Machine Learning and Cloud Technologies for Control and State Estimation in Battery Management Systems within Reconfigurable Batteries

Dominic Karnehm

Vollständiger Abdruck der von der Fakultät für Elektrische Energiesysteme und Informationstechnik (EIT) der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktors rerum naturalium (Dr. rer. nat.)

angenommenen Dissertation.

Gutachter/Gutachterin:

1. Prof. Dr. rer. nat. Antje Neve
2. Univ.-Prof. Dr.-Ing. Thomas Weyh

Die Dissertation wurde am 25. März 2025 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Elektrische Energiesysteme und Informationstechnik (EIT) am 26. Juli 2025 angenommen. Die mündliche Prüfung fand am 29. Juli 2025 statt.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Antje Neve, for her unwavering support, guidance, supervision, and patience throughout my doctoral journey. Her positive approach, even when targets seemed difficult to achieve, always motivated me to persevere.

I would also like to extend my sincere thanks to my colleagues, Sebastian Pohlmann and Andreas Wiedenmann. Furthermore, I am deeply grateful to Prof. Thomas Weyh and the entire EIT 1.3 team for their support and the enjoyable moments we shared. Your encouragement and camaraderie made my Ph.D. journey all the more fulfilling. Special thanks to Manuel Kuder and Michael Hohenegger from Pulsetrain for introducing me to the theory and application of reconfigurable batteries and for continuously expanding my knowledge in this area.

A heartfelt thank you goes to Prof. Sheldon Williamson, Akash Samanta, Latha Anekal, Chandan Chetri, and all the members of the Ontario Tech STEER group. I will always cherish my research stay in Oshawa and the incredible experiences we shared—both there and wherever our paths crossed around the world.

To my parents, thank you for always believing in me. Your unwavering support has been invaluable. I am also especially grateful to Zahraa Khais and all those who have encouraged and supported me throughout this challenging journey.

Finally, I would like to acknowledge the funding support provided by the dtec.bw MORE project. Special thanks to Thorsten Lüttel, whose steadfast support as a subproject manager played a crucial role in making this research possible.

Holzkirchen, March 2025
Dominic Karnehm

– *Learning by doing*
– *Paddle your own canoe*
– *Look at the boy/girl*

Scout educational principles
adapted from Robert S. S. Baden-Powell, 1st Baron Baden-Powell

Zusammenfassung

Eine präzise und zuverlässige Zustandsbestimmung für Lithium-Ionen-Batterien (LIBs) ist entscheidend für die Sicherheit von Batteriespeichersystemen und damit verbunden die Verbreitung der Elektromobilität. Allerdings mangelt es bestehenden Batteriemanagementsystemen (BMS) an Flexibilität über den gesamten Lebenszyklus eines Batteriepacks. Intelligente, cloudbasierte BMS und rekonfigurierbare Batteriesysteme (RBS) ermöglichen eine gezielte Nutzung einzelner Batteriezellen und verbessern dadurch die Gesamtleistung des Batteriesystems. Darüber hinaus bieten digitale Zwillinge in BMS-Architekturen vorausschauende Wartungsstrategien sowie adaptive Methoden zur Zustandsbestimmung. Die Nutzung verteilter Rechenressourcen ermöglicht eine effiziente Implementierung cloudbasierter BMS und digitaler Zwillinge. Ein entscheidender Parameter für ein solches Systems ist die Round-Trip-Latenz, einschließlich der Berechnungskosten. In dieser Arbeit werden moderne Methoden zur Ladezustandsbestimmung (SOC) diesbezüglich verglichen. Für ein Feedforward-Neuronales-Netzwerk (FNN) wurde bei einer Distanz von 450 km über Long-Term-Evolution (LTE) als Mobilfunknetz eine mittlere Round-Trip-Latenz von 54.6 ms gemessen. Dieser Wert liegt deutlich innerhalb der typischen Grenze von 100 ms. Temperatur hat einen erheblichen Einfluss auf die Leistung und Alterung von LIBs. Überhitzung beschleunigt die Degradation und kann zu einem thermischen Durchgehen führen. Eine präzise Temperaturabschätzung ist daher essenziell. Zu diesem Zweck werden in dieser Arbeit ein Kolmogorov-Arnold-Netzwerk (KAN) sowie ein interconnected Long Short-Term Memory (LSTM) Modell zur Schätzung der Kern- und Oberflächentemperatur einer LIB vorgeschlagen. Mit dem KAN-Modell eine mittlere quadratische Fehlerabweichung (RMSE) von 0.735 °C und 0.520 °C für Kerntemperatur und Oberflächentemperatur während eines UDDS-Fahrzyklus bei 25 °C erzielt werden. Neben der präzisen Zustandsbestimmung ist eine robuste Softwarearchitektur erforderlich, um einen bidirektionalen Datenaustausch zwischen Fahrzeugen und Cloud zu ermöglichen. Diese Arbeit schlägt ein fünfstufiges Internet-of-Things (IoT) Framework sowie eine universelle Datenspezifikation für Batteriespeichersysteme vor. Im Zusammenhang mit der zentralen Rolle der effizienten Datenkomprimierung wird eine verlustbehaftete, adaptive Datenkompressionstechnik auf Basis der Polynomkodierung vorgestellt. Dabei konnte während eines UDDS-Fahrzyklus ein RMSE von 5.09 mV bei 99.1 % Kompression erreicht werden. Die Gesamtleistung von cloudbasierten BMS kann durch ein intelligentes RBS weiter optimiert werden. In dieser Arbeit werden verstärkungslernbasierte Steuerungsalgorithmen für diesen Zweck entwickelt. Für das Training dieser Algorithmen wurde ein elektrophysikalisches Simulationsmodell als Trainingsumgebung konzipiert. Dieses Modell wurde mithilfe des Just-in-Time-Compilers Numba für Python implementiert und ermöglicht die Simulation einer Sekunde Betriebszeit eines zehn Module Battery Modular Multilevel Management (BM3)-Systems in 0.33 s. Eine Reduktion um den Faktor 130 im Vergleich zu einer MATLAB/SIMULINK-Implementierung. Die zwei maschinelle Lernalgorithmen, Q-Learning und Amortized Q-Learning (AQL), werden in dieser Arbeit zur Steuerung von RBS eingeführt. Die AQL-Methode erreichte eine 4,5-fach schnellere Ladezustandsangleichung als der proaktiver Balancing-Algorithmus bei einem System mit zwölf Zellen. Die in dieser Arbeit vorgestellten Methoden und Frameworks bilden eine Grundlage für datenintensive, cloudintegrierte digitale Zwillinge für RBS. Diese Fortschritte tragen maßgeblich zur Transformation hin zur vollelektrischen Mobilität bei und eröffnen neue Perspektiven für intelligente, cloudbasierte BMS.

Abstract

Accurate and precise state estimation of lithium-ion batteries (LIBs) is essential for enhancing the safety of battery storage systems and increasing the adaptability of e-mobility. However, existing battery management systems (BMS) lack in reliability and adaptability across the entire lifespan of a battery pack. Intelligent cloud-based BMS and reconfigurable battery systems (RBS) enable the smart utilization of individual battery cells, thereby improving the overall performance. Additionally, digital twin-based BMS offers predictive maintenance capabilities and adaptive state estimation. The integration of distributed computing resources facilitates the efficient deployment of cloud-based BMS and digital twins. When implementing cloud-based BMS, round-trip latency including computational cost serves as a critical decision-making parameter. In this thesis, state-of-the-art state-of-charge (SOC) estimation techniques are compared. The mean round-trip latency for feedforward neural network (FNN) over long-term evolution (LTE) mobile network of 54.6 ms for an aerial distance of 450 km between the physical BMS and the cloud server was measured. This computational cost is well within typically allowed limits of up to 100 ms of computational delay. Operating temperature significantly influences LIB performance and degradation. Overheating accelerates degradation and may even lead to thermal runaway. Therefore, accurate battery temperature estimation is crucial for safe and reliable BMS operation. To this end, a Kolmogorov-Arnold Network (KAN) and an interconnected Long Short-Term Memory (LSTM) model are proposed for estimating core and surface temperatures based on current, voltage, and ambient temperature inputs. The proposed KAN model achieved an RMSE of 0.735 °C and 0.520 °C for core and surface temperature estimation, respectively, under a 25 °C UDDS drive cycle, demonstrating its potential effectiveness. Beyond state estimation, a robust software architecture is required to enable bidirectional data transfer between vehicles and the distributed computing environment. This thesis proposes a five-layer Internet of Things (IoT) framework and a universal data specification for battery storage systems. Efficient data compression is crucial for transmitting large data volumes. The proposed lossy and adaptive data compression technique utilizes polynomial coding to compress voltage measurements based on current and time data. This approach achieved an RMSE of 5.09 mV with a 99.1 % rate of compression during a UDDS drive cycle at 25 °C ambient temperature. The overall performance of cloud-based BMS can be further enhanced through intelligent RBS. This thesis introduces reinforcement learning-based control algorithms for this purpose. To develop these algorithms, an electrophysical simulation model was proposed as a training environment. This model, implemented using the Python just-in-time compiler Numba, enables the simulation of a one-second uptime of a battery modular multilevel management (BM3) with ten modules in 0.33 s. The proposed model demonstrated a 130 times speed improvement compared to a MATLAB/SIMULINK implementation. Furthermore, two machine learning algorithms Q-learning and Amortized Q-learning (AQL) were developed for BM3 control using reinforcement learning. The AQL algorithm combines classical control techniques with modern machine learning approaches. When compared to the proactive balancing algorithm, the AQL method demonstrated 4.5 times faster SOC balancing for a 12-cell battery pack. The methodologies and frameworks presented in this thesis establish a foundation for data-intensive, cloud-integrated digital twins of RBS. These advancements will significantly contribute to the transition toward fully electric transportation and open new pathways for intelligent, cloud-based BMS.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Research Aim and Objective . . . 2
- 1.3 Research Questions . . . 2
- 1.4 Thesis Roadmap and Publications . . . 3
- 1.5 Research Contribution . . . 5
- 1.6 Structure of the Work . . . 7

2 Background **9**
- 2.1 Electric vehicles and its Components . . . 9
  - 2.1.1 Battery Modeling and State Estimation . . . 10
  - 2.1.2 Electric Vehicle Inverter . . . 13
  - 2.1.3 Battery Management System . . . 17
- 2.2 Cloud Computing . . . 21
- 2.3 Artificial Intelligence, Machine Learning, and Deep Learning . . . 23
  - 2.3.1 Disciplines in the Field of AI . . . 23
  - 2.3.2 Learning Paradigms . . . 24
  - 2.3.3 Essential Learning Algorithms . . . 25
  - 2.3.4 Evaluation Metrics . . . 30
- 2.4 Digital Twin . . . 30

3 Reinforcement Learning for Control of Reconfigurable Batteries **33**
- 3.1 Review of Related Work . . . 33
- 3.2 Simulated Environment for Reinforcement Learning . . . 34
  - 3.2.1 Physical Model of BM3 Reconfigurable Battery . . . 34
  - 3.2.2 Simulation Results . . . 36
  - 3.2.3 Benchmarking Implementations . . . 38
- 3.3 Reinforcement Learning for SOC Balancing . . . 39
  - 3.3.1 Q-learning . . . 40
  - 3.3.2 Amortized Q-learning (AQL) . . . 45

4 Digital Twinning using Cloud Computing **53**
- 4.1 IoT-Framework for Battery Management Systems . . . 53
  - 4.1.1 Monitoring of a Reconfigurable Battery . . . 54
  - 4.1.2 Five-Layer IoT Framework . . . 56
- 4.2 Universal Real-Time Streaming Data Specification for BMS . . . 59
  - 4.2.1 Proposed Data Specification . . . 60
  - 4.2.2 Experimental Validation . . . 67
- 4.3 Distribution of Computing Resources . . . 69
  - 4.3.1 Computing Distribution Architectures . . . 69
  - 4.3.2 State-of-Charge Estimation in the Cloud . . . 71
  - 4.3.3 Data-Driven Core and Surface Temperature Estimation Models . . . 76
- 4.4 Lossy Data Compression . . . 88
  - 4.4.1 Novel Compression Method for Voltage Measurements . . . 90
  - 4.4.2 Accuracy Evaluation of Lossy Data Compression . . . 94

4.4.3 Compression Runtime Evaluation 99

5 Cloud-based State Estimation and Control of a Reconfigurable Battery 101
5.1 Cloud-based BMS and Digital Twin Setup 101
5.2 Cloud-based Digital Twin Methodology 102
5.3 Evaluation and Discussion 104
5.3.1 Interactive Visualization for Monitoring 104
5.3.2 Request Time Evaluation 105
5.3.3 Bandwidth Evaluation 106
5.3.4 Summary 109

6 Conclusion and Outlook 111
6.1 Results and Consolidation of Contribution 111
6.2 Future Work 114

Work related publications 115

Bibliography 117

Acronyms 139

List of Figures 143

Listings 145

List of Algorithms 145

List of Tables 147

1 Introduction

"The problem so far has been to build a storage battery of light weight which would operate for long distances without recharging. Mr. [Thomas] Edison has been experimenting with such a battery for some time". Since Henry Ford's response in an interview with the New York Times on January 11th, 1914 [1], not only the design of vehicles has changed, but also the range of cars increased to more than 600 km [2], [3]. Even electric trucks are prepared for operation. The Tesla Semi could archive a range of 800 km fully charged and fully loaded during a test over two weeks by DHL [4]. With technological advances and extended driving ranges, battery pack capacity has nearly tripled in Electric Vehicles (EVs) [5]. However, improving energy storage systems in terms of efficiency and lifetime is of high environmental and economic interest. Certain limitations must still be overcome to achieve widespread adoption in battery EVs and grid-storage applications.

In this chapter, we first define the motivation of this Thesis. In the second Section, the research aim and objectives are given. Based on the research aim and objectives, we define in Section 1.3 on the following page the research question we work on in this Thesis, followed by the thesis road map and the publications. Section 1.5 gives an overview of the research contribution during the work on this Thesis. At last, in Section 1.6 on page 7, we look through the structure of this work.

1.1 Motivation

This Section sets out the motivation for our research to enable the development of efficiency-enhancing methods using Internet of Things (IoT) technology for electric vehicles. Our research addresses the issue of climate change through efficient electrification of transport.

Climate Change With the adoption of the Paris Agreement [6], the signatory states committed themselves to hold the increase in global average temperature well below 2 °C above the pre-industrial level. Rapid reduction of greenhouse gas emissions is necessary to achieve this commitment target, with much more significant reductions than current policies. According to forecasts, current policies will lead to a global warming of 3.2 °C with a range of 2.2 °C – 3.5 °C above pre-industrial level by the year 2100 [7].

Transportation Electrification With 7.3 Gt carbon dioxide equivalents in 2020, transportation is the second most emitting sector after the sector of electricity and heat [8]. Transportation includes every kind of transportation, including aeroplanes, railways, shipping, and roads. Road transport has the largest share within transportation, equivalent to 12 % of global greenhouse gas emissions [9]. The most effective means of reducing carbon dioxide emissions from combustion are walking, cycling, using public transport, and reducing speed limits, especially on the German Autobahn, which has, in many cases, no speed limit. If a private vehicle is required, it is recommended to an electric vehicle [10], [11]. EVs with batteries as the energy source have a lower life cycle carbon footprint than internal combustion engine vehicles. Therefore, they can contribute to emissions reductions in the transport sector [12]–[14].

Limited Electricity and Resources for EVs Electric vehicles exhibit higher efficiency than

gasoline and diesel engines. EVs achieve approximately 40 % – 70 % of a well-to-wheel efficiency if renewable energy sources are used and depending on the location and type of renewable energy source. The efficiency of EVs is significantly higher in comparison to the efficiency of gasoline and diesel internal combustion engines, which is approximate 11 % to 27 % [15]. However, with 8913.9 TW h representing 30.24 % of the global electric mix, renewable energy is still a limited resource [16]. Also, limited resources such as lithium are essential in modern battery manufacturing [17]. Currently, lithium remains an essential resource for the production of batteries.

Due to climate change, the limited availability of renewable energy sources, and the shortage of resources such as lithium, the efficiency and resilience of EV battery systems are essential to mitigating the effects of climate change.

1.2 Research Aim and Objective

This Section presents our research aim and objectives within cloud-based digital twinning of battery systems, and controlling reconfigurable batteries utilizing machine learning algorithms.

The Research Aim is to investigate the use of cloud computing and IoT technology for Battery Management Systems (BMS) and reconfigurable batteries. This research aims to improve battery systems' efficiency and safety. Furthermore, we aim to study, develop and validate digital twinning for battery storage systems and cloud-based BMS architecture. This will enable the implementation of state-of-the-art battery state estimation and control methods for modern electric vehicles. To achieve the research aim, we have identified the following research objectives:

Research Objectives (ROs)

RO 1 Identify methodologies to collect and transfer EVs battery measurements to distributed compute units for near real-time data analytics, battery state estimation and digital twinning,

RO 2 Implement and investigate cloud computing environment and machine learning methods for monitoring and state estimation of EVs battery systems.

RO 3 Propose, develop and evaluate machine learning methods to control reconfigurable battery systems.

1.3 Research Questions

In this Section, the key research questions are identified. This Thesis proposes an architecture for cloud-based BMS and reconfigurable battery control to utilize machine learning, the Internet of Things, and cloud computing technologies. To achieve this, we list the research questions (RQs) below:

RQ 1 What type of machine-learning algorithm can be applied to enable a self-learning algorithm to implement a control algorithm of a reconfigurable battery and include the possibility of additional classic control methods in combination?

RQ 2 What cloud-based BMS architecture and methodology can be used to efficiently and effectively transfer data between the vehicle and distributed computing units?

1.4 Thesis Roadmap and Publications

The thesis work resulted in 14 publications. Table 1.1 lists how each publication addresses the objectives while answering the research questions we defined in Section 1.2 on the facing page. Fig. 1.1 illustrates the publications and their relation to each other. The publications include peer-reviewed conferences and journal publications – five journal articles, highlighted in orange, and nine conference papers, highlighted in blue. Papers A – D address the first research question, while papers E – L focus on the second. The paper M combines the proposed machine learning control algorithm and cloud-based methodologies. On the other hand, the publications N – U are not included in this Thesis. The research objectives are related to additional, in this Thesis, not considered, research topics in optimized State of Charge and State of Health estimation - publications N, P – S, U -, decentralized reconfigurable battery control - publication O -, and publication U, reconfigurable batteries for aircraft applications.

Table 1.1: Research objectives (RO) addressed in each paper

Publications	RO1	RO2	RO3
Paper A			✓
Paper B			✓
Paper C			✓
Paper D			✓
Paper E	✓	✓	
Paper F	✓	✓	
Paper G	✓	✓	
Paper H	✓	✓	
Paper I	✓	✓	
Paper J	✓	✓	
Paper K	✓	✓	
Paper L		✓	
Paper M		✓	✓

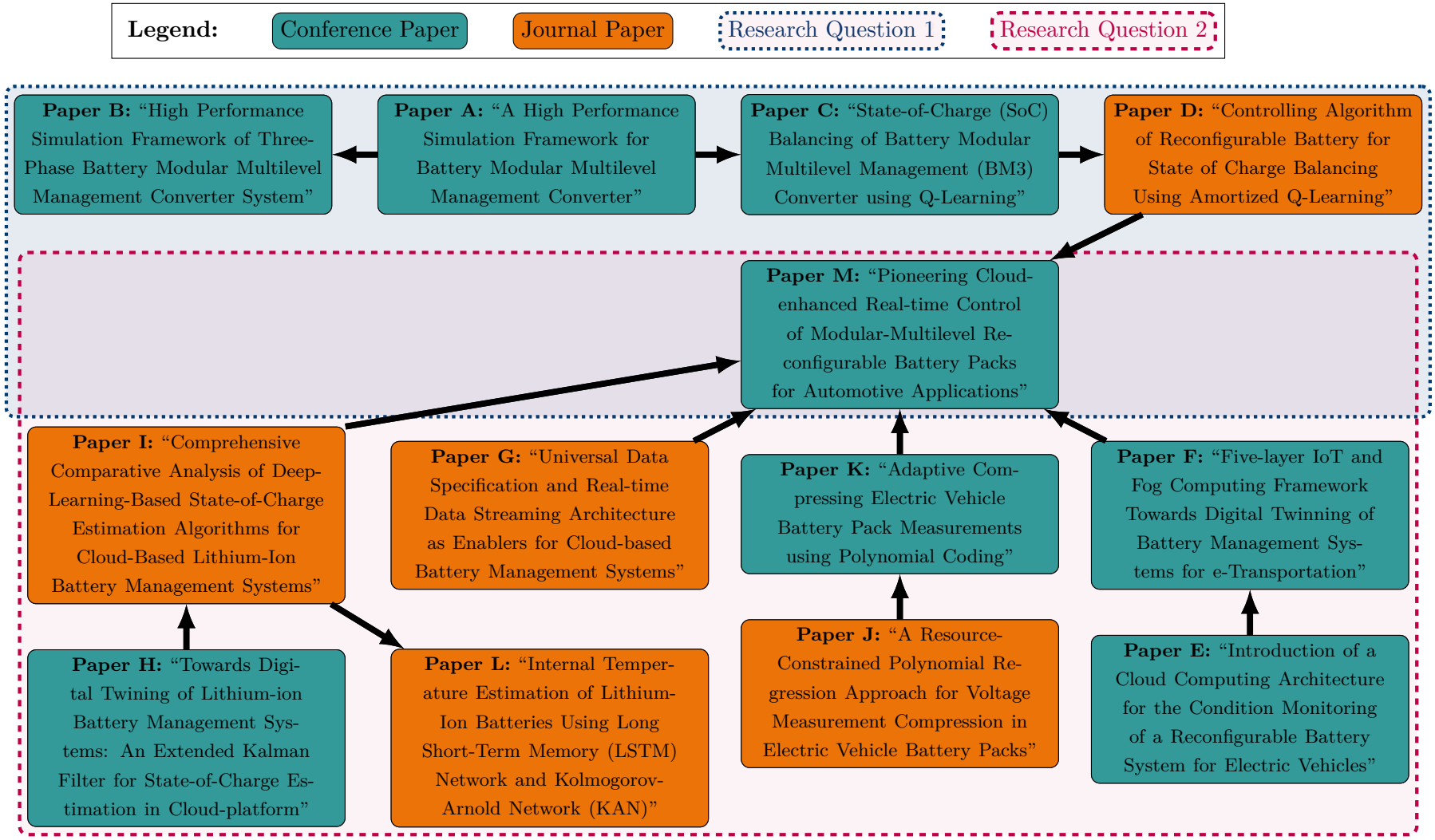


Fig. 1.1. Roadmap of the Thesis with publications, the relation between the publications and the target research question

1.5 Research Contribution

The main contribution of this Thesis is to propose, develop, and evaluate machine learning methods to control reconfigurable batteries, cloud-based BMS, and digital twinning using cloud computing technologies.

The author contributed to 6 journal articles and 15 conference articles during the work on this Thesis. The main research challenge and contributions of the Thesis are presented in terms of publications addressing research objectives and questions. In the following, we give an overview of the thesis and its research objectives and research questions related to the work:

Paper A - "A High Performance Simulation Framework for Battery Modular Multilevel Management Converter" [19]
Authors: Karnehm, Sorokina, Pohlmann, Mashayekh, Kuder, and Gieraths (Neve)
Conference: 2022 International Conference on Smart Energy Systems and Technologies (SEST)
Year: 2022

Paper B - "High Performance Simulation Framework of Three-Phase Battery Modular Multilevel Management Converter System" [18]
Authors: Karnehm, Sorokina, Pohlmann, Ackermann, Kuder, and Gieraths (Neve)
Conference: IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society
Year: 2022

Paper C - "State-of-Charge (SoC) Balancing of Battery Modular Multilevel Management (BM3) Converter using Q-Learning" [20]
Authors: Karnehm, Pohlmann, and Neve
Conference: 2023 IEEE Green Technologies Conference (GreenTech)
Year: 2023

Paper D - "Controlling Algorithm of Reconfigurable Battery for State of Charge Balancing Using Amortized Q-Learning" [21]
Authors: Karnehm, Bliemetsrieder, Pohlmann, and Neve
Journal: Batteries
Year: 2024

Paper E - "Introduction of a Cloud Computing Architecture for the Condition Monitoring of a Reconfigurable Battery System for Electric Vehicles" [24]
Authors: Karnehm, Pohlmann, Wiedenmann, Kuder, and Neve
Conference: 2023 6th Conference on Cloud and Internet of Things (CIoT)
Year: 2023

Paper F - "Five-layer IoT and Fog Computing Framework Towards Digital Twinning of Battery Management Systems for e-Transportation" [23]
Authors: Karnehm, Samanta, Anekal, Pohlmann, Neve, and Williamson
Conference: 4th International Conference on Smart Grid and Renewable Energy (SGRE24)
Year: 2024

Paper G - "Universal Data Specification and Real-time Data Streaming Architecture as Enablers for Cloud-based Battery Management Systems" [26]
Authors: Karnehm, Samanta, Hohenegger, Tashakor, Goetz, Kuder, Neve, and Williamson

Journal: IEEE Journal of Emerging and Selected Topics in Power Electronics
Year: 2024

Paper H - "Towards Digital Twining of Lithium-ion Battery Management Systems: An Extended Kalman Filter for State-of-Charge Estimation in Cloud-platform" [28]
Authors: Karnehm, Anekal, Samanta, Neve, and Williamson
Conference: ECCE 2023 - IEEE Energy Conversion Congress and Expo
Year: 2023

Paper I - "Comprehensive Comparative Analysis of Deep-Learning-Based State-of-Charge Estimation Algorithms for Cloud-Based Lithium-Ion Battery Management Systems" [27]
Authors: Karnehm, Samanta, Anekal, Pohlmann, Neve, and Williamson
Journal: IEEE Journal of Emerging and Selected Topics in Industrial Electronics
Year: 2024

Paper J - "A Resource-Constrained Polynomial Regression Approach for Voltage Measurement Compression in Electric Vehicle Battery Packs" [29]
Authors: Karnehm and Neve
Journal: Batteries
Year: 2024

Paper K - "Adaptive Compressing Electric Vehicle Battery Pack Measurements using Polynomial Coding" [25]
Authors: Karnehm and Neve
Conference: IEEE International Conference on Smart Mobility 2024
Year: 2024

Paper L - "Internal Temperature Estimation of Lithium-Ion Batteries Using Long Short-Term Memory (LSTM) Network and Kolmogorov-Arnold Network (KAN)" [30]
Authors: Karnehm, Samanta, Rosenmüller, Neve, and Williamson
Journal:
Year: 2024
Note: Under Review

Paper M - "Pioneering Cloud-enhanced Real-time Control of Modular-Multilevel Reconfigurable Battery Packs for Automotive Applications" [22]
Authors: Karnehm, Samanta, Neve, and Williamsion
Conference: IEEE ECCE Asia 2025
Year: 2025

In addition, an overview of the author's publications, which are not directly related to the present work, is provided:

Paper N - "State-of-Charge Estimation of Lithium-Ion Batteries Using Machine Learning Based on Augmented Data" [31]
Authors: Pohlmann, Karnehm, Mashayekh, Kuder, Gieraths (Neve), and Weyh
Conference: 2022 International Conference on Smart Energy Systems and Technologies (SEST)
Year: 2022

Paper O - "Multilevel Inverter based Battery System Operation using a Decentralized Controller" [32]
Authors: Schwitzgebel, Buberger, Kuder, Karnehm, Sorokina, Wiedenmann, Mashayekh,

Eckerle, and Weyh
Conference: 2022 International Conference on Smart Energy Systems and Technologies (SEST)
Year: 2022

Paper P - "State of Charge Prediction of Lithium-Ion Batteries Based on Artificial Neural Networks and Reduced Data" [33]
Authors: Pohlmann, Mashayekh, Karnehm, Kuder, Gieraths (Neve), and Weyh
Conference: 2022 24th European Conference on Power Electronics and Applications (EPE22 ECCE Europe)
Year: 2022

Paper Q - "State-of-Health prediction of lithium-ion batteries based on a low dimensional Gaussian Process Regression" [34]
Authors: Pohlmann, Mashayekh, Stroebl, Karnehm, Kuder, Neve, and Weyh
Journal: Journal of Energy Storage
Year: 2024

Paper R - "Voltage Relaxation Pattern Recognition for Efficient Sorting of Healthy Cells for Second-life Applications of Retired Electric Vehicle Batteries" [35]
Authors: Samanta, Karnehm, Neve, and William
Conference: 25th IEEE International Conference on Industrial Technology
Year: 2024

Paper S - "Data-informed Healthy Cell Clustering Technique for Second-life Applications of Retired Electric Vehicle Batteries" [36]
Authors: Samanta, Karnehm, Neve, and Williamson
Conference: 2024 IEEE Transportation Electrification Conference and Expo (ITEC)
Year: 2024

Paper T - "Performance Analysis of Reconfigurable Inverter Battery Systems for Aircraft Applications" [37]
Authors: Sorokina, Bliemetsrieder, Estaller, Karnehm, Weyh, and Kuder
Conference: 2024 24th International Scientific Conference on Electric Power Engineering (EPE)
Year: 2024

Paper U - "Lithium Ion Batteries Impedance Approximation and Model Parameterization Using Artificial Neural Networks" [38]
Authors: Pohlmann, Mashayekh, Karnehm, Kuder, Neve, and Weyh
Conference: 2024 IEEE International Conference on Industrial Technology (ICIT)
Year: 2024

Paper V - "Temporal Sensitivity Analysis of Internal Temperature Informed Charging Algorithms and Rapid Thermal Management System for E-mobility" [39]
Authors: Samanta, Chetri, Karnehm, Neve, and Williamson
Conference: 2024 IEEE Energy Conversion Congress and Exposition (ECCE)
Year: 2024

1.6 Structure of the Work

The Thesis is structured as follows. The theoretical foundations of electric vehicles, batteries, cloud computing, artificial intelligence and digital twins are discussed in Chapter 2 on

the next page. This is followed by the proposal of the simulation environment for a reconfigurable battery system and two reinforcement learning algorithms to control reconfigurable batteries in Chapter 3 on page 33. Digital twinning frameworks and methodologies, such as a five-layer IoT framework and a universal real-time streaming data specification for BMS, are going to be proposed in Chapter 4 on page 53. We evaluate the proposed approaches in Chapter 5 on page 101. Finally, Chapter 6 on page 111 summarizes key findings, acknowledging limitations and the future of reconfigurable batteries and digital twins in electric vehicles.

2 Background

This chapter introduces the background and related work within EVs, battery modelling, topologies of reconfigurable batteries, and BMS. Furthermore, we introduce concepts from Cloud Computing, Artificial Intelligence (AI), and Digital Twins as related work.

2.1 Electric vehicles and its Components

The powertrain of a battery electric vehicle (EVs) contains mainly the following components: a charger, battery pack, BMS, Direct Current (DC)-DC converter, Alternating Current (AC)-DC inverter, and motor engine [40]. Considerable variability exists in the architecture of the components within the EVs. Fig. 2.1 illustrates an example of the architecture of the components and the connection [40]. A battery pack is employed as the vehicle's energy source. The BMS monitors and controls the battery pack to ensure safe, efficient, and reliable operation. The off-board charger enables the charging of the batteries with grid energy. The charger contains an AC-DC converter and ensures the overcharge safety of individual battery cells. The schematic in Fig. 2.1 includes an off-board charger. It is also possible to implement an on-board charger. Charging systems consist of AC-DC power converters to charge the battery in a safe and highly efficient way [41]. The combination of AC-DC and DC-DC converters is used to charge the battery. Furthermore, a unidirectional converter can be utilized to charge and discharge the battery pack. When operating the electric power train, it is necessary to take voltage from different systems and components and feed it into others, which may have different power and voltage requirements. A DC/DC converter is a power electronic circuit that converts a DC voltage into a different DC voltage level. The DC/DC converter supplies a constant voltage and current level, which must be converted into an AC voltage level to supply the electric machine. The inverter supplies variable voltage, current and frequencies. The AC motor is an element that converts the electrical energy from the battery into mechanical energy. The page 149 contains a catalogue of the electric circuit components that are utilized in this work.

Energy Storage devices installed in EVs are super-capacitors, fuel cells, and batteries, which are differentiated by characteristics and functionality. The battery is a component that stores electrochemical energy and supplies electric power. EV battery packs are structured in a pack-module-cell structure. A cell is a single battery. Multiple cells are grouped together into modules. Modules are bundles of cells connected to increase overall voltage and capacity. In general, cells are serial-parallel connected to modules. Modules are serial connected to a pack. The highest level is the pack, which consists of multiple modules. The amount of serial-parallel connections and cells defines the capacity and power of the vehicle [42], [43].

The BMS is a crucial component of modern battery-powered systems. It is fundamental in ensuring safety, monitoring battery performance, estimating battery state, and controlling charging and discharging processes [42], [44].

The Inverter converts the in batteries stored DC power into three-phase AC power to drive the engine [45]. The employed motor system and its topology are another essential component of the EV. Single or several motors can be installed within the vehicle. The

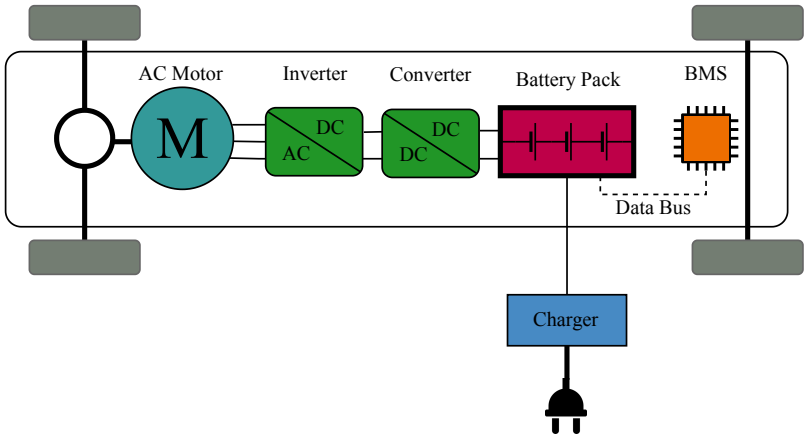


Fig. 2.1. Schematic of a conventional battery electric vehicle with off-board charger, battery pack, BMS, converter, inverter, and a three-phase AC motor [40]

centrally located motor enables integration into existing combustion vehicle chassis. In addition, there are wheel and wheel hub motors that can increase the efficiency of the EV [40].

2.1.1 Battery Modeling and State Estimation

In 1991, Sony Corporation used the first rechargeable Lithium-Ion (Li-Ion) battery for the very first portable phone [46]. The technology of Li-Ionbatteries found its way into many use cases. Li-Ion batteries are widely used in EVs due to their safety, lifetime, and inherent advantage in energy efficiency [47]. A schematic component is illustrated in Fig. 2.2. The components of the battery are a cathode, anode, and separator. The positive cathode and negative anode are connected by an electrolyte holding the lithium ions and isolated by a separator [48]. Lithium ions migrate from the cathode to the anode during the charging process, whereas during discharging, they traverse in the opposite direction. Battery cells differentiate in their form factors. In the field of EVs, solid cylindrical cells, larger solid prismatic cells, and soft pouch or polymer cells are employed as form factors [43]. This work will focus on cylindrical Li-Ion batteries. Cylindrical battery cells have demonstrated significant growth in utilization within the automotive industry over the past years [49]. These batteries have shown great promise in various applications, and understanding their performance is crucial for future advancements.

To estimate cells' inner state, accurate modelling of their behaviour is required. Therefore, various methodologies can be utilized. Depending on the accuracy required and the accessible computational resources, different approaches to battery models can be used [50]. Battery models proposed and reviewed in the literature [51], [52] can be classified into the following three categories:

Electrochemical models characterize battery behaviour through chemical parameters, including the dimensions of the anode and cathode, the materials utilized, electrolyte concentration, and the underlying electrochemical processes.

Equivalent Circuit Model (ECM) uses electrical elements, such as voltage sources, resistors, and capacitors, to replicate the battery performance characteristics.

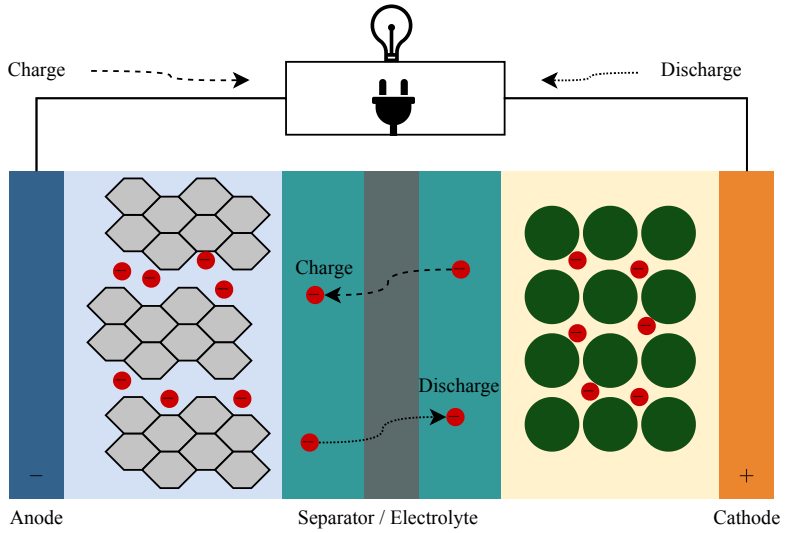


Fig. 2.2. Li-Ion battery schematic containing an anode and cathode with a separator/electrolyte in between and charge/discharge ion movement arrows

Data-Driven models including several types of machine learning approaches. Prior knowledge of battery parameters is not required to define the model. The model's accuracy depends on a large quantity of high-quality data, including different operational conditions, such as load, drive cycle, battery age, and temperature.

Our work focuses on Equivalent Circuit Model (ECM) and data-driven estimation models. In Fig. 2.3, an overview of Integer-Order Models (IOM), a subcategory of ECMs, is illustrated. In Fig. 2.3, the component (a) shows the Rint model, the most simple IOM. It uses the two electrical components ideal voltage source V_{oc} and the internal resistance R_0 connected in series [53]. Both components' parameters depend on State-of-Charge (SOC) and temperature. This model has low complexity but also low accuracy and does not describe the dynamic processes of a battery cell. The other IOM models are the Thevenin model (b) and the higher-order Thevenin Model, also known as the Dual Polarization model (c), and N RC Thevenim Model (d). These models are based on the Rint model and add serial RC networks consisting of parallel connected resistance and capacitors to simulate the polarisation effect of the battery. Such as the voltage source and the internal resistance; the parameters of these components are dependent on battery states such as SOC and temperature. They also change during the ageing of the cell. Electrochemical processes can be produced at multiple cell frequencies with a third- or higher-order RC model. However, a large set of parameters is required to describe these models [54], which leads to increased complexity in parameter identification, calibration, and the risk of overfitting.

The cell voltage u_B of a Dual Polarization model, shown in Fig. 2.3 (c), is defined as [55]:

$$u_B = V_{oc}(SOC) - i_B R_1 \left[1 - \exp\left(-\frac{T}{\tau_1}\right)\right] - i_B R_2 \left[1 - \exp\left(-\frac{T}{\tau_2}\right)\right] - i_B R_0, \tag{2.1}$$

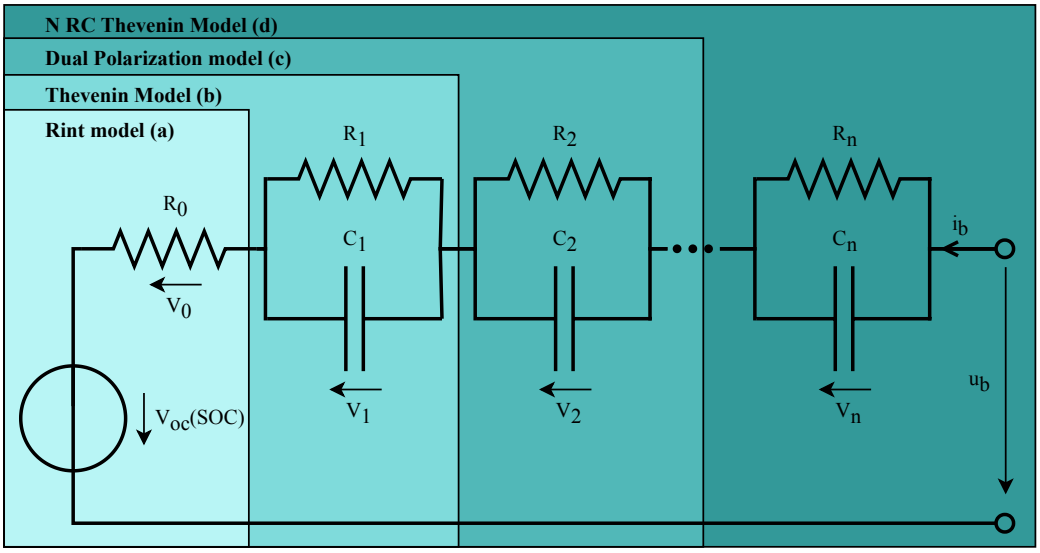


Fig. 2.3. Equivalent Circuit Model (ECM) IOM used to describe Li-Ion battery behaviour [50]; Models in ascending complexity: Rint model (a), Thevenim Model (b), Dual Polarization model (c), and N RC Thevenim Model

where

i_B	battery current,
V_{oc}	open circuit voltage depending on SOC,
T	charge and dischage time of the cell, and
τ_i	$R_i \times C_i$ of ith RC network

The following introduces the battery states SOC, State-of-Health (SOH), and core temperature.

State-of-Charge (SOC)

The relative charge of the battery to the battery capacity defines the State-of-Charge (SOC). It denotes the remaining energy capacity within the battery, which is comparable to the fuel level of the tank of a combustion engine [44]. SOC is conventionally quantified as the percentage of relative charge to battery capacity. The Coulomb counting SOC at the time step $t + \Delta t$ is in [56]:

$$SOC(t + \Delta t) = SOC(t) - \int_0^{\Delta t} \frac{i_B}{Q_c} dt, \tag{2.2}$$

where

Q_c	remaining current capacity of the cell,
i_B	the battery current,
t	the time stamp, and
Δt	the time difference between two time stamps

State-of-Health (SOH)

The actual remaining capacity Q_c of the battery compared to the nominal capacity Q_n of a new cell of the same type defines the SOH [44]:

$$SOH = \frac{Q_c}{Q_n} \tag{2.3}$$

The SOH is impacted by charge/discharge stress, amount of charge-discharge cycles, temperature, and calendar age. It also considers the battery's performance, such as current, internal resistance, voltage, self-discharge rate, and heat generation [44].

Core Temperature

Temperature management is a crucial task within an electric vehicle. Temperature significantly influences battery performance and lifespan of the cell, and high temperatures can cause thermal runaways with serious safety issues for the cell, vehicle, drivers, and their environment [57]. The battery cell is influenced by the ambient temperature around the cell, its surface temperature, and the core temperature in its inner. It is necessary to estimate the core temperature of a battery cell accurately because it can increase the performance and safety of the vehicle [58]. The core temperature is not a directly measurable parameter of a battery cell.

In Fig. 2.4a a schematic of the temperatures T_{amb}, T_S, and T_{core} is illustrated. The ambient, surface, and core temperatures influence each other. Chen et al. [58] propose an electro-thermal model to estimate the core temperature depending on the model and the parameters R_i, R_o, C_S, and C_C are shown in Fig. 2.4b:

Q_{re}	heat generation rate,
T_S	surface temperature,
T_A	ambient temperature,
R_i	thermal resistance between battery core and surface,
R_o	resistance between battery surface and environment,
C_C	heat capacity of internal medium, and
C_S	heat capacity of battery shell end external medium

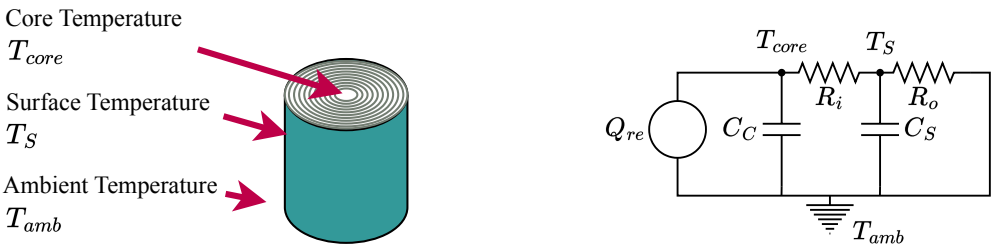
Both resistances are given in [K/W] and the capacities in [J/K].

2.1.2 Electric Vehicle Inverter

The inverter converts the stored DC power from the battery to AC to drive the three-phase AC engine. Islam et al. [45] highlighted that inverters can be classified by different categories, such as input source, output phase, output voltage, number of voltage levels, Pulse Width Modulation (PWM) and others. This work discusses the voltage-level-wise classification: two-level inverters and Multi-level Inverter (MLI).

Two-level Inverter

In the global EVs market, most EVs use a two-level inverter due to its simplicity and cost efficiency [59], [60]. Fig. 2.5 shows a two-level, three-phase full-bridge inverter with six power switches. The three phases, U, V, and W, are utilized to transfer the power of the three phases AC to the engine as indicated in Fig. 2.5. The two levels refer to the positive



(a) (b)

Fig. 2.4. Illustration of temperatures related to a battery (a) and an electro-thermal model (ETM) to describe electrochemical behaviour and estimate the temperatures of the cell (b) [58]

and negative voltage levels the inverter outputs. For example, to achieve a negative level in the U phase, S_2 has to be switched on, and S_1 has to be switched off. In contrast, to achieve the positive level, the reverse configuration of these two switches needs to be applied.

The illustrated topology is one of many two-level power inverters discussed in the literature [59]. In addition to the topology, control schemes that utilize PWM also affect the losses and temperature profiles of the converter's power semiconductors and the power converter itself [61]. Several control schemes, such as space vector-based PWM (SVPWM), triangle comparison-based PWM (TCPWM), and sine-wave pulse width modulation (SPWM), are well-known and employed in electric vehicles to power three-phase engines. Two-level inverters generate undesirable harmonics, compromising power quality and necessitating supplementary filtering [62]. Furthermore, they demonstrate lower efficiency than multilevel inverters, particularly at high power levels, resulting from increasing switching losses compared to lower switching losses in multilevel inverters [63].

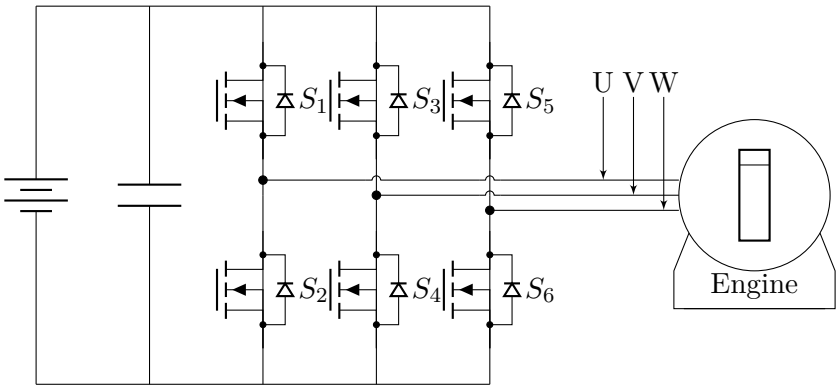


Fig. 2.5. Three-Phase (U, V, W), six Switches (S_1 – S_6), Two-Level Inverter Topology for three Phase EV Motor with Battery Pack and Capacitor

Multi-level Inverter (MLI) for Reconfigurable Batteries

MLI enables the reconfiguration of battery connections in the battery pack during the discharge and charge [64], [65]. Multilevel inverters can be used to improve power quality, efficiency, and reliability. Depending on the number of levels, MLI generate near-sinusoidal

voltages with reduced harmonic distortion, which enhance the motor drive system's power quality and efficiency of the motor drive system [66]. Using multiple voltage levels, MLIs reduce switching losses and improve the overall efficiency of the power conversion process [67]. MLIs enable a higher fault tolerance and can continue operating even if some power switches fail. These factors make them reliable for EV applications [68]. Also, MLI has shown the possibility of bidirectional charging [69], which could also reduce the need for on-board and off-board charger hardware.

The various MLI topologies proposed in the literature include Bidirectional double full-bridge/MMSPC [70], half-bridge [71], unidirectional full-bridge [72], bidirectional full-bridge [73], Battery Modular Multilevel Management (BM3) [74]/Marx [75], and switch-clamped half-bridge [76]. In Fig. 2.6, the circuits of these topologies are illustrated. The topologies differentiate in their amount of switches per module and the potential configurations of each module in conjunction with other connected modules.

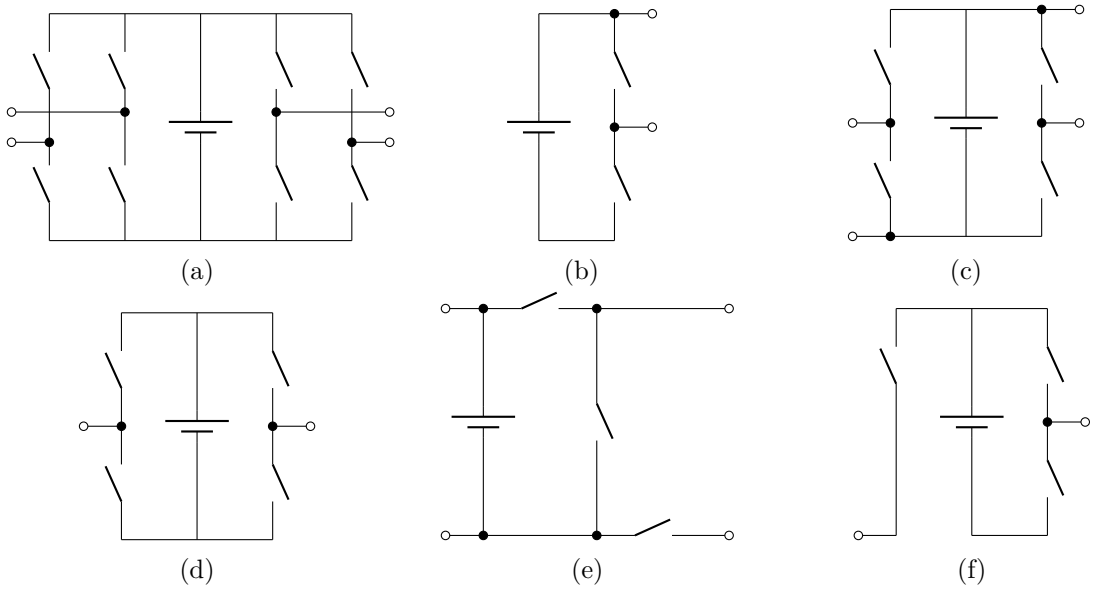


Fig. 2.6. Six different topologies employed in reconfigurable batteries as micro topology: Bidirectional Double Full-Bridge/MMSPC (a), Half-Bridge (b), Unidirectional Full-Bridge (c), Bidirectional Full-Bridge (d), BM3/Marx (e), and Switch-Clamped Half-Bridge (f)

In addition to different MLI topologies, the architecture of reconfigurable batteries can be categorized. In our work, we follow the categorization by Komsiyska et al. [77]:

Type 1 uses the option of reconfiguration to impact the states of the batteries. It employs the reconfigurable battery as a DC voltage power source based on a modular reconfigurable DC battery pack [5], [78]. This type of reconfigurable battery pack can directly replace a conventional battery pack; thus, the DC-DC converter is no longer required.

Type 2 adjusts its configuration dynamically and continuously. Kuder et al. [74] describe this type as an AC three-phase sinosial voltage source with a 120° phase shift among the three phases.

Type 3 are Type 1 or Type 2 battery systems, called a hybrid battery storage system. It operates with different energy storage units within one reconfigurable battery.

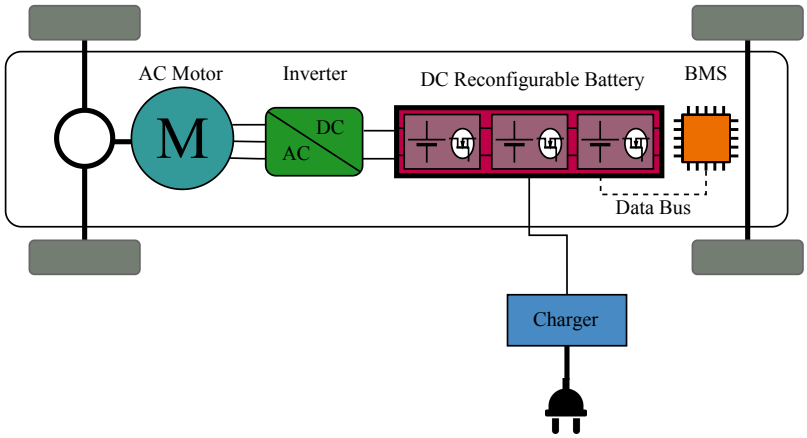


Fig. 2.7. Schematic of an EV with Type 1 reconfigurable battery with off-board charger, Type 1 reconfigurable battery, BMS, inverter, and a three-phase AC motor

In Fig. 2.7 a Type 1 and Fig. 2.8 a Type 2 reconfigurable battery within an electric vehicle is illustrated. Compared to a conventional battery pack, as shown in Figure 2.1 on page 10, a Type 1 reconfigurable battery can produce different DC levels. Therefore, the installation of a DC-DC converter is not necessary. The Type 2 reconfigurable battery does not require the AC-DC converter, as it can be used directly as an AC power source. Furthermore, a Type 2 reconfigurable battery can be employed for bidirectional charging with AC and DC without additional charger hardware [79], [80].

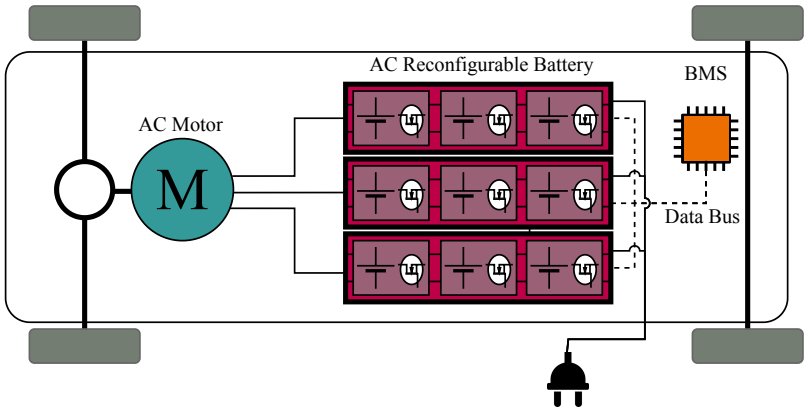
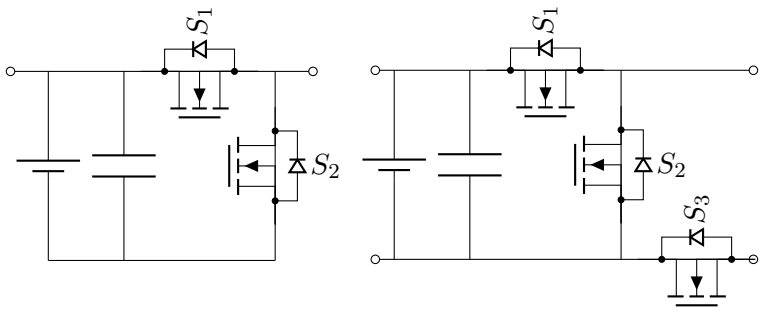


Fig. 2.8. Schematic of an EV with Type 2 reconfigurable battery with a three-phase AC reconfigurable battery, BMS, and three-phase AC motor

Our research lab implements two topologies of Type 2 reconfigurable batteries: half-bridge [72] and BM3 [74] topologies. Fig. 2.9 illustrates the circuit of the two switch module half-bridge and the three switch module BM3. Multiple modules are interconnected with each other in three phases to a MLI to set up a Type 2 reconfigurable battery. In general, more switches lead to more possible configurations and higher flexibility but increase the



(a) (b)

Fig. 2.9. Circuit of a two-switch, serial bypass configurable half-bridge (a) and a three-switch serial, parallel and bypass configurable BM3 module (b)

complexity of controlling. The two-switch module, shown in 2.9a, enables serial and bypass configuration. The BM3, as illustrated in Fig. 2.9b, enables three configurations: serial, bypass, and parallel. Table 2.1 shows how each configuration is switched.

Table 2.1: Principle of operation of Half-Bridge (left), and BM3 (right); Consider off as not connected Metal–Oxide–Semiconductor Field-Effect Transistor (MOSFET), and on as connected

	S_1	S_2
Serial	off	on
Bypass	on	off

	S_1	S_2	S_3
Bypass	on	off	off
Parallel	on	off	on
Series	off	on	off

The half-bridge topology benefits from its simplicity, low complexity in control, and low converter price compared to other MLI topologies [81]. On the other hand, the topology requires voltage balancing and is limited in power management [72], [81]. Thus, the three-switch module BM3 enables power management and voltage balancing with its parallel mode. The parallel mode increases the risk of circulation currents, which causes battery losses [82].

Compared to conventional two-level inverters, reconfigurable batteries gain less current and voltage harmonic distortion and less voltage stress on semiconductor devices [45]. Addressing circulating currents, managing reliability decrease due to increased component count, and achieving voltage balancing is essential for effectively employing the reconfigurable battery. Reconfigurable batteries represent a promising technology with open challenges to increase reliability, safety, and efficiency [70].

2.1.3 Battery Management System

Monitoring, battery state estimation, and controlling cells and battery packs are crucial tasks. The BMS is a hardware and software component within an EVs to control and optimize the use of the batteries [44]. The main aim of BMS is to prevent unsafe or dangerous use, estimate the battery's condition over time, and improve battery performance [83]. The main functions of a BMS are data collection, state estimation, thermal and discharge/charge management, balancing, communication, and alarming [84]. Current, voltage, and temperature are the main parameters of the battery and the battery pack to be monitored by BMS. The sampling rates for these three primary parameters should be at least 1 Hz and higher, as indicated by Yang [84]. The estimation of SOC, SOH, cell temperature and other battery parameters are only some of the tasks to be performed by a modern BMS. Fig. 2.10 shows

software and hardware components of a modern BMS needed to ensure an optimized and safe operation of EVs battery pack. To enable data acquisition, communication between BMS, battery pack and motor control, and alerting, data communication is a significant part of BMS [84]. Furthermore, in the case of a hierarchical structure with a main BMS and submodules of the BMS, stable communication between the main and submodules needs to be ensured.

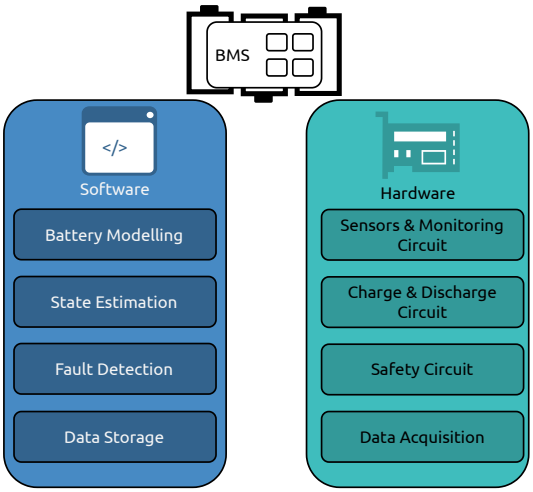


Fig. 2.10. Soft- and Hardware components of a modern BMS [44]

With rising computational and storage requirements for BMS and the developments in cloud computing, big data, and machine learning, cloud-based BMS emerged [85]. Yang et al. [86], [87] propose a cloud-based BMS implementing the Cyber Hierarchy and Interactional Network (CHAIN) framework. The authors propose a distributed BMS framework that employs end-, edge-, and cloud computing and a knowledge repository to achieve highly accurate multi-scale algorithms within the CHAIN framework. In Fig. 2.11, the four subsystems and their relation to each other are illustrated. The subsystems are described in [87]:

End Sensing: Data acquisition from different types of sensors within different measurement accuracy and frequency.

Edge Computing: Time-sensitive monitoring and computing, data collection and transmission into the cloud.

Cloud Computing: Long-term data storage, data analysis with advanced data analytics such as deep learning algorithms.

Knowledge Repository: Orchestration optimization by the monitor-analyze-plan-execute over shared knowledge (MAPE-K) loop.

Li et al. [88] developed a cloud-based BMS for estimating SOC and SOH using an extended Thevenin model paired with an adaptive extended H-infinity filter and particle swarm optimization. Tran et al. [89] suggested a cloud-side model for highly accurate SOH estimation, primarily focusing on the model's accuracy. Their study neglects the analysis of computational cost, technical architectural framework specifics, and cloud-based BMS

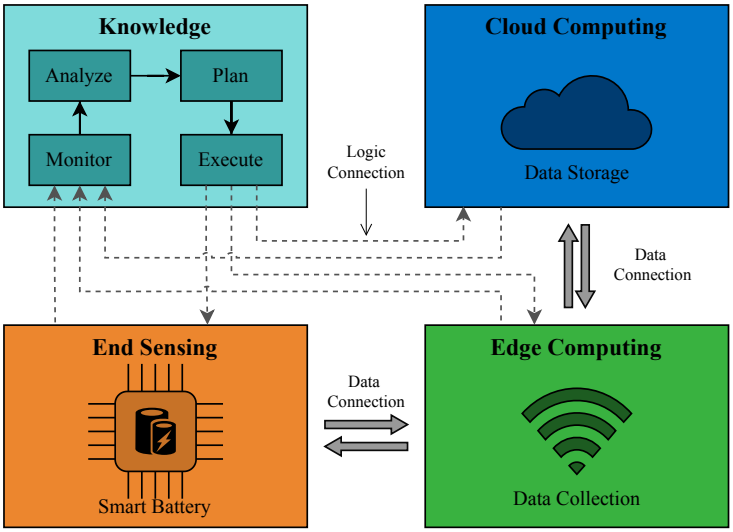


Fig. 2.11. CHAIN framework consists of End, Edge and Cloud components and orchestration component, Knowlege Repository; Compute units relation between each other and knowledge repository is illustrated by data and logic connections [87]

fault detection and prevention. Commercial examples of cloud-based BMS, such as Battery in the Cloud proposed by Bosch Mobility Solutions, EVE-Ai by Electra, Battery Analytics Platform by TWAICE, and Elysia Cloud Platform have been evaluated in the literature and compared with the current state-of-the-art in research [90], [91]. An overview of commercial cloud-based BMS systems, their features, and their description is given in Table 2.2.

Table 2.2: Comparison of commercial industrial cloud-based BMS solutions [90], [91]

Solution	*Health Monitoring*	*Life Forecasting*	*Real Time Data*	*Unique*	*Description*
Bosch Mobility Solutions	✓	✓	✓	"Usage certificate to go" service	Considers factors such as driving style and the surrounding driving conditions to optimize battery operations, thus extending the battery's lifespan. Wear and tear on the battery can be reduced by approximately 20 %.
Elecra EVE-Ai	✓	✓	✓	EVE-Ai 360 Adaptive Control	A versatile system equipped with AI-driven SOC/-SOH prediction features. It provides guidance on remaining range and ideal driving speeds. Enhanced diagnostics and minimized downtime can be attained.
TWAICE Battery Analytics Platform	✓		✓	Hybrid modelling, data security, and compliance	Integrates deep battery knowledge, AI, and real-life battery data to provide a single source of truth for battery development and operation.
Elysia Cloud Platform	✓	✓		Simulation toolkit, SIMULINK integration	Provides detailed health and life insights. Life forecasting captures non-linear degradation and monitors fleet-wide degradation trends to identify potential issues at the cell and pack level.

2.2 Cloud Computing

As shown in Fig. 2.12, cloud computing is one of many computing paradigms. Surianarayanan and Chelliah [92] categorize computing paradigms into three types based on application types. The three aspects are enterprise applications, high-performance computing applications, and selling IT resources models.

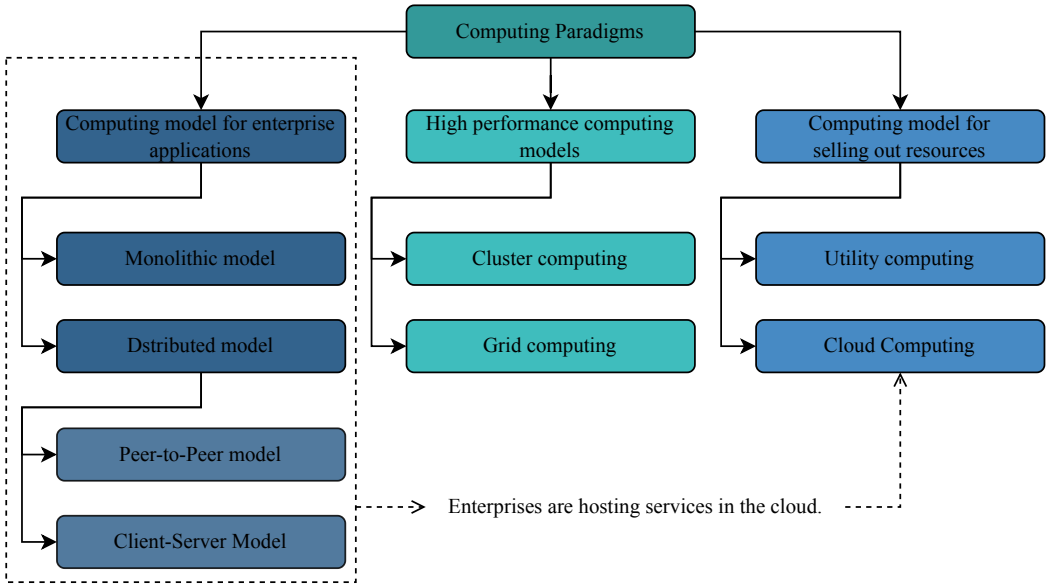


Fig. 2.12. Hierarchy of Computing Paradigms and relation from enterprise to selling out resources [92]

Enterprise applications are based mainly on a modular concept to decrease the complexity of the individual components, increase flexibility, and achieve better maintainability. A monolithic model employs only one physical machine to enable the presentation, business and data access logic. In addition, the distributed model can be separated into peer-to-peer models or Client-Server models. Within the peer-to-peer model, all computing nodes are interconnected and serve as clients and servers. The nodes share their resources without a separate server. The Client-Server models follow a main-worker configuration. In this model, clients and workers connect to a dedicated server, the main control node.

High-performance computing applications perform compute-intensive tasks in business or science. A cluster assembles parallel or distributed computing nodes that are interconnected [93]. These nodes collaborate to execute tasks that individual nodes cannot handle. Individual nodes share the computational workload as one virtual server. Grid computing combines computers from various administrative domains to achieve a common objective, to fulfil a single task, and then it may disband afterwards. Grid computing entails performing calculations in a distributed manner, which may also include combining large cluster-based computing systems.

Selling out of computational resources is the third type. Utility computing enables a pay-and-use model for computing power. The user of the hardware manages the utility

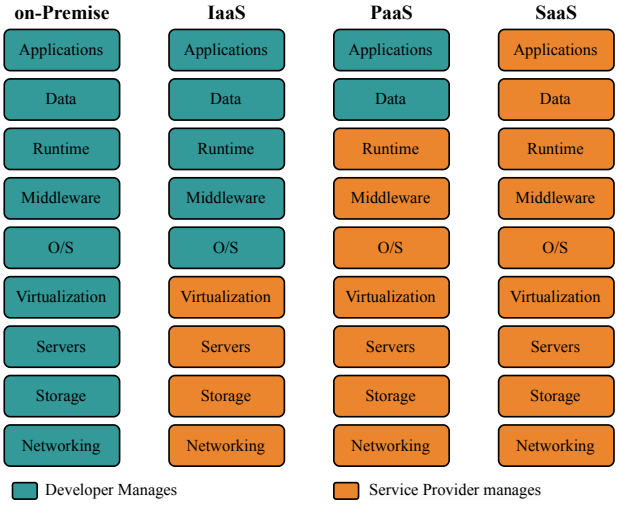


Fig. 2.13. On-Premise as classic approach compared to the three cloud computing service models PaaS, IaaS, and SaaS

computing environment and is deployed on-premise [94]. Cloud computing provides computational resources on-demand over the internet [93] as a pay-as-you-go model.

The National Institute of Standards and Technology (NIST) [95] differentiates between the three service models: Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and Software as a Service (SaaS). Fig. 2.13 illustrates the components to be managed in these three service models. It also includes on-premise as the model, where the developer has to manage the components. The developer is responsible for setting up the application network in on-premise environments. In contrast, in the case of SaaS, the service provider manages the entire stack. In the IaaS and PaaS, the responsibility for managing the stack is shared. In the case of PaaS, the application and the data are provided by the developer and the rest of the stack, from runtime to network, is provided by the service provider.

Over the last few years, serverless computing and Function as a Service (FaaS) applications become the fourth service model. Shahrad et al. [96] define functions in FaaS as event-driven, simple, standalone, developer-written to scale automatically the execution depending on requirements.

In addition, it can be differentiated between several deployment models. The cloud definition of NIST differentiates between private, public, hybrid, and community cloud. Sumit Goyal [97] points out that the community cloud is a mixture of private and public clouds. Furthermore, it is of minor relevance in practical implementation, so it will not be discussed further. Fig. 2.14 illustrates the following described deployment models [95].

Private Cloud: Infrastructure is provisioned for exclusive use. It is possible to be operated, owned and managed by third-party organizations.

Public Cloud: Infrastructure is operated, owned and managed by an organization and open for use by the general public.

Hybrid Cloud: A composition of two or more distinct infrastructures, such as a private cloud and a public cloud environment.

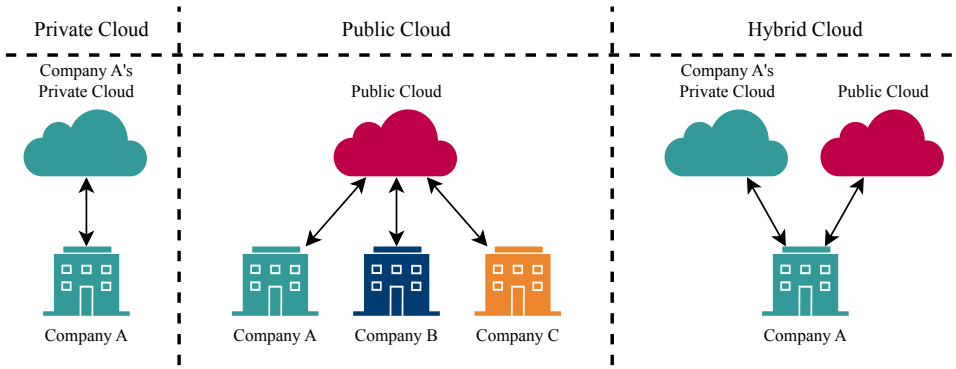


Fig. 2.14. Difference between deployment models private, public and hybrid cloud

Cloud computing enables global distribution of infrastructure and flexible on-demand scaling with its underlying pay-as-you-go model for payment. A wealth of literature provides an overview of cloud computing topics such as the described deployment models of public, private, and hybrid cloud, the four service models and other cloud computing-related concepts, such as the computing paradigms edge computing, fog computing, and cloud computing [92], [98]–[100].

2.3 Artificial Intelligence, Machine Learning, and Deep Learning

The following section gives an overview of the different disciplines in Artificial Intelligence (AI), the three main learning paradigms, and the essential learning algorithms employed in this work, Q-learning, Feedforward Neural Network (FNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Kolmogorov–Arnold Networks (KAN). The rationale behind the chosen algorithm is given depending on the research application in sections 3 and 4.

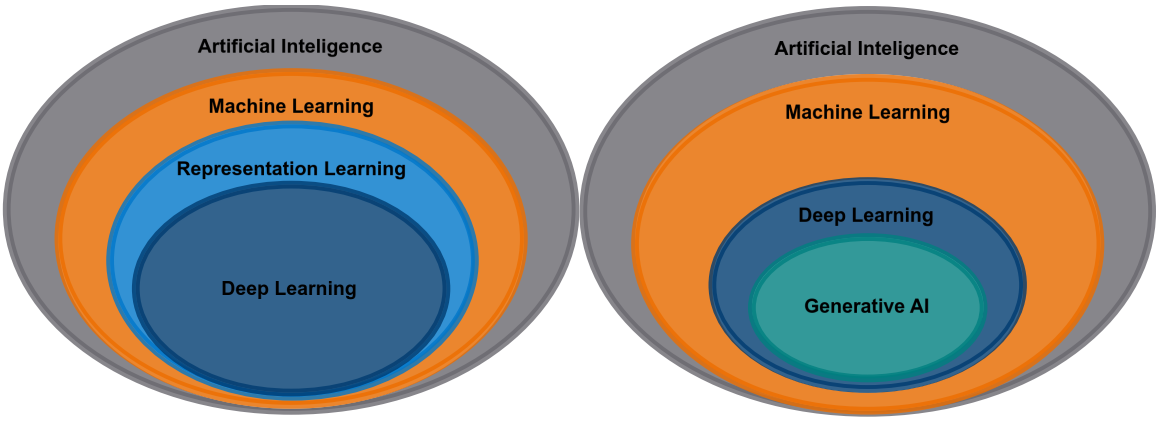
2.3.1 Disciplines in the Field of AI

Fig. 2.15 illustrates the relationship between the different AI disciplines, which discipline is a subdiscipline of the other. Furthermore, the relationship between and the selection of the disciplines has changed over time. The difference between Figure (a) Goodfellow et al. [101] and (b) Zhuhadar and Lytras [102] illustrates this evolution and development between 2016 and 2023.

Artificial Intelligence (AI) encompasses various technologies designed to mimic human-like actions through computational systems [101], [102] and defines the highest level of all the disciplines. An early example of such a AI is the chess computer Deep Blue [103].

Machine Learning (ML) extracts knowledge and patterns from labelled data, raw data and the interaction with an environment. Simple Machine Learning (ML) algorithms are, for example, linear regression [104], naive Bayes [105], and Support Vector Machines (SVM) [106]. Machine learning algorithms are used for spam detection and recommender systems.

Representation learning or *feature learning* enables automated exploration and extraction of features from raw data [107]. Exemplary algorithms are principal component analysis (PCA) [108] and deep auto-encoders [109]. While in Zhuhaar and Lytras [102] definition, the discipline of representation learning is not considered.



(a) (b)

Fig. 2.15. The Venn diagram shows relationship between AI and subtypes; (a) as original proposed in [101] and (b) over time advanced [102]

Deep Learning (DL) methods are traced back to the early 1940s, known as cybernetics and connectionism. During the 1980s and 1990s, the developments were inspired by biological brain functions [101]. DL uses deep neural networks consisting of multiple layers of different types of interconnected nodes. These layers sequentially extract and refine features from the input data. Deep Learning (DL) algorithms are for example FNN, LSTM [110], and Convolutional neural network (CNN) [111]. Deep Learning is used for object detection, such as YOLO [112], and deep reinforcement learning, such as playing Atari 2600 games [113] and the board game Go [114].

Generative AI is designed to produce text, images, and other media by utilizing input from the model. Generative adversarial networks (GANs) [115] and generative pre-trained transformers (GPTs) [116] are the main frameworks enabling this discipline. Generative AI is used in large language models, such as GPT-3 [117] and LLaMA [118], and text-to-image models, such as DALL-E 2 [119].

In this work, ML and DL methodologies are proposed to enable state estimation within the domain of batteries and the control of reconfigurable batteries.

2.3.2 Learning Paradigms

AI has three main learning paradigms: supervised, unsupervised, and reinforcement learning [120]. Fig. 2.16 illustrates these three paradigms. The figure illustrates the input and output during training and prediction. A supervised learning algorithm (a) acquires knowledge from annotated datasets by mapping input variables to a target output. Unsupervised learning (b) does not require annotations in the training data. In reinforcement learning (c), an agent interacts with its environment and learns to maximize its reward during interactions. This work focuses on supervised and reinforcement learning.

In supervised learning , the model gets labelled data as training input to approximate a function $f(D) \rightarrow L$ with input data D to the output labels L. The labels received during the training allow the model to estimate the relationship between the input data and the output. Using the model, it estimates a label based on the previously trained approximation. Supervised learning is applicable to classification, regression, and ranking tasks [121], [122]. In the field of battery research, supervised learning can

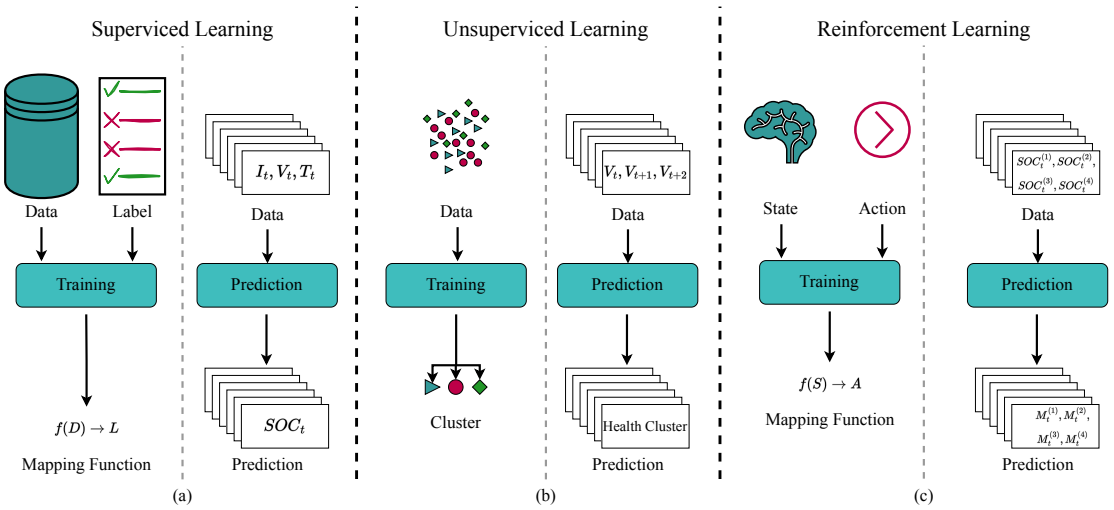


Fig. 2.16. Learning paradigms: supervised (labelled data to mapping function), unsupervised (raw data to clusters) and reinforcement learning (states and actions to mapping function).

be employed to predict battery states, such as the SOC, employing measurements, such as current, voltage, and ambient temperature [28].

In unsupervised learning , models get data input to identify clusters within the input data. The data received during training enables the identification of similarities and separate clusters from each other. An example of unsupervised learning in battery research is the identification of health state clusters to identify batteries for second-life applications at the end of the life of a battery pack [29].

A reinforcement learning model does not require training data. An agent learns by interacting with an environment. The goal of learning is to maximize the cumulative reward during the interaction. Fig. 2.17 illustrates the interaction between agent and environment during training. A Markov Decision Process (MDP) defines the learning problem. The MDP contains possible agent actions $A = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m\}$, the set of all posible environment states $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$ and the reward function $r(\mathbf{a}, \mathbf{s})$. The action $\mathbf{a}_t$ is chosen through the agent based on the current environment state $\mathbf{s}_t$. As a response, the environment returns a reward $r_t := r(\mathbf{a}_t, \mathbf{s}_t)$, determined by the reward function, and the next state $\mathbf{s}_{t+1}$ to the agent [123], [124]. Reinforcement learning can select the configurations M of reconfigurable battery modules based on the state of the cells [21].

2.3.3 Essential Learning Algorithms

AI leverages a diverse toolbox of algorithms. Although there are numerous techniques, this work focuses on researching Q-learning, FNN, LSTM GRU, and KAN as AI algorithms to control reconfigurable battery and battery state estimation.

Q-Learning

Q-learning is a model-free reinforcement learning algorithm [125], [126]. The agent aims to maximize the expected total reward of the actions taken, thereby finding the optimal

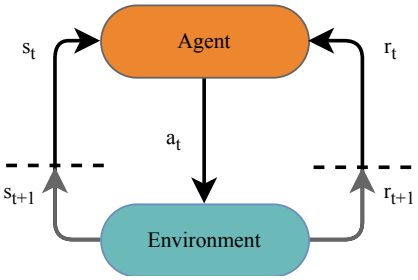


Fig. 2.17. Agent-Environment Interaction in Reinforcement Learning [21]; Agent performs action a_t based on state s_t and rewards state s_{t+1} and reward r_{t+1}

solution of the finite MDP [127]. In its simplest form, Q-learning stores the results of the learning process in a table, known as a Q-Table. The table can be employed as a lookup table to map state and action to a Q-value. The learning process during each episode t, a sequence of interactions between agent and environment, in Q-learning, is defined in [127], [128] as follows:

- Observe the current state s_t of the environment
- Select and perform an action a_t using the ϵ-greedy policy. The action is chosen by the highest Q-value of state s_t: $a_t = \max(Q(s_t))$. With the probability of ϵ, a random action is selected instead of the action with the highest Q-value.
- Receive the reward r_t and the new state s_{t+1} by the environment.
- Update the Q values with the learning rate α and discount factor γ, where $\alpha \in [0;1]$ and $\gamma \in [0;1]$:

$$\begin{aligned} Q_{new}(s_t, a_t) \leftarrow & Q(s_t, a_t) + \\ & \alpha \left[r_t + \gamma \max Q(s_{t+1}, a_t) - Q(s_t, a_t) \right] \end{aligned} \tag{2.4}$$

Q-learning has a simple Q-function compared to other reinforcement learning methods and has become the basic algorithm for many other reinforcement learning algorithms [123]. However, with large state-action environments, Q-learning has problems achieving accurate learning results. Within the field of power electronics, Q-learning has already shown promising results. The algorithm has been used to balance the SOC in a battery pack [129], increase the efficiency in battery energy control [130], and utilized to reduce the loss of DC-DC converters [131].

Feedforward Neural Network (FNN)

Among Artificial Neuronal Networks, FNN is the most straightforward architecture. Also, FNN is an architecture widely applied to estimate the states of Li-ion batteries [132]. Cui et al. [133] highlight the simple structure, adaptability to time-varying characteristics, and the low computational complexity during the prediction phase as positive aspects of FNN. It composes an input, multiple hidden, and output layer [134]. Each layer is entirely linked to the next layer. The link allows data to flow from the input node to the hidden until the output layer nodes. The connection between neuron i in the current layer and neurons j in the preceding layer is specified as:

$$o_j = \sigma\left(\sum_{i=1}^{N}(w_{j,i}x_i + b_{j,i})\right) \tag{2.5}$$

where N describes the number of inputs, x_i the input value, $w_{j,i}$ the weight of the input i in the neuron j, $b_{j,i}$ the bias, and $\sigma(x)$ the activation function. Fig. 2.18 illustrates the architecture of a FNN with two hidden layers and the structure of a single neuron.

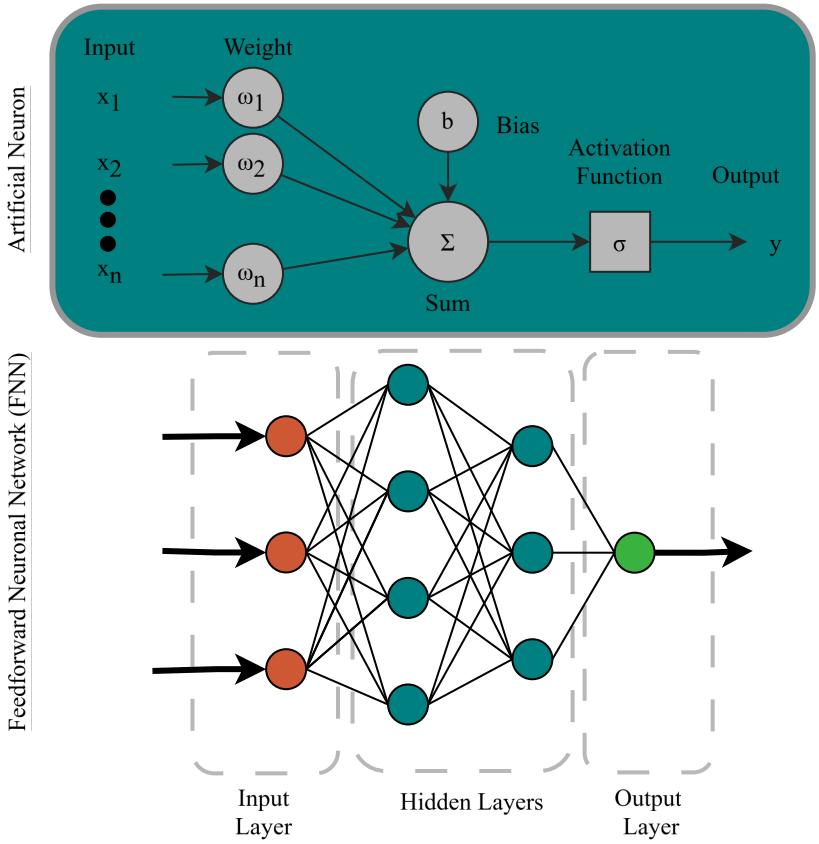


Fig. 2.18. Feedforward Neural Network (FNN) Architecture of the network with its neurons connected and a single neuron [28]

Kolmogorov–Arnold Networks (KAN)

Unlike traditional Multi-Layer Perceptron (MLP) [135] and FNN where the activation functions are fixed during training and weights are optimized, KAN [136] learn the activation function for each node. This fundamental difference extends beyond the type of function; KANs employ learnable, single-variable spline functions, in contrast to the linear weights of MLPs.

For high-precision function learning, a neural network must effectively capture individual functions and their compositional structure. Liu et al. [136] propose that the inherent flexibility of spline functions within the MLP framework allows KANs to achieve this proficiency. Figure 2.19 illustrates the structure of a KAN and its learnable activation functions.

Previous studies [137], [138] have identified KAN as highly suitable for time-series data with a high interdependence. These studies demonstrate the network's high adaptability and accuracy in predicting time-series data. Furthermore, compared to similar algorithms,

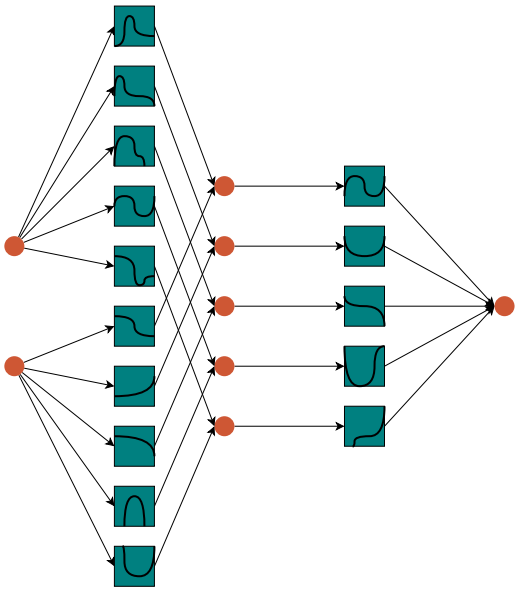


Fig. 2.19. Structure of a Kolmogorov–Arnold Networks (KAN) [30], [136]

such as MLP and LSTM, used in time series data estimation and prediction applications KAN uses fewer parameters and still achieves an accuracy that is higher or on par with other models.

The design of KANs is inspired by the Kolmogorov-Arnold representation theorem [139]. This theorem states that any continuous function on a unit n-dimensional matrix can be expressed as a sum of continuous functions of single variables composed in a specific way:

$$f(x) = f(x_1, \cdots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right) \tag{2.6}$$

where $\phi_{q,p} : [0,1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$.

In a KAN, the activation value of a neuron in layer $(l+1)$ is calculated by summing the post-activations from all incoming connections. Here, l represents the layer number, j the specific neuron in layer $(l+1)$, and i denotes a neuron in layer l [136]:

$$x_{l+1,j} = \sum_{i=1}^{n_l} \phi_{l,j,i}(x_{l,i}), \qquad j = 1, \cdots, n_{l+1}. \tag{2.7}$$

In matrix form, defined as:

$$\mathbf{x}_{l+1} = \underbrace{\begin{pmatrix} \phi_{l,1,1}(\cdot) & \cdots & \phi_{l,1,n_l}(\cdot) \\ \vdots & \ddots & \vdots \\ \phi_{l,n_{l+1},1}(\cdot) & \cdots & \phi_{l,n_{l+1},n_l}(\cdot) \end{pmatrix}}_{\Phi_l} \mathbf{x}_l \tag{2.8}$$

where Φ_l is the function matrix of the lth KAN layer. A KAN network with L layers and a given input vector x is defind as:

$$\mathrm{KAN}(x) = (\Phi_{L-1} \circ \cdots \circ \Phi_1 \circ \Phi_0)\, x \tag{2.9}$$

The notation in Eq. (2.9) represents the sequential application of the function matrices from the last layer to the first layer on the input vector x.

Long Short-Term Memory (LSTM)

LSTM [110] is a sub-type of Recurrent Neuronal Network (RNN). These networks contain bidirectional cells. The output of a RNN cell influences the subsequent input of the cell. Subsequent influence enables the prediction and processing of time series data. The LSTM algorithm addresses the limitation of RNN by adding a memory function, which makes them suitable for long-term sequence problems and time series data.

The memory unit is the main component of a LSTM cell. The LSTM cell employs an input gate i, a forget gate f and an output gate o, as illustrated in Fig. 2.20a. By learning temporal relationships and capturing them in a compact state representation, LSTM networks are particularly good at detecting contextual anomalies [140]. They can handle stationary and non-stationary dynamics and short- and long-term dependencies, making them particularly effective for modelling multivariate time series and time-varying systems [141]. LSTM models have shown potential in estimating the state of batteries, such as SOH, SOC, and temperature [142].

Gated Recurrent Unit (GRU)

Another type of RNN is GRU [143], the architecture illustrated in Fig. 2.20b. As the figure illustrates, a GRU node consists of an update gate z, a reset gate r and the output gate h, and the parameter matrices and vector W_r, W_z, and W_h needed to be learned during the training. The GRU also addresses the long-term memory. However, compared to LSTM, it offers simplicity and cost-effectiveness in computational operations. GRU achieves shorter training time and outperforms LSTM networks in low-complexity sequences, while in high-complexity sequences, LSTMs perform better [144]. In the field of battery SOC estimation, the results of LSTM and GRU are comparable in accuracy [145].

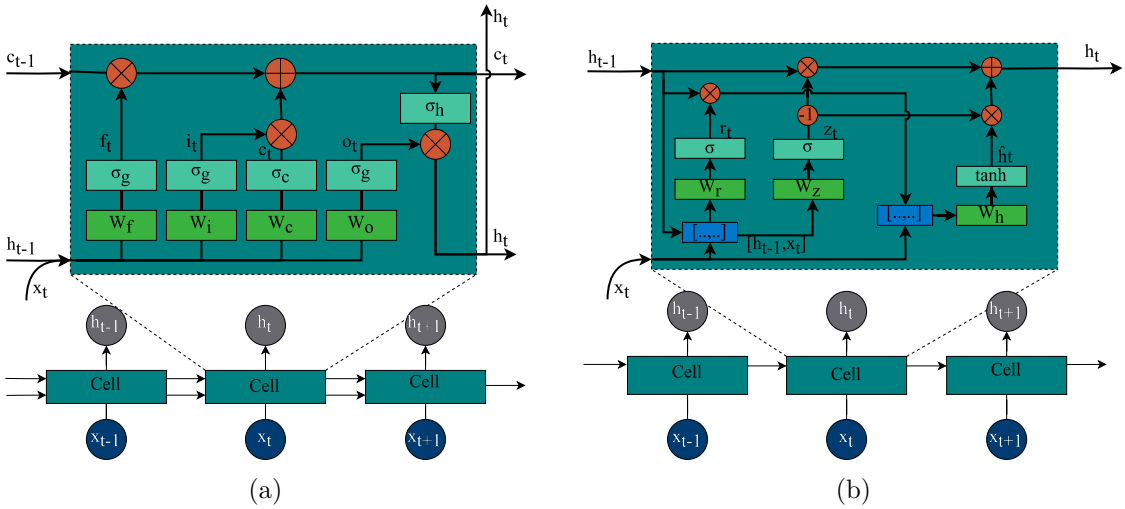


Fig. 2.20. Architecture of LSTM (a) and GRU (b) and structure of a single cell of both algorithms [28]

2.3.4 Evaluation Metrics

In the disciplines of AI, evaluating the models is essential. Within this work, prediction tasks are proposed and evaluated. In particular, Root Mean Square Error (RMSE) is a widely used metric for evaluating the performance of an algorithm. RMSE is defined as the following:

$$\text{RMSE} = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}, \tag{2.10}$$

where n is the number of predicted items, y_i is the true value of the ith item, and $\hat{y}_i$ is the predicted value. This metric is suitable to evaluate the errors of negative and positive values.

Another metric utilized in this work is the Mean Absolute Error (MAE), which is described as follows:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i| \tag{2.11}$$

According to Chicco et al. [146], MAE can be applied when there are outliers in the data, for example, because of measurement errors or corrupted data. Moreover, MAE does not heavily punish training outliers.

2.4 Digital Twin

The digital twin is a digital model that has emerged in recent years, primarily due to their important role in addressing industrial challenges [147]. Digital twin technology combines two approaches to predicting the state of a physical object: mathematical models are used to simulate the actual physical object, and data-driven methods are used, using machine learning and deep learning methods. The data-driven models are based on data generated during the product's lifespan and historical data used for model creation and tuning [148]. A digital twin is the bidirectional link between the physical entity and its digital representation. It enables automated bidirectional data exchange between the two entities in real-time [149]. Yao et al. [150] gives an overview through several definitions of a digital twin. The authors highlight, among others, the model-focused paradigm proposed by NASA [151], the digital twin for the product design process [152], the digital twin as the combination of model-focused and data-driven methods [153], and the description throughout the lifecycle of a product [154].

The different definitions indicate that a comprehensive and universally accepted definition covering all aspects of digital twins remains difficult to define. The definition of digital twin by the classification of digital twin integration as proposed by Kritzinger et al. [155] is widely used in the literature, as in this work. Kritzinger et al. differentiate between Digital Model, Digital Shadow, and Digital Twin. Fig. 2.21 represents the difference between these three integration levels. Naseri et al. [156] illustrates the three variations of integration in the case of a battery:

A Digital Model is a manually modelled digital representation without automated data exchange between physical and digital representation. A digital model of a battery is a battery model that does not contain any data interconnection with the physical object. Battery parameters have to be updated manually.

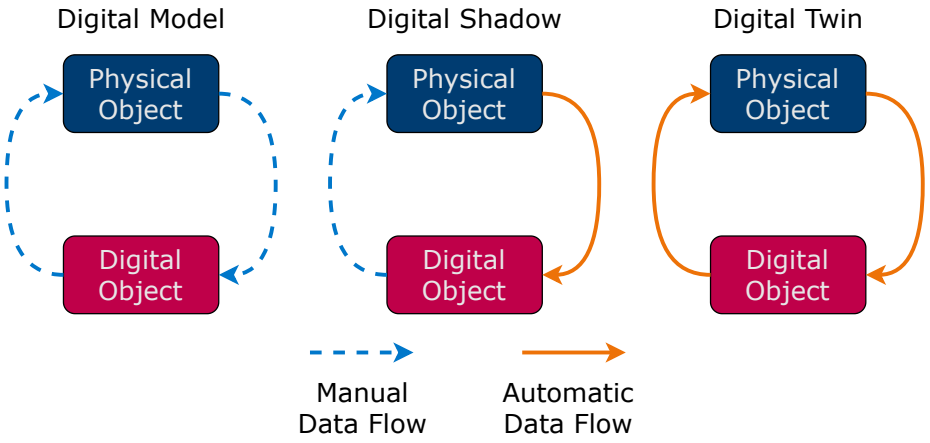


Fig. 2.21. Different types of integration: Digital Model with manual data flow only, Digital Shadow with automatic data flow from physical object to digital object, and Digital Twin with fully automatic data flow between physical and digital object [155]

A Digital Shadow enables automated one-way data exchange from a physical object to a digital representation. Sensor data can be utilised to adjust the battery model in real time.

A Digital Twin is a fully automated bidirectional data exchange between physical and digital entities. The digital twin of the battery utilises sensor data to calibrate itself, and the physical battery receives feedback from the digital twin to optimise the control and operation.

Gourisetti et al. [157] hint at several possibilities to use digital twins, such as fault prediction, performance and lifespan optimisation, predictive maintenance and product improvement, and customisation. Ibrahim et al. [158] assert that developing model-based digital twin platforms, including MATLAB/Simulink and Ansys, has demonstrated promising results. Digital twin systems are in place with technologies such as cloud computing, IoT, and AI in recent years [149], [159], [160]. These technologies are employed for real-time connectivity and data collection, processing, and analysis to observe anomalies and estimate unobservable states of the physical entities. Furthermore, it enables forecasting future states based on assumptions and past observations.

3 Reinforcement Learning for Control of Reconfigurable Batteries

This section proposes reinforcement learning control algorithms to control reconfigurable batteries and targets the third research objective and first research question of this work, as proposed in Section 1.2 on page 2. First, an overview of controlling and balancing algorithms is given. In section 3.2 on page 34, we introduce the simulation environment used following. In section 3.3, two balancing algorithms are introduced: Q-learning, in subsection 3.3.1, and Amortized Q-learning (AQL), in subsection 3.3.2 on page 45.

3.1 Review of Related Work

The behaviour of the individual batteries within a battery pack is heterogeneous [161]. The battery with the lowest capacity limits the usability and lifetime of the whole system. This requires equilibration of battery states such as SOC and SOH between the battery pack cells. Two strategies for achieving SOC balance are mainly active and passive techniques. The passive balancing approach is related to high power losses. Battery cells with a higher SOC release excess energy as heat through a shunt resistor [162]. This method is simple and cost-efficient; however, it reduces the efficiency of the battery system. Active balancing employs active equalization circuits to enable the shifting of energy from cells with higher SOC to cells with lower [163]–[166].

Machine learning can be used to implement control algorithms in power electronics. In the literature, different reinforcement learning algorithms have been proposed to optimize the system operation parameters of power electronic devices. For instance, balancing the voltage of the capacitor and the thermal stress of a Modular Multilevel Converter (MMC) [167] and the optimization of the efficiency of a dual active bridge (DAB) DC-DC converter [131], [168].

Within reconfigurable batteries, classic approach algorithms have been proposed for active SOC [169]–[171], voltage [72], and thermal balancing [172]. Previous studies have focused on improving the performance of the reconfigurable battery to allow a higher range of the car with each load of the battery, lower losses of the reconfigurable battery, or faster charging of the battery cells [173]. However, several machine-learning algorithms have been proposed to improve the control of reconfigurable batteries. The employed machine learning algorithms can be categorized into centralized and decentralized approaches.

The centralized approach involves the modules transmitting their measurements to a central/main unit, specifying each module's switching states. On the other hand, in the decentralized approach, each module operates with minimal interaction between the modules and the main unit. Bowen et al. [174] proposed a centralized approach to control a three-switch BM3 [74] system using a Deep Q-Network (DQN) reinforcement learning algorithm. This algorithm enables the balancing of SOC within a specified voltage level. This method enables the control of a Type 1 reconfigurable battery, which is used as DC source. The reconfigurable battery types are defined in Section 2.1.2 on page 14. A DQN algorithm to balance the SOC of a 64-cell Type 1 reconfigurable battery is proposed by Yang et al. [175]. Stevenson et al. [176] proposed a DQN to balance the SOC and the current of a Type 1 reconfigurable battery.

Another multiparameter balancing algorithm employing DQN have been proposed in [177]. The authors balance SOC and SOH of a 10-module half-bridge Type 1 reconfigurable battery. All previously referenced DQN algorithms utilize the centralized approach.

The decentralized approach aims to minimize the communication between the central unit and the individual modules and among the modules themselves. Each module defines its configuration. Mashayekh et al. [178] proposed an algorithm to balance the SOC of a half-bridge Type 2 reconfigurable battery. A game-theory algorithm is employed to achieve the decentralized approach, in which each module seeks to maximize its reward. The system's overall performance improves when individual modules optimize their rewards due to the careful design of the reward function for each module.

DQN based reinforcement learning algorithms have shown promising potential in controlling reconfigurable batteries. The centralized algorithms proposed in the literature are limited in controlling Type 1 reconfigurable batteries and have not been used to control a Type 2 reconfigurable battery, which is an AC source. Furthermore, the proposed algorithms do not enable the combination of the machine learning algorithms with classical approaches to control reconfigurable batteries. This thesis proposes a centralized reinforcement learning algorithm to control a Type 2 reconfigurable battery with 12 cells. The algorithm balances SOC of the cells. The proposed algorithm enables a hybrid approach combining classic and machine learning algorithms.

3.2 Simulated Environment for Reinforcement Learning

As mentioned in Section 2.3.2 on page 24, reinforcement learning depends on an environment. Before proposing and implementing reinforcement learning algorithms, it is necessary to establish an environment used for training. Therefore, we propose a physical model to simulate the environment, and we select based on the required runtime for simulation of the programming platform used for implementation. We investigated different platforms that originated from both electrical engineering and machine learning focuses. Matlab/SIMULINK is widely used in electrical engineering as a programming platform [82], [179], [180]. MATLAB and Simulink have become standard tools for electrical engineers due to their adaptability, efficiency, and robust modelling, simulation, and analysis capabilities. Engineers can quickly iterate on designs, test different scenarios, and make informed decisions. In machine learning, many programming languages and frameworks are applied [181]. In recent years, Python has been one of the main programming languages for machine learning [182]. Machine learning and deep learning libraries, such as Tensorflow, PyTorch, and sci-kit-learn, support the programming language.

A physical model of a BM3 system must be developed. The physical model is implemented and evaluated to compare the computational cost of both programming platforms, Matlab and Python. Based on the evaluation results, the programming platform is selected. We proposed physical models and the runtime evaluation of the programming platforms of single-phase BM3 system [19] and three-phases BM3 system [18].

3.2.1 Physical Model of BM3 Reconfigurable Battery

A three BM3 modules reconfigurable battery system is illustrated in Fig. 3.1. The system comprises three interconnected modules (blue) into one reconfigurable battery (red). Furthermore, the behaviour of a simple electric motor (orange) is modelled by an ohmic resistance R_{AC}, an electromagnetic coil L_{AC}, and an ideal voltage source u_{AC}. Each BM3

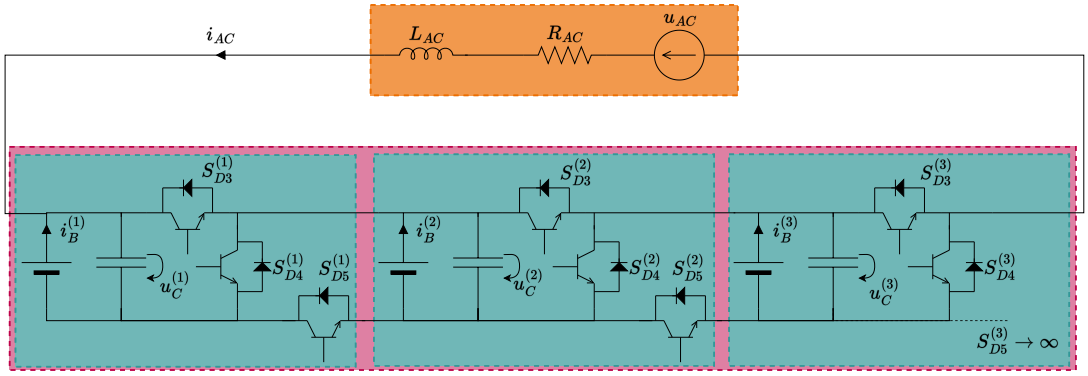


Fig. 3.1. Electric circuit of three interconnected BM3 modules (blue) to a power train (red) with the ideal voltage source, ohmic resistance and electromagnetic coil to simulate the behaviour of simple motor (orange)

module consists of a battery, a capacitor, and the three MOSFET switches $S_{D3}^{(k)}$, $S_{D3}^{(k)}$ $S_{D4}^{(k)}$, and $S_{D5}^{(k)}$, where k identifies the module. The dynamic characteristics of MOSFETs S are not considered; instead, they are represented as resistors. The on and off states of the MOSFETs are defined as fixed resistance in the on state and infinite resistance in the off state. Table 2.1 on page 17 shows the switch states of the three configurations: bypass, parallel, and serial. The i_B characterizes the battery current, and u_C defines the capacitor voltage, which is similar to the battery voltage.

The state vector $\vec{x}$ describes a system consisting of N interconnected BM3 modules as following:

$$\vec{x} = \left(i_{AC} \begin{pmatrix} SOC^{(1)} & i_B^{(1)} & u_C^{(1)} \end{pmatrix} \dots \begin{pmatrix} SOC^{(N)} & i_B^{(N)} & u_C^{(N)} \end{pmatrix}\right)^T, \tag{3.1}$$

where i_{AC} denotes the current within the reconfigurable battery, $i_B^{(k)}$ signifies the battery current, $u_C^{(k)}$ represents the voltage across the capacitor, and $SOC^{(k)}$ the SOC of the battery cell within module k.

The nonlinear dynamics of the system parameters as defined in Eq. (3.1) can be estimated using numerical integration via the forward Euler method with uniform time intervals. The system's present state $\vec{x}$ at time t is assumed to be known, and the objective is to find the state at the subsequent time step $t + \Delta t$.

The following algebraic system of $N + 1$ equations describes the values of i_{AC} and $u_C^{(k)}$ at $t + \Delta t$:

$$\begin{pmatrix} u_C^{(1)}(t+\Delta t) \\ u_C^{(2)}(t+\Delta t) \\ \vdots \\ u_C^{(N)}(t+\Delta t) \\ \hline i_{AC}(t+\Delta t) \end{pmatrix} = \overbrace{\left(\begin{array}{ccccc|c} a_{1,1} & a_{1,2} & 0 & \dots & 0 & a_{1,N+1} \\ a_{2,1} & a_{2,2} & a_{2,3} & \dots & 0 & a_{2,N+1} \\ \vdots & \ddots & \ddots & \dots & \vdots & \vdots \\ 0 & \dots & 0 & a_{N,N-1} & a_{N,N} & a_{N,N+1} \\ \hline a_{N+1,1} & \dots & \dots & a_{N+1,N-1} & a_{N+1,N} & a_{N+1,N+1} \end{array}\right)}^{\mathbf{A}^{-1}} \vec{v} \tag{3.2}$$

Following the definition of the elements $a_{n,m}$ of the matrix $\mathbf{A}$:

$$
\begin{aligned}
a_{k,k-1} &= -\frac{\Delta t}{R_{D3}^{(k-1)} C^{(k-1)}} \frac{G_{D5}^{(k-1)}}{G^{(k-1)}} \\
a_{k,k} &= 1 + \frac{\Delta t}{R_{D3}^{(k)} C^{(k)}} \frac{G_{D4}^{(k)} + G_{D5}^{(k)}}{G^{(k)}} + \\
&\underbrace{\frac{\Delta t}{C^{(k)} R_{D5}^{(k-1)}} \frac{G_{D3}^{(k-1)} + G_{D4}^{(k-1)}}{G^{(k-1)}}}_{\text{if } k \leq 1 \text{ then } 0} \\
a_{k,k+1} &= -\frac{\Delta t}{R_{D3}^{(k)} C^{(k)}} \frac{G_{D5}^{(k)}}{G^{(k)}} \\
a_{k,N+1} &= \frac{\Delta t}{C^{(k)}} \frac{G_{D5}^{(k)}}{G^{(k)}} \\
a_{N+1,k} &= \frac{\Delta t}{L_{AC}} \frac{G_{D5}^{(k)}}{G^{(k)}} \\
a_{N+1,N+1} &= 1 + \frac{\Delta t}{L_{AC}} \left(R_{AC} + \sum_{k=1}^{N} \frac{1}{G^{(k)}} \right)
\end{aligned}
\tag{3.3}
$$

The time between two timesteps is defined as Δt, and the capacity of the capacitor in module k is given as $C^{(k)}$. Electrical conductance, denoted by G and measured in Siemens, is the reciprocal of electrical resistance. The values $G_{D3/4/5}^{(k)}$ and $G^{(k)}$ are contingent upon the switch state of module k:

$$
\begin{aligned}
G_{D3/4/5}^{(k)} &= \frac{1}{R_{D3/4/5}^{(k)}} \\
G^{(k)} &= \left(G_{D3}^{(k)} + G_{D4}^{(k)} + G_{D5}^{(k)} \right)
\end{aligned}
\tag{3.4}
$$

The vector $\vec{v}$ of Eq. (3.2) is defined as:

$$
\vec{v} = (v_1 \quad v_2 \quad \dots \quad v_N \quad | \quad v_{N+1})^T \tag{3.5}
$$

$$
v_k = u_C^{(k)}(t) + \frac{\Delta t}{C^{(k)}} \left(i_{AC}(t) \frac{G_{D4}^{(k)} + G_{D5}^{(k)}}{G^{(k)}} + i_B^{(k)}(t) \right)
$$

$$
v_{N+1} = i_{AC}(t) + \frac{\Delta t}{L_{AC}} \left(\sum_{k=1}^{N-1} u_C^{(k+1)}(t) \frac{G_{D5}^{(k)}}{G^{(k)}} - u_{AC}(t) \right) \tag{3.6}
$$

To model the battery current $i_B^{(k)}$ a second-order RC Thevenim model is employed as introduced in Eq. (2.1) on page 11. Furthermore, to describe SOC of the cell, Coulomb counting is used, as introduced in Eq. (2.2) on page 12.

3.2.2 Simulation Results

Fig. 3.2 illustrates the reconfigurable battery voltage u_{conv}, resulting current i_{AC} in 3.2a, the decreasing SOC over time in 3.2b, the simulated battery currents i_b^k of the batteries in 3.2d, and the capacitor voltage u_c^k in 3.2c. The battery model parameters of the KeepPower

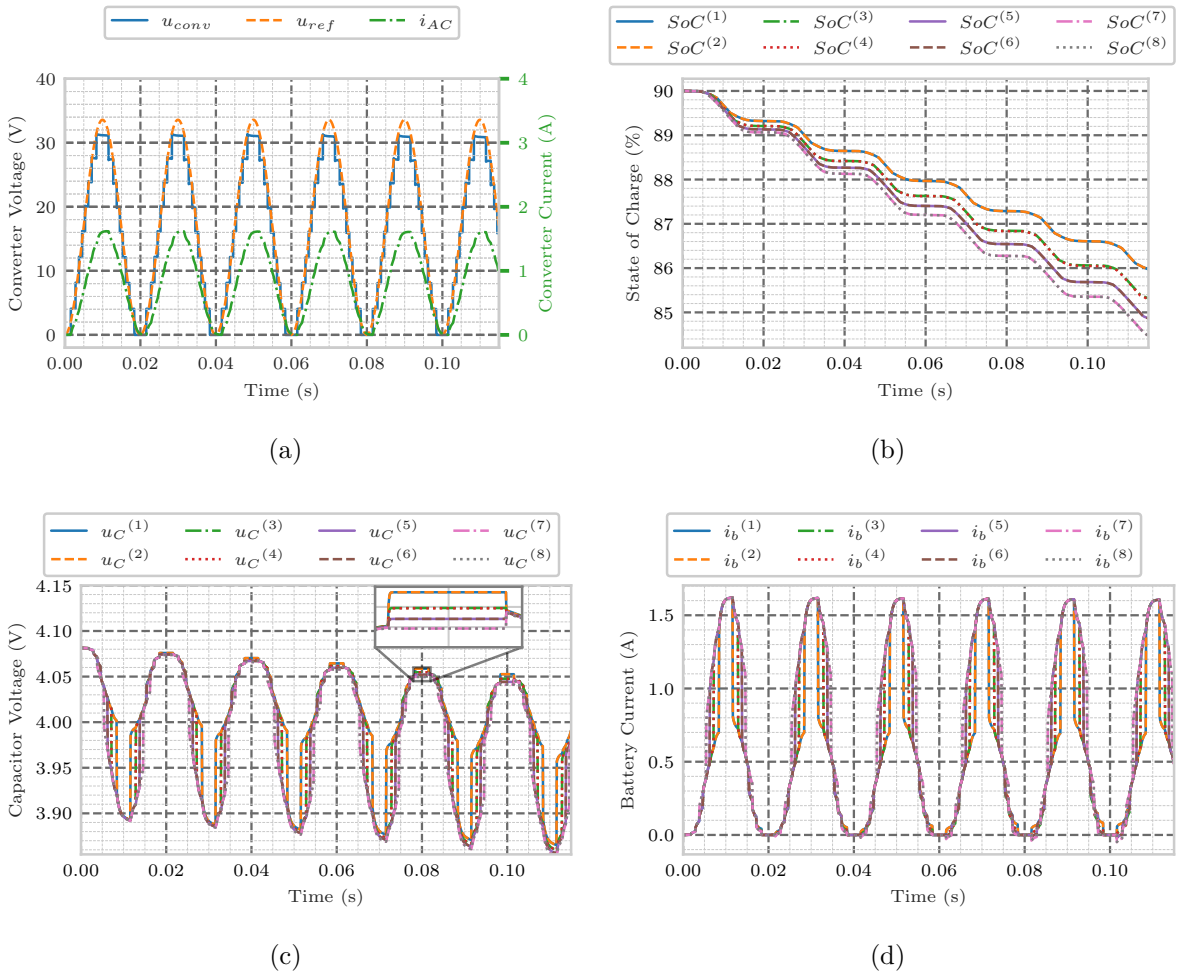


Fig. 3.2. Simulation of System with eight BM3 modules, over t =0.12 s, with step size Δt =10 µs; reconfigurable battery voltage and current (a), decreasing SOC (b), capacitor voltage of the module capacitors (c), and battery current (d)

P1834J evaluated by Estaller et al. [183] with a 1.5 A s capacity are utilized. The figure shows a 0.12 s simulation with eight modules with a step size of $\Delta t = 10$ µs.

A Type 2 reconfigurable battery approximates the sinusoidal voltage control signal step-by-step. The simulation visualizes nine levels caused by the eight BM3 modules. The reference voltage signal u_{ref} and the reconfigurable battery voltage u_{conv} are illustrated in orange and blue in Fig. 3.2a. The smooth sinusoidal current i_{AC} is illustrated in green. Due to the inductance L_{AC} within the system, the current delays the voltage by 34.6°.

The change of SOC is illustrated in Fig. 3.2b. The absence of an implemented balancing algorithm causes an increasing difference between the SOCs of the different cells. During the discharge time of 0.1 s the cells 1, and 2 demonstrate a reduction from 90 % down to 86 % SOC. On the other hand, cells 7 and 8 indicate a higher discharge, which ends in a SOC of 84.2 %, which causes the unbalanced voltage across the modules. This behaviour is illustrated in Fig. 3.2c, which shows the capacitor voltage $u_C^{(k)}$ of each module. The zoom-in around 0.1 s illustrates the mentioned difference in voltage among cells caused by the imbalanced SOC. Furthermore, a reduction in the minimum and maximum voltages within each sinus period can be observed.

Fig. 3.2d illustrates each system module exhibits a current distinct from the others during

a sinus period. This variation arises from the differing connection types of the modules at various operating times. The peak current occurs at the stage corresponding to the highest possible voltage of 31 V. All modules are connected in series to achieve this voltage level, resulting in an even current over all modules.

3.2.3 Benchmarking Implementations

The programming platform is selected based on the implementation's computational cost, which is defined as the needed runtime. Therefore, the physical model was implemented using the programming language MATLAB and three Python variations: the library NumPy [184], the Python just-in-time compiler Numba [185], and CuPy Graphical Processing Unit (GPU) compiler [186]. The just-in-time compiler Numba focuses on scientific computing, and array-oriented computing [185]. Numba allows developers to implement Python code that can reach the execution time of C or FORTRAN programmes [185]. The programming language C has been evaluated as the most energy and time-efficient programming language within several programming benchmarking problems among 27 well-known software languages [187]. The CuPy library allows the execution on GPU while using NumPy syntax. Furthermore, NVIDIA CUDA kernels can be implemented inside Python code [186]. Parallelization of matrix operations on GPUs allows high-performance increase.

The following hardware and operating system are employed to perform the experiments for validation of the runtime of the four implementations: a DELL PowerEdge R750xa server with 512 GB RAM, an Intel Xeon Gold 6338 CPU, an NVIDIA A40 GPU, and the Linux distribution Ubuntu version 20.04 LTS[1] as operating system, Python 3.10[2], NumPy 1.21.1[3], numba 0.55.1[4], CuPy 10.2.0[5], and MATLAB R2022b[6]. The experimental setup was to measure the runtime of a 1 s simulation with a 10 µs time step size of a system with a variable number of modules.

Fig. 3.3 shows the results of experiments with 10 up to 200 modules for all implementations and up to 400 modules for CuPy and Numba. In addition, Table 3.1 shows the computational cost for ten, 200, and 400 modules. The table also includes the computational cost of the literature proposed MATLAB/Simulink simulation model of a BM3 reconfigurable battery [169]. The NumPy implementation (green) shows the highest increase in computational cost as the number of modules increases. To simulate 1 s run time of 200 modules within this implementation, 163.52 s are required. The CuPy implementation illustrated in orange shows the least increase in overall implementations by increasing the number of modules to simulate up to 200 modules. MATLAB (red) and Numba (blue) perform the best out of all the compared solutions. In the case of Numba, a runtime time of 49.43 s for a 200-module reconfigurable battery.

The results of 200 modules do give the impression that CuPy will outperform Numba, the implementation with the lowest computational cost over all implementations up to 200 modules, around 270 modules. Experiments were carried out to validate this expectation with up to 400 modules, and the CuPy and the Numba runtime were compared to prove this impression.

In the CuPy implementation, a noticeable increase in runtime can be observed within the hardware setup around 220 to 240 simulated modules.Between 240 and 400 modules, a parallel increase in the required runtime of the Numba and Cupy implementation can be

[1] `https://releases.ubuntu.com/focal/`
[2] `https://www.python.org/downloads/release/python-3100/`
[3] `https://pypi.org/project/numpy/1.21.1/`
[4] `https://pypi.org/project/numba/0.55.1/`
[5] `https://pypi.org/project/cupy/10.2.0/`
[6] `https://www.mathworks.com/products/new_products/release2022b.html`

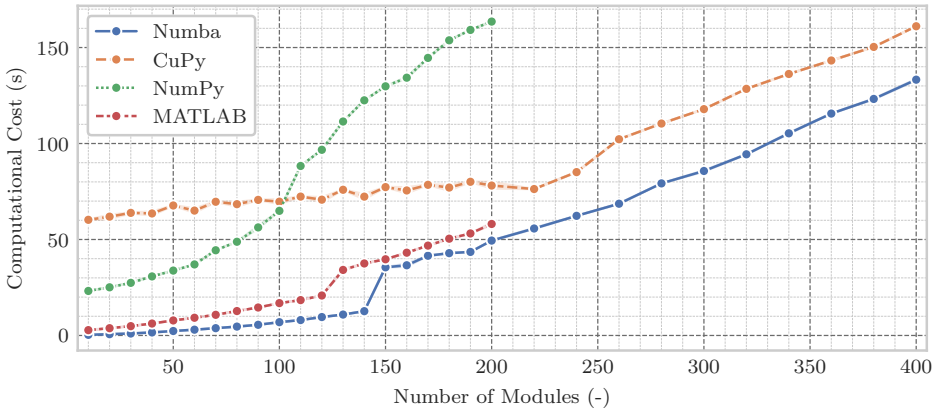


Fig. 3.3. Computational cost per implementation by number of simulated modules up to 200 modules for NumPy and MATLAB and up to 400 modules for Numba and CuPy; Numba outperforms other implementations

Table 3.1: Computational cost per implementation for ten, 200, and 400 modules in seconds

	Number of Modules		
Implementation	*10*	*200*	*400*
Python Numba	0.33 ± 0.00	49.43 ± 0.09	133.25 ± 0.54
Python CuPy	60.25 ± 0.26	78.11 ± 3.80	161.08 ± 0.03
Python NumPy	23.20 ± 0.03	163.52 ± 0.06	–
MATLAB	2.78 ± 0.28	58.10 ± 1.00	–
MATLAB/Simulink [169]	42.10 ± 0.73	–	–

observed. Given the utilized hardware, the findings do not indicate any specific number of modules where the GPU implementation exceeds the best-performing CPU implementations.

Finally, the observed runtime is compared with the literature proposed MATLAB/Simulink simulation model of a BM3 reconfigurable battery [169]. With the simulation model, Mashayekh et al. [169] simulated an up-time of 1 s of a reconfigurable battery with ten modules and the same step size of 10 µs, which is the same uptime and step size as used within our evaluation. The MATLAB/Simulink [169] model needed a runtime of 42.10 s ± 0.73 s for simulation. In our evaluation, illustrated in Fig. 3.3, the implementation using Numba needed 0.33 s. The simulation results archived do not show significant differences. This comparison shows that the Python implementation of the physical model improves computational performance by factor 130 compared to the MATLAB/Simulink proposed in [169].

Based on these results, the Python Numba implementation will be employed as an environment to implement a reinforcement learning algorithm to control a Type 2 reconfigurable battery.

3.3 Reinforcement Learning for SOC Balancing

After introducing the physical model of a BM3 reconfigurable battery, the model will be employed as an environment to train a reinforcement learning model. This section presents and evaluates two reinforcement learning models to balance SOC of batteries within a Type

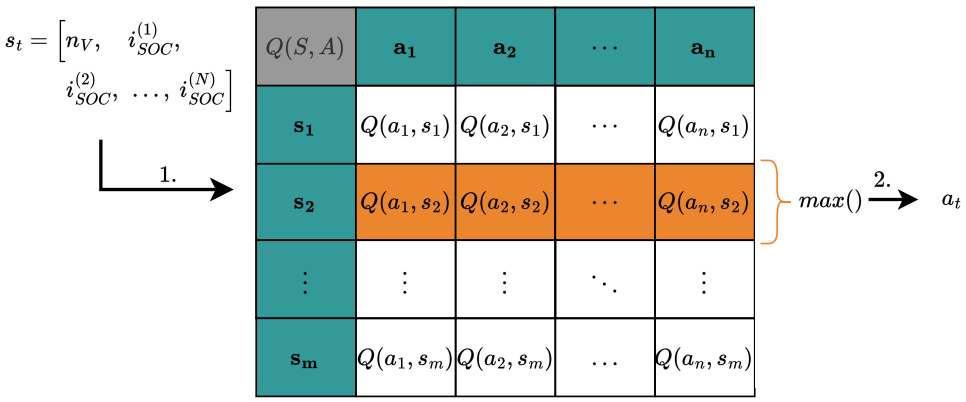


Fig. 3.4. Q-Table for reconfigurable battery control; 1. select the Q-values row according to the state; 2. select action determined on max-function

2 reconfigurable battery. We proposed the following Q-learning [20] and AQL [21] models in our published work. Insights into the Q-learning model's feasibility and limitations formed the basis to develop the AQL model.

3.3.1 Q-learning

As mentioned in Section 2.3.3 on page 25, the Q-learning algorithm stores the learned experiences in the Q-table, a table of Q-values [131]. The optimal action is chosen by the Q-table's Q-values related to the environment's state. The learning process of a Q-learning model has been mentioned in Section 2.3.3 on page 25. Fig. 3.4 illustrates the operating algorithm. As the first step, based on the state s_t, the row in the Q-table $Q(S, A)$ is selected. In step two, the action is chosen by the max-function. The chosen action is the action a of the maximum Q-value $Q(a, s)$. The proposed Q-learning algorithm addresses the optimal reconfigurable battery operation strategy for SOC balancing. The state and action space and the reward function implemented in the Q-learning are introduced at first.

State Space

Jang et al. [123] define the state space S as all observable states of the environment. The state space of the proposed Q-learning consists of the required voltage of the system V_{conv} and the SOCs of the reconfigurable battery modules. To employ Q-learning, the state space has to be discrete. Continuous action and state spaces must be approximated as discrete spaces. For this purpose, the required converter voltage V_{conv} is represented as n_V, a multiple of the open-circuit voltage of the cell V_{OC}. Furthermore, the SOC is discretized as the index $i^{(k)}_{SOC}$ of the module k sorted by the SOC of the modules. This results in the following state space definition:

$$S = \left[n_V, \quad i^{(1)}_{SOC}, \quad i^{(2)}_{SOC}, \quad \ldots, \quad i^{(N)}_{SOC}\right], \tag{3.7}$$

Fig. 3.5 shows three batteries, their SOCs, the required voltage V_{conv}, and the open-circuit voltage V_{OC} of the used battery type. The state S in this case would be [2, 2, 3, 1], caused by the following:

- 8.4V is twice 4.2V, therefore $n_V = 2$

- Module 3 has the highest SOC, than Module 1 and the lowest SOC can be observed for Module 2, therefore the indexes $i_{SOC}^{(k)}$ are $2, 3, 1$.

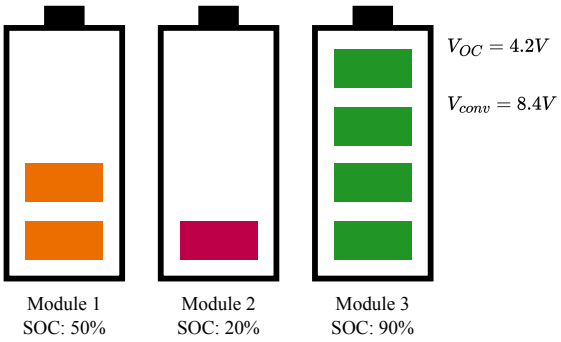


Fig. 3.5. Example for a state S; Two modules required to achieve voltage level, therefore $n_V = 2$; Indexes of sorted SOCs are 2,3,1; Causing $S = [2, \quad 2, \quad 3, \quad 1]$

Time Dependent State Space

The resulting state s_{t+1} at the time step $t + 1$ of a MDP depends only on the state s_t and the action a_t of time step t [123]. It has to be evaluated if a modulation of the state of reconfigurable battery without time dependency enables a correct SOC balancing. To compare a time-independent state with a time-dependent state, the state space S is modified by introducing a time dependency as follows:

$$\begin{aligned} S = \Big[n^V, \quad & i_{SOC}^{(1),t}, \quad i_{SOC}^{(2),t}, \quad \ldots, \quad i_{SOC}^{(N),t}, \\ & i_{SOC}^{(1),t-1}, \quad i_{SOC}^{(2),t-1}, \quad \ldots, \quad i_{SOC}^{(N),t-1}, \\ & i_{SOC}^{(1),t-T}, \ldots, \quad i_{SOC}^{(N),t-T} \Big], \end{aligned} \tag{3.8}$$

where $i_{SOC}^{(k),t}$ represents the index of the k^{th} module, ordered according to the SOC of the module's battery at the time step t, and T denotes the number of preceding time steps encompassed in the state space S.

Reward Function

Based on the environment state s_t and the performed action a_t, the reward r_t is given. The reward function $r(s_t, a_t)$ is defined as following:

$$\begin{aligned} r(s_t, a_t) = & \alpha \cdot \Big[\big[\max\left(SOC(t+1)\right) - \min\left(SOC(t+1)\right) \big] - \\ & \big[\max\left(SOC(t) - \min(SOC(t)\right) \big] \Big], \end{aligned} \tag{3.9}$$

where $SOC(t + 1)$ is the list of all cells SOC at state s_{t+1}, and α is a weight value to increase the effect of the reward during training.

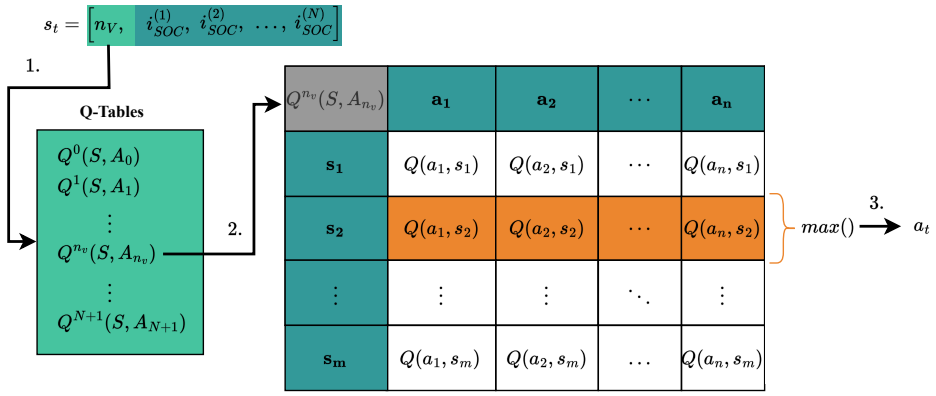


Fig. 3.6. Multi-Q-Table for reconfigurable battery control; 1. select Q-Table determined on voltage level n_V; 2. select Q-values row based on the state; 3. select action according to max-function

Action Space

There are three main action space categories within reinforcement learning: discrete action space, continuous action space, and discrete-continuous hybrid action space [188]. Counting the number of possible actions is possible for discrete action spaces. In a continuous action space, the number of actions can not be counted, but their range can be described; for example, a robot's arm can be moved between 0 and 360 degrees. In discrete-continuous hybrid action spaces, the algorithm must generate discrete actions and the continuous parameters associated with those actions. The number of possible configurations of the reconfigurable battery is limited, leading to discrete action spaces.

The set of possible actions A encompasses all allowable actions a within the state s_t at time step t [123]. An action a is defined as follows:

$$A = \left\{ M^{(1)}, \quad M^{(2)}, \quad \ldots, \quad M^{(N)} \right\} \tag{3.10}$$

where n represents the number of modules in the reconfigurable battery, and the switch state M for module k is given by its configuration:

$$M^{(k)} = \begin{cases} 0 & \text{if Bypass} \\ 1 & \text{if Serial} \\ 2 & \text{if Parallel} \end{cases} \tag{3.11}$$

Multi-Q-Table Approach

A multi-Q-table method is suggested to minimize the required memory usage. For each voltage level n_V, an individual Q-table $Q^{n_V}(S, A_{n_V})$ is utilized. This approach helps decrease the training time and complexity of the Q-table. Distinct action spaces are assigned to each voltage level, including only configurations that achieve the desired voltage level. This forms a policy $Q(S, A)$ defined as follows:

$$Q(S, A) = \left[Q^0(S, A_0), \quad Q^1(S, A_1), \quad \ldots, \quad Q^{N+1}(S, A_{N+1}) \right] \tag{3.12}$$

where Q^{n_V} is the Q-table, and A_{n_V} the action space used at voltage level n_V.

Fig. 3.6 illustrates the three steps of operation of the multi-Q-table algorithm. Within the first step, the Q-table is selected based on the voltage level n_V. The Q-table $Q^{n_V}(S, A_{n_V})$ is selected consisting of the state space S and the action space A_{n_V} dependent on the voltage level n_V. The steps two and three are the same as the steps one and two in Fig. 3.4 on page 40. According to the state s the row in the Q-table $Q^{n_V}(S, A_{n_V})$ is chosen. As the third step, the selection of an action a is determined by the highest Q-value $Q(a, s)$.

The Q-learning training uses the Bellman optimal equation and ϵ-greedy policy [123] and is implemented as shown in Algorithm 1.

Algorithm 1 The proposed Q-learning Algorithm [20] ©IEEE 2023

procedure Q-LEARNING PROCESS
  Initialize $Q(S, A)$
  $s_t, n_V \leftarrow$ environment.initialState()
  while epoch $<$ epochs **do**
    done $\leftarrow$ *true*
    epoch $\leftarrow$ epoch $+ 1$
    while not done **do**
      if random() $< \epsilon$ **then**
        $a_t \leftarrow$ chooseRandomAction($Q^{n_V}(s)$)
      else
        $a_t \leftarrow \max(Q^{n_V}(s))$
      end if
      done, s_{t+1}, n_V, $r_t \leftarrow$ environment.step(a_t)
      Use Equation (2.4) to calculate $Q^{n_V,new}$
      $s_t \leftarrow s_{t+1}$
    end while
  end while
end procedure

Baseline Approach: Q-learning for Reconfigurable Battery Control

For validation, the simulation of a five-module BM3 reconfigurable battery is used within the simulated environment. The in Eq. (3.8) introduced state space S includes a time dependency. To evaluate the need for time-dependent state space, the convergence of the Q-values during training of Q-tables with time dependency, as in Eq. (3.8), and without, as in Eq. (3.7), are compared. It could be observed that a time dependency of $T = 2$ results in Q-values convergence, which leads to accurate SOC balancing. Without time dependency, such as in Eq. (3.7), balancing does not take place. Increasing the time steps to $T = 3$ or more leads to a longer training time and an increase in the required memory. Also, the method with higher time steps has not shown a significantly different result in the required time to archive the balanced state.

Fig. 3.7a displays the sinusoidal voltage reference signal u_{ideal}. The reconfigurable battery approximates this signal, as the figure illustrates as u_{conv}. Six distinct voltage levels are identifiable, corresponding to the number of connected modules, $N = 5$, plus an additional level at 0 V, where all modules are in bypass configuration. Fig. 3.7b shows that all cells reach balanced SOC after 65 ms. Battery cell 3 exhibited a below-average discharge rate between 35 and 43 ms, followed by a more rapid discharge, ultimately contributing to system equilibrium at 65 ms. The above-described faster discharge of the battery of module three is also shown in Fig. 3.7c. Here, it can be seen for the generation of the 4 V voltage level

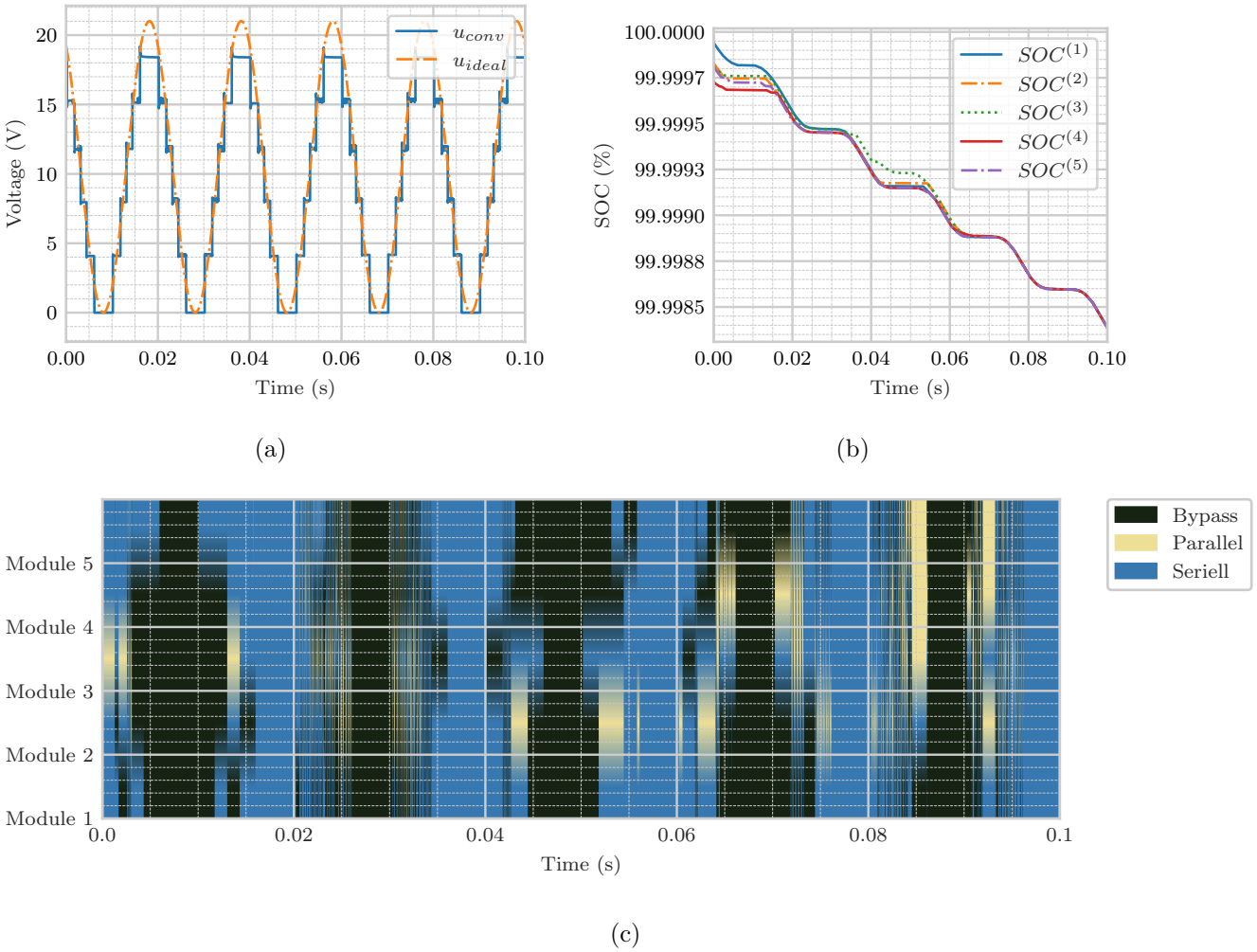


Fig. 3.7. Q-learning balancing with system of $N = 6$ modules, simulation time of t=0.1 second; reconfigurable battery voltage u_{conv} in blue and ideal voltage u_{ideal} in orange (a), decreasing SOC over time with balanced state after 65 ms (b) and configurations per module over time (c) [20] ©IEEE 2023

between 44 - 52 ms only module three is switched in series. Due to this configuration, module three experiences a higher discharge than the other cells switched into bypass.

The results clearly demonstrate that the proposed Q-learning method effectively controls a reconfigurable battery system and balances its batteries' state of charge.

Advantages The Q-learning method is a reinforcement learning algorithm with low computational complexity during use. The Q-table can be used as an efficient lookup table for practical implementation. A hashmap implementation of the lookup table enables direct access and enables a computational complexity of O(1). The Q-learning algorithm provides a straightforward assessment of each decision taken by the model, giving it an edge over neural networks.

Drawbacks The discretization of the state space and the introduction of a time-dependent state space substantially increase the required memory during training. The number of states can be described by $(n!)^{1+T}$, where n is the number of modules, and T is the number of previous time steps added in the state. Due to memory constraints, the Q-table size grows significantly, making it infeasible to practically implement with

more than seven modules using the existing hardware.

3.3.2 Scaling Reinforcement Learning: Amortized Q-learning (AQL) for Reconfigurable Battery Control

As demonstrated in the prior section, applying Q-learning creates significant challenges as the number of actions and states increases. With a rising number of modules within the reconfigurable battery, the number of actions and possible states increases significantly. The proposal of DQN addresses the limitations of Q-learning by using a deep neural network as an improvement compared to the Q-table [113], [189]. The neural network approximates the optimal action-value function $Q(s, a)$. The approach of DQN enables large or continuous state spaces and computes for each action a Q-value in a single forward pass. Thus, DQN solves the limitation in increasing state spaces but is still limited within the action space size.

The standard actor-critic approach is an actor function $\pi_\Theta : S \rightarrow A$, where π_Θ is the leaned policy. The function π_Θ scales approximately linear to the number of actions [190]. Furthermore, the actor-based approach of Q-learning does not generalize over the action space as naturally as value-based approaches [190]. It cannot extend to previously unseen actions. This limitation increases the required training time with the increasing action space.

Van de Wiele et al. [191] proposed AQL to replace the time-consuming Q-value maximation over all possible actions with a maximization over a small subset of actions sampled from a learned proposal distribution. This approach is inspired by amortized inference [192] techniques. AQL is adaptable to discrete, continuous, or hybrid action spaces, making it suitable for diverse settings. This is particularly useful in scenarios where the action space is too enormous for traditional Q-learning methods. Furthermore, AQL has shown great potential in combination with classical planning methods in pre-limiting the action space [124]. Because of this reason, AQL is chosen as a methodology to control a reconfigurable battery. The implemented approach will be described in the following section.

Amortized Q-Learning Methodology

In Q-learning, the Q-value of each action a within the action space A has to be computed for the state s. AQL enables continuous state spaces, which causes a redefining of the state space, as previously defined in Eq. (3.8). The continuous state s at a specific time step is defined as follows:

$$s = \left[SOC_{norm}^{(1)}, \quad SOC_{norm}^{(2)}, \quad \ldots, \quad SOC_{norm}^{(N)} \right], \tag{3.13}$$

where $SOC_{norm}^{(n)}$ is the normalized SOC of the nth module within a system of N modules. The normalization takes place by a Min-Max-Scaler, which is defined as the following:

$$SOC_{norm}^{(n)} = \frac{SOC^{(n)} - \min(SOC)}{\max(SOC) - \min(SOC)} \tag{3.14}$$

AQL computes the Q-value based on the state s and an action a. A neural network is employed as an approximation function. To select the action a with the highest Q-value, a batch of Q-values has to be calculated. Therefore, the network's input includes the state $\mathbf{s}_t$ and the action space A_t. The action $\mathbf{a}_t$ with the highest Q-value is identified by the function $argmax$. Each element of the input batch is defined as $[\mathbf{s}_t, \mathbf{a}_t^i]$, where $\mathbf{a}_t^i \in A_t$. The action space A_t is defined as the following described:

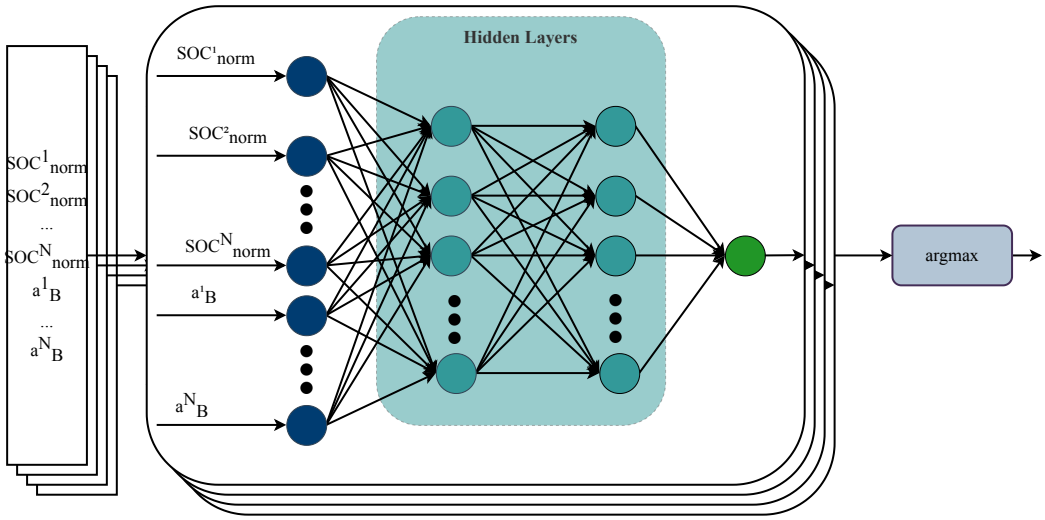


Fig. 3.8. Schematic of proposed and implemented Amortized Q-learning (AQL) neural network with two hidden layers and one output neuron and argmax function to identify best action in performed batch [21]

The action space A specifies all possible actions of the reconfigurable battery and is defined in Equation (3.10) on page 42.

The subset A_{n_V} defines all possible configurations to generate the voltage level n_V at the time step t.

The set A_C contains configurations that should be excluded from the action space A because the switches are inoperative or batteries are overheating. The possibility of excluding configurations enables safety functionality and the combination with classic approach algorithms.

This results in the following definition for A_t:

$$A_t = A_{n_V} - A_C, \tag{3.15}$$

Fig. 3.8 shows the architecture of the proposed neural network. The figure illustrates the batch calculation of Q-values for the state containing the normalized SOC of N modules and the action a containing the configurations. The neural network consists of a FNN architecture, as described in Section 2.3.3 on page 26, with multiple hidden layers and one output neuron.

The algorithm 2, proposed in [21], outlines the training algorithm that has been put into practice.

Model Implementation and Training

Python 3.8[7] and the machine learning library PyTorch 2.0.1[8] are utilized to implement the neural network. The proposed AQL method is evaluated over a 12-module reconfigurable battery. The architecture of the implemented FNN is shown in Table 3.2. It consists of

[7] `https://www.python.org/downloads/release/python-380/`
[8] `https://pypi.org/project/torch/2.0.1/`

Algorithm 2 AQL training [21]

procedure AQL TRAINING(Network parameters θ^Q, exploration probability ϵ, epochs E, epoch length T)
  Initialize local replay buffer R
  for $e \rightarrow 1 \cdots E$ **do**
    for $t \rightarrow 1 \cdots T$ **do**
      Observe state $\mathbf{s}_t$
      if random() $\leq \epsilon$ **then**
        $a_t := a_t^R \in A_t$ ▷ Select a_t at random
      else
        $\mathbf{a}_t := \arg\max_{\mathbf{a} \in A_t} Q(\mathbf{s}_t, \mathbf{a}; \theta^Q)$
      end if
      $r_t, \mathbf{s}_{t+1} \leftarrow \text{environmentStep}(\mathbf{s}_t, \mathbf{a}_t)$
      $R_t \leftarrow (\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})$
    end for
    Train θ^Q based on R as defined (2.4)
    Reset *environment* to random state
  end for
end procedure

an input layer with 24 neurons, two neurons per module, three hidden layers, employing a rectified linear unit (ReLU) as an activation function, and the output layer with one neuron.

Two models are trained to evaluate the proposed methodology. One model is trained to control a reconfigurable battery based on a half-bridge topology. The other one is to control a reconfigurable battery employing BM3 as topology. Both topologies are described in Section 2.1.2 on page 14. The reward for each epoch during the training is illustrated in Fig. 3.9. The simulation proposed in Section 3.2 on page 34 is employed as a training environment. Caused by the more possible configurations and the resulting larger action space within the BM3 topology, more epochs are required during training than the half-bridge topology.

Experimental Evaluation

Following the proposed AQL method is evaluated within the balancing of SOC in reconfigurable batteries using the topologies BM3 and half-bridge. Therefore, two different settings were used for evaluation and discussion.

BM3: Comparative analysis of the proposed AQL algorithm with the balancing algorithm suggested by Mashayekh et al. [169] employing a 12-module setup in a simulation environment.

Half-Bridge: Comparative analysis of the proposed AQL algorithm with the balancing algorithm suggested by Zheng et al. [193] employing a 12-module physical hardware.

Table 3.3 shows the required time to balance the SOC. The proposed AQL algorithm outperforms the algorithm proposed in Mashayekh et al. [169] employing the BM3 topology. For the Half-Bridge topology, the algorithm by Zheng et al. [193] shows less required time to balance in the results. The results and the employed settings for both topologies are described below.

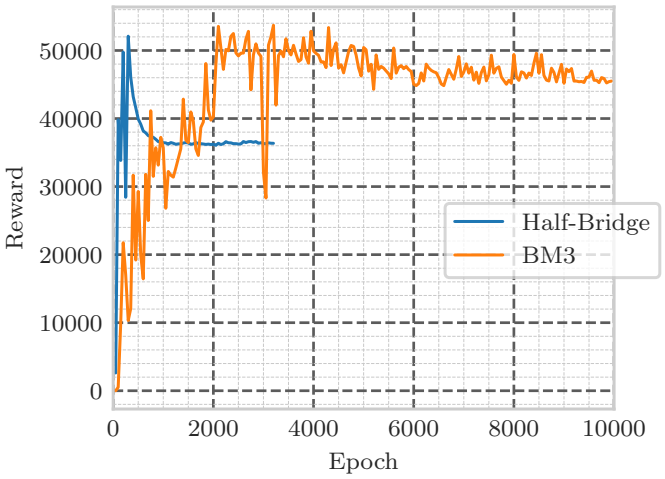


Fig. 3.9. Reward over the training of the model for Half-Bridge (blue) and BM3 (orange) controlling; Half-Bridge reaches stable state faster than BM3 [21].

Layers	*Model*
Input Layer	Dense(24)
Hidden Layer 1	Dense(128)
	ReLU
	Dropout(0.1)
Hidden Layer 2	Dense(64)
	ReLU
	Dropout(0.1)
Hidden Layer 3	Dense(32)
	ReLU
	Dropout(0.1)
Output Layer	Dense(1)

Table 3.2. Architecture of the model [21]

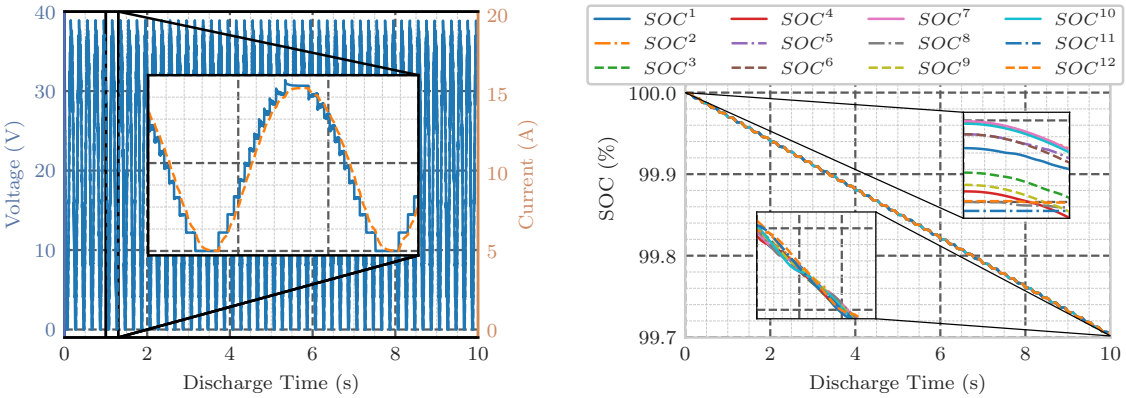
Table 3.3: Required time to balance SOC within four settings

Topology	Proposed AQL	Mashayekh et al. [169]	Zheng et al. [193]
BM3	0.061 ± 0.029	0.275 ± 0.207	-
Half-Bridge	70.9 ± 11.9	-	58.9 ± 10.0

BM3: Simulation Evaluation The three switching states of the BM3 topology increase the complexity of controlling and balancing with the rising number of modules. At five modules, the number of possible configurations is 534. This is the size of the system evaluated in Section 3.3.1 on page 40. In the case of a 12-module system, the number of possible configurations rises to 975 325. In contrast, a half-bridge reconfigurable battery enables 32 configurations with five modules and 4096 with 12 modules. The simulation environment proposed previously was utilized to evaluate the AQL methodology.

In Fig. 3.10 a discharge time of 10 s with a step size of Δt 0.1 ms is illustrated. During this discharge time, the batteries show a discharge of 0.3 %. Fig. 3.10a shows the current (orange) and voltage (blue). The incremental formation of the resulting voltage by the Type 2 reconfigurable battery is observable. The zoom-in at the beginning and end of the discharge time in Fig. 3.10b illustrates the achieved balancing of the cells. There is a clear indication that the balanced state is reached and maintained.

The results that were previously shown illustrate the general applicability of the proposed method. The results of the methods are compared with the proactive balancing algorithm proposed by Mashayekh et al. [169] to evaluate the method's efficiency. Therefore, the balancing experiment was repeated with 50 random initial SOCs. The initial SOC of each cell is in between 99.999 95 % and 100 %. That is a difference of $5e{-}5\%$. This experiment defines the balanced state by a maximum difference of $5e{-}4\%$ between all cells. The needed time to balance each experiment's SOC is illustrated in Fig. 3.11. The figure shows both methods' mean time in solid lines and standard deviation as shaded regions. The proposed AQL method is shown in blue, and the proactive balancing is in orange. The observed mean time to balance the SOC with the proactive balancing algorithm is 0.275 s. The proposed AQL method could archive a 4.5 times faster balancing. A mean balancing time of 0.061 s



(a) (b)

Fig. 3.10. Simulated voltage and current (a), and SOC (b) of a BM3 reconfigurable battery over time employing the proposed AQL algorithm for 10 s and a step size for $\Delta t = 10^{-4} s$ [20].

could be achieved. It can be seen that the presented AQL method is significantly faster than the algorithm proposed by Mashayekhl et al. [169]. In addition, the standard deviation is much smaller within the AQL method. This significant difference in the time until the balanced state is reached can be explained by the fact that the proactive balancing approach strongly focuses on switching in parallel. This target is not specified in the AQL, where during training, the model approximated a benefit in utilizing the bypass configuration to avoid discharge cells with lower SOC.

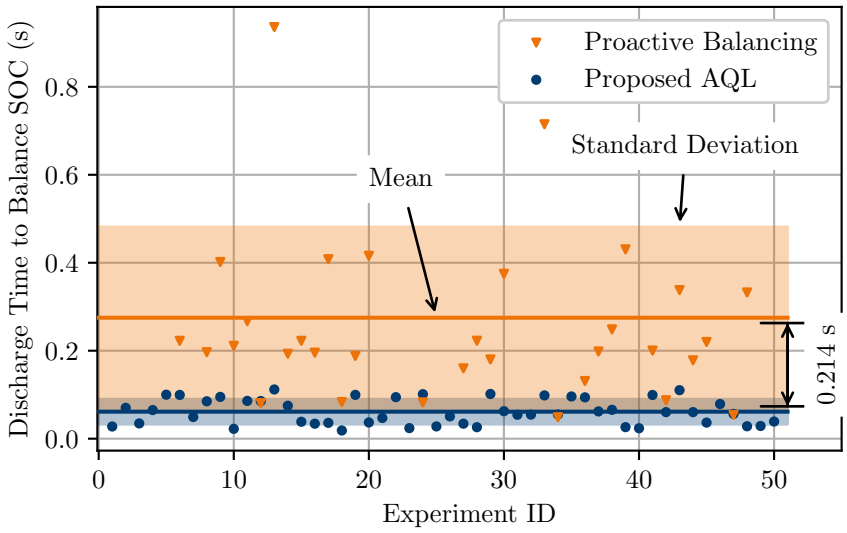


Fig. 3.11. Comparision of required time to reach balanced state with BM3 as topology using proactive balancing [169] and the proposed AQL method over 50 experiments; Proposed method outperforms proactive balancing with 0.214 s

Half-Bridge: Physical Hardware Experiment The equipment used for experimental validation is shown in Fig. 3.12 and described as follows:

- Device Under Test (DUT): 12-cell hybrid cascaded multilevel converter [193], a topol-

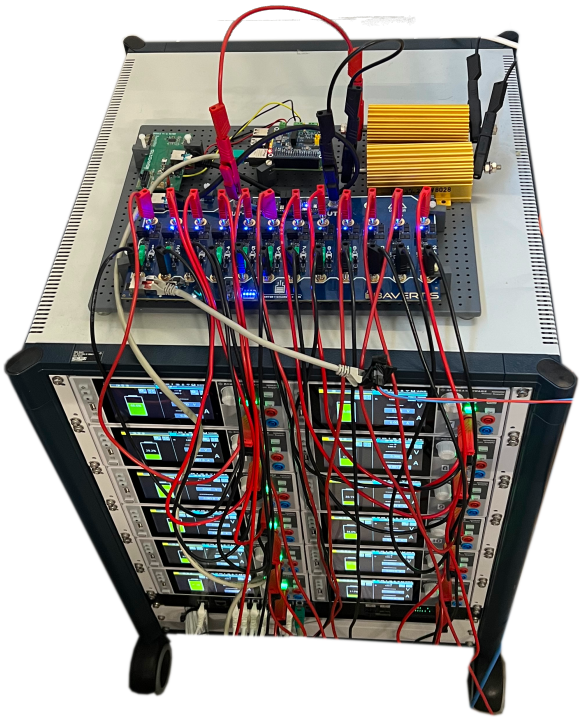


Fig. 3.12. Experimental setup with a 12-cell hybrid cascaded multilevel converter as DUT, ohmic resistance as load, and power supplies to simulate battery behaviour [20]

ogy of interconnected half-bridge modules and a surrounding H-bridge converter

- Raspberry Pi 4 as Control Unit
- Lenovo ThinkPad-P15-Gen-1 as Computing Unit
- 2× Load Resistor: MAL-200 MEG 10 Ω in series
- 12× Battery Cell Simulator: Rohde & Schwarz NGM202 Power Supply

Battery cell simulators are used to simulate the battery behaviour, establish randomly chosen SOCs, and ensure a reproducible setup. Initial SOCs between 70 % and 100 % are set. The cells are discharged to 20 % during each experiment. The computing unit collects the battery cell simulators' battery state data and executes the neural network of the AQL model. The control unit is connected via Controller Area Network (CAN) to the DUT to enable the setting of the switching states. The control unit adjusts the DUT switching states in alignment with the actions determined by the balancing algorithm executed on the computing unit. The control unit, the computing unit, and the battery cell simulators communicate through an Ethernet connection.

As previously for the BM3 topology, the proposed AQL method is discussed and compared with a state-of-the-art SOC balancing algorithm. The algorithm proposed by Zheng et al. [193] prioritizes cells to discharge with the highest SOC. Following this algorithm is mentioned as the switching-max algorithm. Select the n cells with the highest SOC, where n is the number of cells needed to achieve the desired voltage level. This approach intentionally decreases the SOC difference between the cells of the reconfigurable battery.

The experiment was conducted 50 times per methodology, each with a randomly chosen starting charge level for every battery cell. Fig. 3.13 illustrates the time required to reach the balanced state. The balanced state is specified as a maximum SOC deviation of 1 %

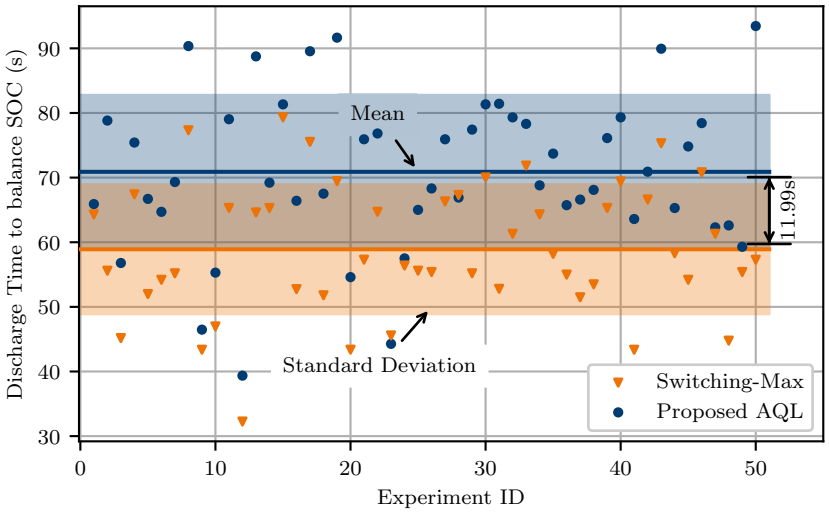
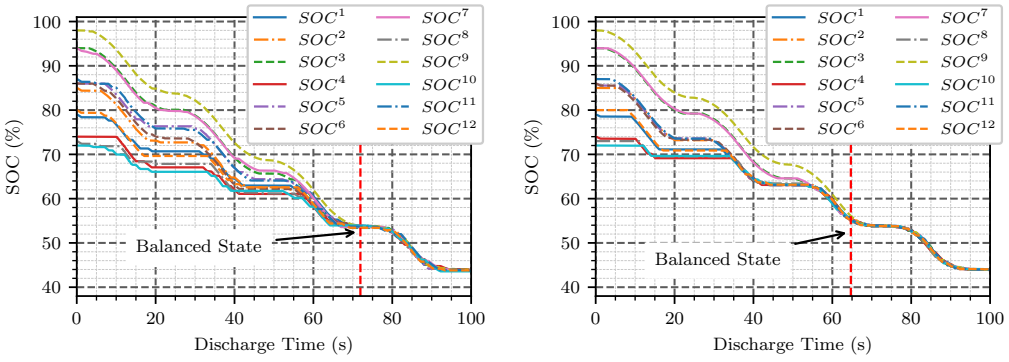


Fig. 3.13. Comparasion of the required time to reach balanced state with Half-Bridge topology using the switching-max algorithm and the proposed AQL algorithm over 50 experiments; Switching-max algorithm outperforms proposed AQL method with 11.99 s [20]

within the cells for simulated cells with a capacity of 0.1 A h. The swiching-max algorithm archives a mean time of 58.9 s ± 10.0 s, which is 11.99 s faster compared to the proposed AQL method. The achieved mean time to balance with the AQL methodology is 70.9 s with a standard deviation of 11.9 s.

To analyze this finding in detail, in Fig. 3.14, the resulting SOC over time for 100 s discharge time is illustrated. In Fig. 3.14a, the proposed AQL methodology is illustrated, and 3.14b shows the switching-max algorithm. By employing the AQL method, the system reaches the balanced state, illustrated with a red line, after 71.9 s at a SOC of 54 %. Apart from the discharge time, the discharge curves of the individual cells display an approximately linear decrease. The switch-max algorithm reaches the balanced state after 64.7 s at a SOC of 55.4 %, which is 7.2 s faster than the proposed AQL methodology.



(a) (b)

Fig. 3.14. SOC balancing using the proposed AQL algorithm (a) and switch-max algorithm (b); Balanced state after 71.9 s and 64.7 s (red) of discharge [20].

The results show that for the BM3 topology, the proposed method can lead to better results than established controlling methods. For the half-bridge topology, characterized by

its two possible configurations, the switching-max algorithm is preferred to control when balancing the SOC.

4 Digital Twinning using Cloud Computing

This chapter addresses the second research question, about the architecture and methodology to transfer data between vehicles and distributed compute units, outlined in Section 1.2 on page 2 and tackles our initial research objectives RO1 and RO2 of this work. In section 4.1 on page 53 a significant advancement in cloud-based Battery Management Systems (BMS) through a comprehensive framework rooted in digital twinning principles. The framework offers a dedicated data specification, as introduced in section 4.2 on page 59, an efficient compute power distribution strategy in section 4.3 on page 69, a core temperature estimation methodology is proposed as a potential use-case of a functional digital twin with cloud integration, and a novel data compression method in section 4.4 on page 88.

4.1 IoT-Framework for Battery Management Systems

Nowadays, a BMS estimates the state of the batteries in addition to the control of the battery pack [194]. Modern BMS has to estimate states such as SOC, SOH, and core temperature. In recent years, approaches such as data-driven methods and digital twins have been proposed to estimate these battery states [133], [195]. The volume of data produced by individual vehicles is constantly increasing, as is the necessity of computational power. A solution to this is shifting some of the tasks into the cloud by utilizing flexibility, scalability, and the provision of computing resources on demand [196], [197]. Increased data reliability, lower cost in acquisition and operation of hardware, improved security, and reduced time-to-market. These are only a few of the positive effects of cloud computing. This section overviews existing frameworks and architectures to enable cloud integration in EVs. In Section 4.1.1 on the next page we introduce a monitoring architecture for reconfigurable batteries. With the Five-Layer IoT Framework in Section 4.1.2 on page 56 we introduce a framework to enable cloud-based BMS.

Standardized frameworks and architectures enable the implementation and comparison of different systems. These frameworks establish a structure and well-defined interfaces, enabling a modular approach to system development. Due to such reasons, the three-layer IoT framework [198] has been applied for the use in cloud-based BMS [199]. The necessity for reference and standard architectures has been identified not only in the context of research but also by industry. With Amazon Web Services (AWS) IoT Fleet Wise, the cloud service provider AWS introduced a service to enable cloud-based fleet monitoring at the end of 2021 [200]. Fig. 4.1 illustrates the structure of Fleet Wise and the three-layer IoT framework. The three-layer IoT Framework illustrated in Fig. 4.1a consists of the layers perception, network, and application layers. As illustrated by the arrows in the figure, the framework is intended exclusively for vehicle-to-cloud communication. Cloud-to-vehicle communication is not available. Furthermore, the framework provides only a data collection layer on the vehicle side. Insufficient attention is given to encapsulating data acquisition, buffering, and transmission. The purpose of the AWS IoT Fleetwise Connected Vehicle platform, shown in Fig. 4.1b, is mainly to monitor the vehicle. It focuses on the software and data abstraction components with the intention of generalization. This platform separates the vehicle's connectivity to the cloud from the data collection.

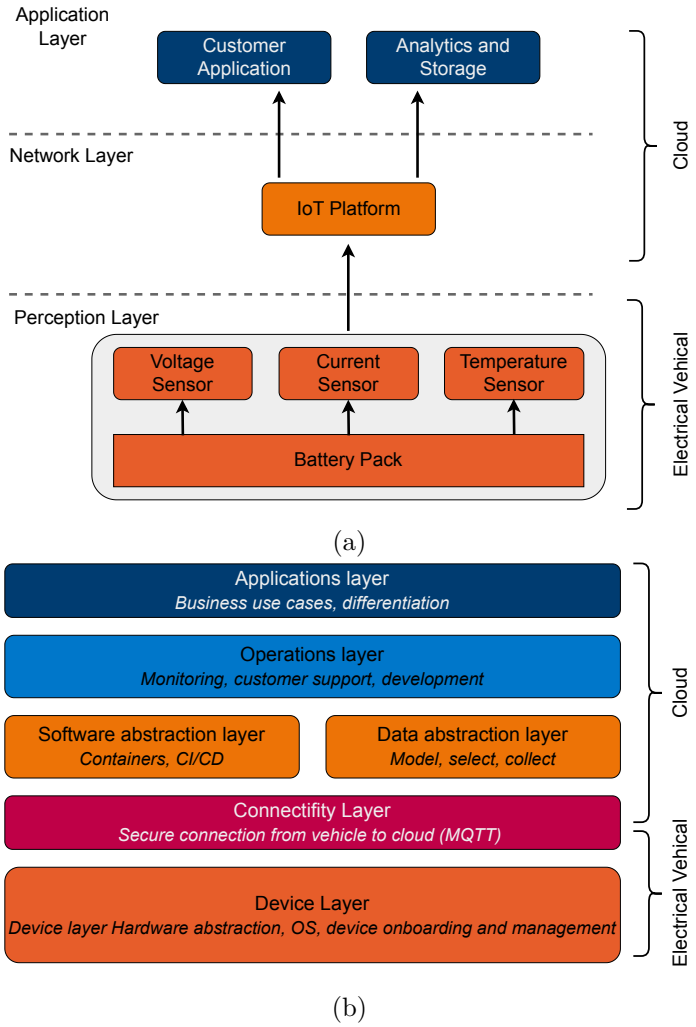


Fig. 4.1. The applied three-layer IoT framework (a) and the AWS IoT Fleetwise Connected Vehicle platform (b) [23], [199], [201] ©IEEE 2024

4.1.1 Monitoring of a Reconfigurable Battery

The service of AWS lacks the possibility of monitoring high-frequent data. The maximum number of messages per device per second is 100 [202]. To enable the monitoring of reconfigurable battery systems and their high-frequent data, we proposed a monitoring architecture in the cloud [24]. The central concept of this architecture is to have an infrastructure for live monitoring, to visualize historical data, and to enable an automated archive of previous experiments for afterwards evaluation. Fig. 4.2 illustrates the proposed architecture. The main components are:

1. Data Collection Units
   An IoT device is the gateway in the vehicle and the laboratory environment. This device has to collect and transform the measurement and ensure robust authentication and authorization with the cloud infrastructure. A reconfigurable battery is used for experimental validation.

2. Cloud Gateway
   In this architecture, the cloud gateway, applied as AWS Kinesis, is the data collection point between the physical devices and the cloud applications. AWS Kinesis is

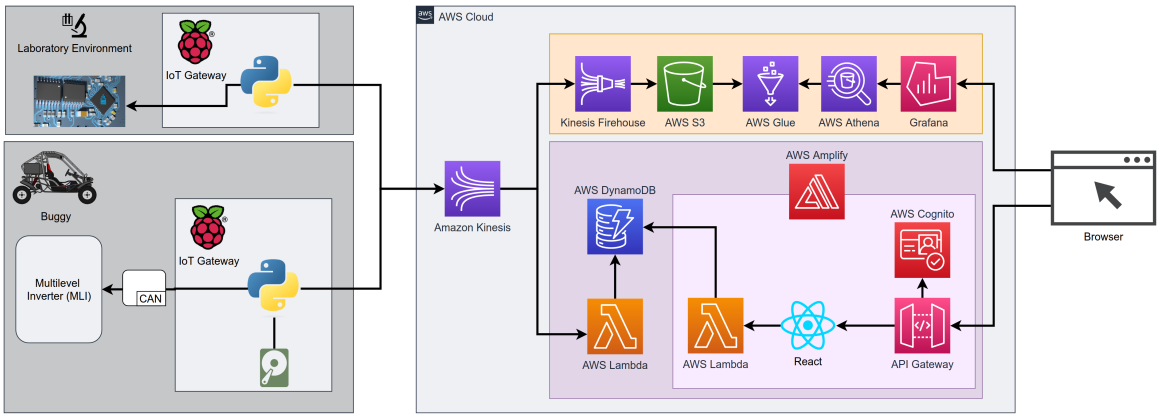


Fig. 4.2. Proposed monitoring framework using AWS services; Data acquisition on laboratory and mobile environment (grey); Amazon Kinesis as cloud gateway; Storing and processing for visualization in orange and life monitoring a real-time application using serverless services to deploy web application [24] ©IEEE 2023

an event stream service based on the open-source software Apache Kafka. An event stream service is a system that processes, stores, and distributes events in real-time. It is like a messaging platform for applications, allowing them to communicate and react to changes in data. Apache Kafka is a popular open-source distributed event streaming platform. It is designed to handle high-throughput, low-latency data streams. The benefit of AWS Kinesis over Apache Kafka is its accessibility as a managed service, which automates installation, initial configuration, scaling, and maintenance. Furthermore, the integration into other AWS services makes it simple to implement.

3. Monitoring Application
   The architecture introduces a serverless web application for real-time monitoring. The app enables remote monitoring of the experiments.

4. Storing of Historical Data
   Standard services such as S3, Athena, and Grafana are implemented to store and analyze historical data. As with Kinesis, the focus of selecting the services was on providing the fullest possible integration of AWS services and minimizing configuration effort.

This architecture extends the concept for automatized analyzing and processing of experimental data from modular multilevel inverters proposed by Wiedenmann et al. [203]. It allows the monitoring of long-term measurements in mobile and stationary experiments. Cloud computing infrastructure allows for straightforward scalability and data sharing with collaboration partners. The deployment of global infrastructure enables worldwide accessibility to the data.

In addition to the advantages of the architecture, there are also limitations. A generalization of architectural frameworks is essential in the context of a broad application of technologies. Moreover, the architecture is currently limited to unidirectional communication. It enables the automation of monitoring for reconfigurable battery systems. Towards cloud-based BMS and digital twins, bidirectional communication becomes crucial. Consequently, we proposed the five-layer IoT framework to further investigation.

4.1.2 Five-Layer IoT Framework

Reference architectures aim to increase interoperability and optimize the development process. Weyrich and Ebert [204] give an overview through reference architectures for IoT. The authors highlight the importance of connectivity, device management, communication between different devices, and the propagation of configurations. In addition, they identified the importance of components for data collection, analysis, and actuation, the scalability of the system and robust security features in reference architectures.

To address the abovementioned requirements, we apply the five-layer IoT framework [205] in the context of BMS. We have proposed this framework in our study [23]. Fig. 4.3 shows the framework layers:

Layer 1 Sensor and Actor,

Layer 2 Device,

Layer 3 Gateway,

Layer 4 IoT Integration Middleware, and

Layer 5 Application.

This five-layer IoT framework aims to use existing components in an EV and add additional features. This approach enables the use of existing and widely used technologies and architectures and combines them with the new technology of IoT. The first layer includes *sensors and actors* in a battery pack, such as voltage, current, and temperature sensors and converter switches. Only the *on-board BMS* is directly connected to this layer for safety and security objectives. In general, an additional device, the *gateway*, should be used to isolate the on-board controlling and monitoring of the batteries from cloud computing infrastructure. The gateway module enables the transformation and aggregation of data from vehicle protocols to those utilized in cloud computing and web technologies. Additionally, the gateway is responsible for authorization through the cloud computing infrastructure, which integrates safety, security, and cryptographic integrity dimensions. The gateway is linked to the BMS to prevent unauthorized external access. The gateway buffers data and transmits only validated information to ensure reliable BMS operation even during network disruptions. This modularity guards the BMS from receiving incorrect data, as data from the gateway is considered extra and not essential for the core functionality of the EV. Optional, the architecture also directly links the onboard BMS and the IoT Integration Middleware. If this is selected, however, it should be noted that the additional security layer is unavailable, and the sizing of the BMS hardware resources must be adjusted. For prototype development or as part of research projects, however, eliminating additional complexity by omitting the gateway layer can lead to faster development.

Furthermore, the cloud infrastructure consists of the fourth layer *IoT Integration middleware* and the fifth layer *application*. The monitoring architecture, described in section 4.1.1 and shown in Fig. 4.2, has implemented AWS Kinesis as a data collection point. The principle of a gateway into the cloud infrastructure is generalized with the IoT integration middleware. The fourth layer is the intermediate interface between the physical vehicles and the cloud applications. Device management, bidirectional data collection, and load balancing are some functions implemented at this layer. Decoupling cloud integration and cloud applications into two individual layers enables the modular development of different services for fleet management, state monitoring, vehicle control, and state estimation.

The main principles of the five-layer IoT framework are modularity and encapsulation of individual components. Within this framework's layers, we employ the existing components,

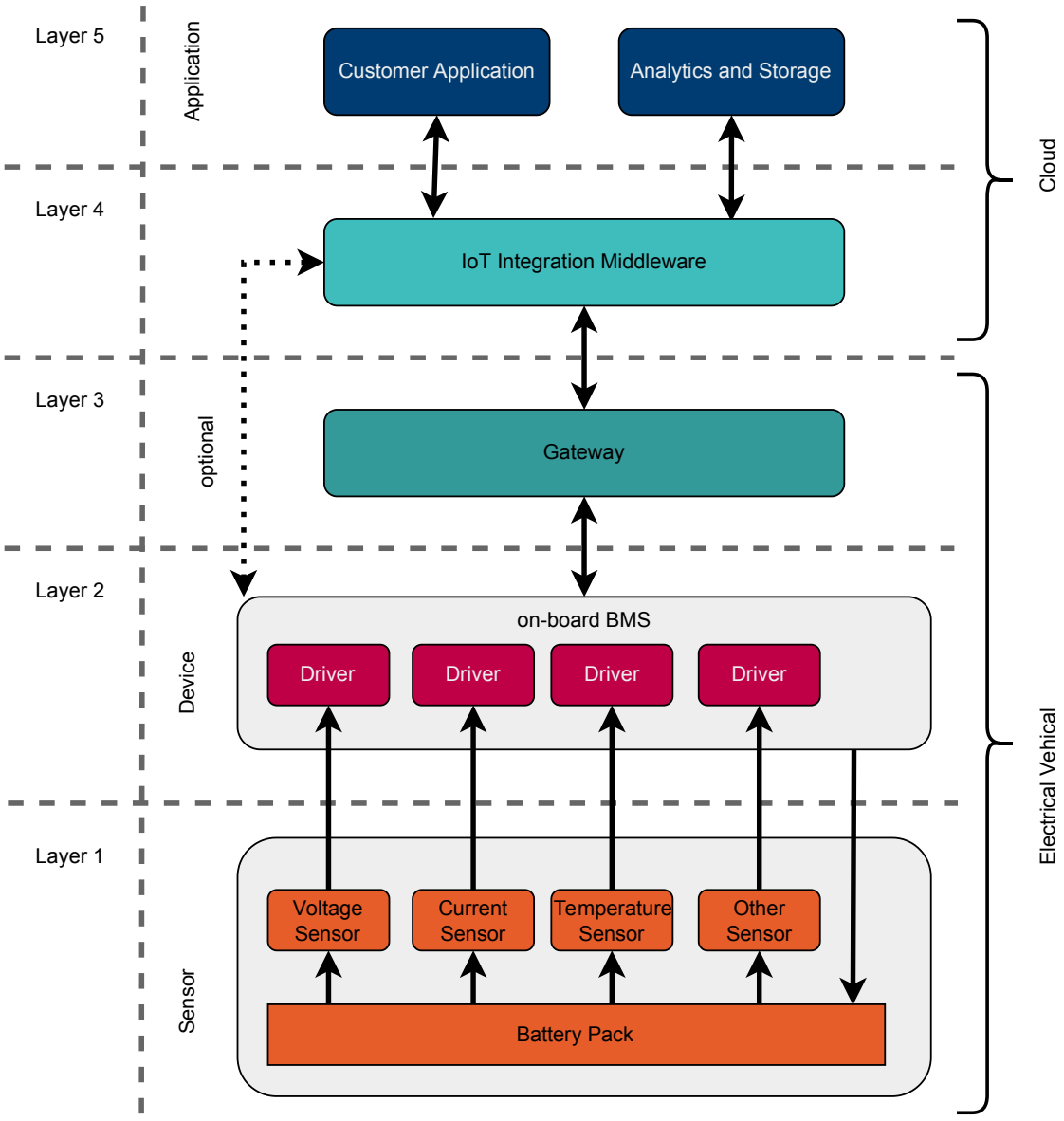


Fig. 4.3. Five layer IoT Framework Guth et al. [205] proposed and applied for BMS [23] containing the layers sensor, device, gateway, IoT integration middleware, and application

sensors and actors, and onboard-BMS of the EV and combine them with IoT technologies. It transforms the EV and especially the battery pack into a driving IoT device. Open standards and open-source software can be developed based on such a framework to increase the impact of EVs research. When it comes to Vehicle-to-Infrastructure (V2I), Vehicle-to-Vehicle (V2V) and Vehicle-to-everything (V2X) [206], the gateway layer also serves as the connection point for communication with other devices besides the cloud.

One method to evaluate a framework, such as the proposed five-layer IoT Framework, is the analysis based on non-functional requirements. Therefore, the product quality model defined in ISO/IEC 25010 [207] is employed. The five-layer IoT framework is evaluated against nine quality characteristics according to ISO 25010. This assessment shows how the proposed framework addresses non-functional qualities compared to the two existing frameworks shown in Fig. 4.1 and highlights areas for improvement.

Functional suitable characterizes the degree to which the system provides functions that meet needs. Research objective 1, defined in Section 1.2 on page 2, establishes the need for two-way data communication to enable the digital twinning of electric vehicles. This objective is provided within the framework presented. The two comparison frameworks only enable one-way communication.

Performance efficiency defines in this context the degree to which the system performs its functions within specific time, resources and capacities. Compared to the three-layer IoT framework, the additional layers of the framework can cause a higher need for time and compute resources caused by the higher complexity of the system; in Section 4.3.2 on page 71, the characteristic of the time behavior is evaluated.

Compatibility describes the degree to which a system can exchange data with other systems. The choices made throughout the development process primarily influence the system's interoperability. For example, the technologies selected, such as the data transfer protocol, can significantly determine how interoperable the system is. The framework does not define technologies. The gateway layer enables the integration of the framework in existing car architectures. As described above, the gateway can also be used for data exchange for V2G or V2V. Similar, as the three-layer IoT framework is a model that does not provide any technological specifications, only the structure of individual components, no statement can be made about compatibility. This depends on the respective implementation. Due to its abstraction layers, the AWS platform is highly interoperable from the perspective of the vehicle platform used but is limited to the technologies of AWS themselves.

Interaction Capability delegates how a system can interact with the user. This aspect is not covered within the framework presented.

Reliability describes the degree to which a system performs specified functions under specified conditions for a specified period. As the framework considers, vertical decoupling of functionality within the electric vehicle enables a higher fault tolerance. It is of considerable importance to the system's reliability that failures or compression at the cloud levels of the IoT integration middleware or an application do not result in errors within the vehicle's critical functions.

Security characterizes the degree to which a system defends against attack patterns. The security depends on the architecture and implementation of a system. Implementing a layered architecture that separates the gateway from the device can enhance security by making it more challenging, due to multiple layers of security and authentication,

for external attackers to gain unauthorized access to the vehicle. Also, the IoT Integration Middleware, as the fourth layer, serves as an authentication layer for the applications and enables the isolation of the applications from each other to increase resistance against failures. The AWS IoT Fleetwise Connected Vehicle platform includes the Connectivity Layer security features, which need to be added to the concept of the 3-layer IoT Framework.

Maintainability defines the degree of effectiveness and efficiency in a system that can be modified to improve, correct and adapt to changes. Decoupling into individual layers of functionality enables modifications on each layer with minimal effect on the other layers, increasing modularity. Besides vertical decoupling, the application layer allows horizontal decoupling of the applications from each other, increasing modularity, the application level analysis, and the modifiability of individual applications and levels. The abstraction layer of the AWS IoT Fleetwise Connected Vehicle platform enables a robust decoupling of vehicle and cloud data and thus significantly increases the possibilities of modularity and modifiability. The modelling of the vehicle within a single layer, the perception layer, demonstrates that the 3-layer IoT framework [199], as it can be seen in Fig. 4.1 a, was not designed with modularity in accordance with the relevant standards.

Flexibility describes how a system can adapt to changes. The separation of device and gateway increases the flexibility towards several different types of vehicles connected to one cloud environment. Considering the vehicle layers, the AWS solution also has a high degree of flexibility. Still, this platform is tied to a cloud provider, which limits cloud-side flexibility in terms of the replaceability of central components. On the other hand, the framework presented in our work does not assume any technologies as a precondition, which can increase the flexibility.

Safety delegates the degree to which a system under defined conditions avoids a state where human life, health, property and the environment are endangered. Separating the devices and gateway can primarily ensure operational safety in case of cloud connectivity loss or loss of data integrity. The AWS solution incorporates this safety function as an essential component of its design by separating the connectivity and device layers.

The analysis shows the advantages of the five-layer IoT framework, as proposed in this work, over the three-layer IoT framework by missing out on essential concepts to ensure non-functional quality characteristics. In particular, regarding flexibility and functional suitability to the research objective defined in Section 1.2 on page 2, the five-layer IoT framework also represents a clear advantage over the AWS solution.

4.2 Universal Real-Time Streaming Data Specification for BMS

The previous section addressed the question of how to structure components of a IoT environment for EVs to enable cloud-based BMS and digital twinning. In this context, another question arises: How can data be transferred between a vehicle and the cloud infrastructure? In other words, are there cloud-ready data specifications for real-time streaming of EVs data? Therefore, we will address this question in the next section. The following section is based on our published work [26].

There are several standards and codes for the exchange of various kinds of EVs data [208], such as ISO 15118 standardizes vehicle-to-grid communication [209], and ISO 13400

standardizes the diagnostic communication over Internet protocol (DoIP) [210]. Clark et al. [211] aims to increase the utilization of published data and facilitate the collaboration and comparison of data-driven research and development methods. The authors proposed a unified ontology concerning battery pack data to improve interoperability and seamless information integration across diverse domains. Many open-source software projects have been published to transform and analyze battery tester data, such as Battery Evaluation and Early Prediction (BEEP) [212], cellpy [213], Battery Data Toolkit [214] and Galvanalyser [215]. The use of these software solutions leads to different data formats. In addition, there are also open databases, such as the Universal Battery Database [216] and BatteryArchive [217], which have predefined data structures. Lininger et al. [218] introduced a Comma-Separated Values (CSV)-based data specification designed to facilitate the storage, sharing, and processing of measurement data of testers and battery packs. Furthermore, multiple publicly available datasets using various data formats, such as .mat, .csv, and .txt [219], [220].

Despite the availability of many formats and infrastructures, the need for standard and widely accepted data specifications is noticeable. Such missing specifications lead to longer delays and complexity in pre-processing the data. Furthermore, inadequate and missing documentation makes the data difficult to understand and reuse in developing the next generation of modern BMS with cloud integration.

The idea of a data hub with standardized data specifications for battery data is introduced in the Battery Genome Project [221]. The aim is to make community-wide cooperation more convenient and impactful in battery research. The methodology we propose in this work introduces a data specification for conventional and reconfigurable battery packs. The utilization of the specification is for historical data and the potential implementation of cloud-based BMS. The Battery Genome Project lacks in a data specification enabling real-time data streaming and a data specification for reconfigurable battery packs.

4.2.1 Proposed Data Specification

The data specification we propose in this work [26] addresses the previously mentioned limitations by fulfilling several essential features and qualities:

- storing of historical battery data and cloud-ready streaming during operation and testing
- providing in-data experimental setup documentation, such as battery cells and power electronics equipment;
- minimizing the requirements of the data transformation process that occur among researchers, organizations, data platforms and modern programming languages;
- open data specification can facilitate the development of open source solutions in the field of cloud-based BMS and reconfigurable batteries

The proposed specification contains three types of data: (a) metadata, (b) time series measurement data, and (c) cell configurations of reconfigurable batteries.

In contrast to previous data specifications mentioned in [218]–[220], ours separates the description of the experimental setup and its related measurements and presents both in a machine-readable way [26]. The metadata contains information about the installed battery cells and sensors, the micro- and macro-topology of the system, general information about the experiment, and others. The time-series data contains all the measurements collected within the experiment and the cell configurations of reconfigurable batteries over time.

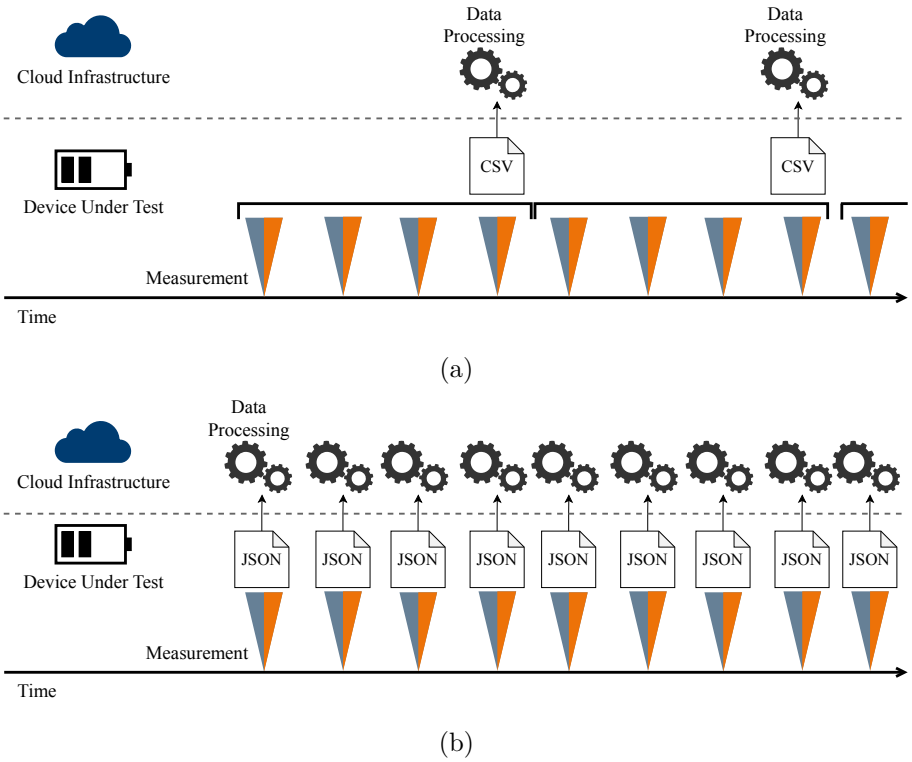


Fig. 4.4. Comparison of data streaming methods: CSV (a) aggregates multiple measurements into a single file, and JSON (b), each measurement stored as a separate package [26].

The data specification uses different data formats for specific use cases to facilitate human and machine readability. The employed data formats are JavaScript Object Notation (JSON) and CSV. Fig. 4.4 illustrates the main difference between CSV (a) and JSON (b) data formats when it comes to data streaming. The figure illustrates measurements, represented by grey-orange triangles, over time and the resulting data transmission using the specific data format. In the case of CSV, the aggregation of multiple measurements in a single file is shown in (a), while JSON is designed to process each measurement itself, as shown in (b). JSON is more suitable for streaming data applications due to its inherent flexibility, support for schema evolution, resolution of delimiter issues, and efficacy in real-time processing. However, it requires greater storage capacity for long-term data retention compared to CSV. Ben Smith [222] introduces the data format JSON. JSON provides the possibility of describing hierarchical structures and relations. Due to, the proposed data specification uses JSON to describe the metadata associated to the experiment. For streaming and processing in real-time of time-series and cell-configuration data JSON is recommended as data format. On the other hand, CSV files are recommended to store historical time-series and cell-configuration data.

Location Schema

The precise location of components, batteries, and sensors within the system is necessary to ensure the correct interpretation, evaluation and comparison of measurements. The components differ by the level at which they are located. Figure 4.5 shows the various levels within a system and their interrelationships: arm, module, submodule, and cell. Within the proposed data specification, each system element is described by a numerical index linked

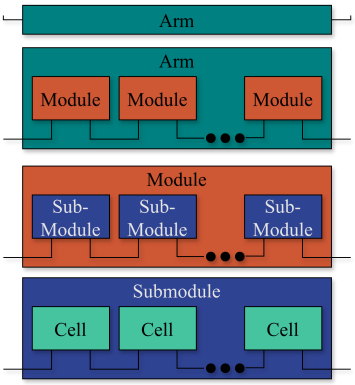


Fig. 4.5. Levels of components [26]

to the lowercase initial of the corresponding level. These levels are arranged hierarchically in descending order and are separated by underscores. For example, a location scheme such as a2_m4_s1_c3. Within this example, the schema identifies cell 3, in submodule 1, in module 4, in the second arm of the module. Including system levels is optional; a level can be omitted if it does not exist, such as in a1_c2, which identifies the second cell within the first arm.

Table 4.1: Metadata datatype description; Name of the field, indentation represents hierarchy; Field required and Description of the field [26]

Name	*Required*	*Description*
Experiment ID	X	A unique identifier of the Experiment
Start Time	X	Start time of measurements series in UTC per ISO 8601 standard: (yyyy-MM-dd'T'H:mm:ss)
Format Version	X	The version of the data specification. It changes when the specification changes.
Test Protocol		Path/URL to the test protocol. If data is published, the protocol also has to be published.
Description		Description of the performed test.
Battery		List of battery types used.
Battery ID	X	Identifier of the cell type in the worldwide/company/organization.
Chemistry		Cell chemistry, such as $LiFePO_4$.
Manufacturer		Manufacturer of the battery cell.
Model		Model of a battery cell.
Voltage min		Minimal voltage of battery according to the manufacturer in volt.
Voltage max		Maximal voltage of battery according to the manufacturer in volt.
Current max charge		Maximal battery current during charge according to manufacturer in ampere.
Current max discharge		Maximal battery current during discharge according to the manufacturer in ampere.
Temperature min		Minimal temperature in Celsius of the battery according to the manufacturer in Celsius.
Temperature max		Maximal temperature in Celsius of the battery according to the manufacturer in Celsius.
Nominal capacity		Nominal capacity according to manufacturer in ampere-hour.
Cells	X	List of used battery cells.
Cell ID	X	Unique identifier in the company/organization.
Cell location	X	Location as described in Section 4.2.1
Serial Number		Serial number of the cell manufacturer.
Calendar Age Production		Calendar date/month of production in ISO 8601 date format (yyyy-MM-dd)
Calendar Age Storage		Calendar date/month of installation in the system in ISO 8601 date format (yyyy-MM-dd)
Charge input		The charge input during the lifetime of the cell before this measurement started in ampere-seconds
Current Capacity		The current capacity of the battery cell in Ampere-hour.
Sensors	X	List of used sensors.
Sensor ID	X	A unique identifier of the sensor. The proposed schema is <measurement/estimation identifier>_location.

Table 4.1: Metadata datatype description; Name of the field, indentation represents hierarchy; Field required and Description of the field [26]

Name	*Required*	*Description*
Location	X	Location of the sensor as defined in Section 4.2.1
Unit	X	The unit prefix and the unit of the sensor
Step Type		Is required if the sensor describes step type. Key-value pair to describe the meaning of the step types.
Description		Describes the sensor. This has to be indicated if the value is calculated or determined via averaging.
Manufacturer		Manufacturer of the sensor.
Part Number		Identifier of sensor given by the manufacturer.
Calibration		List of calibration values.
Measured Value	X	Value measured by the sensor.
Reference Value	X	The calibrated reference value.
Calibration Date		Date of the calibration per ISO 8601 standard: (yyyy-MM-dd)
Measurement Error		List of measurement errors
Measured Value	X	Value measured by the sensor.
Standard Deviation	X	The standard deviation of the background noise.
Calibration Date		Date of the measurement per ISO 8601 standard: (yyyy-MM-dd)
Cell Configuration	X	List of cell Configurations.
Location	X	Location of the cell as defined in Section 4.2.1
Micro Topology	X	Name of the micro topology
Configuration States		Dictionary to map configuration as stored in Section 4.2.1 to configure state. Required, if not hard-wired.
Switch		List of installed switches.
Manufacturer	X	Manufacturer of installed switch
Part Number	X	Part number of installed switch
Production Date		Production date of the installed switch per ISO 8601 standard: (yyyy-MM-dd)
Macrotopologie	X	The name of the macro topology
Files		Required if it is historical data and not real-time streaming. Includes references to time-depending data
Time Series	X	List of files with time series data in descending order
Cell Configuration		List of files with cell configuration data in descending order

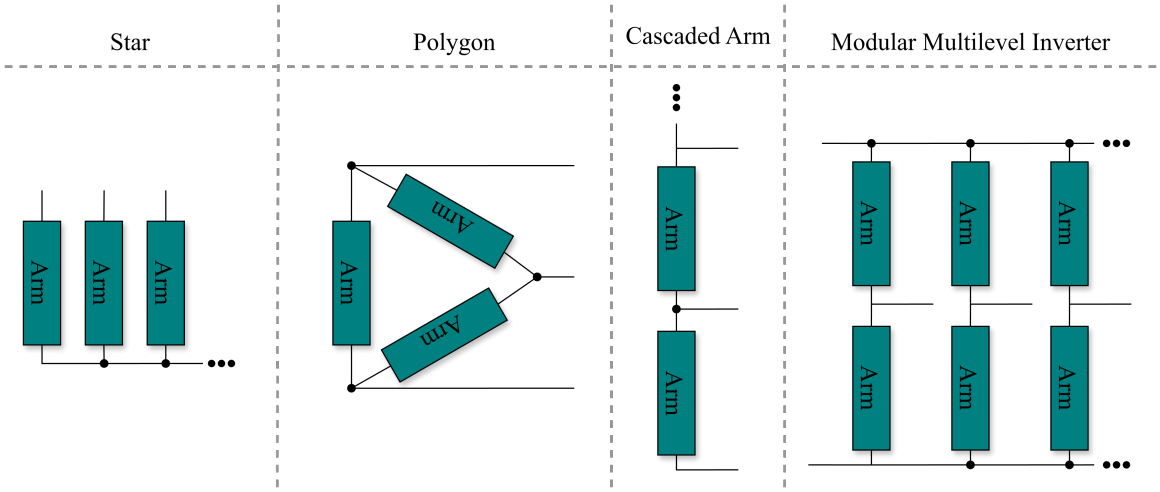


Fig. 4.6. Schematic illustration of reconfigurable batteries macro topologies: star, polygon, cascaded arm, and modular multilevel inverter; differentiation of the arrangement of the arms of the topologies [26]

Metadata

All metadata attributes and hierarchy, illustrated by its indentation, are shown in Table 4.1. Each field is specified by its hierarchy, the required field and its description. The attributes can be mandatory or optional. The metadata is stored as JSON file containing general documenting, static information of the experiment. The main attributes are experiment ID, start time, and description, among others. In addition, macro-topology is an important attribute for reconfigurable batteries. In the literature, various topologies have been proposed, such as series, parallel, serial-parallel configuration, matrix, star, cascaded arm, modular multilevel converter, polygonal and polygonal star configuration [70], [223]. Figure 4.6 shows the macro topologies start, polygon, cascaded arm, and modular multilevel converter with interconnected arms as examples of topologies. As shown in Table 4.1, battery, sensors, and files include subsequent attributes of the metadata specification and are delineated below:

Battery: The type of battery and cell chemistry characterize the behaviour of the battery pack. In addition, parameters such as minimal and maximal voltage, maximal discharge current, and the intended temperature range to operate the battery help to interpret experimental setups. The battery subsequent attribute lists several battery types installed within the system. Each battery type must list its installed cells. A single cell is identified by location and unique identifier and is characterized by optional parameters such as charge input during lifetime, current capacity, day of production, and day of storage.

Sensors: The physical location and the installed sensor are required to interpret the context of the collected time-series measurements. A defined sensor can be a physical sensor or an estimation model, such as a SOC estimation model. Sensor accuracy is crucial when implementing electrothermal, electrochemical, and data-driven models. Including detailed information on sensor noise and calibration procedures helps to improve the comparative analysis of different models. Accuracy, measurement frequency, and similar information can be found in the datasheet accessed by the manufacturer, and the part number is included as part of the sensors' attributes.

Cell Configuration: In a conventional battery pack, cells are connected in series and parallel. Thus, a time-dependent configuration is not in place. On the other hand, the topology of a reconfigurable battery must be known to reconstruct the configurations. Bidirectional double full-bridge/MMSPC [70], half-bridge [71], battery modular multilevel management/Marx, unidirectional full-bridge [72], bidirectional full-bridge [73], and switch-clamped half-bridge [76] are some of Modular Multilevel Inverter (MMI) topologies proposed in the literature and illustrated in Figure 2.6 on page 15.

The configuration of a half-bridge topology is shown as an example in Listing 4.1. Bypass and serial are the possible configurations of this topology. The topology is located in the first arm, which is the second module. Installed switches are not specified in this example.

```
"cell_configuration": [{
  "location": "a1_m2",
  "micro_topology": "half-bridge",
  "configuration_states": {
    0: "bypass",
    1: "serial"
}}]
```

Listing 4.1: Metadata of a half-bridge submodule [26]

Time Series Data

The time series data includes measurements from sensors specified in the metadata. Each datapoint of measurement includes the timestamp, which is relative to the start time of the experiment or an absolute value as ISO 8601 standard [224]. The time series measurements can be stored as JSON and CSV. The Listings 4.2 and 4.3 compare these two data formats. Both listings contain identical measurements but in two different representations. The example illustrated in Listing 4.2 shows JSON, which includes attributes with their values. This allows different measurement frequencies for each sensor and enables the transferring of empty values in case of errors, such as shown with the attribute soc_a1. While CSV, as shown in Listing 4.3, each attribute must be included within each line. It is independent if the attribute has a value or not. The separation between empty and missing, as explained for JSON, is not feasible in the CSV data format. An empty value does not show if it is missing, caused by an error or by different measurement frequencies. This can lead to an incorrect interpretation of the measured values. While both listings represent the same information, CSV requires less storage space since it uses fewer characters to describe the data compared to JSON.

```
{"timestamp": 9266.23, "cur_a1": -0.073, "vol_a1": 7.31, "soc_a1":
    undefined}
{"timestamp": 9630.11, "vol_a1_m1": 3.7, "soc_a1_m1": 0.85, "vol_m2"
    : 3.6, "vol_a1_m3": 3.5}
```

Listing 4.2: Example of time series data in the JSON format [26]

```
timestamp,cur_a1,vol_a1,vol_a1_m1,soc_a1_m1,vol_a1_m2,vol_a1_m3,
    soc_a1
9266.23,-0.073,7.31,,,,,
9630.11,,,3.70,0.85,3.67,3.65,
```

Listing 4.3: Example of time series data in the CSV format [26]

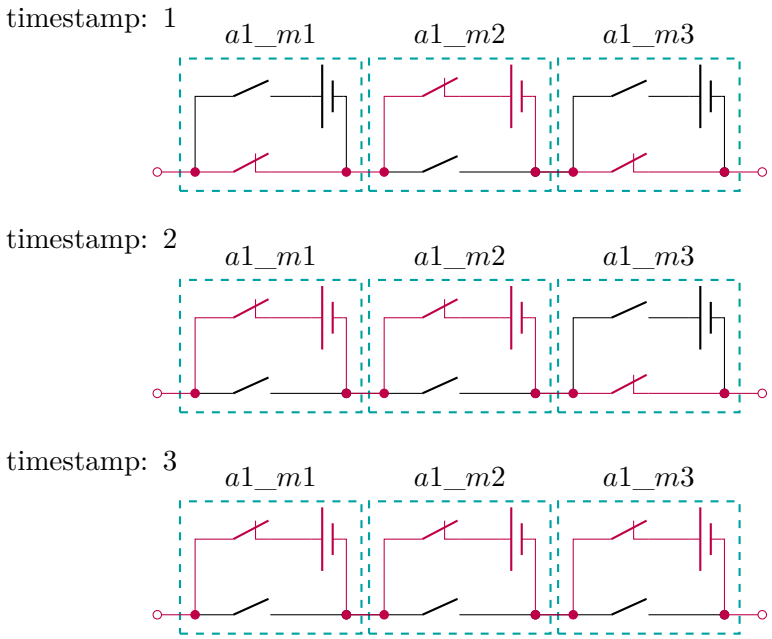


Fig. 4.7. Configuration of a half-bridge reconfigurable battery with three modules $a1\_m1$, $a1\_m2$, and $a1\_m3$ over three timesteps; at timestamp 1 only module 2 is serial connected; at timestamp 2 the modules 1 and 2 are switched serial; at timestamp 3 all modules are switched in serial

Cell Configuration

Streaming and storing their cell configurations is necessary to monitor and analyze the system control of reconfigurable batteries. The data specification for cell configuration at each timestamp is comparable to the previous proposed time series data, which uses JSON or CSV. Each data point comprises a timestamp and the system cell configurations. In Fig. 4.7, an arm of a reconfigurable battery with three half-bridge modules is illustrated. The figure shows the configuration state at three timestamps. Listing 4.4 shows the representation of these three timestamps as JSON. The configuration of the arm, module, submodule, and cell is delineated in the cell configuration, while the metadata enumerates the potential states of each component. The data interpretation of these elements facilitates mapping the specified values, exemplified by 0 and 1, to their corresponding configurations. This information is annotated in Listing 4.1, which is included in the metadata.

```
1 {"timestamp":1, "a1_m1":0, "a1_m2":1, "a1_m3":0}
2 {"timestamp":2, "a1_m1":1, "a1_m2":1, "a1_m3":0}
3 {"timestamp":3, "a1_m1":0, "a1_m2":0, "a1_m3":0}
```

Listing 4.4: Example of cell configuration data in the JSON format as seen in Fig. 4.7

4.2.2 Experimental Validation

Cloud infrastructure has been deployed to evaluate the proposed data specification in the first step [26]. The system implements the five-layer IoT framework proposed in section Section 4.1.2 on page 56. As the framework's electrical vehicle component, we have established the setup as shown in Fig. 4.8. The components are described as follows:

Sensors and Actors: 12-cell hybrid cascaded multilevel converter [193] (green) as DUT, a

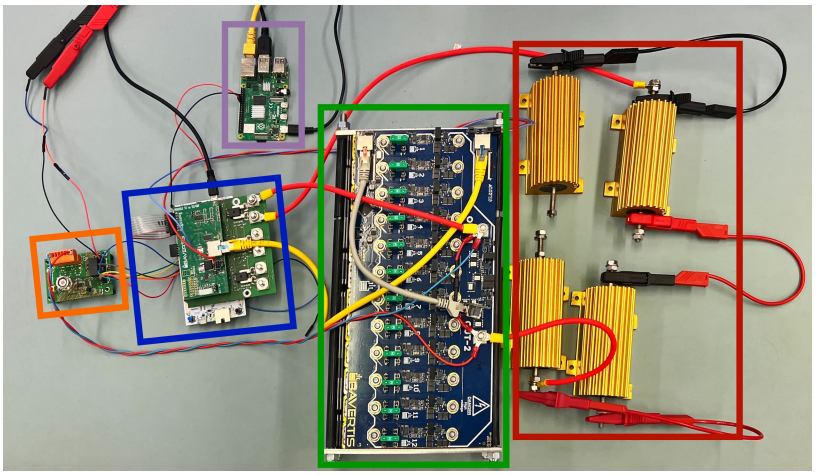


Fig. 4.8. Experimental setup containing reconfigurable battery (green), load resistance (red), a master controller (blue), voltage measurement unit (orange), and Raspberry Pi 4 as gateway (purple) [26].

Table 4.2: Required storage per data format for experiment

	Cell Configuration	*Time Series*
Data points	94,633,109	15,278,114
Storage		
JSON	21.5 GB	4.2 GB
CSV	7.2 GB	1.6 GB

topology of interconnected half-bridge modules and a surrounding H-bridge converter and a voltage measurement unit (orange)

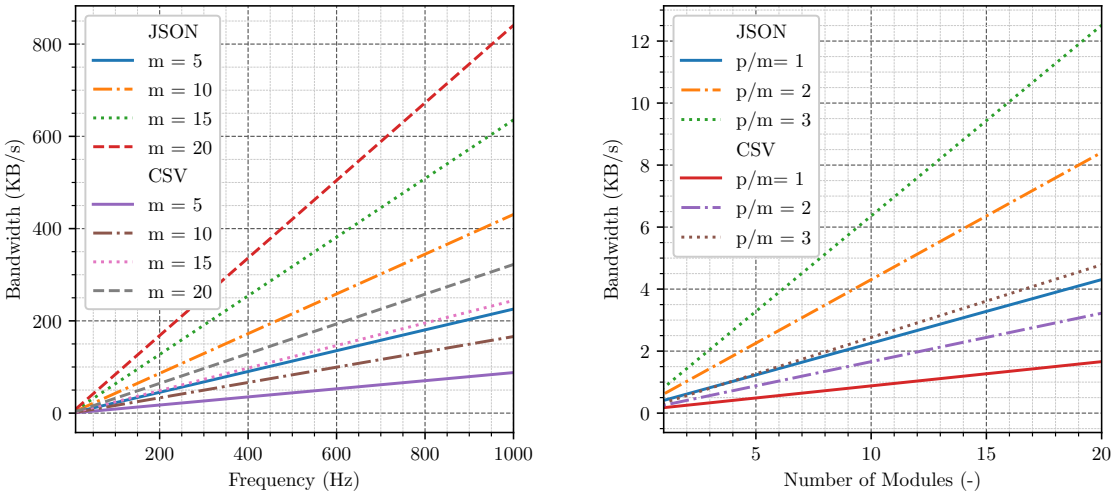
Gateway: Raspberry Pi 4 (purple) as Gateway

On-board BMS: Custom Master Controller (blue)

Load: 4× MAL-200 MEG 2× 10 Ω load resistor in series and 2× 10 Ω in parallel

The experimental data contains measurements and cell configurations of a 3.27 h discharge test with a 2 Hz sinus wave. The experiment enables the evaluation of the required storage for both usable data formats JSON and CSV for both kinds of data time series and cell configuration. Furthermore, the benefit of subsequent calibration and measurement noise information has been discussed in the original work [26]. Table 4.2 shows the number of measurements and cell configurations stored during the experiments. Also, it shows the storage required to store cell configurations and measurements for each data format, JSON and CSV.

We assessed the bandwidth and storage requirements for assumed measurements to assist the scalability of our proposed data specification of various assumptions, considering factors like measurement frequency and the number of batteries employed. This analysis is essential for a comprehensive evaluation of the proposed data specification. In Fig. 4.9, the calculated required bandwidth to transfer the measurements under several conditions is shown. Fig. 4.9a shows the bandwidth depending on the frequency for both data formats. It ranges from five to 20 modules and has two measurement parameters for each module. The accuracy of



(a) (b)

Fig. 4.9. Required bandwidth in KB/s under several configurations, number of modules m and parameters per module p/m and used data format; Linear increasing with the increase of measurement frequency (a) and number of modules (b); JSON requires more bandwidth than CSV if other parameters are equal;

the assumed measurements is three digits with three decimal places. Fig. 4.9b illustrates the bandwidth at changing module amounts. The number of parameters per module (p/m) is set between one and three, and a measurement frequency of 10 Hz. Both graphs show a linear correlation between bandwidth and increase of modules or frequency. This visualization makes it possible to determine the required storage capacities (CSV) and bandwidth (JSON) for the initial stages of developing a cloud-based BMS.

4.3 Distribution of Computing Resources

After proposing the five-layer IoT framework for EVs and the universal data specification for BMS, as discussed in Section 4.1 on page 53 and Section 4.2 on page 59, this section investigates the crucial question of compute power distribution. Regarding digital twinning and cloud-based BMS, the research question arises of where to compute which tasks and how to distribute the computing tasks. Tran et al. [89] highlight the limited resources regarding computationally intensive estimation methods. Cloud computing infrastructure needs to be considered to overcome this limitation. Dynamic accessible, globally distributed computing and storage infrastructure can enable the data-driven estimation of SOC and SOH and other battery states. The economic implications, connectivity, data transmission rates, and latency represent critical factors in assessing the feasibility of this technology. In the following, various architectures are discussed to distribute computational tasks and the computational cost of several state estimation methods.

4.3.1 Computing Distribution Architectures

Our analysis extends beyond the basic paradigms of cloud computing in examining prospective computational architectures for intelligent BMS. Therefore, we discuss fog computing for e-transportation to mitigate latency between the vehicle and the computational infras-

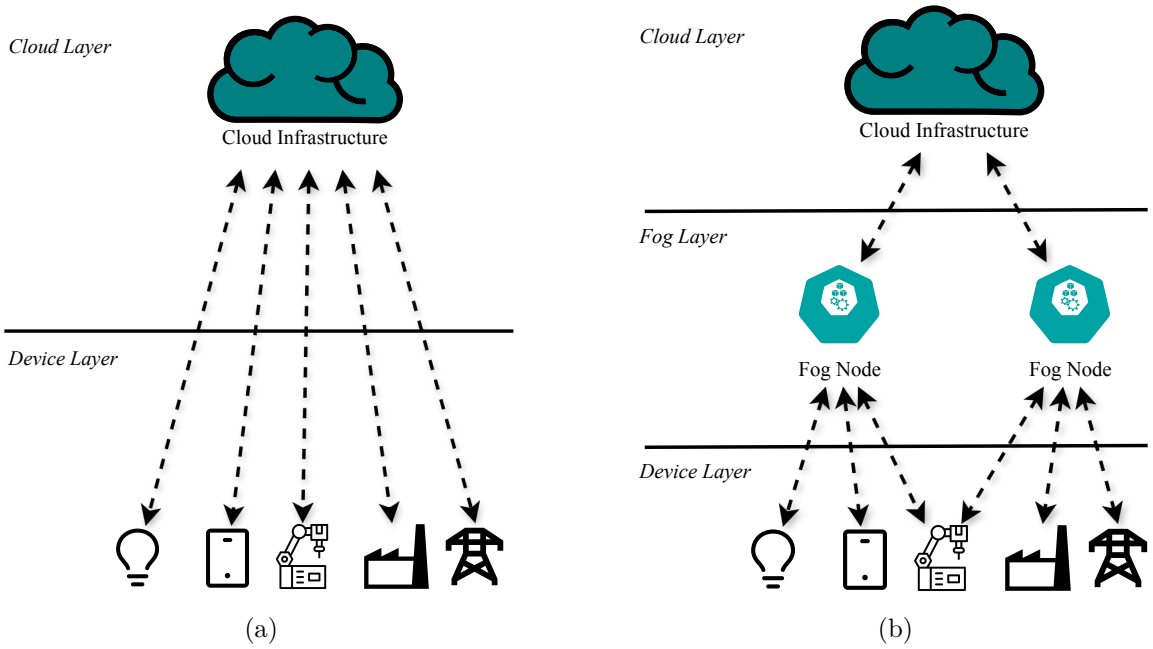


Fig. 4.10. The difference between the distributed computing paradigms cloud computing (a), direct connection between devices and cloud environment, and fog computing (b), fog nodes as a bridge between devices and the cloud environment

tructure [23].

Within the Cyber Hierarchy and Interactional Network (CHAIN) framework, Yang et al. [87] proposes a framework for cloud-based BMS. This framework contains four subsystems: end, edge, cloud and knowledge. For time-critical tasks, the edge computing component is implemented in the vehicle. The component is embedded within electronic devices for communication and interaction with the battery systems for monitoring and controlling. This edge computing framework consists of different computing services. The edge component reduces cloud load and enables efficient real-time data processing. The cloud component handles less time-sensitive functions, including SOH, core temperature estimation, adaptive control, fallout prediction, and proactive safety alerts.

We can identify a research gap in the potential of fog computing for cloud-based BMS. Fig. 4.10 illustrates the fog computing and cloud computing schema. Fog computing is a geographically distributed computing architecture, a bridge between the cloud and the devices [225], [226] to reduce the latency between the devices and the distributed computing resources. Fog computing addresses limitations in computing and storage resources on the device. Furthermore, it compresses, filters, and pre-processes the communication between the device and the cloud.

The potential of fog computing as a distributed computing paradigm has been seen and discussed in fields such as Industrial Internet of Things (IIoT), agriculture, healthcare, entertainment, intelligent transportation and smart cities [227]. Due to the given infrastructure in domains such as IIoT and smart cities, challenges such as rural area coverage, global scalability for mobile devices, and network connectivity disruptions are considered minimal concerns. However, the methodologies and frameworks for employing fog computing within these fields can be effectively adapted for e-transportation.

In [228], differences between mobile fog and fixed fog resources are discussed. The mobile fog consists of vehicles, such as cars and buses. These computing units can be accessed at close range but have restricted resources. In contrast, fixed fog computing resources

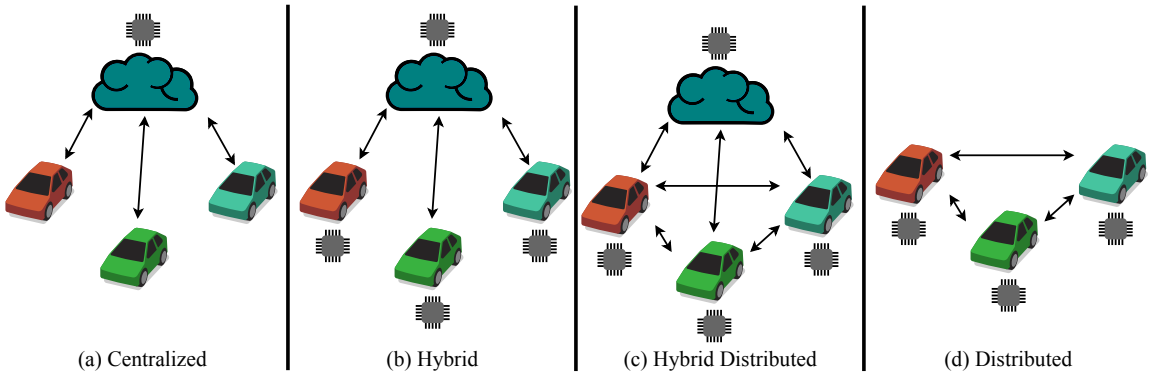


Fig. 4.11. Four fog computing architectures with mobile and fixed fog computing units: centralized (a), hybrid (b), hybrid distributed (c), and distributed (d)[229]

are available at a moderate distance and possess greater computational power. Moreover, Wang et al. [229] describe four different architectures to implement distributed computing employing fixed and mobile fog computing and cloud computing. Fig. 4.11 illustrates these four different architectures: centralized (a), hybrid (b), hybrid distributed (c), and distributed (d). The architectures differ regarding where the generated tasks are executed. A task defines a computing operation to be executed. The following cloud unit is defined in that way: it includes fixed fog and cloud computing resources.

Centralized: All tasks are executed in cloud units only.

Hybrid: Tasks are executed locally on the vehicle or uploaded to cloud units for execution.

Distributed: Tasks can be executed locally or shared with mobile fog units.

Hybrid Distributed: Tasks can be executed locally, on cloud or mobile fog units.

Despite the low complexity of centralized computing, it suffers from complete control loss during network outages, which can be a significant issue, especially in rural areas. While leveraging shared computational power among vehicles for complex tasks, the distributed architecture presents various challenges. The challenges include interoperability between different types of vehicles and manufacturers, fault tolerance, privacy, and security. The exclusive use of a mobile fog can be of interest in the context of campus applications and localized bus and truck fleets. However, the hybrid-distributed architecture, which combines in-vehicle computing with fog, mobile and fixed, and cloud resources, offers a more robust solution. It enables features like V2V communication for safety and swarm-based fleet management and maintains functionality even during network failures. Based on the above-mentioned limitations of the centralized, hybrid-distributed, and distributed architectures, the hybrid architecture has the highest potential, particularly in real-world high-scale e-transportation applications. By utilizing supplementary services in fixed fog and cloud computing layers while the automobile retains a degree of autonomy, a failure in connectivity does not result in an interoperative vehicle because of the remaining vehicle autonomy.

4.3.2 State-of-Charge Estimation in the Cloud

The evaluation of potential applications is required in addition to the general concepts of cloud-based BMS and theoretical opportunities. We conduct in our work [27], [28] an evaluation based on empirical data to explore the feasibility of SOC estimation within a cloud-based framework. This work aims to evaluate the applicability of previously proposed

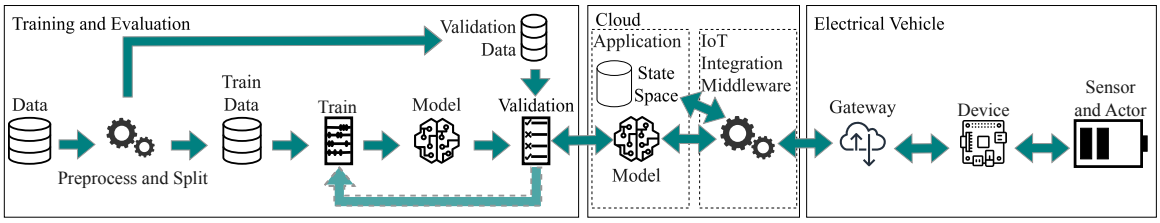


Fig. 4.12. Cloud Computing architecture to estimate the SOC containing training and evaluation steps, and cloud and electric vehicle components which implements five-layer IoT Framework

estimation models in a cloud environment. For this reason, various estimation models are implemented and deployed. As an adaptive filter and a classic approach, an Extended Kalman Filter (EKF) is employed. The schematic of EKF is proposed in [230]. This method is compared with three deep learning models proposed by Guo and Ma [231], which introduced several neural network architectures and compared them in accuracy and computational cost. We evaluate FNN, LSTM, and GRU as three deep learning methods which show promising results in Guo and Ma [231], as FNN and LSTM as demonstrated minimal and maximal computational cost and GRU as the highest accuracy among the evaluated models. SOC has been chosen as an example of different types of battery states. Large datasets exist to develop and evaluate SOC estimation methods; several well-discussed methods require intensive computation, and the state estimation is highly time-sensitive - all of which are well-suited to the capabilities of cloud computing. The scalability, reliability, and cost-effectiveness of cloud computing make it an ideal platform for SOC estimation. This choice allows us to demonstrate the potential of cloud computing in battery state estimation towards cloud-based BMS.

Cloud Computing Architecture

The architecture of this work [27] implements the five-layer IoT framework. The framework is introduced in Section 4.1.2. Fig. 4.12 shows the in this section implemented architecture consisting of the five-layer IoT framework, and a third component, training and evaluation. This component is added besides the cloud and electrical vehicle components of the five-layer IoT framework. It contains the steps of the machine learning workflow [232]: data pre-processing and split, training, and validation. Our primary objective of the architecture is to ensure cloud readiness. This readiness enables both cloud and on-premise deployment, where Docker containers are used to facilitate this objective. Jonathan Bartlett [233] introduces containerized software engineering for cloud computing with Docker. The deep learning models are implemented in Keras 2.12.0 [234]. The model is provisioned with the Docker image of TensorFlow Serving 2.12.1 [235]. The architectural elements, training and evaluation, operate independently of the cloud applications and can be implemented on various infrastructures. Training occurs on-premise hardware, and cloud components are deployed to a cloud service provider. The chosen cloud computing service provider is AWS.

Dataset and Experimental Setup

Several key details must be specified for the experimental evaluation. First, the measurement data utilized to train the deep learning models and parameters for the adaptive filter must be clearly defined. Also, the location of the experimental setup should be specified to establish the physical distance to the cloud infrastructure. Kollmeyer et al. [236] published charge, discharge, and driving cycle data of a Turnigy Graphene 5 A h 65C cell. The data

Table 4.3: MAE and RMSE of 45 min (discharge of SOC 100 % − 91.81 %) UDDS driving cycle at 0 °C and a measurement frequency of 10 Hz [27].

Model	*MAE*	*RMSE*
EKF	**0.0002**	**0.0002**
FNN	0.0067	0.01
LSTM	0.0744	0.0763
GRU	0.0455	0.0493

includes experiments at a wide range of temperatures between 40 °C and −20 °C. Based on this dataset, Khaum et al. [237] determined temperature-depending ECM parameters. This parameters are employed in our experiments within the EKF. Next, we evaluated the precision of the four methodologies, using Urban Dynamometer Driving Schedule (UDDS) drive cycle at 0 °C as test data. This data has been excluded from the training data used to train the deep learning models.

The cloud computing components are deployed at two locations to evaluate the computing time in the case of a global fleet and the impact of only local deployments. The system is deployed in eu-central-1 and ca-central-1, located in Frankfurt, Germany, and Montreal, Canada, respectively. As computing infrastructure, at each location, a t2.2xlarge instance with Amazon Linux is chosen as the operating system of the virtual machine service Elastic Cloud Computing (EC2). The DUT is located at the Ontario Tech University in Oshawa, Ontario, Canada. The approximately straight-line distance between the DUT and the cloud infrastructure can be seen in Table **??**.

Discussion of Accuracy and Applicability

The following parameters and their impact for the applicability of the state estimation methods within the cloud computing environment are going to be discussed:

- Precision of the four estimation methodologies EKF, FNN, LSTM, and GRU;
- Influence of geographical distance on request latency;
- Effects of selected network technology; Comparative analysis of wired versus Long Term Evolution (LTE) networks;
- Precision of selected estimation methods in terms of network instability

The accuracy of the four estimation models EKF, and neural networks FNN, LSTM, and GRU is discussed. In Table 4.3, the MAE and RMSE of the models are listed. The EKF model showed better results than the other evaluated models, with an MAE and RMSE of 0.0002. The FNN model shows the highest accuracy among the neural networks evaluated with a MAE of 0.0067 and RMSE 0.01. In contrast to the results shown in Table 4.3, Guo et al. [231] had a lower accuracy in SOC estimation applying the FNN model compared to LSTM and GRU, this may be due to the evaluated datasets.

The use of high-accuracy battery parameters by Khaum et al. [237] produced a higher accuracy of estimation than the use of the Kollmeyer et al. [236] data set to train the neural networks proposed by Guo et al. [231]. The results shown in Table 4.3 can be seen as a baseline for accuracy to discuss the following computational cost and the impact of loss of network connection on the performance of could-based BMS.

Using international cloud service providers such as AWS, Microsoft Azure and Google Cloud enables flexibility, scalability, and global distributed deployment. Fig. 4.13 shows

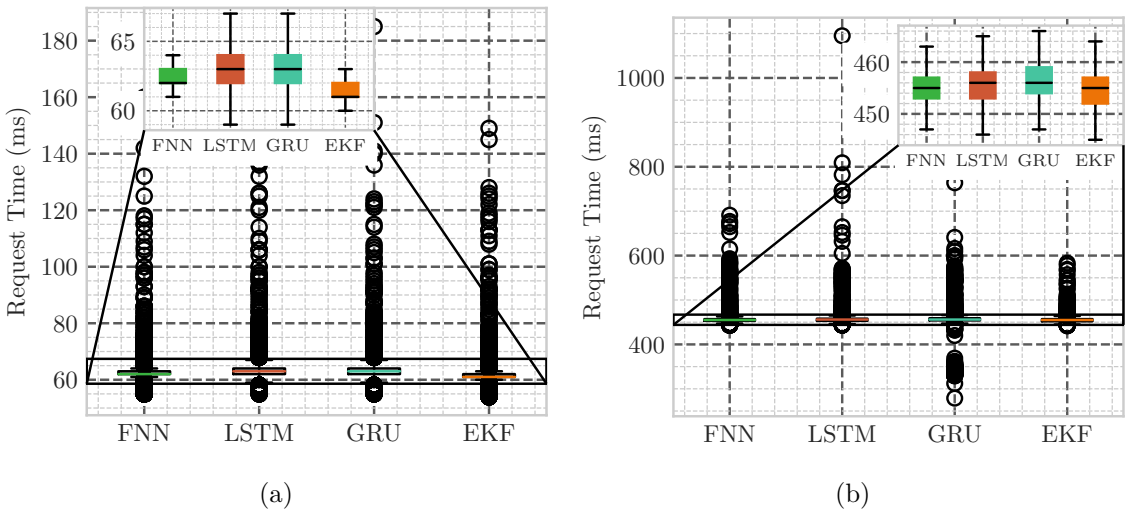


Fig. 4.13. Request time depending on the model and distance between DUT in Oshawa, Canada and cloud infrastructure in (a) Montreal, Canada and (b) Frankfurt, Germany [27];

the measured request time in ms of the different models and the two geographical locations of the cloud infrastructure, Montreal and Frankfurt. The distances between DUT and the cloud infrastructure are given in Table **??**. The request time in ms increases on a far-distance deployment, shown in Fig. (b), than the local deployment in Fig. (a). The request time among the evaluated models in Fig. (a) does not show a significant difference. However, FNN and EKF show slightly shorter request times than the other methods. Fig. (b) shows no observable difference in request time between the four models.

This leads to the conclusion that in the context of cloud-based state estimation models, the deployment location has more impact than the methods used within each evaluated location.

Guo et al. [231] found significant differences in on-board execution time. This means the execution time of the different models is less impactful when deploying the model in the cloud context than within on-board execution. This implies that models with higher computational demands can be executed in the cloud before deployment locally, where computing resources are constrained.

Regarding a general discussion of the feasibility of cloud-based BMS, the maximum applicable request time must be considered. To the best of our knowledge, no formal standard exists for the frequency of SOC estimation. Previous work has mentioned 1-10 Hz as a typical sample frequency according to current practice [238], [239]. It is also mentioned that the higher the frequency, the better the accuracy. This work defines an estimation frequency of 10 Hz as a baseline to reach. None of the methods can reach this condition if deployed in Frankfurt. All of the models in Montreal fulfil this requirement with an average request time of 62.5 ms overall methods.

The above-mentioned experiments took place using wired connections. Mobile connection is the necessary technology for real-world applications in EVs. Therefore, LTE is used as public mobile network infrastructure to evaluate request time. For this experiment, the two models EKF and FNN are chosen as the models with the consistently highest accuracy and lowest computational costs regarding request time. Fig. 4.14 depicts the outcomes of this experimental analysis. During the experiment, DUT was static at Ontario Tech University's campus. In the case of FNN, a request time of 54.8 ms $\pm$ 10.3 ms could be observed, which is shorter than the request time with wired internet connection shown in

Fig. 4.13a. The average request time for EKF is 66.1 ms ± 29.4 ms. However, a network interruption of 1.1 s occurred during the experiment. After this interruption, a shorter request time of 53.5 ms ± 11.9 ms could be archived. This shows no significant difference between the evaluated models after the network interruption. The results of both methods could exceed the requirements of 0.1 s per request. In conclusion, this series of experiments demonstrates that cloud-based SOC estimation is feasible in real-world scenarios. However, attention must be given to the request time to ensure optimal performance.

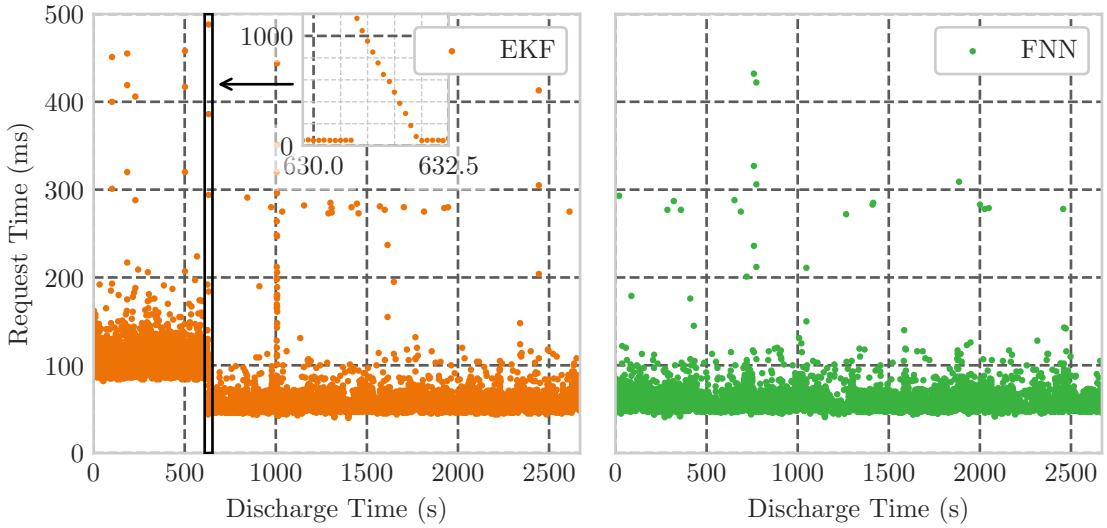


Fig. 4.14. Request time over during discharge using LTE as mobile network; comparing the performance of EKF (orange) and FNN (green); Network interruption of 1.2 s observed in EKF [27]

In application-based research, packet loss is an important parameter that needs to be evaluated within a cloud-based BMS context. There are two kinds of package loss: random loss of individual packages and total loss of connectivity. In this work [27] FNN and EKF are evaluated under random package loss. We evaluated the two models' accuracy at different loss rates, as shown in Table 4.4.

To handle the package losses for EKF, which is missing observations, Cipral et al. [240] propose a solution to overcome this issue. The absence of observations resulting from connection disruptions can be mitigated by substituting zero values, thereby addressing data incompleteness. The MAE of both models depending on the loss rate in % is given in Table 4.4. In the case of EKF, an increase in MAE with a higher loss rate is observed. The missing of observations cause this. Nevertheless, following the increment of 0.0003 in MAE, the accuracy remains substantially superior to that of the FNN, which exhibits a constant accuracy of 0.0062 MAE. The time-series independence of FNN is the reason for no measurable accuracy change. Fig. 4.15 shows estimated SOC with a loss rate of 10 %. The illustration shows the estimated SOC and the measured SOC, using the capacity-based method, which is considered as baseline. The results in Fig. 4.15 and Table 4.4 show a significantly higher accuracy of the EKF compared to FNN. However, in the event of a complete connection loss, non-time-dependent data-driven approaches like FNN can be employed to recalibrate the EKF, thereby enhancing and preserving the estimation accuracy.

Loss Rate	*MAE FNN*	*MAE EKF*
0 %	0.0062	0.0002
1 %	0.0062	0.0002
2.5 %	0.0062	0.0002
5 %	0.0062	0.0004
7.5 %	0.0062	0.0005
10 %	0.0062	0.0005

Table 4.4. MAE of estimated SOC by FNN, and EKF compared to the capacity-based SOC at different loss rates [27].

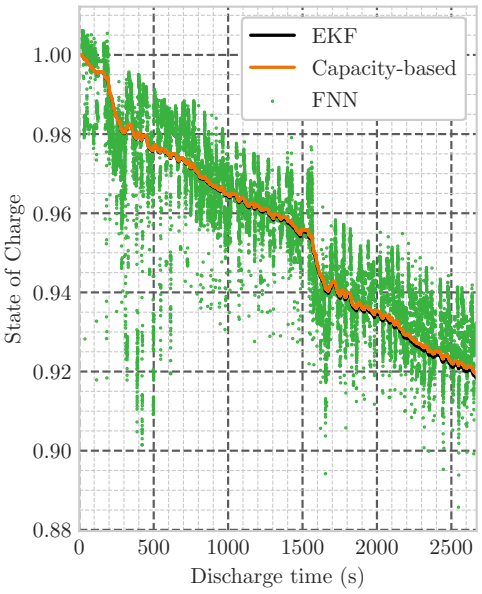


Fig. 4.15. SOC estimation of EKF and FNN compared to capacity-based SOC; EKF outperforms FNN, which has high scatter [27]

Assessment of the Feasibility for State-Estimation in Cloud Computing Environments

The previous part of this work evaluated the applicability of SOC in cloud computing infrastructure. SOC has been chosen as an example of battery state estimation. The system implements the five-layer IoT framework and compares four estimation models. These models have been deployed at AWS as a cloud service provider. The significant findings of this section include the following:

- The least average request time among the compared models and network infrastructures is archived by FNN with a mobile network connection.
- The EKF model demonstrates the highest accuracy, even in the presence of packet loss.
- No significant difference in request time among the evaluated models could be observed.
- Distance between DUT and cloud infrastructure is the main impact on request time. This confirms the need for a global cloud and fog computing infrastructure in the case of cloud-based BMS on fleet scale.
- The short distance deployment located in Montreal, Canada, reached the target of minimum 10 Hz estimation frequency for wired and mobile network connection.

4.3.3 Data-Driven Core and Surface Temperature Estimation Models

The previous section has shown that implementing cloud-based SOC estimation in the cloud can be performed under the specified conditions, according to which the estimation must

be performed within 0.1 s. Another crucial battery parameter is the temperature of the battery, especially the core temperature. In Section 2.1.1 on page 13, we have defined the difference between a battery cell's ambient, surface, and core temperature. Thermal management often involves using cooling fluids or materials, the flow of which is controlled and adjusted based on the measured temperature using control and feedback algorithms [89]. Estimating battery temperature as part of the cloud-based BMS is essential to improve local controlling [89], [241]. In our work [30], we proposed two models to estimate the core and surface temperature of cells.

Li-ion batteries' safety, performance, and durability depend significantly on the cells' operating time [242], [243]. Operating batteries outside the manufacturer's defined conditions also leads to safety issues such as performance degradation and thermal runaways [244], [245]. Thermal complications increase in EVs battery packs, which usually comprise hundreds to thousands of tightly packed individual cells. Close packing causes thermal imbalance, worsened by uneven heating and cooling, especially during dynamic operations and rapid charging. Hence, it is crucial to closely monitor the temperatures of individual cells to ensure a longer battery life.

Monitoring the temperature of each cell with sensors in a battery pack increases the cost and the amount of installed hardware. Therefore, temperature sensors are placed only at crucial points within the battery pack, which reduces the observability of cell temperature [246] and requires temperature estimation. Even with surface temperature sensors, the core temperature can differ from the surface temperature up to 10 °C within a cylindrical battery [247]. Thus, keeping track of or approximating the core temperature is essential to ensure the thermal safety of Li-ion batteries. Temperatures can be predicted by high-fidelity thermal models [248]–[251], reduced order models [252]–[254], lumped thermal models [247], [255], Numerical methods [256]–[258] such as finite element methods, and Electrochemical Impedance Spectroscopy (EIS)-based internal temperature estimation methods [259]–[261] all these methods differ significantly in their accuracy, the resolution of the temperature distribution within the individual cell and the battery pack and the required computing resources.

In addition to classical approaches, the researchers proposed several data-driven estimation methods implementing architectures such as FNN [262], RNN [263], GRU [143], [263], and LSTM [263]–[265] to predict the temperatures of the battery core. These models utilize measurements such as battery voltage, current, surface temperature, SOC, and internal heat losses to the battery as input. The need for a measured surface temperature causes a limitation. Reducing the reliance on physical sensors is crucial for efficient thermal state estimation and management in battery packs. Similarly to SOC, the internal heat losses are an estimated state. Equivalent thermal models, as shown in Figure 2.4b on page 14, for estimation is inaccurate over time as the parameters are subject to change due to battery ageing. Therefore, heat generation is not an accurate input parameter for machine learning models. In Table 4.5, an overview through estimation models, the model inputs, the estimated parameters and their performance can be seen.

Table 4.5: Summary of references for temperature estimation [30]

Reference	*Model*	*Model Input*	*Estimated Parameters*	*Performance*
Lin et al. [246]	H_∞ model	I, inlet coolant temperature	T_S, T_{core}	Worst case $\leq 3.1\,°C$
Richardson et al. [258]	EKF-based estimator	I, V	Temperature distribution	RMSE $< 0.7\,°C$
Chen et al. [58]	EKF-based estimator	I, V	T_S, T_{core}	$T_{core} \pm 1.5\,°C$ /$T_S \pm [-0.5\,°C; 1\,°C]$
Schmidth et al. [259]	Impedance-based	EIS-Measurement, SOC	Bulk temperature	with SOC $\pm 0.17\,°C$ without SOC $\pm 2.5\,°C$
Wang et al. [264]	LSTM with transfer learning	I, V, T_S, T_{amb}	T_{core}	RMSE $< 0.3302\,°C$ MAE $0.3302\,°C$
Zheng et al. [261]	LSTM	Impedance, I, V	volume-average temperature	RMSE $0.46\,°C$
Surya et al. [265]	2D grid LSTM	I, V, $\dot{Q}$, T_S	T_{core}	RMSE $\leq 0.81\,°C$
Xu et al. [266]	Nonlinear Spatiotemporal Modeling	I, V	Temperature distribution	RMSE $\leq 0.1439\,°C$
Kleiner et al. [267]	ECM-NARX-network	SOC_{int}, I	T_{core}	MSE $< 0.5\,°C$
Wei et al .[262]	LTNN-UKF	I, V	Temperature distribution	RMSE $\leq 0.4266\,°C$ MAE $0.3180\,°C$
Zhang et al. [263]	LSTM	I, V, T_S	T_{core}	RMSE $\leq 0.171\,°C$ MAE $\leq 0.148\,°C$
Zhang et al. [263]	GRU	I, V, T_S	T_{core}	RMSE $\leq 0.131\,°C$ MAE $0.099\,°C$
Chin et al. [268]	Electro-Thermal State-Space model	I, T_{amb}	T_S, T_{core}	T_S MSE $\leq 3\,°C$

In our work [30], we propose and evaluate two neural network architectures as a solution to the research gaps mentioned above in the temperature estimation of Li-ion batteries. The contributions can be summarised in the following points:

- LSTM and KAN as neural network architectures for core and surface temperature estimation.
- No need of surface temperature as input, reduces cost and wire-harness of BMS.
- Operating ambient temperatures of −20 °C to 40 °C of prediction during charging and discharging conditions used to train and validate.

Experimental Data

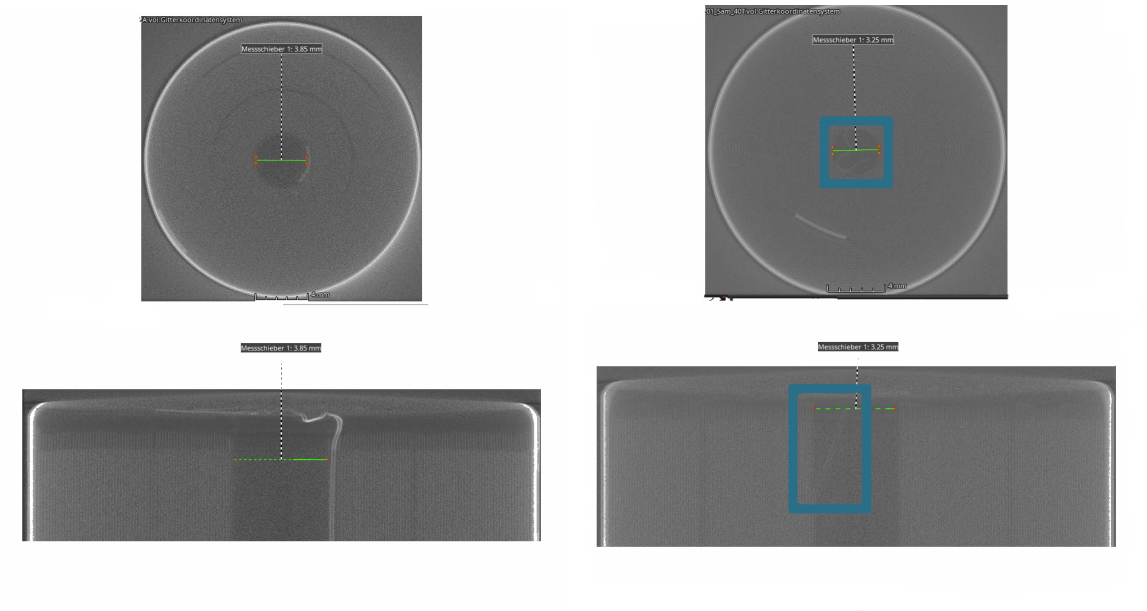
Before discussing the estimation methods, we describe the experimental setup used to generate the core and surface temperature data to train the models.

As the first step, a computed tomography (CT) scan was performed to ensure the possibility of a core temperature sensor insertion. The sensor can be inserted if the winding mandrel has been removed during production and the innermost winding of the cell has not collapsed, meaning that the core is free of material. For this purpose, scans of the negative pole of a Molicel P42A and a Samsung 40T were performed. The results of these CT scans are illustrated in Fig. 4.16. The winding mandrel of both cells has been removed during production, and a gap of 3.85 mm (Molicel P24A) and 3.25 mm (Samsung 40T) can be noticed. As far as can be seen from the scan, the core of the Molicel P42A, as seen in Fig. 4.16a, is free of material and is suitable for inserting a sensor. The core of the Samsung 40T, as seen in Fig. 4.16b, on the other hand, has a collapsed inner winding, which means that inserting a sensor can lead to a short circuit and result in a thermal runaway. The collapsed inner winding is highlighted in blue. Based on this finding, Molicel P42A was chosen as the cell for subsequent experiments. The battery specification in the manufacturer's data sheet is given in Table 4.6.

Table 4.6: Molicel P42A information

Property	*Molicel P42A*
Nominal Capacity	4200 mA h
Continuous Discharge Rating (max)	45 A
Nominal Voltage	3.6 V
Maximum Voltage	4.2 V
Discharge cut-off Voltage	2.5 V
Discharge cut-off SOC	30 %
Nominal discharge	4 A
Maximal discharge	45 A
Nominal charge	2 A
Maximal charge	8.4 A

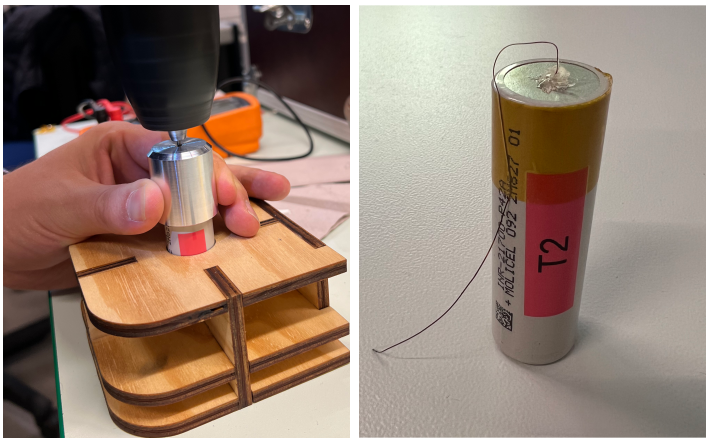
Before inserting a physical temperature sensor (TE Connectivity, GA10K3MCD1) to measure the core temperature, the cells were discharged to SOC 0 % and cooled down at ambient temperature of 0 °C. Before inserting the sensor, the cells are drilled with a width of 0.55 mm vertically through the negative terminal to insert the temperature sensor. Fig. 4.17a shows the drilling of the sensor. The figure shows a cylindrical casing attached temporarily to the cell. The casing has a central borehole with free space to collect metal shavings to prevent inserts into the battery. A Negative Temperature Coefficient (NTC)



(a) (b)

Fig. 4.16. Computed tomography (CT) scans from negative pole with top and side view of a Molicel P42A (a) and Samsung 40T (b) with collapsed winding within the core highlighted in blue

thermistor is employed as a core temperature sensor. The sensor provides a accuracy of ± 0.2 °C at 20 °C. After drilling and inserting, the whole is carefully sealed with adhesive to prevent air from entering and electrolytes from escaping. The prepared cell is shown in Fig. 4.17b. Eight cells were prepared for the experiments. Due to sensor damage or breakage, four cells could be used during the experiments.



(a) (b)

Fig. 4.17. Battery cell preparing; (a) drilling of cooled down, discharged cell, (b) inserted core temperature sensor,

A K-type thermocouple sensor is also installed on the surface of each cell for surface temperature measurement. The K-type sensor provide an accuracy of ± 2 °C for temperatures between −25 °C to 400 °C.

The experimental setup to collect training, test and validation data is shown in Fig. 4.18. The setup enables the collection of voltage, current, surface, and core temperature. The

setup consists of the following equipment:

- Battery cycler: Ivium OctoStat5000
- Analogue data collector: Ivium OctoPDA-T
- Environment control champer: Binder MKF 240
- Power Supply: MCP LBN-1990
- System control unit: Microsoft Surface Pro 7
- Instrument Control Software: IviumSoft platform

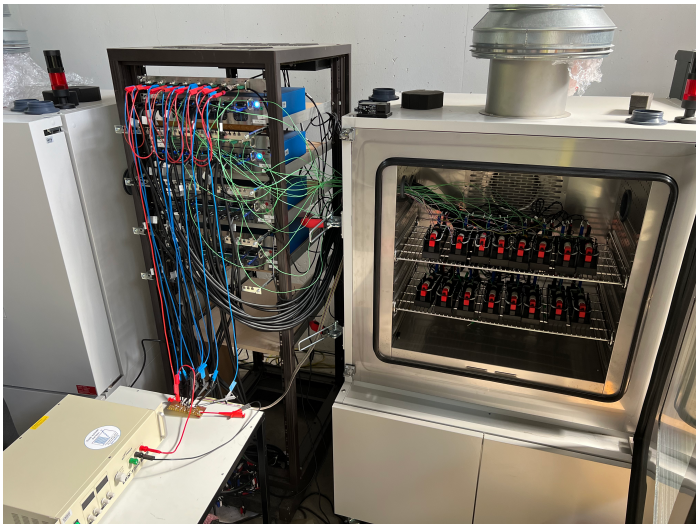


Fig. 4.18. Battery cycling setup with cycler and climate chamber with seven prepared cells and nine validation cells

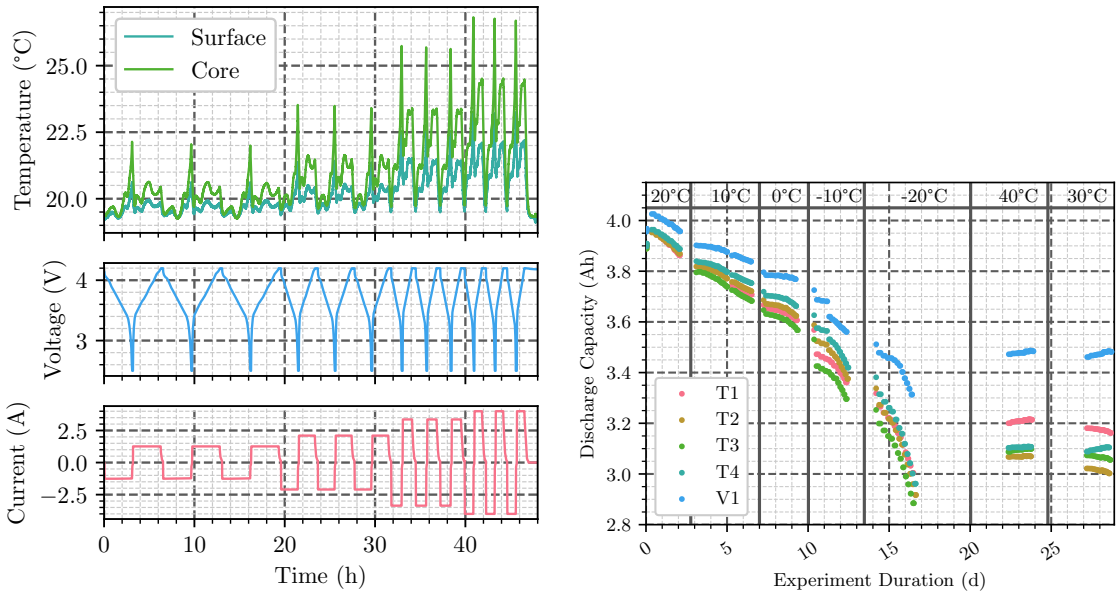
The experiments were carried out over a wide temperature range and different loading and unloading rates at each temperature. The test arrangements were as follows:

- Ambient Temperatures in performed order: 20 °C 10 °C, 0 °C, −10 °C, −20 °C, 30 °C, 40 °C
- 3× each Constant Current (CC)-Constant Voltage (CV) charge/discharge: 0.3C / 0.3C; 0.5C / 0.5C; 0.8C / 0.8C; 1C / 1C

Due to damage during transport and installation, only four of eight prepared cells, identified as $T1 - T4$, could be used as test cells. A non-drilled cell $V1$ is cycled at the same level in the climate chamber for validation purposes.

Fig. 4.19a illustrates the measured core and surface temperature, cell voltage, and current during the experiment series at 20 °C ambient temperature of cell T_1. The experiment series includes 12 CC-CV charge and discharge cycles with different C-rates.

During the experiments, a reduction in the capacity of cells can be observed, as shown in 4.19b. The figure illustrates the measured capacity of each cell during each charge and discharge cycle. Nevertheless, the validation cell $V1$ shows a significantly lower reduction in capacity compared to cells $T1 - T4$ with an inserted core temperature sensor. The first cycle illustrates a significant difference between test and validation cells. Possible explanations for this difference are loss of the electrolyte, insertion of the sensor, which represents a



(a) (b)

Fig. 4.19. Measurement data of surface and core temperature, voltage, and current at 20 °C ambient temperature (a); Change of discharge capacity during performed experiments; Comparison between validation cell V1 and drilled test cells T1 – T4 (b); Significant difference between capacities of validation cell and test cells

foreign body, and changes in the cell chemistry due to reaction with the outside air during opening. The figure also illustrates a difference between the prepared cells, especially at the end of experiments.

Model Proposal: Long Short-Term Memory (LSTM)

In our work, we proposed an interconnected LSTM network to estimate surface and core temperature as two parameters affecting each other. The architecture of LSTM has been described in Section 2.3.3 on page 29. The approach of employing interconnected LSTM networks to estimate parameters that are related to each other has been utilized in previous studies. Nguyen Van and Ta Quang [269] have introduced an interconnected LSTM to estimate the battery's internal resistance and the SOH with temperature, current, and voltage as input. Furthermore, SOC and SOH influence each other. This effect was successfully employed for a state estimation utilizing interconnected LSTM [270]. The results of the mentioned studies indicate the potential of interconnected LSTM networks for state estimation of interdependent battery states. Fig. 4.20 shows the architecture of the proposed network. The input of the first LSTM model is the ambient temperature T_{amb}, voltage V, and current I. The network consists of LSTM cells, a hidden FNN layer, and an output neuron, which outputs the surface temperature T_S. The second network receives the first output in addition to the input of the first network. The core temperature, denoted as T_{core}, results from a hidden FNN layer and an output neuron.

Model Proposal: Kolmogorov–Arnold Networks (KAN)

KAN, as described in Section 2.3.3 on page 27, has been identified particulary effective for high interdependency time-series data [137], [138]. High interoperability and accuracy in

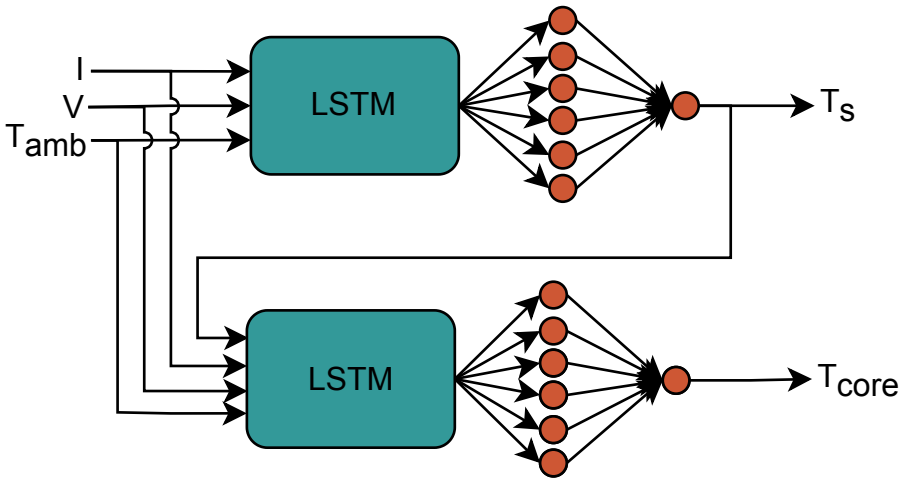


Fig. 4.20. Structure for estimating surface and core temperature using LSTM network

time-series data prediction could be demonstrated. Moreover, KAN achieves fewer parameters combined with comparable or higher accuracy compared to widely used architectures such as FNN and LSTM. This work uses a KAN model to predict the core and surface temperatures of battery cells within a single model. The model takes as input a vector x, comprising battery current (I_t), voltage (V_t), and ambient temperature ($T_{amb,t}$) of three sequential time steps: t_n, t_{n-1}, and t_{n-2}. The input vector is described as follows:

$$\begin{aligned} x = [&I_{t_n}, \quad I_{t_{n-1}}, \quad I_{t_{n-2}}, \\ &V_{t_n}, \quad V_{t_{n-1}}, \quad V_{t_{n-2}}, \\ &T_{amb,t_n}, \quad T_{amb,t_{n-1}}, \quad T_{amb,t_{n-2}}] \end{aligned} \tag{4.1}$$

Model Implementation

For implementation, Python 3.10[1] is used as a programming language. As deep learning framework PyTorch 2.3.1[2] comes into account, and to implement KAN pykan 0.0.5[3] is used. A grid search was conducted to determine the hyperparameters for both network architectures. For the LSTM, the grid search explored variations in hidden layers (from 1 to 3) and hidden sizes (25, 50, and 75) across its estimation networks. As for the KAN, the search focused on the sizes of the two hidden layers, testing configurations of 8/4, 16/8, 32/16, and 64/32. Table 4.7 shows the structures of the chosen hyperparameter configurations.

Comparative Analysis and Discussion

To evaluate and discuss the proposed KAN and LSTM models to estimate surface and core temperature employing current, voltage, and ambient temperature, two different datasets

[1] https://www.python.org/downloads/release/python-3100/
[2] https://pypi.org/project/torch/2.3.1/
[3] https://pypi.org/project/pykan/0.0.5/

Table 4.7: Architecture of Models

Layer	*LSTM*	*KAN*
1	LSTM(input=3, hidden size = 25, layers = 1)	KAN(9)
2	Dense(32)	KAN(32)
3	Dense(1)	KAN(16)
4	LSTM(input=4, hidden size = 50, layers = 1)	KAN(2)
5	Dense(32)	
6	Dense(1)	

were utilized to train and evaluate independently.

The CC-CV Dataset contains the above-mentioned and introduced measurements of core temperature experiments.

The Drive Cycle Dataset was introduced by Samanta and Williamson [271] and consists of CC-CV cycles up to 4 C at ambient temperatures between −20 °C – 40 °C. Furthermore, drive cycle data, Urban Dynamometer Driving Schedule (UDDS), and Highway Fuel Economy Driving Schedule (HWFET), at 25 °C 30 °C, and 40 °C ambient temperature. Cylindrical Samsung INR21700-40T cells were used as test batteries. The dataset contains data with a measurement frequency of 1 Hz. The model is evaluated once with the UDDS driving cycle at 25 °C and once more with HWFET 30 °C. The corresponding driving cycle is excluded from the respective training in this case.

CC-CV Dataset The dataset is split seeded randomly for training and validation, ensuring a robust evaluation of the models. The split was performed by cell and ambient temperature. Per ambient temperature, one cell is chosen for validation and three for training. Table 4.8 presents the performance of the two proposed models, KAN and LSTM, in estimating core and surface temperature. The table comprehensively shows the models' performance across all ambient temperatures. Notably, KAN consistently outperforms the LSTM model in MAE and RMSE, demonstrating its superior accuracy and reliability. KAN achieved an MAE of 0.579 °C and 0.353 °C for core and surface temperature, respectively. At the same time, LSTM showed an MAE of 0.613 °C for core and 0.370 °C for surface temperature.

Table 4.8: MAE, and RMSE Core and Surface Temperature per Model of drilled cells

	KAN	*LSTM*
MAE		
Core	0.579	0.613
Surface	0.353	0.370
RMSE		
Core	0.814	0.875
Surface	0.519	0.520

In addition to the overall accuracy, in Table 4.9, the MAE performance of the KAN and LSTM model is shown depending on the ambient temperatures. The MAE for the surface temperature estimated by the KAN model ranges from 0.230 °C at 40 °C up to 0.422 °C at an ambient temperature of −20 °C. In general, the MAE of the estimated surface temperature of both models is much less than 0.5 °C irrespective of the ambient temperature. In nine out of 14 metrics, KAN outperforms the LSTM model. Noticeabl is the high error in estimating core temperature at 30 °C ambient temperature. The high inaccuracy can be explained by

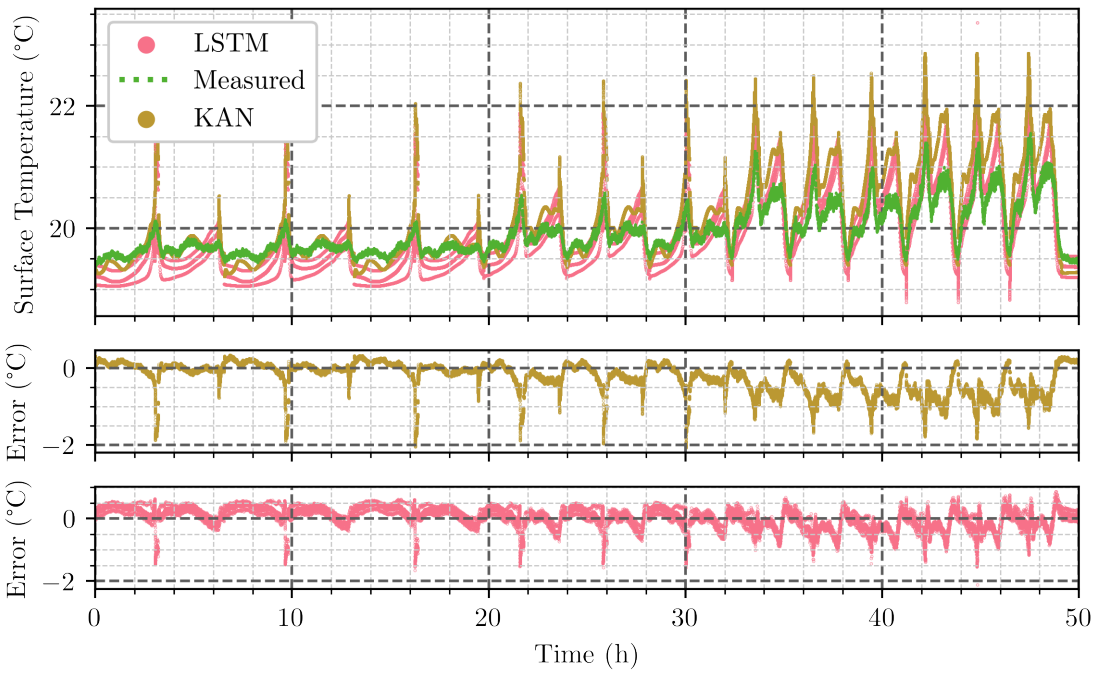


Fig. 4.21. Measured surface temperature (green) compared with KAN (brown) and LSTM (brown) estimation models at 10 °C ambient temperature

the high difference in ageing, especially at an ambient temperature of 30 °C, as seen by its difference in capacity in Fig. 4.19b on page 82.

Table 4.9: MAE Surface and Core Temperature per Model in °C at ambient temperatures; Higher accuracy is highlighted in bold

	Surface		*Core*	
Ambient Temperature	*KAN*	*LSTM*	*KAN*	*LSTM*
40	**0.230**	0.293	**0.464**	0.768
30	**0.363**	0.473	1.403	**1.143**
20	0.325	**0.264**	**0.207**	0.419
10	**0.403**	0.413	0.714	**0.639**
0	**0.267**	0.306	0.532	**0.528**
-10	0.375	**0.362**	**0.285**	0.355
-20	**0.422**	0.432	**0.493**	0.571

Figure 4.21 displays the estimated surface temperature, and Figure 4.22 presents the estimated core temperature, both at an ambient temperature of 20 °C. These figures show that the estimated temperatures align closely with the measured results. The figures also include the error as an additional graph. The error is illustrated as the difference between the measured and estimated values. The error is increasing for core and surface temperature at both estimation models with higher charge and discharge rates, which causes higher temperature increases in core and surface.

In addition to validating the proposed models employing the prepared cells, the model is also validated through the validation cell. The validation cell has not been drilled and does not include a core temperature sensor. Due to this, only the surface temperature estimation is validated. Table 4.10 presents the MAE and RMSE of the KAN and LSTM model in total and through the performed ambient temperatures. A reduced accuracy compared to

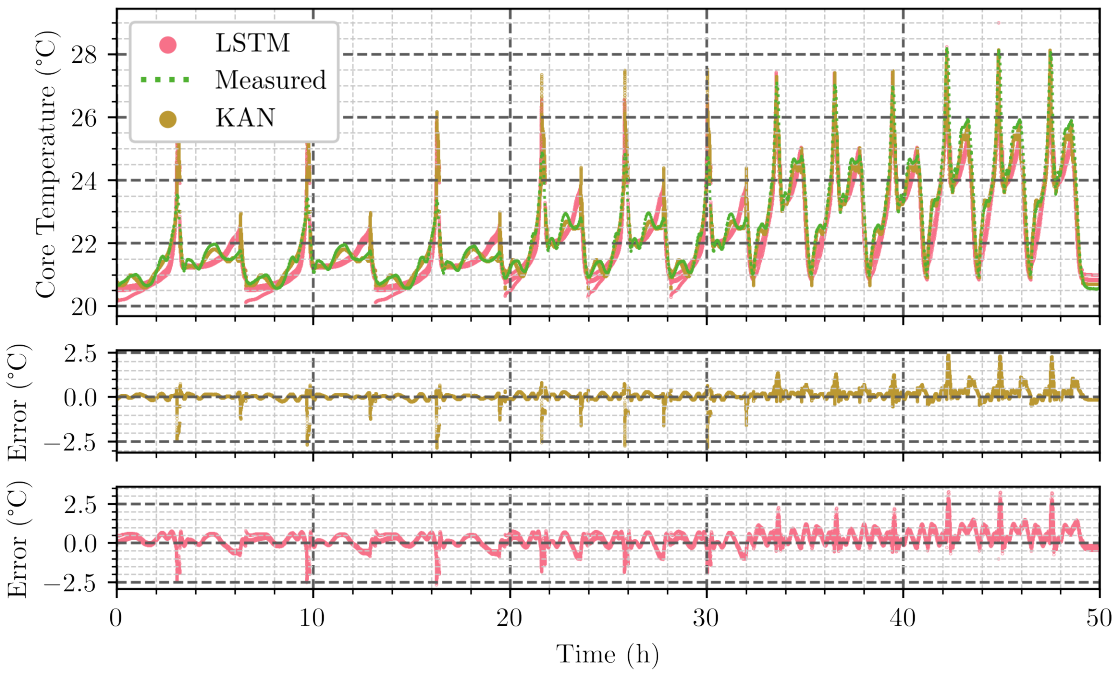


Fig. 4.22. Measured core temperature (green) compared with KAN (brown) and LSTM (red) estimation models at 10 °C ambient temperature

the drilled cells, as shown in Table 4.9, can be observed. A minor difference can be seen for the KAN model at ambient temperatures of 20 °C and higher. The MAE difference between the results of prepared cells and the validation cell is below 0.1 °C. At decreasing temperatures, the error increases. The LSTM model indicates a generally higher difference in error. Nevertheless, the apparent difference in accuracy between cells with and without an inserted temperature sensor can be explained by the physical effect on the cell caused by inserting the sensor. This effect has already been observed with the difference in capacity over time, as shown in Fig. 4.19b.

Drive Cycle dataset The precision of the proposed KAN and LSTM under dynamic drive cycle conditions are evaluated with the dataset Samanta and Williamson [271] proposed.

Table 4.10: MAE, and RMSE Surface Temperature in °C per Model and ambient temperature of Validation Cell

	MAE		*RMSE*	
Ambient Temperature	*KAN*	*LSTM*	*KAN*	*LSTM*
-	0.578	0.619	0.948	0.915
40	0.285	0.360	0.389	0.510
30	0.369	0.383	0.557	0.528
20	0.373	0.583	0.567	0.823
10	0.856	0.821	1.191	1.090
0	0.552	0.579	0.874	0.915
-10	0.609	0.629	0.914	0.942
-20	0.889	0.877	1.340	1.357

The training and validation is performed within three different settings, as follows:

UDDS 25 °C

Traininig CC-CV cycles, UDDS (30 °C, 40 °C), HWFET (25 °C, 30 °C, 40 °C)

Validation UDDS 25 °C

UDDS 25 °C (Unknown UDDS profile)

Training CC-CV cycles, HWFET (25 °C, 30 °C, 40 °C)

Validation UDDS 25 °C

HWFET 30° C

Training CC-CV cycles, UDDS (25 °C, 30 °C, 40 °C), HWFET (25 °C, 40 °C)

Validation HWFET 30 °C

Table 4.11 gives MAE and RMSE of core and surface temperature estimation under the three training and validation settings. KAN indicated higher accuracies at the UDDS drive cycle compared to the LSTM performance. There is only a slight deviation in the observed performance parameters for already known UDDS driving cycles compared to the unknown. For example, there is only a deviation of 0.157 °C RMSE for the surface temperature at KAN. Within HWFET drive cycle LSTM outperforms KAN. The RMSE of 1.150 °C core and 1.007 °C surface temperature indicate better performance of LSTM under high speed conditions.

Table 4.11: MAE, and RMSE Core and Surface Temperature per Model at variations of training in °C

	UDDS 25 °C		*UDDS* 25 °C *(Unknown UDDS profile)*		*HWFET* 30 °C	
	KAN	*LSTM*	*KAN*	*LSTM*	*KAN*	*LSTM*
MAE						
Core	**0.610**	0.901	**0.598**	0.887	0.936	**0.727**
Surface	**0.411**	1.206	**0.481**	1.030	0.775	**0.725**
RMSE						
Core	**0.735**	1.354	**0.787**	1.433	1.507	**1.150**
Surface	**0.520**	1.459	**0.677**	1.636	1.305	**1.007**

Figure 4.23 presents the measured core and surface temperatures during the UDDS drive cycle at 25 °C (in green) alongside the estimated values from the KAN model (in brown) and the LSTM model (in pink). The training data for these models consists of the data above described as UDDS 25 °C. The figure indicates an increase in surface and core temperatures of 4.0 °C and 4.9 °C, respectively. The proposed KAN model captures the battery's behaviors and dynamics more accurately than the interconnected LSTM. While both models effectively modulate the rising temperatures, the LSTM exhibits significantly higher fluctuations that do not accurately reflect the battery dynamics. Although the KAN model also shows higher dynamics compared to the measured temperatures, these fluctuations are less pronounced than those in the LSTM.

Computational Cost Computational cost is a critical component of evaluation when it comes to applying state estimation models in addition to the previously discussed accuracy of models. Table 4.12 contains the measured runtime for estimation with LSTM and KAN

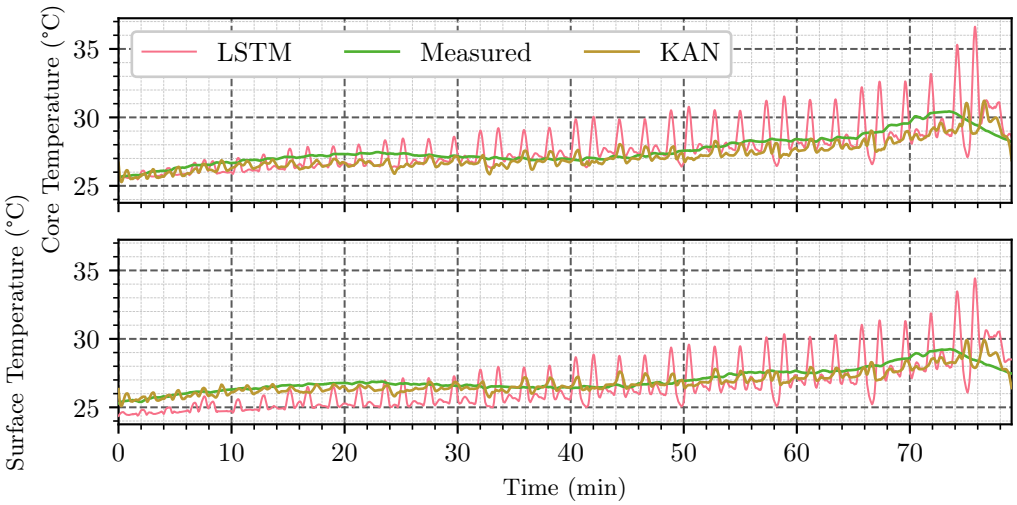


Fig. 4.23. Measured core and surface temperature (green) compared with KAN (brown) and LSTM (pink) estimation models at UDDS 25 °C [30]

employing two different CPU setups. The results show, with 0.55 ms to 0.58 ms LSTM outperforms KAN, which requires 2.92 ms to 3.28 ms. The results show that better-performing CPU setups could reduce the computational cost for both models. Additionally, the KAN implementation remains in its early stages of development. As the implementation continues to evolve and undergo further optimization, significant performance improvements are anticipated in the months and years ahead.

Table 4.12: Computational cost comparison of models and different hardware setups

CPU-Setup	*LSTM*	*KAN*
Intel Xeon Gold 6338 2G	0.55 ms ± 0.10 ms	2.92 ms ± 0.45 ms
Intel Core i9-10885H	0.58 ms ± 0.04 ms	3.28 ms ± 0.34 ms

Although KAN provides enhanced predictive capabilities and greater flexibility, particularly in managing complex nonlinear relationships, these benefits come with the trade-off of higher computational time compared to the proposed LSTM model.

4.4 Lossy Data Compression

Upon validating the feasibility of cloud-based state estimation for battery cells, the focus shifts to minimizing the required bandwidth and storage. Therefore, in the following section, we propose and evaluate a lossy data compression algorithm.

The high amount of data points required for cloud-based fleet monitoring, digital twinning, and cloud-based BMS has already been discussed in previous studies [24], [272]. The main focus of recent research has been to increase the accuracy of estimation methods [273]. Thus, the requirements needed for storage and bandwidth increase. The compression of the data can address this issue. Data compression can be defined as reducing the required storage to represent the same data [274]. In the context of implementing cloud-based services for an entire fleet of vehicles; data compression is required. It will reduce the required bandwidth for each vehicle and the cost required for storage to archive and process the data. Furthermore, computationally efficient compression methods can also be applied to the on-board BMS and efficiently use the system's available local bandwidth. This enables

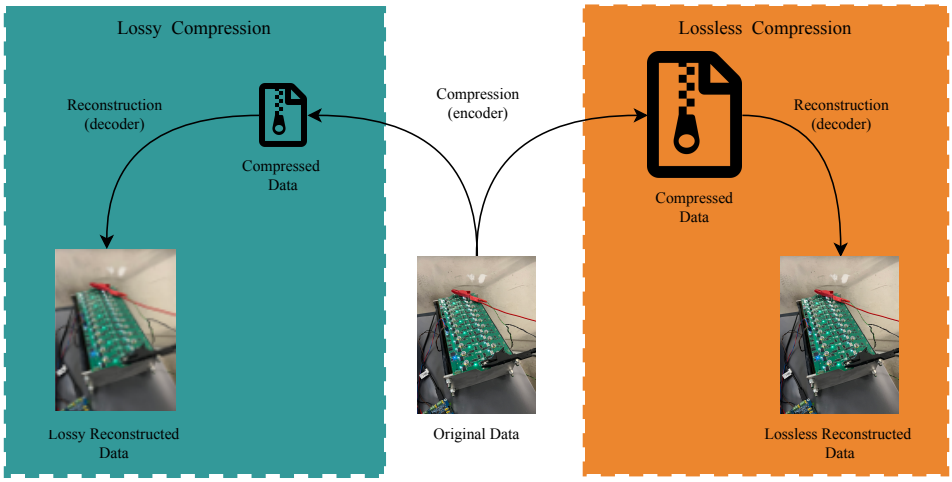


Fig. 4.24. Comparison between lossless and lossy compression. Lossless decompression is similar to the original data; Lossy decompression contains some errors depending on the used method and CR

higher measurement frequencies, as the limited bandwidth is better utilized. Data compression can be implemented in two main approaches: lossless and lossy data compression. Lossless compression permits the complete recovery of the initial data, and it is primarily used for data that need to stay unchanged after compression, such as textual documents or database records containing vital information. The compression rate of the data within lossless methods is technologically limited.

The Compression Ratio (CR) and Rate of Compression (RoC) are defined as follows [275]:

$$\mathrm{CR} = \frac{\text{Original Size}}{\text{Compressed Size}} \tag{4.2}$$

$$RoC = 1 - \frac{1}{\mathrm{CR}} \tag{4.3}$$

An evaluation of various lossless compression algorithms has demonstrated an average CR of 7.088 and a RoC of 85.9 % for PPMonstr [276], outperforming the performance of other compression algorithms across multiple datasets [277]. A lossy data compression approach does not allow the total reconstruction of the original data. An error can be caused during reconstruction, resulting in a different outcome compared to the original data. The difference between lossy and lossless data compression approaches is illustrated in Fig. 4.24. The compressed data requires less storage in the case of lossy compression than in lossless compression. Therefore, the decompressed data contains errors compared to the original data. Lossless decompressed data are always identical to the original data. However, utilizing both compression approaches depends on the application and use case.

Multiple lossy compression techniques have been proposed and reviewed [274]. The authors of the review study classified the methods into the following types:

Transform: The measured signal's domain change. An example is the change of time dimension to frequency dimension by Fourie Transformation [278].

Artificial Intelligence (AI): Neural networks can compress and decompress data as encoders and decoders. For example, image compression by the fully convolutional network (FCN) model [279].

Interpolation: The interpolation is defined by approximating new data points based on a set of known data points. This method can be used to reconstruct data based on the approximation of representative data points.

Hybrid: Combining different lossy and lossless compression methods.

In addition to lossless and lossy compression, Chiarot and Silvestri [275] introduce additional classifications for compression algorithms:

Adaptive and non-adaptive algorithms: These refer to whether a model's training is necessary to apply the compression method. Non-adaptive algorithms rely on model training.

Symmetric and asymmetric: An algorithm is symmetric if the decoder executes the same steps as the encoder but in reverse; otherwise, the algorithm is asymmetric.

Many compression methods are known in audio and image compression, as well as for IoT [280]–[282]. Several research studies have already been published in the BMS field. One of the most critical features of an BMS is the estimation of battery cell states based on measured values of current, voltage, and temperature [283]. Measuring, transmitting, processing, and storing these values is essential. Historical data points are essential to enable customer service and forecast cell usability in second-life applications. Hence, the use of efficient data compression methods is necessary.

Zheng et al. [284] introduced a dynamic recording frequency that adjusts based on the battery pack's dynamics to minimize the number of necessary data points. They successfully decreased the recording frequency from 4 Hz to 2 Hz and interpolated the missing measurements, with an average voltage difference of 0.2 mV.

Zhou et al. [285] proposed a compression method that utilizes the frequency division model of a battery pack. This model regards the battery pack as a whole and compares the average voltage difference between each cell and the whole pack. The method uses a Thevenim model to reconstruct the voltage measurements. Figure 2.3 on page 12 has previously shown a Thevenim model. It is a computational, low-cost, and simple model. Eight battery cells have been utilized to evaluate the proposed method. A mean RMSE of 9.7 mV at a recording frequency of 0.2 Hz could be observed. This recording frequency implies a RoC of 28 %.

A hybrid method based on Artificial Intelligence (AI) and interpolation is proposed in [286]. A neural network approximates voltage measurements based on low-frequent measured values. Also, a battery model is utilized to reconstruct these values based on the measured current values of the battery cell. Different RoCs have been tested for evaluation. The reconstruction RMSE of 2.9 mV, 4.16 mV, and 10.01 mV could be observed at RoC of 90 %, 95 %, and 99 %.

In our work [25], [29], we proposed a lossy compression algorithm to compress and decompress voltage measurements. This algorithm utilizes polynomial regression to approximate voltage measurements of individual cells depending on the current in a battery pack of a series of connected cells. The proposed lossy compression algorithm is an adaptive, asymmetric algorithm. To the best of our knowledge, this compression method is the first adaptive, asymmetric compression algorithm in power electronics and electric vehicles. The individual steps of the method can be seen in Fig. 4.25.

4.4.1 Novel Compression Method for Voltage Measurements

Figure 2.3 on page 12 illustrates ECM higher-order models with multiple RC links to describe the different dynamics in battery behaviour. In Eq. (2.1) on page 11, the voltage

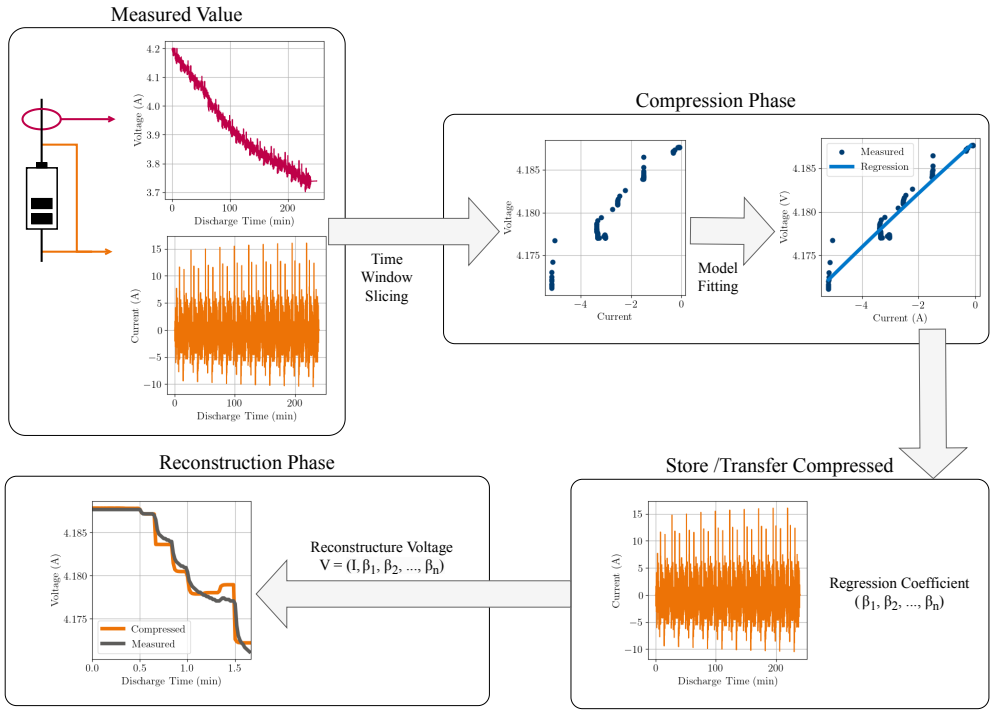


Fig. 4.25. Proposed method to compress voltage measurements to transfer and store battery measurements

u_B of a battery cell has been defined depending on the open circuit voltage V_{OC} as a function of several parameters. These parameters are the charge of the cell SOC, i_B the battery current, T the discharge/charge time, internal resistance R_0, and the capacity C_k and resistance R_k of the RC link k.

In the first step, we proposed that the change in SOC within a short time can be ignored [29]. In addition, the changes in the parameters in the ECM model caused by changes in temperature, SOC, and cell ageing are negligibly small. In that case, the voltage u_B can be described as a function of current i_B and the parameters β_i where $i \in [0; k]$:

$$u_B \approx \beta_0 + \beta_1 i_B + \cdots + \beta_k i_B, \tag{4.4}$$

In our work [25], we improved the previous method proposed in [29]. The impact of the time parameter T was considered in this algorithm. This permits the voltage to be approximated as a function of current and time:

$$\begin{aligned} u_B \approx \beta_0 - \beta_{R,1} i_B - \cdots - \beta_{R,k} i_B \\ - \beta_{T,1} i_B T - \cdots - \beta_{T,k} i_B T, \end{aligned} \tag{4.5}$$

$$i_b R_n \left[1 - \exp\left(-\frac{T}{\tau_n}\right)\right] \approx \beta_{R,n} i_B - \beta_{T,n} i_B T, \tag{4.6}$$

where $\beta_{R/T,[1;k]}$ defines an approximation of the RC link and β_0 of $V_{oc}(SOC)$.

The proposed method in this work is to transfer and store the current measurement i_B and polynomial function parameters β. The voltage u_B is reconstructed based on the current and polynomial function parameters. Regression is a statistical method to identify these

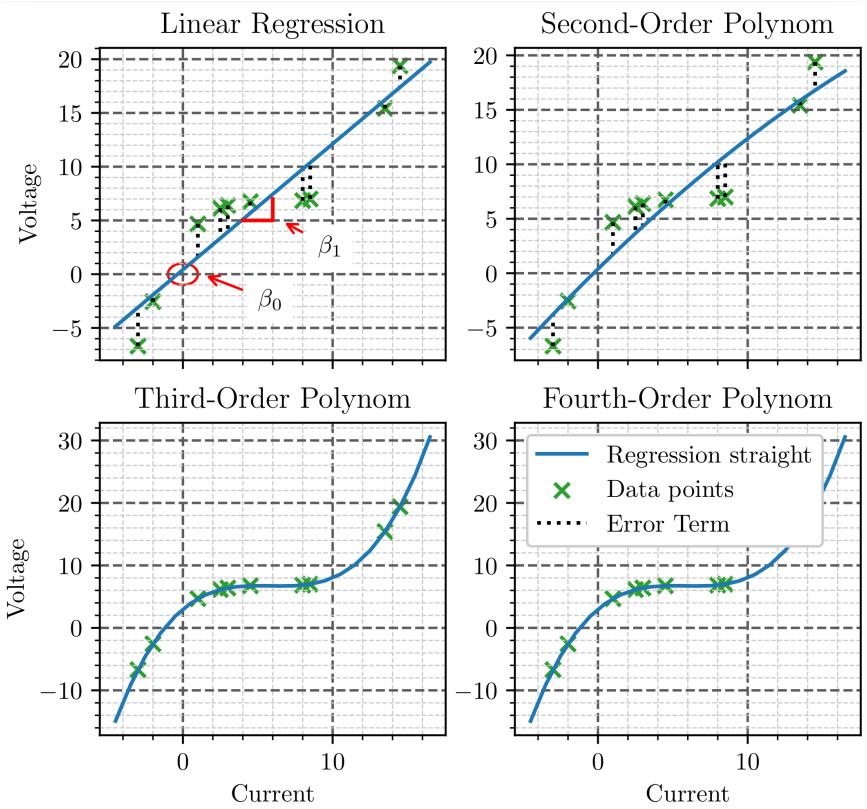


Fig. 4.26. Visualization of variations of regression from linear regression up to fourth-order polynomial Regression; Decreasing errors at higher orders of the regression are noticeable

function parameters. Regression describes the relationship between the dependent variables and one or more independent variables [287]. Fig 4.26 shows data points $P = (x, y)$, with the independent variable x and the dependent variable y. In the given example, current is represented as the independent variable on the x-axis, while voltage is depicted as the dependent variable on the y-axis. The points are illustrated as green data points, the regression as the blue line, and the error as a black dotted line. In the figures, the same points P are illustrated. The figure illustrates the different regressions, such as linear regression, also known as first-order regression, second-order, third-order, and fourth-order regression. Up to third-order, the error decreases with increasing orders. A difference between the third- and fourth order can not be noticed. The linear regression model is defined as:

$$y = \beta_0 + \beta_1 x + \epsilon, \tag{4.7}$$

where the intercept is defined by β_0, the slope β_1, and the random component ϵ. The most widely used method to identify linear regression parameters is the ordinary least squares method [104], [287], [288].

The correlation between current, time, and voltage can not be described linearly due to the approximation Eq. (4.5). Higher-order polynomial regression, as illustrated with second-, third-, and fourth-order regression in Fig. 4.26, can be utilized to describe such higher-order correlations. Acito et al. [289] define a $(k + 1)$th order regression, and Ryan [290] describes the employment of the least squares method for higher-order regression.

Higher-order regression is defined as the following:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_k x^k + \epsilon, \tag{4.8}$$

As previously stated, polynomial regression facilitates the characterization of nonlinear dynamics within a battery cell over a specified temporal duration. For instance, a 4th-order polynomial model requires 16 coefficients to describe the relationship between current, time, and voltage.

For instance, compressing 100 voltage measurements using a fourth-order polynomial implies that 100 measurements are represented with 16 function parameters. This leads to a RoC of 1 - (16/100) = 0.84 or 84 %.

Karimi et al. [291] highlight the influence of high currents through parameters of the ECM. This factor can not be targeted within the polynomial model. Thus, an adaptive time window is considered for this phenomenon [25]. This is facilitated by an adjustable voltage threshold V_{th}. The voltage is regulated by various resistances dependent on current, SOC, and temperature. Consequently, the threshold is applied to the voltage rather than the current.

The compression algorithm applying the adaptive time window is proposed in our work [25] and shown in Algorithm 3, where the input variables are I and V: list of current and voltage measurements, and V_{th} the voltage threshold.

Algorithm 3 Voltage compression within time window

```
1: procedure COMPRESSION(I, V, V_th)
2:     p ← amountRegressionParameters()
3:     ΔV ← V_t − V_{t+1}
4:     x ← Σ_{v ∈ΔV} 𝟙_ΔV(v)                          ▷ Indicator function
5:     if x ≤ p then
6:         return Compress based on ΔV
7:     end if
8:     if |ΔV_t| ≥ V_th, where t ∈ [0, length(V)] then
9:         i_{V_th} ← indexOfFirstElement(|ΔV_t| ≥ V_th)
10:        if i_{V_th} ≤ p then
11:            return [V[i]|0 ≤ i ≤ i_{V_th}]
12:        else
13:            return PCCompress([V_0, V_{i_{V_th}}], [I_0, I_{i_{V_th}}])
14:        end if
15:    end if
16:    return PCCompress(V, I)
17: end procedure
```

Lines four to seven in the algorithm deal with the case of a small number of different voltage measurements within the time window. When the number of measurements with a change compared to the previous measurements falls below the number of regression parameters, only the timestamps corresponding to the measurements with a change are stored, and no regression is used.

The influence of high dynamic events during measurements has to be reduced. Thus, the adaptive window size used within the regression modelling must employ the threshold V_{th} as outlined in lines 8-15 in the Algorithm 3. A special case is handled in lines 10 to 12 of the Algorithm. If the index of the threshold is lower or equal to the number of regression parameters, the voltage measurements are not compressed and stored in the original.

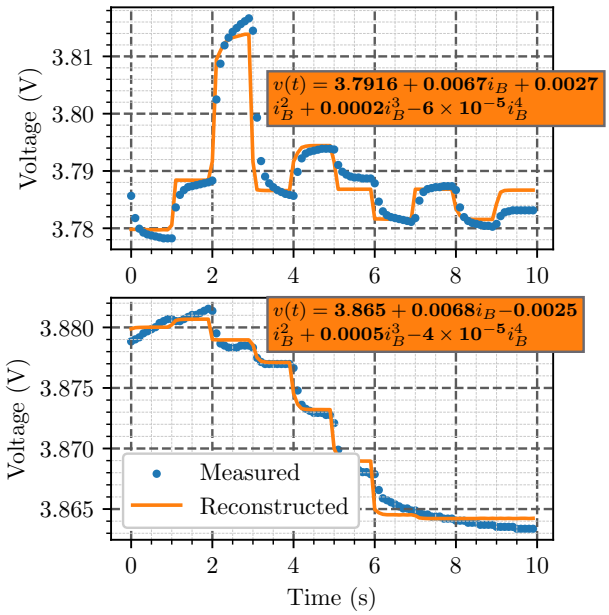


Fig. 4.27. Reconstruction of voltage values after compression and curve fitting; The equations (orange box) illustrate the fourth-order regression employed for compression; achieved RoC of 95 %; Compression only based on current [29]

The PCCompress function, which utilizes the ordinary least squares method, identifies the higher-order polynomial regression parameters. This function's output encompasses the regression model and the associated current values.

Reconstruction Method

Fig. 4.27 illustrates the temporal evolution of reconstructed voltage through two time windows of 10 s each. The adaptive, non-symmetrical lossy reconstruction of the voltage data $u_B(i_b, t, \beta)$ is executed utilizing the time and current values, and the polynomial coefficients β as shown in Eq. (4.8). The figure illustrates the slight difference between measured and reconstructed values. For simplicity and readability, the illustrated polynomial regression utilizes only the regression of voltage and current, as proposed in the initial study [29].

4.4.2 Accuracy Evaluation of Lossy Data Compression

For the evaluation of the proposed Algorithm 3, three datasets are used:

Pulsed dataset: 12 min charge and discharge pulses of 10 A [25].

Aging dataset: Data of 28 high-energy round cells that were tested for ageing. Drive cycles are applied to age the cells at an ambient temperature of 25 °C [292].

Temperature dataset: Data of the drive cycles Urban Dynamometer Driving Schedule (UDDS), Highway Fuel Economy Test (HWFET), LA92, US06, and various random combinations (referred as mix 1-8) at different ambient temperatures between 40 °C and −20 °C [236].

The RMSE and MAE are used as evaluation matrices. Furthermore, as seen in Algorithm 3, there is a difference between the set RoC and the achieved RoC. The set RoC indicates the amount of regression coefficients and the size of the defined window. A window size of

50 data points causes a set RoC of 68.0 %. Due to the voltage threshold, the achieved RoC can be smaller than that.

Pulsed dataset

Battery behaviour under pulsed conditions is of high interest in research and development. Hybrid Pulse Power Characterization (HPPC) can be used to identify the parameters and states of the battery, such as SOC [293]. Furthermore, the characterization under pulsed conditions environment is crucial for reconfigurable batteries because of the high frequent switching of the reconfigurable battery system [294]. Proper control of reconfigurable batteries depends on accurate measurements; however, local bus connections to transmit the data have limited bandwidth, which causes constraints on the maximum measurement frequency. The proposed algorithm is evaluated under pulsed conditions to address these challenges. Experimental data encompassing a duration of 130 min with a measurement frequency of 30 Hz for current and voltage. The data contains charge and discharge pulses of 10 A over 12 min. Figure 4.28 illustrates the measured (brown) and reconstructed voltage (pink). In addition, the reconstruction error (green) in mV is illustrated as a scatter plot. The set window size is 500 data points, a RoC of 96.8 %, and the achieved RoC is 96.7 %. The maximum reconstruction error observed is 28 mV and is shown in the zoom-in of the Figure. The measurements with the highest reconstruction error observed are at the time steps of the pulses. Apart from the described high error events, a reconstruction error remains consistent at 0.1 mV and less.

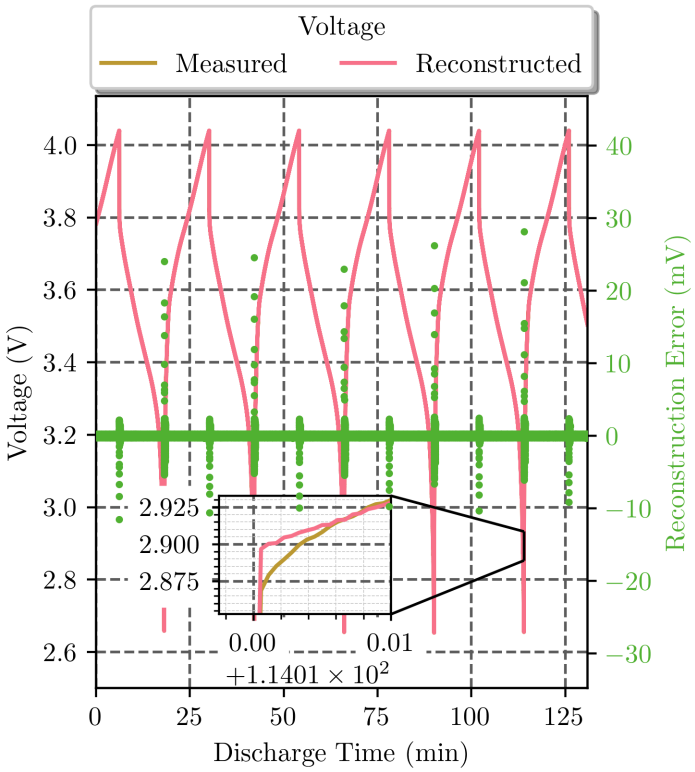


Fig. 4.28. Measured (brown) and reconstructed voltage (pink); reconstruction error (green) in mV of 130 min with 30 Hz measurement frequency and achieved RoC of 96.7 % and a MAE of 0.11 mV, a set window size of 500;

In addition to the error at each time step, in Table 4.13 the MAE and RMSE depending

Table 4.13: MAE, RMSE, and achieved RoC of reconstructed values under pulsed conditions regarding the set RoC calculated by the error between original and reconstructed values

Set RoC (%)	*Achieved RoC* (%)	*MAE* (mV)	*RMSE* (mV)
68.0	68.0	0.08	0.11
84.0	84.0	0.08	0.12
96.8	96.7	0.11	0.31
98.4	98.4	0.15	0.58
99.2	99.2	0.31	1.19

Table 4.14: MAE and RMSE, of reconstructed values across aging datasets regarding the set RoC calculated by the error between original and reconstructed values

Set RoC (%)	*Achieved RoC* (%)	*MAE* (mV)	*RMSE* (mV)
68.0	63	0.26	0.48
84.0	77.0	0.39	0.72
96.8	88.0	1.32	2.27
98.4	89.0	1.83	3.44
99.2	89.6	2.20	4.13

on the set RoC between 68.0 % up to 99.2 % is shown. The set RoC comes from the set window sizes of 50, 100, 500, 1000, and 2000 datapoints. Caused by the low dynamics and the long times of constant current, the difference between set and achieved RoC is not significant. The low errors, 0.31 mV MAE and 1.19 mV, at the highest evaluated set RoC of 99.2 %, do show the high potential of the proposed method to compress pulsed data.

Aging dataset

During battery cell ageing experiments, large amount of data is generated. This gives great potential to reduce storage costs with an accurate lossy data compression method. Several window sizes have been set to evaluate the accuracy within the dataset published by Jost et al. [292]. Table 4.14 shows the MAE, RMSE, and achieved RoC depending on the set RoC. The observed difference between the set and the achieved RoC is relatively high, at set RoC 99.2 %, a difference of 10.6 percent points. A lower threshold could be set to reduce the difference, decreasing the reconstruction error. This shows a high flexibility and practical adaptation of the proposed method.

Temperature dataset

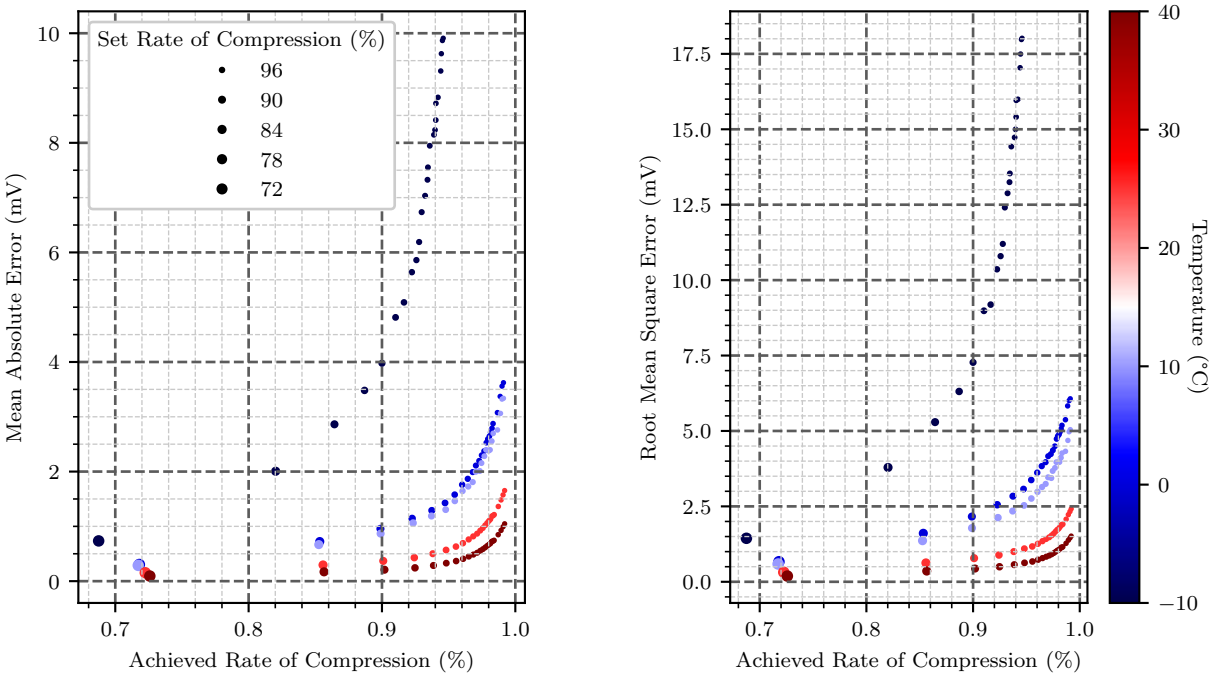
The temperature dataset [236] allows the evaluation and discussion of the compression algorithm under several conditions. The dataset enables accurate comparison at different ambient temperatures and drive cycles. Furthermore, the effect of different thresholds is discussed for the US06 drive cycle at an ambient temperature of 25 °C.

1. Temperature Conditions

The proposed algorithm has been evaluated previously under pulsed and ageing conditions. As ambient and core temperatures greatly influence the electrochemical processes within a cell [295], therefore it is important to examine the reconstruction error at several ambient temperatures. Data of the LA92 drive cycle is utilised at temperatures from −10 °C to 40 °C. In addition to temperatures set RoC between 68.0 % and 99.2 % are evaluated.

Fig. 4.29 shows MAE and RMSE of this experiment. It can be seen that the higher the temperature, the higher the accuracy. A RMSE of 1.52 mV could be achieved at 40 °C with a set RoC of 99.2 %. This is significantly better than the results of previous studies [285], [286]. Also, the figure shows that the achieved RoC decreases with decreasing temperature. Achieving low reconstruction MAE and RMSE becomes increasingly difficult as the temperature decreases. The errors of achieved RoC of 82 % are higher at −10 °C than all the compressions evaluated at temperature ranges between 30 °C and 40 °C.

A significant shift to lower achieved RoC is seen mainly in the data at an ambient temperature of −10 °C compared to higher temperatures at similar set RoC (size of the marker). This huge difference between set and achieved RoC cannot be seen with any other ambient temperature.



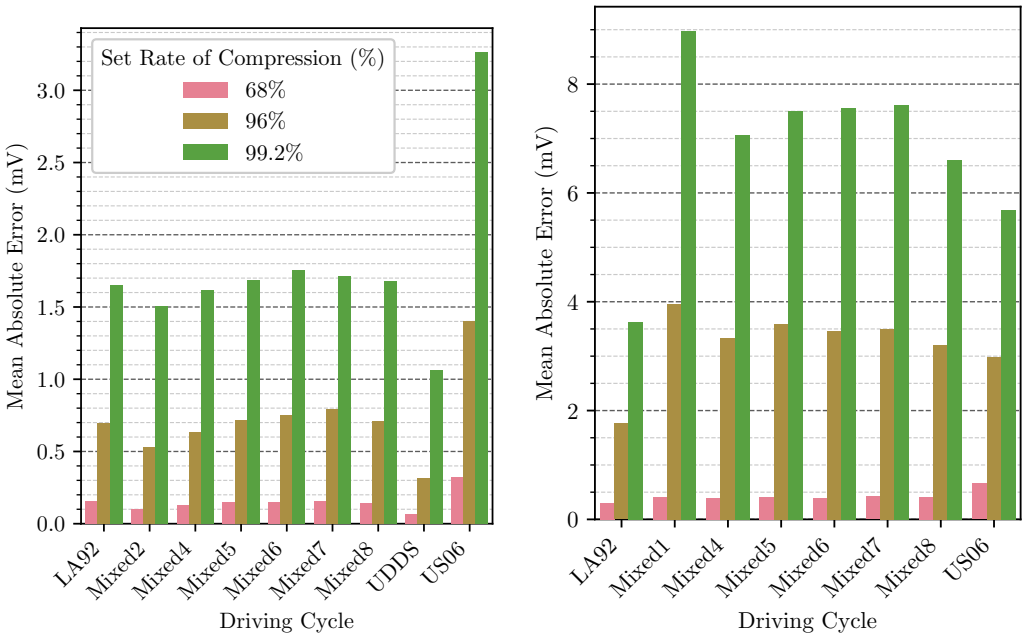
(a) (b)

Fig. 4.29. MAE (a) and RMSE (b) depending on the Achieved Rate of Compression over LA92 drive cycle by set RoC (marker size). Decreasing temperature leads to less accuracy and lower RoC; Significant correlation between temperature and MAE/RMSE at same set RoC can be observed

2. Drive Cycle Conditions

The impact of different driving cycles on the accuracy of reconstruction errors is evaluated in the following section. The MAE of different drive cycles at different set compression rates are shown in Figs. 4.30a and 4.30b. The US06 drive cycle exhibits the highest MAE, registering values of 3.3 mV, 1.4 mV, and 0.3 mV under 25 °C ambient temperature conditions, as shown in Fig. 4.30a. These errors are significantly worse than those in the compared drive cycles at the same ambient temperature. Under freezing conditions, shown in Fig. 4.30b, the discrepancy of the US06 drive cycle is not observable. At 0 °C, the US06 drive cycle demonstrates the second-highest accuracy following LA92.

3. Threshold Conditions



(a) (b)

Fig. 4.30. MAE at set RoC 68 %, 96 %, and 99.2 % and drive cycles at 25 °C (a) and 0 °C (b) ambient temperature; At 25 °C, the data for US06 and UDDS, and at 0 °C, the data from Mixed1 and LA92 diverge from the overall trend at all set RoCs

Table 4.15: Threshold depending MAE and RMSE in mV for drive cycle US06 at 25 °C and set RoC of 99.2 %

Threshold (mV)	*MAE* (mV)	*RMSE* (mV)	*Achieved RoC (%)*
1.25	0.03	0.10	29.32
2.5	0.11	0.26	50.04
5	0.72	1.33	83.03
10	2.43	3.76	97.38
25	3.27	5.07	99.16
50	3.27	5.09	99.19

A voltage threshold can be utilized to increase the accuracy of the proposed method. By setting the hyperparameters of the proposed method, there is a trade-off between accuracy and RoC. Table 4.15 gives the accuracy and the achieved RoC at different thresholds for a set RoC of 99.2 % of US06 at 25 °C ambient temperature. The results demonstrate a substantial influence on accuracy across various threshold values. Between 5 mV and 10 mV threshold, a significant increase in error can be seen. Depending on the use case, the achieved RoC and accuracy are highly dependent on the selected threshold. The threshold setting depends on the use-case, but based on the shown results, a threshold of 2.5 mV enable a achieved with 50.04 % RoC and an MAE of 0.11 mV a good tread-off between high accuracy and RoC for high accurate scenarios. If bandwith is the main limiting factor, 10 mV gives still good accuracies and a high RoC. With higher threshold, the accuracy drops noticeably with only a slight increase in the ROC.

4.4.3 Compression Runtime Evaluation

Computational cost is another crucial factor regarding real-world cloud-based BMS applications. For evaluation of the algorithm to compress voltage based on current is implemented at an STM32L432 microcontroller with an Arm Cortex-M4 core at 80 MHz and the computation time has been measured [29].

An input of 600 data points from the LA92 drive cycle at 25 °C is used as validation data. The experiment evaluates two aspects: the total execution time needed to compress 600 data points and the execution time necessary to compress each time window. The time window size is corresponded to each RoC. Decompression of the data is not included in the execution time evaluation. The results are illustrated in Fig. 4.31. Table 4.16 shows a mapping of window size to the resulting RoC. As previous experiments have shown, a higher compression rate leads to less accuracy. The results show that higher compression rates lead to shorter execution times. The microcontroller needs 54 ms to compress 600 data points at RoC 90 %. The higher compression rate of 99.2 % for 600 data points can be achieved in 35 ms. Herein a decrease of 35 % in the computation time compared to RoC 90 %.

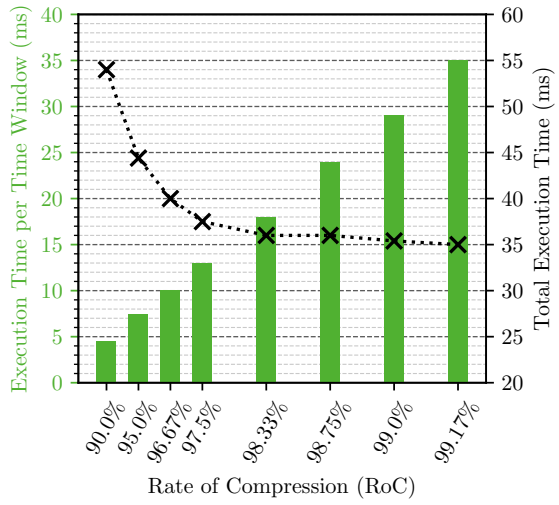


Fig. 4.31. Compression performance analysis on STM32L432 microcontroller to compress one-time window (green) and 600 data points (black); At higher RoC, decreasing total execution time to compress 600 data points, but higher execution time per time window

Rate of Compression (%)	*Window Size*
99.00	50
95.00	100
96.67	150
97.50	200
98.33	300
98.75	400
99.00	500
99.17	600

Table 4.16. Mapping of window size to RoC

5 Cloud-based State Estimation and Control of a Reconfigurable Battery

In this chapter, we address both research questions (RQ) introduced in Section 1.2 on page 2. We combine the machine-learning algorithm to control a reconfigurable battery with an cloud-based BMS approach. The setup of the proposed cloud-based BMS for reconfigurable batteries is illustrated. In Section 5.2 on the next page the methodology is described followed by the evaluation and discussion of the experimental results.

5.1 Cloud-based BMS and Digital Twin Setup

As machine learning control algorithm proposed the AQL algorithm in Section 3.3.2 on page 45. In addition, we proposed and discussed architectures and technologies to facilitate a cloud-based BMS and digital twins of battery energy storage systems in Chapter 4 on page 53. In this chapter, we are going to combine these two fields in employing the AQL control algorithm with state estimation in the cloud, employing the five-layer IoT framework, the universal data specification, and data compression and target RQ1 and RQ2 within a single setup. Fig. 5.1 illustrates the comprehensive process from measurement through configuration control employed within this experiments. The results of these experiments have been published in our study [22].

To evaluate the applicability of BMS in the cloud, we evaluate the total computational cost of state estimation and reconfigurable battery control. These experiments were conducted under two distributed compute architectures, hybrid and centralized, Fig. 4.11 on page 71. On the other and the experiments were performed under the distributed compute paradigms fog and cloud computing, as illustrated in Fig. 4.10b on page 70. Additionally, the necessary bandwidth is assessed using the data specifications outlined in Section 4.2 on page 59, utilizing the JSON data format along with the compression of voltage data as presented in Section 4.4 on page 88. Furthermore, we evaluate the required bandwidth by employing the data specification using the JSON data format, as proposed in Section 4.2 on page 59, and compression of voltage data, as proposed in Section 4.4 on page 88. The system's architecture implements the five-layer IoT framework, as proposed in Section 4.1.2 on page 56, whereas the third layer, the gateway layer, is not implemented. The device implements the functionality of the third and second layers. In order to reduce the overall

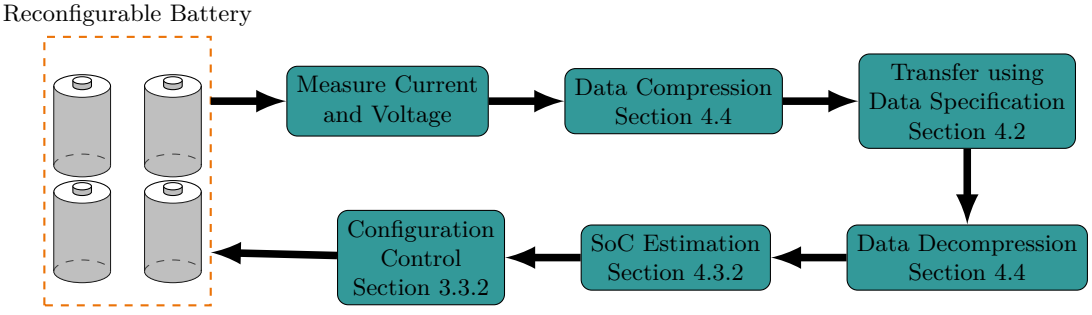
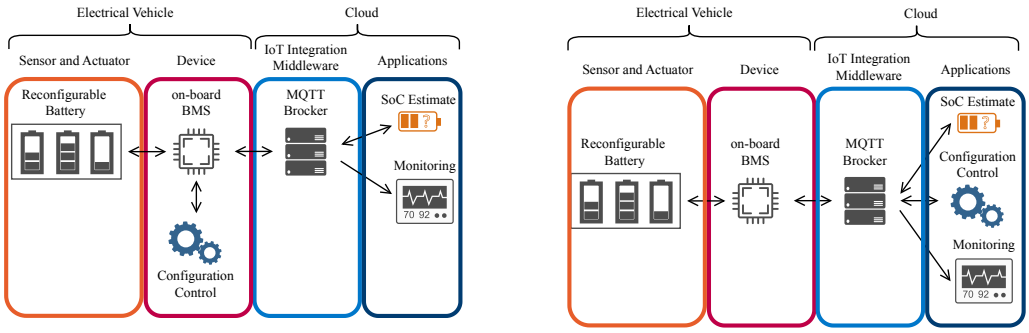


Fig. 5.1. Cloud-based BMS and digital twin data flow



(a) (b)

Fig. 5.2. Cloud-enhanced BMS architecture for SoC estimation and control of reconfigurable battery pack (a) hybrid architecture and (b) centralized architecture

complexity of the system, the five-layer IoT framework contains the possibility of a direct connection between the IoT Integration Middleware and the Device layer.

Fig. 5.2 illustrates the architecture of the two distributed compute architectures implemented: centralized and hybrid. The compute architectures described in Section 4.3.1 on page 69 can be encapsulated as subsequently:

Hybrid Computational tasks are executed on distributed compute nodes and on the vehicle. Fig. 5.2a shows the architecture of this approach. The application layer of the IoT framework contains the SOC estimation and monitoring. The onboard BMS executes the configuration control of the reconfigurable battery. Vehicle and distributed compute nodes execute computational tasks.

Centralized Computational tasks are executed only on the distributed compute nodes. The vehicle only measures and controls actuators. Fig. 5.2b shows this approach, where state estimation and controlling are executed entirely in the cloud. The reconfigurable battery executes the control commands set by the cloud-based control algorithm.

5.2 Cloud-based Digital Twin Methodology

The experimental setup includes hardware and software components – the architectures in Fig. 5.2 introduce the components and their interconnection. Cloud and fog deployment are differentiated as distributed computing units that deploy the layers of IoT integration middleware and application. The characteristics of the cloud and fog computing infrastructure are the following:

Fog Computing

Location: Datacenter container within the university campus

Operating System: Ubuntu 20.04 LTS

Server Configuration: Dell PowerEdge R750xa with Intel Xeon Gold 6338 CPU

Cloud Computing

Location: AWS Region Frankfurt, Germany; Availability Zone eu-central-1b

Operating System: Ubuntu 24.04 LTS[1]

Server Configuration: EC2 instance type t2.2xlarge

Fig. 5.3 illustrates the centralized cloud computing implementation. The applications, SOC estimation, configuration control, and monitoring, are deployed as Docker containers. The MQTT Broker, which implements the IoT Integration Middleware, is also deployed as a Docker container. As shown in the figure, the automated data flow between the DUT and the applications is an implementation of a digital twin of a reconfigurable battery as defined by Kritzinger et al. [155] and illustrated in Fig. 2.21 on page 31.

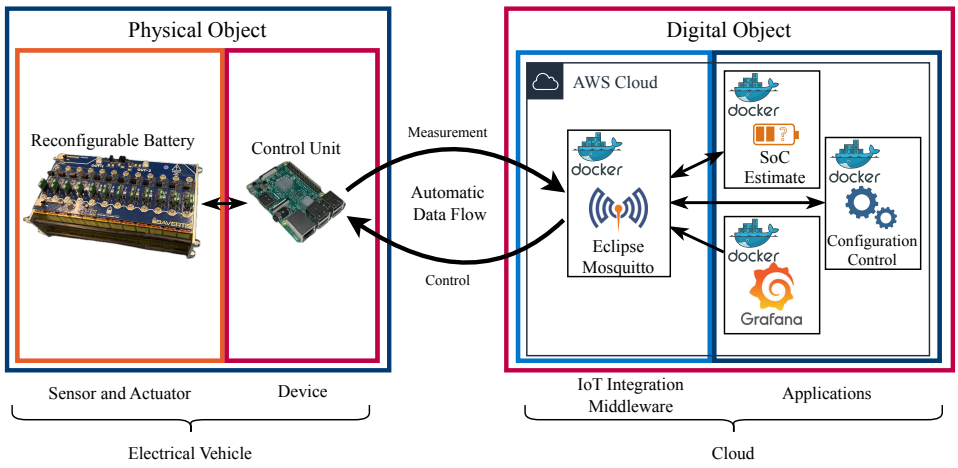


Fig. 5.3. Implementation of centralized cloud computing digital twin with reconfigurable battery and control unit as physical object, and the cloud computing environment as digital representation

Due to the computational demands of the AQL control algorithm, an additional computing unit is utilized to run the configuration control application in hybrid control. A Lenovo ThinkPad P15 gen 1 with an Intel Core i9-10885H CPU is used as a computing unit.

Battery cell simulators are employed instead of battery cells for reproducibility. This experiment does not evaluate the accuracy of the employed SOC estimation method and control algorithm. In Section 4.3.2 on page 71 and Section 3.3.2 on page 45, the SOC estimation method and the control algorithm have been evaluated. Fig. 5.4 shows the physical setup and its components as follows:

- DUT: 12-cell hybrid cascaded multilevel converter (orange) [193], a topology of interconnected half-bridge modules and an h-bridge converter, as a reconfigurable battery module
- Raspberry Pi 4 (red) as Control Unit
- Lenovo ThinkPad-P15-Gen-1 as Computing Unit in case of the hybrid approach
- 3× Load Resistor: MAL-200 MEG 10 Ω (blue) in parallel
- 6× Battery Cell Simulator: Rohde & Schwarz NGM202 Power Supply (orange)

[1] AMI: ami-0084a47cc718c111a

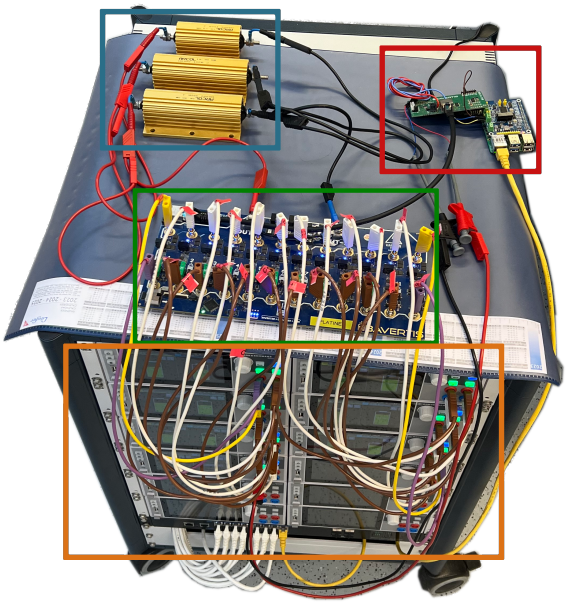


Fig. 5.4. Experimental setup with a 12-cell hybrid cascaded multilevel converter as DUT (blue), power supply as battery cell simulator (orange), Raspberry Pi 4 as control unit (red), and ohmic resistance as load (blue).

5.3 Evaluation and Discussion

The evaluation of the experimental outcomes comprises three independent parts: illustration of monitoring, request time, and bandwidth measurements.

5.3.1 Interactive Visualization for Monitoring

A web application is employed and hosted in the cloud environment to enable real-time online monitoring. Fig. 5.5 shows a screenshot of an Grafana instance deployed in AWS. The figure represents the system's real-time monitoring, employing five diagrams to show the battery status. The heatmap on the top left illustrates the cell configurations over time. The x-axis shows the time, and the y-axis shows the cells. The colours in the heatmap show the configuration of each cell at each time step, in blue bypass and green serial configuration. The second and third diagrams in the first row illustrate the SOC of the cells $c1 - c12$. The bar chart shows the current SOC. Because of the focus of this experiment, we decided to display the SOC of the simulator and not the estimated state. The accuracy of the estimated SOC values deviates significantly from the values of the simulator. The line plot, the third diagram in the first row, shows the decreasing SOC over time. The second row contains the current and voltage diagrams per cell. Implementing an interactive web application enables the researcher to conduct a live analysis during the experiment. It enables instant interaction with real-time measured experimental data. Error analysis and observation of functionalities, such as state estimation methods and control algorithms implemented in this experiment, can be observed immediately. This observation is independent of the location of the experiment and the person analyzing it. Also, this implementation provides the possibility of analyzing and comparing historical data.

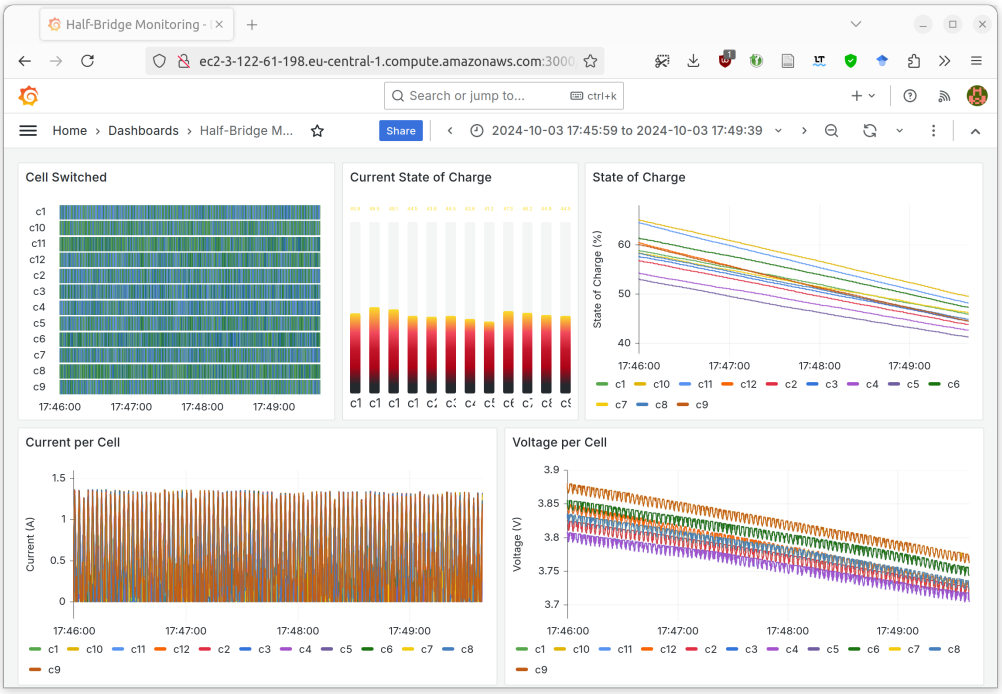


Fig. 5.5. Interactive visualization of real-time measurements, state estimation, and control commands during the experiments in Grafana hosted at AWS infrastructure.

5.3.2 Request Time Evaluation

The literature mentions a typical sampling frequency of SOC with 1 - 10 Hz [238], [239]. Within this experiment, we define 10 Hz as the baseline for estimation and control. The evaluation contains eight different system configurations, the combinations of the following settings: compute paradigms, cloud- and fog computing; compute architecture, hybrid and centralized approach; and control algorithms, switching-max and AQL.

Fig. 5.6 illustrates the concluded time measurements of these eight experiments. The request time consists of the time between the battery measurements taken and the configuration executed by the control unit. This time includes the time needed to transfer the data from DUT to the compute unit, perform SOC estimation, compute configuration, and transfer data to the control unit. A boxplot has been chosen, representing the median, shown as a black horizontal line, and the 25 % quartiles. A green-black triangle represents the mean request time of each configuration. The graph does not include outliers. There is an apparent deviation between the mean and median, except for the Cloud Centralized and Cloud Hybrid results with the switching-max control approach. The cloud hybrid switching-max control approach archived the shortest mean request time with 67.7 ms and median request time of 71.3 ms. Cloud Centralized switching-max control approach archived with mean request time of 70.2 ms and median request time of 69.1 ms the shortest mean request time. The fog centralized switching-max approach demonstrated the highest request time, with a mean of 94.3 ms and a median of 73.3 ms. The results demonstrate that all configurations exhibit average and median values below the defined baseline. However, cloud computing deployments indicate better request times than their fog computing variants, excluding the hybrid AQL control configurations.

For a more detailed explanation of these results, Figs. 5.7a and 5.7b show the request time in seconds [s] for the AQL and switching-max control, respectively. The representation

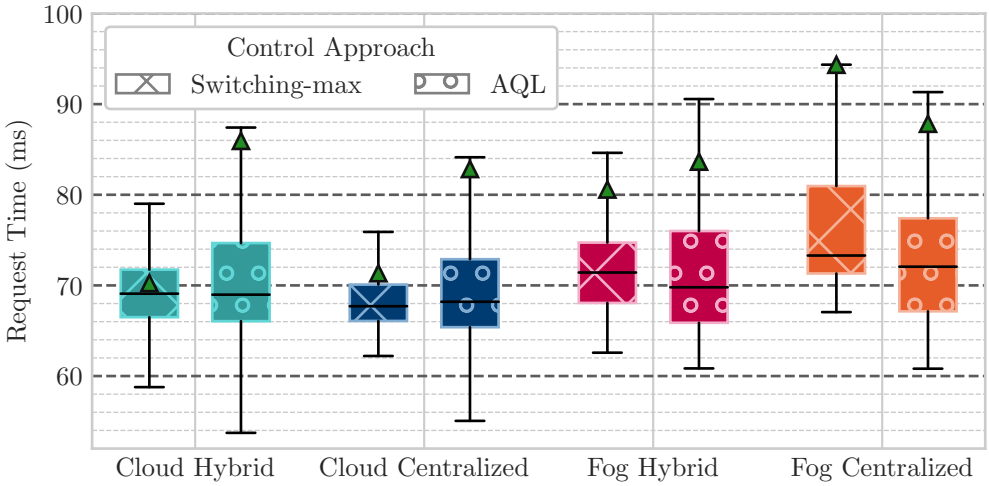


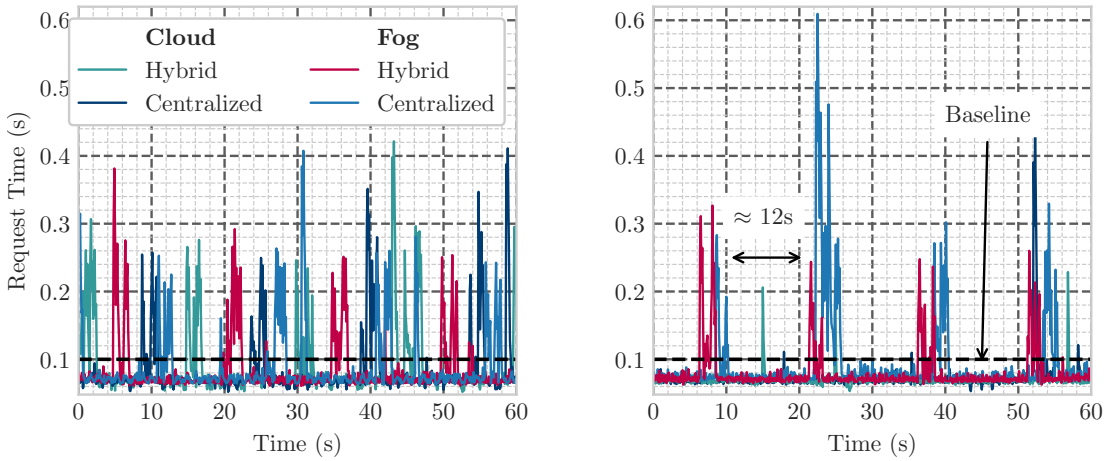
Fig. 5.6. Request time over experiment time of one minute dependent on distributed computing environment (cloud/fog), computing architecture (hybrid/centralized), and control approach (switching-max/AQL); summarizing of time measurements; green triangle mean, black line median, control approach by texture

of the required request time of each request enables the evaluation of archiving the defined baseline of 10 Hz per request. The results show that the AQL control algorithm requires significantly extended durations compared to the switching-max control algorithm. For fog centralized AQL control, a total of 14.0 % of all requests exceeded the defined baseline of 0.1 s per request. Even in the cloud centralized AQL control configuration, 11.8 % of requests were above the defined baseline. This configuration exhibited the fewest requests above the baseline within AQL control. This result highlights the high computing requirements for the AQL control algorithm.

As illustrated in Fig. 5.7b, the percentage of outlaid requests above the baseline is significantly less in the case of switching-max control compared to AQL control. With 1.1 % and 1.5 % respectively, the cloud hybrid and cloud centralized configuration outperforms the fog configurations with 13.8 % (hybrid) and 8.7 % (centralized). The exceeding of the baseline follows a recurring pattern. For the Fog configurations, as shown in Fig. 5.7b, a repeating pattern of approximately 2 s – 4 s of transmission shows significant delays, which then show only transmissions within the set limits for 12 s. As the experiments with cloud computing infrastructure do not show this pattern, it can be assumed that this can be explained by network configurations within the university network. However, this does not occur for connections outside the university campus.

5.3.3 Bandwidth Evaluation

The required bandwidth depends on multiple settings: measurement frequency, experimental setup, measured parameters, employed data format, and if data compression is in use, the method and the rate of compression. To compare the required bandwidth of different battery systems, we compare the bandwidth requirements for the DUT, a reconfigurable battery with a hybrid cascaded multilevel converter as topology, with the bandwidth of a battery pack. Therefore, the reconfigurable battery is employed as a serial-connected battery pack without dynamic reconfiguration. Both systems contain 12 battery cells. The battery pack setup requires transferring voltage measurements from 12 cells and one battery pack current measurement. On the other hand, the reconfigurable battery configuration requires



(a) (b)

Fig. 5.7. Request time over time with switching-max control approach (a) and AQL approach (b)

the transfer of voltage and current of 12 cells. As data format, the JSON implementation of the proposed universal data specification is used, as proposed in Section 4.2 on page 59. The measurement of one data point for both setups, battery pack and reconfigurable battery, are shown in Listing 5.1 and 5.2, respectively.

```
{"timestamp": 1727886569.2028234,
 "vol_m1_c1": 3.902268, "vol_m1_c2": 3.902268,
 "vol_m1_c3": 3.918042, "vol_m1_c4": 3.918042,
 "vol_m1_c5": 3.962061, "vol_m1_c6": 3.962061,
 "vol_m1_c7": 3.972382, "vol_m1_c8": 3.972382,
 "vol_m1_c9": 4.01781, "vol_m1_c10": 4.01781,
 "vol_m1_c11": 4.041907, "vol_m1_c12": 4.041907,
 "cur_m1": 0.7707577}
```

Listing 5.1: Transfered JSON data of one data point of a battery pack with 12 voltage and one current measurement

```
{"timestamp": 1727885074.3059,
"vol_m1_c1": 3.91324, "cur_m1_c1": 7.808e-05,
"vol_m1_c2": 3.91324, "cur_m1_c2": 0.00018624,
"vol_m1_c3": 3.929683, "cur_m1_c3": 0.00022464,
"vol_m1_c4": 3.929683, "cur_m1_c4": 0.0001728,
"vol_m1_c5": 3.966075, "cur_m1_c5": 0.5224461,
"vol_m1_c6": 3.966075, "cur_m1_c6": 0.0001664,
"vol_m1_c7": 3.98312, "cur_m1_c7": 0.2234317,
"vol_m1_c8": 3.98312, "cur_m1_c8": 0.1706842,
"vol_m1_c9": 4.029879, "cur_m1_c9": 0.2205907,
"vol_m1_c10": 4.029879, "cur_m1_c10": 0.00012928,
"vol_m1_c11": 4.049043, "cur_m1_c11": 0.5383738,
"vol_m1_c12": 4.049043, "cur_m1_c12": 0.44576}
```

Listing 5.2: Transfered JSON data of one data point of reconfigurable battery with 12 voltage and current measurements

Fig. 5.8 illustrates the required bandwidth in kbit/s over 1 min. The bandwidth includes the transferred data and the protocol overhead caused by the used protocols Message Queuing Telemetry Transport (MQTT) and Transmission Control Protocol (TCP). MQTT is a lightweight messaging protocol with a small code footprint for networks and devices with limited bandwidth. MQTT uses the TCP protocol for transmitting data. TCP is a communication standard for delivering data and messages in order and lossless bidirectional through networks. The data is measured and transferred during the experiment with a frequency of 10 Hz. Red and turquoise illustrate the required bandwidth for the battery pack and reconfigurable battery without compression. The blue (reconfigurable battery) and orange (battery pack) lines illustrate the required bandwidth for compressed data. To compress the data, the proposed voltage compression data algorithm, as introduced in Section 4.4 on page 88, has been implemented with a 6 s time window for compression. In the case of a reconfigurable battery, the battery voltage data is compressed based on the battery current due to the individual switching of each battery. The battery pack setup archives higher compression; the battery voltage is compressed based on the battery pack current, whereby fewer parameters are transmitted. As shown in the figure, compressed data causes peaks in required bandwidth of up to 125 kbit/s (reconfigurable battery) and 55 kbit/s (battery pack) and uncompressed causes permanent data transmission of 60 kbit/s and 40 kbit/s, respectively.

Table 5.1 shows the transferred data over the measurement time of 1 min of all four variations. Also, it includes the RoC over the whole amount of transferred data. With 64.7 % and 77.1 %, significant compressions could be archived for reconfigurable battery and battery pack measurements. Noc compressed are timestamps and current measurements.

Table 5.1: Transfered data in kByte over one minute and Rate of Compression

	Reconfigurable Battery	Battery Pack
Uncompressed (kB)	438.0	290.4
Compressed (kB)	154.5	66.6
Rate of Compression (%)	64.7	77.1

It is necessary to compare the required bandwidth with the usable bandwidth to evaluate the usability of BMS in the cloud. Therefore, it is necessary to review bandwidths for the mobile networks used: LTE and 5G. In 3rd Generation Partnership Project (3GPP) Release 8 LTE is specified, which enables a peak upload of 75 Mbit/s [296]. 3GPP is a

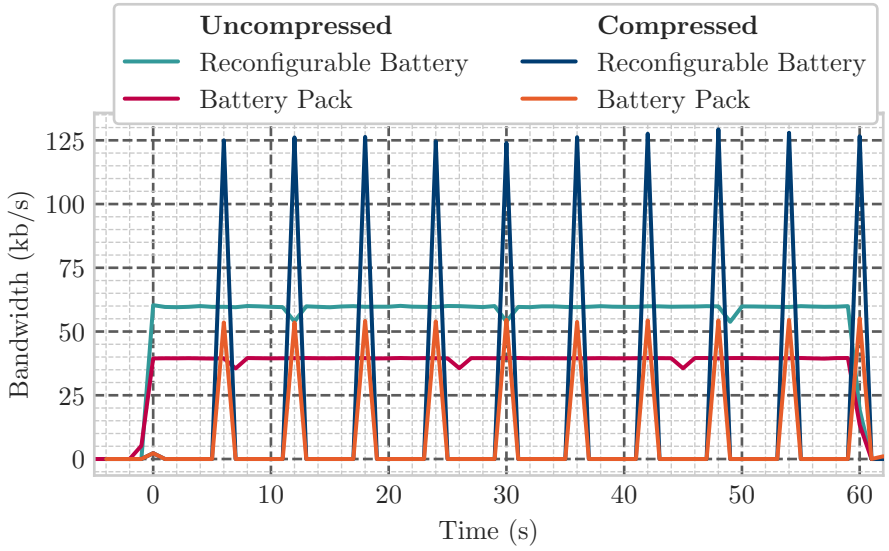


Fig. 5.8. Bandwidth required over time in kb/s of compressed and uncompressed data transfer for reconfigurable battery and conventional battery pack

group of several standards organizations that develop mobile telecommunications protocols. LTE-Advanced specified in 3GPP Release 10 enables upload peaks of 500 Mbit/s by specification. With International Mobile Telecommunications (IMT) Version for 2020 and future technology trends for the land-based IMT systems, known as IMT-2020, the International Telecommunication Union (ITU) defined the requirements for the fifth generation of mobile communications, also known as 5G. IMT-2020 defines 10 Gbit/s peak upload and 50 Mbit/s as user-experienced upload rate. Ghoshal et al. [297] evaluated the uplink of LTE and 5G under driving conditions. The mentioned driving speed was 32 km/h. The authors could measure a upload rate of 33.1 Mbit/s ± 6.2 Mbit/s for LTE and at 5G 39.7 Mbit/s with a standard deviation of 21.2 Mbit/s. Other studies in the USA and several European countries obtained similar results [298], [299]. At stationary conditions for the most 5G operators evaluated an average upload between 20 Mbit/s to 80 Mbit/s and under less channel conditions, the average throughput of nearly all operators falls to around 20 Mbit/s [298]. In conclusion, while the theoretical peak upload speeds of LTE and 5G networks are sustainable, real-world performance, particularly under moving conditions, is significantly lower. However, the achieved average upload speeds shown in the literature of around 30 – 40 Mbit/s are several times higher than the required bandwidth of 0.06 Mbit/s or short-term bandwidth of 0.125 Mbit/s when applying data compression, measured in the experiment in our work. This indicates that a 12-cell module uses a mere fraction of less than 1 % of the bandwidth available with LTE and 5G.

5.3.4 Summary

The evaluation reveals that centralized and hybrid architectures achieve the target estimation and control frequency of 10 Hz in the mean. However, the centralized approach, primarily when implemented in the cloud, demonstrates superior performance in terms of request time. However, fallbacks must be considered in the event of a loss of connection for centralized implementations. Cloud computing consistently outperforms fog computing, particularly in the centralized architecture.

The work also highlights the significant computational demands of AI-based control.

Using the AQL algorithm proposed in this work leads to increased request times. In contrast, the switching-max control approach exhibits lower computational requirements and shorter response times.

Data compression is crucial for efficient data transmission, especially in bandwidth-constrained scenarios. The proposed compression technique effectively reduces bandwidth requirements for reconfigurable and conventional battery systems. A buffered data transfer of compressed data can significantly reduce the required bandwidth but increase the time between measurement and data transmission.

Even without compression, the required bandwidth for the reconfigurable battery system is well within the capabilities of current mobile networks like LTE and 5G. The in literature achieved average upload speeds of 30 – 40 Mbit/s in real-world scenarios are several times higher than the required bandwidth of 0.06 Mbit/s without compression or short-term bandwidth of 0.125 Mbit/s when applying data compression. This results indicate that a 12-cell module employs a mere fraction of less than 1 percent of the available bandwidth in LTE and 5G networks.

The cloud-based approach demonstrates the feasibility of remote monitoring, state estimation, and control of reconfigurable batteries. While AI-based control offers potential advantages, its computational demands must be carefully considered. Data compression is crucial for efficient data transmission, especially in bandwidth-constrained scenarios.

6 Conclusion and Outlook

This chapter summarizes the results of this work and evaluates the results concerning the research questions defined in Section 1.3 on page 2. We discuss the results of the chapters 3, 4, and 5. Ultimately, the future work section outlines potential research objectives for further exploration.

6.1 Results and Consolidation of Contribution

The conclusion of this work will be marked by a discussion of the research questions(RQs) defined in Section 1.3 on page 2. As a recap:

RQ 1 What type of machine-learning algorithm can be applied to enable a self-learning algorithm to implement a control algorithm of a reconfigurable battery and include the possibility of additional classic control methods in combination?

RQ 2 What cloud-based BMS architecture and methodology can be used to efficiently and effectively transfer data between the vehicle and distributed computing units?

With the first research question, we wanted to propose and evaluate machine learning algorithms that can be applied to enable a self-learning algorithm to implement a control algorithm of a reconfigurable battery. With the AQL algorithm, we propose, implement and evaluate in Section 3.3.2 on page 45 a promising method. Compared to the classic approach, the algorithm has shown its potential in simulated and hardware setups, especially in balancing the SOC within the BM3 approach. The full-bridge topology has shown promising results, and future investigations in multi-parameter optimizations have to show the full potential of the algorithm with this topology. Furthermore, the algorithm enables the combination of machine learning algorithms with the possibility of the combination with classic control algorithms to limit the set of possible configurations selected by the machine learning algorithm. This functionality enables the benefits of both sides: the self-learning of machine learning and the implemented expert knowledge of classic control approaches.

The second research question was investigated by introducing the universal data specification, enabling the transfer and storage of data from conventional battery packs and reconfigurable batteries. We proposed a compression method for lossy compression of voltage measurements to efficiently use bandwidth during transfer and storage for archiving. With the five-layer IoT framework, we introduced a framework for bidirectional data exchange with modular components adapting EV components and cloud computing standard architectures to enable cloud-based BMS and digital twinning for battery storage systems. Thus, evaluating cloud-based state estimation, SOC, and the proposed core and surface temperature estimation methodology shows promising results for implementing cloud-based BMS systems.

Finally, in Chapter 5 on page 101, we have evaluated implementing a hybrid and a centralized cloud-based BMS employing cloud-based state estimation and the AQL algorithm to control the reconfigurable battery. We have closed the automated data loop, implemented the cloud-based state estimation with the control algorithm, and implemented a cloud-based digital twin for our reconfigurable battery. We have shown the applicability of

hybrid cloud-based BMS systems and could observe latencies below 100 ms for the cloud and fog computing setup.

A summary of all the research topics covered in this thesis is described below:

Highly Performant Simulation A flexible framework enabling single-phase reconfigurable battery system simulations with a variable count of BM3 modules was introduced in Section 3.2 on page 34. This framework's Python implementation, utilizing the just-in-time compiler Numba, achieved a simulation duration of 0.33 s ± 0.7 ms to simulate 1 s of operational time for a 10-module BM3 system, offering a performance improvement of over 130 times in comparison to the Matlab/SIMULINK comparison implementation introduced in [169].

Control using Reinforcement Learning With the introduction of Q-learining and AQL algorithms, proposed in Section 3.3 on page 39, two reinforcement learning algorithms to balance the SOC of modules within a reconfigurable battery have been proposed. The AQL algorithm can be used to combine a classic approach and machine learning control of a reconfigurable battery. The algorithm has been evaluated through experimental setups involving a hybrid cascaded multilevel topology and the simulation of a BM3 converter. Using the hybrid cascaded multilevel topology a comparison with the algorithm proposed by Zheng [193] shows a 20.8 % slower SOC balancing of the proposed AQL algorithm. However, for the BM3 topology the proposed algorithm achieved a 4.5 times faster balancing than the proactive balancing algorithm proposed by Mashayekh et al. [169]. Despite this, the in this thesis proposed AQL algorithm requires higher complexity and computational cost during execution than classic control algorithms such as the conventional approaches [169], [193]. Due to the lower switching time requirements, Type 1 reconfigurable batteries, variable DC sources, can be a field of application of the proposed algorithm.

Universal Data Specification A novel data specification in the domain of batteries, reconfigurable batteries, and cloud-based BMS has been proposed in Section 4.2 on page 59. The proposed data specification implements interoperability by being open, free, and universally implementable. It is a standardized data specification that is reusable and richly described with a plurality of accurate and relevant attributes, as two of the FAIR [300] principles: Findable, Accessible, Interoperable, and Reusable. The other two principles, Findable and Accessible, are dependent on globally unique and persistent identifiers and accessible and shared, respectively. The data specification suits long-term data storage and real-time applications, such as cloud-based BMS or real-time data visualization. The implementation of the data specification is intended to measure data of battery packs and reconfigurable batteries with time-dependent switching configurations and measurements. The data specification integrates additional information in separated metadata with information about the installed components, such as sensor measurement errors. This information allows a better interpretation of the measured data and pre-processing for developing data-driven estimation algorithms. Furthermore, the proposed data specification also enables the in-data documentation of experiments and experimental setups.

Five-Layer IoT Framework In Section 4.1.2 on page 56, we proposed a five-layer IoT framework applied for cloud-based BMS. It contains a sensor and actuator, a device, a gateway, an IoT integration middleware, and an application layer. With a non-functional quality characteristics analysis, we identified the framework as the preferred option for the digital twinning of BMS in e-mobility applications. The analysis demonstrates the benefits of the five-layer IoT framework compared to the three-layer IoT framework

[198]. In particular, due to the flexibility and functional suitability of the research objective, the framework also represents a distinct advantage over the AWS IoT Fleetwise [200] solution.

Data Compression To compress battery voltage measurements, we proposed a polynomial coding lossy data compression algorithm in Section 4.4 on page 88. The algorithm performs effectively under pulsed conditions and low temperatures across various drive cycles. Thus, a MAE of 3.27 mV and RMSE of 5.09 mV as reconstruction error for the US06 drive cycle at 25 °C could be achieved. Furthermore, the applicability to real BMS systems could be demonstrated with a computing time of 35 ms required to compress 600 data points with a RoC of 99.2 % on an STM32L432 microcontroller. In Chapter 5 on page 101, we have shown the use of the compression algorithm in a cloud-based BMS setting for reconfigurable batteries and battery packs. The application of the compression algorithm has shown the potential to reduce the quantity of total data to be transmitted by a factor of 64.7 % and 77.1 % for reconfigurable batteries and battery packs, respectively.

Cloud-Based BMS Section 4.3.2 on page 71 presents a comparative analysis among the classical approach EKF and neural networks. Furthermore, the evaluation of several cloud-based BMS architectures in Chapter 5 on page 101 for researching the possibility of implementing a cloud-based BMS has been outlined. These methods have been implemented and deployed on a cloud-based platform towards the practical application for cloud- and digital twin-based BMS for e-mobility applications. For this purpose, an architecture for SOC estimation has been deployed using AWS. A minimal average computing time of 54.8 ms to estimate SOC in the cloud is noticed in FNN with LTE mobile network connection. The main impact on computing time is the distance between the DUT in Oshawa, Canada, and the cloud infrastructure in Montreal, Canada, and Frankfurt, Germany. Due to a typical sampling frequency for state estimation of 1 Hz – 10 Hz, the computational time of 66.1 ms ± 29.4 ms for EKF and for FNN 54.8 ms ± 10.3 ms under LTE as network connection is considered as reasonably brief. After evaluating cloud-based state estimation, we evaluated cloud-based state estimation and control in Chapter 5 on page 101. With 70.3 ms ± 12.0 ms, the best results could be achieved with the cloud hybrid approach implementing the classic control approach deployed at AWS located in Frankfurt, Germany and the DUT located in Neubiberg, Germany.

Core Temperature Estimation In Section 4.3.3 on page 76 we proposed with KAN and LSTM two neural networks to estimate the surface and core temperature at once using voltage, current, and ambient temperature measurements as input. The experimental validation with Molicel P42A as DUT with inserted core temperature sensors have demonstrated with KAN as algorithm an MAE of 0.579 °C and 0.353 °C and RMSE of 0.814 °C and 0.519 °C for core and surface temperature, respectively. Furthermore, at dynamic drive cycle conditions KAN has shown an MAE of 0.598 °C and 0.481 °C for core and surface temperature, respectively, through the UDDS drive cycle at 25 °C ambient temperature. Within the training of this model, the UDDS drive cycle has been excluded of the training data. The fact that both temperatures are estimated based only on current, voltage, and ambient temperature eliminates the need for battery surface temperatures in the battery system. Eliminating these sensors can significantly reduce the weight and cost of battery systems.

6.2 Future Work

The scope of this work opens new opportunities for exploring how cloud computing and machine learning in state estimation and controlling can further revolutionize battery management systems and reconfigurable batteries, particularly in real-time data analytics, digital twinning, and remote diagnostics.

The proposed AQL algorithm to balance SOC has shown the possibilities of machine learning to control reconfigurable batteries. Reducing switching losses, thermal management, or balancing the SOH are further target parameters for the algorithm and will be included in future work. Likewise, computing time is an aspect that must be considered for use in real-time environments.

The core temperature estimation method based on KAN presented promising results. Experiments with higher charge-discharge rates are necessary for further improvements and evaluations of this method. Also, physic-informed neural networks and neural operators can be evaluated to improve the proposed method's accuracy.

Lossy data compression has shown great potential and can be applied on the car to reduce bandwidth between sensors and the onboard BMS and to transfer data into the cloud and to be stored there. The present work has focused on compressing voltage measurements. This approach is to be combined with the compression of battery current measurements to reduce the total amount of data to be transmitted further.

Lastly, this work defined, evaluated and discussed architectures, frameworks, and technologies to enable cloud-based BMS, intelligent batteries, and digital twins of EVs battery electric storage systems. In future work, these results will help to implement generalized and personalized digital twins for EVs to improve battery efficiency, safety, and lifespan. The next step is implementing the five-layer IoT framework in a test vehicle in combination with the presented universal data specification, automated archiving of test data, and deploying real-time estimation methods in the cloud to implement cloud-based digital twins and BMS.

Work related publications

[18] D. Karnehm, N. Sorokina, S. Pohlmann, M. Ackermann, M. Kuder, and A. Gieraths (Neve), "High Performance Simulation Framework of Three-Phase Battery Modular Multilevel Management Converter System", in *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, 2022, pp. 1–5. DOI: 10.1109/IECON49645.2022.9968544.

[19] D. Karnehm, N. Sorokina, S. Pohlmann, A. Mashayekh, M. Kuder, and A. Gieraths (Neve), "A High Performance Simulation Framework for Battery Modular Multilevel Management Converter", in *2022 International Conference on Smart Energy Systems and Technologies (SEST)*, IEEE, 2022, pp. 1–6. DOI: 10.1109/SEST53650.2022.9898406.

[20] D. Karnehm, S. Pohlmann, and A. Neve, "State-of-Charge (SoC) Balancing of Battery Modular Multilevel Management (BM3) Converter using Q-Learning", in *2023 IEEE Green Technologies Conference (GreenTech)*, Denver, CO, USA: IEEE, 2023, pp. 107–111. DOI: 10.1109/GreenTech56823.2023.10173832.

[21] D. Karnehm, W. Bliemetsrieder, S. Pohlmann, and A. Neve, "Controlling Algorithm of Reconfigurable Battery for State of Charge Balancing Using Amortized Q-Learning", *Batteries*, vol. 10, no. 4, 2024. DOI: 10.3390/batteries10040131.

[22] D. Karnehm, A. Samanta, A. Neve, and S. Williamsion, "Pioneering Cloud-enhanced Real-time Control of Modular-Multilevel Reconfigurable Battery Packs for Automotive Applications", in *IEEE ECCE Asia 2025*, 2025.

[23] D. Karnehm, A. Samanta, L. Anekal, S. Pohlmann, A. Neve, and S. Williamson, "Five-layer IoT and Fog Computing Framework Towards Digital Twinning of Battery Management Systems for e-Transportation", in *4th International Conference on Smart Grid and Renewable Energy (SGRE24)*, IEEE, Jan. 2024.

[24] D. Karnehm, S. Pohlmann, A. Wiedenmann, M. Kuder, and A. Neve, "Introduction of a Cloud Computing Architecture for the Condition Monitoring of a Reconfigurable Battery System for Electric Vehicles", in *2023 6th Conference on Cloud and Internet of Things (CIoT)*, Lisbon, Portugal: IEEE, 2023, pp. 29–33. DOI: 10.1109/CIoT57267.2023.10084883.

[25] D. Karnehm and A. Neve, "Adaptive Compressing Electric Vehicle Battery Pack Measurements using Polynomial Coding", in *IEEE International Conference on Smart Mobility 2024*, 2024.

[26] D. Karnehm, A. Samanta, M. Hohenegger, *et al.*, "Universal Data Specification and Real-time Data Streaming Architecture as Enablers for Cloud-based Battery Management Systems", *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2024. DOI: 10.1109/JESTPE.2024.3413163.

[27] D. Karnehm, A. Samanta, L. Anekal, S. Pohlmann, A. Neve, and S. Williamson, "Comprehensive Comparative Analysis of Deep-Learning-Based State-of-Charge Estimation Algorithms for Cloud-Based Lithium-Ion Battery Management Systems", *IEEE Journal of Emerging and Selected Topics in Industrial Electronics*, vol. 5, no. 2, pp. 597–604, 2024. DOI: 10.1109/JESTIE.2024.3373267.

[28] D. Karnehm, L. Anekal, A. Samanta, A. Neve, and S. Williamson, "Towards Digital Twining of Lithium-ion Battery Management Systems: An Extended Kalman Filter for State-of-Charge Estimation in Cloud-platform", in *ECCE 2023 - IEEE Energy Conversion Congress and Expo*, Nov. 2023.

[29] D. Karnehm and A. Neve, "A resource-constrained polynomial regression approach for voltage measurement compression in electric vehicle battery packs", *Batteries*, vol. 10, no. 9, 2024, ISSN: 2313-0105. DOI: 10.3390/batteries10090305. [Online]. Available: https://www.mdpi.com/2313-0105/10/9/305.

[30] D. Karnehm, A. Samanta, C. Rosenmüller, A. Neve, and S. Williamson, "Internal Temperature Estimation of Lithium-Ion Batteries Using Long Short-Term Memory (LSTM) Network and Kolmogorov-Arnold Network (KAN)", 2024, In Review.

[31] S. Pohlmann, D. Karnehm, A. Mashayekh, M. Kuder, A. Gieraths (Neve), and T. Weyh, "State-of-Charge Estimation of Lithium-Ion Batteries Using Machine Learning Based on Augmented Data", in *2022 International Conference on Smart Energy Systems and Technologies (SEST)*, 2022, pp. 1–6. DOI: 10.1109/SEST53650.2022.9898474.

[32] F. Schwitzgebel, J. Buberger, M. Kuder, *et al.*, "Multilevel Inverter based Battery System Operation using a Decentralized Controller", in *2022 International Conference on Smart Energy Systems and Technologies (SEST)*, IEEE, 2022, pp. 1–6.

[33] S. Pohlmann, A. Mashayekh, D. Karnehm, M. Kuder, A. Gieraths (Neve), and T. Weyh, "State of Charge Prediction of Lithium-Ion Batteries Based on Artificial Neural Networks and Reduced Data", in *2022 24th European Conference on Power Electronics and Applications (EPE22 ECCE Europe)*, IEEE, 2022, pp. 1–10.

[34] S. Pohlmann, A. Mashayekh, F. Stroebl, *et al.*, "State-of-Health prediction of lithium-ion batteries based on a low dimensional Gaussian Process Regression", *Journal of Energy Storage*, vol. 88, p. 111 649, 2024. DOI: 10.1016/j.est.2024.111649.

[35] A. Samanta, D. Karnehm, A. Neve, and S. William, "Voltage Relaxation Pattern Recognition for Efficient Sorting of Healthy Cells for Second-life Applications of Retired Electric Vehicle Batteries", in *25th IEEE International Conference on Industrial Technology*, 2024.

[36] A. Samanta, D. Karnehm, A. Neve, and S. Williamson, "Data-informed Healthy Cell Clustering Technique for Second-life Applications of Retired Electric Vehicle Batteries", in *2024 IEEE Transportation Electrification Conference and Expo (ITEC)*, 2024. DOI: 10.1109/ITEC60657.2024.10598850.

[37] N. Sorokina, W. Bliemetsrieder, J. Estaller, D. Karnehm, T. Weyh, and M. Kuder, "Performance Analysis of Reconfigurable Inverter Battery Systems for Aircraft Applications", in *2024 24th International Scientific Conference on Electric Power Engineering (EPE)*, 2024. DOI: 10.1109/EPE61521.2024.10559556.

[38] S. Pohlmann, A. Mashayekh, D. Karnehm, M. Kuder, A. Neve, and T. Weyh, "Lithium Ion Batteries Impedance Approximation and Model Parameterization Using Artificial Neural Networks", in *2024 IEEE International Conference on Industrial Technology (ICIT)*, 2024. DOI: 10.1109/ICIT58233.2024.10540962.

[39] A. Samanta, C. Chetri, D. Karnehm, A. Neve, and S. Williamson, "Temporal sensitivity analysis of internal temperature informed charging algorithms and rapid thermal management system for e-mobility", in *2024 IEEE Energy Conversion Congress and Exposition (ECCE)*, 2024, pp. 2302–2304. DOI: 10.1109/ECCE55643.2024.10861466.

Bibliography

[1] "Henry Ford Explains Why He Gives Away $10,000,000", *New York Times*, Jan. 11, 1914. [Online]. Available: https://www.nytimes.com/1914/01/11/archives/henry-ford-explains-why-he-gives-away-10000000-declares-that-he-is.html.

[2] J. Wieler, *Electric cars in the test: this is how long the range really is*, Elektroautos im Test: So hoch ist die Reichweite wirklich, German, Accessed on August 9, 2024, 2024. [Online]. Available: https://www.adac.de/rund-ums-fahrzeug/elektromobilitaet/elektroauto/stromverbrauch-elektroautos-adac-test/.

[3] G. S. Fink, *Longest-Range Electric Cars We Ever Tested*, Accessed on August 9, 2024, 2024. [Online]. Available: https://www.caranddriver.com/features/g32634624/ev-longest-driving-range/.

[4] D. Group. "Creating a more sustainable fleet: Could this electric semi truck be a game-changer?" (2024), [Online]. Available: https://www.dhl.com/global-en/delivered/responsibility/dhl-tests-tesla-semi-electric-truck.html (visited on 10/15/2024).

[5] N. Tashakor, "Novel topologies and/or techniques for emerging applications", in *Novel Highly Flexible Modular Power Electronics for Energy Storage and Conversion Systems*. Cham: Springer Nature Switzerland, 2023, pp. 145–207, ISBN: 978-3-031-36843-1. DOI: 10.1007/978-3-031-36843-1_5.

[6] *Adoption of the paris agreement fccc/cp/2015/l.9/rev.1 (united nations framework convention on climate change, 2015).*

[7] IPCC, "Summary for Policymakers", in *Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, IPCC, Geneva, Switzerland, 2023. DOI: 10.59327/IPCC/AR6-9789291691647.001.

[8] H. Ritchie, P. Rosado, and M. Roser, "Breakdown of carbon dioxide, methane and nitrous oxide emissions by sector", *Our World in Data*, 2020. [Online]. Available: https://ourworldindata.org/emissions-by-sector.

[9] *Global GHG emissions shares by subsector*, en, https://www.statista.com/statistics/1167298/share-ghg-emissions-by-sub-sector-sector-globally, Accessed: 2024-10-14.

[10] *Your guide to climate action: Transport.* [Online]. Available: https://www.un.org/en/actnow/transport (visited on 10/14/2024).

[11] M. Schmauss, J. Bawidamann, M. Haberl, *et al.*, "Fluid transport for climate protection and air pollution control", Flüssiger Verkehr für Klimaschutz und Luftreinhaltung, German, 2023.

[12] J. Buberger, A. Kersten, M. Kuder, R. Eckerle, T. Weyh, and T. Thiringer, "Total CO2-equivalent life-cycle emissions from commercially available passenger cars", *Renewable and Sustainable Energy Reviews*, vol. 159, p. 112 158, May 1, 2022, ISSN: 1364-0321. DOI: 10.1016/j.rser.2022.112158. (visited on 12/20/2022).

[13] A. García, J. Monsalve-Serrano, S. Martinez-Boggio, and R. Soria Alcaide, "Carbon footprint of battery electric vehicles considering average and marginal electricity mix", *Energy*, vol. 268, p. 126 691, 2023, ISSN: 0360-5442. DOI: 10.1016/j.energy.2023.126691. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544223000853.

[14] X. Xia, P. Li, Z. Xia, R. Wu, and Y. Cheng, "Life cycle carbon footprint of electric vehicles in different countries: A review", *Separation and Purification Technology*, vol. 301, p. 122 063, 2022, ISSN: 1383-5866. DOI: 10.1016/j.seppur.2022.122063. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383586622016185.

[15] A. Albatayneh, M. N. Assaf, D. Alterman, and M. Jaradat, "Comparison of the overall energy efficiency for internal combustion engine vehicles and electric vehicles", *Rigas Tehniskas Universitates Zinatniskie Raksti*, vol. 24, no. 1, pp. 669–680, 2020.

[16] H. Ritchie and P. Rosado, "Electricity Mix", *Our World in Data*, 2020. [Online]. Available: https://ourworldindata.org/electricity-mix (visited on 10/14/2024).

[17] P. Greim, A. A. Solomon, and C. Breyer, "Assessment of lithium criticality in the global energy transition and addressing policy gaps in transportation", *Nature Communications*, vol. 11, no. 1, p. 4570, Sep. 2020.

[40] H. H. Heimes, A. Kampker, M. Kehrer, *et al.*, "Fahrzeugstruktur", in *Elektromobilität: Grundlagen einer Fortschrittstechnologie*, A. Kampker and H. H. Heimes, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2024, pp. 69–106, ISBN: 978-3-662-65812-3. DOI: 10.1007/978-3-662-65812-3_5.

[41] S. S. G. Acharige, M. E. Haque, M. T. Arif, N. Hosseinzadeh, K. N. Hasan, and A. M. T. Oo, "Review of Electric Vehicle Charging Technologies, Standards, Architectures, and Converter Configurations", *IEEE Access*, vol. 11, pp. 41 218–41 255, 2023. DOI: 10.1109/ACCESS.2023.3267164.

[42] M. K. Hasan, M. Mahmud, A. Ahasan Habib, S. Motakabber, and S. Islam, "Review of electric vehicle energy storage and management system: Standards, issues, and challenges", *Journal of Energy Storage*, vol. 41, p. 102 940, 2021, ISSN: 2352-152X. DOI: 10.1016/j.est.2021.102940. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X21006575.

[43] M. Zwicker, M. Moghadam, W. Zhang, and C. Nielsen, "Automotive battery pack manufacturing – a review of battery to tab joining", *Journal of Advanced Joining Processes*, vol. 1, p. 100 017, 2020, ISSN: 2666-3309. DOI: 10.1016/j.jajp.2020.100017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666330920300157.

[44] R. R. Kumar, C. Bharatiraja, K. Udhayakumar, S. Devakirubakaran, K. S. Sekar, and L. Mihet-Popa, "Advances in Batteries, Battery Modeling, Battery Management System, Battery Thermal Management, SOC, SOH, and Charge/Discharge Characteristics in EV Applications", *IEEE Access*, vol. 11, pp. 105 761–105 809, 2023. DOI: 10.1109/ACCESS.2023.3318121.

[45] R. Islam, S. M. S. H. Rafin, and O. A. Mohammed, "Comprehensive Review of Power Electronic Converters in Electric Vehicle Applications", *Forecasting*, vol. 5, no. 1, pp. 22–80, 2023, ISSN: 2571-9394. DOI: 10.3390/forecast5010002. [Online]. Available: https://www.mdpi.com/2571-9394/5/1/2.

[46] J. B. Goodenough, "How we made the Li-ion rechargeable battery", *Nature Electronics*, vol. 1, no. 3, pp. 204–204, 2018. DOI: 10.1038/s41928-018-0048-6.

[47] W. Liu, T. Placke, and K. Chau, "Overview of batteries and battery management for electric vehicles", *Energy Reports*, vol. 8, pp. 4058–4084, 2022, ISSN: 2352-4847. DOI: 10.1016/j.egyr.2022.03.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484722005716.

[48] D. Deng, "Li-ion batteries: basics, progress, and challenges", *Energy Science & Engineering*, vol. 3, no. 5, pp. 385–418, 2015. DOI: 10.1002/ese3.95.

[49] S. Baazouzi, N. Feistel, J. Wanner, I. Landwehr, A. Fill, and K. P. Birke, "Design, Properties, and Manufacturing of Cylindrical Li-Ion Battery Cells—A Generic Overview", *Batteries*, vol. 9, no. 6, 2023. DOI: 10.3390/batteries9060309. [Online]. Available: https://www.mdpi.com/2313-0105/9/6/309.

[50] M. Schmid, U. Vögele, and C. Endisch, "A novel matrix-vector-based framework for modeling and simulation of electric vehicle battery packs", en, *Journal of Energy Storage*, vol. 32, p. 101 736, Dec. 2020, ISSN: 2352152X. DOI: 10.1016/j.est.2020.101736. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352152X20315735 (visited on 03/16/2022).

[51] M. Adaikkappan and N. Sathiyamoorthy, "Modeling, state of charge estimation, and charging of lithium-ion battery in electric vehicle: A review", *International Journal of Energy Research*, vol. 46, no. 3, pp. 2141–2165, 2022. DOI: 10.1002/er.7339. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/er.7339. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/er.7339.

[52] Y. Wang, J. Tian, Z. Sun, *et al.*, "A comprehensive review of battery modeling and state estimation approaches for advanced battery management systems", *Renewable and Sustainable Energy Reviews*, vol. 131, p. 110 015, 2020, ISSN: 1364-0321. DOI: 10.1016/j.rser.2020.110015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032120303063.

[53] W. Zhou, Y. Zheng, Z. Pan, and Q. Lu, "Review on the battery model and SOC estimation method", *Processes*, vol. 9, no. 9, p. 1685, 2021.

[54] C. Zou, L. Zhang, X. Hu, Z. Wang, T. Wik, and M. Pecht, "A review of fractional-order techniques applied to lithium-ion batteries, lead-acid batteries, and supercapacitors", *Journal of Power Sources*, vol. 390, pp. 286–296, 2018.

[55] D. Liu, X. Wang, M. Zhang, and M. Gong, "SOC Estimation of Lithium Battery Based on N-2RC Model in Electric Vehicle", in *2019 Chinese Control And Decision Conference (CCDC)*, ISSN: 1948-9447, Jun. 2019, pp. 2916–2921. DOI: 10.1109/CCDC.2019.8833008.

[56] W.-Y. Chang, "The State of Charge Estimating Methods for Battery: A Review", en, *ISRN Applied Mathematics*, vol. 2013, pp. 1–7, Jul. 2013, ISSN: 2090-5572. DOI: 10.1155/2013/953792. [Online]. Available: https://www.hindawi.com/journals/isrn/2013/953792/ (visited on 04/25/2022).

[57] Y. Wang, Q. Gao, G. Wang, P. Lu, M. Zhao, and W. Bao, "A review on research status and key technologies of battery thermal management and its enhanced safety", *International Journal of Energy Research*, vol. 42, no. 13, pp. 4008–4033, 2018. DOI: 10.1002/er.4158.

[58] L. Chen, M. Hu, K. Cao, *et al.*, "Core temperature estimation based on electro-thermal model of lithium-ion batteries", *International Journal of energy research*, vol. 44, no. 7, pp. 5320–5333, 2020.

[59] K. Sayed, A. Almutairi, N. Albagami, O. Alrumayh, A. G. Abo-Khalil, and H. Saleeb, "A Review of DC-AC Converters for Electric Vehicle Applications", *Energies*, vol. 15, no. 3, 2022, ISSN: 1996-1073. DOI: 10.3390/en15031241. [Online]. Available: https://www.mdpi.com/1996-1073/15/3/1241.

[60] J. Loncarski, M. Leijon, M. Srndovic, C. Rossi, and G. Grandi, "Comparison of Output Current Ripple in Single and Dual Three-Phase Inverters for Electric Vehicle Motor Drives", *Energies*, vol. 8, no. 5, pp. 3832–3848, 2015, ISSN: 1996-1073. DOI: 10.3390/en8053832. [Online]. Available: https://www.mdpi.com/1996-1073/8/5/3832.

[61] M. Karamuk, "Review of Electric Vehicle Powertrain Technologies with OEM Perspective", in *2019 International Aegean Conference on Electrical Machines and Power Electronics (ACEMP) & 2019 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, 2019, pp. 18–28. DOI: 10.1109/ACEMP-OPTIM44294.2019.9007175.

[62] K. Aditya, Y. Suresh, B. S. Naik, and B. N. Rao, "A Fault Tolerant Nine-level Inverter Topology With Full DC Utilization for Electric Vehicle Application", in *2022 IEEE International Conference on Power Electronics, Smart Grid, and Renewable Energy (PESGRE)*, 2022, pp. 1–6. DOI: 10.1109/PESGRE52268.2022.9715895.

[63] S. Satpathy, S. Bhattacharya, and V. Veliadis, "Comprehensive Loss Analysis of Two-level and Three-Level Inverter for Electric Vehicle Using Drive Cycle Models", in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, 2020, pp. 2017–2024. DOI: 10.1109/IECON43393.2020.9254520.

[64] S. Muhammad, M. U. Rafique, S. Li, Z. Shao, Q. Wang, and X. Liu, "Reconfigurable Battery Systems: A Survey on Hardware Architecture and Research Challenges", *ACM Trans. Des. Autom. Electron. Syst.*, vol. 24, no. 2, Mar. 2019, ISSN: 1084-4309. DOI: 10.1145/3301301.

[65] S. Ci, N. Lin, and D. Wu, "Reconfigurable Battery Techniques and Systems: A Survey", *IEEE Access*, vol. 4, pp. 1175–1189, 2016. DOI: 10.1109/ACCESS.2016.2545338.

[66] L. Tolbert, F. Peng, and T. Habetler, "Multilevel inverters for electric vehicle applications", in *Power Electronics in Transportation (Cat. No.98TH8349)*, 1998, pp. 79–84. DOI: 10.1109/PET.1998.731062.

[67] A. Poorfakhraei, M. Narimani, and A. Emadi, "A Review of Multilevel Inverter Topologies in Electric Vehicles: Current Status and Future Trends", *IEEE Open Journal of Power Electronics*, vol. 2, pp. 155–170, 2021. DOI: 10.1109/OJPEL.2021.3063550.

[68] V. Kumar, P. Kumari, and N. Kumar, "Comparison of different levels of cascaded H bridge multilevel inverter using PSPWM technique for EV applications", in *2023 IEEE International Students Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2023, pp. 1–8. DOI: 10.1109/SCEECS57921.2023.10062982.

[69] P. K. Yadav, H. P. Vemuganti, and M. Biswal, "A novel asymmetrical multiple-source multilevel inverter for electric vehicles", in *2023 IEEE 3rd International Conference on Smart Technologies for Power, Energy and Control (STPEC)*, 2023, pp. 1–5. DOI: 10.1109/STPEC59253.2023.10430492.

[70] L. Komsiyska, T. Buchberger, S. Diehl, *et al.*, "Critical review of intelligent battery systems: Challenges, implementation, and potential for electric vehicles", *Energies*, vol. 14, no. 18, 2021. DOI: 10.3390/en14185989.

[71] N. Tashakor, B. Arabsalmanabadi, L. O. Cervera, E. Hosseini, K. Al-Haddad, and S. Goetz, "A Simplified Analysis of Equivalent Resistance in Modular Multilevel Converters with Parallel Functionality", in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, 2020, pp. 4158–4163. DOI: 10.1109/IECON43393.2020.9255157.

[72] N. Tashakor, Z. Li, and S. M. Goetz, "A generic scheduling algorithm for low-frequency switching in modular multilevel converters with parallel functionality", *IEEE Transactions on Power Electronics*, vol. 36, no. 3, pp. 2852–2863, 2020. DOI: 10.1109/TPEL.2020.3018168.

[73] L. Baruschka and A. Mertens, "Comparison of cascaded H-bridge and modular multilevel converters for BESS application", in *2011 IEEE Energy Conversion Congress and Exposition*, IEEE, 2011, pp. 909–916. DOI: 10.1109/ECCE.2011.6063868.

[74] M. Kuder, J. Schneider, A. Kersten, T. Thiringer, R. Eckerle, and T. Weyh, "Battery modular multilevel management (bm3) converter applied at battery cell level for electric vehicles and energy storages", in *PCIM Europe digital days 2020; International Exhibition and Conference for Power Electronics, Intelligent Motion, Renewable Energy and Energy Management*, VDE, 2020, pp. 1–8.

[75] J. I. Rodriguez and S. B. Leeb, "A Multilevel Inverter Topology for Inductively Coupled Power Transfer", *IEEE Transactions on Power Electronics*, vol. 21, no. 6, pp. 1607–1617, 2006. DOI: 10.1109/TPEL.2006.882965.

[76] K. Ilves, L. Bessegato, L. Harnefors, S. Norrga, and H.-P. Nee, "Semi-full-bridge submodule for modular multilevel converters", in *2015 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia)*, IEEE, 2015, pp. 1067–1074. DOI: 10.1109/ICPE.2015.7167913.

[77] L. Komsiyska, T. Buchberger, S. Diehl, *et al.*, "Critical review of intelligent battery systems: Challenges, implementation, and potential for electric vehicles", *Energies*, vol. 14, no. 18, 2021, ISSN: 1996-1073. DOI: 10.3390/en14185989. [Online]. Available: https://www.mdpi.com/1996-1073/14/18/5989.

[78] J. Kacetl, J. Fang, T. Kacetl, N. Tashakor, and S. Goetz, "Design and Analysis of Modular Multilevel Reconfigurable Battery Converters for Variable Bus Voltage Powertrains", *IEEE Transactions on Power Electronics*, vol. 38, no. 1, pp. 130–142, 2023. DOI: 10.1109/TPEL.2022.3179285.

[79] F. Eroğlu, M. Kurtoğlu, and A. M. Vural, "Bidirectional DC–DC converter based multilevel battery storage systems for electric vehicle and large-scale grid applications: A critical review considering different topologies, state-of-charge balancing and future trends", en, *IET Renewable Power Generation*, vol. 15, no. 5, pp. 915–938, Apr. 2021, ISSN: 1752-1416, 1752-1424. DOI: 10.1049/rpg2.12042. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1049/rpg2.12042 (visited on 11/03/2022).

[80] J. Buberger, M. Hohenegger, J. Estaller, *et al.*, "Bidirectional charging for bevs with reconfigurable battery systems via a grid-parallel proportional-resonant controller", *Electricity*, vol. 4, no. 2, pp. 171–184, 2023, ISSN: 2673-4826. DOI: 10.3390/electricity4020011. [Online]. Available: https://www.mdpi.com/2673-4826/4/2/11.

[81] D. Vinnikov, T. Jalakas, and M. Egorov, "Feasibility study of half- and full-bridge isolated DC/DC converters in high-voltage high-power applications", in *2008 13th International Power Electronics and Motion Control Conference*, 2008, pp. 1257–1262. DOI: 10.1109/EPEPEMC.2008.4635441.

[82] N. Sorokina, J. Estaller, A. Kersten, *et al.*, "Inverter and Battery Drive Cycle Efficiency Comparisons of Multilevel and Two-Level Traction Inverters for Battery Electric Vehicles", in *2021 IEEE International Conference on Environment and Electrical Engineering and 2021 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, Sep. 2021. DOI: 10.1109/EEEIC/ICPSEurope51590.2021.9584705.

[83] M. Kesic, C. Chen, and A. C. Knoll, "Battery-Management-Systems", VDE Verband der Elektrotechnik Elektronik Informationstechnik, Tech. Rep., 2021.

[84] S. Yang, X. Liu, S. Li, and C. Zhang, "Key technologies of bms", in *Advanced Battery Management System for Electric Vehicles*. Singapore: Springer Nature Singapore, 2023, pp. 33–51, ISBN: 978-981-19-3490-2. DOI: 10.1007/978-981-19-3490-2_3.

[85] S. Yang, X. Liu, S. Li, and C. Zhang, "Cloud battery management system", in *Advanced Battery Management System for Electric Vehicles*. Singapore: Springer Nature Singapore, 2023, pp. 301–313, ISBN: 978-981-19-3490-2. DOI: 10.1007/978-981-19-3490-2_16.

[86] S. Yang, R. He, Z. Zhang, Y. Cao, X. Gao, and X. Liu, "CHAIN: Cyber Hierarchy and Interactional Network Enabling Digital Solution for Battery Full-Lifespan Management", en, *Matter*, vol. 3, no. 1, pp. 27–41, Jul. 2020, ISSN: 25902385. DOI: 10.1016/j.matt.2020.04.015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2590238520301867 (visited on 05/31/2023).

[87] S. Yang, Z. Zhang, R. Cao, *et al.*, "Implementation for a cloud battery management system based on the CHAIN framework", *Energy and AI*, vol. 5, Sep. 1, 2021, ISSN: 2666-5468. DOI: 10.1016/j.egyai.2021.100088. (visited on 05/31/2023).

[88] W. Li, M. Rentemeister, J. Badeda, D. Jöst, D. Schulte, and D. U. Sauer, "Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation", *Journal of energy storage*, vol. 30, p. 101 557, 2020.

[89] M.-K. Tran, S. Panchal, T. D. Khang, K. Panchal, R. Fraser, and M. Fowler, "Concept Review of a Cloud-Based Smart Battery Management System for Lithium-Ion Batteries: Feasibility, Logistics, and Functionality", *Batteries*, vol. 8, no. 2, 2022, ISSN: 2313-0105. DOI: 10.3390/batteries8020019. [Online]. Available: https://www.mdpi.com/2313-0105/8/2/19.

[90] F. Naseri, Z. Kazemi, P. G. Larsen, M. M. Arefi, and E. Schaltz, "Cyber-physical cloud battery management systems: Review of security aspects", *Batteries*, vol. 9, no. 7, 2023, ISSN: 2313-0105. DOI: 10.3390/batteries9070382. [Online]. Available: https://www.mdpi.com/2313-0105/9/7/382.

[91] M. Ismail and R. Ahmed, "A Comprehensive Review of Cloud-Based Lithium-Ion Battery Management Systems for Electric Vehicle Applications", *IEEE Access*, vol. 12, pp. 116 259–116 273, 2024. DOI: 10.1109/ACCESS.2024.3446880.

[92] C. Surianarayanan and P. R. Chelliah, "Introduction to Cloud Computing", in *Essentials of Cloud Computing: A Holistic Perspective*. Cham: Springer International Publishing, 2019, pp. 1–32, ISBN: 978-3-030-13134-0. DOI: 10.1007/978-3-030-13134-0_1.

[93] N. Sadashiv and S. M. D. Kumar, "Cluster, grid and cloud computing: A detailed comparison", in *2011 6th International Conference on Computer Science & Education (ICCSE)*, 2011, pp. 477–482. DOI: 10.1109/ICCSE.2011.6028683.

[94] A. Patel, A. Seyfi, Y. Tew, and A. Jaradat, "Comparative study and review of grid, cloud, utility computing and software as a service for use by libraries", *Library Hi Tech News*, vol. 28, no. 3, pp. 25–32, Jan. 2011, ISSN: 0741-9058. DOI: 10.1108/07419051111145145.

[95] P. Mell and T. Grance, "The NIST Definition of Cloud Computing", *Recommendations of the National Institute of Standards and Technology*, 2011. DOI: 10.6028/NIST.SP.800-145.

[96] M. Shahrad, J. Balkind, and D. Wentzlaff, "Architectural Implications of Function-as-a-Service Computing", in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '52, Columbus, OH, USA: Association for Computing Machinery, 2019, pp. 1063–1075, ISBN: 9781450369381. DOI: 10.1145/3352460.3358296.

[97] S. Goyal, "Public vs private vs hybrid vs community-cloud computing: a critical review", *International Journal of Computer Network and Information Security*, vol. 6, no. 3, pp. 20–29, 2014.

[98] N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*, en. Cham: Springer International Publishing, 2018. DOI: 10.1007/978-3-319-77839-6. [Online]. Available: http://link.springer.com/10.1007/978-3-319-77839-6 (visited on 04/21/2022).

[99] S. Pal, D.-N. Le, and P. K. Pattnaik, *Cloud Computing Solutions: Architecture, Data Storage, Implementation, and Security*. John Wiley & Sons, 2022.

[100] A. Sunyaev, "Fog and Edge Computing", in *Internet Computing: Principles of Distributed Systems and Emerging Internet-Based Technologies*. Cham: Springer International Publishing, 2020, pp. 237–264, ISBN: 978-3-030-34957-8. DOI: 10.1007/978-3-030-34957-8_8.

[101] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org, ISBN: 0262035618.

[102] L. P. Zhuhadar and M. D. Lytras, "The Application of AutoML Techniques in Diabetes Diagnosis: Current Approaches, Performance, and Future Directions", *Sustainability*, vol. 15, no. 18, 2023, ISSN: 2071-1050. DOI: 10.3390/su151813484. [Online]. Available: https://www.mdpi.com/2071-1050/15/18/13484.

[103] M. Campbell, A. Hoane, and F.-h. Hsu, "Deep Blue", *Artificial Intelligence*, vol. 134, no. 1, pp. 57–83, 2002, ISSN: 0004-3702. DOI: 10.1016/S0004-3702(01)00129-1.

[104] S. Weisberg, *Applied linear regression*. John Wiley & Sons, 2005, vol. 528.

[105] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers", in *Proceedings of the Tenth National Conference on. Artif. Intell.*, 1992, pp. 223–228.

[106] V. N. Vapnik, *The nature of statistical learning theory*. Berlin, Heidelberg: Springer-Verlag, 1995, ISBN: 0387945598.

[107] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013. DOI: 10.1109/TPAMI.2013.50.

[108] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space", *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901. DOI: 10.1080/14786440109462720.

[109] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima", *Neural Networks*, vol. 2, no. 1, pp. 53–58,

1989, ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90014-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900142.

[110] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[111] Y. LeCun, B. Boser, J. S. Denker, *et al.*, "Backpropagation Applied to Handwritten Zip Code Recognition", *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989. DOI: 10.1162/neco.1989.1.4.541.

[112] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, " You Only Look Once: Unified, Real-Time Object Detection ", in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.91.

[113] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, ISSN: 1476-4687. DOI: 10.1038/nature14236.

[114] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of Go with deep neural networks and tree search", *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016, ISSN: 1476-4687. DOI: 10.1038/nature16961.

[115] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial networks", *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, ISSN: 0001-0782. DOI: 10.1145/3422622.

[116] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training", 2018. [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[117] T. Brown, B. Mann, N. Ryder, *et al.*, "Language Models are Few-Shot Learners", in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[118] H. Touvron, T. Lavril, G. Izacard, *et al.*, *LLaMA: Open and Efficient Foundation Language Models*, 2023. arXiv: 2302.13971 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2302.13971.

[119] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, *Hierarchical Text-Conditional Image Generation with CLIP Latents*, 2022. arXiv: 2204.06125 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2204.06125.

[120] F. Chollet, *Deep learning with Python*. Shelter Island, NY 11964: Manning Publications Co., 2021, ISBN: 9781617296864.

[121] M. Mohri, *Foundations of machine learning*, 2018.

[122] M. Anthony and P. L. Bartlett, *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[123] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-Learning Algorithms: A Comprehensive Classification and Applications", *IEEE Access*, vol. 7, pp. 133 653–133 667, 2019, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2941229.

[124] B. Mirchevska, M. Hügle, G. Kalweit, M. Werling, and J. Boedecker, "Amortized Q-learning with Model-based Action Proposals for Autonomous Driving on Highways", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1028–1035. DOI: 10.1109/ICRA48506.2021.9560777.

[125] R. S. Sutton, "Learning to predict by the methods of temporal differences", *Machine Learning*, vol. 3, no. 1, pp. 9–44, Aug. 1988, ISSN: 1573-0565. DOI: 10.1007/BF00115009.

[126] C. Watkins, "Learning from delayed rewards", Ph.D. dissertation, Kings College, Cambridge United Kingdom, 1989.

[127] C. J. C. H. Watkins and P. Dayan, "Q-learning", *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992, ISSN: 1573-0565. DOI: 10.1007/BF00992698. [Online]. Available: 10.1007/BF00992698.

[128] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[129] y. yin yanli, L. Zhang, S. Zhan, Y. Ma, and S. Ma, "A novel state energy spatialization regenerative braking control strategy based on Q- learning algorithm for a super-mild hybrid electric vehicle", *International Journal of Green Energy*, vol. 18, no. 12, pp. 1263–1276, Sep. 2021, ISSN: 1543-5075. DOI: 10.1080/15435075.2021.1904402. (visited on 11/02/2022).

[130] S. Abedi, S. W. Yoon, and S. Kwon, "Battery energy storage control using a reinforcement learning approach with cyclic time-dependent Markov process", en, *International Journal of Electrical Power & Energy Systems*, vol. 134, p. 107 368, Jan. 2022, ISSN: 0142-0615. DOI: 10.1016/j.ijepes.2021.107368. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061521006074 (visited on 11/02/2022).

[131] Y. Tang, W. Hu, J. Xiao, *et al.*, "Reinforcement Learning Based Efficiency Optimization Scheme for the DAB DC–DC Converter With Triple-Phase-Shift Modulation", *IEEE Transactions on Industrial Electronics*, vol. 68, no. 8, pp. 7350–7361, 2021, ISSN: 1557-9948. DOI: 10.1109/TIE.2020.3007113.

[132] A. Olabi, A. A. Abdelghafar, B. Soudan, *et al.*, "Artificial neural network driven prognosis and estimation of Lithium-Ion battery states: Current insights and future perspectives", *Ain Shams Engineering Journal*, vol. 15, no. 2, p. 102 429, 2024, ISSN: 2090-4479. DOI: 10.1016/j.asej.2023.102429. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090447923003180.

[133] Z. Cui, L. Wang, Q. Li, and K. Wang, "A comprehensive review on the state of charge estimation for lithium-ion battery based on neural network", *International Journal of Energy Research*, vol. 46, no. 5, pp. 5423–5440, 2022.

[134] A. Zeller, *Simulation Neuronaler Netze [Simulation of Neuronal Networks] (in German)*. Addison-Wesley Bonn, 1994, vol. 1, ISBN: 3-89319-554-8.

[135] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1998.

[136] Z. Liu, Y. Wang, S. Vaidya, *et al.*, "Kan: Kolmogorov-arnold networks", *arXiv preprint arXiv:2404.19756*, 2024.

[137] M. Liu, S. Bian, B. Zhou, and P. Lukowicz, *iKAN: Global Incremental Learning with KAN for Human Activity Recognition Across Heterogeneous Datasets*, 2024. arXiv: 2406.01646 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2406.01646.

[138] C. J. Vaca-Rubio, L. Blanco, R. Pereira, and M. Caus, "Kolmogorov-arnold networks (kans) for time series analysis", *arXiv preprint arXiv:2405.08790*, 2024.

[139] A. N. Kolmogorov, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition", in *Doklady Akademii Nauk*, Russian Academy of Sciences, vol. 114, 1957, pp. 953–956.

[140] Z. K. Shahid, S. Saguna, and C. Åhlund, "Forecasting and Detecting Anomalies in ADLs in Single-Resident Elderly Smart Homes", in *Proceedings of the 2023 International Conference on Research in Adaptive and Convergent Systems*, ser. RACS '23, Gdansk, Poland: Association for Computing Machinery, 2023. DOI: 10.1145/3599957.3606216.

[141] B. Lindemann, B. Maschler, N. Sahlab, and M. Weyrich, "A survey on anomaly detection for technical systems using LSTM networks", *Computers in Industry*, vol. 131, p. 103 498, 2021.

[142] W. He, Z. Li, T. Liu, *et al.*, "Research progress and application of deep learning in remaining useful life, state of health and battery thermal management of lithium batteries", *Journal of Energy Storage*, vol. 70, p. 107 868, 2023, ISSN: 2352-152X. DOI: 10.1016/j.est.2023.107868. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X23012653.

[143] K. Cho, B. van Merriënboer, C. Gulcehre, *et al.*, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds., Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. [Online]. Available: https://aclanthology.org/D14-1179.

[144] R. Cahuantzi, X. Chen, and S. Güttel, "A comparison of lstm and gru networks for learning symbolic sequences", in *Intelligent Computing*, K. Arai, Ed., Cham: Springer Nature Switzerland, 2023, pp. 771–785, ISBN: 978-3-031-37963-5.

[145] Z. Zhang, Z. Dong, H. Lin, *et al.*, "An improved bidirectional gated recurrent unit method for accurate state-of-charge estimation", *IEEE Access*, vol. 9, pp. 11 252–11 263, 2021.

[146] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation", *Peerj computer science*, vol. 7, e623, 2021. DOI: doi.org/10.7717/peerj-cs.623.

[147] A. Göksoy, H. Y. Yalırsu, and Ö. Vayvay, "Digital Twin: Is It Hype?", in *Multidimensional and Strategic Outlook in Digital Business Transformation: Human Resource and Management Recommendations for Performance Improvement*, Springer, 2023, pp. 81–93.

[148] D. Hartmann and H. Van der Auweraer, "Digital Twin", in *Progress in Industrial Mathematics: Success Stories*, Springer, 2021, pp. 3–14.

[149] M. Jafari, A. Kavousi-Fard, T. Chen, and M. Karimi, "A review on digital twin technology in smart grid, transportation system and smart city: Challenges and future", *IEEE Access*, vol. 11, pp. 17 471–17 484, 2023.

[150] J.-F. Yao, Y. Yang, X.-C. Wang, and X.-P. Zhang, "Systematic review of digital twin technology and applications", *Visual Computing for Industry, Biomedicine, and Art*, vol. 6, no. 1, p. 10, May 2023, ISSN: 2524-4442. DOI: 10.1186/s42492-023-00137-4.

[151] E. Glaessgen and D. Stargel, "The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles", in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. DOI: 10.2514/6.2012-1818.

[152] J. Ríos, J. C. Hernández, M. Oliva, and F. Mas, "Product avatar as digital counterpart of a physical individual product: Literature review and implications in an aircraft", *Transdisciplinary Lifecycle Analysis of Systems*, pp. 657–666, 2015. DOI: 10.3233/978-1-61499-544-9-657.

[153] J. Lee, E. Lapira, B. Bagheri, and H.-a. Kao, "Recent advances and trends in predictive manufacturing systems in big data environment", *Manufacturing Letters*, vol. 1, no. 1, pp. 38–41, 2013, ISSN: 2213-8463. DOI: 10.1016/j.mfglet.2013.09.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2213846313000114.

[154] K. Y. H. Lim, P. Zheng, and C.-H. Chen, "A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives", *Journal of Intelligent Manufacturing*, vol. 31, no. 6, pp. 1313–1337, Aug. 2020, ISSN: 1572-8145. DOI: 10.1007/s10845-019-01512-w.

[155] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification", *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018, 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2018.08.474. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318316021.

[156] F. Naseri, S. Gil, C. Barbu, *et al.*, "Digital twin of electric vehicle battery systems: Comprehensive review of the use cases, requirements, and platforms", *Renewable and Sustainable Energy Reviews*, vol. 179, p. 113 280, 2023, ISSN: 1364-0321. DOI: 10.1016/j.rser.2023.113280. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032123001363.

[157] S. N. G. Gourisetti, S. Bhadra, D. J. Sebastian-Cardenas, M. Touhiduzzaman, and O. Ahmed, "A theoretical open architecture framework and technology stack for digital twins in energy sector applications", *Energies*, vol. 16, no. 13, p. 4853, 2023.

[158] M. Ibrahim, V. Rjabtšikov, and R. Gilbert, "Overview of Digital Twin Platforms for EV Applications", *Sensors*, vol. 23, no. 3, 2023, ISSN: 1424-8220. DOI: 10.3390/s23031414. [Online]. Available: https://www.mdpi.com/1424-8220/23/3/1414.

[159] R. da Silva Mendonça, S. de Oliveira Lins, I. V. de Bessa, F. A. de Carvalho Ayres, R. L. P. de Medeiros, and V. F. de Lucena, "Digital Twin Applications: A Survey of Recent Advances and Challenges", *Processes*, vol. 10, no. 4, 2022, ISSN: 2227-9717. DOI: 10.3390/pr10040744. [Online]. Available: https://www.mdpi.com/2227-9717/10/4/744.

[160] D.-E. A. Mansour, M. Numair, A. S. Zalhaf, *et al.*, "Applications of IoT and digital twin in electrical power systems: A comprehensive survey", *IET Generation, Transmission & Distribution*, vol. 17, no. 20, pp. 4457–4479, 2023. DOI: 10.1049/gtd2.12940.

[161] Z. M. Pinter, D. Papageorgiou, G. Rohde, M. Marinelli, and C. Træholt, "Review of Control Algorithms for Reconfigurable Battery Systems with an Industrial Example", in *2021 56th International Universities Power Engineering Conference (UPEC)*, 2021, pp. 1–6. DOI: 10.1109/UPEC50034.2021.9548259.

[162] J. Gallardo-Lozano, E. Romero-Cadaval, M. I. Milanes-Montero, and M. A. Guerrero-Martinez, "A novel active battery equalization control with on-line unhealthy cell detection and cell change decision", en, *Journal of Power Sources*, vol. 299, pp. 356–370, Dec. 2015, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2015.09.005. [On-

line]. Available: https://www.sciencedirect.com/science/article/pii/S0378775315302676 (visited on 10/25/2022).

[163] Z. Zhang, L. Zhang, L. Hu, and C. Huang, "Active cell balancing of lithium-ion battery pack based on average state of charge", en, *International Journal of Energy Research*, vol. 44, no. 4, pp. 2535–2548, 2020, ISSN: 1099-114X. DOI: 10.1002/er.4876. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/er.4876 (visited on 10/25/2022).

[164] N. Ghaeminezhad, Q. Ouyang, X. Hu, G. Xu, and Z. Wang, "Active Cell Equalization Topologies Analysis for Battery Packs: A Systematic Review", *IEEE Transactions on Power Electronics*, vol. 36, no. 8, pp. 9119–9135, Aug. 2021, Conference Name: IEEE Transactions on Power Electronics, ISSN: 1941-0107. DOI: 10.1109/TPEL.2021.3052163.

[165] Y. Cao and J. A. Abu Qahouq, "Hierarchical SOC Balancing Controller for Battery Energy Storage System", *IEEE Transactions on Industrial Electronics*, vol. 68, no. 10, pp. 9386–9397, Oct. 2021, Conference Name: IEEE Transactions on Industrial Electronics, ISSN: 1557-9948. DOI: 10.1109/TIE.2020.3021608.

[166] C. N. Van, T. N. Vinh, M.-D. Ngo, and S.-J. Ahn, "Optimal SoC Balancing Control for Lithium-Ion Battery Cells Connected in Series", en, *Energies*, vol. 14, no. 10, p. 2875, Jan. 2021, Number: 10 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 1996-1073. DOI: 10.3390/en14102875. [Online]. Available: https://www.mdpi.com/1996-1073/14/10/2875 (visited on 10/25/2022).

[167] J.-H. Jung, E. Hosseini, M. Liserre, and L. M. Fernández-Ramírez, "Reinforcement Learning Based Modulation for Balancing Capacitor Voltage and Thermal Stress to Enhance Current Capability of MMCs", in *2022 IEEE 13th International Symposium on Power Electronics for Distributed Generation Systems (PEDG)*, 2022, pp. 1–6. DOI: 10.1109/PEDG54999.2022.9923188.

[168] Y. Tang, W. Hu, D. Cao, *et al.*, "Artificial Intelligence-Aided Minimum Reactive Power Control for the DAB Converter Based on Harmonic Analysis Method", *IEEE Transactions on Power Electronics*, vol. 36, no. 9, pp. 9704–9710, 2021, ISSN: 1941-0107. DOI: 10.1109/TPEL.2021.3059750.

[169] A. Mashayekh, A. Kersten, M. Kuder, *et al.*, "Proactive SoC Balancing Strategy for Battery Modular Multilevel Management (BM3) Converter Systems and Reconfigurable Batteries", in *2021 23rd European Conference on Power Electronics and Applications (EPE 21 ECCE Europe)*, 2021. DOI: 10.23919/EPE21ECCEEurope50061.2021.9570543.

[170] H. Huang, A. M. Ghias, P. Acuna, Z. Dong, J. Zhao, and M. S. Reza, "A fast battery balance method for a modular-reconfigurable battery energy storage system", *Applied Energy*, vol. 356, p. 122 470, 2024.

[171] W. Han, C. Zou, L. Zhang, Q. Ouyang, and T. Wik, "Near-fastest battery balancing by cell/module reconfiguration", *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 6954–6964, 2019.

[172] M. Kristjansen, A. Kulkarni, P. G. Jensen, R. Teodorescu, and K. G. Larsen, "Dual Balancing of SoC/SoT in Smart Batteries Using Reinforcement Learning in Uppaal Stratego", in *IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society*, 2023, pp. 1–6. DOI: 10.1109/IECON51785.2023.10311828.

[173] W. Han, T. Wik, A. Kersten, G. Dong, and C. Zou, "Next-Generation Battery Management Systems: Dynamic Reconfiguration", *IEEE Industrial Electronics Magazine*, vol. 14, no. 4, pp. 20–31, 2020. DOI: 10.1109/MIE.2020.3002486.

[174] B. Jiang, J. Tang, Y. Liu, and L. Boscaglia, "Active Balancing of Reconfigurable Batteries Using Reinforcement Learning Algorithms", in *2023 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2023, pp. 1–6. DOI: 10.1109/ITEC55900.2023.10187076.

[175] F. Yang, F. Gao, B. Liu, and S. Ci, "An adaptive control framework for dynamically reconfigurable battery systems based on deep reinforcement learning", *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 12 980–12 987, 2022.

[176] A. Stevenson, M. Tariq, and A. Sarwat, "Reduced Operational Inhomogeneities in a Reconfigurable Parallelly-Connected Battery Pack Using DQN Reinforcement Learning Technique", in *2023 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2023, pp. 1–5. DOI: 10.1109/ITEC55900.2023.10187040.

[177] X. Yang, P. Liu, F. Liu, *et al.*, "A DOD-SOH balancing control method for dynamic reconfigurable battery systems based on DQN algorithm", *Frontiers in Energy Research*, vol. 11, p. 1 333 147, 2023.

[178] A. Mashayekh, S. Pohlmann, J. Estaller, *et al.*, "Multi-Agent Reinforcement Learning-Based Decentralized Controller for Battery Modular Multilevel Inverter Systems", *Electricity*, vol. 4, no. 3, pp. 235–252, 2023.

[179] K. Thakre, K. B. Mohanty, and A. Chatterjee, "Modelling and Simulation of an Asymmetrical Modular Multilevel Inverter with Less Number of Components", *EPE Journal*, vol. 30, no. 2, pp. 69–79, Apr. 2020, ISSN: 0939-8368. DOI: 10.1080/09398368.2020.1725857.

[180] R. Niraimathi and R. Seyezhai, "Analysis, simulation and implementation of a novel dual bridge asymmetric cascaded multi level inverter using MGWO-PI-PWM controller", en, *Microprocessors and Microsystems*, vol. 77, Sep. 2020, ISSN: 01419331. DOI: 10.1016/j.micpro.2020.103103.

[181] S. Raschka, J. Patterson, and C. Nolet, "Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence", *Information*, vol. 11, no. 4, 2020, ISSN: 2078-2489. DOI: 10.3390/info11040193. [Online]. Available: https://www.mdpi.com/2078-2489/11/4/193.

[182] S. Cass, "The Top Programming Languages 2023", *IEEE Spectrum*, 2023.

[183] J. Estaller, A. Kersten, M. Kuder, *et al.*, "Battery Impedance Modeling and Comprehensive Comparisons of State-Of-The-Art Cylindrical 18650 Battery Cells considering Cells' Price, Impedance, Specific Energy and C-Rate", in *2021 IEEE International Conference on Environment and Electrical Engineering and 2021 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*, Sep. 2021, pp. 1–8. DOI: 10.1109/EEEIC/ICPSEurope51590.2021.9584562.

[184] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, "Array programming with NumPy", en, *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, ISSN: 1476-4687. DOI: 10.1038/s41586-020-2649-2.

[185] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: a LLVM-based Python JIT compiler", in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, ser. LLVM '15, New York, NY, USA: Association for Computing Machinery, Nov. 2015, ISBN: 978-1-4503-4005-2. DOI: 10.1145/2833157.2833162.

[186] R. Okuta, Y. Unno, D. Nishino, S. Hido, and C. Loomis, "CuPy: A NumPy-Compatible Library for NVIDIA GPU Calculations", en, in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017.

[187] R. Pereira, M. Couto, F. Ribeiro, *et al.*, "Ranking programming languages by energy efficiency", *Science of Computer Programming*, vol. 205, p. 102 609, 2021, ISSN: 0167-6423. DOI: 10.1016/j.scico.2021.102609. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167642321000022.

[188] J. Zhu, F. Wu, and J. Zhao, "An Overview of the Action Space for Deep Reinforcement Learning", in *Proceedings of the 2021 4th International Conference on Algorithms, Computing and Artificial Intelligence*, ser. ACAI '21, Sanya, China: Association for Computing Machinery, 2022. DOI: 10.1145/3508546.3508598.

[189] V. Mnih, "Playing atari with deep reinforcement learning", no. arXiv:1312.5602, 2013.

[190] G. Dulac-Arnold, R. Evans, H. van Hasselt, *et al.*, "Deep reinforcement learning in large discrete action spaces", *arXiv preprint arXiv:1512.07679*, 2015.

[191] T. Van de Wiele, D. Warde-Farley, A. Mnih, and V. Mnih, "Q-Learning in enormous action spaces via amortized approximate maximization", en, arXiv, Tech. Rep. arXiv:2001.08116, 2020. [Online]. Available: http://arxiv.org/abs/2001.08116 (visited on 12/20/2022).

[192] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel, "The Helmholtz Machine", *Neural Computation*, vol. 7, no. 5, pp. 889–904, Sep. 1995, ISSN: 0899-7667. DOI: 10.1162/neco.1995.7.5.889.

[193] Z. Zheng, K. Wang, L. Xu, and Y. Li, "A Hybrid Cascaded Multilevel Converter for Battery Energy Management Applied in Electric Vehicles", *IEEE Transactions on Power Electronics*, vol. 29, no. 7, pp. 3537–3546, Jul. 2014, IEEE Transactions on Power Electronics, ISSN: 1941-0107. DOI: 10.1109/TPEL.2013.2279185.

[194] H. Dai, B. Jiang, X. Hu, X. Lin, X. Wei, and M. Pecht, "Advanced battery management strategies for a sustainable energy future: Multilayer design concepts and research trends", *Renewable and Sustainable Energy Reviews*, vol. 138, p. 110 480, 2021.

[195] C. Semeraro, A. Olabi, H. Aljaghoub, *et al.*, "Digital twin application in energy storage: Trends and challenges", *Journal of Energy Storage*, vol. 58, p. 106 347, 2023.

[196] P. Arthurs, L. Gillam, P. Krause, N. Wang, K. Halder, and A. Mouzakitis, "A Taxonomy and Survey of Edge Cloud Computing for Intelligent Transportation Systems and Connected Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6206–6221, 2022. DOI: 10.1109/TITS.2021.3084396.

[197] Azzedine Boukerche and Robson E. De Grande, "Vehicular cloud computing: Architectures, applications, and mobility", *Computer Networks*, vol. 135, pp. 171–189, 2018, ISSN: 1389-1286. DOI: 10.1016/j.comnet.2018.01.004.

[198] R. r. Minerva, A. Biru, and D. Rotondi, "Towards a definition of the Internet of Things (IoT)", *IEEE Internet Initiative*, vol. 1, no. 1, pp. 1–86, 2015.

[199] P. Sivaraman and C. Sharmeela, "IoT-Based Battery Management System for Hybrid Electric Vehicle", in *Artificial Intelligent Techniques for Electric and Hybrid Electric Vehicles*, John Wiley & Sons, Ltd, 2020, pp. 1–16, ISBN: 978-1-119-68203-5. DOI: 10.1002/9781119682035.ch1. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119682035.ch1 (visited on 07/13/2023).

[200] Amazon Web Services Inc. "Announcing AWS IoT FleetWise (Preview), a new service for transferring vehicle data to the cloud more efficiently", Amazon Web Services, Inc. (Nov. 2021), [Online]. Available: https://aws.amazon.com/about-aws/whats-new/2021/11/aws-iot-fleetwise-transferring-vehicle-data-cloud/ (visited on 07/13/2023).

[201] Amazon Web Services, Inc., *Building Connected Vehicle Platforms with AWS IoT*. 2023. [Online]. Available: https://pages.awscloud.com/GLOBAL-other-DL-Connected-vehicle-ebook-2023-learn.html (visited on 07/13/2023).

[202] Amazon Web ServicesInc. "General References". (2024), [Online]. Available: https://docs.aws.amazon.com/pdfs/general/latest/gr/aws-general.pdf.

[203] A. Wiedenmann, W. Grupp, T. Högerl, *et al.*, "Automation of Modular Multilevel Converter Topology Evaluation including Thermal Monitoring", in *2022 International Conference on Smart Energy Systems and Technologies (SEST)*, 2022, pp. 1–6. DOI: 10.1109/SEST53650.2022.9898447.

[204] M. Weyrich and C. Ebert, "Reference Architectures for the Internet of Things", *IEEE Software*, vol. 33, no. 1, pp. 112–116, 2016. DOI: 10.1109/MS.2016.20.

[205] J. Guth, U. Breitenbücher, M. Falkenthal, F. Leymann, and L. Reinfurt, "Comparison of IoT platform architectures: A field study based on a reference architecture", in *2016 Cloudification of the Internet of Things (CIoT)*, IEEE, 2016, pp. 1–6.

[206] K. Kiela, V. Barzdenas, M. Jurgo, *et al.*, "Review of V2X–IoT standards and frameworks for ITS applications", *Applied sciences*, vol. 10, no. 12, p. 4314, 2020.

[207] "Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Product quality model", International Organization for Standardization, Geneva, CH, Standard, Nov. 2023, ISO/IEC 25010:2023.

[208] American National Standards Institute (ANSI), "ANSI EVSP Roadmap", Jun. 2023. [Online]. Available: https://share.ansi.org/evsp/ANSI_EVSP_Roadmap_June_2023.pdf (visited on 02/05/2024).

[209] "Vehicle to grid communication interface", International Organization for Standardization, Geneva, CH, Standard, Apr. 2019, ISO 15118-1:2019.

[210] "Diagnostic communication over Internet Protocol (DoIP)", International Organization for Standardization, Geneva, CH, Standard, Dec. 2019, ISO 13400-2:2019.

[211] S. Clark, F. L. Bleken, S. Stier, *et al.*, "Toward a unified description of battery data", *Advanced Energy Materials*, vol. 12, no. 17, 2022.

[212] P. Herring, C. Balaji Gopal, M. Aykol, *et al.*, "BEEP: A Python library for Battery Evaluation and Early Prediction", *SoftwareX*, vol. 11, p. 100 506, Jan. 1, 2020, ISSN: 2352-7110. DOI: 10.1016/j.softx.2020.100506. (visited on 01/10/2023).

[213] Jan Petter Maehlen, Muhammad Abdelhamid, and Asbjoern Ulvestad. "cellpy". original-date: 2015-12-17T11:33:35Z. (Jan. 10, 2023), [Online]. Available: https://github.com/jepegit/cellpy (visited on 01/12/2023).

[214] N. H. PAULSON, L. T. WARD, J. J. KUBAL, B. J. BLAISZIK, and A. LDRD, *battery-data-toolkit*, Apr. 2021. DOI: 10.11578/dc.20210819.5. [Online]. Available: https://www.osti.gov/biblio/1814181.

[215] A. Lewis-Douglas, L. Pitt, and D. A. Howey, *Galvanalyser: A Battery Test Database*, Oct. 28, 2020. DOI: 10.48550/arXiv.2010.14959. arXiv: 2010.14959[cs,eess]. (visited on 01/10/2023).

[216] Samuel-Buteau. "Universal Battery Database". (Jan. 10, 2023), [Online]. Available: https://github.com/Samuel-Buteau/universal-battery-database (visited on 01/12/2023).

[217] "BatteryArchive.org". (), [Online]. Available: http://www.batteryarchive.org/index.html (visited on 01/12/2023).

[218] C. N. Lininger, T. Thai, T. R. Juran, E. S. Leland, and T. Z. Sholklapper, "Voltaiq Data Format–A standard data format for collection of battery data to enable big data comparisons and analyses across the battery lifecycle", *Frontiers in Energy Research*, 2022.

[219] G. dos Reis, C. Strange, M. Yadav, and S. Li, "Lithium-ion battery data and where to find it", *Energy and AI*, vol. 5, 2021, ISSN: 2666-5468. DOI: 10.1016/j.egyai.2021.100081.

[220] M. Hassini, E. Redondo-Iglesias, and P. Venet, "Lithium–Ion Battery Data: From Production to Prediction", *Batteries*, vol. 9, no. 7, 2023, ISSN: 2313-0105. DOI: 10.3390/batteries9070385.

[221] L. Ward, S. Babinec, E. J. Dufek, *et al.*, "Principles of the Battery Data Genome", *Joule*, vol. 6, no. 10, pp. 2253–2271, 2022, ISSN: 2542-4351. DOI: 10.1016/j.joule.2022.08.008.

[222] B. Smith, *Beginning JSON*, en. Berkeley, CA: Apress, 2015, ISBN: 978-1-4842-0202-9. DOI: 10.1007/978-1-4842-0202-9.

[223] J. Fang, F. Blaabjerg, S. Liu, and S. M. Goetz, "A Review of Multilevel Converters With Parallel Connectivity", *IEEE Transactions on Power Electronics*, vol. 36, no. 11, pp. 12 468–12 489, 2021. DOI: 10.1109/TPEL.2021.3075211.

[224] "Data elements and interchange formats — Information interchange - Representation of dates and times — Part 1: Basic rules", International Organization for Standardization, Geneva, CH, Standard, 2016, ISO 9801-1:2016.

[225] S. Yi, Z. Hao, Z. Qin, and Q. Li, " Fog Computing: Platform and Applications ", in *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, Los Alamitos, CA, USA: IEEE Computer Society, Nov. 2015, pp. 73–78. DOI: 10.1109/HotWeb.2015.22.

[226] B. Costa, J. Bachiega, L. R. de Carvalho, and A. P. F. Araujo, "Orchestration in Fog Computing: A Comprehensive Survey", *ACM Comput. Surv.*, vol. 55, no. 2, Jan. 2022, ISSN: 0360-0300. DOI: 10.1145/3486221. [Online]. Available: 10.1145/3486221.

[227] S. N. Srirama, "A decade of research in fog computing: Relevance, challenges, and future directions", *Software: Practice and Experience*, vol. 54, no. 1, pp. 3–23, 2024.

[228] J. Wang, T. Liu, K. Liu, B. Kim, J. Xie, and Z. Han, "Computation offloading over fog and cloud using multi-dimensional multiple knapsack problem", in *2018 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2018, pp. 1–7.

[229] H. Wang, T. Liu, B. Kim, *et al.*, "Architectural Design Alternatives Based on Cloud/ Edge/ Fog Computing for Connected Vehicles", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2349–2377, 2020. DOI: 10.1109/COMST.2020.3020854.

[230] G. L. Plett, *Battery management systems, Volume II: Equivalent-circuit methods*. Artech House, 2015, ISBN: 9781630810283.

[231] S. Guo and L. Ma, "A comparative study of different deep learning algorithms for lithium-ion batteries on state-of-charge estimation", *Energy*, vol. 263, p. 125 872, 2023, ISSN: 0360-5442. DOI: 10.1016/j.energy.2022.125872. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S036054422202758X.

[232] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine Learning Operations (MLOps): Overview, Definition, and Architecture", *IEEE Access*, vol. 11, pp. 31 866–31 879, 2023. DOI: 10.1109/ACCESS.2023.3262138.

[233] J. Bartlett, *Cloud Native Applications with Docker and Kubernetes: Design and Build Cloud Architecture and Applications with Microservices, EMQ, and Multi-Site Configurations*, en. Berkeley, CA: Apress, 2023, ISBN: 978-1-4842-8876-4. DOI: 10.1007/978-1-4842-8876-4. [Online]. Available: https://link.springer.com/10.1007/978-1-4842-8876-4 (visited on 10/11/2023).

[234] "Keras: Deep Learning for humans". [Accessed: 2023-08-08]. (), [Online]. Available: https://keras.io/.

[235] "Serving Models | TFX", TensorFlow. [Accessed: 2023-08-08]. (), [Online]. Available: https://www.tensorflow.org/tfx/guide/serving.

[236] P. Kollmeyer and M. Skells, "Turnigy graphene 5000mAh 65C li-ion battery data", *Mendeley Data*, vol. 1, 2020. DOI: 10.17632/4fx8cjprxm.1.

[237] F. Khanum, E. Louback, F. Duperly, C. Jenkins, P. J. Kollmeyer, and A. Emadi, "A Kalman Filter Based Battery State of Charge Estimation MATLAB Function", in *2021 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2021, pp. 484–489. DOI: 10.1109/ITEC51675.2021.9490163.

[238] Z. Wei, C. Zou, F. Leng, B. H. Soong, and K.-J. Tseng, "Online Model Identification and State-of-Charge Estimate for Lithium-Ion Battery With a Recursive Total Least Squares-Based Observer", *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1336–1346, 2018. DOI: 10.1109/TIE.2017.2736480.

[239] David Rosewater. "Battery Management System Standard - IEEE P2686 Recommended Practice for Battery Management Systems in Energy Storage Applications". Energy Storage Systems Safety & Reliability Forum. [Accessed: 2023-06-19]. (May 5, 2022), [Online]. Available: https://www.osti.gov/servlets/purl/2002547 (visited on 06/19/2023).

[240] T. Cipra and R. Romera, "Kalman filter with outliers and missing observations", *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research*, vol. 6, no. 2, pp. 379–395, 1997. DOI: 10.1007/BF02564705.

[241] O. Lizaso-Eguileta, E. Martinez-Laserna, M. Rivas, E. Miguel, U. Iraola, and I. Cantero, "Module-Level Modelling Approach for a Cloudbased Digital Twin Platform for Li-Ion Batteries", in *2021 IEEE Vehicle Power and Propulsion Conference (VPPC)*, 2021, pp. 1–6. DOI: 10.1109/VPPC53923.2021.9699271.

[242] F. Leng, C. M. Tan, and M. Pecht, "Effect of temperature on the aging rate of Li ion battery operating above room temperature", *Scientific reports*, vol. 5, no. 1, p. 12 967, 2015.

[243] C. Chetri, A. Samanta, and S. Williamson, "Critical Understanding of Temperature Gradient During Fast Charging of Lithium-ion Batteries at Low Temperatures", in *IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2023, pp. 1–6.

[244] J. S. Edge, S. O'Kane, R. Prosser, *et al.*, "Lithium ion battery degradation: what you need to know", *Physical Chemistry Chemical Physics*, vol. 23, no. 14, pp. 8200–8221, 2021.

[245] X. Feng, M. Ouyang, X. Liu, L. Lu, Y. Xia, and X. He, "Thermal runaway mechanism of lithium ion battery for electric vehicles: A review", *Energy storage materials*, vol. 10, pp. 246–267, 2018.

[246] X. Lin, H. E. Perez, J. B. Siegel, and A. G. Stefanopoulou, "Robust estimation of battery system temperature distribution under sparse sensing and uncertainty", *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 753–765, 2019.

[247] C. Forgez, D. V. Do, G. Friedrich, M. Morcrette, and C. Delacourt, "Thermal modeling of a cylindrical LiFePO4/graphite lithium-ion battery", *Journal of power sources*, vol. 195, no. 9, pp. 2961–2968, 2010.

[248] C. Wang and V. Srinivasan, "Computational battery dynamics (CBD)— electrochemical/ thermal coupled modeling and multi-scale modeling", *Journal of power sources*, vol. 110, no. 2, pp. 364–376, 2002.

[249] S. Al Hallaj, H. Maleki, J.-S. Hong, and J. R. Selman, "Thermal modeling and design considerations of lithium-ion batteries", *Journal of power sources*, vol. 83, no. 1-2, pp. 1–8, 1999.

[250] H. Maleki and A. K. Shamsuri, "Thermal analysis and modeling of a notebook computer battery", *Journal of Power Sources*, vol. 115, no. 1, pp. 131–136, 2003.

[251] W. B. Gu and C. Y. Wang, "Thermal-electrochemical modeling of battery systems", *Journal of The Electrochemical Society*, vol. 147, no. 8, p. 2910, 2000.

[252] R. Mahamud and C. Park, "Reciprocating air flow for Li-ion battery thermal management to improve temperature uniformity", *Journal of Power Sources*, vol. 196, no. 13, pp. 5685–5696, 2011.

[253] K. Smith and C.-Y. Wang, "Power and thermal characterization of a lithium-ion battery pack for hybrid-electric vehicles", *Journal of power sources*, vol. 160, no. 1, pp. 662–673, 2006.

[254] D. Bernardi, E. Pawlikowski, and J. Newman, "A general energy balance for battery systems", *Journal of the electrochemical society*, vol. 132, no. 1, p. 5, 1985.

[255] C. Park and A. K. Jaura, "Dynamic thermal model of li-ion battery for predictive behavior in hybrid and fuel cell vehicles", SAE Technical Paper, Tech. Rep., 2003.

[256] G. Guo, B. Long, B. Cheng, S. Zhou, P. Xu, and B. Cao, "Three-dimensional thermal finite element modeling of lithium-ion battery in thermal abuse application", *Journal of power sources*, vol. 195, no. 8, pp. 2393–2398, 2010.

[257] S. Du, M. Jia, Y. Cheng, *et al.*, "Study on the thermal behaviors of power lithium iron phosphate (LFP) aluminum-laminated battery with different tab configurations", *International Journal of Thermal Sciences*, vol. 89, pp. 327–336, 2015.

[258] R. R. Richardson, S. Zhao, and D. A. Howey, "On-board monitoring of 2-D spatially-resolved temperatures in cylindrical lithium-ion batteries: Part II. State estimation via impedance-based temperature sensing", *Journal of Power Sources*, vol. 327, pp. 726–735, 2016.

[259] J. P. Schmidt, S. Arnold, A. Loges, D. Werner, T. Wetzel, and E. Ivers-Tiffée, "Measurement of the internal cell temperature via impedance: Evaluation and application of a new method", *Journal of Power Sources*, vol. 243, pp. 110–117, 2013.

[260] R. Srinivasan, B. G. Carkhuff, M. H. Butler, and A. C. Baisden, "Instantaneous measurement of the internal temperature in lithium-ion rechargeable cells", *Electrochimica Acta*, vol. 56, no. 17, pp. 6198–6204, 2011.

[261] Y. Zheng, Y. Che, J. Guo, *et al.*, "Real-time Sensorless Temperature Estimation of Lithium-ion Batteries Based on Online Operando Impedance Acquisition", *IEEE Transactions on Power Electronics*, 2024.

[262] Z. Wei, P. Li, W. Cao, *et al.*, "Machine learning-based hybrid thermal modeling and diagnostic for lithium-ion battery enabled by embedded sensing", *Applied Thermal Engineering*, vol. 216, p. 119 059, 2022.

[263] X. Zhang, H. Xiang, X. Xiong, Y. Wang, and Z. Chen, "Benchmarking core temperature forecasting for lithium-ion battery using typical recurrent neural networks", *Applied Thermal Engineering*, vol. 248, p. 123 257, 2024.

[264] N. Wang, G. Zhao, Y. Kang, *et al.*, "Core temperature estimation method for lithium-ion battery based on long short-term memory model with transfer learning", *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 11, no. 1, pp. 201–213, 2021.

[265] S. Surya, A. Samanta, V. Marcis, and S. Williamson, "Hybrid electrical circuit model and deep learning-based core temperature estimation of lithium-ion battery cell", *IEEE Transactions on Transportation Electrification*, vol. 8, no. 3, pp. 3816–3824, 2022.

[266] K.-K. Xu, H.-X. Li, and H.-D. Yang, "Local-properties-embedding-based nonlinear spatiotemporal modeling for lithium-ion battery thermal process", *IEEE Transactions on Industrial Electronics*, vol. 65, no. 12, pp. 9767–9776, 2018.

[267] J. Kleiner, M. Stuckenberger, L. Komsiyska, and C. Endisch, "Real-time core temperature prediction of prismatic automotive lithium-ion battery cells based on artificial neural networks", *Journal of Energy Storage*, vol. 39, p. 102 588, 2021, ISSN: 2352-152X. DOI: 10.1016/j.est.2021.102588. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X21003200.

[268] C. S. Chin, Z. Gao, and C. Z. Zhang, "Comprehensive electro-thermal model of 26650 lithium battery for discharge cycle under parametric and temperature variations", *Journal of Energy Storage*, vol. 28, p. 101 222, 2020.

[269] C. Nguyen Van and D. T. Quang, "Estimation of SoH and internal resistances of Lithium ion battery based on LSTM network", *International Journal of Electrochemical Science*, vol. 18, no. 6, p. 100 166, 2023, ISSN: 1452-3981. DOI: 10.1016/j.ijoes.2023.100166.

[270] P. Hu, W. F. Tang, C. H. Li, S.-L. Mak, C. Y. Li, and C. C. Lee, "Joint state of charge (soc) and state of health (soh) estimation for lithium-ion batteries packs of electric vehicles based on nssr-lstm neural network", *Energies*, vol. 16, no. 14, 2023, ISSN: 1996-1073. DOI: 10.3390/en16145313.

[271] A. Samanta and S. Williamson, "Cloud-integrated adaptive deep learning framework for real-time battery core temperature estimation and enhanced thermal safety", 2025, In Review.

[272] G. Bhatti, H. Mohan, and R. Raja Singh, "Towards the future of smart electric vehicles: Digital twin technology", *Renewable and Sustainable Energy Reviews*, vol. 141, 2021, ISSN: 1364-0321. DOI: 10.1016/j.rser.2021.110801. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032121000964.

[273] D. Shi, J. Zhao, C. Eze, *et al.*, "Cloud-Based Artificial Intelligence Framework for Battery Management System", *Energies*, vol. 16, no. 11, 2023, ISSN: 1996-1073. DOI: 10.3390/en16114403. [Online]. Available: https://www.mdpi.com/1996-1073/16/11/4403.

[274] J. D. A. Correa, A. S. R. Pinto, and C. Montez, "Lossy data compression for iot sensors: A review", *Internet of Things*, vol. 19, p. 100 516, 2022.

[275] G. Chiarot and C. Silvestri, "Time series compression survey", *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–32, 2023. DOI: 10.1145/3560814.

[276] D. Shkarin, *PPMonstr*, version J rev.1, May 10, 2006. [Online]. Available: http://www.compression.ru/ds/ (visited on 06/17/2024).

[277] A. Gupta, A. Bansal, and V. Khanduja, "Modern lossless compression techniques: Review, comparison and analysis", in *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2017, pp. 1–8. DOI: 10.1109/ICECCT.2017.8117850.

[278] A. C. Gilbert, P. Indyk, M. Iwen, and L. Schmidt, "Recent developments in the sparse Fourier transform: A compressed Fourier transform for big data", *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 91–100, 2014.

[279] D. Mishra, S. K. Singh, and R. K. Singh, "Wavelet-based deep auto encoder-decoder (wdaed)-based image compression", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1452–1462, 2020.

[280] M. Bosi and R. E. Goldberg, *Introduction to digital audio coding and standards*. Springer Science & Business Media, 2002, vol. 721, ISBN: 978-1-4020-7357-1. DOI: 10.1007/978-1-4615-0327-9.

[281] K. Marlapalli, R. S. B. P. Bandlamudi, R. Busi, V. Pranav, and B. Madhavrao, "A Review on Image Compression Techniques", in *Communication Software and Networks*, S. C. Satapathy, V. Bhateja, M. Ramakrishna Murty, N. Gia Nhu, and J. Kotti, Eds., Singapore: Springer Singapore, 2021, pp. 271–279, ISBN: 978-981-15-5397-4.

[282] I. Nassra and J. V. Capella, "Data compression techniques in IoT-enabled wireless body sensor networks: A systematic literature review and research trends for QoS improvement", *Internet of Things*, vol. 23, p. 100 806, 2023, ISSN: 2542-6605. DOI: 10.1016/j.iot.2023.100806. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660523001294.

[283] K. See, G. Wang, Y. Zhang, *et al.*, "Critical review and functional safety of a battery management system for large-scale lithium-ion battery pack technologies", *International Journal of Coal Science & Technology*, vol. 9, no. 1, 2022.

[284] Y. Zheng, M. Ouyang, X. Li, *et al.*, "Recording frequency optimization for massive battery data storage in battery management systems", *Applied Energy*, vol. 183, pp. 380–389, 2016.

[285] L. Zhou, L. He, Y. Zheng, X. Lai, M. Ouyang, and L. Lu, "Massive battery pack data compression and reconstruction using a frequency division model in battery management systems", *Journal of Energy Storage*, vol. 28, 2020.

[286] X. Tang, F. Gao, and X. Lai, "Compressing and reconstructing the voltage data for lithium-ion batteries using model migration and un-equidistant sampling techniques", *eTransportation*, vol. 13, p. 100 186, 2022, ISSN: 2590-1168. DOI: 10.1016/j.etran.2022.100186.

[287] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[288] J. Groß, *Linear regression*. Springer Science & Business Media, 2003, vol. 175.

[289] F. Acito, "Ordinary Least Squares Regression", in *Predictive Analytics with KNIME: Analytics for Citizen Data Scientists*. Cham: Springer Nature Switzerland, 2023, pp. 105–124, ISBN: 978-3-031-45630-5. DOI: 10.1007/978-3-031-45630-5_6.

[290] T. P. Ryan, *Modern regression methods*. John Wiley & Sons, 2008, vol. 655, pp. 18–19.

[291] D. Karimi, H. Behi, J. Van Mierlo, and M. Berecibar, "Equivalent Circuit Model for High-Power Lithium-Ion Batteries under High Current Rates, Wide Temperature Range, and Various State of Charges", *Batteries*, vol. 9, no. 2, 2023, ISSN: 2313-0105. DOI: 10.3390/batteries9020101. [Online]. Available: https://www.mdpi.com/2313-0105/9/2/101.

[292] D. Jöst, A. Blömeke, D. U. Sauer, and F. Ringbeck, "Timeseries data of a drive cycle aging test of 28 high energy NCA/C+ Si round cells of type 18650", Institut für Stromrichtertechnik und Elektrische Antriebe, Tech. Rep., 2021. DOI: 10.18154/RWTH-2021-02814.

[293] I. M. Monirul, L. Qiu, and R. Ruby, "Accurate SOC estimation of ternary lithium-ion batteries by HPPC test-based extended Kalman filter", *Journal of Energy Storage*, vol. 92, p. 112 304, 2024, ISSN: 2352-152X. DOI: 10.1016/j.est.2024.112304. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X24018905.

[294] R. Thomas, F. Lehmann, J. Blatter, G. Despesse, and V. Heiries, "Performance Analysis of a Novel High Frequency Self-Reconfigurable Battery", *World Electric Vehicle Journal*, vol. 12, no. 1, 2021, ISSN: 2032-6653. DOI: 10.3390/wevj12010010. [Online]. Available: https://www.mdpi.com/2032-6653/12/1/10.

[295] S. Ma, M. Jiang, P. Tao, *et al.*, "Temperature effect and thermal impact in lithium-ion batteries: A review", *Progress in Natural Science: Materials International*, vol. 28, no. 6, pp. 653–666, 2018, ISSN: 1002-0071. DOI: 10.1016/j.pnsc.2018.11.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1002007118307536.

[296] "Standards History of Cellular Systems Toward 6G", in *6G Key Technologies*. John Wiley & Sons, Ltd, 2022, ch. 1, pp. 1–41, ISBN: 9781119847502. DOI: 10.1002/9781119847502.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119847502.ch1. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119847502.ch1.

[297] M. Ghoshal, Z. J. Kong, Q. Xu, *et al.*, "An in-depth study of uplink performance of 5G mmWave networks", in *Proceedings of the ACM SIGCOMM Workshop on 5G and Beyond Network Measurements, Modeling, and Use Cases*, ser. 5G-MeMU '22, Amsterdam, Netherlands: Association for Computing Machinery, 2022, pp. 29–35, ISBN: 9781450393935. DOI: 10.1145/3538394.3546042.

[298] R. A. Fezeu, J. Carpenter, C. Fiandrino, *et al.*, "Mid-Band 5G: A Measurement Study in Europe and US", *arXiv preprint arXiv:2310.11000*, 2023.

[299] M. I. Rochman, W. Ye, Z.-L. Zhang, and M. Ghosh, "A Comprehensive Real-World Evaluation of 5G Improvements over 4G in Low- and Mid-Bands", in *2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2024, pp. 257–266. DOI: 10.1109/DySPAN60163.2024.10632844.

[300] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, *et al.*, "The FAIR Guiding Principles for scientific data management and stewardship", en, *Scientific Data*, vol. 3, no. 1, p. 160 018, Mar. 2016, ISSN: 2052-4463. DOI: 10.1038/sdata.2016.18. [Online]. Available: https://www.nature.com/articles/sdata201618 (visited on 02/02/2023).

Acronyms

3GPP	3rd Generation Partnership Project	108
AC	Alternating Current	9
AI	Artificial Intelligence	9
AQL	Amortized Q-learning	33
AWS	Amazon Web Services	53
BM3	Battery Modular Multilevel Management	15
BMS	Battery Management Systems	2
CAN	Controller Area Network	50
CC	Constant Current	81
CR	Compression Ratio	89
CSV	Comma-Separated Values	60
CV	Constant Voltage	81
DC	Direct Current	9
DL	Deep Learning	24
DQN	Deep Q-Network	33
DUT	Device Under Test	49
EC2	Elastic Cloud Computing	73
ECM	Equivalent Circuit Model	10, 11
EIS	Electrochemical Impedance Spectroscopy	77
EKF	Extended Kalman Filter	72
EVs	Electric Vehicles	1
FNN	Feedforward Neural Network	23
GPU	Graphical Processing Unit	38
GRU	Gated Recurrent Unit	23
HPPC	Hybrid Pulse Power Characterization	95
IIoT	Industrial Internet of Things	70
IMT	International Mobile Telecommunications	109
IOM	Integer-Order Models	11
IoT	Internet of Things	1
ITU	International Telecommunication Union	109
JSON	JavaScript Object Notation	61
KAN	Kolmogorov–Arnold Networks	23

Li-Ion	Lithium-Ion	10
LSTM	Long Short-Term Memory	23
LTE	Long Term Evolution	73
MAE	Mean Absolute Error	30
MDP	Markov Decision Process	25
ML	Machine Learning	23
MLI	Multi-level Inverter	13
MLP	Multi-Layer Perceptron	27
MMC	Modular Multilevel Converter	33
MMI	Modular Multilevel Inverter	66
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor	17
MQTT	Message Queuing Telemetry Transport	108
PWM	Pulse Width Modulation	13
RMSE	Root Mean Square Error	30
RNN	Recurrent Neuronal Network	29
RoC	Rate of Compression	89
SOC	State-of-Charge	11
SOH	State-of-Health	12
TCP	Transmission Control Protocol	108
UDDS	Urban Dynamometer Driving Schedule	73

Nomenclature

Batterie Parameter

Δt	Delta in time
$\tau_{1/2/3}$	Time constance $\tau = RC$
C_C	Heat capacity of the internal medium
C_S	Heat capacity of the battery shell and the external medium
$C_{1/2/3}$	Capacity of ith RC-network
G	Electrical conductance
i_B	Battery current
i_{AC}	Current within the reconfigurable battery
Q_c	Current Capacity of the cell
Q_n	Nominal Capacity of the cell
Q_{re}	Heat generation rate
R_0	Inner resistance
R_i	Thermal resistance between the battery core and the surface
R_o	Thermal resistance between the battery surface and the environmen
$R_{1/2/3}$	Resistance of ith RC-network
SOC	State of Charge
SOH	State of Health
T	Charge/Discharge time of battery
t	Time
T_S	Surface Temperature
T_{amb}	Ambient Temperature
T_{core}	Core Temperature
u_B	Battery voltage
u_C	Capacitor Voltage
V_{OC}	Open curcuit voltage

CR	Compresion Rate
RoC	Rate of Compression

Machine Learning

α	Learning Rate
ϵ	Exploration factor for ϵ-greedy policy
γ	Discounting factor
$\hat{y}$	Predicted output value
θ	Network parameters
A	Action space
a	Action
A_{n_V}	Action space at voltage level n_V
$M^{(k)}$	Switch state for module k
n_V	Voltage level
Q	Q-Function executed
r_t	Reward at time step t
s_t	State at time step t
x	Input vector
y	Expected output value

List of Figures

1.1 Thesis and publication roadmap 4

2.1 Schematic of a conventional battery electric vehicle 10
2.2 Schematic Illustration of a Li-Ion Battery 11
2.3 ECM IOM used to describe Lithium-ion batteries [50] 12
2.4 Temperatures of a battery and electro-thermal model (ETM) 14
2.5 Three-Phase, six switches, two-level inverter topology 14
2.6 Reconfigurable batteries micro topologies 15
2.7 DC power train reconfigurable battery 16
2.8 Three-phase AC reconfigurable battery 16
2.9 Half-Bridge and BM3 module 17
2.10 Soft- and Hardware components of a modern BMS [44] 18
2.11 Schematic Illustration of CHAIN framework for BMS in the Cloud 19
2.12 Computing Paradigms 21
2.13 Service models 22
2.14 Difference between deployment models private, public and hybrid cloud . . 23
2.15 Relationship between AI and subtypes 24
2.16 Learning paradigms 25
2.17 Agent-Environment Interaction in Reinforcement Learning 26
2.18 FNN architecture 27
2.19 Structure of a Kolmogorov–Arnold Networks (KAN) 28
2.20 Architecture of LSTM and GRU 29
2.21 Different types of integration: Digital Model, Digital Shadow, and Digital Twin 31

3.1 Single-phase of three interconnected BM3 modules 35
3.2 Simulation of System with eight BM3 modules 37
3.3 Computational cost per implementation by number of simulated modules . 39
3.4 Q-Table for reconfigurable battery control 40
3.5 Q-Learning state example 41
3.6 Multi-Q-Table for reconfigurable battery control 42
3.7 Q-Learning balancing with six modules 44
3.8 Amortized Q-learning (AQL) neural network 46
3.9 Reward over the training of AQL 48
3.10 AQL balancing with 6 modules 49
3.11 Comparision balance time required for BM3 using proactive balancing and proposed AQL method 49
3.12 Experimental setup for AQL with Half-Bridge topology 50
3.13 Comparision balance time required for Half-Bridge using switching-max balancing and proposed AQL method 51
3.14 AQL and switching-max balancing with 12 modules 51

4.1 Three-layer IoT Framework and AWS IoT Fleetwise Connected Vehicle platform 54
4.2 Proposed monitoring framework employing AWS services 55
4.3 Five layer IoT Framework 57

4.4 Comparison of data streaming methods CSV and JSON 61
4.5 Levels of components 62
4.6 Reconfigurable batteries macro topologies 65
4.7 Configuration of a half-bridge reconfigurable battery over three timestamps 67
4.8 Experimental setup of reconfigurable battery to evaluate proposed universal data specification 68
4.9 Required bandwidth under several configurations and data formats 69
4.10 Difference between cloud and fog computing 70
4.11 The four fog computing architectures 71
4.12 Cloud Computing architecture to estimate the SOC 72
4.13 Request time depending on the model and distance 74
4.14 Request time over during discharge using LTE as mobile network 75
4.15 SOC estimation of EKF and FNN compared to capacity-based SOC 76
4.16 Computed tomography (CT) scans of Molicel P42A (a) and Samsung 40T (b) 80
4.17 Battery cell preparing 80
4.18 Battery cycling setup with cycler and climate chamber 81
4.19 Measurement data of surface and core temperature, voltage, and current . . 82
4.20 Structure for estimating surface and core temperature using LSTM network 83
4.21 Measured and estimated surface temperature at 10 °C 85
4.22 Measured and estimated core temperature at 10 °C 86
4.23 Measured and estimated core and surface temperature at UDDS 25 °C . . . 88
4.24 Comparison of lossless and lossy compression 89
4.25 Implemented stages of proposed voltage compression method 91
4.26 Variations of Regression 92
4.27 Reconstruction of voltage values after compression 94
4.28 Reconstruction of pulsed data 95
4.29 MAE and RMSE depending on RoC over LA92 drive cycle by set RoC . . . 97
4.30 MAE at different set RoC and drive cycle at 25 °C and 0 °C ambient temperature 98
4.31 Compression performance analysis on STM32L432 microcontroller 99

5.1 Cloud-based BMS and digital twin data flow 101
5.2 Cloud-enhanced BMS architecture for SoC estimation and control 102
5.3 Implementation of centralized cloud computing digital twin 103
5.4 Experimental setup with a 12-cell hybrid cascaded multilevel converter . . . 104
5.5 Interactive visualization of real-time measurements, state estimation, and control commands 105
5.6 Request time over experiment time of one minute dependent 106
5.7 Request time over time with switching-max control and AQL 107
5.8 Bandwidth over Time in kb/s 109

Listings

4.1 Metadata of a half-bridge submodule [26] 66
4.2 Example of time series data in the JSON format [26] 66
4.3 Example of time series data in the CSV format [26] 66
4.4 Example of cell configuration data in the JSON format as seen in Fig. 4.7 . 67

5.1 Transfered JSON data of one data point of a battery pack with 12 voltage and one current measurement 107
5.2 Transfered JSON data of one data point of reconfigurable battery with 12 voltage and current measurements 108

List of Algorithms

1 The proposed Q-learning Algorithm [20] ©IEEE 2023 43
2 AQL training [21] 47

3 Voltage compression within time window 93

List of Tables

1.1 Research objectives (RO) addressed in each paper 3

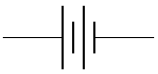
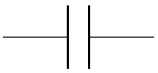
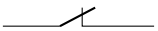
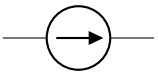
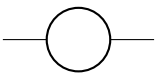
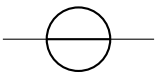
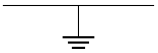
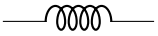
2.1 Principle of operation Half-Bridge, and H-Bridge 17
2.2 Commercial cloud-based BMS solutions 20

3.1 Computational cost per implementation for ten, 200, and 400 modules in seconds 39
3.2 Architecture of the model [21] 48
3.3 Required time to balance SOC within four settings 48

4.1 Metadata datatype description 63
4.1 Metadata datatype description 64
4.2 Required storage per data format for experiment 68
4.3 MAE and RMSE of 45 min (discharge of SOC 100 % − 91.81 %) UDDS driving cycle at 0 °C and a measurement frequency of 10 Hz [27]. 73
4.4 MAE of estimated SOC by FNN, and EKF compared to the capacity-based SOC at different loss rates [27]. 76
4.5 Summary of references for temperature estimation [30] 78
4.6 Molicel P42A information 79
4.7 Architecture of Models 84
4.8 MAE, and RMSE Core and Surface Temperature per Model of drilled cells 84
4.9 MAE Surface and Core Temperature per Model in °C at ambient temperatures; Higher accuracy is highlighted in bold 85
4.10 MAE, and RMSE Surface Temperature in °C per Model and ambient temperature of Validation Cell 86
4.11 MAE, and RMSE Core and Surface Temperature per Model at variations of training in °C 87
4.12 Computational cost comparison of models and different hardware setups . . 88
4.13 MAE, RMSE, and achieved RoC of reconstructed values under pulsed conditions regarding the set RoC calculated by the error between original and reconstructed values 96
4.14 MAE and RMSE, of reconstructed values across aging datasets regarding the set RoC calculated by the error between original and reconstructed values . 96
4.15 Threshold depending MAE and RMSE in mV for drive cycle US06 at 25 °C and set RoC of 99.2 % 98
4.16 Mapping of window size to RoC 99

5.1 Transfered data in kByte over one minute and Rate of Compression 108

List of Electric Circuit Components

Name	Symbol
Battery	
Battery Pack	
Capacitor	
Closed Switch	
Current Source	
Source	
Voltage Source	
Ground	
Inductor	
Open Switch	
Resistor	
MOSFET	