UNIVERSITY
OF OSLO

Master's thesis

Module-scale degradation state assessment of lithium-ion batteries through machine learning and electrochemical impedance spectra

Knut Magnus Aasrud

Computational Science: Physics
60 ECTS study points

Department of Physics
Faculty of Mathematics and Natural Sciences

Spring 2024

Knut Magnus Aasrud

Module-scale degradation state assessment of lithium-ion batteries through machine learning and electrochemical impedance spectra

Supervisors:
Dr. Rodrigo Carvalho
M.Sc. Amund Midtgard Raniseth
Prof. Morten Hjorth-Jensen

Contents

1 Introduction . . . 1
2 Theory . . . 3
2.1 Introduction to lithium-ion batteries . . . 3
2.2 Electrochemical impedance spectroscopy (EIS) . . . 6
2.3 Machine learning . . . 9
2.4 Statistical evaluation . . . 13
3 Method . . . 15
3.1 Data representation . . . 15
3.2 Dataset . . . 15
3.3 Data selection and preprocessing . . . 17
3.4 Network architecture . . . 19
3.5 Training and hyperparameter optimization . . . 20
3.6 Frameworks and implementation . . . 22
4 Results and discussion . . . 23
4.1 Trimming frequencies . . . 23
4.2 Hyperparameter tuning . . . 24
4.3 Training session . . . 25
4.4 Performance evaluation and comparison . . . 27
4.5 Future prospects . . . 30
5 Conclusion . . . 31
A Acronyms . . . 33
B External resources and licenses . . . 35
C Calculations . . . 37
C.1 Time and energy requirements of measuring battery capacity directly . . . 37
References . . . 39

List of Figures

1.1 Open chassis of a Nissan Leaf showing its exposed battery modules. Each of the aliminium cannisters represents a module. The modules contain 2 lithium-ion battery cells in series and 2 in parallel, and they use cathodes made of a combination of $LiMn_2O_4$ and $LiNiO_2$ [1]. The image is captured at the 2009 Tokyo Motor Show by 天然ガス [2]. 1

2.1 Diagram of a typical lithium-ion battery while discharging. The figure is reproduced from Wikimedia Commons by user Sdk16420 [15]. 3
2.2 Diagram showing the external and internal factors to lithium-ion battery degradation, as well as the modes and the direct effect of the degradation phenomena. Figure from Liu, Liu, et al. [19].. 5
2.3 Nyquist plot representing the impedance characteristics of a lithium-ion battery [24]. Z' is the real impedance (resistance) and Z'' is the negative imaginary impedance (negative reactance). The numbered regions are explained below. Reproduced with permission from The Electrochemical Society. 7
2.4 Nyquist plots of NMC 18650 cells at varying degradation levels. The data shown is from the cell-level dataset described in section 3.2.. 8
2.5 Nyquist plot showing the impedance spectra of a lithium nickel cobalt aluminium oxide (NCA) cell. The measurements are done at 60% State of charge (SoC) and varying temperatures [26]. Reproduced with permission from Elsevier. 8
2.6 Impedance spectra of a NCA cell at varying SoC and temperature [26]. Both plots are reproduced with permission from Elsevier.. 9
2.7 Comparison between different activation functions. 12
2.8 Diagram showing the individual folds and selection done in K-fold cross-validation. . . 14

3.1 Nyquist plot illustrating the general impedance behavior of an electric vehicle NMC battery module with 4 cells in series and 3 in parallel. The specific impedance values are omitted due to confidentiality agreements with Evyon AS. 17
3.2 A schematic of the neural network architecture. The network accepts complex-valued input tensors of shape `(26,)`, representing an impedance spectrum measured from a lithium-ion battery. It outputs a real valued tensor of shape `(1,)` with a value between 0 and 1 which represents the SoH of that battery. 19

4.1 MAE, MSE and R^2 of a GPR trained and evaluated with different number of frequencies trimmed from the low or high end of the spectrum, shown in (a) and (b), respectively. The optimal point is calculated as $\operatorname{argmin}_{n\in\mathbb{N}} \frac{\mathrm{MAE}(n)\times\mathrm{MSE}(n)}{R^2(n)}$.. 23
4.2 Evolution of the fitness score of the genetic algorithm over 29 generations.. 24

4.3 Progression of statistical metrics for training and validation metrics during training of the proposed model. Subfigure (a) shows the loss function behavior during model training, while subfigure (b) depicts the MAE, MSE, and R^2 values on the validation set, both across 200 epochs.. 26

4.4 This set of plots collectively demonstrate the predictive capabilities of the neural network and GPR models in assessing SoH at both the cell and module level. Both the GPR and the neural network are here trained on the cell-level training set. Each graph provides a visual correlation between predicted SoH values and true measurements, and an identity line is given in red as a guide. Errorbars are added to the true module-level SoH values, as the C-rate used by the manufacturer when finding the nominal capacity is unknown. These bars thus reflect the range that the true value could be in. A detailed description of this can be found in section 3.2. 29

List of Tables

4.1 The optimal hyperparameters found after performing the GA tuning. 25
4.2 MAE, MSE, and R^2 obtained by predictions performed with GPR across different splits of training and testing sets. The evaluation includes both cell-level and module-level datasets, with the models being trained on one dataset or a combination of them. When using the module data as the testing set, 5-fold cross-validation is employed, in order to utilize all of the data to find the statistical metric in question. 27
4.3 Statistical metrics, including MAE, MSE, and R^2, for the proposed neural network as showed in figure 3.2. The assessment is performed across cell-level and module-level datasets, training the models either separately on each dataset or on a merged dataset. For module data testing, 5-fold cross-validation is once again applied to fully leverage all available data in determining the specific statistical metric. 27

Preface

Writing this thesis has been a joy, largely because I have had help from so many wonderful people. To my main supervisor, Rodrigo Carvalho: Thank you for everything you have taught me and for supporting me ever since we first met. You are a great mentor and an even greater friend. To my co-supervisor, Morten Hjorth-Jensen: Thanks for introducing me to the joys of machine learning and computational methods, and for your guidance through academic bureaucracy. To my co-supervisor, Amund Midtgard Raniseth: Thank you for believing in me from the start and literally changing the course of my career. I am very lucky to have such a good friend. Additionally, a big thanks to all my colleagues at Evyon, who not only have provided the equipment and resources I needed to write this thesis, but who have been incredibly welcoming from the start and provided invaluable support throughout the whole process.

Thank you to my family; I would never have managed this without you. Everything from the deeply technical conversations with my dad, to the patience of my mom listening to them. To Martine, for all the long and consoling conversations; to Maren, for visiting hipster coffee spots with me and having so many fits of laughter together; to my dad, for being a role model and always giving wise advice; and to my mom, for your unconditional love and support.

I would like to extend a final thanks to my girlfriend, Kristine, without whom I would never have gotten through this master's degree. You were there when I was approved for the program and you are here now as I'm delivering my thesis. We have gone through a lot these two years, but your support has been ever steadfast. Thank you.

Twenty years at school,
Thinking it will never end.
Suddenly, it does.

Abstract

To support sustainable practices in the battery industry, the feasability of using a data-driven approach for efficient estimation of the state of health (SoH) of lithium-ion battery modules is explored. A combination of machine learning, particularly convolutional neural networks, with data from electrochemical impedance spectroscopy is used to perform these estimations. By focusing on the entire module instead of just single cells, this study works towards a method that is practically deployable at a larger scale than current methods. This work provides an in-depth examination of battery behavior, degradation, and the nuances of impedance spectroscopy, like the impact of confounding variables on battery impedance. Furthermore, it also delves into the fine-tuning of machine learning models for this specific application. In the study, a neural network is trained on cell-level data to perform state of health estimations on module-level data with an accuracy of approximately 5%. While this is not adequate for industrial application, the results successfully demonstrate module-level state of health estimation based on electrochemical impedance spectroscopy.

Sammendrag

For å oppnå bærekraftig bruk av ressursene som ligger i elektriske kjøretøy, er det essensielt å finne en metode for rask estimering av batterienes helsetilstand. I denne masteroppgaven forsøkes det å bevise at det er mulig å bruke data-drevne metoder for å oppnå dette målet. Dette gjøres ved hjelp av elektrokjemisk impedansspektroskopi og maskinlæringsmetoder. Studien tar for seg målinger gjort på modulnivå, i motsetning til tidligere forskning som har fokusert på enkeltceller. Med denne oppskaleringen er målet å oppnå en metode som kan anvendes praktisk på en større skala, eller i en operasjonell kontekst. Oppgaven dykker ned i batteriers grunnleggende oppførsel og nedbrytningsprosesser, samt hvordan impedansspektroskopi kan identifisere disse prosessene. Den tar også for seg prinsippene bak maskinlæring og hvordan en modell kan justeres for å oppnå optimal ytelse til formålet. Et nevralt nettverk trenes på data fra enkeltceller, for deretter å utføre estimater på et sett med impedansspektra på modulnivå. Nøyaktigheten i estimatene ender på 5%. Selv om dette ikke er godt nok til industriell bruk, viser resultatene at trening av en modell på denne skalaen er mulig.

Chapter 1

Introduction

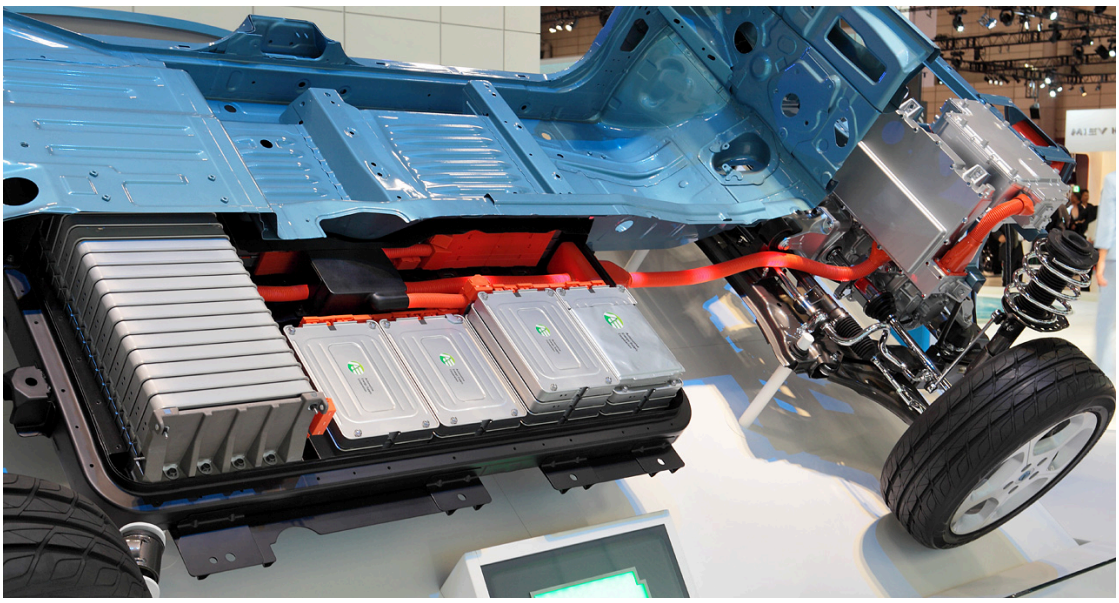


Figure 1.1: Open chassis of a Nissan Leaf showing its exposed battery modules. Each of the aliminium cannisters represents a module. The modules contain 2 lithium-ion battery cells in series and 2 in parallel, and they use cathodes made of a combination of $LiMn_2O_4$ and $LiNiO_2$ [1]. The image is captured at the 2009 Tokyo Motor Show by 天然ガス [2].

The beginning of the 21st century has seen a significant shift in the global automotive industry. Electric vehicles (EVs) are presented as one of many solutions in the fight against climate change and global warming. and demand for them is growing exponentially [3]. This shift is driven by the urgent need to reduce reliance on fossil fuels and mitigate the transport sector's impact on carbon emissions. As highlighted by Kastanaki & Giannis [4], "many EU countries have set mandatory targets to eventually end all sales of internal combustion engine vehicles by 2030 or 2040". However, this rapid growth is accompanied by an environmental challenge: the disposal and recycling of batteries.

Batteries are composed of rare and expensive materials. The extraction of e.g. cobalt often involves unethical mining practices [5], while nickel production is a highly energy-intensive process that generates large amounts of sulfur dioxide during refinement [6]. This highlights a clear contradiction in the green narrative of electric mobility. Moreover, the chemical and structural complexity of a lithium-ion battery makes them difficult to recycle [7], presenting a significant barrier in the switch to a sustainable future. Given the strict safety regulations and the requirement for optimal performance in electric vehicles, batteries are usu-

ally considered unfit for automotive application when their capacity reaches 70% to 80% of their original rating [8]. This leads to a considerable volume of battery waste, projected to reach between 880 million and 8 billion end of life (EOL) cells generated in the year 2040 from EVs alone [9], much of which possesses an untapped potential for second-life applications. However, the realization of this potential depends on a comprehensive understanding of a battery's state of health (SoH) throughout its lifecycle — a metric that is notoriously difficult to measure accurately without causing additional damage to the battery.

Traditional approaches to assessing the SoH of batteries are time and resource-demanding and may accelerate battery degradation, thus conflicting with the aim of conserving resources. In this context, electrochemical impedance spectroscopy (EIS) has emerged as a non-invasive method for evaluating batteries, offering crucial insights into their electrochemical characteristics and, consequently, the health of battery cells [10]. Despite its promise, EIS remains largely a laboratory tool due to its reliance on costly equipment [11].

The increased availability of computing power and data storage capabilities in recent years has paved the way for a surge in data-driven methodologies and machine learning as solutions to complex problems. Some notable achievements include achieving superhuman performance in the game of go [12], developing sophisticated language models capable of understanding and generating human-like text [13], and significant strides in performing quantum many-body simulations [14]. In this vein, deep learning — leveraging vast datasets to iteratively improve performance — offers a fresh and automated approach to tackling intricate challenges. This shift towards harnessing computational power and big data is introducing new solutions to longstanding problems.

Indeed, data-driven methods have become an increasingly popular way of performing SoH estimation based on EIS[1] [11]. Techniques like gaussian process regresion and neural networks have reached an impressive level of accuracy, many at around 1% mean absolute error [11]. Despite this, the application of EIS has been mostly focused on individual cells. This is not only because data collection requires expensive high voltage equipments (see Appendix C.1), but also because the examination of data obtained at the scale of a battery module remains constrained by the complexities of having multiple battery cells — with different SoH — connected in series and/or parallel [11].

This thesis presents the potential of advanced machine learning methods for interpreting EIS data at a module level, aiming to demonstrate the feasibility of developing a data-driven model for estimating battery SoH, even on a larger scale. Specifically, the aim is to achieve this primarily based on existing cell-level data, due to the current lack of openly available module-level datasets. Coupled with the continuous development of EIS measurement techniques, such a model would make the implementation of second-life battery applications more efficient and resource-conserving. This research aims to close the gap between the theoretical potential of module-based EIS and its actual application, and in that contribute to the sustainable battery management practices that create a circular economy and contribute to the global effort to combat climate change.

[1]This is further evidenced by a search on the Google Scholar platform for `"eis battery soh data driven"`, which returns 6700 results.

Chapter 2

Theory

2.1 Introduction to lithium-ion batteries

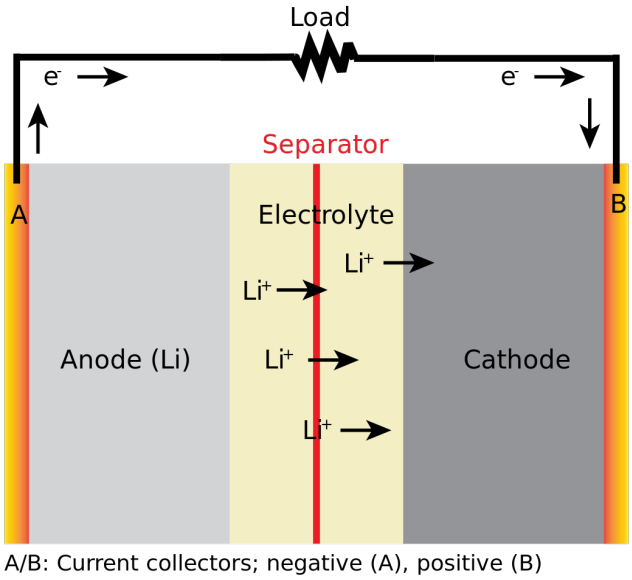


Figure 2.1: Diagram of a typical lithium-ion battery while discharging. The figure is reproduced from Wikimedia Commons by user Sdk16420 [15].

The fundamental operation of lithium-ion batteries (LIBs) is based on the transfer of electrons and lithium ions between an anode (negative electrode) and a cathode (positive electrode). When the battery is charging, ions move from the cathode to the anode within the battery cell, while electrons flow through an external circuit. At the anode, the ions are stored in the lattice of the anode material. During discharge, the ions and electrons then travel back to the cathode, releasing energy to power a load. This reversible flow of electrons and ions is what enables the battery to be recharged multiple times. A simplified schematic of a LIB is depicted in the diagram shown in figure 2.1, and the continuous chemical reactions at the electrodes can be summarized as follows:

$$\begin{aligned} \text{Cathode:} \quad & \mathrm{Li}\mathit{Cat} \leftrightarrow \mathrm{Li}^{+} + e^{-} + \mathit{Cat}, \\ \text{Anode:} \quad & \mathrm{Li}^{+} + e^{-} + \mathit{An} \leftrightarrow \mathrm{Li}\mathit{An}. \end{aligned}$$

Here, Cat is the host cathode material, which can be made from a variety of lithium oxides. The anode An typically consists of graphite. While cathode materials can be found in many different chemical structures, layered structures are the most commercially evolved [16], and choice of material drastically influences the properties of a battery [17]. So much, in fact, that batteries are often categorized based on the materials used for the cathode. A few examples include:

- **$LiCoO_2$ or lithium cobalt oxide (LCO)**: A commonly used cathode material in LIBs, but it is challenged by the high cost of cobalt and general stability issues, particularly when overcharged [17].
- **$LiNi_xMn_yCo_{1-x-y}O_2$ or lithium nickel manganese cobalt oxide (NMC)**: Offers a blend of high capacity, improved stability, and cost-effectiveness. Initially, $LiNiO_2$ was introduced because of its lower cost and higher energy density compared to LCO. However, since it is less stable, manganese and cobalt were incorporated into the material to increase the stability [17].
- **$LiFePO_4$ or lithium iron phosphate (LFP)**: Has an energy-density of 72% compared to typical NMC batteries [16], but is less costly and has a longer lifetime. Highly valued because it removes the need for cobalt altogether and because it has a flat discharge profile with stable two-phase operation [17].

Each chemistry presents a trade-off between energy density, life cycle, safety, and cost, influencing their suitability for different applications. Given its prevalence in the automotive field, this study will specifically focus on NMC cathode materials.

2.1.1 Lithium-ion batteries in electric vehicles

By far the most prevalent batteries used in EVs are LIBs. This is because of the high energy density and long cycle life, as explained above. At the fundamental level, an EV battery is comprised of individual electrochemical cells. These cells are the basic building blocks where energy is stored in the form of chemical potential. To increase the overall energy storage capacity and manageability, cells are grouped into modules. A module is a collection of cells connected in series and/or parallel, and it may include a cell management system (CMS) to monitor and balance the voltages and temperatures of each cell, in order to optimize the efficiency and longevity of the module. Then, at the highest level, multiple modules are aggregated into battery packs, usually located within the chassis of an electric vehicle. The pack includes a battery management system (BMS) responsible for overseeing the health and performance of the entire pack.

2.1.2 Degradation phenomena

With usage and aging, LIB cells experience many different degradation phenomena, leading to a decline in performance over time. Understanding these phenomena is crucial for deploying LIBs into second-life use, especially when it comes to batteries from EVs, as they are often used in unstable environments and pushed to the limits. A general categorization can be performed, grouping internal and external factors into three main *degradation mechanisms*: Loss of lithium-ion inventory (LLI), loss of active material (LAM) and conductivity loss (CL) [18]. While this study will not delve deep into the electrochemical aspects of these mechanisms, a few notable examples of degradation phenomena include:

- **Solid electrolyte interface (SEI) growth**: The SEI layer is an accumulation of lithium-ions in the interface between the electrolyte and electrode. It is usually desireable, since it gives access to free lithium-ions and forms a passivation layer that protects the electrode, but with further growth the available lithium ions are consumed and a thicker layer can cause blockage of the anode pores, degrading the battery by e.g. increasing internal resistance. During fast charging, especially at low temperatures, lithium can plate on the anode's surface, irreversibly forming metallic lithium instead of reacting with the anode. This plating can even evolve into dendritic structures that penetrate the separator, potentially causing short circuits [19].

- **Loss of cathode material**: At the cathode side, the aging is mostly manifested in loss of active material. This commonly occurs due to stress when releasing and accepting lithium, which causes phase changes that can alter the crystal structure or even make it collapse. These effects often arise over time and with increasing number of cycles [19].

In general, battery aging is influenced by both the environment and usage patterns. Factors such as time, temperature, voltage levels, charging speed, and physical stress often interact, compounding degradation through a variety of complex phenomena [19]. Figure 2.2 provides a detailed illustration of this complexity, illustrating how interconnected the factors are.

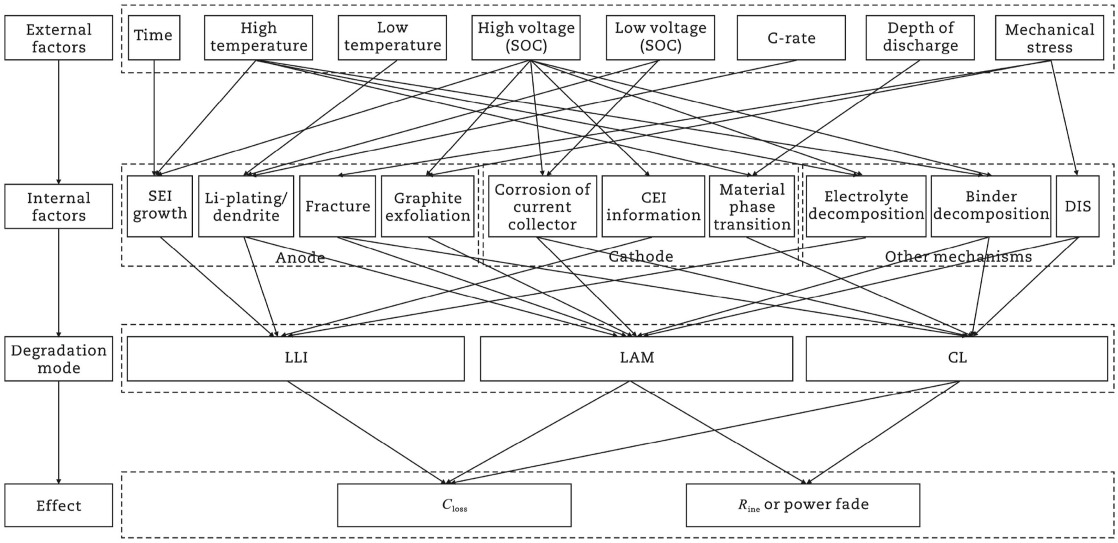


Figure 2.2: Diagram showing the external and internal factors to lithium-ion battery degradation, as well as the modes and the direct effect of the degradation phenomena. Figure from Liu, Liu, et al. [19].

2.1.3 State of health (SoH)

To measure the overall degradation of a LIB over time, a metric like SoH is often used. SoH is defined as the comparison of the battery's current state x_{current} to its state at the time of manufacture (or at least the state described in its specification) x_{initial}. In some cases, the ratio may be adjusted by the state at its EOL x_{EOL}. The physical quantity given by x is not agreed upon in the industry, but it typically refers to one of or a combination of the internal resistance and capacity [20]. The SoH is then given by the following equation.

$$\text{SoH} = \frac{x_{\text{current}} - x_{\text{EOL}}}{x_{\text{initial}} - x_{\text{EOL}}} \tag{2.1}$$

While percentages are commonly used to represent the SoH of batteries, this study uses a scale from 0 to 1 to simplify the integration with machine learning algorithms. 1 represents when the battery is brand new and decreases as the battery gets older [21]. This drop in SoH is due to the chemical and physical changes as described above, which gradually impair the battery's capacity to store and release energy. Depending on the requirements of the measure, one may set x_{EOL} such that a battery still possesses some functional capacity at an SoH of 0 [21]. In this study, however, x_{EOL} is set to 0, as the goal is to reach a holistic assessment of a battery's applicability for second-life use. An SoH of 0 thus means there is no remaining useful capacity in the battery.

Traditionally, the assessment of SoH has been conducted through direct measurements of cell capacity and/or internal resistance [22]. This is done through *cycling* experiments — fully charging and discharging a battery. In this context, the concept of C-rate is often used to indicate the speed at which the battery is

charged or discharged, relative to its total capacity. For example, a C-rate of 1 means the battery charges or discharges in one hour, while a rate of 0.5 C extends this to two hours. As shown by Latifatu et al. [23], the selection of C-rate affects the battery's apparent capacity during discharge tests, with higher C-rates typically resulting in reduced capacity due to greater internal losses. Although straightforward to conduct, cycling has practical limitations in some use-cases — especially in second-life repurposing: The process of fully charging and discharging a battery to measure its capacity can be time-consuming and energy-intensive (see Appendix C.1). Furthermore, it subjects the battery to additional cycles of wear.

Alternatively, post-mortem analysis is also a commonly used family of techniques for understanding battery degradation mechanisms. By extracting the battery cells and dissecting them after their life cycle, researchers can use *x-ray* based techniques, *electron and scanning probe microscopy* or *spectroscopic* techniques to identify failure modes and degradation patterns [22]. However, these methods require destructing the battery, and are thus not applicable in second-life characterization or operational scenarios.

2.2 Electrochemical impedance spectroscopy (EIS)

EIS is the process of measuring the complex impedance of a battery at different frequencies, which provides information about the internal resistance, capacitance, and other properties of the battery. This information can be used to diagnose various types of battery degradation, such as capacity fade, electrode corrosion, and electrolyte depletion [10].

An EIS measurement involves applying a small amplitude alternating current (AC) signal to the battery and analyzing the resulting voltage response. Depending on the approach, this signal can be either a controlled voltage or a controlled current, typically sinusoidal, and is applied across a range of frequencies. In a voltage-controlled (potentiostatic) approach, a known voltage signal is applied to perturb the system, with the current response being recorded. Conversely, during a current-controlled (galvanostatic) experiment, a specified current signal is applied, and the resulting voltage is measured. The impedance Z of the battery is then determined by measuring the ratio of the voltage U to the current I, as seen in equation 2.2.

$$Z = \frac{U}{I} = \frac{U_0 e^{i(\omega t + \phi_U)}}{I_0 e^{i(\omega t + \phi_I)}} = \frac{U_0}{I_0} e^{i\theta}, \qquad \text{where } \theta = \phi_U - \phi_I. \tag{2.2}$$

Here, ω is the signal frequency, ϕ_U and ϕ_I are the voltage and current phases, respectively, and θ is the phase shift between the voltage and the current. Note that, given the relaxation processes in the system being measured θ is dependent on the frequency ω.

EIS as a characterization technique stands out as advantageous for several reasons: Firstly, it is a low-amplitude technique, meaning it does not impose the same level of stress on the battery as methods requiring full charge-discharge cycles or high resistance loads. The frequencies in a typical spectrum are also high when compared to e.g. cycling methods. As a result, EIS can be performed relatively quickly and without significantly affecting the battery's existing SoH. EIS also relies on simple measurements of voltage and current, allowing for easy integration into existing battery management systems by just requiring an AC source. This simplicity, along with its non-invasive nature, makes EIS an attractive option for characterization uses in operational batteries and those being evaluated for second-life uses.

2.2.1 Identifying degradation patterns with impedance spectra

The results of an EIS measurement are often presented in Nyquist and Bode plots. Nyquist plots represent the real and imaginary parts of impedance in a complex-plane graph, allowing for the visualization of different electrochemical processes. Bode plots display the impedance magnitude and phase angle as functions of frequency, aiding in the identification of key frequency-dependent features. Interpretation of impedance spectra involves analyzing the shapes and trends in the Nyquist and Bode plots, and can be

done manually by a human, through physical models and parameter fitting, or — as is the case for this study — by using data-driven machine learning methods.

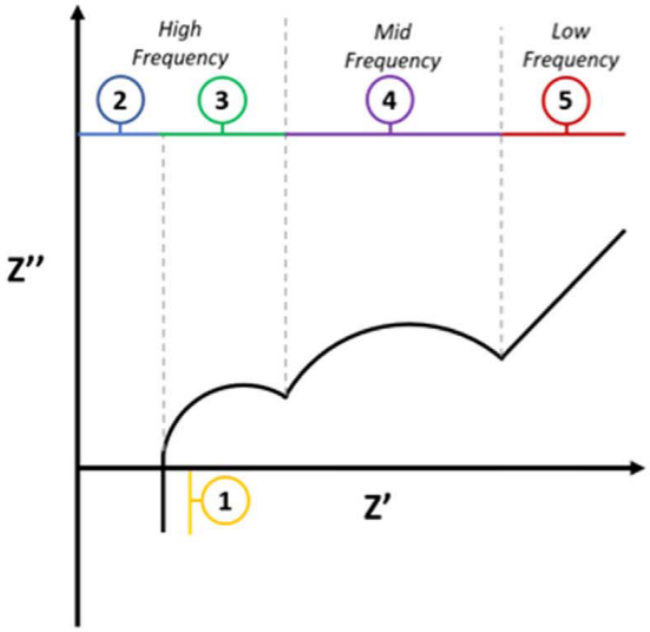


Figure 2.3: Nyquist plot representing the impedance characteristics of a lithium-ion battery [24]. Z' is the real impedance (resistance) and Z'' is the negative imaginary impedance (negative reactance). The numbered regions are explained below. Reproduced with permission from The Electrochemical Society.

Degradation patterns within LIBs are clearly identifiable in their impedance spectra, as summarized in the scientific review by McCarthy et al. [24]. Figure 2.3 shows a mocked up Nyquist plot, where the following spectrum regions are marked in order of decreasing frequency:

1. The high frequency range (kHz) reveals the inductance attributed to the battery's metallic parts, indicative of e.g. the porosity of the electrodes [24].
2. The composite resistance of electrodes, electrolyte, and separator — encountered at the spectra's intersection with the real axis — typically varies with reduced ion mobility due to electrolyte depletion [24].
3. SEI growth can be observed as alterations in the high-frequency semi-circle, characterized by reduced cell capacity and increased resistance. As a result, the semi-circle is often flattened and stretched with increasing degradation [24].
4. Additionally, the mid-frequency semi-circle gives insight about double-layer capacitance and charge-transfer resistance [24].
5. At the spectrum's lower end, the rate of diffusion can be seen. This is usually identified by a region of constant phase, seen as a straight line in the Nyquist plot [24].

In figure 2.4, we can see a clear demonstration of impedance as a degradation indicator.

Equivalent circuit modelling (ECM) is traditionally used to interpret these patterns, by identifying a circuit that mimics the battery's behavior, fitting the circuit parameters to the observations and thereafter simulating the battery's behavior based on the circuit [24]. However, applying ECM to battery modules is

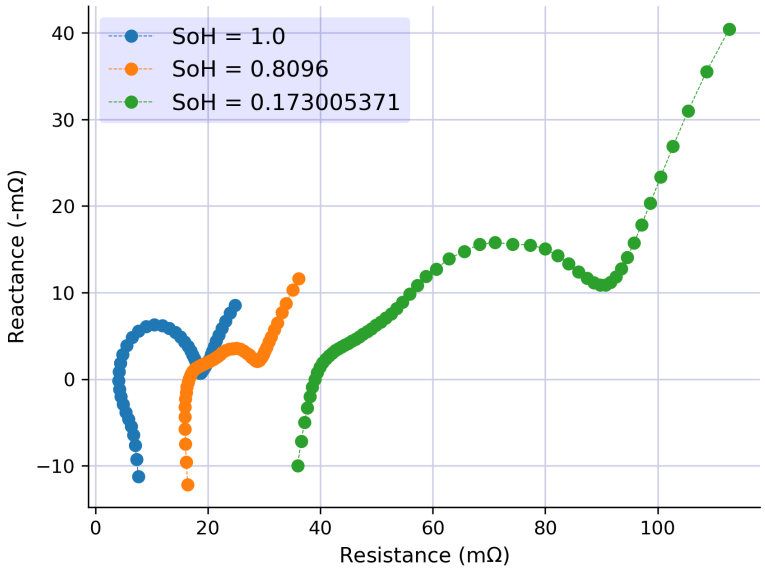


Figure 2.4: Nyquist plots of NMC 18650 cells at varying degradation levels. The data shown is from the cell-level dataset described in section 3.2.

challenging due to the added complexity, stemming from the combined impedance of different cells, the connection points between them and the module circuitry. This makes it hard to represent the system with a reasonable number of circuit elements. Despite this complexity, one or more faulty cells in a multi-cell setup has been shown by Ank et al. [25] to signifciantly influence the overall impedance of the whole system, making EIS a viable option still. By deploying data-driven approaches to estimate the SoH directly from raw impedance data, the ECM method can be circumvented, offering a more flexible and automated method to predict battery degradation.

2.2.2 Free variables in battery impedance

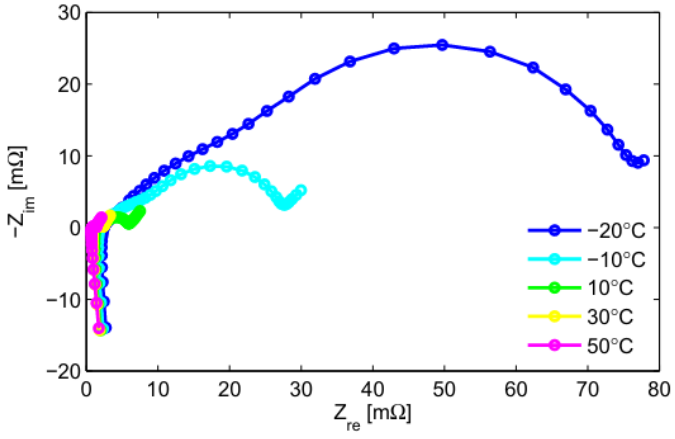
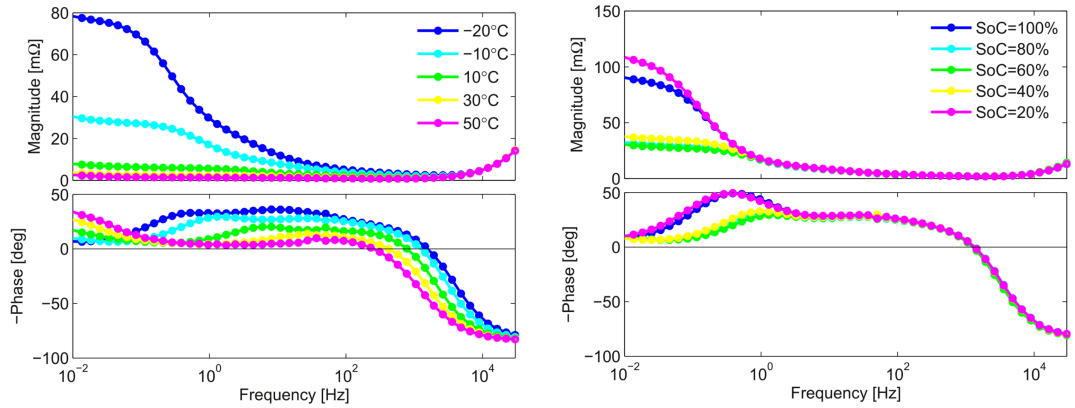


Figure 2.5: Nyquist plot showing the impedance spectra of a lithium nickel cobalt aluminium oxide (NCA) cell. The measurements are done at 60% State of charge (SoC) and varying temperatures [26]. Reproduced with permission from Elsevier.

Temperature has a clear impact on the relaxation time of a dielectric. The thermodynamics of electrode reactions, charge-transfer, and the overall performance of the electrolyte are all affected [27]. As a result,

the impedance of a LIB can be substantially changed at different temperatures, as shown in figure 2.5. Additionally, as indicated by Raijmakers et al. [26], temperature-induced changes in impedance are not confined to specific frequency bands. We can clearly see in figure 2.6a that it rather influences the entire relevant spectrum, affecting both low and high frequencies. In contrast, we see from figure 2.6b that the effect of state of charge (SoC) on impedance may be less impactful than temperature. It is not as broad-reaching and mostly affects the lower region of the spectrum.

While the findings of Raijmakers et al. [26] are from measurements done on NCA cells, studies by Gopalakrishnan et al. [28] confirm the same phenomena in NMC cells as well.



(a) Bode plot of the spectra. The measurements are done at 60% state of charge and different temperatures.

(b) Bode plot of the spectra. The measurements are done at -10 °C and different state of charge.

Figure 2.6: Impedance spectra of a NCA cell at varying SoC and temperature [26]. Both plots are reproduced with permission from Elsevier.

2.3 Machine learning

Machine learning, at its essence, represents the application of statistical methods to teach machines how to learn from data. This allows the creation of algorithms which are capable of identifying both linear and non-linear patterns in data. With advancements in computing technology, these techniques have become applicable on a much larger scale, opening up new possibilities for predictive analysis and intelligent decision-making across various fields.

In supervised machine learning, we train a model $\hat{f}$ to learn the relationship between inputs $\mathbf{X}$ and outputs $\mathbf{y}$. The data is usually divided into two distinct subsets:

- *Training set*: Initially, the model is trained by adjusting its internal parameters to fit the training set.
- *Testing set*: After training, the testing set serves as the benchmark of the model's performance, showing how well it generalizes to unseen data.

Additionally, a third division — the *validation set* — can be introduced to tune the model's external parameters (parameters which are not adjusted by the training session,) as explained in section 2.3.3. The validation set acts as a mediator, ensuring that the adjustments made to the model or training process not only improve its performance on the training data but also enhance its ability to generalize well to new data, avoiding overfitting.

This section will describe two different machine learning approaches to analysing EIS spectra: Gaussian process regression (GPR) and neural networks.

2.3.1 Gaussian process regression (GPR)

GPR is a flexible, Bayesian-based regression method that has been effectively used for estimating battery capacity through EIS data [29]. In GPR, predictions are generated under the assumption that there is a Gaussian distribution across the space of possible functions. This approach enables the inclusion of prediction uncertainties, making GPR highly suitable for EIS data, which in many cases is fairly noisy [30]. An existing implementation of GPR from the Python package `scikit-learn` [31] will be used in this study and only as a comparison point. Thus, the reader is directed to other sources for more details about GPR, e.g. the introduction by MacKay [32].

2.3.2 Neural networks

Neural networks are among the more foundational components of machine learning[1], designed to recognize patterns and solve complex problems. These networks mimic the human brain's structure and function, learning from vast amounts of data to make decisions and predictions. They can be as complex as desired and consist of multiple nested networks of different types.

During the *training process*, a network is tuned to optimize a so-called *loss function* as evaluated on the training set. The network learns during this process by feeding data from the training set through the network to make a prediction. Based on the loss of this prediction, *backpropagation* is utilized, which involves calculating the gradient of the loss function with respect to the network's parameters by using the chain rule. The parameters are then adjusted in the direction that reduces the loss, and the next datapoint is considered. This iterative process continues until the network achieves satisfactory performance or the process is halted. The training process may also include a step called *dropout*, which involves randomly nullifying some of the tensor values by some probability (called the dropout rate.) This is done to avoid overfitting and too much reliance on certain features of the input data.

2.3.2.1 Artificial neural networks

Artificial neural networks, also known as fully-connected or dense networks, are the simplest form of neural networks, where each neuron in one layer is connected to each neuron in the next layer. The primary operation in these networks is a linear transformation of the input data, followed by a non-linear activation function. This activation function is what enables the network to identify non-linear patterns in the data. Each layer n can be described as in equation 2.3.

$$\mathbf{y}^n = f^n\left(\mathbf{W}^n\mathbf{x}^{n-1} + \mathbf{b}^n\right). \tag{2.3}$$

Here, $\mathbf{x}^{n-1}$ is the input to the layer and $\mathbf{y}^n$ the output. $\mathbf{W}^n$ contains what are called the *weights* of the layer and $\mathbf{b}^n$ holds the *bias* of the layer, while f^n represents the activation function. The weights and biases are determined by identifying which values minimize the difference between the predicted values and the actual values. Since the core of each layer in the network is a linear transformation, the layers in an artificial neural network are often referred to as *linear layers*.

2.3.2.2 Convolutional neural networks

An impedance spectrum can be considered a complex-valued 1-dimensional signal in the frequency domain. As described in section 2.2.1, this signal exhibits characteristic shapes and patterns that are key to extracting the state of health of the system. For such a task, 1-dimensional convolutional neural networks (CNNs) have proven to be an especially effective tool, even with a limited dataset [33].

[1]Searching for the term `"neural networks"` on the Google Scholar platform gives 4 380 000 results.

Mathematically, a 1-dimensional convolutional layer computes the running average of a signal over a window of a given size — called the kernel. For the layer n, the input consists of C channels, which are the innermost dimension of the tensor. For each datapoint i and channel c, the convolution output $y_i^{c,n}$ is obtained by convolving the input signal $x_i^{c,n-1}$ with the K number of corresponding kernel weights $\mathbf{k}^{c,n}$. This process is repeated for each position i in the input signal, and the resulting output for each channel is added element-wise and fed through the activation function f^n to produce the final output of the layer. This process can be expressed as:

$$y_i^{c,n} = \mathbf{k}^{c,n} * x_i^{c,n-1} = \sum_{j=1}^{K} k_j^{c,n} x_{i+j-1}^{c,n-1},$$
$$\text{and} \quad y_i^n = f^n\left(\sum_{c=1}^{C} y_i^{c,n}\right).$$

2.3.2.3 Activation functions

Section 2.3.2.1 and section 2.3.2.2 shows artificial and convolutional networks, where the network's output is in essence a complex linear combination of its input. In addition, both apply a non-linear activation function to this output (except, possibly, in the case of the last layer.) This is a crucial addition, which allows the network to learn non-linear relationships between the inputs and outputs. As such, the choice of activation function plays an important role in determining the network's performance.

The Mish activation function is a non-linear activation function that has gained attention for its potential to improve the performance of many deep neural networks. Introduced by Misra [34] as an alternative approach to traditional activation functions like rectified linear unit (ReLU) and sigmoid, it is defined as shown in equation 2.4.

$$\text{Mish}(x) = x \cdot \tanh(\log(1 + \exp(x))) \tag{2.4}$$

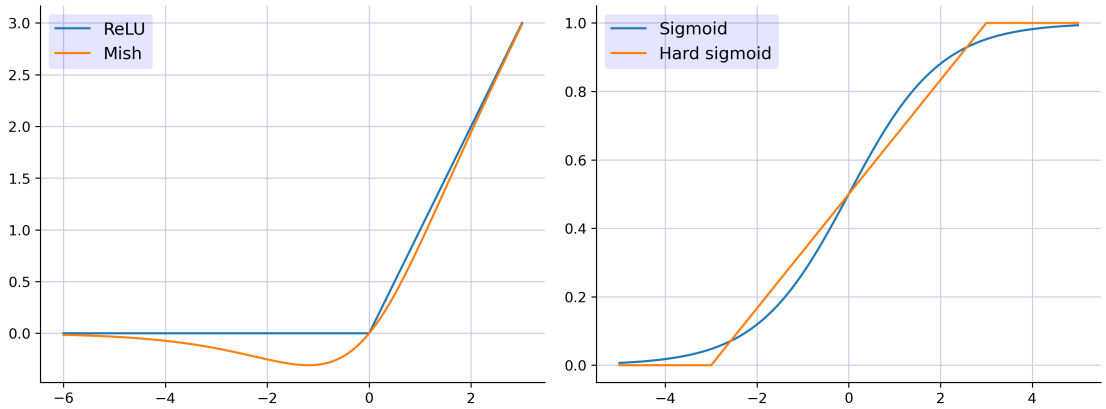
A comparison between Mish and the ReLU activation function is shown in figure 2.7a. Compared to ReLU — and many other activation functions — Mish offers improved accuracy and a smoother loss landscape [34]. It has been observed that it enhances the representation power of neural networks[2], leading to superior performance in tasks such as image classification and object detection [34], which is especially relevant for this study. Hu et al. [36] additionally proves that using Mish in conjuction with 1-dimensional CNNs can promote model performance slightly, compared to ReLU.

The value of SoH is expected to be constrained between 0 and 1. In such cases, applying the sigmoid function to the output layer is a common approach due to its asymptotic approach to these limits. However, this function only asymptotically approaches 0 and 1, which might not be ideal for cases where a clear distinction at these boundaries is necessary. In many datasets, an SoH of 1 is prevalent and must be distinctly recognized. A value of 0 is not as common, but may still be of significance, depending on the definition of SoH. To address these challenges, the hard sigmoid function is introduced as a linear approximation of the sigmoid, that is more performant, but which also ensures the output explicitly reaches 0 and 1 within finite bounds. It is defined by PyTorch Contributors [37] as a piecewise linear function:

$$\text{Hardsigmoid}(x) = \begin{cases} 0 & \text{if} \quad x \leq -3, \\ 1 & \text{if} \quad x \geq 3, \\ \frac{x}{6} + \frac{1}{2} & \text{otherwise.} \end{cases}$$

[2] In the case of recurrent neural networks, it can also help mitigate something called the *vanishing gradient problem*. This is not relevant for this study, but for the curious, a great description of the problem is given by Hochreiter [35].

A comparison between it and the regular sigmoid function is shown in figure 2.7b. As seen there, the function effectively truncates the sigmoid curve and replaces it with a linear mapping from 0 to 1 in the range $[-3, 3]$.



(a) A plot showing the difference between Mish and ReLu around the origin.

(b) A plot showing the difference between the hard sigmoid and sigmoid function around the origin.

Figure 2.7: Comparison between different activation functions.

2.3.3 Hyperparameters

The performance of a machine learning model is significantly influenced by its hyperparameters. Unlike parameters that are learned from the data, hyperparameters denote the set of predefined parameters which cannot be estimated directly from the data. They are used to control the learning process and network structure, and have a significant impact on the performance of machine learning models. Examples include learning rate, the number of hidden layers and units in neural networks, or dropout rates. There are many ways to find the best hyperparameters that make a model perform better. Key methods include grid search, random selection, and genetic algorithms, which are explained below.

2.3.3.1 Grid search

Grid search is a traditional method for hyperparameter optimization that involves exhaustively searching through a manually specified subset of the hyperparameter space. A grid search algorithm searches exhaustively through every combination of hyperparameters. As such, the method can be very time-consuming and computationally expensive — especially as the number of hyperparameters grows [38]. For this reason, it is useful in the cases where the hyperparameter space is small, since we are guaranteed to find the best performing hyperparameter set. For bigger hyperparameter spaces, grid search is too computationally expensive to be a viable method [39].

2.3.3.2 Random selection

As an alternative, random selection is a technique where hyperparameter values are randomly chosen from a defined range. This method contrasts with the exhaustive nature of grid search by only sampling a specified subset of the hyperparameter space. As such, it is often more efficient than grid search, especially when dealing with a large number of hyperparameters, but it still lacks any directed search capabilities and fails to guarantee hitting the best results [39].

2.3.3.3 Genetic algorithm

As described above, grid search and random selection are regarded as naïve and generally inefficient approaches which leads to increased computational complexity and cost. In contrast, genetic algorithms (GAs) offer a more efficient alternative, which provides quick convergence for many hyperparameter tuning cases [38]. They work by simulating the process of natural selection to solve optimization and search problems. The theoretical foundation of GAs is grounded in the principles of genetics and natural selection, making them robust and adaptable to a wide range of problems [40].

To perform hyperparameter tuning using GAs, a set of hyperparameters is generated randomly, constituting a *population*. Binary encodings of each of the hyperparameters are then created — aptly named *chromosomes* [40]. These chromosomes undergo a *fitness* evaluation, which assesses their suitability as solutions based on predefined criteria [40]. In the case of hyperparameters, that may be some statistical metric on the validation set, or similar. For each generation, selection in the population is guided by the fitness score, ensuring that more successful solutions have a greater likelihood of being propagated to the next generation. The selected chromosomes are then combined via *crossover* and *mutation* to produce offspring in order to complete the next generation [40]. Creating offspring from the most fit specimen means the desired traits are kept, while mutation introduces random changes, which is an effective strategy to avoid stagnating evolution [40]. In the language of statistics, this can be compared to avoiding local minima.

The GA iterates through these steps, generating new populations with each iteration. During this cycle, the fittest solutions are retained along with the offspring in the next generation. The cycle is repeated until a termination condition is met [40].

2.4 Statistical evaluation

To evaluate the performance of the proposed model, a set of statistical metrics are introduced. The set chosen for this study are mean absolute error (MAE), mean squared error (MSE), and the coefficient of determination (R^2). To describe them, consider the set of n true values $\mathbf{y}$ and the corresponding predictions $\hat{\mathbf{y}}$ predicted by our model $\hat{f}$.

MAE is a measure of the average magnitude of the errors between predicted and actual observations, without considering their direction. It gives a straightforward indication of the average error magnitude. It is mathematically defined as:

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

MSE, on the other hand, assesses the average of the squares of the errors. This metric penalizes larger errors more than smaller ones, making it particularly sensitive to outliers in the data set, which can give useful insights. It is expressed as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

Lastly, the coefficient of determination or R^2 provides a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model. It can intuitively be understood as how much better the model is, compared to the mean, with respect to the MSE score. R^2 can be computed as such:

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

where $\bar{y}_i$ is the mean of all the true values. It is important to note that R^2 can yield negative values in cases where the mean of the data provides a better fit to the outcomes than the model. This is usually an indication that the model does not follow the trend of the data, thus explaining less variability than the mean of the dataset itself. Incorporating these metrics into the evaluation framework of computational models allows for a comprehensive assessment of their predictive performance. By analyzing the MAE and MSE, the general error and impact of outliers can be assessed, while R^2 can serve as a benchmark for the model's explanatory power.

2.4.1 K-fold cross-validation

K-fold cross-validation is a statistical method used to evaluate the performance of a model. The technique involves partitioning the dataset into K distinct segments, or "folds". For instance, a $K = 5$ configuration means the dataset is divided into five unique folds. During the evaluation process, each fold sequentially serves as the testing set, while a model is trained on the remaining $K - 1$ folds. This is illustrated in figure 2.8. The statistical metric in question is evaluated on the testing set, the score is stored, and the cycle is repeated for each fold. The model parameters are not kept between each iteration. The average of the statistical scores from each cycle is then considered the final score for the model evaluation.

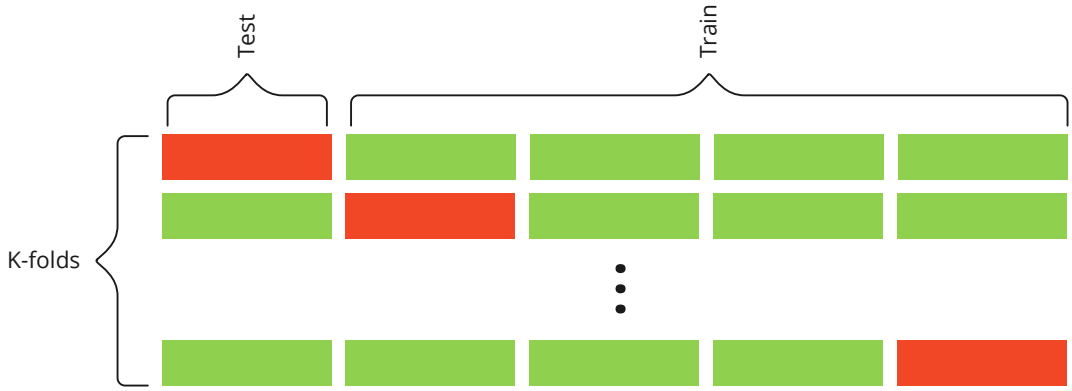


Figure 2.8: Diagram showing the individual folds and selection done in K-fold cross-validation.

The primary strength of K-fold cross-validation lies in its ability to utilize the available data for both training and testing purposes. This is particularly beneficial in scenarios where the dataset is small, as all of the available data can be used to validate the model's performance, without mixing training and testing.

Mathematically, the cross-validation estimate of a statistical metric L can be expressed as:

$$\mathrm{CV} = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, \hat{f}^{-\kappa(i)}(\mathbf{X}_i)\right),$$

where $\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\}$ is the function mapping a datapoint to its corresponding fold and $\hat{f}^{-k}(X)$ is the model trained on everything but the kth fold.

Chapter 3

Method

3.1 Data representation

A set of impedance spectra, represented by a two-dimensional tensor $\mathbf{X}$, is used as the model's input. Each spectrum is indexed by i and the impedance measurements within it is indexed by j. The mathematical representation is as follows in equation 3.1

$$\mathbf{X} = \begin{bmatrix} Z_{11} & \cdots & Z_{i1} & \cdots & Z_{m1} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ Z_{1j} & \cdots & Z_{ij} & \cdots & Z_{mj} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ Z_{1n} & \cdots & Z_{in} & \cdots & Z_{mn} \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_m \end{bmatrix} \tag{3.1}$$

Here, Z_{ij} is the impedance measured at the jth frequency in the ith spectrum. y_i refers to the SoH corresponding to spectrum i. Note also that n represents the number of frequencies in the spectra, while m is the size of the dataset or batch. An important consequence of this is that all spectra must be of the same size, and since the input tensor does not include the frequencies themselves, the spectra must include impedances corresponding to the same frequencies. See section 3.3.2 for further details.

3.2 Dataset

This study utilizes two datasets for developing and testing the model: one at the level of battery cells and one at the level of battery modules.

The cell-level dataset dataset comes from an openly available source provided by Chan et al. [41], and includes 13 life cycle test with accompanying EIS measurements, conducted on 34 individual 3 Ah NMC 18650 Lithium-ion cells by six different institutions. The dataset notably includes 5422 capacity measurements and 5479 EIS spectra [41]. A life cycle test consists of the following steps:

- The cells' capacity is measured by fully discharging from 4.2 V to 2.9 V at 23 °C and 1.25 A [41].
- After capacity measurement, the cells' SoC is adjusted to multiple predefined levels by discharging at 1.25 A from 4.2 V, followed by a rest period of at least 30 minutes before further measurements [41].
- EIS measurements are conducted with an excitation amplitude of 0.5 A at each SoC level [41].
- The cells are cycled from SoC 100% down to a SoC of 0%, giving a depth of discharge (DoD) of 100%, with the exception of three cells having a DoD of 70%, 50% and 30% respectively. A constant current-constant voltage (CCCV) charging protocol and constant current (CC) discharging is used [41].

Additionally, the dataset contains operational conditions, such as SoC and temperature. With a nominal capacity given by the manufacturer as 3 Ah [41], the SoH is also calculated, as shown in equation 2.1. All EIS spectra without a corresponding capacity measurement are discarded, as they provide no use for the purposes of this study. Moreover, a notable inductive EIS distortion was observed by institution 5, along with modified cycling behavior due to increased aging before the experiments at institution 6 [41]. This study aims to tackle noise in EIS spectra, and as such the measurements from institution 5 are kept. However, data from institution 6 is excluded to avoid training errors into the model. To ground the model in cell-level degradation patterns, this dataset will serve as the bulk of the training set.

The module-level dataset is based on original EIS measurements conducted on two distinct types of electric vehicle NMC battery modules. Both originate from the same EV model, but are from different generations and produced by different manufacturers. The first module type, referred to in this thesis as 4s3p, consists of 4 cells in series and 3 in parallel. The second module type, identified as 4s2p, consists of 4 cells in series and 2 in parallel. According to the manufacturers, the nominal capacities are 118 Ah for 4s3p and 78 Ah for 4s2p. All measurements were performed under an ambient temperature of 23 °C, akin to the cell-level dataset. A total of 27 modules were measured once each, being 8 4s2p modules and 19 4s3p modules.

For each EIS spectrum, the battery modules were charged and discharged at a CC of 20 A, with a DoD of 100%. This current translates to a C-rate of approximately 0.16 for the 4s3p module and about 0.25 C for the 4s2p module. The precise C-rate used by the module manufacturers to determine the nominal capacities is unknown. This also goes for the exact discharge curves, which means it is difficult to find the difference between the nominal capacity given in the specification and one found using the C-rates used in this study. However, given the industry standard of using 1 C or 0.5 C for these types of ratings, an educated guess is done: Experiments conducted by Latifatu et al. [23], Alqahtani & Williams [42] and Martha et al. [43] give discharge curves for various NMC chemistries at different C-rates. Based on these discharge curves, the difference in capacity when measured at 0.2 C and 1 C can be found to be as big as 20%. With the C-rates used in this study's experiments, a conservative uncertainty is therefore given for the nominal capacities of 5% for the 4s2p modules and 10% for the 4s3p modules.

Due to confidentiality agreements with Evyon AS, further details about the battery modules, the lab equipment and the EIS excitation amplitude is withheld. The raw EIS data is also not provided. Nevertheless, a Nyquist plot is presented in figure 3.1 to illustrate the general impedance behavior observed in these modules, without disclosing numerical details.

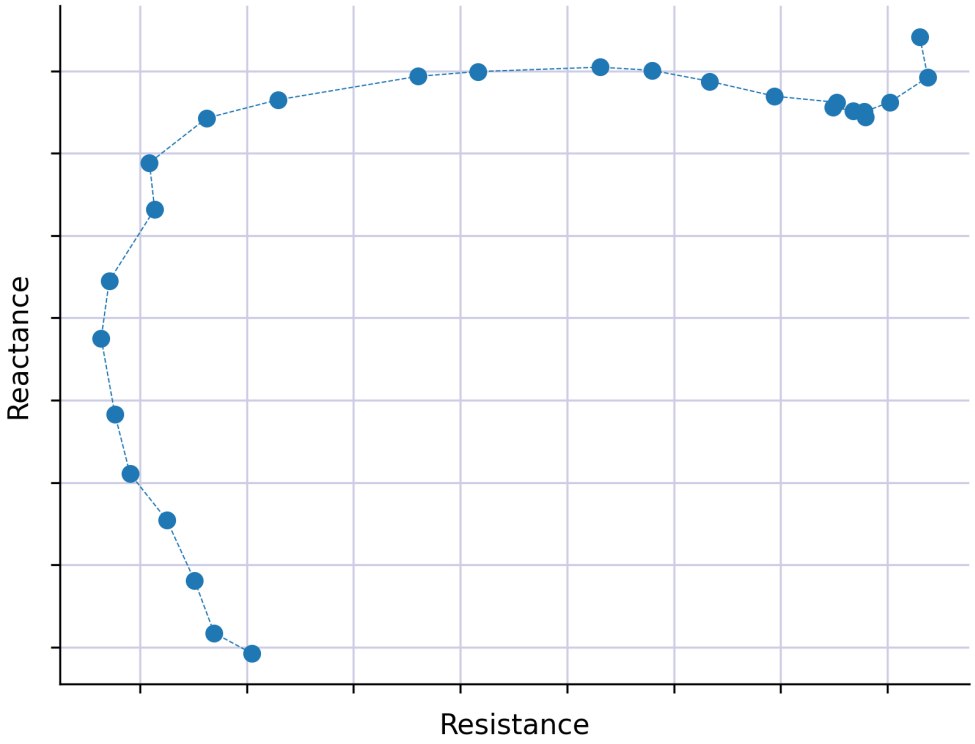


Figure 3.1: Nyquist plot illustrating the general impedance behavior of an electric vehicle NMC battery module with 4 cells in series and 3 in parallel. The specific impedance values are omitted due to confidentiality agreements with Evyon AS.

3.3 Data selection and preprocessing

The selection of data for model training and testing is a crucial step that ensures the robustness and accuracy of the predictions. Certain considerations have to be done when choosing the data, especially in order to minimize the impact of confounding variables.

3.3.1 Filtering based on confounding variables

To address the inherent temperature dependence explained in section 2.2.2, only EIS measurements performed at the specific temperature of 23 °C are chosen. This choice aligns with real-world scenarios, as it is the ambient conditions commonly encountered in labs and factories and mirrors the temperature at which nominal battery capacities are measured.

In contrast, filtering based on a specific SoC would risk a significant loss of data. As such, the variability imposed by the change in SoC is kept in the dataset and the SoC data is discarded. This choice aligns with the objective of developing a robust model that is representative of a wide range of real-world operating conditions. While this means accepting some level of SoC-induced variability, it ensures a comprehensive analysis that does not overly narrow the utilization of the model. Actions are also taken to address the effect of SoC variablility, as discussed in section 3.4.

3.3.2 Spectrum resampling

In addition to controlling other variables, an essential step is to harmonize the spectral data from different sources. The cell-level spectra size varies among institutions and measurements, with counts of 39, 48, 49, 57, 58 60 and 61 frequencies. The range of the spectra is from 0.01 Hz to 10 kHz. In contrast, the module-level spectra features 26 frequencies, and has a spectrum range from 0.05 Hz to 1 kHz.

To ensure that each spectrum matches in size and has identical frequencies — a requirement for the input tensor, which includes only impedance values and omits frequency information — a resampling of the cell-level data is performed to fit the module-level spectrum. This approach was chosen to avoid the need to extrapolate data, which would introduce hypothetical data points into the dataset. As a result, the cell-level dataset is resized to feature the same 26 frequencies found in the module-level dataset, namely: {0.05 Hz, 0.1 Hz, 0.2 Hz, 0.3 Hz, 0.5 Hz, 1 Hz, 1.5 Hz, 2 Hz, 3.5 Hz, 7 Hz, 10.5 Hz, 14 Hz, 25 Hz, 35 Hz, 75 Hz, 100 Hz, 150 Hz, 200 Hz, 300 Hz, 400 Hz, 500 Hz, 600 Hz, 700 Hz, 800 Hz, 900 Hz, 1000 Hz}.

3.3.3 Spectrum trimming

As shown in figure 2.6, external factors like temperature and SoC contributes to the impedance in varying degree at different parts of the spectrum. As such, trimming targetted frequencies might be an effective way to mitigate these contributions. To evaluate this, a number of frequencies — ranging from 0 to 24 — are systematically trimmed from the lower and higher ends of the module-level spectrum. For each spectrum subset, a resampling is performed on the cell-level dataset and a GPR model is applied to predict the SoH. The model's performance is then assessed using MAE, MSE and R^2. This analysis gives a set of metric values corresponding to each number of frequencies removed, and it is repeated for the proposed neural network to assess the generalizability of the conclusions.

To identify an optimal point $\mathcal{D}$ based on this analysis, a composite metric is calculated as the product of MAE and MSE divided by R^2. The minimum of this metric with regards to the number of frequencies trimmed n is used as the optimal point. It is defined as

$$\mathcal{D} = \text{argmin}_{n \in \mathbb{N}} \frac{\text{MAE}(n) \cdot \text{MSE}(n)}{R^2(n)}$$

This metric attempts to reflect the point which minimizes error and maximizes correlation.

3.3.4 Scaling of module data

Lastly, to perform estimations on modules based on learning from cell-level data, a way to generalize between cell-level and module-level impedance spectra must be found. SoH is a relative measure, so it does not depend on scale. However, the impedance does. It is essential to consider that modules consist of multiple cells arranged in series and parallel configurations, which means the impedance compounds. This compounding effect adheres to Kirchoff's circuit law, where serial impedances are summed, and parallel impedances are combined through the addition of their reciprocals. If this ideal compounding was the sole factor, estimating the average cell impedance based on the module impedance would be straightforward. However, the impedance contributions from connections between cells and the module's internal circuitry complicate this estimation.

Instead of building a detailed phyics-based model to estimate the average impedance of cells in a battery module, we use a simpler approach. This method assumes that multiplying each impedance value in the module's spectrum by a a scalar value can give us a close estimate of the average cell's impedance. Essentially, this approach ignores the detailed interactions within the module, suggesting that a straightforward multiplication can account for the combined effects of the cells and their connections.

3.4 Network architecture

The proposed neural network architecture is setup as a typical convolutional neural network and is kept as simple as possible. A diagram of it is shown in figure 3.2.

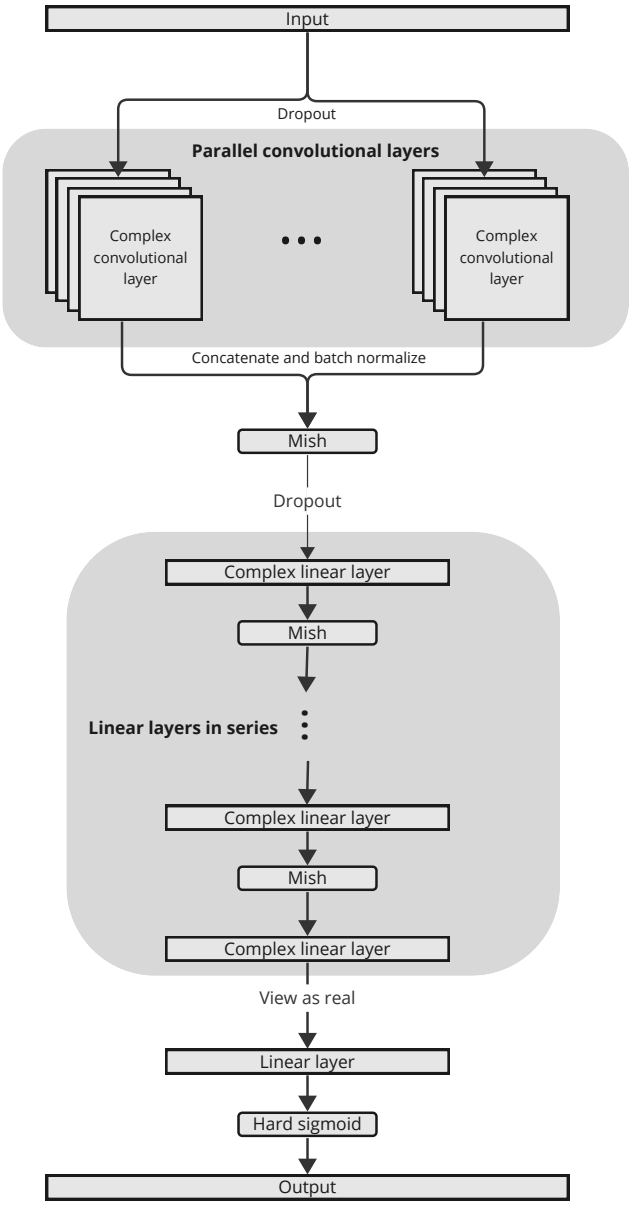


Figure 3.2: A schematic of the neural network architecture. The network accepts complex-valued input tensors of shape `(26,)`, representing an impedance spectrum measured from a lithium-ion battery. It outputs a real valued tensor of shape `(1,)` with a value between 0 and 1 which represents the SoH of that battery.

The input consists of a complex-valued tensor, as explained in section 3.1. This tensor is fed through the network in batches of size `batch_size`. The tensors in each batch are first processed through multiple 1-dimensional convolutional layers arranged in parallel (meaning they are all given the same input data,) each with varying kernel sizes. This approach is used to extract different shapes in the spectra, which may be better identified by wider or more narrow kernels.

Following the convolutional layers, the network concatenates their outputs and applies batch normalization to ensure consistent learning. The tensor then passes through a series of artificial neural networks with interspersed Mish activation functions to reduce the output to the shape `(batch_size, 1)`. Since the output is a real value, the complex tensor is viewed as real by splitting the complex numbers into real and imaginary parts, resulting in a tensor of shape `(batch_size, 2)`. This tensor is passed through the last artificial layer, collapsing it to shape `(batch_size, 1)`. Finally, the hard sigmoid activation function is used to limit the network's output to a range from 0 to 1.

The rationale for integrating dropout into the architecture is to address the variability in the battery's impedance spectrum imposed by the change in SoC. As shown in section 3.3.1, this mainly affects the lower frequencies. However, these also encode information with regards to SoH — as described in section 2.2.1 — so removing them could potentially lead to loss of important data. This concern is mostly addressed via the spectrum trimming analysis (see section 3.3.3), but the first dropout layer serves as a method of letting the training process identify the ideal non-reliance on different parts of the spectrum automatically. Following the convolutional blocks, the second dropout layer ensures that the refined features extracted are not overfitted, allowing the network to develop a deeper understanding of the SoH indicators.

In addition, performing EIS measurements in real-world applications, for example in an operational EV context, may result in noisy data and bad reproducibility [24]. Dropout can serve as an effective mitigation strategy of this too, as it prevents the network from overfitting based on specific features. This approach helps distribute the learning process more evenly across the frequency spectrum, which reduces confusion based on noise and makes it more robust to unseen data.

3.5 Training and hyperparameter optimization

The model's loss function is set to the MAE and it is trained via backpropagation using the Adam optimizer, provided PyTorch. Introduced by Kingma and Ba [44], the Adam optimizer is an algorithm for efficiently performing gradient based optimization. The algorithm accepts a learning rate, which adjusts how aggressively the model parameters are adjusted. The initial learning rate is set to a known value, and reduced during training according to the rules specified by the `RLReduceOnPlateau` learning rate scheduler, also provided by PyTorch. This is done to initially ensure that the optimizer does not get stuck in a local minima, while still managing to converge on the optimal point at the end of the training session. The hyperparameters are:

1. **Module scale**: A scalar that each value in the impedance spectrum is multiplied by, as described in section 3.3.4. This scalar is only used on the module-level data.
2. **Batch size**: The dataset is divided into batches of this size, and these batches are processed through the network prior to the adjustment of the model parameters. If the size of the dataset is not divisible by the batch size, the last batch may be smaller.
3. **Starting learning rate**: The initial value for the learning rate.
4. **Learning scheduler threshold**: When the validation loss changes less than the minimum loss multiplied by this threshold, the learning rate is reduced.
5. **Learning scheduler patience**: The scheduler waits this many epochs before potentially reducing the learning rate.
6. **First dropout rate**: The dropout rate of the first dropout in the neural network.
7. **Kernel sizes of convolutional blocks**: This list is configurable both in its length and its values. The length specifies the number of convolutional block that are stacked in parallel, and the values specify the kernel size of the CNN in each block.
8. **Second dropout rate**: The dropout rate of the second dropout in the neural network.

9. **Sizes of artificial hidden layers**: This list is also configurable in its length and values. The length specifies the number of linear layers that are stacked in a row, and the values specify the size of each layer. The layers are sorted decreasingly by size before the network architecture is created.

The hyperparameter tuning is conducted using a GA, as outlined in section 2.3.3.3. The GA proceeds as follows:

1. The process begins by creating an initial population of n randomly generated hyperparameter candidates, each encoded as a chromosome. The chromosomes are represented as boolean arrays.
2. For each generation, the following process is performed:
    i. The fitness of each individual in the population is found. This involves decoding the chromosome into hyperparameters, instantiating a neural network, training it on the training set through 100 epochs and evaluating the R^2 score of the network on the validation set. This score serves as the fitness of that individual.
    ii. A set of *parents* from the current population is selected to create the next generation. This is done by sorting the population by fitness and selecting the top m individuals. $m < n$ and is determined by a selection rate p as the product $m = \lfloor np \rfloor$, where $p \in (0, 1)$.
    iii. Pairs of individuals are randomly chosen from the pool of parents to undergo crossover, producing offspring that inherit characteristics from both parents. Crossover is done by uniformly selecting genes from each of the parent's chromosomes.
    iv. To maintain genetic diversity within the population and to prevent converging on local optima, mutation is applied. This step involves uniformly making small random changes in the offspring's chromosome. The mutation rate r controls the probability of a mutation happening.
    v. Offspring are generated until the number of parents and offspring total to n. This replaces the population for the next generation.
3. The algorithm terminates after a set number of generations. The best solution (highest fitness) from the final generation is decoded to retrieve the optimal set of hyperparameters.

n is set to 40, p is set to 0.25 and r is set to 0.04. The number of generations is set to 50. The GA is set to tune the hyperparameters to maximize the R^2 metric for the cell-level validation set.

3.6 Frameworks and implementation

The model is implemented in Python using the PyTorch [45] library and its implementations of 1-dimensional convolutional networks, linear layers, the Mish activation function, the MAE loss function and the Adam optimizer. The rest of the code used in this study — like the GA, K-fold cross-validation and tensor handling functionality — is implemented specifically for this study, and combined into a Python package named `thesis`, which is available in the Git repository hosted on https://git.sr.ht/~kmaasrud/code. This package is then used in scripts to produce the results.

3.6.1 On PyTorch modules without support for complex tensors

In version 2.2 of PyTorch — which is used in this study — complex tensors and many operations on them is supported [46]. However, not all relevant modules, like dropout, most activation functions and batch normalization, are supported [47]–[49]. There are many reasons why this is not done yet: For example, many of the functions provided by PyTorch are not well-defined in the complex plane, which means that a concious choice has to be made on whether it should be applied to the real and imaginary components separately, or to the magnitude and phase [49]. Because of this, the PyTorch modules that lack support for complex tensors have been replaced by custom modules written specifically for this study. These all share the concious choice of applying the operation on the real and imaginary parts, separately. The implemented modules are:

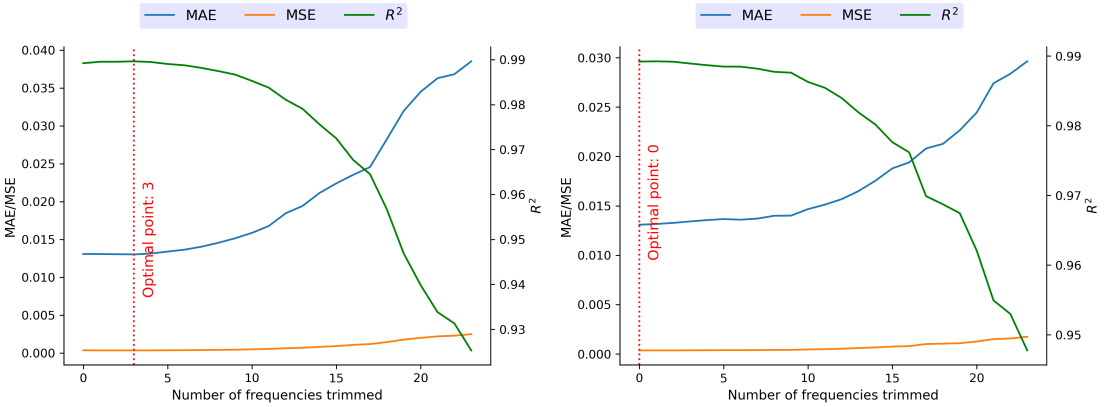
- `thesis.net.ComplexMish` in place of `pytorch.nn.Mish`.
- `thesis.net.ComplexDropout` in place of `pytorch.nn.Dropout`.
- `thesis.net.ComplexBatchNorm1d` in place of `pytorch.nn.BatchNorm1d`.

Chapter 4

Results and discussion

4.1 Trimming frequencies

As described in section 3.3.3, a model was trained and evaluated with different variants of the module spectrum. To get a feel for how the trimming itself affected the accuracy of the predictions, this was done with a model that is known to produce good results, namely GPR. Originally, the spectrum contained 26 frequencies, and a process was followed where up to 24 frequencies were eliminated from both the lower and upper extremes. The performance of the model was assessed for each modified spectrum. These assessments are visualized in figure 4.1. It was discovered that the removal of 3 frequencies from the lower end of the spectrum gave MAE $= 0.013$, MSE $= 0.00035$ and $R^2 = 0.99$, yielding the *optimal point* as defined in section 3.3.3. Conversely, the optimal point when frequencies were trimmed from the upper end of the spectrum occured at no frequencies removed, exhibiting similar metrics with MAE $= 0.013$, MSE $= 0.00036$ and $R^2 = 0.99$.



(a) Frequencies trimmed from the lower end of the spectra

(b) Frequencies trimmed from the higher end of the spectra

Figure 4.1: MAE, MSE and R^2 of a GPR trained and evaluated with different number of frequencies trimmed from the low or high end of the spectrum, shown in (a) and (b), respectively. The optimal point is calculated as $\text{argmin}_{n\in\mathbb{N}} \frac{\text{MAE}(n)\times\text{MSE}(n)}{R^2(n)}$.

While the removal of 3 frequencies from the lower end provides a marginal performance boost, the results are generally as expected: Data-driven methods perform better given more data. However, the model performance is surprisingly stable with increasing number of trimmed frequencies. Case-in-point, the lower end of the spectrum appears to be relatively dispensable up to a certain threshold, beyond which the loss of information compromises model accuracy. The higher end shows a similar trend, perhaps with a slightly smaller margin for removal. In practical terms, this has significant implications. Reducing the spectrum size naturally decreases the model's complexity and the computational resources needed for training and evaluation. This again offers less computational cost and potentially faster iteration during the model development phase. More importantly, the potential of removing lower-end frequencies without loss in quality can lead to faster EIS measurements. Since lower frequencies require a longer measurement time, removing them could substantially shorten the total measurement duration without compromising the SoH estimation's accuracy. This improvement in efficiency could prove crucial for large-scale deployment.

It is worth noting again that this analysis was only performed on the cell-level dataset. Extrapolating the findings to module-level data thus remains speculative at best. Given the increased complexity of module data, a broader dataset than the one available in this study would be required to perform a similar analysis. This shows an important avenue for future research. Given this, the decision was made to retain the complete spectrum of 26 frequencies for all subsequent experiments conducted in this study.

4.2 Hyperparameter tuning

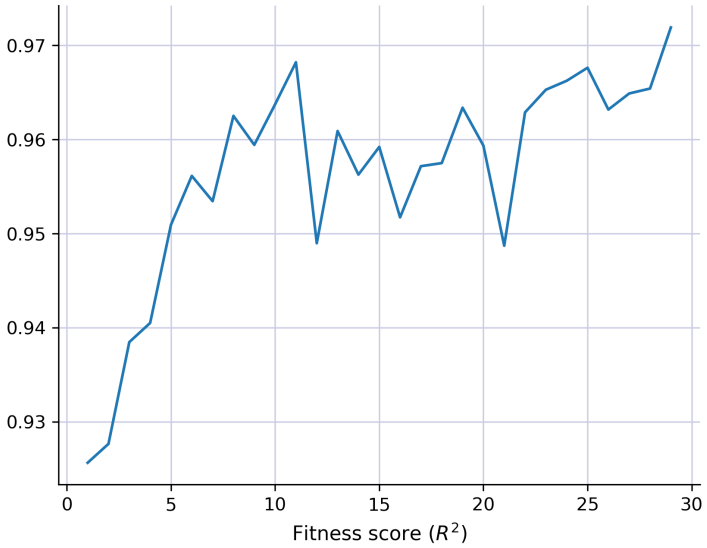


Figure 4.2: Evolution of the fitness score of the genetic algorithm over 29 generations.

The GAs search for optimal hyperparameters began with a relatively substantial population size, to cover a broad search space and increase the probability of finding a high-performing initial solution. This can be seen by the fact that the first generation starts off with a high fitness score. The score progression can be seen in figure 4.2, and it is relatively noisy. However, it shows a steady increase, which is a good indication that the GA mutation is effective in avoiding local minima. In summary, employing a GA for hyperparameter tuning has proven to be successful.

The final hyperparameters found by the genetic algorithm are shown in table 4.1. Throughout the generations, several hyperparameters were adjusted. Notably, the starting learning rate was fine-tuned to the relatively low value of 0.0000397, which suggests that the model benefits from a gradual and nuanced learning process, and that the loss landscape has a low occurence of local minima. The tuning of dropout rates to relatively low levels, specifically to 0.00052 for the first rate and 0.0055 for the second, might suggest that the model requires most of the data, challenging the rationale for using dropout as outlined in section 3.4. However, it is worth noting that the GA had the option to adjust these rates to 0, but chose to maintain, for instance, the second dropout rate at 5‰. This decision implies that there might be a benefit in using a small dropout as a method of enhancing the network's resilience to variations in parameters. Lastly, the GA finds larger networks to perform better, favoring a bigger number of CNN's and linear layers.

Table 4.1: The optimal hyperparameters found after performing the GA tuning.

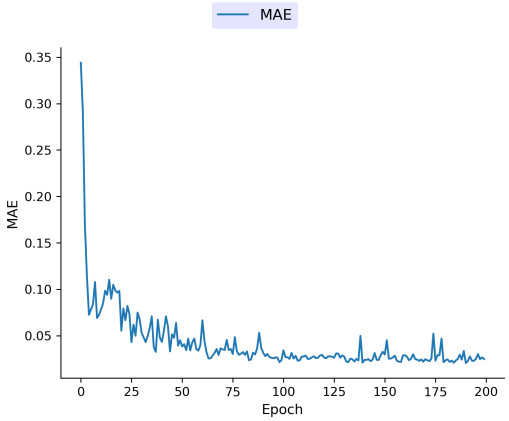
Hyperparameter	**Tuned value**
Module scale	0.1
Start learning rate	2.62×10^{-5}
Scheduler threshold	0.00865
Scheduler patience	18
Batch size	17
First dropout rate	0.000996
Second dropout rate	0.0285
CNN kernel sizes	$\{9, 9, 8, 7, 7, 5, 4, 3, 2\}$
Size of lineary layers	$\{56, 43, 41, 24, 23, 17, 12, 9, 5\}$

4.3 Training session

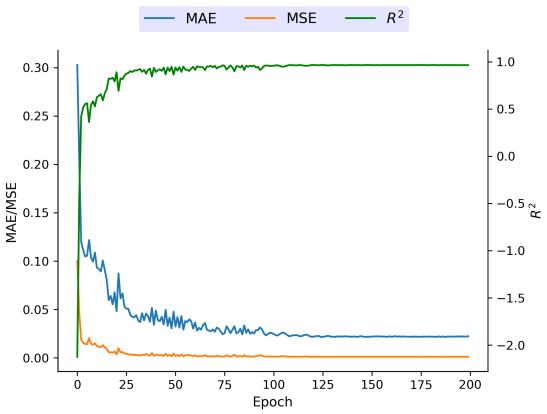
During training of the proposed model on cell-level — data using the tuned hyperparameters listed in section 4.2 — the performance metrics bettered (increased/decreased) and stabilized quickly, as seen in figure 4.3. This rapid convergence, typically within the first 100 epochs, suggests an efficient adaptation to the data.

The loss function, as shown in figure 4.3a, presents a pronounced decline within the initial epochs, which indicates a rapid phase of learning. Following this steep learning curve, the model enters a phase of stabilization. The loss function appears to plateau, suggesting that the model has reached a state of minimal error improvement and is delivering consistent performance. Figure 4.3b further reinforces these findings, and indicates that the model is able to generalize well to unseen data — even during training. Notably, the convergence and stabilization of these values occur within the first 100 epochs, which indicates that the model adapts to the training data relatively quickly.

This fast stabilization hints that the hyperparameter optimization performed by the GA resulted in a well-tuned model, and that the training likely identifies the minimum for the network architecture. The absence of significant improvement or further stabilization past the 100-epoch mark aligns with this, suggesting that the model efficiently captures the underlying patterns. Having a relatively simple network architecture — with fewer parameters — usually makes this training efficiency simpler to achieve, and it is reassuring to see this confirmed. A faster training session is particularly beneficial when scaling up to larger datasets, and it implies a reduced computational cost of training and development.



(a) Loss function over 200 epochs during the training phase of a model using cell-level data. The trend indicates a sharp decline in loss initially, followed by a gradual stabilization, suggesting the model is improving its performance in a desireable fashion.



(b) Mean absolute error, mean squared error and R^2 of the validation set across 200 training epochs. The mean absolute error and mean squared error both show a sharp decrease initially and then plateau, which indicates initial learning followed by convergence to a minimum error. The R^2 value increases and stabilizes at roughly the same rate. This convergence of loss metrics and stability of R^2 suggest that the model achieves consistent performance on the validation set after a certain number of epochs.

Figure 4.3: Progression of statistical metrics for training and validation metrics during training of the proposed model. Subfigure (a) shows the loss function behavior during model training, while subfigure (b) depicts the MAE, MSE, and R^2 values on the validation set, both across 200 epochs.

4.4 Performance evaluation and comparison

Both the cell-level and module-level datasets were split into a training and testing set, and both the `scikit-learn` GPR model and the proposed neural network were trained and evaluated on different combinations of these sets. Since the testing set for the modules would be so small (a fifth of the complete dataset would include only 5 spectra,) any metrics that are not evaluated on the whole module-level dataset are found using 5-fold cross-validation. The results are seen in table 4.2 for the GPR and in table 4.3 for the proposed neural network. It is worth noting that the broader range of possible SoH values differs between the cell-level dataset (ranging from 0.2 to 1) and the more limited range in the module dataset (from 0.89 to 0.94.) Thus, a negative R^2 score for the module-level data does not necessarily indicate poor model performance, as the smaller range gives a low MSE for the mean.

The GPR model shows strong performance on the cell-level data, achieving high accuracy with minimal error and a substantial R^2 score. This outcome is as expected, given GPR's frequent application and proven efficacy in cell-level SoH estimation tasks. However, its ability to adapt when applied to module-level data is notably weaker, as seen in the drastic increase in the error. This highlights that the model is struggling to generalize from cell-level training to module-level evaluation effectively. This improves slightly when the training set includes module-level data, but that is likely attributed to identifying whether a datapoint is a module or not, and performing estimations accordingly.

Table 4.2: MAE, MSE, and R^2 obtained by predictions performed with GPR across different splits of training and testing sets. The evaluation includes both cell-level and module-level datasets, with the models being trained on one dataset or a combination of them. When using the module data as the testing set, 5-fold cross-validation is employed, in order to utilize all of the data to find the statistical metric in question.

Trained on	**Evaluated on**	**Using 5-fold cross-validation**	**MAE**	**MSE**	R^2
Cell-level training set	Cell-level testing set	*No*	0.0132	0.000303	0.990
Cell-level training set	Complete module-level dataset	*No*	0.240	0.0660	−392
Module-level training set	Module-level testing set	*Yes*	0.0108	0.000176	−0.470
Complete cell-level dataset and module-level training set	Module-level testing set	*Yes*	0.417	0.258	−3378

While the proposed neural network showed worse performance than the GPR model in pure cell-level predictions, it demonstrates a much better ability to generalize to the module data. Its predictive performance is notably better in this regard, even when trained exclusively on cell-level datasets. This suggests that the neural network has managed to uncover patterns within the data that the GPR model failed to identify.

Table 4.3: Statistical metrics, including MAE, MSE, and R^2, for the proposed neural network as showed in figure 3.2. The assessment is performed across cell-level and module-level datasets, training the models either separately on each dataset or on a merged dataset. For module data testing, 5-fold cross-validation is once again applied to fully leverage all available data in determining the specific statistical metric.

Trained on	**Evaluated on**	**Using 5-fold cross-validation**	**MAE**	**MSE**	R^2
Cell-level training set	Cell-level testing	*No*	0.0207	0.00102	0.970

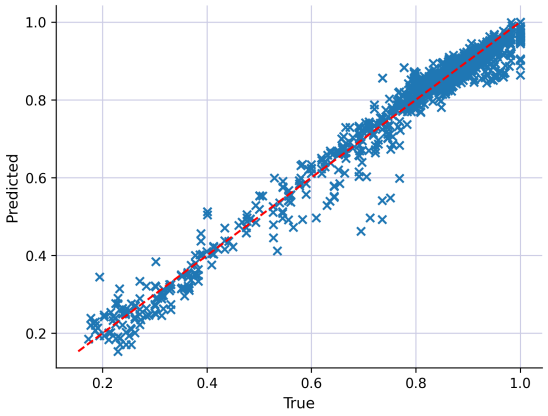
Trained on	Evaluated on	Using 5-fold cross-validation	MAE	MSE	R^2
Cell-level training set	Complete module-level dataset	*No*	0.0505	0.00364	−20.7
Module-level training set	Module-level testing set	*Yes*	0.0757	0.00591	−40.9
Complete cell-level dataset and module-level training set	Module-level testing set	*Yes*	0.0935	0.0123	−160

Considering the uncertainties in nominal capacities of 5% for the 4s2p modules and 10% for the 4s3p modules, the performance of the predictive models may still be seen in a positive manner. Figure 4.4 shows a plotted comparison between the true and predicted SoH values for both the cell-level and module-level dataset, as given by the GPR and the proposed neural network. For the cells, the performance can be considered relatively similar, although the GPR has a much lower deviation and manages to predict most of the cells' SoH correctly. When viewing the module predictions, the neural network is notably more accurate for the 4s2p modules, likely because of the smaller uncertainty in the nominal capacity. While the predictions for the 4s3p modules are not as precise — often underestimating the SoH — they do intersect with the identity line within the margins of uncertainty. This deviation suggests that the actual retained capacities could be lower than measured, because the manufacturers likely have used a higher C-rate than in the experiments performed at Evyon AS. Correcting this would give lower true SoH values, potentially validating the network's predictions as more accurate than initially thought. Though this theory remains speculative, it offers a constructive avenue for future investigation. The GPR, in contrast, fails to provide realistic predictions for the module-level dataset, indicating all modules to have SoH values above 1, a clear overestimation.

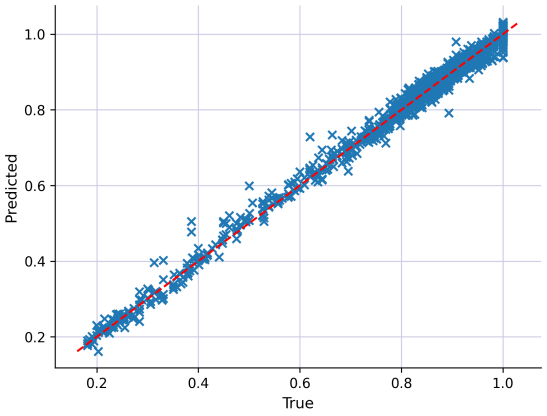
Compared to an ECM model by Q. Zhang et al. [50], which gave a MAE of 0.0270[1] for predictions on NMC cells at a similar temperature range, the proposed neural network performs marginally better — with a MAE of 0.0212. It also surpassed another GPR model trained by data from LCO cells by Y. Zhang et al. [51], which had an R^2 of 0.88, inferior to the one achieved by the proposed model, where $R^2 = 0.968$. However, the model's performance, in terms of MSE (0.00105), did not exceed that of the IPSO-CNN-BiLSTM by Li et al. [52], which is an advanced recent design that combines physical modeling, CNNs and a bidirectional long short-term memory (LSTM) network. It was trained on LCO cells and demonstrated a lower MSE of 0.00042849[2]. While the neural network presented in this study does not set new benchmarks in cell-level state of health estimations, it delivers performance comparable to the current state of the art. More importantly, it demonstrates the potential for a neural network configuration of this nature to identify the SoH of battery modules. Although the current accuracy of 5% does not meet the standard required for industrial deployment, the development of new techniques and a bigger dataset may improve the performance to a point where it can be used in a real-world application.

[1]The errors in this paper are presented as averaged error in percentage points. This is done individually for an SoC of 100%, 80%, 50% and 30%, and a temperature of 10 °C, 20 °C, 30 °C, 40 °C and 50 °C. The given MSE is found by averaging the errors for all SoC values and the temperatures 20 °C and 30 °C, and converting to decimal form by dividing by 100.

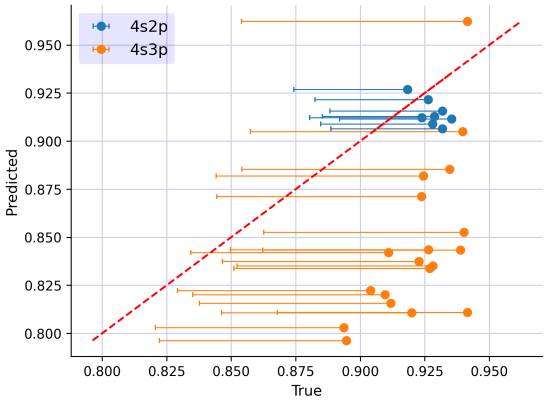
[2]The error is given in RMSE and is calculated for two different cells. To compare it to the model in this study, the average RMSE for the two cells is found and then converted to MSE by squaring.



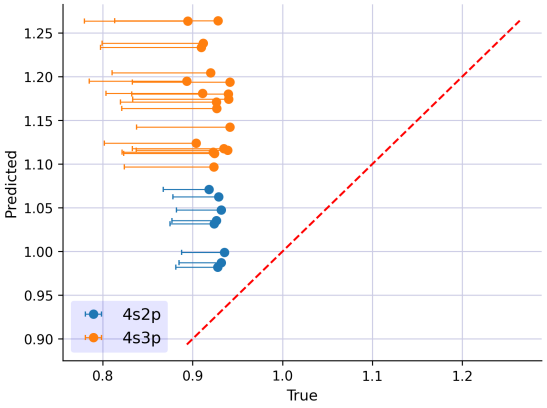
(a) SoH values from the cell-level dataset predicted by the proposed neural network.



(b) Cell-level SoH predictions by the GPR against actual values.



(c) SoH predictions by the neural network for the 4s2p and 4s3p modules, compared to their true values.



(d) Plot showing the SoH as predicted for the 4s2p and 4s3p modules by means of GPR.

Figure 4.4: This set of plots collectively demonstrate the predictive capabilities of the neural network and GPR models in assessing SoH at both the cell and module level. Both the GPR and the neural network are here trained on the cell-level training set. Each graph provides a visual correlation between predicted SoH values and true measurements, and an identity line is given in red as a guide. Errorbars are added to the true module-level SoH values, as the C-rate used by the manufacturer when finding the nominal capacity is unknown. These bars thus reflect the range that the true value could be in. A detailed description of this can be found in section 3.2.

4.5 Future prospects

The results of this study pave the way for several avenues of future research that are essential for the advancement of EIS-based battery health assesment. A critical piece of that puzzle is to expand the data that is available from modules, to encompass a broader range of SoH values and capture the full spectrum of degradation. However, collecting such a dataset poses significant challenges, as it demands substantial time, resources, and power to systematically degrade batteries — more so given the larger capacity of battery modules. This process also involves performing complex EIS measurements at the module level, which is technically demanding. Given all these challenges and the value that can be produced with such data, the battery industry is naturally still protective of any produced dataset.

Moreover, the influence of temperature on battery impedance warrants further investigation. Exploring a wider temperature range could shed light on its effects on impedance and potentially improve model accuracy. Incorporating temperature as an input parameter to the model might also give improved performance and make it usable in more varying environments. While adding SoC as to the input might also increase performance, SoC is difficult to measure, so maintaining a model independent of SoC — as attempted in this study — is still preferable.

Extending the exploration to other cathode chemistries is another significant area of interest. It would be valuable to assess whether a chemistry-agnostic model is feasible, given the specific degradation patterns and impedance characteristics inherent to each chemistry type. The same goes for differing battery configurations. It is essential to examine how models adapt to varying numbers of cells in different series and parallel combinations. The current study's success in predicting module-level SoH, based on a single-cell configuration, is encouraging. Yet, determining whether this success can be extrapolated to other configurations requires further research.

Lastly, the exploration of alternative network architectures could lead to enhanced generalizability in module-level SoH estimation. Investigating whether increased network complexity or alternative architectures can contribute to this is an important research avenue. The end goal is to devise a machine learning model that is both efficient and robust, capable of providing accurate SoH estimations across a myriad of battery types and conditions.

Chapter 5

Conclusion

This thesis shows that it is possible to perform SoH estimations on lithium-ion battery modules in EVs by using EIS combined with machine learning, contributing to the field of battery health diagnostics.

It was demonstrated that the use of machine learning methods, especially CNNs, might be an efficient way of interpreting complex EIS data. While the proposed neural network performs worse than existing data-driven methods like GPR on cell-level data, the results are still promising: They show that a model can indeed be trained to predict module-level SoH with reasonable accuracy, using only EIS data from cells. However, the findings show that the model still fails to fully capture the correlation in the degradation patterns. As such, the accuracy of the model is not adequate for usage in real-world scenarios.

Using genetic algorithms for hyperparameter optimization was shown to be an efficient way of exploring the hyperparameter space, without getting trapped in local optima. With that, it helped in fine-tuning the machine learning model to perform better on the task at hand. Additionally, a promising strategy for EIS measurements was found through the process of frequency trimming. It is shown that by finding frequencies that have minimal impact on predictive accuracy, the process of acquiring EIS measurements can be made more efficient. This approach could enable faster data collection, which is crucial for generating the large datasets required for improving models.

The model's adaptability to module-level data indicates its potential usefulness in practical settings, marking a step forward in achieving quicker and more precise battery diagnostics. This represents a critical component in e.g. second-life battery utilization. The study provides many opportunities for future studies, especially in enhancing the model's precision and investigating its applicability to various battery chemistries and configurations. Therefore, this thesis should not be viewed as a conclusion but as an opening for further innovation and exploration in combining electrochemical science with computational intelligence, **towards a fully renewable future for everyone**.

Appendix A

Acronyms

AC Alternating current
BMS Battery management system
CC Constant current
CCCV Constant current-Constant voltage
CL Conductivity loss
CMS Cell management system
CNN Convolutional neural network
CV Constant voltage
DoD Depth of discharge
ECM Equivalent circuit modelling
EIS Electrochemical impedance spectroscopy
EOL End of life
EV Electric vehicle
EoL End of life
GA Genetic algorithm
GPR Gaussian process regression
LAM loss of active material
LCO Lithium cobalt oxide
LFP Lithium iron phosphate
LIB Lithium-ion battery
LLI Loss of lithium-ion inventory
LSTM Long short-term memory
MAE Mean absolute error
MSE Mean squared error
NCA Lithium nickel cobalt aluminium oxide
NMC Lithium nickel manganese cobalt oxide
PCA Principal component analysis
ReLU Rectified linear unit
SEI Solid electrolyte interface

SoC State of charge

SoH State of health

Appendix B

External resources and licenses

The source code producing the results in this thesis are found on Sourcehut, through this link: https://git.sr.ht/~kmaasrud/code.

The source code for producing this PDF file is also availabel on Sourcehut. It can be found through this link: https://git.sr.ht/~kmaasrud/thesis.

The full cell-level dataset and accompanying data release document can be found at Zenodo at this link: https://zenodo.org/records/6418665. The data is transformed from a MATLAB table into a CSV and processed into a binary Numpy file by using a custom script. The CSV can be found at this link: https://git.sr.ht/~kmaasrud/code/tree/main/item/data/tensor-src/NMC. The processing script can be found at this link: https://git.sr.ht/~kmaasrud/code/tree/main/item/data/tensor-src/process_nmc.py.

Figures 2.3, 2.5 and 2.6 are reproduced with permission. The license agreements detailing the rights of this usage are listed in the directory hosted on this link: https://git.sr.ht/~kmaasrud/thesis/tree/main/item/licenses.

Appendix C

Calculations

C.1 Time and energy requirements of measuring battery capacity directly

When directly measuring battery capacity, a common practice involves using a C-rate of 0.5. That equates to a charging rate such that the battery is expected to fully charge or discharge over two hours. Given this, the total time required for a complete charge-discharge cycle, essential for measuring capacity, is four hours. For comparison, to perform an EIS measurement where each frequency is ran sequentially for 10 periods, the time spent would be just over 7 minutes using the spectrum in this study. This can be calculated as shown in the Python snippet below.

```
>>> freqs = [0.05, 0.1, 0.2, 0.3, 0.5, 1, 1.5, 2, 3.5, 7, 10.5, 14, 25,
...          35, 75, 100, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000]
>>> sum((10 / f) / 60 for f in freqs)
7.202183201058201
```

For a battery cell — which typically has a capacity of 2–3 Ah — it is a viable option to increase the C-rate to get faster measurement speeds. However, battery modules have a much bigger capacity. Given a module with a capacity of 100 Ah (some modules have an even higher capacity,) matching the measurement time of EIS would involve charging with a current of $\frac{1}{\frac{7}{60}\,\mathrm{h}} \cdot 100\ \mathrm{Ah} \approx 857\ \mathrm{A}$, which is unfeasible. Another point is that the C-rate used when finding the *nominal capacity* would have to be the same one. Often, the nominal capacity is measured according to some standard followed by the original manufacturer, and the measurement can not always be replicated after the fact.

The energy spent while performing a direct capacity measurement is also considerable. By making a rough estimate that there is a loss of $\eta = 0.1$ while charging and no loss while discharging, we can find the energy required for measuring a 100 Ah battery module at an average voltage of 50 V and a C-rate of 0.5 as:

$$E_{\text{charge}} = (1+\eta)PT = \eta UIT = 1.1 \cdot 50\ \mathrm{V} \cdot 50\ \mathrm{A} \cdot 2\ \mathrm{h} = 5.5\ \mathrm{kWh}$$
$$E_{\text{discharge}} = PT = UIT = 50\ \mathrm{V} \cdot 50\ \mathrm{A} \cdot 2\ \mathrm{h} = 5\ \mathrm{kWh}$$

This results in a total energy requirement of 500 Wh. For comparison, given an EIS excitation with an amplitude of 5 A measured at the same rough voltage $V = 50$ V and with the same loss of $\eta = 0.1$, we find the energy spent as:

$$E_{\text{EIS}} = PT = UT \int_0^T \eta(I) I(t) dt.$$

Here, $T = 7$ minutes is the total time spent as calculated above, and $I(t)$ is the full sequential EIS excitation curve. The integral can be calculated with the following Python script:

```
import numpy as np

def eta_scalar(eta, i):
    return 1 + eta if i > 0 else 1

freqs = [0.05, 0.1, 0.2, 0.3, 0.5, 1, 1.5, 2, 3.5, 7, 10.5, 14, 25,
         35, 75, 100, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000]

n_periods = 10
period = sum(n_periods / f for f in freqs)
n = 1000
v = 50
i = 5 * np.concatenate(
    [
        np.sin(
            2
            * np.pi
            * f
            * np.linspace(0, n_periods / f, int(n_periods / (f * period) * n))
        )
        for f in freqs
    ]
)
eta = np.array([eta_scalar(0.1, ii) for ii in i])

print(v * period * np.sum(eta * i * period / n) / 3600)
```

This yields an energy requirement of 396.5 Wh, which is approximately 80% the energy requirement of normal cycling.

References

[1] Greentec Auto, "Gen1 nissan LEAF battery module 7.6V. Greentec auto." Accessed: Mar. 14, 2024. [Online]. Available: https://www.greentecauto.com/hybrid-battery/repurposed-batteries/nissan-leaf/g1-nissan-leaf-nmo-7-6v-64ah-500wh-battery-module

[2] 天然ガス, *Nissan leaf at the 2009 tokyo motor show.* Wikimedia Commons, 2009. Available: https://commons.wikimedia.org/wiki/File:Nissan/_Leaf/_012.JPG

[3] International Energy Agency, *Global EV outlook 2019: Scaling-up the transition to electric mobility*. in Global EV outlook. OECD, 2019. doi: 10.1787/35fb60bd-en.

[4] E. Kastanaki and A. Giannis, "Dynamic estimation of end-of-life electric vehicle batteries in the EU-27 considering reuse, remanufacturing and recycling options," *Journal of Cleaner Production*, vol. 393, p. 136349, Mar. 2023, doi: 10.1016/j.jclepro.2023.136349.

[5] Amnesty International and Afrewatch, ""This is what we die for": Human rights abuses in the democratic republic of the congo power the global trade in cobalt," Jan. 2016. Available: https://www.amnesty.org/en/documents/afr62/3183/2016/en/

[6] B. E. Murdock, K. E. Toghill, and N. Tapia-Ruiz, "A perspective on the sustainability of cathode materials used in lithium-ion batteries," *Advanced Energy Materials*, vol. 11, no. 39, p. 2102028, Oct. 2021, doi: 10.1002/aenm.202102028.

[7] B. Niu, Z. Xu, J. Xiao, and Y. Qin, "Recycling hazardous and valuable electrolyte in spent lithium-ion batteries: Urgency, progress, challenge, and viable approach," *Chem. Rev.*, vol. 123, no. 13, pp. 8718–8735, Jul. 2023, doi: 10.1021/acs.chemrev.3c00174.

[8] C. H. Illa Font *et al.*, "Second life of lithium-ion batteries of electric vehicles: A short review and perspectives," *Energies*, vol. 16, no. 2, p. 953, Jan. 2023, doi: 10.3390/en16020953.

[9] K. Richa, C. W. Babbitt, G. Gaustad, and X. Wang, "A future perspective on lithium-ion battery waste flows from electric vehicles," *Resources, Conservation and Recycling*, vol. 83, pp. 63–76, Feb. 2014, doi: 10.1016/j.resconrec.2013.11.008.

[10] S. Wang, J. Zhang, O. Gharbi, V. Vivier, M. Gao, and M. E. Orazem, "Electrochemical impedance spectroscopy," *Nat Rev Methods Primers*, vol. 1, no. 1, p. 41, Jun. 2021, doi: 10.1038/s43586-021-00039-w.

[11] Y. Liu, L. Wang, D. Li, and K. Wang, "State-of-health estimation of lithium-ion batteries based on electrochemical impedance spectroscopy: A review," *Prot Control Mod Power Syst*, vol. 8, no. 1, p. 41, Dec. 2023, doi: 10.1186/s41601-023-00314-w.

[12] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017, doi: 10.1038/nature24270.

[13] OpenAI, "GPT-4 technical report," 2023, doi: 10.48550/ARXIV.2303.08774.

[14] F. Metz and M. Bukov, "Self-correcting quantum many-body control using reinforcement learning with tensor networks," *Nat Mach Intell*, vol. 5, no. 7, pp. 780–791, Jul. 2023, doi: 10.1038/s42256-023-00687-5.

[15] Sdk16420, *Schematics of a discharging li-ion battery using a wet electrolyte and solid seperator.* Wikimedia Commons, 2015. Available: https://commons.wikimedia.org/wiki/File:General/_discharging/_Li/_battery/_diagram.svg

[16] A. M. Raniseth, "Synthesis, structural characterization, electrochemical properties, and thermal stability of al-substituted LiMn_1.5Ni_0.5O_4," Master's thesis, University of Oslo, 2022.

[17] J. W. Fergus, "Recent developments in cathode materials for lithium ion batteries," *Journal of Power Sources*, vol. 195, no. 4, pp. 939–954, Feb. 2010, doi: 10.1016/j.jpowsour.2009.08.089.

[18] S. Atalay, M. Sheikh, A. Mariani, Y. Merla, E. Bower, and W. D. Widanage, "Theory of battery ageing in a lithium-ion battery: Capacity fade, nonlinear ageing and lifetime prediction," *Journal of Power Sources*, vol. 478, p. 229026, Dec. 2020, doi: 10.1016/j.jpowsour.2020.229026.

[19] Y. Liu, C. Liu, Y. Liu, F. Sun, J. Qiao, and T. Xu, "Review on degradation mechanism and health state estimation methods of lithium-ion batteries," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 10, no. 4, pp. 578–610, Aug. 2023, doi: 10.1016/j.jtte.2023.06.001.

[20] S. Yang, C. Zhang, J. Jiang, W. Zhang, L. Zhang, and Y. Wang, "Review on state-of-health of lithium-ion batteries: Characterizations, estimations and applications," *Journal of Cleaner Production*, vol. 314, p. 128015, Sep. 2021, doi: 10.1016/j.jclepro.2021.128015.

[21] L. Ungurean, G. Cârstoiu, M. V. Micea, and V. Groza, "Battery state of health estimation: A structured review of models, methods and commercial devices: Battery state of health estimation: A structured review," *Int. J. Energy Res.*, vol. 41, no. 2, pp. 151–181, Feb. 2017, doi: 10.1002/er.3598.

[22] P. Iurilli, C. Brivio, and V. Wood, "On the use of electrochemical impedance spectroscopy to characterize and model the aging phenomena of lithium-ion batteries: A critical review," *Journal of Power Sources*, vol. 505, p. 229860, Sep. 2021, doi: 10.1016/j.jpowsour.2021.229860.

[23] M. Latifatu *et al.*, "Structural effect of conductive carbons on the adhesion and electrochemical behavior of LiNi0.4Mn0.4Co0.2O2 cathode for lithium ion batteries," *J. Electrochem. Sci. Technol*, vol. 9, no. 4, pp. 330–338, Dec. 2018, doi: 10.33961/JECST.2018.9.4.330.

[24] K. McCarthy, H. Gullapalli, K. M. Ryan, and T. Kennedy, "Review–Use of impedance spectroscopy for the estimation of li-ion battery state of charge, state of health and internal temperature," *J. Electrochem. Soc.*, vol. 168, no. 8, p. 080517, Aug. 2021, doi: 10.1149/1945-7111/ac1a85.

[25] M. Ank, J. Göhmann, and M. Lienkamp, "Multi-cell testing topologies for defect detection using electrochemical impedance spectroscopy: A combinatorial experiment-based analysis," *Batteries*, vol. 9, no. 8, p. 415, Aug. 2023, doi: 10.3390/batteries9080415.

[26] L. H. J. Raijmakers, D. L. Danilov, J. P. M. V. Lammeren, M. J. G. Lammers, and P. H. L. Notten, "Sensorless battery temperature measurements based on electrochemical impedance spectroscopy," *Journal of Power Sources*, vol. 247, Feb. 2014, doi: 10.1016/j.jpowsour.2013.09.005.

[27] P. Suresh, A. K. Shukla, and N. Munichandraiah, "Temperature dependence studies of a.c. Impedance of lithium-ion cells," *Journal of Applied Electrochemistry*, vol. 32, no. 3, pp. 267–273, Mar. 2002, doi: 10.1023/A:1015565404343.

[28] R. Gopalakrishnan *et al.*, "Electrochemical impedance spectroscopy characterization and parameterization of lithium nickel manganese cobalt oxide pouch cells: Dependency analysis of temperature and state of charge," *Ionics*, vol. 25, no. 1, Jan. 2019, doi: 10.1007/s11581-018-2595-2.

[29] J. Wang *et al.*, "High-efficient prediction of state of health for lithium-ion battery based on AC impedance feature tuned with gaussian process regression," *Journal of Power Sources*, vol. 561, p. 232737, Mar. 2023, doi: 10.1016/j.jpowsour.2023.232737.

[30] B. Py, A. Maradesa, and F. Ciucci, "Gaussian processes for the analysis of electrochemical impedance spectroscopy data: Prediction, filtering, and active learning," *Electrochimica Acta*, vol. 439, p. 141688, Jan. 2023, doi: 10.1016/j.electacta.2022.141688.

[31] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[32] D. J. MacKay *et al.*, "Introduction to gaussian processes," *NATO ASI series F computer and systems sciences*, vol. 168, pp. 133–166, 1998.

[33] S. Kiranyaz, T. Ince, O. Abdeljaber, O. Avci, and M. Gabbouj, "1-d convolutional neural networks for signal processing applications," in *ICASSP 2019 - 2019 IEEE international conference on acoustics , speech and signal processing (ICASSP)*, May 2019, pp. 8360–8364. doi: 10.1109/ICASSP.2019.8682194.

[34] D. Misra, "Mish: A self regularized non-monotonic activation function," 2019, doi: 10.48550/ARXIV.1908.08681.

[35] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Unc. Fuzz. Knowl. Based Syst.*, vol. 6, no. 2, pp. 107–116, Apr. 1998, doi: 10.1142/S0218488598000094.

[36] X. Hu, W. Yang, H. Wen, Y. Liu, and Y. Peng, "A lightweight 1-d convolution augmented transformer with metric learning for hyperspectral image classification," *Sensors*, vol. 21, no. 5, p. 1751, Mar. 2021, doi: 10.3390/s21051751.

[37] PyTorch Contributors, "Hardsigmoid. PyTorch 2.2 documentation." Accessed: Feb. 28, 2024. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Hardsigmoid.html

[38] I. D. Raji, H. Bello-Salau, I. J. Umoh, A. J. Onumanyi, M. A. Adegboye, and A. T. Salawudeen, "Simple deterministic selection-based genetic algorithm for hyperparameter tuning of machine learning models," *Applied Sciences*, vol. 12, no. 3, p. 1186, Jan. 2022, doi: 10.3390/app12031186.

[39] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for NAS," 2019, doi: 10.48550/ARXIV.1912.06059.

[40] J. McCall, "Genetic algorithms for modelling and optimisation," *Journal of Computational and Applied Mathematics*, vol. 184, no. 1, pp. 205–222, Dec. 2005, doi: 10.1016/j.cam.2004.07.034.

[41] H. S. Chan *et al.*, "LiBforSecUse data release - impedance spectra of life cycle tests of commercial 18650 cells." Zenodo, Mar. 22, 2022. doi: 10.5281/ZENODO.6418665.

[42] Y. M. Alqahtani and Q. L. Williams, "Reduction of capacity fading in high-voltage NMC batteries with the addition of reduced graphene oxide," *Materials*, vol. 15, no. 6, p. 2146, Mar. 2022, doi: 10.3390/ma15062146.

[43] S. K. Martha, J. Nanda, Y. Kim, R. R. Unocic, S. Pannala, and N. J. Dudney, "Solid electrolyte coated high voltage layered–layered lithium-rich composite cathode: Li1.2Mn0.525Ni0.175Co0.1O2," *J. Mater. Chem. A*, vol. 1, no. 18, p. 5587, 2013, doi: 10.1039/c3ta10586e.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, doi: 10.48550/ARXIV.1412.6980.

[45] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in neural information processing systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[46] PyTorch Contributors, "Complex numbers. PyTorch 2.2 documentation." Accessed: Feb. 01, 2024. [Online]. Available: https://pytorch.org/docs/stable/complex/_numbers.html

[47] kshitij12345, "[Complex] dropout and it's variants should support complex tensors issue #80256 pytorch/pytorch. GitHub." Accessed: Feb. 01, 2024. [Online]. Available: https://github.com/pytorch/pytorch/issues/80256

[48] nissy-shota, "BatchNorm for complex tensor issue #81749 pytorch/pytorch. GitHub." Accessed: Feb. 01, 2024. [Online]. Available: https://github.com/pytorch/pytorch/issues/81749

[49] rjkilpatrick, "Activation functions for complex tensors issue #47052 pytorch/pytorch. GitHub." Accessed: Feb. 01, 2024. [Online]. Available: https://github.com/pytorch/pytorch/issues/47052

[50] Q. Zhang, C.-G. Huang, H. Li, G. Feng, and W. Peng, "Electrochemical impedance spectroscopy based state-of-health estimation for lithium-ion battery considering temperature and state-of-charge effect," *IEEE Trans. Transp. Electrific.*, vol. 8, no. 4, pp. 4633–4645, Dec. 2022, doi: 10.1109/TTE.2022.3160021.

[51] Y. Zhang, Q. Tang, Y. Zhang, J. Wang, U. Stimming, and A. A. Lee, "Identifying degradation patterns of lithium ion batteries from impedance spectroscopy using machine learning," *Nat Commun*, vol. 11, no. 1, p. 1706, Apr. 2020, doi: 10.1038/s41467-020-15235-7.

[52] D. Li, D. Yang, L. Li, L. Wang, and K. Wang, "Electrochemical impedance spectroscopy based on the state of health estimation for lithium-ion batteries," *Energies*, vol. 15, no. 18, p. 6665, Sep. 2022, doi: 10.3390/en15186665.