOI: 10.3390/wevj12030113, 2021

[15] L. Cong, W. Wang und Y. Wang, „A review on health estimation techniques of end-of-first-use lithium-ion batteries for supporting circular battery production“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2024.112406, 2024

[16] M. Zhang, D. Yang, J. Du, H. Sun, L. Li, L. Wang und K. Wang, „A Review of SOH Prediction of Li-Ion Batteries Based on Data-Driven Algorithms“, in *Energies*, DOI: 10.3390/en16073167, 2023

[17] L. Ungurean, G. Cârstoiu, M. V. Micea und V. Groza, „Battery state of health estimation: a structured review of models, methods and commercial devices“, in *International Journal of Energy Research*, DOI: 10.1002/er.3598, 2017

[18] A. Manoharan, K. M. Begam, V. R. Aparow und D. Sooriamoorthy, „Artificial Neural Networks, Gradient Boosting and Support Vector Machines for electric vehicle battery state estimation: A review“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2022.105384, 2022

[19] Z. Deng, X. Hu, P. Li, X. Lin und X. Bian, „Data-Driven Battery State of Health Estimation Based on Random Partial Charging Data“, in *IEEE Transactions on Power Electronics*, DOI: 10.1109/tpel.2021.3134701, 2022

[20] S. Guo und L. Ma, „A comparative study of different deep learning algorithms for lithium-ion batteries on state-of-charge estimation“, in *Energy*, DOI: 10.1016/j.energy.2022.125872, 2023

[21] T. Alsuwian, S. Ansari, Ammirrul Atiqi Mohd Zainuri, Muhammad, A. Ayob, A. Hussain, M. S. Hossain Lipu, A. R. Alhawari, A. Almawgani, S. Almasabi und A. Taher Hindi, „A review of expert hybrid and co-estimation techniques for SOH and RUL estimation in battery management system with electric vehicle

application“, in *Expert Systems with Applications*, DOI: 10.1016/j.eswa.2023.123123, 2024

[22] A. Haraz, K. Abualsaud und A. Massoud, „State-of-Health and State-of-Charge Estimation in Electric Vehicles Batteries: A Survey on Machine Learning Approaches“, in *IEEE Access*, DOI: 10.1109/ACCESS.2024.3486989, 2024

[23] O. Demirci, S. Taskin, E. Schaltz und B. Acar Demirci, „Review of battery state estimation methods for electric vehicles-Part II: SOH estimation“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2024.112703, 2024

[24] J. Gong, B. Xu, F. Chen und G. Zhou, „Predictive Modeling for Electric Vehicle Battery State of Health: A Comprehensive Literature Review“, in *Energies*, DOI: 10.3390/en18020337, 2025

[25] S. G. Padder, J. Ambulkar, A. Banotra, S. Modem, S. Maheshwari, K. Jayaramulu und C. Kundu, „Data-Driven Approaches for Estimation of EV Battery SoC and SoH: A Review“, in *IEEE Access*, DOI: 10.1109/access.2025.3539528, 2025

[26] C. Liu, H. Li, K. Li, Y. Wu und B. Lv, „Deep Learning for State of Health Estimation of Lithium-Ion Batteries in Electric Vehicles: A Systematic Review“, in *Energies*, DOI: 10.3390/en18061463, 2025

[27] E. Kim und S. Jung, „Battery State of Health Estimation Methods: Implementation and Comparison of 11 Algorithms“, in *IEEE Access*, DOI: 10.1109/access.2025.3541631, 2025

[28] Korthauer, R. (Hrsg.), „Handbuch Lithium-Ionen-Batterien“, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013

[29] T. Rahman und T. Alharbi, „Exploring Lithium-Ion Battery Degradation: A Concise Review of Critical Factors, Impacts, Data-Driven Degradation Estimation Techniques, and Sustainable Directions for Energy Storage Systems“, in *Batteries*, DOI: 10.3390/batteries10070220, 2024

[30] J. Zhang und K. Li, „State-of-Health Estimation for Lithium-Ion Batteries in Hybrid Electric Vehicles—A Review“, in *Energies*, DOI: 10.3390/en17225753, 2024

[31] Q. Li, R. Song und Y. Wei, „A review of state-of-health estimation for lithium-ion battery packs“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2025.116078, 2025

[32] J. Ren, J. Ma, H. Wang, T. Yu und K. Wang, „A comprehensive review on research methods for lithium-ion battery of state of health estimation and end of life prediction: Methods, properties, and prospects“, in *Protection and Control of Modern Power Systems*, DOI: 10.23919/PCMP.2024.000211, 2024

[33] M. Doppelbauer3333, „Grundlagen der Elektromobilität“, Wiesbaden: Springer Fachmedien Wiesbaden, 2020

[34] P. Nur Halimah, S. Rahardian und B. A. Budiman, „Battery Cells for Electric Vehicles“, in *International Journal of Sustainable Transportation Technology*, DOI: 10.31427/IJSTT.2019.2.2.3, 2019

[35] S. Baazouzi, N. Feistel, J. Wanner, I. Landwehr, A. Fill und K. P. Birke, „Design, Properties, and Manufacturing of Cylindrical Li-Ion Battery Cells—A Generic Overview“, in *Batteries*, DOI: 10.3390/batteries9060309, 2023

[36] L. Su, Y. Xu und Z. Dong, „State-of-health estimation of lithium-ion batteries: A comprehensive literature review from cell to pack levels“, in *Energy Conversion and Economics*, DOI: 10.1049/enc2.12125, 2024

[37] H. Löbberding, S. Wessel, C. Offermanns, M. Kehrer, J. Rother, H. Heimes und A. Kampker, „From Cell to Battery System in BEVs: Analysis of System Packing Efficiency and Cell Types“, in *World Electric Vehicle Journal*, DOI: 10.3390/wevj11040077, 2020

[38] C. Shan, C. S. Chin, V. Mohan und C. Zhang, „Review of Various Machine Learning Approaches for Predicting Parameters of Lithium-Ion Batteries in Electric Vehicles“, in *Batteries*, DOI: 10.3390/batteries10060181, 2024

[39] J. S. Edge, S. O'Kane, R. Prosser, N. D. Kirkaldy, A. N. Patel, A. Hales, A. Ghosh, W. Ai, J. Chen, J. Yang, S. Li, M.-C. Pang, L. Bravo Diaz, A. Tomaszewska, M. W. Marzook, K. N. Radhakrishnan, H. Wang, Y. Patel, B. Wu und G. J. Offer, „Lithium ion battery degradation: what you need to know“, in *Physical chemistry chemical physics : PCCP*, DOI: 10.1039/D1CP00359C, 2021

[40] C. R. Birkl, M. R. Roberts, E. McTurk, P. G. Bruce und D. A. Howey, „Degradation diagnostics for lithium ion cells“, in *Journal of Power Sources*, DOI: 10.1016/j.jpowsour.2016.12.011, 2017

[41] I. B. Mansir und P. C. Okonkwo, „Component Degradation in Lithium-Ion Batteries and Their Sustainability: A Concise Overview“, in *Sustainability*, DOI: 10.3390/su17031000, 2025

[42] X. Han, L. Lu, Y. Zheng, X. Feng, Z. Li, J. Li und M. Ouyang, „A review on the key issues of the lithium ion battery degradation among the whole life cycle“, in *eTransportation*, DOI: 10.1016/j.etran.2019.100005, 2019

[43] R. Vaghela, P. Ramani, J. Sarda, K. L. Hui und M. Sain, „Analysis of State-of-Health Estimation Approaches and Constraints for Lithium-Ion Batteries in Electric Vehicles“, in *International Journal of Energy Research*, DOI: 10.1155/2024/6488186, 2024

[44] P. Dini, A. Colicelli und S. Saponara, „Review on Modeling and SOC/SOH Estimation of Batteries for Automotive Applications“, in *Batteries*, DOI: 10.3390/batteries10010034, 2024

[45] E. H. E. Bayoumi, M. de Santis und H. Awad, „A Brief Overview of Modeling Estimation of State of Health for an Electric Vehicle's Li-Ion Batteries", in *World Electric Vehicle Journal*, DOI: 10.3390/wevj16020073, 2025

[46] A. Xiao und W. Liu, „Review of SOH Prediction Methods for Lithium-Ion Batteries", DOI: 10.1109/ACEEE62329.2024.10652029

[47] L. Kong, Y. Luo, S. Fang, T. Niu, G. Chen, L. Yang und R. Liao, „State estimation of lithium-ion battery for shipboard applications: Key challenges and future trends", in *Green Energy and Intelligent Transportation*, DOI: 10.1016/j.geits.2024.100192, 2025

[48] A. Jokar, B. Rajabloo, M. Désilets und M. Lacroix, „Review of simplified Pseudo-two-Dimensional models of lithium-ion batteries", in *Journal of Power Sources*, DOI: 10.1016/j.jpowsour.2016.07.036, 2016

[49] M.-S. Lee, J. Oh, D.-C. Lee, K. Lee, S. Park und Y. Hong „Forward and Inverse Simulation of Pseudo-Two-Dimensional Model of Lithium-Ion Batteries Using Neural Networks" 02.12.2024

[50] T. H. Dar und S. Singh, „A comprehensive review, perspectives and future directions of battery characterization and parameter estimation", in *Journal of Applied Electrochemistry*, DOI: 10.1007/s10800-024-02217-6, 2025

[51] S. E. Sabev, N. D. Madjarov und D. D. Dankov, „Overview of modern methods for state-of-health and state-of-charge electric vehicles applications", in *E+E*, DOI: 2024

[52] Q. Liu, Y. Tian, Y. Chai, M. Liu und L. Sun, „Design of unscented Kalman filter based on the adjustments of the number and placements of the sampling points", in *ISA Transactions*, DOI: 10.1016/j.isatra.2020.08.013, 2021

[53] H. Pan, C. Chen und M. Gu, „A State of Health Estimation Method for Lithium-Ion Batteries Based on Improved Particle Filter Considering Capacity Regeneration", in *Energies*, DOI: 10.3390/en14165000, 2021

[54] X. Xu, C. Yu, S. Tang, X. Sun, X. Si und L. Wu, „Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Wiener Processes with Considering the Relaxation Effect", in *Energies*, DOI: 10.3390/en12091685, 2019

[55] S. Park, J. Lee und S. Heo5555, „Gaussian Process Regression-Based Lithium-Ion Battery End-of-Life Prediction Model under Various Operating Conditions", 2024

[56] Y. Wu, D. Bai, K. Zhang, Y. Li und F. Yang, „Advancements in the estimation of the state of charge of lithium-ion battery: a comprehensive review of traditional and deep learning approaches", in *Journal of Materials Informatics*, DOI: 10.20517/jmi.2024.84, 2025

[57] Q. Zhai und Z.-S. Ye, „RUL Prediction of Deteriorating Products Using an Adaptive Wiener Process Model“, in *IEEE Transactions on Industrial Informatics*, DOI: 10.1109/TII.2017.2684821, 2017

[58] X. Xu, C. Yu, S. Tang, X. Sun, X. Si und L. Wu, „State-of-Health Estimation for Lithium-Ion Batteries Based on Wiener Process With Modeling the Relaxation Effect“, in *IEEE Access*, DOI: 10.1109/access.2019.2923095, 2019

[59] Y. Zhu, S. Liu, K. Wei, H. Zuo, R. Du und X. Shu, „A novel based-performance degradation Wiener process model for real-time reliability evaluation of lithium-ion battery“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2022.104313, 2022

[60] X.-S. Si, W. Wang, C.-H. Hu und D.-H. Zhou, „Remaining useful life estimation – A review on the statistical data driven approaches“, in *European Journal of Operational Research*, DOI: 10.1016/j.ejor.2010.11.018, 2011

[61] Y. Han, C. Ma, S. Tang, F. Wang, X. Sun und X. Si, „Residual life estimation of lithium-ion batteries based on nonlinear Wiener process with measurement error“, in *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, DOI: 10.1177/1748006X221080345, 2023

[62] Y. Huang, W. Liu und X. Zang „Improving Fuzzy Rule Classifier with Brain Storm Optimization and Rule Modification“ 10.02.2024

[63] A. Zenati, P. Desprez und H. Razik, „Estimation of the SOC and the SOH of li-ion batteries, by combining impedance measurements with the fuzzy logic inference“: IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, S. 1773–1778. IEEE, 2010

[64] A. J. Salkind, C. Fennie, P. Singh, T. Atwater und D. E. Reisner, „Determination of state-of-charge and state-of-health of batteries by fuzzy logic methodology“, in *Journal of Power Sources*, DOI: 10.1016/S0378-7753(99)00079-8, 1999

[65] I. Marri, E. Petkovski, L. Cristaldi und M. Faifer, „Comparing Machine Learning Strategies for SoH Estimation of Lithium-Ion Batteries Using a Feature-Based Approach“, in *Energies*, DOI: 10.3390/en16114423, 2023

[66] Y. Xie, S. Wang, G. Zhang, P. Takyi-Aninakwa, C. Fernandez und F. Blaabjerg, „A review of data-driven whole-life state of health prediction for lithium-ion batteries: Data preprocessing, aging characteristics, algorithms, and future challenges“, in *Journal of Energy Chemistry*, DOI: 10.1016/j.jechem.2024.06.017, 2024

[67] X. Qiang, Y. Tang, L. Wu und Z. Lyu, „Li-Ion Battery State of Health Estimation Using Hybrid Decision Tree Model Optimized by Bayesian Optimization“, in *Energy Technology*, DOI: 10.1002/ente.202301065, 2024

[68] C. Lin, J. Xu, D. Jiang, J. Hou, Y. Liang, Z. Zou und X. Mei, „Multi-model ensemble learning for battery state-of-health estimation: Recent advances and perspectives“, in *Journal of Energy Chemistry*, DOI: 10.1016/j.jechem.2024.09.021, 2025

[69] G. Wang, Z. Lyu und X. Li, „An Optimized Random Forest Regression Model for Li-Ion Battery Prognostics and Health Management“, in *Batteries*, DOI: 10.3390/batteries9060332, 2023

[70] M. Kurucan, M. Özbaltan, Z. Yetgin und A. Alkaya, „Applications of artificial neural network based battery management systems: A literature review“, in *Renewable and Sustainable Energy Reviews*, DOI: 10.1016/j.rser.2023.114262, 2024

[71] R. Guo und W. Shen, „A data-model fusion method for online state of power estimation of lithium-ion batteries at high discharge rate in electric vehicles“, in *Energy*, DOI: 10.1016/j.energy.2022.124270, 2022

[72] G. Lee, D. Kwon und C. Lee, „A convolutional neural network model for SOH estimation of Li-ion batteries with physical interpretability“, in *Mechanical Systems and Signal Processing*, DOI: 10.1016/j.ymssp.2022.110004, 2023

[73] X. Hu, F. Feng, K. Liu, L. Zhang, J. Xie und B. Liu, „State estimation for advanced battery management: Key challenges and future trends“, in *Renewable and Sustainable Energy Reviews*, DOI: 10.1016/j.rser.2019.109334, 2019

[74] P. Venugopal und V. T., „State-of-Health Estimation of Li-ion Batteries in Electric Vehicle Using IndRNN under Variable Load Condition“, in *Energies*, DOI: 10.3390/en12224338, 2019

[75] Q. Liu, M. Zheng und P. Li, „A Review of Data-Driven SOH and RUL Estimation for Lithium-ion Batteries“: 2023 42nd Chinese Control Conference (CCC), S. 8769–8774. IEEE, 2023

[76] Q. Li und W. Xue, „A review of feature extraction toward health state estimation of lithium-ion batteries“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2025.115453, 2025

[77] Z. Xiao, B. Jiang, J. Zhu, X. Wei und H. Dai, „State of Health Estimation for Lithium-Ion Batteries Using an Explainable XGBoost Model with Parameter Optimization“, in *Batteries*, DOI: 10.3390/batteries10110394, 2024

[78] Z. Zhang, X. Si, C. Hu und Y. Lei, „Degradation data analysis and remaining useful life estimation: A review on Wiener-process-based methods“, in *European Journal of Operational Research*, DOI: 10.1016/j.ejor.2018.02.033, 2018

[79] T. Talluri, H. T. Chung und K. Shin, „Study of Battery State-of-charge Estimation with kNN Machine Learning Method“, in *IEIE Transactions on Smart Processing & Computing*, DOI: 10.5573/IEIESPC.2021.10.6.496, 2021

[80] I. D. Mienye und Y. Sun, „A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects“, in *IEEE Access*, DOI: 10.1109/access.2022.3207287, 2022

[81] E. Ipek, M. K. Eren und M. Yilmaz, „State-of-Charge Estimation of Li-ion Battery Cell using Support Vector Regression and Gradient Boosting Techniques“: 2019 International Aegean Conference on Electrical Machines and Power Electronics (ACEMP) & 2019 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)

[82] R. Li, P. Liu, K. Li und X. Zhang, „AdaBoost.Rt-LSTM Based Joint SOC and SOH Estimation Method for Retired Batteries“, in *Batteries*, DOI: 10.3390/batteries9080425, 2023

[83] Vinodkumar und N. Singh Singha, „Battery Management System Health Monitoring Using Artificial Intelligence“: 2023 International Conference on Sustainable Emerging Innovations in Engineering and Technology (ICSEIET)

[84] S. Song, X. Zhang, D. Gao, F. Jiang, Y. Wu, J. Huang, Y. Gong, B. Liu und Z. Huang, „A Hierarchical State of Charge Estimation Method for Lithium-ion Batteries via XGBoost and Kalman Filter“: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC). 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 10/11/2020 - 10/14/2020, S. 2317–2322. IEEE, 2020 - 2020

[85] S. Hussain, M. A. Ahmed und Y.-C. Kim, „Efficient Power Management Algorithm Based on Fuzzy Logic Inference for Electric Vehicles Parking Lot“, in *IEEE Access*, DOI: 10.1109/ACCESS.2019.2917297, 2019

[86] L. Qian, L. Xuan und J. Chen, „Battery SOH estimation based on decision tree and improved support vector machine regression algorithm“, in *Frontiers in Energy Research*, DOI: 10.3389/fenrg.2023.1218580, 2023

[87] T. Chen und C. Guestrin, „XGBoost“: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, S. 785–794. New York, NY, USA: ACM, 2016

[88] S. B. Ramezani, L. Cummins, B. Killen, R. Carley, A. Amirlatifi, S. Rahimi, M. Seale und L. Bian, „Scalability, Explainability and Performance of Data-Driven Algorithms in Predicting the Remaining Useful Life: A Comprehensive Review“, in *IEEE Access*, DOI: 10.1109/ACCESS.2023.3267960, 2023

[89] N. H. Anuar, T. R. Razak und N. H. Kamis, „Advancing Fuzzy Logic: A Hierarchical Fuzzy System Approach“, in *AIUB Journal of Science and Engineering (AJSE)*, DOI: 10.53799/ajse.v23i1.1022, 2024

[90] R. Tan, W. Hong, J. Tang, X. Lu, R. Ma, X. Zheng, J. Li, J. Huang und T.-Y. Zhang9090, „BatteryLife: A Comprehensive Dataset and Benchmark for Battery Life Prediction“, 2025

[91] Y. Lu „EV field datasets Tsinghua“ 2024

[92] H. Nguyen, X.-N. Bui, H.-B. Bui und D. T. Cuong, „Developing an XGBoost model to predict blast-induced peak particle velocity in an open-pit mine: a case study“, in *Acta Geophysica*, DOI: 10.1007/s11600-019-00268-4, 2019

[93] T. Chen, T. He, M. Benesty und V. Khotilovich9393, „Package 'xgboost'“, 2019

[94] Z. Lu, Z. Fei, B. Wang und F. Yang, „A feature fusion-based convolutional neural network for battery state-of-health estimation with mining of partial voltage curve“, in *Energy*, DOI: 10.1016/j.energy.2023.129690, 2024

[95] E. Chemali, P. J. Kollmeyer, M. Preindl, Y. Fahmy und A. Emadi, „A Convolutional Neural Network Approach for Estimation of Li-Ion Battery State of Health from Charge Profiles“, in *Energies*, DOI: 10.3390/en15031185, 2022

[96] S. Shen, M. Sadoughi, X. Chen, M. Hong und C. Hu, „A deep learning method for online capacity estimation of lithium-ion batteries“, in *Journal of Energy Storage*, DOI: 10.1016/j.est.2019.100817, 2019

[97] D. Zhou, Z. Li, J. Zhu, H. Zhang und L. Hou, „State of Health Monitoring and Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Temporal Convolutional Network“, in *IEEE Access*, DOI: 10.1109/ACCESS.2020.2981261, 2020

[98] C. Vidal, P. Malysz, P. Kollmeyer und A. Emadi, „Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation: State-of-the-Art“, in *IEEE Access*, DOI: 10.1109/access.2020.2980961, 2020

[99] D. Saji, P. S. Babu und K. Ilango, „SoC Estimation of Lithium Ion Battery Using Combined Coulomb Counting and Fuzzy Logic Method“: 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)

[100] C. Hu, G. Jain, P. Zhang, C. Schmidt, P. Gomadam und T. Gorka, „Data-driven method based on particle swarm optimization and k-nearest neighbor regression for estimating capacity of lithium-ion battery“, in *Applied Energy*, DOI: 10.1016/j.apenergy.2014.04.077, 2014

[101] H. Deng und G. Runger „Feature Selection via Regularized Trees“ 07.01.2012

[102] Z. Zhang, L. Li, X. Li, Y. Hu, K. Huang, B. Xue, Y. Wang und Y. Yu, „State-of-health estimation for the lithium-ion battery based on gradient boosting decision tree with autonomous selection of excellent features“, in *International Journal of Energy Research*, DOI: 10.1002/er.7292, 2022

APPENDIX I

Reasoning behind the rating of the accuracy criterion for each model-based SoH estimation method

Method	*Accuracy rating and rationale*
ECM (Thevenin)	The Thevenin model offers "Moderate" accuracy due to oversimplification not accurately depicting complex dynamics or aging effects of the battery [9]. In addition, it struggles to represent nonlinear behaviors [24].
EM (P2D)	The P2D model offers "Very High" accuracy as it closely depicts the battery's degradation processes implicitly modeling nonlinear behavior [15, 32, 74].
Empirical model	Empirical models offer low to moderate accuracy [15, 36]. However, its performance deteriorates for unseen test data, leading to "Low" overall accuracy rating. [73]
KF	KFs offer moderate to high accuracy depending on the underlying battery model [10, 43]. However, they are not capable to deal with nonlinearity resulting in the overall ranking of "Moderate". [2, 13, 31]
EKF	EKFs offer higher accuracy than KFs [50]. In addition, they are able to handle nonlinearities by utilizing linearization techniques [2, 50, 51]. However, this introduces approximation errors which can lead to inaccuracies [50]. In addition, the EKF still struggles with very high nonlinearity [51]. Overall, the accuracy is rated "Moderate".
UKF	UKFs accuracy is similar to that of EKFs [27]. However, it yields these results also under high nonlinearity as it does not use linearization techniques for approximation [2]. Therefore, its accuracy assessment is "High".

PF	PFs offer accuracies comparable to EKFs and UKFs [27]. The advantage it has on KFs and its extension is its confidence interval [13]. In addition, PFs effectively deal with nonlinear and non-gaussian scenarios [23, 50]. Overall, its rating is "High".
GPR	GPRs offer one of the highest accuracies, outperforming multiple other approaches both on test and training data [1, 16, 17, 19]. It manages nonlinearity and high-dimensionality well and has the capacity to provide probabilistic forecasts, which provide insights into uncertainty levels [29–31, 75]. Finally, GPR has a moderate tendency to overfitting [19, 66, 76]. Overall, the rating is "Very High".
WP	The WP offers good estimation accuracy due to its probabilistic structure which allows for capturing gradual trends and abrupt deviations in battery behavior [17, 54, 59]. However, the conventional WP is only suitable to model linear trends [78]. Overall, its accuracy rating is "Moderate".
FL	FL offers moderate accuracy [22, 27]. Compared to different SoH estimation methods its results are average [17]. In addition, FL can deal with nonlinearity resulting in an overall score of "Moderate" [2, 43].
SVM/SVR	SVM and SVR offer high accuracy rivaling that of ANNs and RFs [3, 19]. It is able to deal with nonlinear as well as high-dimensional data and its structural risk minimization concept reduces overfitting [13, 27, 29, 50, 56]. However, it does not have a confidence interval and its performance worsens for unseen test data [13, 31]. Overall, its rating is "High".

RVM	The accuracy of RVMs is similar to SVMs, but it also provides a confidence interval giving a probability estimate [17, 75]. It manages nonlinearities well and, compared to SVMs, it can achieve flexible control over overfitting and underfitting by adjusting parameters [14]. Overall, the rating is "High". [13]
kNN	kNN receives a "Low" rating for the accuracy criterion. It has generally low accuracy and a high tendency for overfitting. [3]
DT	DTs have low to moderate accuracy compared to other methods [1]. In addition, it manages complex nonlinear relationships well, but tends to overfit [30, 56]. Overall, it receives a "Low" rating.
RF	RFs offer high accuracy on both training and test data [1]. The model can capture complex and nonlinear relationships and has a low tendency for overfitting [24, 27]. The overall rating is "High". [3]
XGBoost	XGBoost offers very high accuracy compared to GPR or SVM [47, 77]. It can capture nonlinear relationships and performs well on test and training data with slightly less accuracy on the latter [24, 77]. Although, they use sub-sampling or column sampling reducing overfitting, on small datasets it still may occur [32]. The overall rating is "Very High".
AdaBoost	AdaBoost algorithms achieve higher accuracies and have reportedly outperformed single data-driven models such as SVR and LSTM [6, 68]. They are capable of capturing complex and nonlinear relationships [24]. In addition, the careful selection and tuning of weak learners avoid overfitting [14, 30]. Their overall rating is "High".
FFNN	FFNNs are highly accurate, outperforming most other model-based approaches [1, 27]. They manage nonlinearity very well and can capture complicated correlation in battery data [47, 50]. However, among other ANNs it is the least accurate [20, 70]. In addition, it has a tendency for overfitting [47, 50]. The overall rating is "High".

CNN	CNNs offer very high accuracy compared to other ANNs and ML approaches [1, 16, 22, 70]. They are capable of modelling nonlinearity and deal with high-dimensionality [3, 50, 75]. One of their main advantages is the ability to automatically identify complicated patterns and correlations from battery data [29]. However, they are susceptible to overfitting [3]. The overall rating is "Very High".
RNN	Similar to CNNs, RNNs offer very high accuracy and are able to model nonlinear and high-dimensional data as well as automatically identify complicated patterns and correlations from battery data [3, 22, 29, 32, 46, 50, 70]. Overall, their rating is "High", because their tendency to overfitting is higher than that of CNNs [3, 50, 70].
LSTM	LSTM yields very high accuracy on both training and test data [1, 13, 16, 20, 22]. They are able to capture nonlinear and complex relationships [24, 50]. However, they are prone to overfitting especially for sparse data and their performance on unseen test data worsens [24, 29, 50]. Overall, the accuracy rating is "High".
GRU	GRUs offer similar to RNNs and LSTMs very high accuracy [20, 22]. They effectively model nonlinearity and have a minimal risk of overfitting [50, 66]. The overall rating is "High".

APPENDIX II

Reasoning behind the rating of the computational efficiency criterion for each model-based SoH estimation method

Method	*Computational efficiency rating and rationale*
ECM (Thevenin)	Computationally, the Thevenin model offers "High" efficiency [9, 46]. It is a simple model with few parameters and therefore low computational complexity [5, 9, 32, 47]. Furthermore, it allows for real-time application [5, 14, 23]. On the downside, its training is time-consuming due to complex parameter initialization [15]. Finally, a high-performance controller is needed to periodically update the model parameters [30, 43, 73, 98].
EM (P2D)	The P2D model has "Very low" computational efficiency and requires significant memory storage, because of its complex structure consisting of multiple differential equations and parameters describing the battery characteristics [6, 30, 32, 77]. Building the model, identifying the parameters and subsequently updating the model periodically requires significant time and computational resources [9, 23, 46]. To deal with the complexity, a high performance controller is used [2]. Combined with the overall complexity of the system, this makes EMs not suitable for real-time application [24, 74].
Empirical model	Empirical models offer "High" computational efficiency due to simple equations, few parameters and fast response speed. [15, 30, 43, 46, 47]. Their simple computation and structure make them suitable for real-time SoH estimation [46, 47]. However, parameter identification is time-consuming, as they need to be determined via comprehensive aging tests [15, 73].

KF	KFs provide "High" computational efficiency, because of simple calculations, fast processing times and rapid model updating [2, 31]. In addition, KFs allow for real-time estimation [17, 38, 74]. On the downside, the development phase may be extensive as KFs require pre-validation [2, 10].
EKF	EKFs allow for real-time implementation due to their recursive nature but in resource limited systems it is limited due to its computational complexity [50, 51]. This complexity stems from an increased number of calculations compared to KFs that are used to deal with nonlinearities [13]. However, using a high performance controller, the processing time is fast [2]. On the other hand, the development phase, similar to KFs, is extensive due to required pre-validation and the need for careful tuning of parameters [2, 44]. Overall, the computational efficiency is "Moderate".
UKF	UKFs are computationally more demanding than both EKF and KF due to the propagation of multiple sigma points at each step [44, 52]. While they do not require derivates or Jacobian matrix calculation, their structural complexity and inference load are higher [13]. Using a high-performance controller, they are real-time capable with fast processing times [2]. Overall, the rating is "Low".

PF	PFs are computationally demanding methods whose efficiency largely depends on the number of particles used [13, 27]. As this number grows, the computational complexity increases significantly and worsens runtime performance and memory demands, especially in online applications [14, 50]. Although PFs can be applied in real-time settings, this requires substantial computational resources and optimized implementations [17, 50]. While PFs benefit from fast training and do not require a learning phase in the conventional sense, the inference stage is complex and resource-intensive [21]. Consequently, PFs are rated as "Low" computational efficiency.
GPR	GPR exhibits moderate model simplicity but its inherent structure leads to high computational overhead [1, 75]. Logarithmic and exponential functions, along with matrix operations contribute to increased computational cost and memory demand [29–31]. This limits the models ability to anticipate battery degradation in real-time as computation is comparably slow [1, 16, 29]. In addition, extensive training time is required to tune kernels and hyperparameters which are need to achieve high precision [1, 17]. Overall, GPR is rated as "Very Low" for computational efficiency.
WP	The WP exhibits "Moderate" computational efficiency. While the model is conceptually simple and widely adopted due to its intuitive formulation, its implementation involves mathematically intensive operations, which can be computationally demanding, particularly during inference [17, 57]. Additionally, offline parameter estimation may require complex mathematical modeling [30]. Once the model is trained, it features very few parameters that need to be adjusted, leading to efficient runtime execution and low memory demands [14].

FL	FL exhibits relatively high computational complexity due to operations involving multiplication, division and exponential functions [10, 17, 27]. Although FL avoids complex mathematical models and can directly estimate SoH without intermediate transformations, the design and parameter selection process is often trial-and-error based, which adds development overhead [9, 74]. FL systems are real-time capable, but usually require a high-performance controllers or specialized computational units, as their memory consumption and execution complexity are substantial [17, 83]. Additionally, FL models may require a large number of rules and membership functions, further increasing the memory footprint and computational burden [27, 99]. Consequently, the computational efficiency is rated as "Low".
SVM/SVR	SVM/SVR methods exhibit "Moderate" computational efficiency. This may vary depending on the selected kernel function and model parameters [29, 44]. The rating is rooted in computationally intensive training on the one hand and fast inference on the other hand [50]. Training is often time-consuming, especially for large datasets and high-dimensional feature spaces, as it involves complex operations like exponential functions and vector dot products, along with extensive cross-validation and parameter tuning [3, 17, 27, 44, 50, 56]. In addition, memory usage is generally high, but also varies depending on the number of support vectors [13]. Once trained, SVM/SVR can deliver fast and efficient inference suitable for real-time applications [17, 31, 50].

RVM	RVMs offer sparse solutions through Bayesian inference, often requiring very few correlation vectors, which simplifies the model compared to SVMs and reduces inference cost [31]. However, their training involves iterative optimization and computations such as exponential functions and vector dot products, resulting in high time and memory complexity [17, 29, 67]. Although their sparsity improves efficiency post-training, slow computation speed and long training times make them less appropriate for real-time applications compared to SVMs [31]. Overall, RVMs are rated as "Low".
kNN	Compared to methods like SVMs or RNNs, kNN has moderate structural complexity [3]. Also, it does not require a formal training phase, which eliminates upfront computational overhead [3]. Its inference step, however, relies on calculating distances between the input sample and all points in the training dataset, which can become computationally intensive as the dataset grows [56]. While predictions are executed quickly in small datasets, the method suffers from high memory consumption and increasing inference latency in high-dimensional spaces [79]. Overall, it is rated as "Moderate".
DT	DTs exhibit "Moderate" computational efficiency. While inference is typically fast and lightweight, the training process is long and computationally expensive due to the iterative nature of tree construction [56]. Furthermore, DTs often require extensive experimentation and parameter tuning, such as adjusting maximum depth or employing pruning strategies to control model complexity and overfitting [30]. Despite this, the simple structure of DTs, combined with pruning techniques to manage complexity, ensures that the method remains computationally feasible.

RF	RFs exhibit "Moderate" computational efficiency. Training involves building multiple decision trees, which can be time-consuming and computationally intensive, depending on the dataset size and number of estimators [30]. While the model structure is relatively simple, hyperparameter tuning, is necessary and increases computational burden [1, 30]. In addition, RFs tend to require more memory than SVM or kNN and less than RNN [3]. Lastly, RFs benefit from not requiring preprocessing, as raw data can be directly fed into the model, reducing overall computational overhead [3, 32].
XGBoost	XGBoost exhibits "Moderate" computational efficiency. It outperforms traditional ML models such as SVM and SVR in terms of inference speed, making it highly suitable for real-time prediction tasks [56, 80, 81]. Its architecture supports parallel computation and column sampling, in which a random subset of features is selected during each boosting round, enhancing training speed and scalability [32]. However, training times remain relatively long resource-intensive due to its higher model complexity compared to methods like RFs [68].
AdaBoost	AdaBoost exhibits "Moderate" computational efficiency. The algorithm is relatively simple to implement, benefiting from a sequential ensemble structure that builds weak learners iteratively [80]. It delivers very fast inference, making it suitable for deployment in real-time systems once trained [68]. However, the training process is typically long and computationally intensive, leading to increased runtime and memory demands [68]. Additionally, its model complexity tends to be higher than RF, particularly as the number of weak learners increases [68].

FFNN	FFNNs exhibit comparatively low structural complexity, characterized by a simple architecture with fewer parameters compared to RNNs [1, 72]. Their inference phase is computationally efficient and fast, making them suitable for real-time applications [1, 20, 50, 70, 82]. However, training can be computationally intensive and slow due to the need of significant amounts of high-quality data for reliable parameter estimation [50, 70]. The overall computational efficiency is rated "Moderate".
CNN	CNNs exhibit a complex and highly variable structure, requiring extensive parameter adjustments and the use of very large datasets for training, generally larger than for RNNs and FFNNs [14, 16, 70]. Their training process is computationally intensive, consuming significantly more time than models such as GPR, SVR and LSTM [1]. Furthermore, the number of parameters is considerably higher than in simpler architectures like FFNNs, resulting in high memory usage and longer processing times [1]. While real-time inference is possible on high-end hardware, the overall computational demands during model development and training make CNNs impractical for real-time applications [38]. Therefore, CNNs are rated as "Very Low".
RNN	RNNs demonstrate very low computational efficiency, driven by their structural complexity, significant computational resource demands and substantial memory usage [3, 38, 46, 70]. Compared to simpler methods like RF, SVM or kNN, RNNs require significantly longer training time [3]. Although they typically demand slightly smaller datasets than CNNs, their reliance on high-performance computing platforms remains a bottleneck [38]. Therefore, RNNs are rated as having "Very Low" computational efficiency.

LSTM	LSTMs are among the most computationally demanding ANNs. They exhibit a complex architecture and extensive training time [1, 22, 75]. The training efficiency is affected by the length of the input sequences and the size of hidden layers, which also drives up memory usage [13, 44]. While LSTMs can operate in real-time with significant computational resources, their inference latency and power consumption are typically too high for such applications [44]. Among similar models, e.g., GRUs and FFNNs, they show the highest computing time [20]. Therefore, LSTMs are rated "Very Low".
GRU	GRUs simplify the structure of LSTMs by reducing the number of gates, which improves computational efficiency while preserving temporal modeling capabilities [26]. Although GRUs remain computationally intensive, they are faster and more efficient than LSTMs, both in terms of training time and inference latency [22, 26]. GRUs are capable of real-time application on higher-end systems, but may struggle on constrained embedded platforms [22]. As a result, GRUs are assigned a "Low" rating.

APPENDIX III

Reasoning behind the rating of the interpretability criterion for each model-based SoH estimation method

Method	*Interpretability rating and rationale*
ECM (Thevenin)	The Thevenin model is simple and easy to implement [24]. This structure based on few parameters and equations leads to "High" interpretability. The connection to degradation mechanism adds additional insights into the inner workings of the battery [23, 32]. Only the parameters themselves require expert knowledge to be initialized and periodically updated [23].
EM (P2D)	The interpretability of the P2D model is rated as "Low". Although the model offers deep physical insight by simulating electrochemical processes through coupled partial differential equations, its internal decision logic is not easily accessible [2, 14, 24, 30]. The complexity of its mathematical formulation makes it difficult for experts to trace how specific inputs lead to a particular SoH estimate, resulting in reduced transparency.
Empirical model	The interpretability of empirical models is "High", because the formulas are simple and their implications can be read-off [43]. Expertise is required in understanding the parameters of the model as these require extensive modelling and testing to allow for improved model accuracy [15, 73].

KF and EKF	Both algorithms are highly interpretable, because each step of the computations can be traced making the logic transparent. Experts can tune noise covariances and examine the state estimates. The only expertise is required when implementing the equations which are based on detailed understating of the battery parameters and the dynamic models involved in charging and discharging [44]. Therefore, it is rated "High".
UKF	Compared to KF and EKF; UKF is less interpretable and, therefore, rated "Moderate". While the interpretable Kalman structure is maintained, the inner workings are less intuitive due to the use of sigma points and weightings [52].
PF	PFs operate via a set of weighted particles, which makes them essentially black-box estimators [23, 31, 32]. Based on that, it is difficult to interpret how the filter arrives at a given estimate. Thus, interpretability is "Low".
GPR	GPR is somewhat interpretable because the kernel function and the learned weights can be inspected. In addition, explicit uncertainty is provided and one can examine which past data points influence the estimate [1]. However, the fitted function does not have a simple analytic form and the predictions come from complex covariance computations. Overall, GPR rate "Low" on the interpretability scale.

WP	The interpretability of the WP is "Moderate". On the one hand, it offers a mathematically transparent structure based on a well-known stochastic differential equation [58]. In addition, its core parameters can be conceptually linked to average degradation trends, which makes the model's input-output relationships explainable and its internal logic traceable to informed users [58]. However, the stochastic nature of the model introduces probabilistic outputs rather than deterministic predictions, which may challenge practical interpretability [59].
FL	FL is rated "Very High". Their core structure is based on *if-then* rules that use linguistic variables and fuzzy sets, making the reasoning process explicit and transparent [7]. This design allows expert users to understand, trace and manually adjust the logic behind the system's decisions without needing to analyze complex mathematical models. The rule-based structure clearly maps input conditions to output decisions, offering a direct and intuitive understanding of how SoH is estimated.
SVM/SVR	SVM/SVR are often considered black-box models, as their decision boundaries result from complex combinations of support vectors [24, 50]. This opaque internal structure limits transparency and makes it difficult for experts to explain the model's reasoning [23]. As a result, they are rated "Low".
RVM	RVM provides sparser solutions than SVM/SVR by relying on fewer relevance vectors, which simplifies the decision function and improves transparency [29]. This sparsity enhances the model's interpretability, allowing expert users to better understand the input-output relationships. Therefore, it is rated "Moderate".

kNN	The kNN algorithm provides a "High" level of interpretability. Its decision-making process is relatively transparent as predictions are made by averaging the outcomes of the k most similar instances in the training data [79]. In addition, experts can inspect which data points were most influential in a particular output. However, as the number of neighbors increases and the dimensionality of the data grows, the interpretability diminishes.
DT	DTs exhibit "Very High" interpretability due to their transparent, rule-based structure [30, 56, 67]. Each decision path from root to leaf node represents a traceable and understandable logic that links input features to output predictions. The hierarchical splitting of data allows experts to follow the decision-making process step-by-step. Individual trees are inherently easy to visualize and interpret, enabling users to understand how specific predictions are made without requiring advanced mathematical knowledge [68].
RF	RFs exhibit "Low" interpretability. While they are not fully transparent, their structure, based on an ensemble of DTs, provides insight into the model's decision logic offering more interpretability than complex ANNs or GPR [1, 65]. However, the aggregation of numerous trees reduces this intuitive understanding, making the model appear as a "black box" [24, 47, 68].
XGBoost	XGBoost exhibits "Low" interpretability. While its base learners, typically DTs, are individually interpretable, the overall model becomes opaque due to the aggregation of a large number of base models [24, 47, 68]. Although the algorithm provides feature importance scores, which can support feature selection and offer some limited insight into input relevance, this does not compensate for the lack of transparent logic in its overall structure [80].

AdaBoost	AdaBoost is characterized by "Low" interpretability. Like XGBoost, it builds upon base learners like DTs, that are interpretable in isolation. However, the ensemble structure combines these weak learners through weighted aggregation over multiple iterations, resulting in a model whose logic is difficult to follow [24, 47, 68].
FFNN	FFNNs are considered black-box models, as their internal decision-making processes are generally opaque and difficult to interpret [32, 50]. However, their structure is comparatively simpler than RNNs or CNNs. As a result, they are rated as having "Low" interpretability.
CNN	CNNs function as black-box models, offering very limited interpretability due to their highly abstract internal operations and parameter-heavy architecture [32, 47, 50]. In addition, the prediction process is opaque, making it hard to diagnose errors or justify outputs [16]. As a result, CNNs are assigned a "Very Low" rating for interpretability.
RNN, LSTM and GRU	RNNs, LSTMs and GRUs are all black-box models [24, 50]. RNNs contain hidden states that evolve over time [70]. This makes tracing the decision logic difficult, especially for long sequences. LSTMs add gates (input, forget, output) and memory cells, increasing complexity and making interpretation even harder [38, 70]. The internal state transitions are abstract and nonlinear. Like LSTMs, GRUs manage memory with gating mechanisms, though they are simpler than LSTMs [26]. Still, they remain black-box models with little interpretability [50].

APPENDIX IV

Reasoning behind the rating of the data requirements criterion for each model-based SoH estimation method

Method	*Data requirements rating and rationale*
ECM (Thevenin)	The Thevenin model has "Moderate" data requirements due to defining the model's parameters and subsequent recalibration. [9, 36]
EM (P2D)	To build the P2D model and calibrate it, expert knowledge and extensive experimental data is required [23, 50]. This results from the difficulty of identifying several model parameters and the necessity of accurate parametrization [15, 74]. Conversely, data requirements are "Very High".
Empirical model	Empirical models perform at acceptable accuracy only with large and high-quality data as input [5]. A sizable number of tests has to be conducted under specific decisions to derive the required parameters [15]. Therefore, the data requirements criterion is rated "High".
KF, EKF and UKF	KF, EKF and UKF do not require any initial data for training their algorithm [43]. However, their accuracy depends on the underlying battery model which have to be constructed, requiring extensive experimental testing and datasets (see ECM and EM) [44, 50, 74]. Therefore, when implementing these methods, this additional effort of model development has to be considered. The rating for the data requirements criterion is "High".

PF	PFs have "High" data requirements. They require a large volume of high-quality experimental data to estimate system states effectively and capture system dynamics under various conditions [23, 74]. Compared to KFs, EKFs and UKFs, PFs do not rely heavily on explicit models, but they still require appropriate models or fitness functions [50].
GPR	GPRs do not require large datasets and are suitable for small sample sizes [75]. However, they rely heavily on the quality of the input data and the selected features [24, 56]. Model performance is highly sensitive to kernel choice and hyperparameter settings, requiring careful calibration or automated optimization [56, 75]. Overall, the data requirements are rated as "Moderate" due to the low data volume but high preparation complexity.
WP	The WP requires some training data for offline parameter computation. Depending on the application, it may also involve complex mathematical modeling and the use of degradation data to fit the underlying stochastic process [17, 30]. However, compared to data-driven models, the overall volume of required data is typically lower and the model structure does not depend on high-dimensional input or extensive labeling. Therefore, data requirements are rated as "Low."
FL	FL exhibits "High" data requirements. Its effectiveness and modeling accuracy depend heavily on the quality, precision and diversity of the input data used [2, 7, 10, 83]. Additionally, it requires domain expertise to design the rule base [7]. While preprocessing steps are generally moderate, the manual effort involved in rule formulation and validation further increases the data burden.

SVM/SVR	SVM/SVR models are sensitive to the amount, quality and diversity of the training data, as prediction accuracy heavily depends on well-prepared datasets [2, 24, 74]. While they can perform well with finite datasets, optimal performance typically requires a large volume of high-quality training data [11, 50]. Additionally, careful kernel selection, hyperparameter tuning and manual feature engineering is crucial [16, 30, 31, 50]. Overall, the data requirements of SVM/SVR are rated as "High".
RVM	Compared to SVMs, RVM reduces reliance on kernel and penalty factor selection and automated operation is more feasible, reducing manual configuration effort [13, 14]. However, effective initialization still requires access to sufficient historical data [75]. Therefore, the data requirements of RVM are rated as "Moderate".
kNN	The data requirements of kNN are rated "High", because its performance depends heavily on the availability of high-quality, representative datasets [56]. The absence of such data can significantly degrade the models' accuracy and generalization ability [100]. The size of the dataset is not as big as that of RNN or RF but exceeds that of SVM [3]. In addition, kNN requires manual feature extraction to construct effective input representations [3].
DT	DTs have "High" data requirements. As the model is sensitive to initial parameter selection, it often requires extensive experimentation data for appropriate tuning [56]. In addition, DTs tend to overfit on limited datasets [56]. On the other hand, their ability to manage both numerical and categorical input features reduces the need for extensive manual feature engineering [101].

RF	RFs exhibit "Moderate" data requirements. For optimal performance they require a larger dataset than SVM or kNN but smaller than that of more complex models like RNNs [3]. A key advantage of RFs is their low dependence on manual feature extraction and preprocessing, as it can directly use raw sensor data and automatically determine feature importance [3, 14, 32]. Nevertheless, careful tuning of hyperparameters is necessary to optimize performance and prediction accuracy strongly depends on the quality and diversity of the input data [24, 30].
XGBoost	XGBoost demonstrates "High" data requirements due to its reliance on large, diverse and high-quality datasets [24, 32, 84]. While XGBoost requires minimal manual feature engineering, it involves a complex set of hyperparameters that must be carefully tuned across a wide range [24, 32, 84]. Techniques like histogram-based approximation improve the efficiency of feature selection and splitting, but the overall prediction accuracy remains strongly influenced by the quality of the input data [68].
AdaBoost	AdaBoost has "Moderate" data requirements. Its prediction accuracy depends heavily on the quality of the input dataset, but it is easy to implement and requires little hyperparameter tuning [14, 24, 80]. Although it benefits from good-quality labeled data, it is less demanding than more complex models like XGBoost in terms of data diversity and volume.

FFNN	FFNNs require a large volume of diverse and high-quality training data to achieve reliable SoH estimations [16, 22, 27, 50, 74]. Their performance depends strongly on the completeness and stability of input data, including relevant features like temperature and impedance [30, 38]. Hyperparameter tuning is necessary to optimize model accuracy, which adds to the data preparation complexity [47]. These factors collectively lead to a "Very High" rating for data requirements.
CNN	CNNs demand "Very High" data requirements, because they rely on large, labeled datasets to learn complex representations [3, 22, 38, 50]. In addition, the model performs automatic feature extraction from raw inputs, eliminating manual feature engineering but significantly increasing the need for diverse and high-quality data [3, 16].
RNN, LSTM and GRU	RNNs, LSTMs and GRUs exhibit "Very High" data requirements. These models rely on large datasets that capture the temporal dynamics of battery aging, requiring high-resolution, long-term sequences with accurate SoH labels [3, 22, 50]. Their performance heavily depends on the availability of reliable, diverse data to generalize across various degradation patterns and avoid overfitting [29, 38]. In addition, manual feature extraction is needed [3].

APPENDIX V

Reasoning behind the rating of the reliability criterion for each model-based SoH estimation method

Method	*Reliability rating and rationale*
ECM (Thevenin)	The Thevenin model is suitable for simulating battery behavior under various load conditions, however, its best performance usually is under steady-state conditions [9, 24]. Furthermore, as all ECMs, the model struggles in high-current and low temperature environments [24, 32]. To deal with noise, a Kalman filter is commonly employed [46]. Overall, the reliability of ECM models, similar to accuracy increases with model complexity [7]. The Thevenin model is rather simple and therefore offers "Moderate" reliability.
EM (P2D)	The reliability of the P2D model is "Moderate". It is applicable and robust under extreme conditions, however, only while having complete data that covers multiple operating conditions [9, 32].
Empirical model	Empirical models are limited to the conditions similar to the experimental data [6]. It lacks robustness under changing conditions, leading to decreasing performance when the environment changes [15, 36, 43, 46, 47, 69, 73]. The overall reliability is "Very Low".
KF	KFs are a self-adaptive filtering method that can handle noise inference in the signal and estimate the battery SoH based on incomplete and noisy data [86]. However, its performance is limited by the accuracy of the underlying model [50]. In addition, operating conditions, such as time-varying current and ambient temperature can influence the accuracy of the algorithm [54]. Overall, it receives a "High" reliability rating.

EKF	EKFs exhibit high robustness towards noise and are capable to deal with data sparsity [13, 51]. In addition, it is able to adapt to changing operating conditions due to its recursive nature. However, similar to the KF; it is limited by the underlying battery model [50]. In addition, in highly nonlinear systems the unprecise prediction of the noise covariance matrix of status and observation introduces a cumulative error worsening the models' accuracy [7]. Overall, the rating is "High".
UKF	UKFs are less sensitive to noise than both EKF and KF [27]. They offer high robustness and adaptability to changing battery conditions and uncertainty due to their model structure [22, 50]. Overall their reliability rating is "Very High", but some operating conditions under varying current or ambient temperature can still influence the algorithm [54].
PF	PFs exhibit a "High" level of model reliability. Their probabilistic framework allows them to effectively handle uncertainty and manage multimodal distributions [50]. They also adapt well to data sparsity and changing battery conditions, showing high flexibility and applicability in different scenarios [13, 23, 31]. However, their robustness is compromised by a sensitivity to noise and model complexity, as well as susceptibility to particle degradation during operation, which can increase estimation errors [23, 32].
GPR	GPR provides robust uncertainty quantification by modeling both mean predictions and predictive variances through a posterior distribution [56]. This allows it to effectively capture complex, nonlinear relationships and adapt to varying operational conditions [56]. Its inherent uncertainty estimation makes GPR resilient to noisy or sparse data, offering confidence intervals alongside predictions [45]. These features contribute to GPR's "Very High" reliability.

WP	WPs are suited to represent stochastic degradation phenomena and explicitly model uncertainties in system behavior [60]. This enables them to remain robust in the presence of measurement noise, operational variability and incomplete data [60]. As such, the reliability of the WP is rated "High."
FL	Fuzzy Logic exhibits a "High" level of reliability due to its robustness in managing uncertainty, imprecision and noise in input data [27]. In addition, it offers high adaptability resulting in more consistent performances under fluctuating conditions compared to methods such as SVM or FFNN [22, 83].
SVM/SVR	SVM/SVR methods demonstrate high reliability due to their inherent robustness to noise and strong generalization capabilities across a wide range of operating conditions [2, 13, 23, 27, 50]. They are effective with small datasets and maintain accurate predictions even under complex or fluctuating driving scenarios [14, 56]. However, reliability is somewhat constrained by sensitivity to kernel and hyperparameter choices, as incorrect configurations may reduce model adaptability and tolerance to noisy or incomplete data [3, 44, 56]. Although not entirely immune to data sparsity or outliers, SVM/SVRs remain generally resilient and stable in diverse real-world applications [13, 56, 75]. Therefore, their reliability is rated as "High."

RVM	RVMs offer high reliability through probabilistic modeling and a strong inherent robustness to noise and small data fluctuations [2, 13]. They outperform SVMs in adaptability and noise resistance, benefiting from their ability to discard irrelevant data and mitigate the impact of outliers [14, 75]. However, their reliability is limited when the training data are overly sparse, leading to reduced stability and poor repeatability [14, 43]. Despite this, RVMs' reduce dependence on strict kernel conditions and their automatic operation contribute to consistent performance across varied inputs [25]. Overall, RVMs are rated as "High".
kNN	The reliability of the kNN method is rated as "Low." While kNN can deliver accurate results under well-controlled conditions, it is sensitive to noise and outliers [3, 56]. In addition, kNN cannot extrapolate beyond training data, which limits its reliability in situations involving unseen operational conditions or battery degradation states [38]. As a result, extensive data preprocessing and careful feature selection are often necessary to mitigate these weaknesses [56].
DT	DTs exhibit "Moderate" reliability. They are generally stable under standard operating conditions and can adapt to varying battery conditions [56]. However, DTs are sensitive to noise and outliers as well as the initial parameters selection [56]. In addition, they lack extrapolation capabilities, meaning they cannot reliably predict outcomes for input values that fall outside the range of the training data [65].

RF	RFs demonstrate "High" reliability. First, they exhibit strong robustness to outliers and noise, allowing raw sensor data to be fed directly into the trained model without preprocessing [3, 11, 30, 47]. This enhances their resilience when handling heterogeneous or incomplete datasets [68]. Moreover, RFs maintain their estimation accuracy despite changes in charging and discharging protocols, supporting adaptability across diverse battery usage scenarios [68]. However, they share the common limitation of many data-driven models: an inability to extrapolate beyond training data [65].
XGBoost	XGBoost exhibits strong generalization ability and robustness, allowing it to perform reliably across multiple battery operating conditions [32, 47, 56]. It can also handle missing values effectively enhancing its resilience to incomplete data [80]. Although, XGBoost is sensitive to outliers significantly impacting the models prediction results, its overall robustness is comparable to that of SVMs [32, 56, 102]. Overall, XGBoost is rated "High" in reliability.
AdaBoost	AdaBoost demonstrates strong robustness compared to simpler models like DTs [6, 47]. However, it is sensitive to noisy data and outliers because its iterative learning approach can amplify errors from such irregularities [14, 80]. Therefore, its reliability is rated "Moderate".
FFNN	FFNNs are capable of handling diverse input types and exhibit adaptability to different operating conditions and battery aging effects [22, 27, 50]. In addition, their generalization ability allows them to perform well even with incomplete or noisy data [50]. However, high ambient temperatures can degrade their estimation accuracy and small amounts of data reduce the models' adaptability [14, 20]. Overall, the reliability rating is "High".

CNN	CNNs demonstrate adaptability across different applications and operational conditions, including changes in battery state and aging [50]. They have better generalization capabilities than FFNNs and are less sensitive to noise compared to LSTM and GRU architectures [22, 38]. CNNs effectively handle incomplete and noisy data due to their robust feature extraction capabilities developed during training [50]. Overall, CNN offer "Very High" reliability.
RNN	RNNs are highly adaptable and well-suited for modeling battery performance across a wide range of operational conditions, including variations in usage patterns and aging effects [22, 50]. They can handle noisy and incomplete data effectively due to strong generalization, which is higher than that of FFNN [38, 50]. Therefore, RNNs are rated "Very High" on the reliability scale.
LSTM and GRU	LSTM and GRU networks are adaptable to changing operating conditions and can handle incomplete and noisy input data [50]. However, both networks are more susceptible to noise than FFNNs and low-temperature conditions can increase error rates [20]. Overall, their reliability rating is "High".

APPENDIX VI

Reasoning behind the rating of the scalability criterion for each model-based SoH estimation method

Method	*Scalability rating and rationale*
ECM (Thevenin)	The Thevenin model simulates one battery cell. To apply the model to modules or packs a redesign of the model is required increasing overall complexity. The battery pack has to be considered as a whole cell, which leads to difficulties in model parameter identification. Therefore, ECM models cannot provide highly accurate quantitative results at pack level. The overall rating is "Low". [36]
EM (P2D)	P2D models have "Very Low" scalability. Each battery cell ages differently and would require a complex EM. As one EM is already computationally intensive, having multiple cells within a battery pack multiplies said complexity making its use impractical. [13]
Empirical model	The scalability of empirical models is "Low". As every cell behaves differently, one model for each cell within a battery pack or module is necessary. On module level, some models exist that describe the charging and discharging responses at different SoH. However, empirical model for cell ageing in terms of charging cycles is not sufficient to describe the inconsistent degradation inside the battery modules. At pack level, the empirical model method is not suitable for SOH estimation. [36]

KF and EKF	KF and EKF demonstrate "Very High" scalability. Their computational complexity scales linearly with dataset size, allowing for efficient operation in larger battery systems [13]. In addition, both filters can accommodate multivariate inputs due to their use of state vectors and covariance matrices [44].
UKF	The UKF achieves "Moderate" scalability, primarily limited by its computational structure leading to higher memory usage and longer inference times [44, 50]. While it can still manage multivariate inputs and operate in parallel architectures, its resource requirements scale less efficiently than KF and EKF [44].
PF	PFs exhibit "Low" scalability due to their inherent computational demands that grow significantly with system complexity [50]. As dimensionality increases, the number of required particles grows exponentially, which restricts PFs from scaling efficiently [50]. Overall, PFs are rated "Low".
GPR	GPR is highly capable of handling large datasets [1]. However, its computational cost typically scales cubically with dataset size, resulting in high memory demands and longer processing times for very large datasets [29, 56]. Sparse representations and approximation techniques can partially reduce this burden but do not fully eliminate scalability challenges [56]. Overall, GPR's scalability is rated as "Moderate".
WP	The WP demonstrates "Moderate" scalability. On the one hand, its mathematical structure remains relatively lightweight. However, extending the WP to larger battery systems or higher-dimensional input spaces increase computational demands, as parameter identification becomes more complex [88].

FL	Fl demonstrates "Low" scalability when applied to complex systems with high input dimensionality. As the number of input variables increases, the system requires exponentially more rules, leading to a rise in parameters and data volume needed to define them, resulting in extensive design and tuning efforts. [89]
SVM/SVR	SVMs and SVRs are known for their strong generalization capabilities, enabling effective performance with new batteries without extensive retraining [13]. However, their scalability is significantly limited by computational complexity, especially when managing large datasets or high-dimensional feature spaces (scaling cubically with dataset size) [13, 22, 50, 75]. Overall, the scalability is rated as "Moderate."
RVM	RVMs exhibit similarly strong generalization to SVMs and can adapt to new data or batteries without substantial retraining [13]. However, their scalability is further limited compared to SVMs due to longer training times and higher computational demands when working with large datasets or complex input structures [29]. Hence, the scalability is rated as "Low."
kNN	kNN exhibits "Low" scalability due to its inherent reliance on storing and comparing all training samples during inference [38]. As the dataset grows, both memory usage and computational cost increase significantly, since distance calculations must be performed for each query against the entire dataset [3, 56]. This issue worsens in high-dimensional feature spaces [3].
DT	DTs exhibit "Moderate" scalability. While they can handle large datasets and multivariate inputs, their performance and efficiency degrade as the data volume or input dimensionality increases, leading to longer training times and increased memory usage. [56]

RF	RFs demonstrate "High" scalability due to their flexibility in handling large-scale and high-dimensional datasets [1, 3]. Unlike NNs and SVR, RFs require fewer optimization parameters, making it scale more efficiently [69].
XGBoost	XGBoost is rated "Very High" in scalability, as it is one of the key factors behind its widespread success [80]. The algorithm incorporates several system-level and algorithmic optimizations that enable efficient learning on large and complex datasets. Additionally, XGBoost supports parallel and distributed computing, significantly accelerating training times and enabling rapid model exploration even as dataset sizes grow. [87]
AdaBoost	While AdaBoost is capable of handling larger datasets, its sequential model-building process makes it inherently slower than more optimized algorithms like XGBoost, especially as dataset size grows, which limits its scalability [68]. Therefore, AdaBoost is rated "Moderate".
FFNN	FFNNs demonstrate "High" scalability. They can manage a wide range of input features and system configurations with reasonable accuracy [50]. Compared to CNNs, their flexibility in model updating and adaptation to big data environments is slightly lower [1]. Nevertheless, FFNNs maintain stable performance when applied to large datasets and multivariate inputs [50].
CNN	CNNs exhibit "Very High" scalability, due to their ability to efficiently process high-dimensional inputs and large-scale datasets [1, 3, 16]. CNNs benefit from automatic feature extraction mechanisms, allowing them to adapt without manual reconfiguration as data complexity increases. However, as CNNs scale, the number of model parameters also grows, resulting in increased computational and memory requirements [26].

RNN, LSTM and GRU	RNNs, LSTMs and GRUs are all capable of capturing temporal dependencies, making them suitable for applications involving sequential inputs [26]. These architectures scale to larger datasets and complex system configurations, although the increase in model parameters imposes greater computational and memory demands [13, 26]. LSTMs offer improved scalability over basic RNNs by mitigating issues like vanishing gradients in moderate-scale settings [16]. GRUs on the other hand, present a lighter alternative with similar generalization capabilities, but, like LSTMs, their performance stability depends on resource availability and careful tuning [13, 26]. Overall, they are rated "High" in scalability.

APPENDIX VII (SDG ALIGNMENT)

This thesis contributes to the UN's 2030 Agenda for Sustainable Development by providing a tool to enhance the practical deployment of LiBs in BEVs. The developed multi-criteria evaluation framework for model-based SoH estimation methods addresses key limitations in existing BEV battery diagnostics, offering a systematic approach to assess, compare and select suitable methods for real-world applications.

The framework extends beyond conventional evaluations, by incorporating six practically relevant criteria (accuracy, computational efficiency, interpretability, data requirements, reliability and scalability) and being validated using a large, real-world dataset. This enables context-specific recommendations for method selection, bridging the gap between academic research and practical deployment in electric mobility.

Specific Contributions to the SDGs

SDG 7 – Affordable and Clean Energy

The framework supports SDG 7 by improving both the reliability and energy efficiency of LiBs in BEVs. Accurate SoH estimation enables failure anticipation, predictive maintenance and battery replacement planning, ensuring batteries operate closer to their optimal performance throughout their lifecycle. This reduces energy losses, prolongs battery service life and contributes to more efficient and sustainable electric mobility, thereby supporting cleaner energy use in the transport sector.

SDG 12 – Responsible Consumption and Production

The research aligns with SDG 12 by promoting the responsible use of critical raw materials in BEV batteries, including lithium, cobalt and nickel. By reducing premature battery replacements and enabling second-life applications within the vehicle sector,

the framework supports circular economy practices and mitigates environmental impacts associated with battery production, consumption and disposal.

SDG 13 – Climate Action

This thesis contributes to SDG 13 by facilitating the broader adoption of BEVs through improved battery health monitoring. Reliable and longer-lasting batteries reduce reliance on fossil fuel-based transportation, supporting the decarbonization of mobility systems and contributing to global efforts to mitigate climate change.

Conclusion

Through the development, validation and application of this multi-criteria evaluation framework, the thesis provides a practical tool for improving the deployment of LiBs in BEVs. By enhancing energy reliability, promoting responsible resource use and supporting the transition to low-carbon transport, the research directly advances the objectives of the UN SGDs.