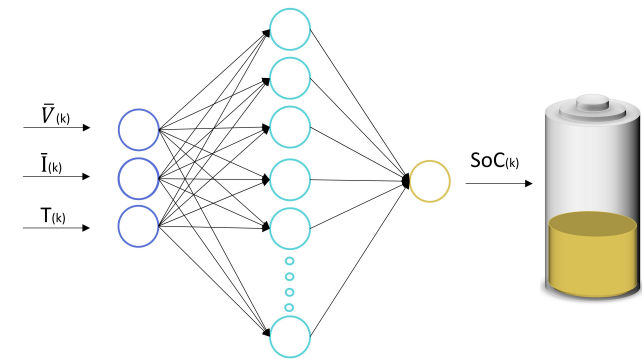


Ingvid Brekke Espedal

State of Charge Estimation of Lithium-ion Batteries by Neural Networks

June 2021





Norwegian University of
Science and Technology

State of Charge Estimation of Lithium-ion Batteries by Neural Networks

Ingvild Brekke Espedal

Mechanical Engineering

Submission date: June 2021

Supervisor: Jacob Joseph Lamb

Co-supervisor: Odne Stokke Burheim

Norwegian University of Science and Technology
Department of Energy and Process Engineering

Preface

In the master's program of Energy and Process Engineering at the Norwegian University of Technology and Science, all students must complete a master thesis, giving 30 ECTS. The work is supervised by Jacob J. Lamb, associate professor in digitalisation and sensor technology at EPT, NTNU and Odne S. Burheim, professor in energy storage at EPT, NTNU.

The choice to write about Lithium-ion batteries roots in curiosity towards its working principles and a personal interest in the industry that is establishing in Norway. Moreover, I wanted to explore what the buzz word "machine learning" was all about. Jacob J. Lamb and Odne Burheim were positive to guide me through a master combining lithium-ion batteries and machine learning, and the decision fell on researching how to estimate the state of charge for electrical vehicles with machine learning. It has been a steep learning curve doing this project as I had no prior knowledge about applying machine learning or electrochemistry before starting this work in August.

In spite of the limited theoretical experience, my practical experience with electric cars is larger. The first car I sat behind the wheel of was the electric vehicle Buddy, and later the experience was extended to include Think, Nissan Leaf, and Tesla. The experience with electrical cars is twofold: I cheer on emission-free driving and believe that batteries are an essential contribution to enable this. However, I have a range anxiety after numerous times pulling over to the side to charge because the car did not drive as far as the state of charge indicated at the start of the trip. Just this spring I drove my parents electric car several degrees bellow the freezing point. It ended up being a nerve-racking travel when the km-range dropped much faster than it should. My poor parents' who followed the travel on their phone switching remotely off our air condition to save energy. For the curious reader, we had a km-range of less than 1 km when parking the car at home. It is interesting to research a topic that has caused much frustration over the years and it motivated me to investigate for a better option. This work has been an exciting journey!

In my study hall there is a quote on the wall: "i dag, tiden bare forsvant... I morgen?". This sentence summarizes the semester. Each day starts with an intuition to create something extraordinary, study efficient, understand new theory, develop new revolutionary concepts, and structure the writing properly. Then, when the evening comes and I look up at the quote thinking

"I did get a little bit further today, but extraordinary, revolutionary?". Thankfully, I've had a lot of fantastic people supporting me through this work both academic and personal when things didn't go as planned. Thanks to Lea and Trym for all memorable moments in study halls, as well as outside, where the time flew by. Thanks to my parents for always being present when needed. Thanks to Inge, Arild, Jake, Mari, and Marte who have contributed to proof reading this work. Thanks to Arild for hugs when I was exhausted, giving me food when I was hangry, giving me space to complain when I was fed up with covid-19, and for all the laughter. I am looking forward to "now and forever".

Moving on to academics, Phil J. Kollmeyer, a research engineer at McMaster University, has been of great help with simulating the power curve of electrical vehicles. Per Arne Jansen, technical leader of Møller Bil, has been a great resource to ask when researching practical aspects of how the battery management system works. Martin Bustadmo, technician at NTNU, made the tab clamps by hand written sketches in no time when needed it. I appreciate constructive discussions about machine learning with three experts in the field; Tor Andre Myrvoll (NTNU), Keith Downing (NTNU), and Bartolomeo Stellato (Princeton University). Even though none of these people I knew me, they all reached out a hand to help. I am inspired and truly grateful!

The battery research team also deserves to be acknowledged. I want to thank Lena Spitthoff and Markus Solberg Wahl for your help in the lab. Thanks to my co-supervisor Odne Burheim who, together with Jake, taught me a lot on how to gain new knowledge and have been great motivators through the exciting and scary process of starting a new project. Last, but not least, a special thanks to my main supervisor Jacob J. Lamb for every weekly meeting throughout the year. Thanks for being supportive when I needed to discuss academic challenges, for all the work on the paper, and for having faith in the work when I did not have it myself, and for the effort to drag me out of all rabbit holes!

Abstract

By using neural networks, this thesis examines machine learning techniques and input features to increase the accuracy of State of Charge (SoC) estimation of lithium-ion batteries of electric vehicles. The aim to estimate the SoC with high accuracy is an underlying requirement to manage the energy system of electrical vehicles in real-time. The main challenge in this field is that the SoC cannot be measured directly. Furthermore, estimating the SoC is a nonlinear problem dependent on numerous variables including ambient temperature, heat generation, voltage, current drawn, aging, and cell chemistry. Machine learning can be a powerful tool to map input variables onto output variable(s) for complex problems and can be used to improve the SoC estimation in lithium-ion batteries.

This thesis contains a critical survey of previous work that utilizes machine learning to map physical battery variables into the SoC. In addition, neural networks have been constructed to research different techniques and input features to improve estimation accuracy. The results indicate that neural network approaches can be adequate when estimating the SoC, where the root mean squared error is around one percent. An important finding of this thesis is the effect of using the accumulated heat from the battery as an additional input feature in addition to the classical input features voltage, current, and battery temperature. With this new feature, the error was reduced by 27% from the benchmark case. However, the maximum obtained error was 6% SoC which is inconvenient in an electric car. This has to be reduced for the model to be used in a real-world application. The overall conclusion is that neural networks have the potential to be used as the SoC estimator in electrical vehicles. However, further work to couple electrical, thermal, and mechanical properties are required to ensure their robustness.

Sammendrag

Formålet med denne avhandlingen er å estimere ladetilstanden til litium-ionbatterier i elektriske biler. Ved bruk av nevrale nettverk undersøkes maskinlæringsteknikker og inputvariabler for å øke nøyaktigheten på estimert ladetilstand. Dette målet grunner i et underliggende krav for å administrere energisystemet til elektriske kjøretøy i sanntid. Fagområdets hovedutfordring er at ladetilstanden ikke kan måles direkte. Videre er estimering av ladetilstanden et ulineært problem som avhenger av flere variabler som omgivelsestemperatur, spenning, trukket strøm, aldring og cellekjemi. Maskinlæring kan være et kraftig verktøy for å transformere inputvariabler til outputvariabler ved komplekse problemstillinger og kan brukes til å forbedre estimeringen av ladetilstand i litiumionbatterier.

Denne rapporten er en kritisk analyse av tidligere arbeid hvor maskinlæring har blitt brukt til å transformere fysiske batterivariabler til batteriets ladetilstand. I tillegg er forovermatede nevrale nettverk konstruert for å undersøke forskjellige teknikker og inngangsfunksjoner for å forbedre estimeringsnøyaktigheten. Resultatene indikerer at nevrale nettverkstilnærminger kan være tilstrekkelig for å estimere ladetilstand med en kvadratisk gjennomsnittlig feil på rundt en prosent. Et viktig funn er at estimeringsfeilen for ladetilstand ble lavere ved å bruke akkumulert varme fra batteriet som en ekstra inputvariabel i tillegg til de klassiske inputvariablene spenning, strøm og batteritemperatur. Med den nye variabelen ble den største feilen redusert med 27% fra referansemодellen. Den maksimale feilen var imidlertid 6%, som er for høyt for en elbil. Feilen må reduseres for at det forovermatede nevrale netverket skal kunne brukes i praksis. Den overordnede konklusjonen er at nevrale nettverk har potensiale til å bli brukt til å estimere ladetilstanden til elektriske kjøretøy. Imidlertid kreves det videre arbeid for å koble sammen elektriske, termiske og mekaniske egenskaper for å sikre et robust estimat for ladetilstanden under ulike forhold.

Contents

Preface	iii
Abstract	v
Sammendrag	vii
Contents	ix
Figures	xiii
Tables	xvii
Code Listings	xix
Acronyms	xxi
Glossary	xxv
1 Introduction	1
1.1 Motivation	1
1.2 Problem Definition	3
1.3 Outline of the Thesis	4
2 Theory on Lithium-ion Batteries	7
2.1 Cell Structure and Working Principle	7
2.2 Voltage Potential and Losses in Lithium-ion Batteries	8
2.3 Capacity and C-rate	10
2.4 LiB Types	11
2.5 State of Charge Defined	12
2.6 Total Capacity	15
2.7 Residual Capacity	16
2.8 Discharge Capacity	16
2.8.1 Ah vs Wh as a Capacity Measure	17
2.9 SoC variations over Time	19
2.10 SoC of EVs	20
2.11 Methods to Estimate SoC	21
2.11.1 Direct Measurement	21
2.11.2 Book-keeping Estimation	22
2.11.3 Physics-Based Methods	22
2.11.4 Adaptive Systems	23
3 Theory on Machine Learning	25

3.1	Overview of Machine Learning Methods	25
3.2	Basic Principles of Supervised Machine Learning	26
3.3	Neural Networks	27
3.3.1	Artificial and Biological Neural Networks	28
3.3.2	Mathematical Explanation of a Neural Network	28
3.3.3	Overfitting and Overtraining	31
3.3.4	Activation Functions	33
3.3.5	Cost Function	34
3.3.6	Training Algorithms	35
3.3.7	Initialization	36
3.3.8	Scaling	37
3.3.9	Recurrent Neural Networks	37
3.3.10	Backpropagation Through Time	39
3.3.11	Radial Basis Function Network	40
3.4	Hyperparameter Tuning	41
3.5	Transfer Learning	42
4	Previous Work	43
4.1	Description of the Battery Data	43
4.2	Feed Forward Neural Networks	46
4.3	Recurrent Neural Networks	49
4.3.1	Gated RNNs	49
4.3.2	Other RNNs	50
4.4	Radial Basis Function Neural Network	50
4.5	Time Delayed Neural Network	51
4.6	Discussion and Analysis of the Literature Study	52
4.6.1	Data Analysis and Comparison	54
4.6.2	Optimization of Hyperparameters	55
4.6.3	Battery Chemistries	56
4.6.4	Accuracy Methods	56
4.6.5	Possible Errors	56
4.6.6	Terminology	57
4.6.7	Network Types	57
4.7	Summary of the Literature Study	57
5	Data Acquisition and Preprocessing	59
5.1	Data Acquisition from Public Database	59
5.2	Pre-Processing Data from Public Database	60
5.2.1	Limitations of the Panasonic Data Set	60
5.2.2	Data Cleaning	61
5.3	Data Acquisition from Cycling	64
5.3.1	Obtaining Drive Cycles	64
5.3.2	Description of Equipment and Lab Set-up	65
5.3.3	Regenerative Braking	66
5.3.4	Limitations and Special Considerations	66
6	Method	71

6.1	SoC Target Value	71
6.2	Input Features	73
6.3	Standard Functions and Parameter Settings	74
6.4	Optimizing the FNN Architecture	75
6.4.1	Random Search with the Panasonic Data Set	75
6.5	Transfer learning	79
6.5.1	Augmented Training Data	79
6.6	Temperature Difference as an Input Feature	80
6.7	Optimizing the RNN Architecture	82
7	Results and Discussion	85
7.1	Preliminary Studies	85
7.1.1	Selection of Train, Validation, and Test Data	85
7.1.2	Feature Selection by Boruta Shap	86
7.2	Hyperparameter Tuning and Feature Selection	86
7.3	Random Search	87
7.3.1	First Tuning Results	87
7.3.2	Window Size for Averaging Current and Voltage	87
7.3.3	Number of Hidden Layers, Nodes, and Input Features	89
7.3.4	Clipped ReLU and Leaky ReLU	92
7.4	Transfer Learning	94
7.5	Augumented data	97
7.6	Temperature Difference as Input Parameter	97
7.7	RNN Results	100
8	Conclusion	103
9	Further Work	105
	Bibliography	107
	Paper I	117
A	Driving Cycles	143
A.1	Plotting Power, Current and Voltage Relations	143
A.1.1	Experiment	144
B	Data Aquisition for the LCO cell	151
B.1	Battery Set-up	151
B.1.1	Simple Configuration	152
B.1.2	Tab Clapms	152
B.1.3	Internal Resistance Measurements	152
C	Results	155
C.0.1	FNN - Random Search Panasonic Data Set	155
C.0.2	ΔT Average scores - LG Data Set	157
D	Standard Deviation of Gaussian Noise	159
E	Boruta Shap Feature Extraction	161
F	Risk Assesment	165

Figures

1.1	Net electricity requirement load for California.	1
2.1	Schematic of a LiB cell	7
2.2	Potential vs the practically usable energy in a LiB	8
2.3	Available work in a LiB cell	9
2.4	Voltage vs time graph	12
2.5	SoC illustration	13
2.6	Discharge capacity in Ah	17
2.7	SoC when the capacity is given by both Ah and Wh. It is the same data [36] set used for estimating Ah and Wh (NMC battery).	18
2.8	Discharge capacity in Wh	19
2.9	SoC variations over time	20
3.1	Overview of subcategories of ML	25
3.2	Training, validation and testing data	27
3.3	Single node of an NN.	29
3.4	General FNN architecture	30
3.5	Overfitting	31
3.6	Illustration of dropout	32
3.7	K-fold	33
3.8	Activation functions	34
3.9	Backpropagation	36
3.10	General RNN Architecture	38
3.11	Simple RNN	38
3.12	LSTM cell	39
3.13	Backpropagation RNN	40
3.14	RBFNN Architecture	41
4.1	Drive Profiles in terms of current and voltage	45
4.2	FNN estimating SoC	47
4.3	SoC estimation with RNN	49

5.1	Flowchart anomaly detection	62
5.2	Observation of anomalies	63
5.3	Battery temperature Turnigy	64
5.4	Description of Equipment and Lab Set-up	65
5.5	Scaled cycles LCO	67
5.6	Drive cycle LCO	68
5.7	LCO high voltage	69
6.1	Averaging of measurements	73
6.2	QR code with scripts	74
6.3	Flowchart optimizing hyperparameters	76
6.4	Clipped and leaky ReLU	78
6.5	Agumented data	81
7.1	RMSE of different drive cycles	86
7.2	Window size for averaging	88
7.3	The legend holds information about the validation cycle used if CCCV charging data was included, and the two searches where the window size was increased to 4800, instead of 1200. More detailed information about each search number is found in Table 7.2.	91
7.4	Predicted SoC after random search	91
7.6	SoC prediction by FNN with Leaky and Clipped ReLU	93
7.7	LG battery. LR = Learning Rate.	94
7.8	Transfer learning scores with Samsung battery	95
7.9	Samsung. LR = Learning Rate. *Do not know why this score is 50 times higher than the rest. Most likely a human error in the saving process.	95
7.10	Turingy. LR = Learning Rate. As seen in the table, model number one did not converge, and hence the RMSE is not plotted.	95
7.11	SoC prediction on UDDS after transfer learning	96
7.12	SoC prediction on US06 after transfer learning	96
7.13	Error scores ΔT	98
7.14	SoC estimation $\overline{\Delta T}$	99
7.15	SoC estimation $\overline{\Delta T}$ HWFET benchmark case obtained by transfer learning	99
7.16	SoC estimation $\overline{\Delta T}$ HWFET benchmark case with randomly initiated weights	100
7.17	RNN Results	101
A.1	Power, Current and Voltage Relations	144
A.2	Experimental Results	150
B.1	LCO simple connection to channels and sense	152
B.2	Tab clamps	152
B.3	Tab clamps with description	153
E.1	Boruta shap 1	161
E.2	Boruta shap 2	162

E.3 Boruta shap 3 163

Tables

2.1	Properties of popular LiBs	11
3.1	Classifications of transfer learning	42
4.1	Results from the Literature Review	53
5.1	Description of the public data sets	60
5.2	Missing CCCV files for Panasonic data set.	61
5.3	Corrupt or missing files	61
5.4	LCO battery specifications	64
5.5	Transfer learning scores with Samsung battery	65
5.6	A table beside a figure	67
6.1	Total capacity calculations	72
6.2	Parameters to calculate standard deviation of noise	80
6.3	Search space random search	82
7.1	Search space random search	87
7.2	Results random search	90
7.3	LG battery transfer learning scores	94
7.4	Yo	98
A.1	Experimental Results	148
C.1	Scores hyperparameter tuning with random search	156
C.2	Scores $\overline{\Delta T}$	157
D.1	Standard deviation for the noise	160

Acronyms

- $C_{s,\text{avg}}$ Average concentration of lithium-ions in the solid electrode material (cathode and anode). 12
- $C_{s,\text{max}}$ Maximum concentration of lithium-ions in the solid electrode material (cathode and anode). 12
- C_r Residual capacity. xxvi, 14
- C_t Total capacity. xxvii, 14, 22, 51, 60
- P Power. 17
- Q^{rev} Reversible heat. xxvi, 8–10
- $V_h(T)$ Highest rated voltage of a battery, specified by the manufacturer. 4, 14, 66, 69
- $V_l(T)$ cutoff voltage. The lowest rated voltage of a battery, specified by the manufacturer. 4, 14, 60, 68
- W^{rev} Reversible work. xxvi, 8–10
- ΔT Temperature difference between battery and its surrounding. xiv, 81, 82, 97, 98
- $\Delta \bar{h}$ Change in entropy. 8
- Δt Time frame when current is drawn from the battery. 10, 18
- η Overpotential. 9
- γ Charge/discharge efficiency. 22, 56, 72
- $\overline{\Delta T}$ Temperature difference between battery and its surrounding averaged over time.. xiv, xvii, 82, 97–99
- ψ Ratio of the average lithium-ion concentration left in the battery and the total maximum possible concentration. 12, 13

i Current. 9, 10, 17, 18, 22, 46

r Internal resistance of a battery.. 9, 10, 46

E^{cell} Cell potential measured in volts. It is the same as . 9, 10, 17, 18

Ah Ampere-hour. 14, 16–19

CCCV Constant Current Constant Voltage. xiv, xvii, xxv, 14, 60, 61, 68, 69, 71–73, 77, 82, 90, 91, 93, 156

DoD Depth of Discharge. 14, 21, 60

EV Electrical Vehicle. 2–4, 11, 17, 20, 21, 23, 24, 44, 45, 47, 48, 52, 57, 59, 64, 66, 67, 72, 93, 104, 106

FNN Forward Neural Network. xiii, xiv, xxvi, 27, 28, 30, 32, 46–48, 51, 52, 54, 57, 58, 61, 62, 73–75, 77, 78, 80, 86, 92, 97, 100, 101, 104, 105, 155

GRU Gated Recurrent Unit. 25, 38, 39, 49, 53

HWFET HighWay Fuel Economy Test. 53, 54, 65, 76, 77, 79

k Time step. 13, 22

LA92 California Unified driving schedule. 44, 53, 65, 79, 81

LCO Lithium Cobalt Oxide. 8, 11, 12, 65, 68, 69, 105

LFP Lithium Iron Phosphorous. 11, 12, 46, 53, 56

LiB Lithium-ion Battery. xiii, 2–4, 7, 11, 12, 15, 16, 22, 26, 42, 43, 47, 48, 54, 57, 59, 66, 71, 72, 81, 104, 105, 153

LiPo Lithium Polymer. 12, 53, 56, 60

LMO Lithium Manganese Oxide. 11, 12

LSTM Long Short Term Memory. 25, 38, 39, 49, 50, 53, 95

MAE Mean Average Error. 34, 50–54, 56, 57, 90, 94, 95, 97, 98, 103, 104, 106

MAX error Maximum error. 50, 61–63, 93, 94, 97, 98, 100, 104, 105

ML Machine Learning. xiii, 2–4, 23–28, 31, 32, 35, 37, 43, 52, 54–58, 63, 71, 73, 77, 86, 92, 106

MSE Mean Square Error. 34, 74, 79

- NCA** Lithium Nickel Cobalt Aluminium Oxide. 11, 12, 52, 60
- NMC** Lithium Nickel Manganese Cobalt Oxide. xiii, 11, 12, 17–19, 47, 52, 53, 56, 60
- NN** Neural Network. xiii, xxvi, 3–5, 24–28, 30–32, 34, 36–38, 40, 42, 46, 51–54, 56, 59, 65, 67, 74, 76, 89, 96, 103–106
- NYCC** New York City Cycle. 44, 65
- OCV** Open Circuit Voltage. xxvi, 2, 10, 12, 14, 21–23
- RBFNN** Radial Basis Function Neural Network. xiii, 40, 41, 50, 51
- ReLU** Rectified Linear Unit. xiv, 33, 78, 101, 104
- RMSE** Root Mean Square Error. xiv, 34, 47, 50, 52, 53, 56, 57, 78, 79, 85–88, 90–100, 103, 104, 106
- RNARX** Recurrent Nonlinear AutoRegressive with Exogenous inputs. 50, 53
- RNN** Recurrent Neural Network. xiii, xxvi, 25, 27, 37–40, 47, 49, 50, 53, 57, 73, 82, 83, 95, 100, 101, 105
- SEI** Solid Electrolyte Interface. 15
- SoC** State of Charge. v, xiii, xiv, xxvii, 2–4, 7–10, 12–24, 26, 27, 31, 32, 37, 42, 43, 46–49, 51–62, 66–68, 71–75, 77–82, 85, 86, 88, 92–101, 103–106, 152, 161, 162
- SoH** State of Health. 4, 15, 52, 54, 64, 65, 69, 152, 153
- UDDS** Urban Dynamometer Driving Scheduled. xiv, 44, 53, 65, 79, 96
- US06** Urban driving cycle called US06. xiv, 44, 49, 52, 53, 65, 76, 77, 79, 96, 98, 99
- Wh** Watt-hour. 17–19
- WLTC** Worldwide Harmonized Light Vehicles Test Cycle. 44, 65

Glossary

batch A subset of a training or test data set is called a batch.. 35

C-rate The rate a battery is charged and discharged. A charge c-rate of 1C implies that the battery is charged from a fully discharged to a fully charged state in one hour. At 0.5C the charging time is two hours, and at 2C it is half an hour. 7, 10, 12, 15–18, 22

capacity Is defined as the product between the current that can be drawn from a battery before the voltage drops to a certain value, and the time frame that it is drawn. It is measured in ampere hours. 10

Constant Current Constant Voltage Is a cycling scheme used to charge or discharge a battery to a certain voltage. If charging is required to reach the desired voltage, the battery is charged up to a maximum voltage by a constant current. Then the current is step-wise reduced to yield a constant voltage. The charging is terminated when a predefined lower current is reached. In this report Constant Current Constant Voltage it is referred to by its acronym CCCV. 14

cutoff voltage Lower voltage limit specified by the battery manufacturer. xxi, 13, 16, 60, 69, 71

cycling Refers to a battery undergoing charging and discharging operations. 15

deintercalate Remove something (in this report: Lithium) between layers in a structure. It is the reverse action of intercalation. 8

discharge capacity It is the capacity defined by the quantity of charge removed at a constant C-rate from fully charged and until the terminal voltage reaches the manufacturer specified lower voltage limit. It is measured in ampere hours. 16, 17, 19

drive cycle Standard electrical vehicle driving patterns usually given by a time-velocity relationship, but it can also be a time-power relationship. The latter is the case in this report is not stated otherwise.. 43–45

epoch When all the training data has been fed once through the model. xxvi, 26, 31, 32

Forward Neural Networks Subclass of Neural Networks known as the simplest type. Abbreviated as FNN. 27

fully charged Is when the battery's OCV = $V_h(T)$ where $V_h(T)$ is the highest voltage specified by the manufacturer at a certain temperature T . 14, 93

fully discharged Is when the battery's OCV = $V_l(T)$ where $V_l(T)$ is the lowest voltage specified by the manufacturer at a certain temperature T . 14

hyperparameter Parameters external to the model whose value cannot be estimated from data, but has to be decided by the scientist. Examples of hyperparameters are the number of nodes, the number of layers, the learning rate, type of activation function(s), optimizer and dropout rate.. xiv, xxvii, 26, 41, 46, 75–77, 79, 82

hysteresis For a lithium-ion battery, hysteresis is the difference between the lithiation and delithiation. It can be observed by a different voltage profile for charging and discharging. 14

intercalate Insert something (in this report: Lithium) between layers in a structure. It's the reverse action of deintercalation. 15

Neural Networks neural network is a collective term used for machine learning algorithms inspired by the neural network in our human brains. In this thesis the abbreviation NN will be used. 3, 27

overfitting Is when a model is losing generalization due to training for too many epochs. It then becomes specialized on the training data, but performs poorly when validated and tested. xxvi, 27, 30–32, 48, 58, 75

primary battery Refers to batteries that cannot be recharged. 7

Recurrent Neural Networks Subclass of Neural Networks, distinguished by having a memory such that the information inputted and outputted from the network is passed forward in time. Abbreviated as RNN. 27

regularization technique to prevent overfitting. 32, 48

residual capacity C_r is theoretically the amount of charge in the cathode that can be removed. Practically it is the amount of energy that can be removed when discharging the battery from its current state until a fully discharged state. It is measured in ampere hours. 14, 16, 19

reversible heat Q^{rev} is reversible due to the fact that the heat is used during charging, and obtained during discharging. $Q^{\text{rev}} = T\Delta\bar{s}$. 8

reversible work W^{rev} is the maximum potential energy in a battery cell measured in volts. . 8

- search space** Predefined subspace of hyperparameter values which defines the boundaries of the search when tuning the hyperparameters. . 41
- secondary battery** refers to batteries that can be recharged. 7
- self-discharge** The capacity is reduced without a connection to a load. Self-discharge is due to internal chemical reactions. 11, 16, 20, 22
- State of Charge** Referred to as SoC in this report. It is a measure of the remaining useful energy left in the battery with respect to the total energy. It is often given as a percentage.. 2
- state of health** A measure of the degradation of a battery. It is mathematically defined as the ratio between the present total capacity and the rated total capacity of a new battery. 15
- supervised machine learning** Subgroup of machine learning where the input features and target data is given to the model. 26
- terminal voltage** Is the voltage at the terminals of an electrical device (like a battery). . xxii, 10, 12, 16, 17, 24, 60, 66, 69
- total capacity** C_t is the measure of the total energy in a battery that can be drawn from a fully charged state to a fully discharged state. It is measured in ampere hours. 14–16, 19
- unsupervised machine learning** Subgroup of machine learning where the only input features are given to the model. The model has to develop purely based on patterns in the data. 26

Introduction

1.1 Motivation

The world is developing towards a cleaner energy future with reduced green house gas emissions. This is partly due to continuous increase in renewable energy installations through the last decade [1], especially conversion technologies from hydro, sun and wind. As we rely less on fossil sources, the fundamental challenges of renewable due to their intermittent nature are more dominant and should be taken notice of. The electricity generated from wind is directly dependent of the wind speed being within the operating range of the turbine, while photovoltaic panels only provide electricity during the daytime when the clouds do not block the rays. Since renewable energy is highly dependent on the weather, electricity production cannot be based on demands, but has to be generated when it is suitable weather conditions. Figure 1.1 shows the weather dependent irregularities of the net electricity load for California with a lot of its electricity from sun power. Here, the net electricity load is defined by the consumed electricity minus the production of electricity by renewable resources. The renewable energy contributes to a duck shaped graph. Today, approximately 28% of the world's generated electricity comes from renewable sources, where the definition of renewable energy sources is restricted to solar, wind, ocean, hydropower, biomass, geothermal resources, biofuels and hydrogen¹ [3] [4]. Furthermore, the number of renewable energy installations is increasing rapidly [5]. This results in a trending behaviour towards a diverging

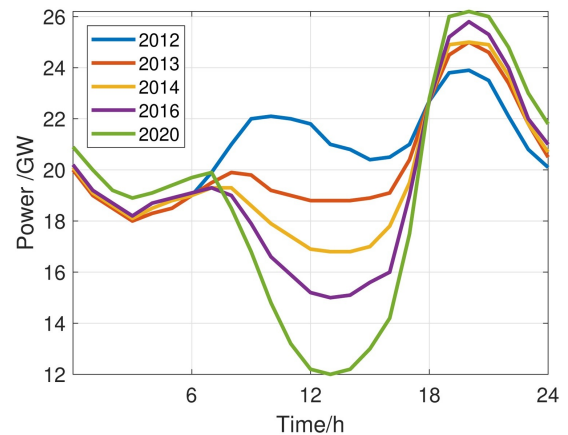


Figure 1.1: Net electricity load requirement for California where 2020 is a forecast. The net electricity load is defined by the consumed electricity minus the production of electricity by renewable resources. The figure is used with permission from [2].

¹Biofuel and hydrogen are only included as a renewable resource if they have been derived from a renewable resources [3].

gap between the consumed electricity and produced electricity. Only few years ago, experts believed that renewable energy would not be scalable due to its unpredictability and unreliable frequencies [6]. Nevertheless, today we know better, and the enabling technology to flatten out the duck curve is energy storage.

Several different approaches to store energy exist, including fly wheels, compressed air storage, thermal storage, hydro pumping and batteries. The latter has the advantage of having a large power density, relatively good energy density and is easily scaled, compared to the aforementioned energy storages. Another desirable characteristic of batteries is that they are portable and have a low self-discharge rate. Consequently, batteries are integrated parts of portable applications like cell phones, tablets, gps maps, laptops and Electrical Vehicles (EVs). Rechargeable batteries, especially Lithium-ion Batteries (LiBs), are fundamental for the development of these electrical applications. Knowing the State of Charge (SoC) is essential to obtain a smart control strategy to save limited energy, maintain safe operations by preventing the battery from over-charge, over-discharge and extending the battery's lifetime [7]. Moreover, in the perspective of a user, it is important to know if the battery will have enough charge to carry out tasks like rounding up an online meeting, navigating a boat safely to shore, or driving home without getting stranded.

SoC is a measure of the remaining capacity of a battery with respect to a fully charged battery. In a gasoline vehicle one can directly measure the level of the fuel in the tank to know how much energy is left. This is not straightforward with a battery EV. Since the energy in batteries is stored as chemical compounds that undergo an electrochemical reaction when converted into electricity, estimating the SoC is difficult due to the changes occurring when shifting between energy forms. There are several factors that have to be considered. For instance, with lower ambient temperatures the internal resistance increases, and a drop in the SoC can be experienced. The amount of energy available in a fully charged battery decreases with the cycle number through ageing causing performance degradation that influences the SoC calculation [8]. The SoC estimation is highly nonlinear and is dependent on time-varying variables and the cycle number, as well as the cell chemistry.

The first documentation of a SoC estimation was done in 1938 by Heyer [9], using direct measurements of the battery voltage. Since then, many different SoC estimation methods have been developed like Coulomb counting and Open Circuit Voltage (OCV) methods. Today, data driven Machine Learning (ML) models have proved to yield accurate results with the advantage of working well with limited physical and chemical insights. This implies that carefully adjusted parameters for modelling each specific battery are not required, potentially reducing the engineering to obtain an efficient SoC estimation model. Conventional methods theoretically perform better than ML from a statistical point of view if appropriately designed, but are unsuitable for real-time applications. Also, as the computational power and the amount of big data increases, SoC estimation models are advancing towards data driven algorithms such as ML [10].

ML is a branch of artificial intelligence and can be a powerful tool to recognize patterns in large data sets without requiring knowledge of the physics of the problem. Different ML approaches are popular in fields like voice recognition, image classification, weather prediction, and data

mining in general. ML has recently rocketed in popularity due to the development of better GPUs enabling faster calculations and the handling of large amounts of data. In turn, this has enhanced the learning capability, accuracy and generalization performance, making ML adequate to address the complex non-linear relationship of LiBs' SoC.

There are four main categories of ML, namely supervised, unsupervised, semi-supervised and reinforcement learning. In supervised ML, a model is trained by predicting an output from data characteristic for the problem, often referred to as input features. Then the model calculates the error between the predicted output and the target output (label). Successively, the parameters of the model are tuned until an appropriate cost function has converged. Supervised learning models are provided with both input features and target labels. Both classification and regression problems commonly utilize supervised learning where the target label is a class or a number, respectively.

For SoC estimation, supervised learning is commonly used to learn a general rule to map inputs into the desired output, meaning that physical battery variables (*e.g.*, current, voltage, and temperature) can be mapped to the SoC. Since numerical input (battery measurements) are mapped to a numerical prediction (SoC), it is a regression problem. Many regression algorithms exist and in this thesis Neural Networks NN will be used. A lot of research has previously been done to predict SoC with NN. Nevertheless, further research is necessary to improve accuracy, robustness and efficiency of the method. Therefore, this project thesis will be focusing on NN to estimate the SoC of LiBs.

1.2 Problem Definition

Since the SoC is required for a broad spectrum of applications using LiBs, the problem is narrowed down and restricted to batteries used in EVs. EVs were chosen based on several reasons. Firstly, the amount of work already done on estimating SoC for EVs is large. This advantage is related to the practicality of comparing different models. If the models are tested on similar data sets (*e.g.*, battery cycles of EV), the results in terms of robustness and accuracy are easier to compare. Secondly, the amount of high quality public data sets on EV battery cycles are greater than for other applications using batteries. Therefore, the time consuming work of generating data can be reduced in this project for the training and testing of the ML model. Furthermore, the EVs power profiles are rapidly changing due to acceleration and deceleration, making it more challenging to estimate the SoC for EVs than for applications drawing constant power. Consequently, if adequate results can be obtained for SoC estimations in EV's, satisfactory results are also possible to obtain for other SoC applications. Lastly, since the majority of SoC estimations using ML are done in the field of EV, it is likely where the state-of-the-art ML models to estimate SoC are obtained. Accordingly, when researching the best ML models to estimate SoC, EV is an adequate place to start.

The aim of this thesis is threefold. The first aim is to investigate the best practice for State of Charge (SoC) estimation on Lithium-ion Batteries (LiBs) by Neural Networks (NNs). This will

be done by training NN models on large battery data sets created to simulate EVs cycled in the laboratory. Then, different features and manipulation of the data will be examined to lower the prediction errors for SoC. In addition, a literature study is conducted to get an overview of the state-of-the-art with its associated challenges.

The second aim is to determine what Machine Learning (ML) techniques that are effective in this field. Three techniques were tried, random search to tune the network, augmented data to create a larger training data set, and transfer learning to increase prediction accuracy with less battery data.

The last aim is to investigate the potential of SoC estimation with aged cells. Aged cells have typically a more complex behavior compared to new cells. This is due to several different aging mechanisms and it was found interesting to explore if an NN is able to learn the aging mechanisms. Therefore, three batteries at various State of Health (SoH) were cycled. Due to time limitations an ML model was not developed. However, the battery data required to execute the investigation was obtained and ready to be used in further work.

The following limitations and simplifications were done:

- The SoC estimation is based on Equation 2.11 and not by the km-range which is the current industry standard. The reason is twofold. Most importantly, no real-world battery data was captured². Secondly, the km-range would be based on mathematical models which would be hard to verify. Therefore, the error scores are more accurately calculated by the percentage SoC than a km-range.
- This project is a proof of concept, rather than a water proof SoC estimator for EVs. Therefore, this thesis is only to research NNs potential to learn the operational principles of LiBs.
- Since real, accurate driving data is absent, laboratory data is used to construct and test the model. The laboratory data is constructed to simulate the operation of an EV battery.
- The SoC is only estimated for one cell and not a whole battery pack.
- The definition of a fully charged and discharge battery is dependent on the battery manufacture specified voltage limits, $V_l(T)$ and $V_h(T)$.

1.3 Outline of the Thesis

To obtain the best practise on how to estimate the SoC on LiBs by ML, an understanding of the operation of LiBs and ML is required. Therefore, a broad introduction to LiBs is given in Chapter 2 and the theory of ML with NN regression methods in focus is found in Chapter 3. In Chapter 4, description of data sets and previous research on the topic is found in a literature review. A comparison is done based on temperature conditions, battery types, the choice on training and testing data and the stated accuracy. The data acquisition and preprocessing of the

²Some real-world data was found from a collaboration with a research group from Beijing Institute of Technology. However, according to Prof. Zhang which is a member of the research group, the accuracy of the measurements was not of good enough quality or amount to be utilized for constructing a ML model.

data is explained in Chapter 5, followed by the method for modeling the NN in Chapter 6. The results and discussion is Chapter 7.

Theory on Lithium-ion Batteries

This Chapter covers some fundamental electrochemical working principles and terminology of Lithium-ion Battery (LiB)s relevant for an understanding of the factors influencing the State of Charge (SoC). This includes the potential energy, capacity, C-rate, and the different LiB types. Then, SoC is explored along with different approaches to estimate it. Furthermore, it should be noted that, if not written otherwise, the definitions and equations in this project are with respect to LiBs and that other principles can apply for other battery chemistries.

2.1 Cell Structure and Working Principle

Batteries can be divided in two main groups; primary and secondary batteries. The difference is that a battery belonging to the former is not rechargeable, while the latter is. LiBs exist as both primary and secondary batteries, but this thesis will only address rechargeable LiBs.

All LiB cells are composed of some essential components, including an anode and cathode, a separator, an electrolyte, and two current collectors. During discharge, the lithium atoms are oxidized at the anode into Li^+ and move to the cathode through the separator. The electrons move through the current collector and further through the external circuit as can be observed in Figure 2.1. The opposite reaction takes place during recharge.

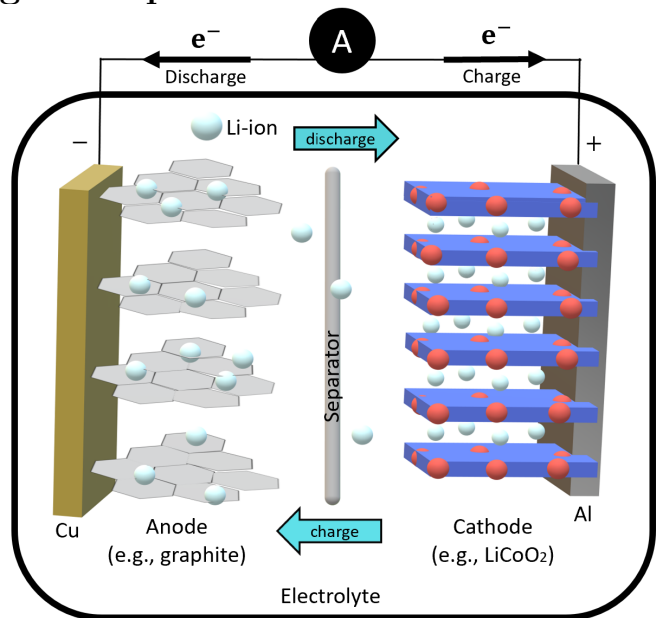


Figure 2.1: Electricity generation: lithium-ions and electrons move from the anode to the cathode. The ions move through the separator, while the electrons move through the external circuit.

ging. When fully discharged, a considerable fraction of the lithium-ions reside in the cathode as illustrated by Figure 2.2.

Not all lithium atoms are utilized during operation. The cathode is a lithium-metal-oxide composition [11], implying that lithium is part of the structure. If too many lithium atoms are deintercalated when charging, the cathode structure would collapse. Therefore, a fully charged cell actually has more potential chemical energy stored in the cathode than utilized. A rule of thumb is that half (or more) of the lithium-ions are contributing to producing electricity in a cycle, as illustrated by Figure 2.2. Consequently, the definition of a fully charged battery is the energy that can be utilized without a conceivable safety risk and irreversible losses in the cell.

The electrochemical reaction is driven by the potential difference between the two electrodes, as explained in more detail in the next section. In order for the reaction to take place, it is essential to choose electrode materials with high potential difference. The potential of a fully charged battery depends on the active materials in the electrode and the cell structure. An example of an overall chemical reaction for a Lithium Cobalt Oxide (LCO) is as follows:

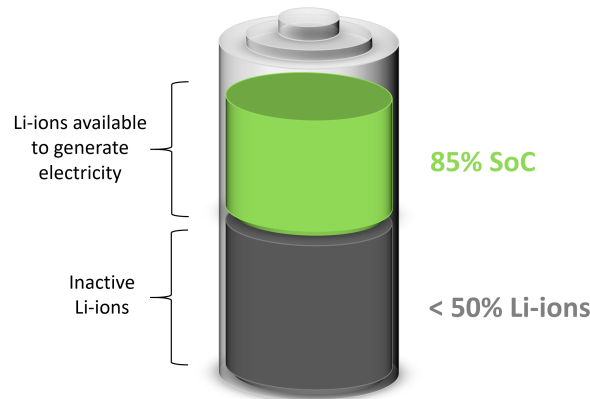
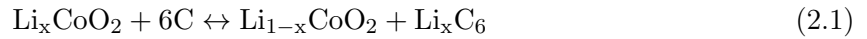


Figure 2.2: Illustrating that the theoretically amount of available lithium-ions cannot be utilized. Therefore, a 85% SoC does not mean that 85% of the lithium is in the anode. If all the lithium was removed from the cathode during charging, the cathode structure would collapse.

2.2 Voltage Potential and Losses in Lithium-ion Batteries

The amount of potential energy available to produce electricity in a battery can be analyzed by the reaction work or, more precisely, the reaction enthalpy ($\Delta\bar{h}$). The reaction enthalpy is the sum of the reversible heat (Q^{rev}) and the maximum potential work, also known as the reversible work (W^{rev}). This relationship is stated as:

$$\Delta\bar{h} = Q^{\text{rev}} + W^{\text{rev}} \text{ [J]} \quad (2.2)$$

Q^{rev} is equal to the temperature multiplied by the change in entropy ($T\Delta\bar{s}$). This heat is reversible because the change in entropy is reversed when the chemical reaction in the cell happens in the opposite direction. In [12], the authors state a relationship between a high SoC and large entropy change, implying an increased reversible heat change at higher SoC. In general the reversible heat from entropy change functions as a heat sink during charging, cooling the cell, while it is a heat source during discharge. There is research [13] showing that the total entropy change is the sum of larger entropy changes at each electrode with opposite signs, which complicated this simplified description. However, this is a topic of its own, which is considered beyond the scope of this thesis.

Due to losses, all W^{rev} cannot be converted into electricity. The difference between W^{rev} and losses yields the available work to produce electricity. The available work is often stated in terms of the cell potential E^{cell} . The relationship between losses and cell potential can be observed by Figure 2.3, or by the following equation:

$$E_{\text{discharging}}^{\text{rev}} = E^{\text{cell}} - r \cdot i - \eta \text{ [V]} \quad (2.3)$$

$$E_{\text{charging}}^{\text{rev}} = E^{\text{cell}} + r \cdot i + \eta \text{ [V]} \quad (2.4)$$

Here, r is the internal resistance of the battery and i is the current drawn from the battery. $r \cdot i$ is known as Ohmic losses, whereas η represents losses related to overpotential.

Ohmic losses occur due to the resistance of the movement of ions in the electrolyte, where the friction between ions and the electrolyte generates irreversible heat. The magnitude is calculated from Ohm's law $\Delta E_{\text{loss Ohmic}} = r \cdot i$, where ΔE is the voltage drop. Here, r is a function of both current and SoC, where high currents increases the internal resistance, as well as a high or a low SoC [14].

The second irreversible loss, η , is caused by two mechanisms. Most important is the friction due to electrons transferred between the reactant and the electrode. The other mechanism is referred to as the concentration polarization overpotential. It occurs due to friction against free movement when reactants diffuse towards the current collector.

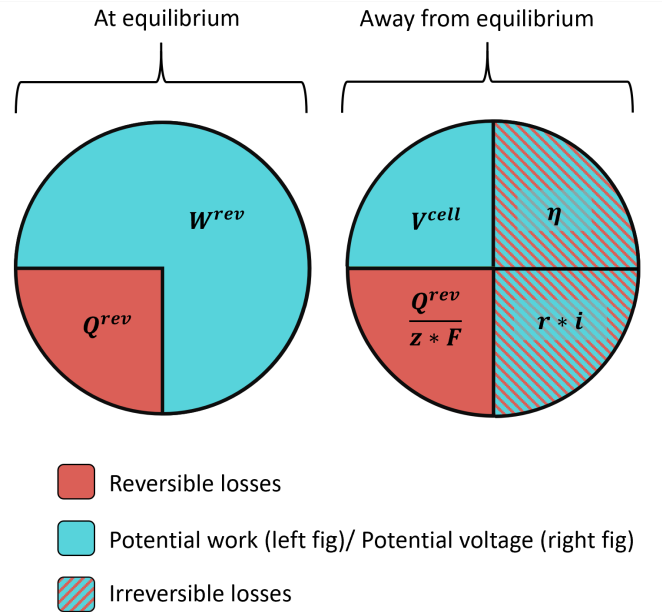


Figure 2.3: Schematic of the components of the thermodynamics and the relationship between reversible heat $Q^{\text{rev}} = T\Delta\bar{s}$, potential electric work $W^{\text{rev}} + Q^{\text{rev}}$, the available potential E^{cell} , and the irreversibly lost electric energy for a spontaneous electrochemical cell. The irreversibly lost energy is from Ohmic losses and losses related to overpotential denoted $r \cdot i$ and η , respectively. The size of the fractions changes with factors like the cell temperature and current drawn.

Theoretically, E^{cell} is the potential difference between the cathode and anode. Practically, E^{cell} is the voltage that can be measured over the two terminals of the battery using a voltmeter and is therefore also known as the terminal voltage.

W^{rev} is equal to E^{cell} at electrochemical equilibrium when no current is drawn and no irreversible losses are present. However, W^{rev} only occurs at electrochemical equilibrium where only a reversible heat loss Q^{rev} is present. Q^{rev} is linearly proportional to the temperature of the cell, indicating that a higher temperature yields a higher reversible loss. Since W^{rev} is measured at equilibrium and with an open circuit, it is logically known as the Open Circuit Voltage (OCV), representing the maximum voltage available.

When measuring the voltage potential at equilibrium, no net current can be drawn from the cell [15]. The reason being that the terminal voltage fluctuates when drawing current and keeps fluctuating even after disconnecting the load. Therefore, the battery cell has to rest for a long time (up to several hours) before the voltage is stable and the Open Circuit Voltage (OCV) can be measured. A lot of research has been put into estimating the OCV when the battery is operated due to a strong relationship to SoC, but with unsatisfying results for accuracy.

Away from equilibrium, when current is drawn from the battery, there are irreversible losses and only E^{cell} can be measured. Like the reversible losses, the irreversible losses contribute to heat generation in the cell and consists mainly of the two losses described above; Ohmic losses $r \cdot i$ and losses related to overpotential η .

2.3 Capacity and C-rate

Simplified, the battery capacity is the energy that can be drawn from a battery. It is defined as the product of the current that can be drawn from a battery before the voltage drops below a certain limit, and the time frame (Δt) in which that current (i) is drawn [16] as stated by Equation 2.5:

$$C = i \cdot \Delta t \text{ [Ah]} \quad (2.5)$$

where C is the capacity. If the current is not constant, the expression have to be integrated with respect to time rather than multiplied with Δt .

The capacity is a result of the cell's chemistry, geometry, and amount of active material [17]. However, a battery can consist of several cells and the capacity can be increased by connecting two or more cells. If a higher voltage than the cell can deliver alone is desired, the cells can be connected in series. If a higher current is required, the cells can be connected in parallel. This configuration of clustered cells is referred to as a battery module. Several connected modules becomes a battery pack, which again is beneficial due to an increased capacity and maximum power.

Whereas capacity measures energy stored in the battery, the C-rate measures how fast energy is transferred. In other words, C-rate is the charge and discharge rate. It is calculated by the

Table 2.1: A comparison of important properties of popular LiBs. The data is obtained from [18].

	LCO	NMC	NCA	LMO	LFP
Specific energy (Wh/kg)	120-150	140-180	80-220	105-120	80-130
Specific Power (W/kg)	600	500-3000	1500-1900	1000	1400-2400
Nominal voltage (V)	3.6-3.8	3.6-3.7	3.6	3.8	3.2-3.3
Cycle life (nr of cycles)	>700	1000-4000	>1000	>500	1000-2000
Self-discharge (% per month)	1-5%	1%	2-10%	5%	<1%

following equation:

$$\text{C-rate} = \frac{\text{Current}}{\text{Capacity/1hr}} \quad (2.6)$$

For instance, at an 1 C discharge rate, a battery is discharged in one hour. At 2 C, a battery is discharged in 30 minutes, while at 0.5 C a battery is discharged in two hours.

2.4 LiB Types

Several different types of LiBs exist, both with respect to chemistry and geometry. The most common geometries are cylindrical, pouch, and prismatic cells. The heat generation is different for the different geometries. Still, it is the chemical composition that has the greatest impact on the performance with respect to aging cycles, power, energy, temperature effects, and voltage characteristics.

Typically for LiBs, the anode chemistry consists of a graphite anode, whereas the cathode material varies. Therefore, LiBs are usually classified by the cathode chemistry. The first commercialized LiB was the LCO, which still constitutes a large market share. However, a popular battery today is the NMC, where the abbreviation represents nickel, manganese, and cobalt¹. Both NCA consisting of nickel, cobalt, and aluminium², and NMC have superior performance in terms of long life cycles and high energy capacity [19], therefore, they are widely employed in EVs. However, both cobalt and nickel are limited resources, and they consequently have a high cost. By simply removing the expensive nickel and cobalt, a battery named LMO is obtained, which only consists of lithium, manganese, and oxygen. This battery has a lower specific energy capacity and shorter lifespan than NMC and NCA [19].

One more well-established chemistry is LFP, which is distinguished from other LiBs by consisting of Iron and Phosphorus³. LFP is known to be of high safety and has high specific power, but a low energy density compared to NMC and NCA. Furthermore, LFP has a longer lifetime than

¹NMC also consist of lithium and oxygen.

²NCA also consist of lithium and oxygen.

³LFP also consist of lithium and the phosphorus-oxygen binding phosphate (PO₄).

NMC and NCA, but LCO has the shortest [19]. A further comparison of properties is found in Table 2.1. In addition, the discharge profile of two battery chemistries, LFP and NMC, with OCV plotted against time, is found in Figure 2.4. Due to the dissimilar discharging behaviour of the different chemistries, the SoC behaves dissimilar and SoC models has to take the variances into account.

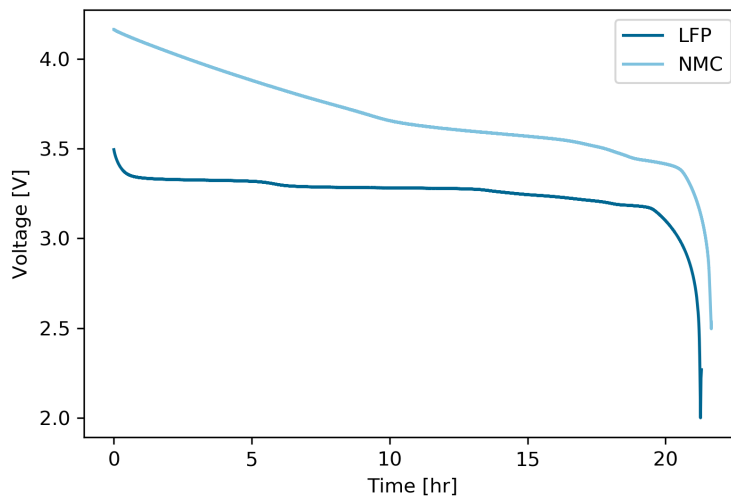


Figure 2.4: Discharge profile with a C-rate of $\sim C/22$. With this low C-rate the terminal voltage is an approximation of the OCV. The LiB data used for this plot is from [20].

Another Lithium-based battery is the Lithium Polymer (LiPo). The name represents that the electrolyte is a polymer electrolyte instead of a liquid electrolyte. The battery has in general higher specific energy than LiBs with liquid electrolyte, but on the downside, it has the shortest lifetime [21–23]. Lithium Polymer (LiPo) cells consist of classical cathode materials (*e.g.*, LCO, NMC, NCA, LFP, and LMO).

2.5 State of Charge Defined

SoC is a measure of the remaining energy in a battery. It can be observed in our daily life on the display of electrical applications as a battery symbol that is fully or partly filled, representing the percentage of energy left in the battery as shown in Figure 2.5.

Physically, the amount of energy left in a battery (ψ) is defined by the averaged concentration of lithium-ions in the anode ($C_{s,avg}$) over the maximum possible concentration in the solid electrode material (anode and cathode) ($C_{s,max}$) yielding:

$$\psi = \frac{C_{s,avg}}{C_{s,max}} \quad (2.7)$$

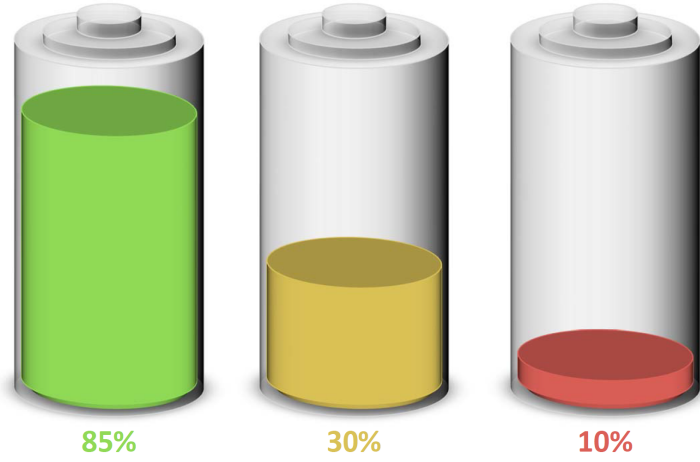


Figure 2.5: Illustration commonly associated with SoC where the color and amount of color represent the remaining energy of the battery.

Theoretically, $\psi = 0$ and $\psi = 100$ is possible. However, it is not feasible since taking out too many lithium-ions from the cathode will destroy the structure and thereby increase degradation. Therefore, some of the lithium-ions are left in the cathode when the battery has a 100% SoC. The same is generally true for the anode, implying that the SoC defines a window of operation of a battery. Moreover, this can be mathematically explained by a ψ window:

$$\psi_{0\%} > 0 \quad (2.8)$$

$$\psi_{100\%} < 1 \quad (2.9)$$

Here, $\psi_{0\%}$ and $\psi_{100\%}$ represents the amount of lithium-ions in the anode when the battery manufacturer has defined the SoC to be 0% and 100%, respectively.

The SoC can now be defined by a ratio of the defined ψ windows as follows:

$$\text{SoC} = \frac{\psi_k - \psi_{0\%}}{\psi_{100\%} - \psi_{0\%}} \quad (2.10)$$

Where ψ_k is the amount of energy left in the battery at the present time step k .

This definition of the SoC is theoretically correct, but it is not practically possible for a battery management system to calculate ψ since the concentration C_s cannot directly be measured. Therefore a definition of SoC not directly based on the concentration is required.

Practically, a fully charged and discharged battery is defined based on the highest voltage V_h and lowest voltage V_l , respectively, where the lowest voltage is referred to as the cutoff voltage. Both voltage limits are specified by the manufacturer at a certain temperature (T) [24]. The

definitions yields as follows:

Definition 1.1. Fully charged is when the battery's OCV = $V_h(T)$

Definition 1.2. Fully discharged is when the battery's OCV = $V_l(T)$

Physically, a fully charged battery is obtained by charging with a Constant Current Constant Voltage (CCCV) charge. The CCCV procedure is performed by first charging the battery with a constant current until it reaches $V_h(T)$. Then the current is step-wise reduced while the voltage remains constant until a certain lower limit for the current is reached (*e.g.*, 0.01 A). The minimum voltage is measured in the same manner, only that the current is drawn out of, instead of into, the battery and the voltage limit is the minimums voltage $V_l(T)$.

A full charging cycle is defined as a cycle between 0% and 100% SoC [25]. Often, a battery is not fully charged and discharged. The battery might be charged up to 80% SoC and discharged down to 20% SoC, meaning only 60% of the total capacity is utilized during the cycle. A convenient term that captures this usage is Depth of Discharge (DoD). DoD is inversely defined from SoC, meaning a 0% DoD corresponds to a fully charged battery, whereas a 100% DoD corresponds to a fully discharged battery. Thus, the following relationship is applicable: $\text{DoD} = 100\% - \text{SoC}$ [25]. The DoD is also referred to as a window, such that an 80% DoD window can for instance mean a cycle from 90% to 10%, or from 100% to 20%.

A way to measure 0% and 100% SoC has now been explored, as well as the terminology for the DoD window. However, the challenge not addressed yet is how to calculate the ranges between 0% and 100% since the OCV cannot be measured when operating the battery. In addition, OCV deviates at the same SoC when charging and discharging [26]. This OCV off-set-effect is known as hysteresis. Due to this complex SoC vs OCV relationship, a further definition is required to calculate the whole SoC range. In order to define this range, two more definitions are adequate concerning capacity measures:

Definition 1.3. Total capacity is the measure of the Ah that can be drawn out of the battery from a fully charged state to a fully discharged state.

Definition 1.4. Residual capacity is the charge in the cathode that can be removed when discharging the battery from its current state until a fully discharged state. It is measured in Ah.

The SoC can now be defined precisely by the following ratio:

$$\text{SoC} = \frac{C_r [\text{Ah}]}{C_t [\text{Ah}]} \cdot 100 [\%] \quad (2.11)$$

where C_t and C_r the total capacity and residual capacity, respectively. Equation 2.11 yields an SoC definition as a function of two variables. It is a simple expression, but still complicated to estimate. Both of the parameters are changing with time. The residual capacity is strongly

related to charging and discharging and oscillates accordingly during the battery's lifetime. On the contrary, the latter is influenced by aging and decreases monotonously with the number of cycles, the operating condition, and by storing it [27]. In order to understand the challenges related to SoC estimation, further exploration of capacity is required.

2.6 Total Capacity

The total capacity is one of the two main parameters needed to define the SoC of a battery. Since the total capacity decline with the LiB's age, the capacity fade should be accounted for when calculating the present total capacity. The processes behind the permanent capacity fade are complex and due to irreversible losses mainly related to the loss of inventory, meaning lithium-ions are made passive. Mechanisms include cathode structure degradation, generation of passive films on the electrodes, lithium plating, and electrical isolation of active sites due to mechanical degradation [28].

Degradation of LiBs is catalyzed by using the battery, but also occurs naturally over time when solely storing it. The expected life-time of a battery purely due to natural degradation is referred to as calendar life. Likewise, the expected life-time of a battery due to cycling, meaning the battery undergoes charging and discharging operations, is referred to as cycling life. Neither the degradation due to cycling life, nor calendar life, can be obtained directly from measurements and monitoring. However, the degradation is numerically described by the state of health (SoH), defined as the ratio between the present total capacity and the rated total capacity of a new battery [29]:

$$\text{SoH} = \frac{\text{Present total capacity [Ah]}}{\text{Total capacity of a new battery [Ah]}} \quad (2.12)$$

Each part of the battery has its own degradation mechanism reducing the SoH. For instance, the anode suffers from lithium plating and the growth of the Solid Electrolyte Interface (SEI), which are the two most essential contributors to irreversible lithium-ion loss [30]. The SEI layer is created when cycling. This is beneficial for two reasons: it protects the anode from the electrolyte and allows the lithium to intercalate into the graphite anode. However, the SEI layer continues its growth yielding a resistive layer removing lithium from the cycling system [31].

Addressing the other significant phenomenon guilty of lithium loss in LiBs, lithium plating, is described as the formation of metallic lithium and situated outside the anode. When metallic lithium is formed, it can no longer contribute to the electrochemical process and the amount of active material is reduced. Low temperature and high C-rates accelerate the lithium plating [32], which again causes performance degradation and increase the potential safety risk for LiBs [33]. In addition, high temperatures boost lithium plating, especially if charging to a high SoC (above 90%) [34].

Summing up, the aging mechanisms are not easily measured or monitored, and the total capacity of a fully charged battery is therefore challenging to estimate. Despite these challenges, the total capacity has to be accounted for when estimating the SoC.

2.7 Residual Capacity

The residual capacity is the maximum amount of charge in the cathode that can be removed when discharging the battery from its current state until a fully discharged state. From the perspective of a user, there are two ways which the residual capacity is reduced; either by drawing current from the LiB, or by storing it for a longer period due to self-discharge. Self-discharge is when the battery's charge gradually reduces over time [35] and should be accounted for when the residual capacity is to be estimated over a larger time frame of days or weeks.

The other way to reduce the residual capacity is by discharging it. The magnitude (Ah) can be estimated by available battery measurements. If perfect measurements of the current could be obtained, the residual capacity would be directly found by integrating the current over time, directly yielding the residual capacity in Ah. However, measurements come with a downside, namely errors. Therefore, several independent measurements have to be included in the estimation of the residual capacity, including current, terminal voltage, and cell temperature.

Error accumulation directly affects the accuracy of the residual capacity estimation and thereby the SoC estimations. Most vital are the errors concerning the voltage and current measurements, where bias and noise error is typically the problem. The bias is from the deviation between the actual value and measured value. The degree of noise error depends on the environment and is therefore difficult to estimate. However, both the bias and the noise can to an extent be reduced by investing in high-precision sensors. Moreover, numerous filtering algorithms like Kalman filters can eliminate much of the noise. This is further described in Section 2.11.4.

2.8 Discharge Capacity

The discharge capacity is defined by the quantity of charge removed at a constant C-rate from fully charged and until the terminal voltage reaches the manufacturer specified lower voltage limit $V_l(T)$. Discharge capacity differs from total capacity in the sense that when $V_l(T)$ is reached, there might be more charge left in the battery. However, the battery system is usually designed to prevent the user from discharging when the terminal voltage reaches $V_l(T)$. This is due to faster degradation of the electrolyte at voltages lower than $V_l(T)$. Therefore, the whole total capacity is normally not utilized.

The physical explanation of why the terminal voltage is reached before the total capacity is utilized was described in Section 2.2. Briefly repeated, it is more frictional losses created at higher C-rates due to faster moving ions and electrons through the electrolyte and active material, respectively. Increased losses will in turn reduce the terminal voltage and therefore the cutoff voltage is encountered earlier with a high C-rate. Consequently, the discharge is stopped before reaching a 0% SoC for higher C-rates, whereas the total capacity is constant. Therefore, discharge capacity will always be smaller or equal to the total capacity. This phenomenon can be observed in Figure 2.6 looking at the difference between the plots of different line-style, but the same color.

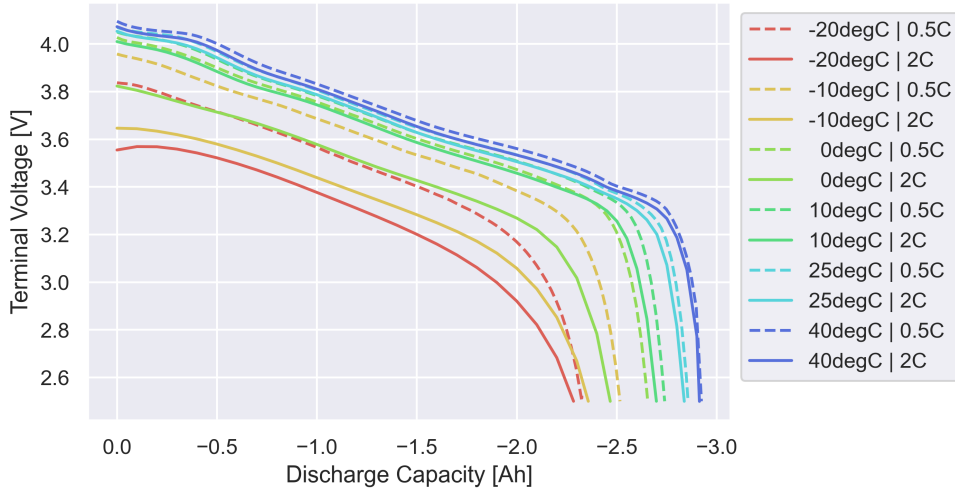


Figure 2.6: Discharge capacity for different temperature and c-rate. The data [36] used for plotting is from an NMC battery.

Another non-permanently, but still limiting factor to the discharge capacity, is temperature. A battery exposed to low external temperatures during operation will experience a rapid reduction in SoC compared to batteries operated at higher temperatures. This temporary capacity reduction becomes more significant when the temperature drops below 0°C and the trend continues when a further temperature reduction is present [37]. As an example, it was reported that the discharge capacity at -20°C was down to 60% of the discharge capacity at room temperature [38]. This phenomenon is caused by the conductivity in the electrolyte decreasing with decreasing temperatures, leading to a reduction in the Li-ions' diffusion rate [39],[40]. In simpler terms, the internal resistance increases as described in the above paragraph, causes the battery to produce less current. This will in turn decrease the terminal voltage of the battery. The relationship between power, voltage, and current can be obtained from a power-law, which states that the Power (P) is $P = E^{cell} \cdot i$. Therefore, for a battery to deliver a constant power with a reduction in the voltage an increased current is required. This phenomenon can be observed in Figure 2.6 by the difference between the lines of equal line styles.

The relations explained above between changes in discharge capacity is only a simplified explanation. Several other mechanisms also play a role in the reduced discharge capacity at lower temperatures and different C-rates including electrode thickness, cell design, separator porosity, and separator wetting properties [41].

2.8.1 Ah vs Wh as a Capacity Measure

One could argue that using Wh for the capacity measurement in Equation 2.11 would be more ideal than using Ah. The reason being that the amount of current drawn to yield a certain power is dependent on the SoC, so the Ah is a non-linear function of SoC during a full battery cycle. The SoC drops more rapidly for lower SoCs when SoC is a function of Ah. EVs require a certain

power to operate regardless of the SoC. So, from an user's perspective, the battery has limited energy and not a limited amount of current. In that respect, it would be better to use Wh as the capacity measurement. It can be calculated as follows:

$$P \cdot \Delta t = E^{cell} \cdot i \cdot \Delta t \text{ [Wh]} \quad (2.13)$$

where P is constant power, E^{cell} is the terminal voltage, i is the current drawn, and Δt the time frame that the current is drawn. If the power is not constant, the expressions have to be integrated with respect to time rather than multiplied with Δt .

The difference when estimating the SoC by Ah and Wh for a specific cycle can be observed in Figure 2.7. In this figure, the mean off-sett between using Ah and Wh as the capacity measurement, is above 2. This implies that there is a considerable deviation between the two estimation methods and that it is essential to reflect upon which method to choose.

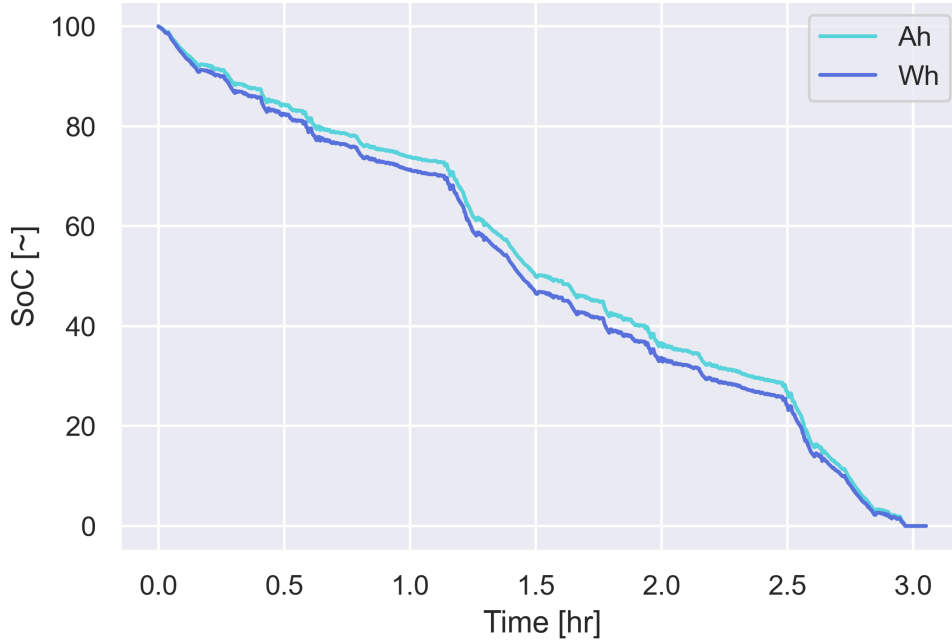


Figure 2.7: SoC when the capacity is given by both Ah and Wh. It is the same data [36] set used for estimating Ah and Wh (NMC battery).

However, the challenge with measuring capacity by Wh is that the total capacity is unknown in advance. In Figure 2.7, the total capacity in Wh is calculated after cycling the battery by approximately a full cycle, and thereby known. However, before cycling the battery, the operating conditions are usually unknown and Wh is prone to internal losses dependent on the C-rate, age and the temperature of the cell. Further, the user's driving patterns for acceleration and deceleration, use of air-condition and heating, also impacts the total Wh of the battery. This is also true for Ah as can be seen in Figure 2.6. However, Ah is more constant with respect to the C-rate and temperature than Wh. In Figure 2.8 it can be seen that the Ah deviates with 20% from it's

minimum and maximum capacity being discharging with 2C on a -20°C and discharging with 0.5C at 45°C , repetitively. For Wh capacity measurement, this deviation is 25%. Therefore, Ah will be used in this thesis to define the capacity when estimating the SoC.

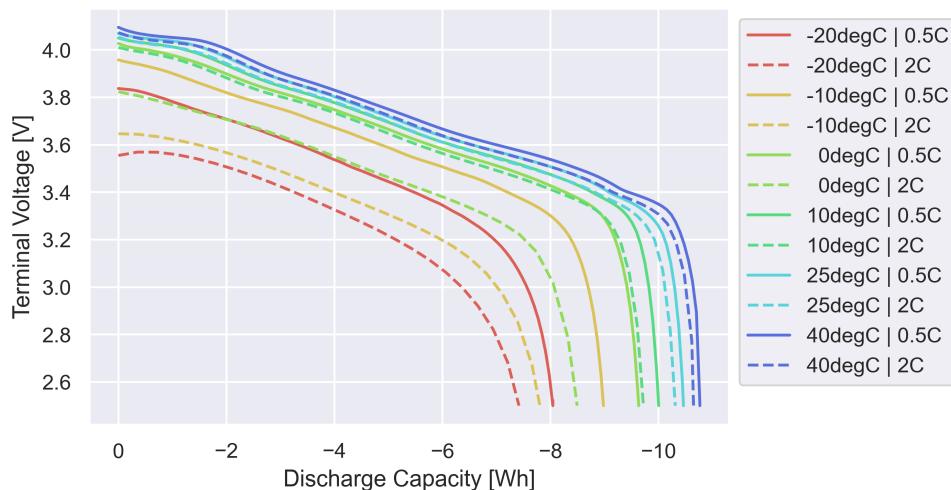


Figure 2.8: Discharge capacity when measured in Wh. The data [36] used for plotting is from an NMC battery.

To summarize, the definition of SoC is given in Equation 2.11 with Ah, but have some limitations and challenges:

- The discharge capacity is smaller than the total capacity, and the SoC will therefore not be zero unless discharged by a small current and at a preferable temperature (*e.g.*, room temperature). However, a small current yields a low power, unfeasible for electrical applications needing a specific power. Therefore the application would shut down before 0% SoC.
- The residual capacity is hard to obtain accurately by measurements due to errors in measurement equipment.
- The residual capacity is not a linear quantity when measured by Ah and the SoC will decrease relatively faster at a low SoC than a higher SoC.

2.9 SoC variations over Time

In [42], the authors emphasize the importance of the time aspect of the SoC estimation due to different properties impacting SoC for different time spans. The three factors are electrical, thermal, and mechanical, listed in the same order the size of the time impact ranging from milliseconds to years. An overview is given in Figure 2.9. To estimate SoC accurately when the time spans overlap, all the overlapping properties have to be taken into account. The time frames with its related properties are as follows:

1. **seconds - less than a minute:** the electrical properties are majorly impacting SoC, while

temperature, stress, and volume remain fairly constant. Therefore, only the electrical properties are required for SoC estimation within this short time frame.

2. **minutes - a few hours:** thermal effects has to be considered. If the stress field or volume varies significantly, these mechanical properties should also be taken into account.
3. **days or longer:** mechanical properties as self-discharge, normally has a considerable impact on the cumulative changes of SoC. Therefore all three properties should be taken into account when estimating SoC over a longer period.

Since the parameters are coupled (*e.g.*, increased temperature yields an increased aging (mechanical) and lower resistance (electrical)), it is important that the model estimating SoC takes this coupling into account when required. Often SoC models only couple the electrical and thermal properties, if any [42].

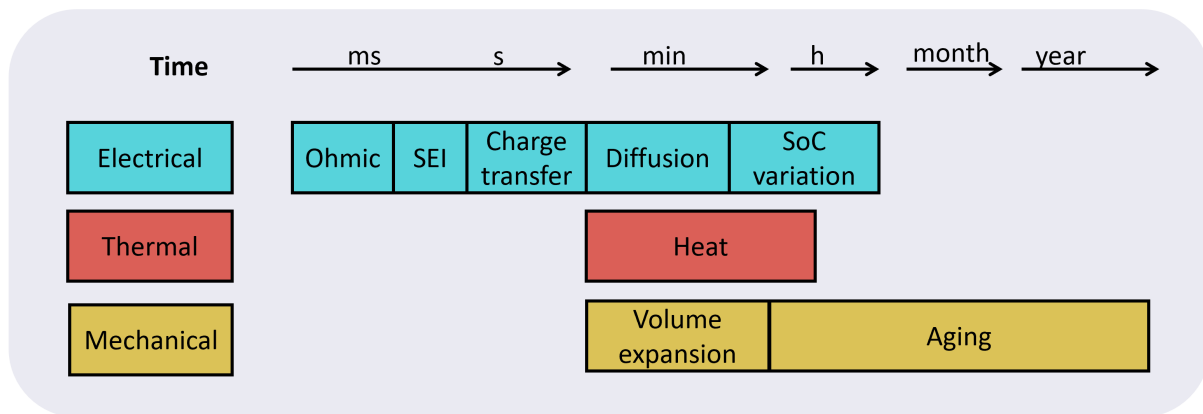


Figure 2.9: The impact on SoC by electrical, thermal, and mechanical properties with respect to time measures.

2.10 SoC of EVs

For today's EVs, the SoC is determined by a km-range, converted from the required power, for a user friendly estimate. Several additional challenges arise when calculating the SoC by the km-range due to its dependence on several external factors. These includes weather conditions (*e.g.*, the wind velocity changes the drag factor and thus the power required to drive a certain distance), the ambient temperature changing the internal resistance of the pack, and topography of the road. In addition, the users driving habits, how heavy the car is loaded, and the amount of air conditioning and heating, will influence the amount of power used. Moreover, braking will charge the battery, complicating the km-range estimation further.

A second concern is that EVs require a battery pack rather than a single cell, implying that the SoC of the battery is dependent on numerous cells. Moreover, all the cells in a pack are not installed with the exact same SoC, and therefore some cells reach 0% SoC before the rest. The

SoC of the whole battery pack is determined by the cell with the lowest SoC, meaning that the cell reaching 0% SoC first defines an empty battery pack. If all cells are charged and discharge with the same current and without modifications, the gap between each cell's SoC will grow even further over time. Eventually this would lead to premature failure of the system [43].

To avoid that the cells in the battery pack have different SoC, each cell's SoC is monitored and adjusted accordingly to yield the same SoC for all cells. This is called cell balancing, where one technique used is passive cell balancing, which involves converting the excess energy of the cells with a high SoC into heat energy. The downside is that the heat energy requires a cooling system and the energy is lost by dissipation. Another way to balance the cells is by using the excess energy to charge the cells with the lower SoC. This is however a costly system.

Another noteworthy concept of the SoC system in a EV, is that the battery is usually never fully charged or discharged with respect to the manufacturer's upper and lower voltage limits. The DoD window is usually smaller than the battery manufacturer has specified in order to improve the lifetime of the battery [44]. As an example, Tesla's model 3 has 78.27 kWh usable capacity out of a total capacity of 80.5 kWh, yielding a 97% DoD window [45]. In addition, it is not recommended to charge it to more than a 90% SoC.

2.11 Methods to Estimate SoC

SoC is a challenging problem due to its correlation to many different parameters including the ambient temperature, C-rate, losses due to resistance and over-potentials, aging mechanisms like increased solid electrolyte interface layer, dendrites and cracks in the electrodes, the OCV with its hysteresis effects, and the battery chemistry. Many of these parameters cannot be determined directly without opening up the battery and consequently destroying the battery permanently, or a need of heavy and costly equipment like x-ray [46], which is unsuitable for most applications.

To overcome the challenges described above and estimate the SoC, different techniques have been developed. In this thesis, four classifications will be made: direct measure, book-keeping estimation, physics-based methods, and adaptive systems. The first one refers to estimating the SoC based on measuring physical battery properties. Book-keeping is referring to integrating the current drawn over a time in order to calculate what fraction of the total capacity that is drawn. Adaptive systems can automatically adjust the SoC for different discharging conditions [47]. The last one, physics-based models, is referring to estimating the SoC by models resembling the physics of the battery and can simulate the battery's operation.

2.11.1 Direct Measurement

Direct measurements refer to the measuring of physical parameters like voltage and impedance. Methods include the OCV, which is widely used in SoC estimations [48]. This method can yield high accuracy, but needs a long duration period of up to several hours before the voltage has

reached a stable condition [49]. When the required relaxation of up to several hours is impossible, there exist methods to predict the OCV. However, each method has some limitations like lowered accuracy for aging batteries [50]. Therefore, they are not suitable for real-time applications [51].

Another disadvantage of using OCV-SoC tables is that the relationship between OCV and SoC varies between the battery chemistries and temperature, requiring extensive tables. Furthermore, the degrees of accuracy when using the tables vary, especially with respect to the battery chemistry. For instance, acid-lead batteries have a relatively linear relationship, whereas LiBs in general have a flatter OCV-SoC profile for the majority of the SoC range. This makes it more difficult to obtain accurate SoC based on OCV. However, the shape of the voltage vs DoD curve also varies between the LiB chemistry's as can be indirectly observed in Figure 2.4 where voltage is plotted against the time.

2.11.2 Book-keeping Estimation

The coulomb counting model is an established and straight forward method [52] amongst the book-keeping methods. The formula for Coulomb counting is as follows:

$$\text{SoC}(k) = \text{SoC}_{\text{init}} - \frac{1}{C_t} \cdot \int_0^k \gamma \cdot i \, dt \quad (2.14)$$

where t and k is the continuous time and current time step respectively, SoC_{init} is the SoC at the initial time $t = 0$, C_t is the total capacity of the battery, and i is the current. γ is the charge/discharge efficiency. It is generally around 0.9 and 1.0 during charging and discharging, respectively. γ depends on temperature and C-rate [53, 54].

Favorable of the coulomb counting model is that internal battery effects as self-discharge, capacity-loss and γ can be included [47]. For example, if the capacity-loss is known, the total capacity C_t can be adjusted, or, if the γ is known, the residual capacity at charging, can be adjusted accordingly [55].

On the downside, coulomb counting is a open loop method yielding a significant final error due to accumulation when integrating the system [8]. In addition, since the charge/discharge efficiency varies with temperature and C-rate, it is not known in advance, especially if the C-rate is changing during the cycle. Moreover, the SoC estimation will never be better than the initial SoC estimation, which is self-explanatory when observing Equation 2.14. However, when the SoC can be calibrated regularly, the measurement equipment is accurate, and little noise from the surroundings is present, coulomb counting is a fairly accurate SoC estimation method.

2.11.3 Physics-Based Methods

The ideal SoC estimator could be a digital twin where all atoms of the LiB are simulated. Nevertheless, the modeling would be very computationally costly to simulate. Models based on physics have been developed to predict microscopic behavior [56]. However, they are limited by several simplifications and need high computational power in order to solve the differential equations

[57]. More simplified battery models exist (*e.g.*, equivalent circuit models), which are advantageous due to a easy parameterization and adaptability for real-time applications [58].

Electrochemical based models have also been under exhaustive research to estimate the SoC. The concept is to model the chemical reactions in the battery and thereby capture the main dynamics. The SoC is predicted directly by the amount of charge in the active material of the electrode, which is represented by the lithium concentration [59]. Both the SoC for each individual electrode and the total SoC for the whole battery can be estimated.

As in most physical modeling, there is a trade-off between the precision and complexity. With a more complex model in terms of defining a greater amount of model parameters, a more precise battery model can be obtained, which can result in precise SoC estimations. However, this will also lead to more time consuming calculations and a higher degree of hand engineered parameters. Complex models are therefore difficult to apply for real-time systems [60]. Still, in the future, complicated physics based models might be an accurate and a good option if utilized together with cloud computing such that the hardware does not need to be situated in the EV, creating a heavy car. With cloud computing, data is sent from the EV to a powerful cloud server which can perform extensive estimations. By enabling cloud computing, the SoC could calibrated by physical models on a regular basis, for instance once every hour, while simpler algorithms in the car are used to estimate the SoC in between the calibrations.

2.11.4 Adaptive Systems

For adaptive systems, Kalman filtering and ML methods are common methods. Adaptive methods can combine direct and indirect measurements. Numerous Kalman filter methods have been applied to estimate SoC based on the OCV method and a battery model [48]. Some of the popular filters are extended Kalman filter [61], adaptive extended Kalman filter [62, 63] and unscented Kalman filter [64]. They have different advantages and disadvantages, but in common they have a recursive estimator that discretize a problem in time in order to linearize around an operating point. Then, the current state is predicted based on the prediction from the previous time step and the measurement from the current time step [65]. Generally, Kalman filters have the ability to reduce measurements noise without prior knowledge about the initial SoC and they can estimate the OCV before the battery has reached its relaxation time [66].

A shortcoming of Kalman filtering is the high mathematical computation time due to complex matrix operations. Therefore it is slow at estimating SoC for real-time applications compared to many ML algorithms [67]. Moreover, the Kalman filters accuracy is in general vulnerable to highly non-linear systems, like the battery (*e.g.*, function of aging and temperature) [68]. This is because of its strong correlated to the accuracy of the battery model, as well as the predetermined variables of the system noise. An inaccurate Kalman filter model could lead to divergence and high errors [69].

Like Kalman filtering, ML has many beneficial aspects compared to the above mentioned tra-

ditional methods. Firstly, it can estimate the SoC for different temperatures without using a look-up table. In the previous works discussed in the next section, a single model is used to estimate SoC for multiple temperatures. Secondly, the model can directly characterize non-linear relationships between terminal voltage, current, battery temperature and SoC without battery models or filtering. This implies that human effort to parameterize each battery model or create the filter is unnecessary because the model is self-learned. Thirdly, when an ML algorithm has been trained, little computational time is required to use it for predictions, contrary to expensive series of matrix multiplications. On the contrary the extended Kalman filter model or model-based methods require intensive calculations like partial differential equations. For a simple NN, the computational time can be an order of magnitude faster than extended Kalman filter, which is important when estimating SoC for a considerable number of cells for instance in an EV [67].

A downside with ML is the difficulty to validate the predictions. However, this is also true for the other models described above. For a thorough review of the current trends in SOC estimation methods for EV, see the attachment Paper I.

Theory on Machine Learning

3.1 Overview of Machine Learning Methods

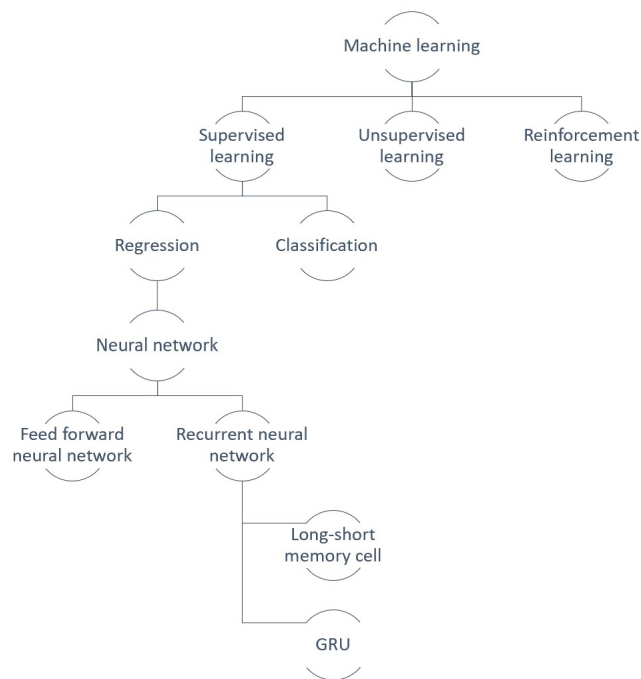


Figure 3.1: Overview of how the ML is subdivided into specific categories, methods and algorithms. Only the algorithms used to estimate the SoC in this report are presented (*e.g.*, NN and RNN with LSTM and GRU cells).

Machine Learning (ML) is a collective term used about algorithms that learns from previous collected data. The goal of any ML model is to extract patterns from a data set and predict one or more outputs. This is done by adapting a model to a specific input-output transformation

task. There are three major categories within ML; supervised machine learning, unsupervised machine learning, and reinforcement learning. For State of Charge (SoC) estimation, supervised machine learning algorithms are popularly used, indicating labeled tasks. Labeled tasks means that the model learns from knowing the actual solution of the task. An analogy to supervised ML would be that a student learns from trying to solve problems and after the problem is solved, the teacher gives the student a solution manual. Then the student can adjust what was wrong, and do better on a similar task the next time. Specifically for a SoC estimator, supervised ML would be executed by the model getting battery parameters as the input, then try to predict the SoC, subsequently the actual SoC and adjust the model accordingly.

Supervised ML can further be divided into two subcategories, regression, and classification. Classification is a method used to group information, for instance to classify whether a battery is a LiB or a lead-acid battery. Regression is used to predict a continuous numerical label, and will be used when predicting the SoC. Numerous of algorithms exists like regression trees, support vector regression and Neural Network (NN), where the latter will be used to estimate the SoC in this project. Figure 3.1 depicts the hierarchy of the main categories explained above.

3.2 Basic Principles of Supervised Machine Learning

Large amounts of data is essential when constructing an ML model since the model learns from real world data. The data should be divided into three; training, validation, and testing as illustrated by Figure 3.2.

Before training on the data, the choice of model and the corresponding hyperparameters have to be determined based on the problem. The hyperparameters are the parameters external to the model whose value cannot be estimated from data, but has to be decided by the scientist. Examples of hyperparameters will be described later and include the number of layers in a NN, the number of nodes in each layer and the learning rate. The hyperparameters are often decided by rules of thumb or by trial-and-error [70].

When the model is constructed, training is done as an iterative process. Training data is sent through the network, an error score is calculated and the model is adjusted accordingly. This process will be described in more detail in Section 3.3.6. Sending the training data once through the network is called an epoch. Often many epochs have to be carried out before the model has converged to a satisfactory level.

Finally, testing is done to check to what extent the model can handle unseen data. Therefore it is essential to put away a decent amount of data which is used for testing. Furthermore, the training data should be split into a training set and a validation set. The validation set is used after each epoch to calculate the error score. If the error of the validation set goes up, training is typically stopped. Therefore, the validation set determines the epoch that training is stopped at.

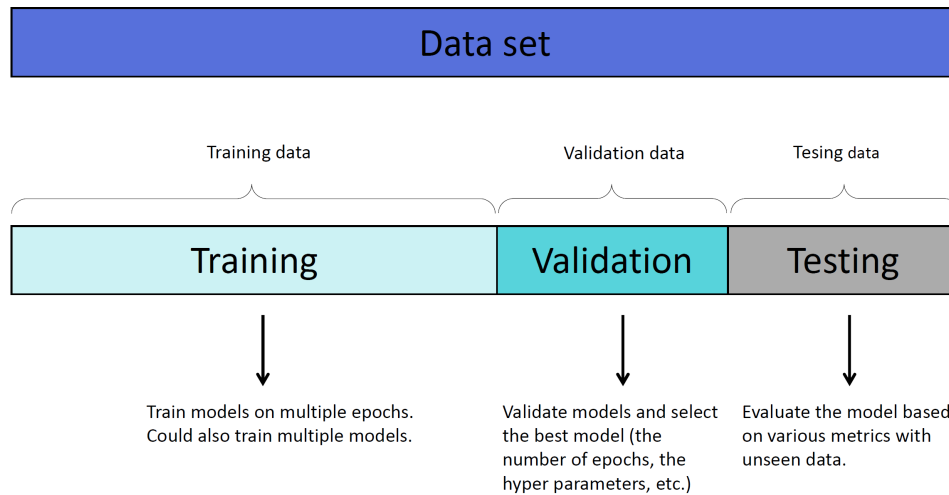


Figure 3.2: How a large data set is divided into training, validation and testing data before used in ML.

In the training stage, the model is fed with the training data. An appropriate cost function must be determined in order to evaluate the performance of the training. Then the parameters are adjusted iteratively until convergence of the cost function. The advantage of using a validation set is to find the point when the cost is still converging for the training data, while diverging for unseen data. After this point, the model is overfitted and further training will decrease the models performance.

Some concerns have to be taken before splitting the data set. Firstly, the amount of data available has to be evaluated in terms of finding an appropriate split ratio. Most models are heavily dependent on a large data set for training to extract generalized patterns. Additionally, the testing data must be large enough to yield unbiased results. Therefore, the conflicting goal of having a large training and having enough validation and testing data to yield reliable test results has to be balanced.

It is fundamental that the test set should contain unseen data that has been decoupled from the training phase. For SoC purposes this could be battery data from another research group to ensure independent measurements. In addition, by using battery data obtained at a different temperature than the training data, a measure of whether or not the ML algorithm was able to abstract and interpolate the relationship between different temperatures can be obtained.

3.3 Neural Networks

The majority of the state-of-the-art methods to predict SoC directly are variants of Neural Networks (NNs) as Forward Neural Networks FNN and Recurrent Neural Networks RNN. Therefore, a general introduction of NN will be given in this section.

NNs can extract patterns in data that are challenging for humans to make a predictive model for. For instance, many physical phenomena are difficult to find the mathematical relationships and create physical equations. NNs should in theory be able to approximate any unknown regression function. The FNN is one of the simplest neural networks to apply [71]. Once trained, it offers a low computational cost since the forward pass only includes consecutive matrix multiplications.

3.3.1 Artificial and Biological Neural Networks

Artificial NNs is black box function in the sense that we to this date do not understand what patterns the network extracts. Moreover NN is a collective term used for machine learning algorithms that are inspired by the architecture of the biological brain. The brain has neurons that are connected together and fire if an electrical potential or chemical signal is strong enough. Analogously, there are neurons in an artificial neural network that passes the information forward. Whereas the biological neurons turn on and off as a binary system, the artificial neurons are not, but passes though a real number (traditionally with a range between zero and one.)

The connection between neurons in an artificial NN, are through weights (w) which are real numbers. In Figure 3.4, the weights are represented by the arrows between the artificial neurons. Our brain learns by modifying the connections between the neurons. Similarly, the artificial neural network learns by modifying the weights [72]. Two artificial neurons connected with a weight of a high value, indicates a strong connection between the neurons.

To distinguish biological brains and ML, neurons in ML are often called artificial neurons or nodes. However, from this point forward, this report will always be referring to artificial neurons when "neuron" is stated.

3.3.2 Mathematical Explanation of a Neural Network

In this section the fundamental working principles of an FNN will be explained by equations and illustrations. There are several abbreviation where the most important once are:

- $x_{i,j}$: input
- y_j : output from the output layer
- $w_{j,i}$: weights
- b_j : bias
- z_j : output from the nodes of the first and hidden layer
- ϕ : activation function

Mathematically, each node takes in a vector of input features that are multiplied by a specific weight. Then the weighted inputs are summed together. In addition, a constant value called bias is added to the sum. Each node has their own bias that belongs to it which determines what influence the node has to the final output. This implies that the bias is independent of the input to the node. Put simple, a weight determines the connection between two nodes, whereas the bias determines the importance of the node. The net input to the node is:

$$z_j = \sum_{i=1}^n x_i w_{j,i} + b_j \quad (3.1)$$

where x_i is the input vector, $w_{j,i}$ is the weight connecting the input i with the node j , b_j is the bias of node j , n is the number of nodes, and z_j is the net output of node j . This net input is passed through an activation function (ϕ) to produce the output of the node which is forwarded to the next layer. Different activation functions are described in Section 3.3.4.

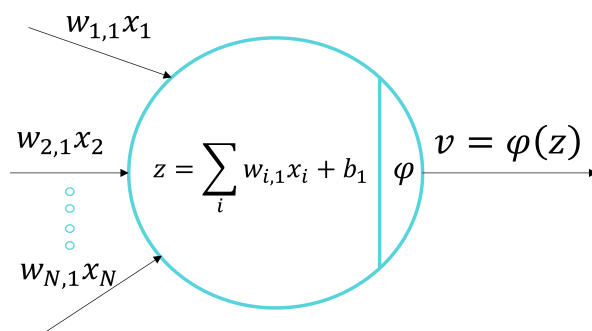


Figure 3.3: Illustration of the first neuron of the hidden layer. The inputs (x) and weights (w) are multiplied and then summed. Next, the bias (b) is added to the sum before it is sent through an activation function (ϕ).

For regression problems, the final output from the network is calculated directly from the outputs of the last hidden layer. There is only one output node in the last layer, because only one regression value is predicted. However, for classification a range of output nodes are utilized where each output node represents the probability of a certain class. A softmax function is used to squeeze the output to a number between zero and one yielding a probability. The final output is calculated by Equation 3.2a and 3.2b for regression and classification, respectively.

$$Y_{regression} = \sum_{i=1}^n w_i \cdot \phi(z_i) \quad (3.2a)$$

$$Y_{j,classification} = \frac{e^{\phi(z_i)}}{\sum_{i=1}^n e^{\phi(z_i)}} \quad (3.2b)$$

here $\phi(z_i)$ belongs to the node from the last hidden layer.

During training, the weights and biases are updated to minimize the error between the predicted output and the targeted output. This procedure is called backpropagation because the error is propagated back through the network to adjust the two parameters. A common backpropagating scheme is gradient descent, which is discussed in Section 3.3.6.

Figure 3.4 displays the general architecture with weights and biases represented by the arrows, and input nodes, output nodes and hidden layer nodes are displayed by the circles. This figure

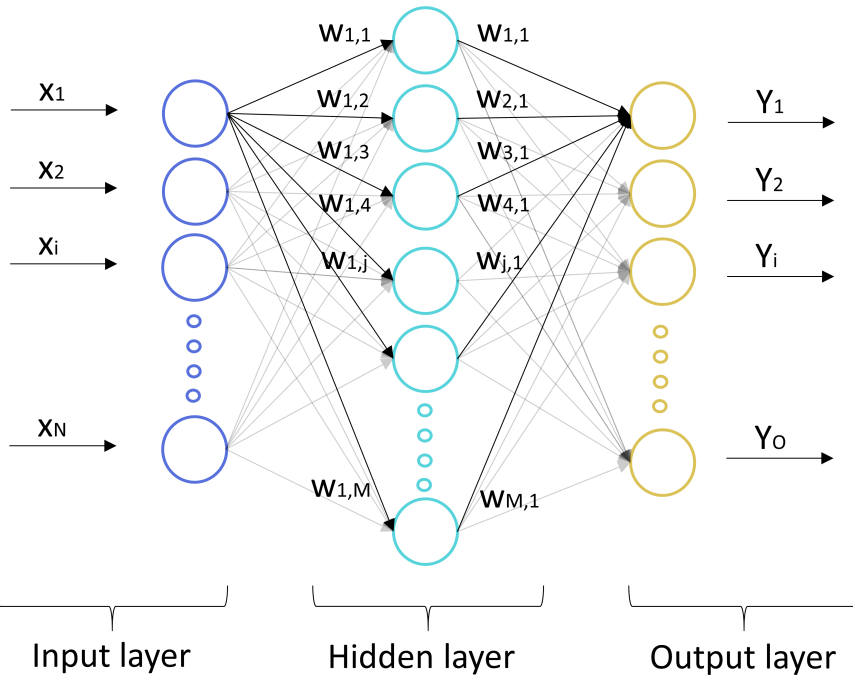


Figure 3.4: Architecture of a generalized FNN of N inputs, O outputs and an undefined number of layers and nodes. For regression problems, the output vector Y is reduced to a single variable.

has only one hidden layer, whereas NNs can have a multiple number of layers and is then called a deep NN or a multilayer perceptron NN.

Put easy, constructing a NN can be done by the following steps:

1. Initialize the data set and divide it into a training set and a validation set.
2. Set the parameters of input, hidden and output layer nodes. Select the activation function(s), cost function(s) and optimization algorithms.
3. Set the hyperparameters of the network: time step, sampling interval, batch size and number of iterations. Initialize weight and biases of the NN.
4. Select a function to evaluate the training performance and train the NN with the designated training data.
5. Validate the obtained NN by the validation set. Check the accuracy and the time effectiveness of the estimations.

Several considerations have to be taken into account before constructing an NN with the steps above. First of all, it is essential to choose an adequate network architecture depending on the application and the data representation. This includes determining the number of layers, hyperparameters and features. Overly complex models learn slowly and could become overfitted. When adding extra feature(s), the number of weights in the first layer will increase by a factor proportional to the number of inputs. For example, if the number of neurons in the first layer is 15, adding one extra feature will yield 15 extra weights for a fully connected first layer. Secondly,

the learning algorithm and cost function that optimize the hyperparameters have to be decided.

The rest of this section considers different functions, concerns and decisions related to creating a NN. Many of the terms and functions are general for all ML methods, but the examples are specifically related to NN.

3.3.3 Overfitting and Overtraining

Overfitting and overtraining are two terms that describe a model that does not perform well because it has been too specialized. Whereas overtraining is caused by training for too many epochs [73], overfitting refers to the network structure. Overfitting occurs when a network has too many parameters and becomes too complex [74]. Consequently it will fit accurately on the training data including potential noise, but yields poor estimations to situations not present in the training data [75]. The terms are related in the sense that overfitting is a consequence of overtraining an overly complex model. It is a delicate balance between having enough parameters in the network to match the unknown regression function of the training data, but not too complex to overfit. The same neat balance is found for the number of training epochs. If trained for too few epochs, the model will not be able to learn the training data, while training for too many epochs, the model is overtrained.

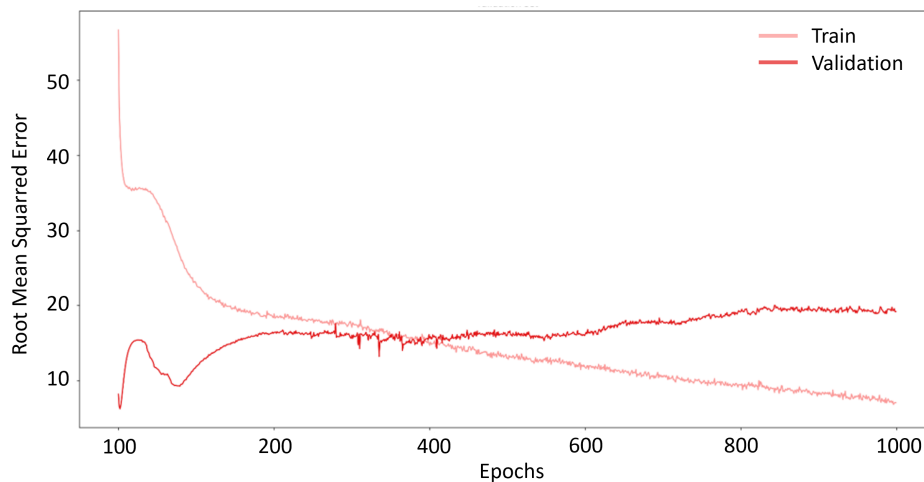


Figure 3.5: The error is calculated after each epoch for the training and validation data. Between 200 and 400 epochs overtraining is experienced. The validation error increases while the training error keeps decreasing. The reason the validation score is low at first is most likely good luck. The network was not trained to predict the SoC (plot made when experimenting with other data).

By partitioning the available data into three; training, validation and test data, overfitting and overtraining problems can be monitored and evaluated. The training set should only be used for the purpose of generating the model. The validation set should contain observations that are distinct from those of the training set. If the model is trained properly, the training error converges. However, when unseen data is continuously input to validate the model, the error

typically has a parabolic shape [76] as seen in Figure 3.5. Therefore, the validation set is used to discover the crossection when the model is still doing better when training it, but performing worse on the validation data. In other words, the point where overtraining occurs and training should stop. Figure 3.5 illustrates the validation error increasing since the a NN has overtrained.

When the model is established, test data is used to provide a score for how well it performs. Ideally, the test data should be new distinct data with respect to the training and validation set in order to evaluate the robustness and the model's ability to generalize. For instance, when testing a SoC ML model, one could have test data with a different ambient temperature than the ones used for training and validation. Other test data could be using different driving cycles or level of aging for the testing, than for the training. Then the model's ability to interpolate between different battery states is tested. Else, if the training and testing sets are too similar, the precision and accuracy results could be biased.

In addition to choosing a suitable algorithm and architecture, different methods called regularization have been developed to deal with overfitting. One of the methods is called dropout, where the idea is that by ghosting different nodes in each training epoch, the network is robust and can perform satisfactorily also without the dropout nodes and its corresponding weights. This is depicted in Figure 3.6.

Since much of the potential training data is lost to testing and validation, the results might depend on only a limited data amount. In order to increase the size of the training data set, cross-validation is frequently used. In cross-validation, the validation set is merged with the training data. Still, the test data should be held out to assess the final result. One cross-validation approach is called k-fold and divides the training data into k smaller sets (folds). Each of the k-1 folds are used as training data. The last fold is used for validating the model. Then a new

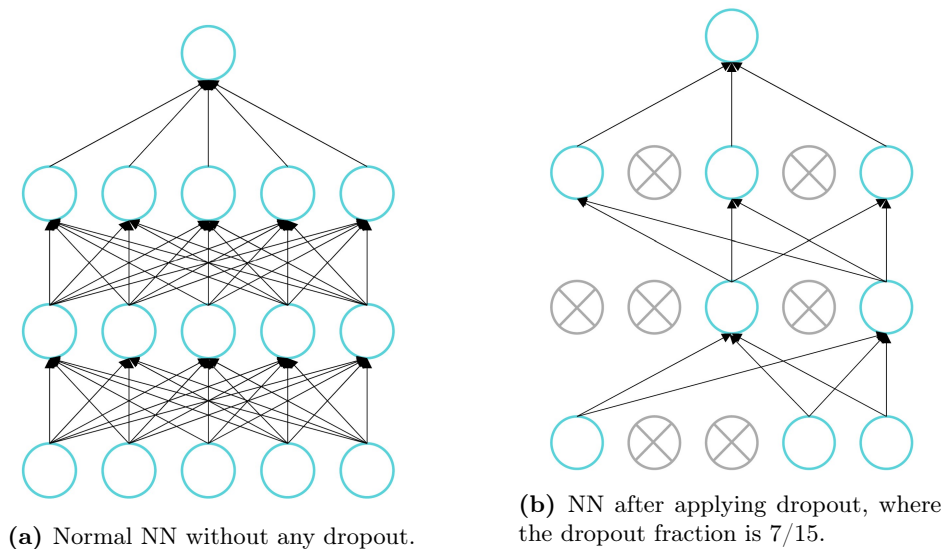


Figure 3.6: Illustration of dropout for an FNN.

model is constructed and a new fold is left out of the training data. By changing the fold used for validation each time, the models have in total been trained on all the folds. Eventually, k models will be constructed and the final accuracy score is recorded from the test data based on the average test accuracy of all the k models. The division of the data when employing k -fold is depicted in Figure 3.7. On a positive note, this approach allows for a larger training set, but can be computationally expensive [77].

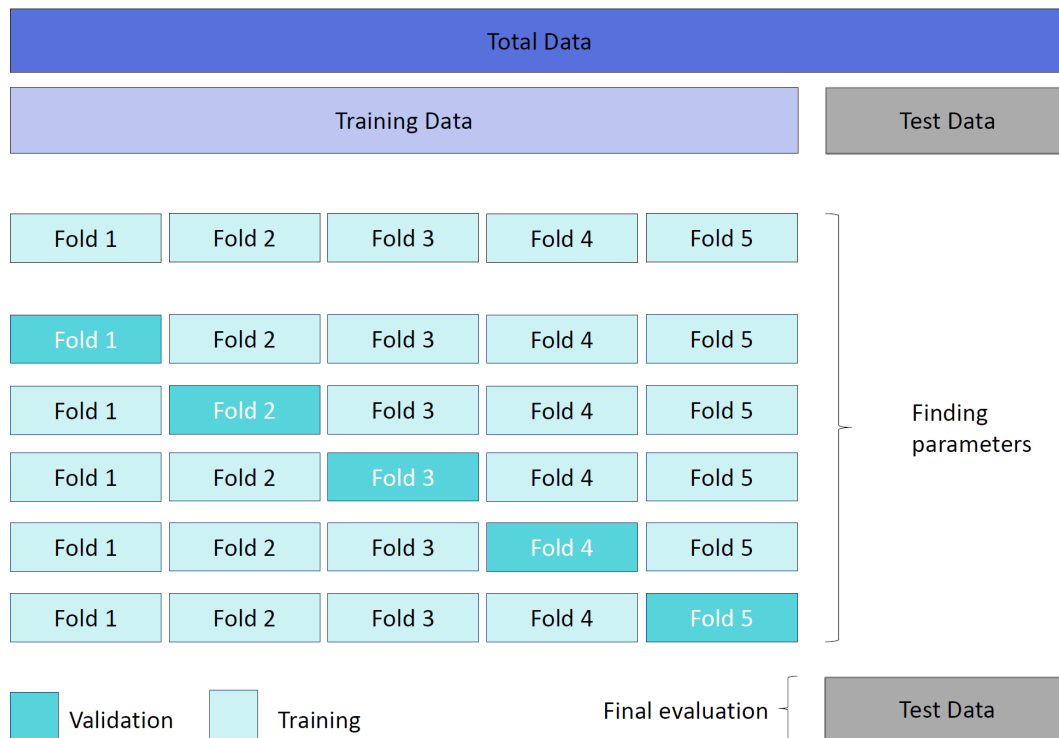


Figure 3.7: Illustrative description of division of the data by K-fold cross-validation.

3.3.4 Activation Functions

The most common activation functions are the rectified linear unit sigmoid, hyperbolic tangent and Rectified Linear Unit (ReLU) given by Equation 3.3a, 3.3b and 3.5c respectively.

$$\phi(z) = \frac{1}{1 + e^{-2z}} \quad (3.3a)$$

$$\phi(z) = \frac{1 - e^{-2z}}{1 + e^{-2z}} \quad (3.3b)$$

$$\phi(z) = \max(0, z) \quad (3.3c)$$

Sigmoid and hyperbolic tangent activation functions can learn complex structures due to their non-linear activation as seen from Equation 3.3. They were much used in the 1990s and in the

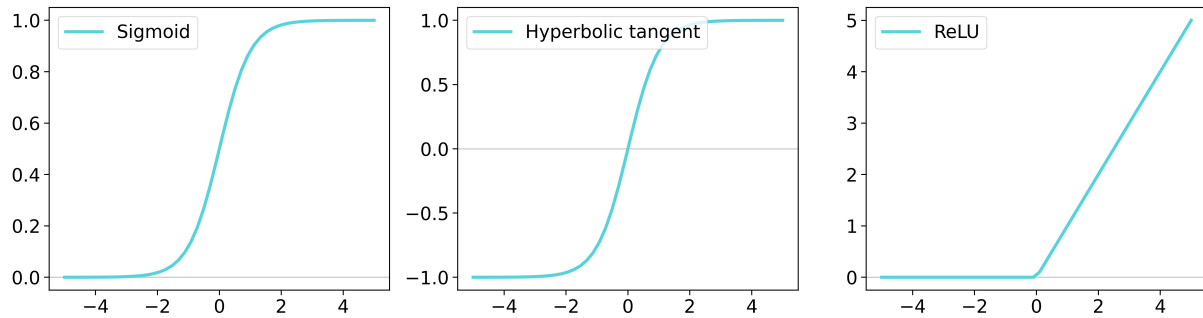


Figure 3.8: Three common activation functions for NNs.

start of 2000s. The non-linearity, which is a strength in complex problems, is also the main drawback for the two activation functions when using them in DNNs. This is due to the risk of saturation, meaning the function goes towards its minimum or maximum value resulting in limited training. In addition, the computational time is large due to the effort spent calculating the derivatives of the non-linear function.

In 2020, the ReLU activation functions is the default activation function for multilayer perceptron and convolutional NNs [78] [79]. It is a piece-wise linear function outputting directly the input if positive, and zero if the input is negative. The main drawback of the ReLU function is that it cannot alone learn complex mapping functions due to its linearity. The advantage is that it is fast to train [80] and more robust with respect to the descending gradient problem than the aforementioned functions. There exist several variant of the ReLU, where clipped ReLU and leaky ReLU are amongst the common ones. The clipped ReLU is obtained by setting a maximum limit for z in Equation 3.3c. The leaky ReLU is distinguished from ReLU by a small, positive gradient when unit is not active which can speed up training and prevent vanishing gradients [81]. Mathematically, leaky ReLU is stated by:

$$\phi(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.} \end{cases} \quad (3.4)$$

To conclude, the sigmoid and hyperbolic tangent activation function are suited for complex mappings in shallow NN due to their non-linear activation. However, they are not preferred in DNNs because of the vanishing gradient problem, whereas the ReLU activation function has a tendency to overcome this problem [80].

3.3.5 Cost Function

A cost function (C) estimates how badly the models perform after each epoch. For NNs it is expressed as the difference between target output and predicted output. Since it quantifies the error, the cost function for NNs is known as an error or loss function [82]. Three popular cost functions are Mean Square Error (MSE), Mean Average Error (MAE) and Root Mean Square Error (RMSE), defined in the following manner:

$$C = MSE = \frac{1}{n} \cdot \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \quad (3.5a)$$

$$C = MAE = \frac{1}{n} \cdot \sum_{i=1}^n |Y_i - \hat{Y}_i| \quad (3.5b)$$

$$C = RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2} \quad (3.5c)$$

Here n is the number of observations, Y_i is the estimated value and \hat{Y}_i is the target value. In addition to updating the weights, these functions are also used as metrics to indicate the accuracy when testing an ML model.

3.3.6 Training Algorithms

The goal of the training algorithm is to tune the weights and biases by minimizing the cost function. This is illustrated by Figure 3.9. The backpropagation algorithm is popular to use [83]. Gradient descent is a type of backpropagation which works by adjusting w and b in the direction of the negative gradient of the cost function (C).

When the training data is large backpropagation can be slow. A way to speed up training is to divide the data into smaller subsets, called batch. The downside is that the subset might not be representative of the whole data set and consequently not updated by the optimally.

There are several different versions of gradient descent, where the simplest method is called simple gradient descent. Other methods include Broyden-Fletcher-Goldfarb-Shanno algorithm [84] and the Levenberg-Marquardt algorithm [85] [86]. The parameters are adjusted after each batch to yield a more sophisticated estimate in the next batch. When using gradient descent any C can be utilized where some of the most typical are described in the previous section (Section 3.3.5). Mathematically, when b and w are updated by the simple gradient descent, they are updated by the chain rule in the following manner:

$$(w_i)_{\text{epoch } a+1} = (w_i)_{\text{epoch } a} - \alpha \frac{\partial C}{\partial w_i} \quad (3.6a)$$

$$b_{i+1} = b_i - \alpha \frac{\partial C}{\partial b_i} \quad (3.6b)$$

$$\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial v_i} \frac{\partial v_i}{\partial z_i} \frac{\partial z_i}{\partial w_i} \quad (3.6c)$$

$$\frac{\partial C}{\partial b_i} = \frac{\partial C}{\partial v_i} \frac{\partial v_i}{\partial z_i} \frac{\partial z_i}{\partial b_i} \quad (3.6d)$$

where

- $\frac{\partial C}{\partial w_i}$ is the value of the gradient of the partial cost function with respect to neuron i .

- $\frac{\partial C}{\partial w_i}$ is the value of the partial derivative of neuron i with respect to the weight parameter w_i .
- $\alpha > 0$ is the learning rate controlling the step size the gradient descent takes for each iteration. The magnitude of α is a compromise between the w and b not converging to the optimal parameters and a too slow learning rate yielding a slow convergence. Often, α has a small magnitude where typically 0.01 is used [87].

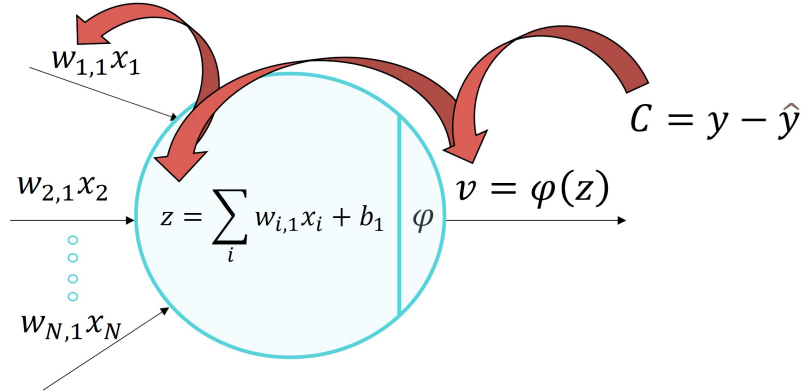


Figure 3.9: Back propagation to update one weight.

There are mainly two challenges related to backpropagation, namely the vanishing and exploding gradient problem which happens when $\frac{\partial C}{\partial w_i} < 1$ and $\frac{\partial C}{\partial w_i} > 1$, respectively. Since the error is propagated back in order to tune the hyperparameters, the gradient of NNs can either explode or vanish, making the training pointless with unfit high weights or too slow with negligible small updates of the weights. The most encountered problem of the two is the vanishing gradient, where the gradient becomes so small that updating the weights would be negligible and the training would be very slow.

3.3.7 Initialization

For NN, parameter initialization involves setting a value of the weights and biases. Initialization influences the convergence rate since the solution space is large and a suitable initialization can effectively reduce it [83]. Moreover, it can reduce the chances of exploding gradient. Heuristic methods have shown to yield good results where random initialization or sparse initialization are amongst the popular techniques [88].

The initialization of the weights are usually done by a partly random value close to one, while the bias' are often assigned a zero value [89]. Sparse initialization can be done by breaking the symmetry by not fully connecting all the nodes of the network. Most of the weights are set to zero, while some of the weights are initialized from a small Gaussian distribution [90].

3.3.8 Scaling

After the hyperparameters have been chosen and the network has been constructed, preprocessing of the data in form of scaling should be done before training starts. Scaling can be an efficient tool to enhance robustness and effective training for ML algorithms that uses gradient decent as an optimization technique. The reason is that in gradient decent, the step size of the gradient that updates the network parameters are effected by the input value. Therefore, large numbered inputs will be of bigger significance than small numbered inputs. The scaling scheme used for a task should reflect the problem and various techniques exist including min-max scaling and standardization.

Min-Max scaling is great to use when the distribution is not Gaussian. In literature, it is often ambiguously refereed to as normalization. The advantage of min-max scaling is that it scales all data points, including outliers. It is expressed by:

$$X' = (L_u - L_l) \cdot \frac{X - X_{\min}}{X_{\max} - X_{\min}} + L_l \quad (3.7)$$

where X_{\min} and X_{\max} is the minimum and maximum value of the training data, respectively. L_u and L_l is the upper and lower boundary of the scaling. For example, $L_l = 0$ and $L_u = 1$, will yield a min-max scaling that squeezes the data between zero and one.

A drawback is that if outliers occur in the data, they will be the minimum and/or maximum values. This would result in squeezing the remaining data together, yielding little variance in the ordinary data. Standardization is usually applied for Gaussian distributed data. It is mathematically done by:

$$X' = \frac{X - \mu}{\sigma} \quad (3.8)$$

where μ is the mean value, and σ is the standard deviation of the training data. Its main drawback is the lack of a bounding range, since consequently outliers will be less effected by standarization.

There is no unambiguous answer to whether min-max scaling or standardization should be done on regression problems. The only consensus is that scaling should be done when solving regression problems with neural networks. In literature on SoC estimation with NN, min-max scaling is what is often used (*e.g.*, min-max scaling with $X' \in [-1, 1]$ was used in [91], [92], and [93], and $X' \in [0, 1]$ was used in [94] and [95]).

3.3.9 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a subgroup of NNs and are used primarily for modeling sequential data. The NN is made recurrent by passing the output, or intermediate state, to the input. For instance the SoC of k-1 can be input in state k. This is illustrated in Figure 3.10.

The main advantage with an RNN is that by storing data from the past state it bases decisions on both weights and previous inputs. Therefore, the network is said to have a memory, enabling the RNN to use complex signals from past time sequences in modeling the present period. On

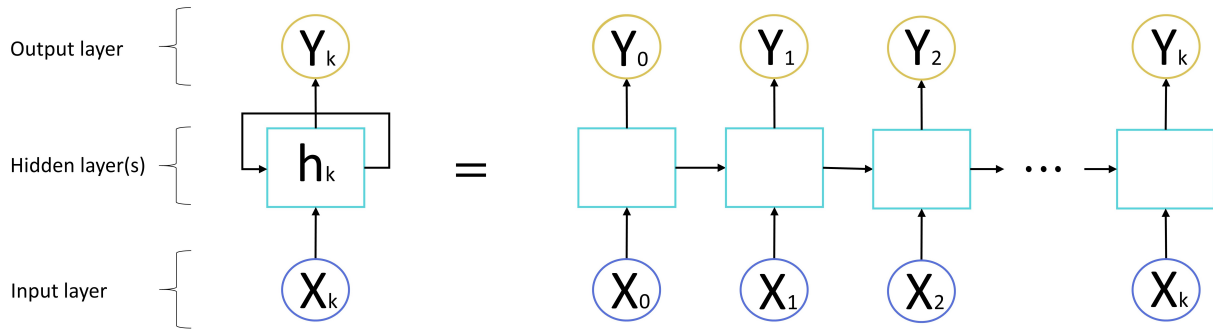


Figure 3.10: General RNN architecture. Each block represents an NN where the input to the network is both from the feature vector represented by X_k and the output from last estimation Y_k .

the downside, the RNN has a short term memory due to vanishing or exploding gradient in backpropagation [96] for deep NNs. Therefore, the long-term dependencies observed for batteries makes the plain RNN unfeasible. The simplest form of an RNN is depicted in Figure 3.11.

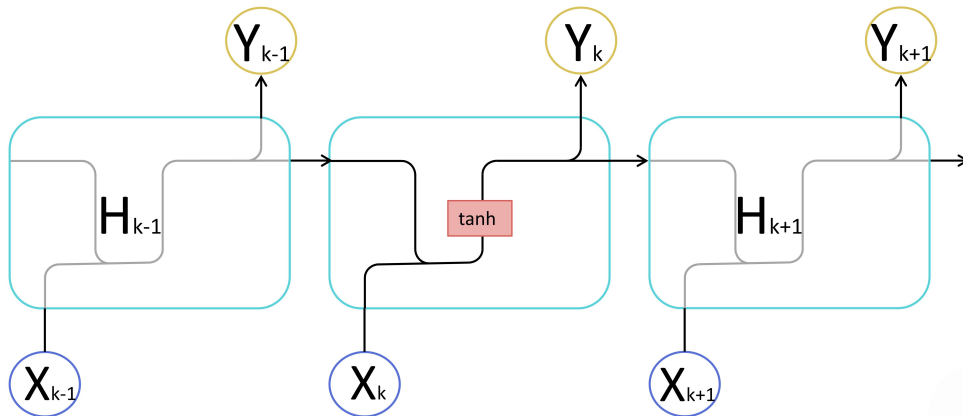


Figure 3.11: Simplest architecture of an RNN with only a hyperbolic tangent activation function.

By substituting the simple cells in Figure 3.11 with Long Short Term Memory (LSTM) or Gated Recurrent Unit (GRU) cells, the RNN is suitable for deep learning. The cells can learn long time dependencies through gates that connect the past and current data. The cells are constructed by tree gates: forget gate, input gate and output gate. An illustration of an LSTM cell is found in Figure 3.12. The purpose of the gates mechanism is to regulate the flow of information that passes from previous time steps to the next. Mathematically, the gates are simply the sigmoid activation function that is stated in Equation 3.3a. It causes the information to disappear if the output of the sigmoid function is zero, partly forgotten if the value is between one and zero and perfectly remembered if the output is one. The hyperbolic tangent function, Equation 3.3b, is also used in the cell in order to regulate the magnitude and the importance of the input.

To elaborating the different cells, they use hyperbolic tangent and sigmoid activation function for the activation of the cell state and node output respectively. This configuration is illustrated of the LSTM cell in Figure 3.12. The GRU have the simpler structure of the two and have less tensor operations and are therefore slightly faster to use than LSTMs. Additionally, it is more robust to vanishing gradient and needs less memory storage [93]. When determining which of the two cells to utilize, the engineering practice is to test the two methods for each explicit problem [97].

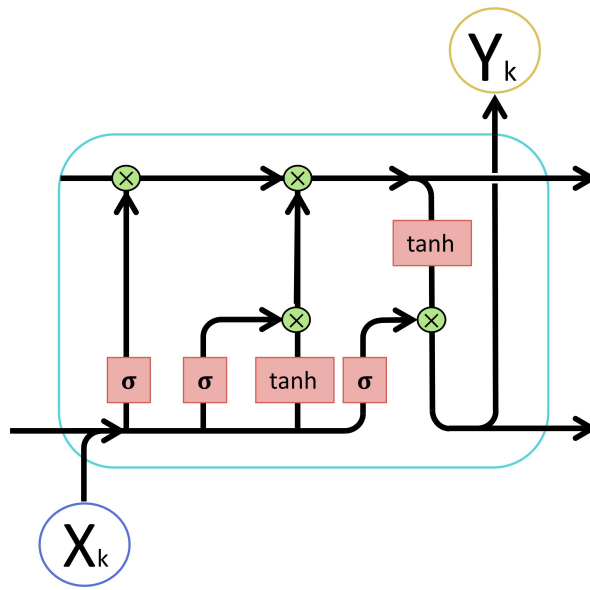


Figure 3.12: Illustration of an LSMT cell. Tanh is short for hyperbolic tangent function stated in Equation 3.3b. Moreover, in this figure σ represents the sigmoid activation function (not to be confused with standard deviation).

3.3.10 Backpropagation Through Time

Since RNNs have a temporal dimension, a back propagation with respect to the time step is necessary. As described by Figure 3.13 the gradient of the cost function with respect to the weights $\frac{\partial C_t}{\partial W}$ is propagated from the current time ($K = t$) to the initial time ($k = 0$) by applying the chain rule. Then the contributions of each time step is summed up to yield the total gradient of the current time step. Mathematically, it is written:

$$\frac{\partial C_k}{\partial W} = \sum_{n=0}^t \frac{\partial C_k}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial h_k} \frac{\partial h_n}{\partial W} \quad (3.9)$$

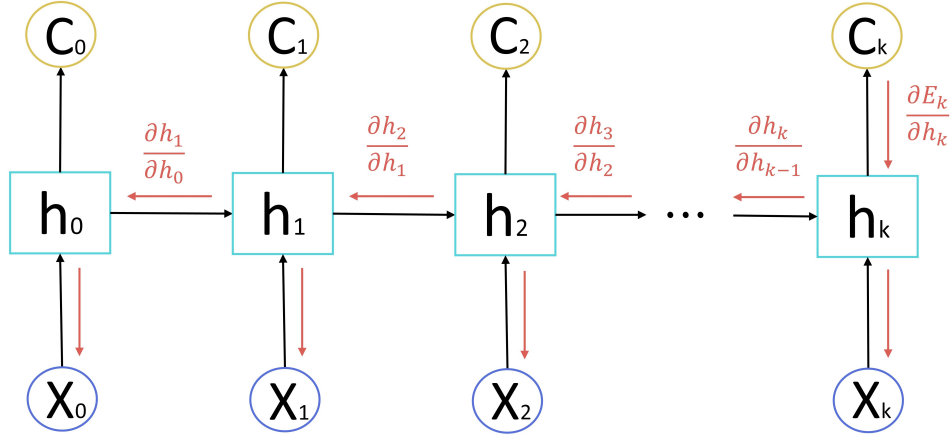


Figure 3.13: Backpropagation through time for RNNs.

3.3.11 Radial Basis Function Network

Radial Basis Function Neural Network (RBFNN) is a subclass of NNs with only one hidden layer, resulting in linear summation. The activation function ϕ is nonlinear and often presumed Gaussian G :

$$\phi(x) = G(\|x - c_i\|) = \exp\left[-\frac{\|x - c_i\|^2}{\sigma_i^2}\right], i = 1, 2, \dots, n \quad (3.10)$$

where $\|\cdot\|$ is the euclidean distance, x is an input vector and c_i is the center of neuron i in the hidden layer. The standard deviation σ determines the scaling of the curve. The sequence number of the Gaussian functions in the hidden layer is represented by i , where n is the total number of Gaussian functions. The output is predicted by the summation of the weighted Gaussian function:

$$Y = \sum_{i=1}^n w_i \cdot \phi_i(x) + b \quad (3.11)$$

where w_i is the weighted value of the Gaussian Function output, and b is the bias value often put to one or neglected. Figure 3.14 illustrates the RBFNN, where the "bells" illustrates the Gaussian activation function.

RBFs are known to be universal approximators and have a fast learning speed compared to other NNs [98]. During the training phase, the centroid vectors are fitted using an unsupervised training algorithm. A more in depth description of the training can be found in [99]. Common applications of the network are regression, classification, time series forecasting and pattern recognition. It is also tolerant to noise [100, 101], which can be beneficial if the sensor data from a battery is noisy.

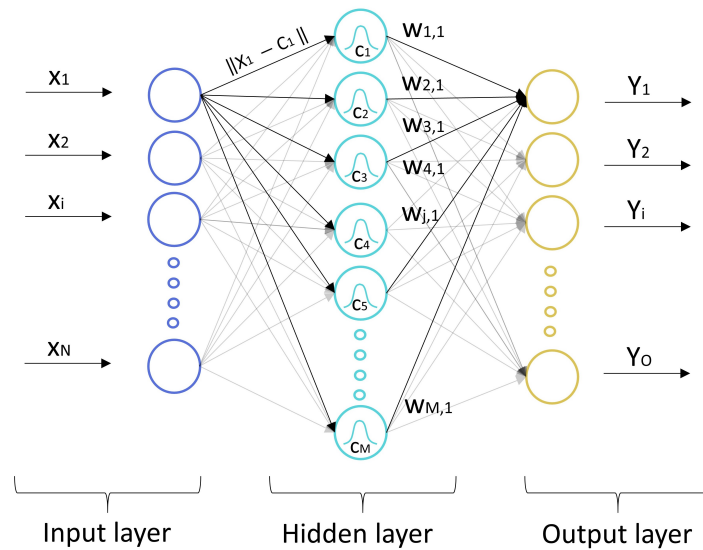


Figure 3.14: Illustrative sketch of a general RBFNN. X are the inputs while the hidden layer with a bell formed graph in each node represents the Gaussian function. For regression problems, the output vector Y is reduced to a single variable.

3.4 Hyperparameter Tuning

Hyperparameter tuning concerns searching for the best subset of hyperparameter values in a predefined space. This predefined subspace will be referred to as the search space. There exist a large amount of hyperparameter, and an even larger amount of combinations of them. It is both time consuming and computationally costly to try out every combination. Therefore, several optimization algorithms have been developed to tune hyperparameters. Two simple algorithms are grid search and random search. In both algorithms models are constructed with different combinations of hyperparameter within this search space and the search with the lowest error score is usually chosen.

The difference between grid search and random search is how the combination of hyperparameter are chosen. In grid search every combination within the search space is tried. In random search only an arbitrary choice of combinations are tried. For both algorithms, ideally, several initializations should be made per configuration. This is done to reduce the impact of the random nature of the neural networks initialization such that one poor initialization does not result in discarding the combinations of hyperparameter. The number of tries and initiations are predetermined by the scientist.

Work have shown that random and grid search yields similar results. However, the computational time of random search is majorly reduced [102].

3.5 Transfer Learning

Transfer learning is a method that uses pre-learned parameters (weights and biases) from a trained NN for training a new NN. As an example, in this thesis, transfer learning is used to learn the battery dynamics of one chemistry and then retrained on a different one. Another example is that a model could be trained to classify pouch cells. If the data on pouch cells is limited, training could first be done on classifying cylindrical cells and coin cells. Then, the network reuses the parameters and trains to also be able to classify pouch cell. Generalized, the principle of transfer learning is to train the NN on one feature space and with the same distribution. Then the model is retrained train on the desired data.

An advantage of applying transfer learning is that the huge amount of data in the domain of interest needed to train an NN will be reduced, if done successfully. Still, a considerable amount of data is needed from the desired feature space, but the data can deviate from the current problem [103].

When retraining the model, it is normal to reduce the learning rate. Moreover, the first layers (layers after the input layer) typically generalise better than the last layers. Therefore, it can be advantageous to set a lower learning rate for the first layers than for the last ones. Another alternative is to totally stop them from updating. This is often referred to as freezing the layers.

In [104] transfer learning is described by moving knowledge from a source domain to a target domain, with its respective learning tasks. Based on the problem's source and target domain, three classifications of transfer learning can be made as shown in Table 3.1.

In this master, the domains consist of LiBs from different manufacturers and a few different chemistries. The task is to estimate the SoC. It is assumed that the source domain and target domain are different but related and the task is the same. Therefore, transductive transfer learning is applied. When utilizing transductive transfer learning, it is expected that the retrained network will have an initially lower error, fewer epochs are required, and an overall lower error score.

Learning Settings		Source and/ Target Domains	Source and/ Target Tasks
Traditional ML		The same	The same
Transfer Learning	Inductive/ Unsupervised	The same	Different but related
		Different but related	Different but related
	Transductive	Different but related	The same

Table 3.1: Classifications of transfer learning based on the task and data at hand.

Previous Work

In this section, the major state-of-the-art Machine Learning (ML) approaches for estimating State of Charge (SoC) in Lithium-ion Battery (LiB)s will be discussed. The use of ML to estimate SoC can be divided into two categories: directly output the SoC from the ML algorithm, or optimize parameters of a non-ML model that in turn estimates the SoC. The non-ML models includes variations of Kalman filters and equivalent circuit models. The focus for this project will be on ML algorithms to directly estimate SoC.

Furthermore, the focus is put on newly published articles, specifically articles written after 2015¹. This choice is reasoned in ML estimating SoC is currently rapidly developing. A note should be taken with respect to the authors of the chosen literature. Some of the papers are written by research groups containing many of the same scientists. Consequently, the literature study does not cover as broadly as the number of literature indicates. Still, the study can indicate how far the field of estimating SoC with ML has come and the challenges of the field.

4.1 Description of the Battery Data

ML is a data driven method where a lot of data is one of the core elements for succeeding. Therefore, a basic understanding of the problem and its respective data is important. In addition some definitions of the terminology used in literature and some specifically for this thesis, is appropriate to be on the same wavelength.

To test the models properly, it is important that the data reflects real world driving. Fortunately, numerous profiles have been developed to simulate realistic driving profiles for vehicles. These are referred to as drive cycles and are commonly represented by velocity *vs.* time data [105]. The data should resemble real vehicles where traffic and road conditions makes the driving patterns differently. Therefore both highway driving cycles with constant speeds and fewer stops, and city driving described by more transient profiles have been constructed. A short description of each drive cycles is given in this section. However, the major difference is the amount of acceleration,

¹One article [92] is from 2014.

deceleration, the magnitude of the speed, and the number of stops. These profiles are produced by different organizations and countries, like Environmental Protection Agency [106] or United Nations Economic Commission for Europe [107], amongst others.

To cycle a battery, the velocity has to be converted into power. In this thesis, drive cycles will refer to the power *vs.* time profiles. The drive cycles most frequently used in this thesis are:

- **LA92**, also referred to as the California Unified Cycle and Urban Dynamometer Driving Scheduled, **UDDS** for short, is developed to yield similar characteristics as a light-duty vehicle in city driving conditions. This resembles both fast accelerations and varying speeds. Nevertheless they are distinguished by LA92 having a higher top velocity and a more rapid acceleration.
- **US06** is a cycle with higher top speed (current drawn) and with its higher acceleration represents an aggressive driving behavior.
- The federal test procedure, known as **FTP-75**, represents fuel economy of passenger cars driving in cities i.e. not light-duty nor heavy-duty vehicles. It is composed of the first part of UDDS.
- **NYCC** is the New York City Cycle (NYCC) which simulates a low speed urban drive cycle with frequent stops.
- The highway fuel economy test **HWFET**, highway for short, describes a highway driving cycle that is fuel economical (or electricity economical) in the sense that driving conditions are under 60 mph ($\sim 97\text{km/h}$). Additionally, the speed is more constant than for the above driving cycles. This characteristic can be seen from the current *vs.* time graph in Figure 4.1g where the current drawn is relative small, and the regeneration of electricity due to breaking is done less frequent than for the other driving profiles.
- **WLTC** is the Worldwide Harmonized Light Vehicles Test Cycle (WLTC), a cycle for light-duty vehicles composed of both low speed phases with more stops and high speed phases without stops.

Other drive cycles used are Beijing Dynamic Stress Test, Dynamic Stress Test, Economic Commission of Europe, Federal Urban Driving Schedule, Gwacheon-city Urban Driving Cycle, Hybrid Pulse Power Characteristic, New European Driving Cycle, and Static Discharge Test. These cycles will not be described in this report, but can be found in the reports in the literature review.

For a further understanding of the drive cycles, the current and voltage measurements of four drive cycle frequently used in this thesis is depicted in Figure 4.1. This data used for plotting [108] was obtained by cycling a battery pack by the power profiles. However, since EVs require a great amount of power, and lab equipment in general cannot extract these huge power drains and a battery pack is costly, the power profile was scaled down to a single battery cell.

An interesting observation from Figure 4.1, is the alternating positive and negative current. The negative current represents consumption of electricity, for example to accelerate the EV or to maintain a constant speed, compensating for losses due to drag. The positive current is due to regenerated electricity when braking.

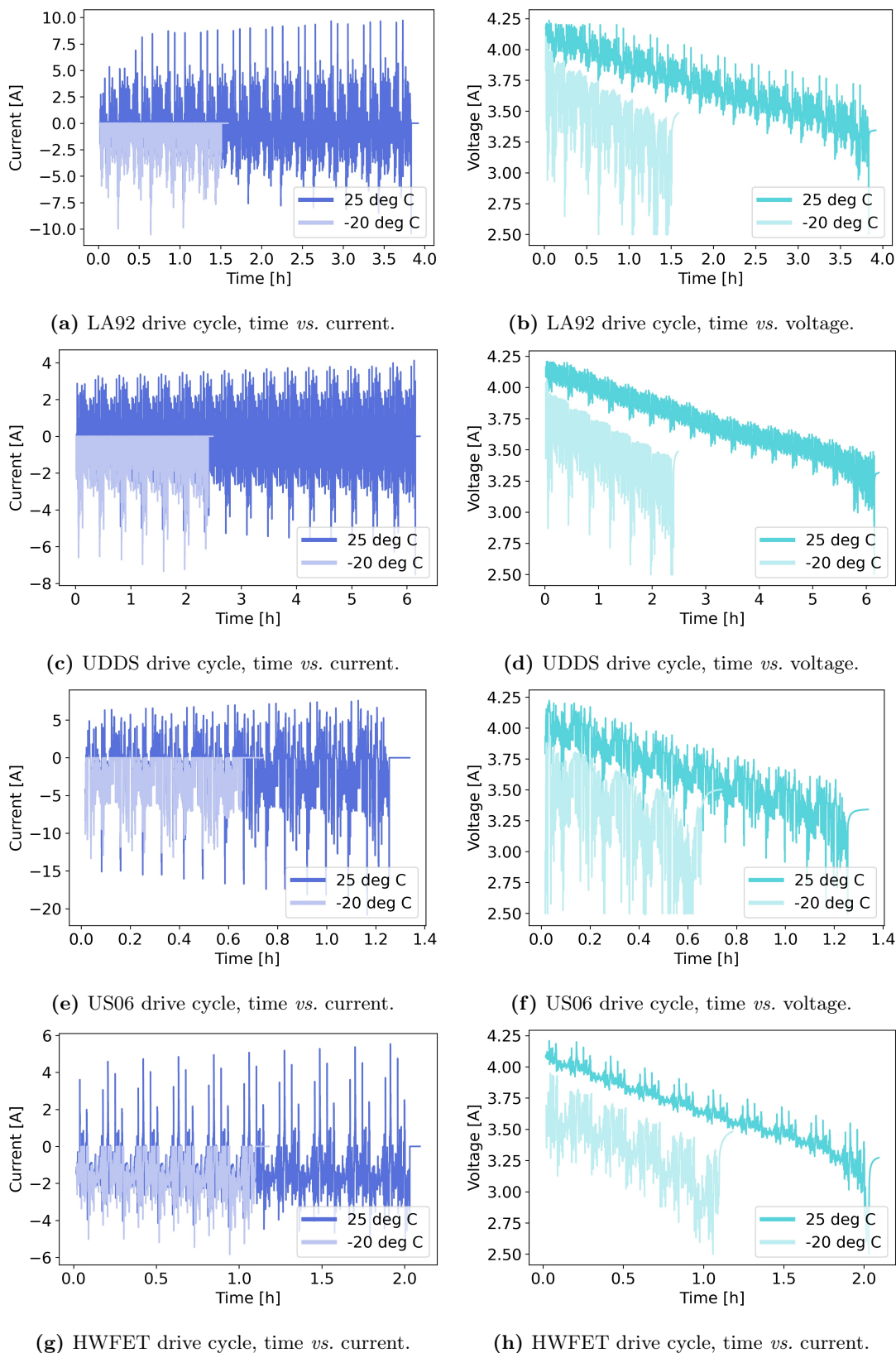


Figure 4.1: Four EV driving profiles in terms of current and voltage *vs.* time. The data is from the Panasonic 18659PF data set [108]. Negative currents indicate that electricity is drawn when discharging, whereas positive currents indicate regeneration of electricity due to braking. The drive cycle tests are terminated the first time the voltage hits 2.5V for 25 °C and at 1.74Ah (60% DOD) when discharged at -20 °C. No regeneration is carried out for -20 °C which explains why only negative currents are present.

Another important observation is found in the voltage profiles (Figure 4.1b, d, f, and h, a). There voltage fluctuate. Therefore, if only an instant time frame of the current, voltage, and temperature is given to an NN without memory (*e.g.*, FNN), it might be hard to estimate the exact SoC, since a range of SoCs can have the same voltage, current and temperature properties at an instant time frame. From Figure 4.1 it can be observed that these fluctuations in voltage are greater with decreasing temperatures. This yields an even larger range in SoC that potentially can be mapped from only knowing the properties of an instant time frame. This upscaled span in voltage fluctuations for lower temperatures can be explained reasoned by Ohm's law $\Delta E = r \cdot i$. When the negative current i is increased, a greater voltage drop ΔE is experienced. For a higher internal resistance r , the voltage drop is even more significant.

Figure 4.1 also reveals the trend between current drawn and the decreasing terminal voltage. As explained in Section 2.7, the terminal voltage in general is lowered by the increased internal resistance. Therefore, a higher current has to be drawn for lower temperatures if the same power should be obtained as for higher temperatures. The capacity is therefore temporarily reduced at lower temperatures. Furthermore, no regenerative braking is done at temperatures below 0°C , indicating a strict reduction in SoC. The combination of increased current rate and no generation of electricity at low temperatures, yields a higher drop in SoC, which is represented by the faster decreasing terminal voltage. An elaborate analysis is found in Appendix A.

4.2 Feed Forward Neural Networks

Some of the state-of-the-art work is presented in this section. The FNN in Figure 4.2 works according to the general NN described above, but the output Y is now the SoC. In the illustration, a mapping is done from the averaged battery parameters voltage and current, and battery temperature, to the SoC. However, other battery parameters might be more efficient as inputs.

An FNN used to directly map battery signals to the SoC was presented by [94] for a hybrid system composed of a 12 V lead-acid battery and a 12 V LFP battery. One FNN was used to estimate the SoC of both batteries at an ambient temperature. The proposed network architecture has relative many input nodes compared to other networks estimating SoC. The reason being, according to the author, there is an insufficient feature space when only using terminal voltage, current, and battery temperature. Therefore, the integral of the battery current and the first derivative of both voltage and current were also used as features. The effect of using temperature measurements (*e.g.*, the ambient temperature and/or the battery temperature), would be an interesting topic for further research. Moreover, the paper did not compare the proposed approach with two FNNs with a single SoC output. A conclusion to whether or not it is beneficial to use the proposed FNN with dual SoC output, cannot be drawn. Further work with a benchmark comparison has to be conducted.

In [68], the authors proposed a scheme to systematically optimize the FNN hyperparameter by an offline optimization algorithm. Among the algorithms used to optimize the FNN structure were the genetic algorithm, particle swarm optimization, artificial bee colony and backtracking

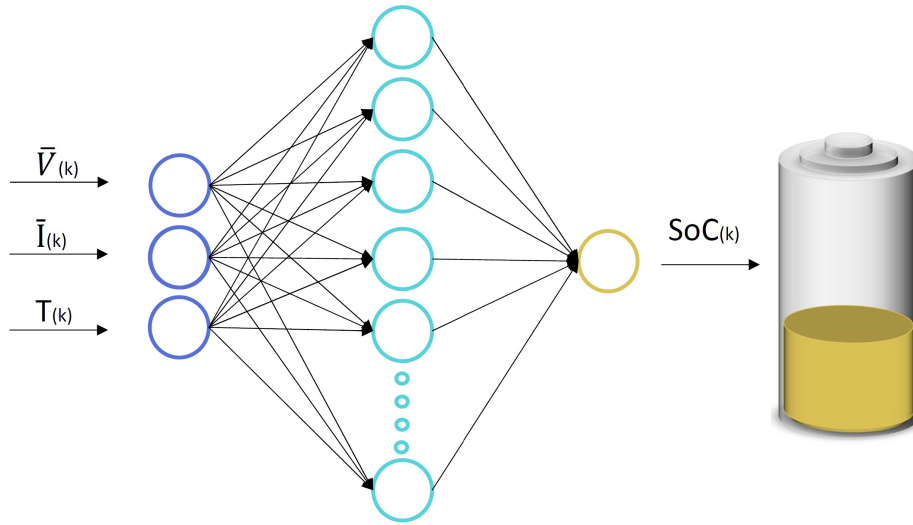


Figure 4.2: An illustration of a simple FNN that directly maps the input vector containing the average voltage (\bar{V}_k), average current (\bar{I}_k) and battery temperature (T_k) into the output; SoC.

search algorithm. The latter was chosen due to its few parameters and, according to the author, easy implementation. The backtracking search algorithm was implemented to determine an optimal number of neurons in the hidden layer and the learning rate value. The optimization and training schemes were divided into four stages. The first stage consisted of collecting data from the two EV drive profiles dynamic stress test and federal urban driving schedule, by cycling an 18650 NMC LiB cell of 3.6V and 2 Ah.

Moreover, the data was filtered and normalized. In the second stage a chosen FNN architecture was constructed to calculate the initial error of the SoC estimation. Based on this primary model, an optimal number of neurons in the hidden layer and the learning rate number were computed based on backtracking search algorithm minimizing the error. For the last stage the suggested model was trained and later tested with data at three different temperatures; 0 °C, 25 °C and 45 °C. The result was that the FNN-backtracking-search-algorithm yielded the lowest RMSE rates. Another noticeable result was that SoC estimation at 0 °C yielded nearly a factor of two higher error than for 25 °C.

The authors in [67] used an FNN to directly map battery signals to SoC for different temperatures. Training data was obtained from their own lab at various temperatures ranging from -20 °C to 25 °C. In contrast to an RNN, which will be discussed later, the FNN does not have a memory. Therefore, in addition to the momentary voltage and ambient temperature, a moving average of the current and voltage were used as features. Consequently, historical data was taken into account with a much simpler model than RNN, still yielding equivalent results to those obtained by RNN. Systematic evaluations of the optimal number of neurons, hidden layers and the size of the sample windows were carried out. Two sample windows consisting of 100 and 400 time steps were investigated. Another finding was a solution to the problem of having a small

data set, which typically leads to overfitting the model. This was overcome by adding Gaussian noise to the training data, resulting in an increased SoC estimation by up to 40% in accuracy and robustness for real world applications. The prime result was obtained at 25 °C where a window of 400 time steps was feasible. Another favorable result, was the network's ability to interpolate between temperatures. The accuracy was high even when tested on a data set where the ambient temperature increased from 10 °C to 25 °C during discharge, which the network had not been trained with. However, the SoC estimations at -20 °C, the MAE was twice as high as for estimations at 25 °C, due to the low temperature effects seen of LiBs as discussed in Subsection 2.5.

Another work where noise was added to the data is [109]. This was done to simulate sensors noise since low-cost sensors typically are used in EVs. The noise added was both a constant offset of +/-110 mA, +/-4 mV, and +/-5 °C for current, voltage, and temperature, and a gain of 2% for the current. This resulted in 14 data sets with noise, also called augmented data sets, used for training and testing. The error scores doubled when trained and tested on augmented data compared to a benchmark model trained and tested on the original data. The work proved that an FNN could estimate the SoC fairly accurate despite the simulated sensor noise. However, an interesting experiment would be to test the benchmark model with augmented data, or test the model trained on augmented data with non-augmented data. By comparing these results to the model trained and tested on augmented data, the potential benefit of only training on augmented data could be demonstrated.

In [110] the authors proposed a methodology to construct an FNN for a LiBs without overfitting it. A performance analysis was also composed, which according to them, follows the best practices in ML. The work was done on two simple deep forward network differentiated by having two and four hidden layers. Only one ambient temperature and one battery was tested, 25 °C and a Panasonic 18650PF LiB, respectively. Moreover, only one driving cycle was trained and tested with. Different regularization techniques were utilized in order to obtain a generalized model. K-fold cross-validation was one of the techniques used to prevent overfitting by creating adequate training, validation, and test sets. Other techniques used were dropout layers and parameter norm penalties. The network with four hidden layers proved to yield the best results in terms of generalization ability due to the capacity of having more layers, as well as the use of regulation techniques. However, they did not look into the consequence of adding more hidden layers, so a universal conclusion of the optimal number of layers that will generalize the network cannot be taken. Since overfitting in general occurs when the network is too complex compared to the problem, and increasing depth yields increasing complexity, it is only up to a point that increasing the number of layers of the network would increase the generalization capabilities. Another remark should be made on the test set. The test set was of the same drive cycle as the training and validation data. If the model was overfitted, the test results might yield a low error. Therefore, an interesting test would be a drive cycle that the model had not been training on.

4.3 Recurrent Neural Networks

The SoC is a time dependent problem, and since Recurrent Neural Network (RNN)s in general have provided good results when modeling sequence data, a lot of research has been done on using RNNs to estimate SoC. This section will provide some of the research.

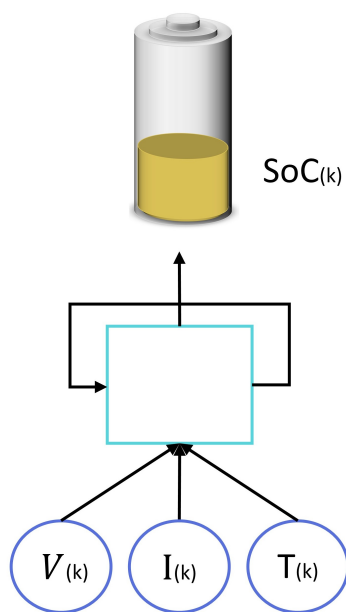


Figure 4.3: Illustrating SoC estimation with RNN. The current time step is inputted to predict the current SoC, implying that no averaging windows are present.

4.3.1 Gated RNNs

In [93], RNN with Gated Recurrent Units (GRUs) was developed to estimate the SoC for three different temperatures with a single model. The model was tested on two public data sets with the vehicle driving cycles federal urban driving schedule, US06 and Beijing dynamic stress test, as well as another high rate pulse discharge condition data set of an 18 Ah battery. All the data sets were measured with three different temperatures of 0°C , 25°C and 45°C . The influence of two hyperparameters, time step and iteration, were evaluated. Furthermore, the size of the training data was also evaluated. The result found was that increasing the number of time steps, number of iteration and the data size, increases the performance of the network. The error was in the same order as for the papers testing LSTM cells, even though GRUs have a simpler set up. Although, a note should be taken on of the reliability of the results due to a lack of structure in the experimental set up (or just imprecisely explained). There are three different data sets that the authors refer to as test sets. However, two of them were actually used for testing, while one set was divided into both training and validation sets. Moreover, it is not clear if the results stated are results from the validation set, or from a test set. Therefore, the testing results seem to be biased.

In [111], the authors constructed a RNN-LSTM network. A novelty in this paper is that the model was tested for varying temperatures ranging from 10 °C to 25 °C, while only training the model on three ambient temperatures: 0 °C, 10 °C and 25 °C. Consequently, the network was tested on temperatures which it had not been trained on, yielding a MAE of 1.66% and thereby proving the networks ability to interpolate between temperatures. This implies that the network can learn a general rule for battery dynamics that it has not seen before, which is an advantage for a limited amount of training data.

Another RNN-LSTM model was constructed in [112]. In this work, transfer learning was thorough investigated yielding a reduced RMSE for all but one out of experiments. This result was obtained when training on the same number of epochs for the reference case and the transfer learning case. However, when the reference case was trained on ten times more epochs than the benchmark case (learning rate of 10^{-4}), only one of four cases yielded lowered MAE, RMSE, and Maximum error. Therefore, no definitive conclusion was drawn.

4.3.2 Other RNNs

Previous work has also focused on RNNs without the gated units introduced in the previous section. The authors in [91], have done a thorough study comparing different learning algorithms as well as optimization methods to modify the learning parameters. Six state-of-the-art ML algorithms were explored to estimate SoC, were the Recurrent Nonlinear AutoRegressive with Exogenous inputs (RNARX) yielded the best results. RNARX is a subgroup of RNN used for complex time-series problems, using one or more feedback loops. The optimization methods were used to tune the three hyperparameters: input delays, feedback delays and number of hidden neurons. Of the four methods that were explored, the lightning search algorithm outperformed the others including backtracking search algorithm, particle swarm optimization, and gravitational search algorithm. The training and testing was done on an NMC battery cell with two EV driving cycles, federal urban driving schedule and dynamic stress test on the different temperature operating conditions of 0 °C, 25 °C and 45 °C. The effect of noise and the effect of aging was also explored. The data sets were divided using cross-validation into training and testing with a 70:30 ratio. It would be interesting to also test the model with a data set of either a different driving profile or different temperatures than the ones used for training. Nevertheless, the authors have conducted a good compassion analysis and the result of which algorithm is most efficient can be of importance for further investigations. Preferably, also other tuning parameters like the initialization of weights and biases, minibatch size and cost function optimizer could beneficially be taken into account to increase training efficiency.

4.4 Radial Basis Function Neural Network

Radial Basis Function Neural Network's (RBFNN's) have not solely been used to calculate the SoC directly, but also as a help for adjusting battery model parameters, parameters in the Kalman filter and to quantify different error measures of parameter uncertainties. In [113] a RBFNN was presented to adjust the parameters of a sliding-mode observer by learning the uncertainty

dynamics of a LiPo battery ECM offline. With respect to Figure 3.14, the input x is the estimated state vector consisting of the terminal voltage (V_t), the estimated SoC from the ECM, and V_{pe} and V_{pc} referring to the voltage across the polarization capacitors as a function of the electrochemical polarization and the concentration polarization respectively. The output y is the updated upper bound of the system uncertainty. Once the network was trained, the parameters of the sliding-mode observer could be adjusted in real time indirectly by the RBFNN. This led to less scattering effects in the SoC estimation and enabled tractability.

An RBFNN where the SoC was directly output from the network is presented in [92]. A thorough work was done on the network's ability to estimate the SoC throughout the LiB's lifetime. Up to 700 cycles were tested, yielding a decrease in SoH by 15%. An emphasis was therefore put on the total capacity C_t of the LiB. The input parameters of the network are distinguished from the previously presented NN that directly estimates the SoC, by inputting the total practical capacity and not inputting the current. For that purpose, a regression model estimating C_t over a range of cycles and temperatures between 10 °C and 40 °C was made. In order to obtain the points of the practical capacity to fit the regression model, an integration was done of the current over a full cycle.

The proposed RBFNN was compared to an FNN with the same hyperparameters, resulting in that the RBFNN performed 29% better than the FNN after 100 full cycles. This result got more apparent with increasing aging, drawing the conclusion that the RBFNN was better suited for predicting SoC when aging occurs. The method was also compared to extended Kalman filter, where the extended kalman filter yielded a 50% reduction in error compared to the RBFNN. Thus higher estimation accuracy was found for kalman filtering. However, according to the authors, accurate models and system parameters are needed when using the extended kalman filter, limiting the extended kalman filter's use for a wide range of applications. The results were found only for 50 aging cycles, thus another interesting result would have been how the models performed with respect to each other when the aging cycles are majorly increased. The final result of the RBFNN was a MAE over all aging cycles and temperatures of below 5% SoC. Even though this is a much higher error than for the previously reviewed articles, the testing set might have been more challenging. This is reasoned in the fact that the authors have only trained on data sets with 50 aging cycles, while tested for 100, 400 and 700 aging cycles. Since aging of the LiB leads to different battery dynamics, the testing data might be very different from the training data. Consequently, the model had to extrapolate in order to estimate the SoC for the aged battery. Another reason could be that the model lacks vital information to estimate the remaining capacity. The the terminal voltage is inputted, but is generally fluctuating during cycling and since no information about the current draw is given, the model might have a hard time learning the battery's SoC.

4.5 Time Delayed Neural Network

A time delayed NN was constructed by the authors in [114], where a focus was put on determining the optimal number of input time delays and hidden neurons. These hyperparameters were determined by the improved firefly algorithm, which was compared to two other optimization

algorithms. Furthermore, the time delayed NN was compared with other NN's based on six different error measurements, yielding the conclusion that time delayed NN preforms with best accuracy. Two battery types were investigated, NMC and NCA where both cycled under two different cycles at room temperature: static discharge test and hybrid pulse power characterization. Other EV drive cycles and temperatures were also used to test the model's accuracy. From reading the article, it seems like the training was done only at the room temperature and for only two different battery cycles². The RMSE was stated to be less than 1% and the MAE less than 0.8% when tested on dynamic stress test, federal urban driving schedule, and US06. Accordingly, the results achieved by testing with these three profiles on different temperatures are very impressive in terms of the models ability to extrapolate the temperature, as well as understanding battery dynamics of more complex driving cycles than it was trained on. Another test that was performed was on aging cycles. The NMC battery was cycled up to 600 times yielding a estimated SoH of 88%. Since a poor SoH impacts the battery's performance and dynamics, a SOC estimation error below 2% when using RMSE metrics validate the robustness of the model.

4.6 Discussion and Analysis of the Literature Study

This section summarizes the findings and discuss the short-falls of the literature. The motivation for this discussion is to improve the foundation for evaluating the ML approaches to estimate SoC. Furthermore, an FNN was built in order to better evaluate the findings found in the state-of-the-art literature by experimenting with different network structures and testing sets. However, the FNN was only built and trained for the purpose of comparing results and not to create the most robust or best SoC estimator. A description of the FNN with its results is given in Appendix A.1.1.

An overview of the articles discussed above is given in Table 4.1. In order to yield a fair comparison between the models, the error obtained is at an ambient temperature of 25 °C if not stated otherwise. All the ambient temperature the model was trained for is also stated, but the results for each temperature is not. If both MAE and RMSE were provided from the articles, both were included in the table. The range between the lowest and highest error is indicated by "-" when testing was reported for more than two drive cycle. The table can be used as an indicator of how robust the models are, but should be carefully reviewed in terms of not only looking at the accuracy results, but also the other columns of the table. In the sections below, the findings are discussed in more detail.

²An e-mail was sent November 24, 2020 to the author for confirmation if the assumption regarding the test set is true. However, no response was obtained by the date this report was handed in, 21st June 2021.

Table 4.1: Overview of SoC estimation done by various NNs. The ambient temperature the error result is obtained at is 25 °C, or at the ambient temperature closest to 25 °C. All error scores have the unit % SoC.

Method	Testing Profiles	Training Profiles	Error	Input	Output	Ambient Temperature	Battery	Neurons per hidden layer
FNN [94]	WLTC	FTP75, NEDC, US06 GUDC and Highway	RMSE 0.33 MAE 0.27	$\psi = [V_i(k), I_i(k), T_i(k), V'_i(k), I'_i(k), V_{ia}(k), I_{ia}(k), T_{ia}(k), V'_{ia}(k), I'_{ia}(k), \sum I_{ia}(k)]$	$Y = [SoC_{ti}(k), SoC_{ia}(k)]$	23 °C	LFP (12 V, 8 Ah)	(4, 2)
FNN w/BSA [68]	70% of FUDS 70% of DST	30% of FUDS 30% of DST	RMSE 0.81 RMSE 0.91	$\psi = [V_{avr}(k), I_{avr}(k), T_{avr}(k)]$	$Y = [SoC(k)]$	0 °C, 25 °C, and 45 °C	NMC (3.6 V, 2 Ah)	(14)
FNN [67]	US06 HWFET	Cyle 1-4, UDDS, LA92, NN	MAE 0.84 MAE 0.61	$\psi = [V(k), T(k), V_{avr}(k), I_{avr}(k)]$	$Y = [SoC(k)]$	-20 °C, -10 °C, 0 °C, and 25 °C	NCA (3.6 V, 2.9 Ah) [108]	(8, 16, 32)
FNN [109]	US06, LA92, UDDS, and HWFET (both batteries)	Random mixes of the four test sets	MAE 0.66, RMSE 1.10 MAE 0.85, RMSE 1.14	$\psi = [V(k), T(k), V_{avr}(k), I_{avr}(k)]$	$Y = [SoC(k)]$	-20 °C to 25 °C -20 °C to 45 °C	NCA (3.6 V, 2.9 Ah) [108] NMC (3.6 V, 3.6 Ah) [115]	(55, 55)
FNN [110]	20% of NN *	80% of NN	MAE 1.60	$\psi = [V(k), T(k), I(k)]$	$Y = [SoC(k)]$	25 °C	NCA Panasonic NCR18650FP (3.6 V, 2.9 Ah) [108]	(256, 256)
RNN GRU [93]	Panasonic: HWFET, UDDS, LA92, US06 Samsung: 50% of FUDS, US06 and BJDST	Panasonic: Cycle 1-4, NN Samsung: 50% of FUDS, US06 and BJDST	MAE 0.32-0.82 MAE 0.86-3.17	$\psi = [V(k), T(k), I(k)]$	$Y = [SoC(k)]$	0 °C, 10 °C, 25 °C, 40 °C (Panasonic) 0 °C, 25 °C, and 45 °C (Samsung)	NCA (3.6 V, 2.9 Ah) [108]	(1000, 50)†††
RNN LSTM [111]	10-20% of random combinations of HWFET, UDDS, LA92 and US06	80-90% of random combinations of HWFET, UDDS, LA92 and US06	RMSE 1.11 MAE 0.77	$\psi = [V(k), i(k), T(k)]$	$Y = [SoC(k)]$	0 °C, 10 °C, and 25 °C	NCA (3.6 V, 2.9 Ah) [108]	(500)
RNN LSTM [112]	US06, UDDS, and LA92	random mixes of the test profiles and HWFET	RMSE 2.50 RMSE 1.62 RMSE 2.56 RMSE 1.22	$\psi = [V(k), i(k), T(k)]$	$Y = [SoC(k)]$		NCA (3.6 V, 2.9 Ah) [108] LG (3.6 V, 3 Ah) [115] Samsung 48C (4.8 Ah) Samsung T30 (3.6 V, 3 Ah) [36]	(10)
RNN (RNARX) [91]	30% of SDT and HPPC 30% of 1 C SDT, HPPC 100% DST, FUDS, US06	70% of SDT and HPPC 70% 1 C SDT, HPPC 70% 1 C SDT, HPPC	RMSE $\in [0.58-0.85]$ RMSE $\in [0.31-0.79]$ RMSE <1, MAE <0.8	$\psi = **$	$Y = [SoC(k)]$ $Y = [SoC(k)]$	Room temperature 0 °C, 25 °C, and 45 °C	NCA (3.6 V, 3.2 Ah) NMC (3.7 V, 2.6 Ah) NMC (2.0 Ah)	(18)
RBFNN [113]	UDDS, Highway ****	UDDS, Highway ****	RMSE $\in [3.32-3.33]$	$\psi = [V, SoC, V_{pe}, V_{pe}]$	$Y = [\text{lupper boundary of system uncertainties}]$	Room temperature	LiPo Turnigy (3.7 V, 0.5 Ah)	(9)
RBFNN [92]	UDDS and ECE @ 50 ageing cycles	UDDS and ECE @ 100, 400 and 700 aging cycles	MAE 2.4% ***	$\psi = [T(k), V(k), SOH(k)]$	$Y = [SoC(k)]$	10 °C, 25 °C, and 40 °C	LMO (6 Ah)	(**)
TDNN w/iFA [114]	NMC: 30% of SDT, HPPC NMC: DST, FUDS, US06 NCA: 30% of SDT, HPPC	70% of SDT, HPPC 70% of SDT, HPPC 70% of SDT, HPPC	MAE [0.31-0.79], RMSE [0.58-0.85] MAE < 0.8, RMSE < 0.9 MAE [0.15-0.33], RMSE [0.58-0.85]	$\psi = [V(k), T(k), i(k)]^\dagger$	$Y = [SoC(k)]$	0 °C, 25 °C, 45 °C	NMC (3.7 V, 2.6 Ah) NMC (3.6 V, 2.0 Ah)†† NCA (3.6 V, 3.2 Ah)	(18)

* The test set is further divided into validation and test set. The ratio is unknown.

** Not stated.

*** MAE after 100 ageing cycles.

**** Unknown train-test split ratio.

† No explicit information was given about the input. However, measurements of current, voltage and the battery's temperature were done. Moreover, it was stated that no filtering of the measurements was done. It is therefore assumed that these three parameters were inputted directly to the network.

†† NMC (2.0 Ah) was used for testing. This report assumes that NMC (3.6 V, 3.2 Ah) was used for training.

††† The hidden layer of 1000 nodes was not fully connected.

4.6.1 Data Analysis and Comparison

The accuracy of data-driven methods depends on the quality and quantity of the data. If the training is only done with a limited amount of data, the ML model might not be able to generalize. Moreover, the amount used for testing would be small, and from a statistical point of view the final accuracy result using limited data could be biased [116]. In addition, the quality of the data in terms of testing a large variety of battery conditions and dynamics, is also important to generalize the network which would in turn yield a high accuracy. Since the literature have used different drive cycles, LiB chemistries, temperatures and SoH, it is challenging to give a set answer to which model that is the best to use in order to estimate the SoC in LiBs. Still, some distinctions on how reliable the accuracy results are in terms of robustness can be made.

Since the different literature can choose their own standards for data partitioning for training and testing, it is hard to compare and distinguish the different studies on the networks adaptability and robustness. The only general rule is that the testing has to be done on unseen data. In addition, a norm is that the testing and training data should be done with data sets that to a degree differ from one another. Often in ML a partition ratio is used, where a part of the data is put aside for the purpose of training, while the other part is used for testing. Examples of random splitting of the data are [110], [111], [93]³, [91]⁴ and [68]. In the latter, the ratio of training and testing was 30 to 70. Consequently, 30% of the data has been put aside for testing, implying the model will not be trained on those 30% and therefore the testing is done on unseen data. However, there is no guarantee that the 30% data that is randomly put aside for testing is differing from the other 70% used for testing. On the contrary, it is likely that the data sets are very similar and that, if the network is overtrained, the test data will not detect this overtraining and still yield a low error result.

To support this reasoning, an FNN was built using the Scikitlearn [117] library from Python [118]. The data set used was the Panasonic NCR18650PF [108]. The goal of the experiment was only to evaluate how different data partitions influence the testing accuracy. The data set was divided with a split ratio of 70:30 for training and testing, respectively. This yielded MAEs as low as 0.17% SoC when trained 400 times on a network with layers of 128 and 64 nodes in each layer. Moreover, the same network was trained with the same data set, but by picking out nine of the ten drive cycles⁵ for training. In general, training on more data yields better models due to the networks ability to generalize from a larger data set. However, when the FNN was trained for 9/10 of the drive cycles (instead of 70% of the data), the error increased. When training for 400 epochs the error ranged from 4.10% SoC to 15.92% SoC depending on the drive cycle. Smaller errors were recorded when training on a more shallow network of one hidden layer with 16 nodes, where the lowest error recorded was 1.47% SoC. This is still a magnitude higher than the errors obtained from splitting the training data into a 30:70 training testing ratio. An extensive table of the results from the experiment where different epochs, number of layers, and number of nodes

³The method of partitioning data based on a split ratio was only used for one of the two data sets in this paper.

⁴A test result with new driving profiles were also stated for one of the two batteries.

⁵Cycle HWFET was done with two separate recordings for 25 °C and named HWFETa and HWFETb. The cycles used for training were cycles 1-4 and the NN.

were used, are found in Table A.1. Moreover, a more detailed description of network, results, and discussions are found in in Appendix A.1.1.

The reason why the split of 30:70 ratio could yield a similar data set for training and testing even though the model had never seen 30% of the data, could be that the sampling frequency of the data set is 10 Hz. As a consequence, the battery parameters changes minimally and the data points that are sampled right after another have very similar values. Therefore, by only taking out randomly 30% of the data for testing, the test set is similar to the training set. If the training data represented all possible drive cycles, it would have made sense to use a split ratio to test it because the network got to train on all of the different scenarios. However, only a small subset of all the possible drive cycles were included in the Panasonic data set. Therefore a distinct test set is needed to check if the model has learned the general battery dynamics and could estimate an unseen drive cycle. To push this point to the extreme, it would be no point in having a SoC estimator that could only estimate the SoC when the drive cycle had the same acceleration and velocity as the LA92 driving profile. For the SoC estimator to be functional for real world applications, it has to estimate the SoC with high accuracy independent of the driving pattern.

Instead of randomly partitioning the data by a given ratio, a few selected data sets should be applied for testing. This was done in [94], [67], [93], [91], and [114] where a selected subset of drive profiles were used for testing. Other criteria for selecting different data sets was testing on a different temperature than used during training. A good example of this is [67] and [111] where the model had to interpolate between two temperatures. An even more challenging example is given in [114], where the model had to extrapolate to estimate a new temperature. Another way of testing the models robustness, which was done in [92] is to test for batteries with a different SoH than the ones used during training. Furthermore, the robustness could be tested by adding noise to the testing data, which was done in [67]. Ideally, all of these tests should be done in order to test the robustness of a model. However, the data sets are usually limited due to the time and effort required to obtain the data. Therefore, a careful evaluation should be done by the data scientist on which criteria the network should be tested on, such that an effort is put on collecting data of essential real world challenges that the network would experience if applied for a real driving situation.

4.6.2 Optimization of Hyperparameters

Overfitting often roots in constructing too complex networks, meaning too many layers and nodes. Optimization algorithms to adjust hyperparameters like the learning rate, the number of neurons in the hidden layers, input time delays and feedback time delays were explored in [11], [68], and [91]. A further research on optimization algorithms would be valuable in order to more efficiently create ML models and to a greater extent be confident on obtaining the hyperparameters yielding the most robust network.

4.6.3 Battery Chemistries

Different battery chemistries are used in the different papers. Since the battery chemistry is important for the voltage window, degradation, power and other operation conditions, the relationship between the SoC and the measured parameters are not necessarily equally easy for a ML algorithm to estimate. As an example, LFP has a flat voltage curve and it would therefore be reasonable to assume that the ML model has a harder time to estimate the SoC for a LFP battery, than for instance a NMC. The cycle life of the different chemistries also varies a lot where LiPo has the lowest cyclability for the battery's from this literature review. Therefore, LiPo would be more vulnerable to degradation changing the battery's properties which in turn would make it harder to estimate the SoC after given number of cycles.

4.6.4 Accuracy Methods

The stated error measurements are calculated by different functions. The different error functions used varied between MAE and RMSE, which cannot directly be compared to one another.

4.6.5 Possible Errors

Several errors are related to SoC estimations by NN and one of those errors is the error measurement in itself. The SoC error is calculated by the target SoC and the predicted SoC. Since the target SoC is an estimate, the error calculation is subject to an inaccuracy. For instance if the target SoC is calculated by Coulomb counting, the error can be associated to the initial SoC, measurements inaccuracies, uncertainties in value of the charge/discharge γ and the total capacity. As an example, the total capacity inputted in the calculation could be the total capacity given from the battery manufacture which varies slightly between each production batch. Moreover, the total capacity will decrease due to aging. For instance, in the public data set of the NCA Panasonic battery [108], the total capacity stated by the manufacture is 2.9 Ah. However, a reference capacity test performed at 1 C at 25 °C was done after approximately 110 cycles. This showed that the capacity had fallen from 2.8 Ah to 2.3 Ah, corresponding to a capacity fade of 18 %. In the articles where testing was done with the Panasonic data set, it was not written whether or not this capacity fade was accounted for.

An article that explicitly stated that the degradation of total capacity was taken into account was [92]. To estimate the total capacity, the current was integrated for a full cycle. However, this requires that the battery is fully charged and fully discharged, which decreases the LiBs capacity and the test is therefore not ideal to execute. Other estimation techniques also exist to estimate the SoH by characterization cycles and impedance measurements [119, 120]. To correctly obtain an accurate target SoC, the SoH should be taken into account.

Another factor that would influence the accuracy result, is testing the model on real world data instead of drive cycles created in the laboratory. The laboratory data is artificially created drive cycles constructed with the goal of representing real driving cycles. Still, they are only simulations generated under controlled conditions, and the patterns of these cycles can only to a certain extent represent data from the real world. Moreover, in a laboratory the current and

voltage values are usually obtained with high quality sensors and little noisy environments, while data obtained from an EV would most likely be of poorer quality. To represent this, Gaussian noise was added to the data in [91] yielding an increase in the error of only 5 % SoC . However, the Gaussian noise added on the data, might not represent the real-world noise. Another finding was done in [67], where the Gaussian noise only was added to the training data, and it was recorded that the model performed up to 40% better on the testing data when trained on noisy data.

4.6.6 Terminology

A large challenge in any research field, especially the fields in rapid development and popularity like ML, is the consistent use of terminology. The challenge discovered by this report is the confusion between validation and testing. As written in Section 3.3.3, the best practice in ML is to divide the data set in three: training, validation and testing. These three data sets should be used in the three stages that have the same name as the data sets. The training stage and validation stage are alternating. In contrast, the testing stage is only done in the end as a final evaluation of the model. Generally in science, the term "to validate" is used to evaluate and verify an experiment. Therefore, from a scientific perspective, it would not be wrong to write that a model is validated when referring to the final evaluation stage. However, in ML, the validation is a part of the experiment (training of the model). Accordingly, from a data scientist's perspective, it is confusing when the word validation is used instead of testing. With one exception [92], all the papers in this literature review are either consequently referring to validation as the testing procedure, or using the two terms interchangeably. To reach an audience that is specialized in ML, it is important to be consistent with this use of terminology.

4.6.7 Network Types

Both RNN and FNN seem to yield low accuracy results. A potential advantage with RNN with gated units is that the voltage, current and temperature of the battery can be inputted directly into the network, and therefore the RNN extract the most important characteristics itself. More explicitly, the RNN will remember values over an arbitrary time interval whose length is optimized during training. On the contrary, when training an FNN with averaged values the time interval have to be decided in advance of the training. Therefore the scientist's assumptions will have a larger influence on the final estimator. The advantage with the FNN is however that the structure is simpler and that the SoC estimator is quicker due to less computational operations. From Table 4.1, is hard to distinguish whether the FNN, RNN, RBFNN or TDNN is the most adequate to use. The accuracy result is dependent on many factors like the hyperparameters, the choice of training and testing data, battery type, input features and the error function.

4.7 Summary of the Literature Study

A literature study was conducted to evaluate various NNs ability to estimate the SoC for LiBs in EVs. All the results from the literature studies state an RMSE less than four percent, where most of the MAEs are around one percent. It is challenging to draw any conclusion of the best model

due to the many factors that have to be compared. The factors include the choice of different hyperparameters, drive cycles, diverse testing procedures, error functions, battery chemistry, and inconsistent terminology resulting in the need for the reader's interpretation.

A doubt is put on the testing procedure in some of the literature, more precisely randomly partitioning of data with a set split ratio. Therefore, an FNN was constructed to evaluate and compare the accuracy measurements' validity. Based on those results, it is clear that the more variety in the testing data compared to the training data, the more reliable the test result is in terms of testing the models' generalizability and robustness. The different partitioning of the data for training and testing roots in the lack of a standard testing procedure in ML. The field of estimating the SoC with ML would benefit from a standardized partition of the training and testing data with important criteria for testing the robustness such that it can be applied for real-world applications.

The suggestion for improvement is to not split the training and testing randomly by a set ratio. This is reasoned by the battery data usually being sampled at a high sampling rate with little change in the battery parameters. To elaborate, the randomly splitting of the data will yield similar training and testing data. Even though the data is unseen, the model is not tested for variations in the data set yielding a large uncertainty of the model's robustness. Consequently, the model could be overfitted or overtrained and still obtain a great accuracy result. Having established that the random partitioning of data is not advisable, it can be concluded that a good testing practice is using other drive cycles, temperatures, and/or age cycles than used in training, depending on the focus of the research. Moreover, for the model to be used in physical applications, the tests should not only be constructed in the laboratory, but be of real-world drive cycles.

Data Acquisition and Preprocessing

This chapter is about the procedure for acquiring battery data, limitations of the hardware, software, and the instrumentation used for battery cycling, and how the data was preprocessed prior to constructing the NNs. To start with, data from public databases is treated, then data sampling from the experiments in this thesis is explained and discussed.

5.1 Data Acquisition from Public Database

A few researchers have made data sets public such that other data scientists can develop and test SoC models with identical data. This enables a less biased comparison of models. The research team that has contributed the most to public battery data is from the University of Wisconsin-Madison, where the data sampling was done by Dr. Kollmeyer. The data was published on Mendeley Data [121] which is a cloud-based repository for researchers to store and share data. Kollmeyer *et al.* have published data sets of four different LiB manufacturers; Panasonic [108], LG [115], Samsung [36], and Turnigy [122]. All four data sets are used in this thesis and will be referred to by the brand of the battery.

Moving on to the process of how they obtained the battery data, cycling was done on a single battery cell. The drive cycles are, as described in Section 4.1, made to simulate the dynamics and operating conditions of EVs given by a power time profile. In the Panasonic data set, the EV is representative of an electrical F150 truck with a 35kWh battery pack, but scaled for a single 2.9Ah Panasonic 18650PF cell. For the other data sets, the exact vehicle is not stated, but referred to as a compact EV. Moreover, an 8 cu.ft. thermal chamber was utilized and a temperature sensor on the battery with +/- 1 °C accuracy. A Digatron Firing Circuits Universal Battery Tester was used to cycle the battery, where the channels have a maximum limit of 75 A, and 5 V, with a measurement accuracy of 0.1% of full scale for both current and voltage. The drive cycles were sampled at a frequency of 10 Hz, while the charging and pauses were sampled at 1/60 Hz. Each battery was new when the first cycling began. Information about the four data sets are found in Table 5.1

Table 5.1: Battery description of four batteries cycled to obtain battery data used to create SoC models. As a reminder of the nomenclature, i is current, C_t is the total capacity, V_n is the nominal voltage, and V_{cutoff} is the lower cutoff voltage ($=V_1(T)$). Moreover, E/m is the energy density (energy (E) per mass (m)).

Battery	Source	EV type	Temp range [celsius]	CCCV cut off i [mA]	Chemistry	Cn [Ah]	Vn [V]	Vcutt [V]	Energy density [Wh/kg]	Cell geometry
Panasonic 18650PF	[108]	EV Ford F150 truck	-20 - 25	50	NCA	2.9	3.6	2.5	207	Cylindrical
Turnigy Graphene 5Ah 65C	[122]	"compact EV"	-20 - 45	5	LiPo	5	3.7	2.8	134	Prismatic
LG 18650HG2	[115]	"compact EV"	-20 - 45	50	NMC	3	3.6	2	240	Cylindrical
Samsung IN21700-30T	[36]	"compact EV"	-20 - 45	10	NMC	3	3.6	2.5	166	Cylindrical

The first battery Kollmeyer *et al.* sampled was the Panasonic battery [108]. This is reflected by a slightly different method for data acquisition. For instance, the battery data is cycled for five temperatures: -20°C , -10°C , 0°C , 10°C , and 25°C , while an additional temperature of 45°C is found in the three other data sets. In addition, the cutoff limit is dissimilar, where the Panasonic data set has a cutoff limit when the terminal voltage reaches 2.5 V at 25°C and 10°C , and after 2.32 Ah (80% DoD), 2.03 Ah (70% DoD), and 1.74 Ah (60% DoD) have been discharged at 0°C , -10°C , and -20°C , respectively. In contrast, with the three other battery sets, the limit is set to when 95% of the 1C discharge capacity at the respective temperature has been discharged. Moreover, no regeneration is done for the Panasonic battery when the temperature is beneath 10°C , while for the other three batteries a limited regeneration is allowed. This is to prevent premature aging of the cells. For the same reason, the charging cycles are done at room temperature. The last comment on differences is that the Turnigy battery has no drive cycles at -20°C due to poor behavior at this temperature.

Despite the difference in cycling, resemblance is found in the drive cycles used for cycling. UDDS, HWFET, LA92, and US06, are present in all four data sets. The cycles were repeated until the respective cutoff voltage limits were reached. In addition, all data sets have time-shifted parts of the four drive cycles named Cycle 1-4 and NN for the Panasonic data and Mixed 1-8 for the three other data sets. Moreover, the batteries are charged with a Constant Current Constant Voltage (CCCV), with a 1C rate to 4.2V, with cutoff currents as specified in Table 5.1.

5.2 Pre-Processing Data from Public Database

5.2.1 Limitations of the Panasonic Data Set

Several of the charging files are missing in the Panasonic data set. The charging data that exist is stated in Table 5.2¹. The paring of the drive cycles with their respective charge file was done manually based on the date and time stamp of the files.

¹In total there should exist nine charging files related to the drive cycles (LA92, HWFET, US06, UDDS, NN, and Cycle 1-4) for each of the five temperatures.

Table 5.2: Overview of the CCCV profiles that are included in the Panasonic data set with a drive cycle connected to them, are stated.

Ambient temperature	Drive cycles that have a corresponding CCCV charging cycle
25 deg C	HWFETa, HWFETb, LA92, UDDS, US06, NN, Cycle2, Cycle3, Cycle4
10 deg C	Cycle1, Cycle3, Cycle4
0 deg C	HWFET, UDDS, US06, Cycle1, Cycle3, Cycle4
-10 deg C	NN, Cycle1, Cycle2, Cycle3
-20 deg C	US06, Cycle1, Cycle2, Cycle3, Cycle4

5.2.2 Data Cleaning

Data cleaning refers to manipulating inconsistencies in the data as noise, missing values, and/or outliers. To start with, data cleaning was conducted by visual analysis. It was done by plotting input features and SoC against its sampling steps. Outliers in the voltage profile were observed with voltages down to 0 V. The discharge cut-off limit for the four batteries is 2.5 V or higher. This is usually a soft margin, and some deviations are acceptable. However, large offsets are fatal for the battery. Therefore, it was concluded that these measurements are unrealistic and a rough cleaning was done where all voltage measured below 2 V were deleted.

Moreover, some missing files and files with missing current measurements were also found. The files this applies to, and how the errors were handled, are stated in Table 5.3.

Table 5.3: Corrupt or missing files.

Error and error handling	Data set	T [°C]	Cycle
Files are missing.	LG	10	LA92
	LG	0	Mixed 1
	Turnigy	-20	All drive cycles
Current values are only zero. Files deleted.	LG	45	Charge no. 10
	LG	25	HWFET
	Turnigy	0	Charge no. 7
Two cycling exist, one is corrupt and deleted.	Turnigy	25	UDDS
Sudently, inexplicable voltage drop. Coulombic efficiency $\ll 1$. Files deleted.	Turnigy	25	Mixed 2
	Turnigy	-10	Mixed 5
Current and Ah are inconsistent with other measurements for a time window. Removed this time window.	LG	25	LA92

The first six files stated in Table 5.3 were handled before developing the FNNs. However, the last corrupted files were observed post constructing the FNN. They were found since the Maximum error (MAX error) of the Turnigy and the LG data was a consistent, unreasonable high value ($>$

70 SoC %). A hypothesis was made that some of the voltage values corresponded to a low SoC, making the network predict a low SoC, while the actual SoC was high (or vice versa). Therefore, a new processing and cleaning of the data was done to handle the anomalies.

Retraining the models was found too time-consuming, so retraining was not done. The FNNs this concerns are the ones made to evaluate the effect of transfer learning, and since both the benchmark networks and the transferred networks were trained on the data with anomalies, these results are comparable and possible to conclude on the effect of transfer learning. However, models were reloaded and reevaluated with the cleaned data to yield valid test scores, and thereby MAX error scores without the unreasonable high values.

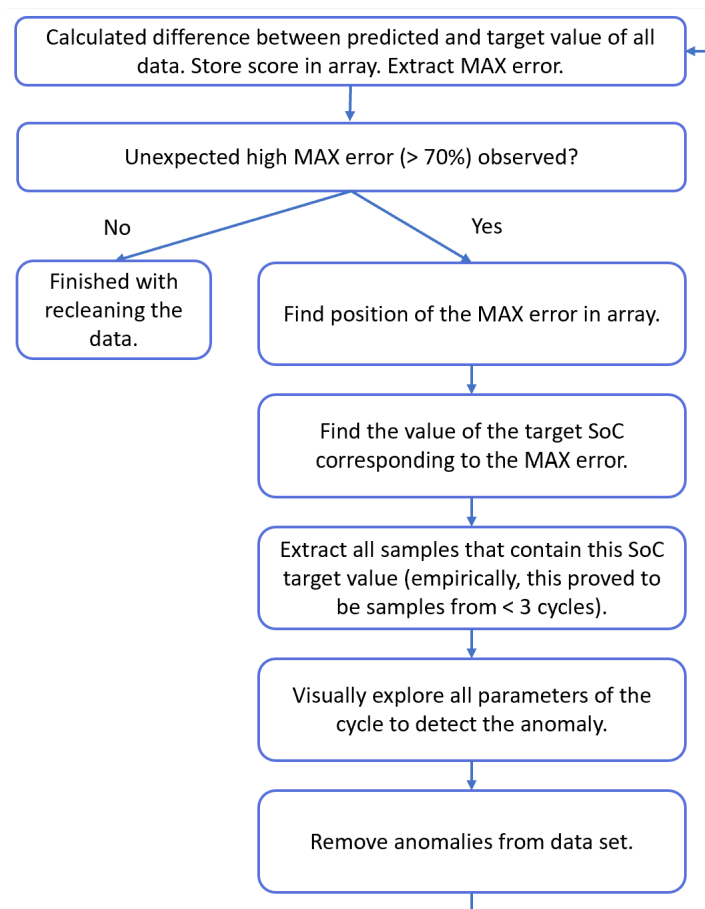


Figure 5.1: Recleaninging the data for anomalies.

Having reviewed the errors present and what was done after they were handled, a step back is now taken to the methodology of how they were eliminated. As a first approach, the sample that yielded the MAX error in the data set was extracted and the sample removed. The flow diagram in Figure 5.1 illustrates this approach.

It should be stated, that in general for ML, samples that yield high error scores should not be taken away without thorough research and confidence that the samples are not just hard to predict, but that anomalies are present. The author of this thesis is confident that anomalies, and not only difficult data, were present. For the skeptical reader, when taking away the anomalies of the Turingy training data (Mixed 2 at 25 °C and Mixed 5 at -10 °C), the MAX error of the training score fell from 97% to 11%.

At a later stage, it was noticed a pattern where the errors occurred. The pattern was found by plotting the difference between each time step as seen in Figure 5.2. Some large deviations in sampling time can be observed by the figure. Some of these spikes are correlated with anomalies in the measurement points, as the samples around $4.85 \cdot 10^6$. One can observe a 5 h time difference between two samplings, and that the voltage has a sudden high value between these points.

Whenever the difference between two samples was larger than 0.1s for drive cycles and 1s for charging cycles, the cycle was evaluated by plotting the respective current, voltage, and temperature measurements. For most of the files, the large time difference was not associated with any anomalies and most likely the cycling had been paused due to an interruption and then started again. These data sets were kept unchanged. The anomalies that were detected are stated in Table 5.3 along with their error handling.

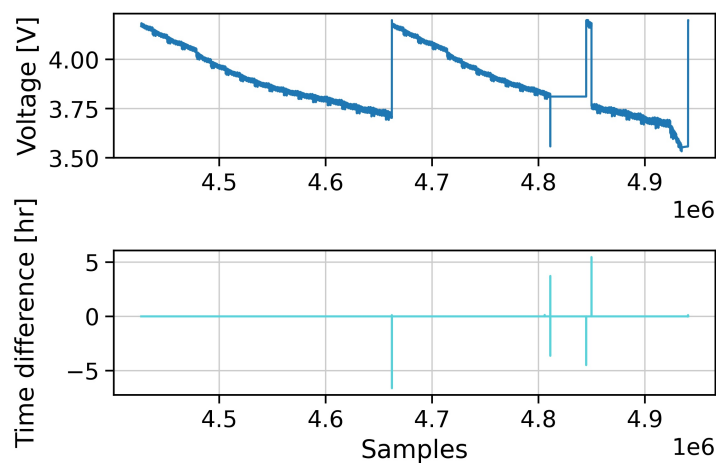


Figure 5.2: Turingy cycled at 25 °C by the UDDS drive cycle. Anomalies detection done by plotting the time difference between each time step. Whenever a high deviation, the measurements around the deviation is researched.

A last data handling was done for the Turingy data set. There are three cycles where charging was done between 0 °C and 15 °C, as Figure 5.3 illustrates. For the experiments when using the temperature difference between ambient and battery temperature as an additional input feature, these files were removed, since the ambient temperature is difficult to estimate, and it is clear that the ambient temperature is not 25 °C for these files.

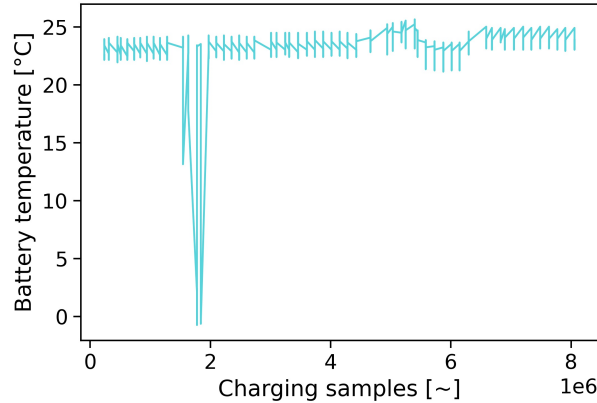


Figure 5.3: Turnigy battery temperature during charging is supposed to be 25 °C (it can be observed that this is off, and it was calculated that the mean charging temperature actually was 23.5 °C). However, some large deviations exist, and the files this applies to are removed in the experiments when using the temperature difference between ambient and battery temperature as an additional input feature. The data is plotted for all samples for all the charging cycles (72 cycles in total).

5.3 Data Acquisition from Cycling

Cycling was conducted to acquire battery data on aged batteries. Three batteries were cycled with, a new one at 100% SoH, and two aged ones at 95% SoH and 77% SoH. The SoH was estimated by Equation 2.12. The cells are aged according to the same method such that the aging mechanisms should be similar. A description of the cells is found in Table 5.4.

Table 5.4: LCO battery specifications

Battery	EV type	Chemistry	CCCV cutoff in [mA]	Cell capacity [Ah]	Nominal voltage [V]	cutoff voltage [V]	Energy density [Wh/kg]	Cell geometry	SoH [%]
Shenzhen Melasta SLPBB042126HN	Chevy Bolt EV	LiCoO2	660	6.6	3.7	3	192	Pouch	100 95 77

5.3.1 Obtaining Drive Cycles

Prior to cycling, drive cycles were converted from velocity to power profiles. This work includes estimating the power required for maintaining a constant velocity when drag forces are present, acceleration, and internal losses in the battery. Moreover, the power required for the whole battery pack of an EV was scaled down to the power required for a single cell. The power modeling of EVs used for this thesis was originally made by Dr. Kollmeyer², but slightly manipulated to include two additional drive cycles. The modeling simulates the lightweight vehicle Chevy Bolt EV.

²It was used with permission of Dr. Kollmeyer by email correspondence.

Six drive cycles were chosen to cycle, namely the four cycles used by Kollmeyer et al. (UDDS, US06, HWFET, and LA92), as well as two more New York City Cycle (NYCC) and Worldwide Harmonized Light Vehicles Test Cycle (WLTC). All the cycles are described in Section 4.1. The two latter are intended to be used as test profiles, while the other four for training, as well as a mix of them (a different mix than the one from the public database). The purpose of only using NYCC and WLTC to test with is that the NN is not trained on any part of the test profiles, as had to be done in the public data set since the mix consisted of the test profiles. Most likely, the mixes are so different from the cycles they were made from that the network is tested on new data set. However, to be on the safe side, NYCC and WLTC are never used to create the mixed training data. The raw data from cycling can be obtained by scanning the QR code in Table 5.4.

5.3.2 Description of Equipment and Lab Set-up

The data was acquired by lab equipment at NTNU. A schematic of the lab set-up is depicted in Figure 5.4 and the equipment's specifications are stated in Table 5.4.

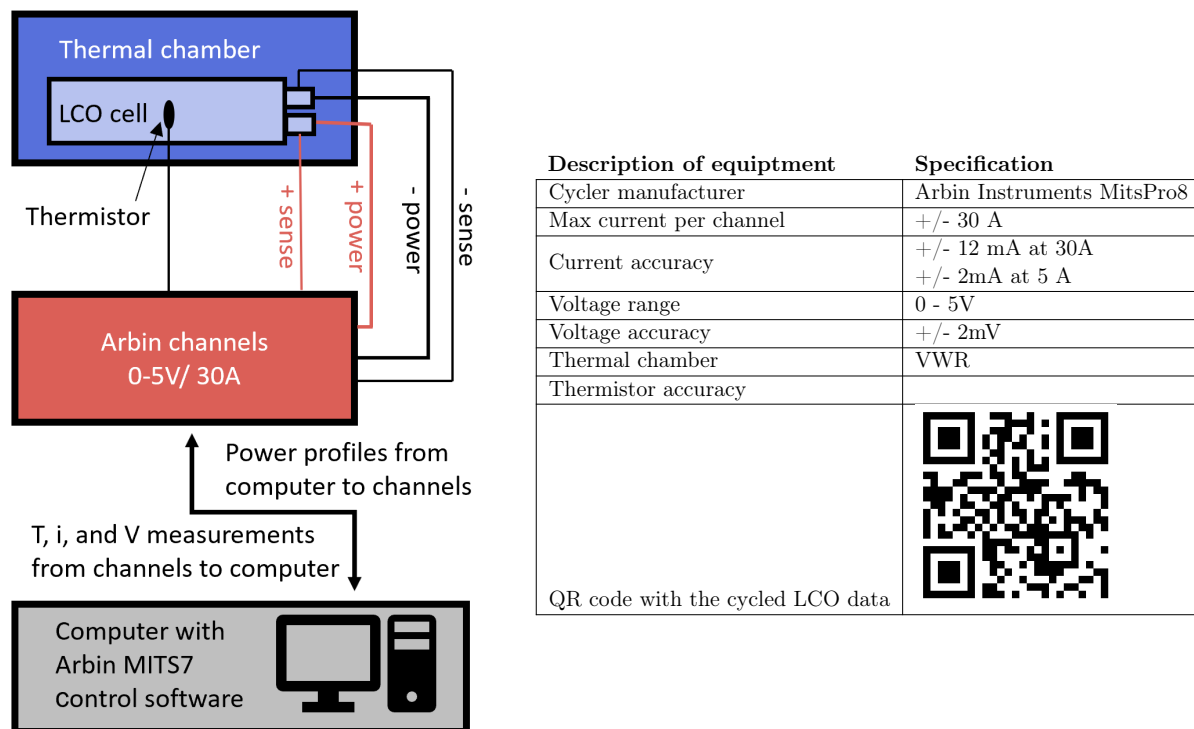


Figure 5.4 & Table 5.5: The figure to the left illustrates the lab set-up for cycling the 100% SoH LCO Melasta cell, along with the specifications of the equipment. The QR code contains all the battery data from the experiment.

5.3.3 Regenerative Braking

An electric mining truck named eDumper can drive without ever charging [123]. Driving downhill it is heavily loaded with stones yielding a large energy potential, which is converted into electric energy by regenerative braking which charges a battery. Then, the load is removed and it can drive uphill on the energy stored in the battery. Commercial EVs can produce a fraction of their electricity from regenerative braking, and this was therefore included in the power simulation in the drive profiles.

When LiBs are exposed for charge pulses like in regenerative braking, the voltage will increase rapidly beyond the cell voltage due to overpotentials and internal resistance. This phenomenon is easily seen in Equation 2.4 in Section 2.2. A problem occurs when the battery has a high SoC since the voltage can exceed the Highest rated voltage of a battery, specified by the manufacturer ($V_h(T)$) when charged at high currents. In EVs, regeneration is disabled at high SoC and only mechanical braking is possible. Along with the SoC decreasing, regeneration is gradually enabled and the mechanical braking reduced³. However, it was not found any easy implementation to gradually increase the amount of charging for the drive cycles. No charging was therefore done before the terminal voltage had first reached 4 V.

5.3.4 Limitations and Special Considerations

5.3.4.1 Equipment limitations

When the power had been scaled down from a battery pack to the power required for a single cell, it was found that the Arbin battery cycler could not handle the high power, high transient profiles. Techniques tried to overcome this problem are elaborated in Appendix B. To cut a long story short, the maximum current of the Arbin battery cycler is 30 A, while the drive cycles reach a current of over 90 A. In order to carry out cycling within these limits, two options were considered:

- Option 1: Scale down all cycles by a factor such that the current is in the range of [-30 A, 30 A].
- Option 2: Upper and lower cutoff power limit. Only the power steps exceeding a current above 30 A will be reduced to a constant number. The rest of the power profile remains unchanged.

Figure 5.5 demonstrates how the two option for scaling would change the cycles. The maximum power limit was set to 111 W (max current 30 A times nominal voltage 3.7 V) in the figure. Moreover, an estimate of the percentage of the data per cycle that is prone to exceed the 30 A limit is found in Table 5.5. The reason this is not hard facts, but only an estimate, is that the scaling would be changed when the terminal voltage drops. Therefore, more power can be drawn when the battery has a high voltage potential than at a lower one.

³This explanation was given through e-mail correspondence by Per-Arne Jansen, head of technical department at Møller bil.

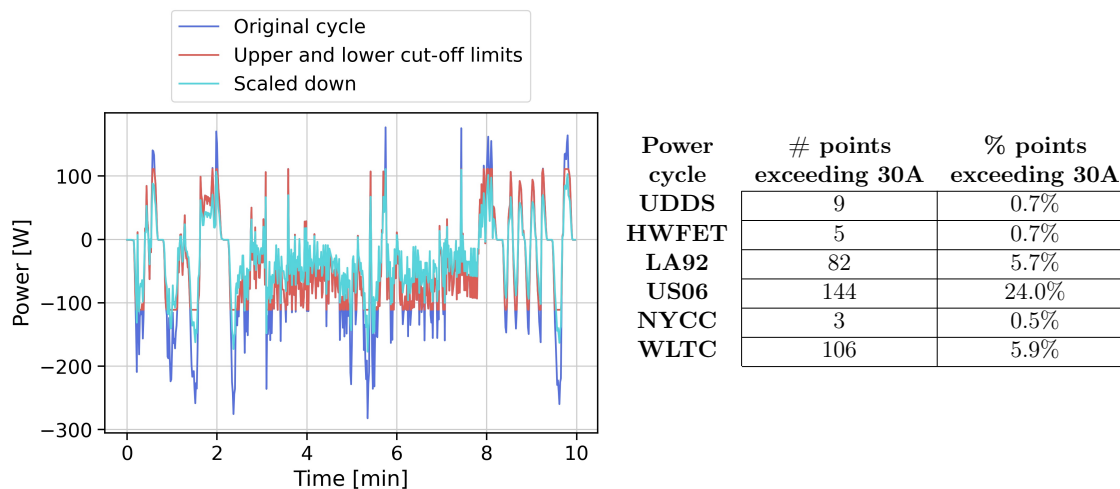


Figure 5.5 & Table 5.6: The figure illustrates what scaling or cutting of at a current limit would look like. The US06 drive cycle is plotted, which is the data set which have the most points exceeding 30 A. This can be seen in the table where the number of points, and the percentage of points, exceeding 30 A per cycle is stated.

Option 1 has an equal power profile shape as the original, but scaled down. It would therefore yield a correct drive cycle simulation if the battery pack had around three times as many cells⁴, connected in series. This would however be undesirable due to the increased weight of the EV and is not realistic.

Option 2 would have many equally high charging and discharging currents since the power is limited to a predefined upper limit such that the current never exceeds 30 A. This is unrealistic in a car since it implies that the acceleration of the car is drastically reduced. In addition, the power profiles have relatively more, and higher, discharge rates than charge rates. Therefore, a greater number of discharge currents would be reduced, meaning the car would break disproportionately more than it accelerates.

Option 2 was chosen based on the goal to obtain data consisting of a broad specter of battery dynamics. This was reasoned with option 2 having a mean power (and C-rate) higher than option 1. Higher C-rates yield faster battery dynamics which again, most likely are harder to predict the SoC for. Moreover, the development of an NN SoC estimator that works on aged batteries is to be tested out as a proof of concept, rather than a complete solution. In such scenes, the unrealistic higher amounts of charge relative to discharge is not perceived as a problem.

⁴U06S is the cycle requiring the highest currents and would have to be scaled down by a factor of approximately three to never draw more than 30 A

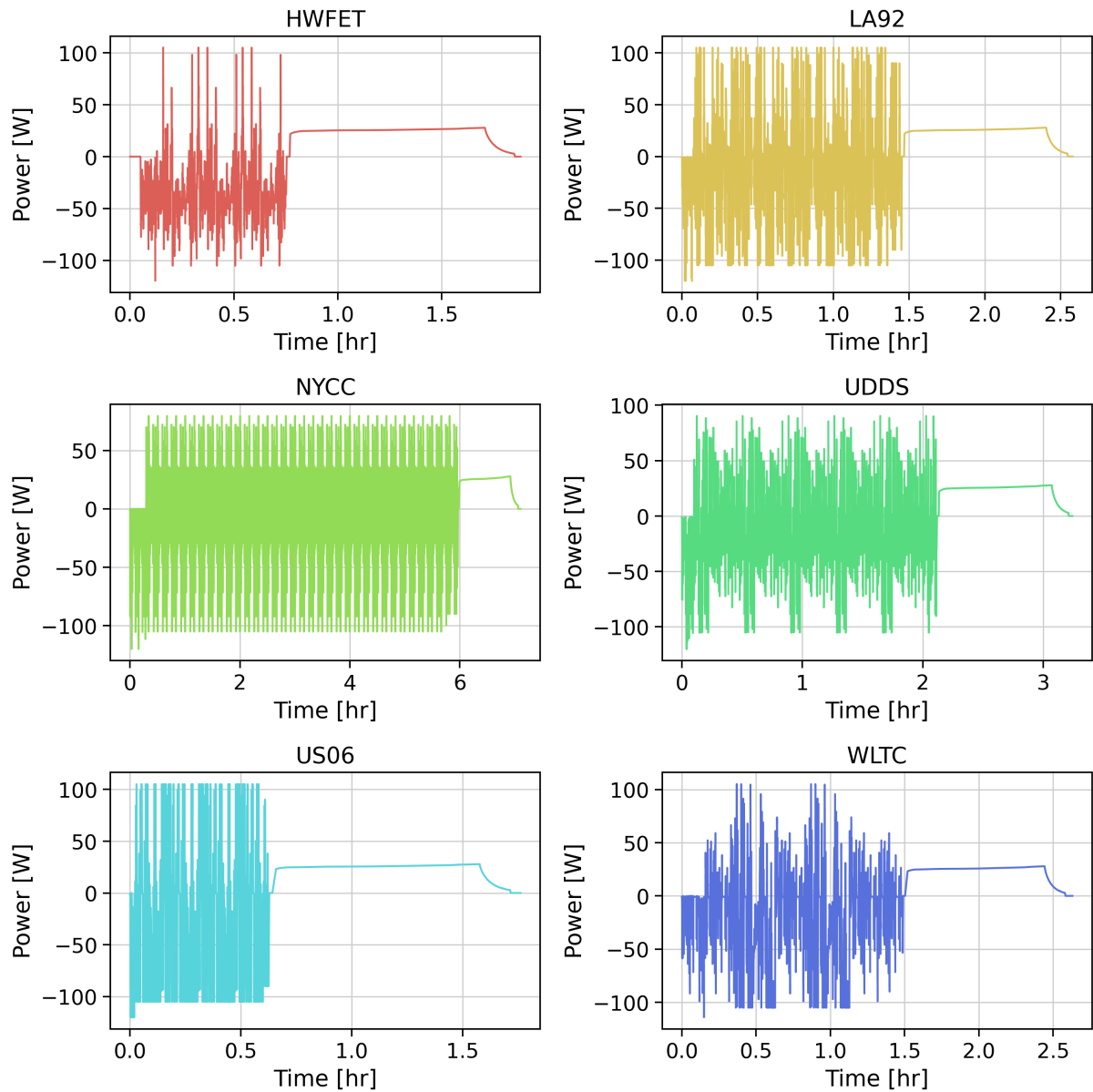


Figure 5.6: Data obtained from cycling the new LCO battery at 25 °C. Each plot is a drive cycle ending with a CCCV charge.

To assure that the current never exceeds 30 A, a simple way would be to set a power limit of 90 W ($30 \text{ A} \cdot V_1(T)$). However, at a higher SoC, the voltage is higher and the power drawn could therefore be higher without exceeding the current limit. Therefore, the following power limits were set within three voltage windows:

- 4.2 V - 4 V: 120 W, no charging
- 4 V - 3.5 V: 105 W
- 3.5 V - 3 V: 90 W

These limitations can be observed in Figure 5.6. Notice that most of the cycle is within the second voltage window, implying that the maximum power is 105 W.

5.3.4.2 Battery limitations

Some challenges cycling the oldest cell at 77% SoH were encountered. First, the problem was to charge the battery with a CCCV charging. When the voltage reaches its maximum plateau after charging by a constant current, the Arbin cyclers is supposed to step-wise reduce the current to obtain a stable voltage at its upper cutoff voltage, $V_h(T)$. However, the voltage gradually reduced beneath $V_h(T)$, and the Arbin software aborted the cycling with an error message concerning the drop in the voltage. This problem was solved by calculating the actual internal resistance based on a hybrid pulse power characterization. The result was an increase by a factor of 6 from the initial manufacturer-measured internal resistance.

The next problem occurred when cycling the battery according to the power profiles. The voltage raised beyond the maximum voltage limit when simulating the regenerative braking, as seen in Figure 5.7. Since the internal resistance is high, the reversible potential (terminal voltage) is increased more than for a cell with a lower internal resistance, if charged at the same current. This reasoning is based on Equation 2.4 in Section 2.2. The solution that enabled cycling the old cell, was to adjust the voltage limit from 4 V to a lower value for when regenerative braking starts⁵.

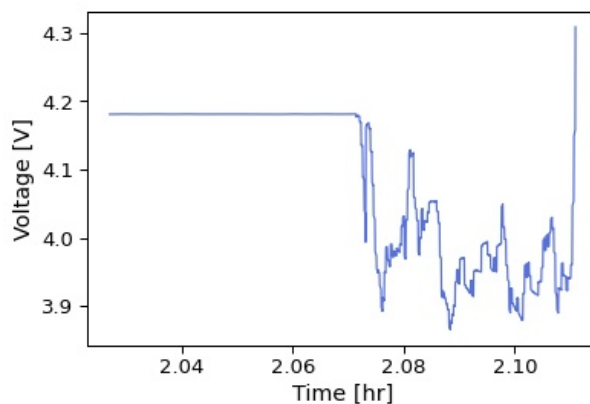


Figure 5.7: Plot of the first minutes of cycling the 77% SoH LCO battery by a drive cycle. When the terminal voltage first hits 4 V regeneration (charging) is enabled. However, for the LCO with 77% SoH, the internal resistance is very high, yielding a high terminal voltage exceeding $V_h(T)$, 4.2 V.

⁵Different voltage limits apply to different cycles and are fitted to the dynamics of the cycle to start regeneration as early as possible. The voltage limits ranges from 3.8 V to 3.5 V.

Chapter 6

Method

6.1 SoC Target Value

The SoC was calculated by Coulomb counting as stated by Equation 2.14. The precision of the measurement equipment that the four battery chemistries were cycled by, is +/- 0.1% for both the current and voltage measurements. Therefore, Coulomb counting was found accurate enough as an estimation method for the target SoC. However, it should be noted that the error accumulated during a cycle when using coulomb counting is not taken into account, and small deviations related to the measurement device are present in the overall error stated for the ML results.

6.1.0.1 Initial State of Charge

To reduce the error associated with the initial SoC (SoC_{init}), SoC_{init} was updated between each cycle. This was done by defining that the battery is at 100% SoC after CCCV charging at 25 °C until 4.2 V is reached with a cutoff current of 50 mA. SoC_{init} before charging the battery was defined as the SoC the battery had when it reached its cutoff voltage from discharging by the drive cycle prior to the charge cycle.

6.1.0.2 Total Capacity

The total capacity is another parameter that has to be determined to calculate the SoC by Coulomb counting. It was determined in two different ways based on the available measurements in each of the four data sets. Preferably, the total capacity should be adjusted during the LiBs lifetime since it decreases. However, for the Panasonic data, the total capacity was set to the total capacity given by the manufacturer which was 2.9 Ah.

For the three other data sets (LG, Samsung, and Turnigy), an approximation of the total capacity was utilized to calculate the SoC. It was approximated as the Ah extracted from each battery during a C/20 discharge. This test was conducted 6 times during the cycling, one time for each

temperature. If it is theoretically correct to use the total capacity at different temperatures in coulomb counting, was researched without finding a clarification. However, it was concluded that from a practical perspective, a C/20 discharge at a given temperature yields a great estimation of the maximum available capacity, and thereby a practical SoC estimate.

6.1.0.3 Charging and Discharging Efficiency

For all the Constant Current Constant Voltage (CCCV) charging cycles, the Charge/discharge efficiency (γ) was calculated by the same formula as used for calculating the Coulombic efficiency:

$$\gamma = \frac{\text{Discharge capacity [Ah]}}{\text{Charge capacity [Ah]}} \quad (6.1)$$

where the charge capacity is calculated from the charging cycle that comes after the drive cycle.

For all drive cycles, γ is set to one. This is, as stated in Section 2.11.2, a typical γ for discharging cycles. The observant reader might question this value since the drive cycles also consist of short charging segments. The argument for why a pure discharge efficiency is used, is that discharging is the major part of the drive cycles (EVs do not generate more electricity than they produce). Also, since the Coulombic efficiency is generally around 99% for LiBs [124], the short charge sequences only have a limited impact on the overall SoC. Table 6.1 states the calculated Coulombic efficiency at C/20 of the three batteries that were cycled, which ranges from 94%-105%¹. This indicates that a γ of one is a rough estimate and that an error is present. However, no further research was conducted since this is a research field on its own [53, 54, 125], and it was not time to include it in this master.

¹The Turnigy battery cycled at -20°C have a Coulombic efficiency of 131%. However, no further cycling was conducted of this battery at -20°C .

Table 6.1: Total capacity calculations approximated as the C/20 discharge capacity at six different temperatures. The corresponding Coulombic efficiency is also stated for each full C/20 cycle.

		Temperature					
		45 °C	25 °C	10 °C	0 °C	-10 °C	-20 °C
	Battery	Total capacity Cn					
	Turnigy	4.74	4.72	4.7	4.66	4.55	3.24
	LG HG2	2.69	2.78	2.65	2.6	2.56	2.49
	Samsung	3.05	3.04	2.99	2.94	2.84	2.74
		Coulombic efficiency gamma					
	Turnigy	0.97	0.98	0.99	0.99	1.01	1.31
	LG HG2	0.94	0.94	0.98	0.97	0.99	1.02
	Samsung	0.97	0.97	0.97	1.01	1.02	1.05

6.2 Input Features

Current (I), voltage (V), and temperature (T) are the measurements used as input features for both RNN and FNN. This choice was based on the previous work summarized by Table 4.1 in Section 4.6. For the RNN, only a single time step of the tree parameters was inputted because the RNN should remember the important previous features by itself. One time step is in this thesis defined as the time between two samplings. A time step for the drive cycles is therefore 0.1 seconds, and for the CCCV it is 60 seconds.

For the FNN, the current and voltage was averaged over a number of samples, (\bar{I}_k and \bar{V}_k , respectively). The number of samples that the measurements are averaged over, is in this thesis referred to as the "averaging window size". The averaging is done because the FNN has no memory, and a single time frame of Voltage, Current, and Temperature does not contain enough information about the SoC to solely be used as the input feature. Figure 6.1 illustrated how the averaging was done.

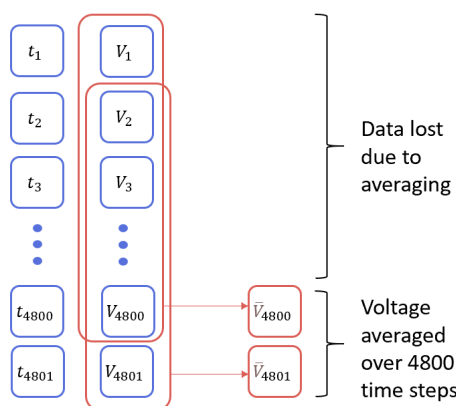


Figure 6.1: Illustration of how the averaging of the input data for the FNN was done. Here 4800 samples are averaged for each time step. From the figure, it is clear that the more samples averaged over, the larger reduction of the data used for training and testing.

Research on the optimal averaging window size was done, ranging from 30 seconds to 16 minutes. Since the drive cycles have a sampling frequency of 1 Hz, while the charging and pauses were sampled at 1/60 Hz, the number of time steps when averaging was 600 times as many for data with charges and pauses, as for drive cycles. This was done to obtain consistency in the number of minutes the averaging was done for.

Further research on input features was done with an algorithm for feature selection, BorutaShap [126]. It was tested to check if the algorithm could prioritize any other features than the three to five mentioned above. The algorithm uses an ML model called random forest, which is trained to predict the SoC based on all the proposed features. However, for each time step, measurements of a different, random time step are also inputted (shadow features). For instance, if an

SoC of 100% should be predicted, the voltage, current, temperature, and power at 100% SoC are inputted into the model. However, shadow features of a different SoC are also inputted to predict the 100% SoC. These have of course no correlation to the 100% SoC target value. When the model is trained on both the original features and the shadow features, the random forest rates the features' correlation to the SoC. However, only the features that are more correlated than the shadow features, actually predict the SoC.

6.3 Standard Functions and Parameter Settings

This section introduces the functions and parameter settings that were used for all the networks throughout this thesis.

All programming was done in the Python [118] programming language. In preliminary studies, the library scikit-learn [117] was used to develop the FNN model due to its simplicity. Despite Sklearns easy implementation, Keras [127] was utilized due to the possibility of implementing more advanced features and thereby custom design networks.

The loss function was chosen to be MSE. This choice was based on that it is more undesirable for an SoC estimator to have a few large off-sets, than many small ones. Since MSE squares the error, the weights and biases are updated more for large errors, than for smaller errors.

The Adam optimizer was used for all optimizations with its default parameters, if not stated otherwise. The default parameters in Adam, implies a learning rate of 0.001, an exponential decay rate for the 1st-moment estimates of 0.9 (β_1), an exponential decay rate for the 2nd-moment estimates of 0.999 (β_2), and ϵ for numerical stability of $1e-7$ (this epsilon is the "epsilon hat" in [128]).

Moreover, it was chosen to use the min-max scaler as stated by Equation 3.7 with an upper and lower boundary of 1 and 0, respectively. Literature on the topic of estimating SoC with NN has not researched the optimal scaling, and min-max scaling and standardization are the two scaling methods previously used for SoC estimation by NNs. Moreover, literature on scaling in general states that there is no broad agreement on what type of scaling that should be used in different problems, but there is a broad agreement that scaling is beneficial [129], [130].

At the start of every script, a random seed of 42 is set, as well as a global seed of 42. This enables reproducible results. Since the results were made by very many files, not all were appended to this thesis, and only example files are added, implying that if manipulated slightly (*e.g.*, chaining the max layers in random search as described by the table), the same results should be obtained. The QR code in Figure 6.2 provides a folder with several example scripts which can be used to easier obtain the results



Figure 6.2: Scan QR code for obtaining a folder with scripts of NNs from this thesis. Jupyter notebook is required to open the scripts.

of this thesis.

Shuffling of the training data is an embedded function in the Keras' fit-method, and was used for training the FNN. The idea is that the model should not be over-fitted on the last part of the data, but always train on a representative sample of the whole data set. Especially when training on the augmented data, this is crucial, since the same drive cycles, only slightly manipulated, come right after one another. So, to prevent the network from overfitting on the last cycle in the data, random shuffling is used.

6.4 Optimizing the FNN Architecture

This section provides detailed information on how the FNN was constructed, the parameters used, and reflections on the choices. In short, the following methods were researched:

1. **Random search** was used to identify optimal hyperparameters and optimal input features for SoC estimation with FNN. Only the Panasonic data set was utilized. Researching input features was done in parallel with optimizing the hyperparameters.
2. **Transfer learning** where a network is trained on the Panasonic data set then fine-tuned for LG, Turingy, and Samsung.
3. **Augmented data** for training on a larger data set (only investigated for LG).
4. **Exploring an additional input feature**; the temperature difference between the ambient temperature and the battery temperature (only investigated for LG).

To obtain the best FNN, the first thing that was done, was to explore how training on different drive cycles and temperatures changed the accuracy of the model. Moreover, it was explored what data sets that were hardest to test with, and if there were any patterns in the data that should be trained with to yield the best results. It was tested on a general architecture of two layers with 32 and 16 nodes per layer, an architecture based on previous results from literature.

6.4.1 Random Search with the Panasonic Data Set

There exist several optimization algorithms to tune hyperparameters, where random search and grid search are the most common and simplest methods [131]. Generally in optimization, a search space is defined containing the parameters that are to be optimized. In random search, the search space is defined as a bounded domain of hyperparameters where random combinations within the domain are used. In contrast, for grid search, every single combination within the search space of hyperparameters is evaluated. In this thesis, random search was chosen to more rapidly search a broad search space since the optimal hyperparameters found in previous work are largely spread as can be found in Table 4.1 in Section 4.6 (*e.g.*, 1-3 hidden layers, 2-256 nodes per hidden layer). The Keras random search tuner [132], was utilized to optimize hyperparameters with a focus on the number of layers, number of nodes, and dropout.

Random search was run 23 times where either the search space, or the input features, were ad-

justed. For each search, 20 different combinations of hyperparameters were evaluated, and each combination was initiated at least two times. The latter was done to reduce the uncertainty caused by the random nature of NNs, where a non-optimal initialization could yield an excessive high error. The networks were only trained for 5 epochs². From heuristics and preliminary studies work (project thesis), 5 epochs are enough to converge the network to a large extent, but without overfitting. Therefore, as well as limited time and for a fair comparison, all networks were trained for 5 epochs. A flowchart of the procedure is found in Figure 6.3.

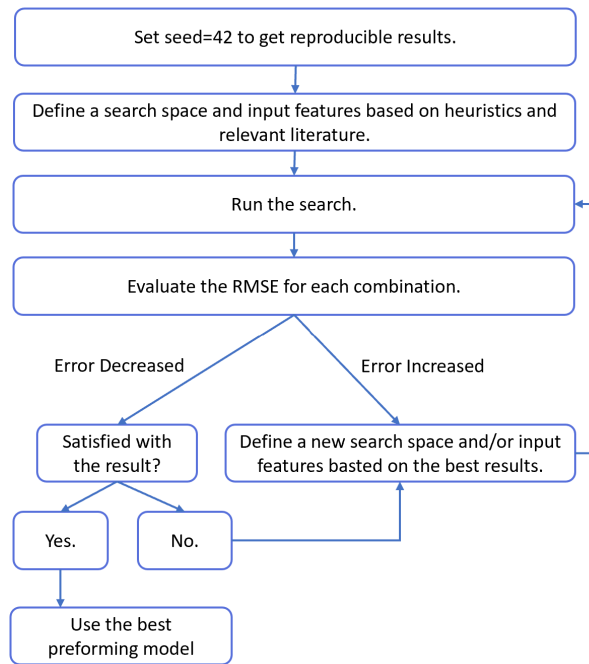


Figure 6.3: Iterating to increase error by tuning hyperparameters and optimizing the averaging window. The tuning algorithm Random Search is used for hyperparameter tuning.

Moving on to the data used for researching the optimal hyperparameters and features, only the Panasonic data was used. To start with, the best practice on training, validation, and test splitting was executed. It was chosen to use HWFET for validation, US06 for testing, and to train with the rest of the data set. US06 and HWFET were chosen based on the results later stated in Section 7.1.1 where these two profiles yielded the highest error scores and were therefore found efficient for testing the robustness of the model. The downside is that the model is not trained on these difficult cycles, but a robust test score was valued higher than a better training since the model is not to be deployed in a physical application, but meant for pure research purposes. However, it should be stated that the random search method was not fully understood when starting the searches. Therefore, the best practices regarding testing and validation were not performed. This is made clear and critically reflected upon when explaining the methodology.

²The two first random searches were run with 20 epochs

In the preliminary studies to obtain an optimal SoC estimator, no CCCV cycles were used to train and test with. The main reason for this choice is, by reducing the amount of data, it takes less time to train the model. The discharging data was chosen due to the assumption that they are the hardest to estimate due to their transient profiles. Thus, the model shell is based on discharge data, and in only the last four searches, when the hyperparameters had been roughly determined, the charging profiles were included.

Random search works in many ways as a normal FNN training process in terms of training, validation, and testing. In random search, as for FNNs, either a predefined validation set, or a split ratio defining the amount of training data that is used for validation is defined, and the validation score is calculated after each epoch. In this thesis, it was chosen to use a specific cycle for validation since a split ratio is prone to overfitting as discussed in Section 4.6.1. The model from the epoch with the lowest validation score was picked as the best model. Then, a new model with other combination of hyperparameters was trained for new 20 epochs, and again, the score of the best model of the 20 epochs was saved.

Furthermore, as normal practice in ML, one has to evaluate the model on the test data after the model is constructed. Since 20 models were constructed for each search, and the model yielding the lowest test score, might be a different one than the model yielding the lowest validation scores, all the 20 models would have to be reloaded after each search to evaluate the test set³. This is time-consuming, and for a rough indication of the optimal parameters, it was assumed enough to only use the error scores from validating with the drive cycle HWFET. Therefore, the results that are used for concluding on the best model are solely based on the validation data. For the 10 first searches, HWFET was used to validate.

For the next six searches, it was chosen to include the US06 in the validation data. This was because the test data was not utilized for choosing input features and tuning hyperparameters, and was therefore without function, and thereby obtain a broader choice basis. As stated, this was done after 10 searches, which is the point where a search for the optimal number of layers begins.

In the final process of choosing the best architecture, only US06 was used for evaluation. This was due to two reasons: firstly, the US06 data set generally has the least missing charging profiles. This can be observed in Table 5.2 in Section 5.3.4 where all the missing charging profiles are stated. Secondly, it is the only drive cycle with a charging profile for -20°C . Since one of the goals when constructing the FNN is a robust model for all types of conditions, it is important to choose a model based on scores obtained by validation on both negative temperatures and charging profiles.

³It might be that evaluating with test set could have been implemented by manipulating the random search function, but this was not researched.

6.4.1.1 Leaky and Clipped ReLU

In [109], it was found that the activation function clipped ReLU and leaky ReLU proved efficient. Clipped ReLU is, as its name suggests, a ReLU clipped at a certain maximum value. For SoC estimation, the maximum value is set to one to force the model to never predict above one (indicating 100% SoC). The clipped ReLU is used as the output activation function and could be beneficial if the vanishing gradient problem is present. This is because it has a gentle slope in the lower limit (here set to 0.03), unlike the general ReLU which maps all negative values to zero. The leaky ReLU was used as the activation function between the dropout layer and the last hidden layer.

To research if these two activation functions, or only one of them, is more efficient than only ReLU, four experiments were done:

- I Only ReLU
- II Clipped ReLU
- III Leaky ReLU
- IV Clipped and Leaky ReLU

The architecture of the four experiments is illustrated by Figure 6.4. The roman numbers in front of the activation function represent if the activation function was used in the experiment. Experiment I is used as the benchmark case and is simply the best model found in the random searches. In all experiments, the input features that yielded the lowest error from the random search were used. In experiment II-IV, the number of hidden layers, number of nodes per layer, and the dropout rate was the same as I. However, a new random weight initialization was made. Therefore, for each of these experiments, three models were constructed with different initialization.

The batch size was chosen to be 32, which is the same as used for all the random searches. It was first investigated training on a batch size of 1 which took several hours to tune the network. In comparison, it took less than 10 minutes with a batch size of 32. Also, with a batch size of 1, the network was overfitted after the first epoch. In addition, a batch size of 64 was tested (with two initializations to reduce the random noise of the initialization, trained for 10 epochs). This yielded approximately the same minimum validation error when trained with a batch size of 32. However, the error oscillated more with a batch size of 64 than for 32, and 32 was therefore chosen.

Early stopping was now implemented such that training stops if the network overfits. The stopping criterion was set to a validation loss of $5 \cdot 10^8$, corresponding to an RMSE of around 0.02% SoC, and a patience of 3. A note to these criteria is that it was the patience criterion that always

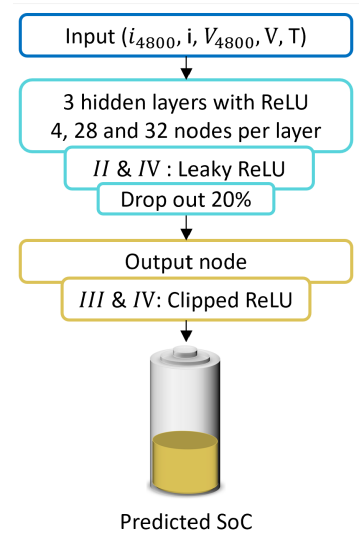


Figure 6.4: Illustration of the FNN architecture when clipped and/or leaky ReLU is added. Four experiments were conducted, and the roman number in front of the activation function represent whether or not the activation function was used in the respective experiment. The sample size of the averaging window is expressed with respect to the drive cycles.

terminated the search. This was due to the validation loss started increasing.

The network that yielded the lowest RMSE in this hyperparameter tuning with the Panasonic data set, was used for both transfer learning and training on augmented data. From this point onward the network will be referred to as the "base network". Similarly, the term "base network structure" refers to its network structure with its hyperparameters, but without its weights. Moreover, the features that yielded the lowest RMSE will be referred to as the "base features".

6.5 Transfer learning

In section 3.5 it was stated that the overall error is expected to decrease through the use of transfer learning. This will be tested with the Panasonic battery data as the source domain and three different target domains, namely LG, Samsung, and Turingy. The description of the data set and the battery is found in Section 5.1. The models were tested by four drive cycles, namely LA92, HWFET, UDDS, and US06, which are described in detail in Section 4.1. For validation, "Mixed 1" was used.

A benchmark case was established for each battery type by training the base network structure with three random initiations. The reason three initiations were made, was to limit the effect of one poor initialization. After an error score for the benchmark case was established, transfer learning was conducted in nine different ways; three different learning rates 10^{-3} , 10^{-4} , and 10^{-5} , and the impact of freezing zero, one, and two hidden layers was investigated. It was always the first layer(s) that were frozen.

Both the benchmark cases and the transfer learning was trained with a batch size of 32 and with early stopping with a maximum number of epochs set to 50. The patience was set to five, and the model stopped when the error decreased less than 0.001% SoC (MSE=1e-10). The model in the epoch yielding the lowest validation score was restored and evaluated on the test data.

6.5.1 Augmented Training Data

To create larger training data sets, random Gaussian noise was added to the original training and validation data⁴. This resulted in four times more data. The noise has a mean of zero, but the standard deviation (σ) is calculated for measurement of each temperature and cycle, by the following formula:

$$\sigma = \frac{\sigma_{\text{drive cycle}}}{K} \quad (6.2)$$

where K is a constant defined in Table 6.2 and $\sigma_{\text{drive cycle}}$ is the standard deviation calculated for each measurement (V, I, T) for each temperature of each cycle. The same approach was done to add noise to the charging cycles. The numerical values after calculating each $\sigma_{\text{drive cycle}}$ are

⁴The random noise was generated by the Python library `numpy.random.normal`

found in Appendix D.

Table 6.2: Overview of the value that the standard deviation to each drive cycle is divided by to obtain the standard deviation of the noise.

Augmented Data Set	K		
	I	V	T
A	10	100	10
B	20	200	20
C	30	300	30

The reason why the voltage is scaled more than the temperature and the current is based on similar works. In [109] they used a constant offset in the current, voltage, and temperature measurements of +/-110 mA, 4 mV, and 5 °C, respectively. In that paper, only the current got a non-constant noise, specifically a gain of 2%. If the training benefited from the added noise was not stated. However, in [67] adding noise yielded reduced error scores. The Gaussian noise had a standard deviation of 2-4%⁵ and the noise for the offset and gain was similar as the former paper. It was found interesting to research if pure Gaussian noise with no offsets could help the FNN in its training process. Moreover, the plan for the research on augmented data was to evaluate the effect of only using offsets and only using gain, but these experiments were not prioritized and due to the time limitation, they were not conducted.

The purpose of calculating the standard deviation first and scaling it down with a factor was to have a σ proportional to the size of the values. However, it is an overly complicated procedure to create the noise, and a simpler way is to set a σ to a constant number much lower than the measurement value.

For the augmented data, as well as the temperature difference, only the LG data was used. This was simply because of the limited time of this thesis. The reason why LG was chosen, is that it has data on -20°C (which the Turingy set is missing), a known charging temperature of 25°C (which is unknown/not constant for the Samsung battery), and few missing charging profiles (opposed to the Panasonic data set).

6.6 Temperature Difference as an Input Feature

In Section 2.9 the electrical, thermal, and mechanical impact on SoC was described. In addition it was stated a need for a coupling between three different properties when estimating SoC over a longer time period. Previous work have inputted thermal properties in terms of the battery temperature (T) and electrical properties in terms of the current (I) and voltage (V). However, the battery temperature alone cannot describe the heat generation in a battery cell. In this sec-

⁵It was assumed that the percentage was not a percentage of a measurement, but simply a standard deviation of 0.02-0.04.

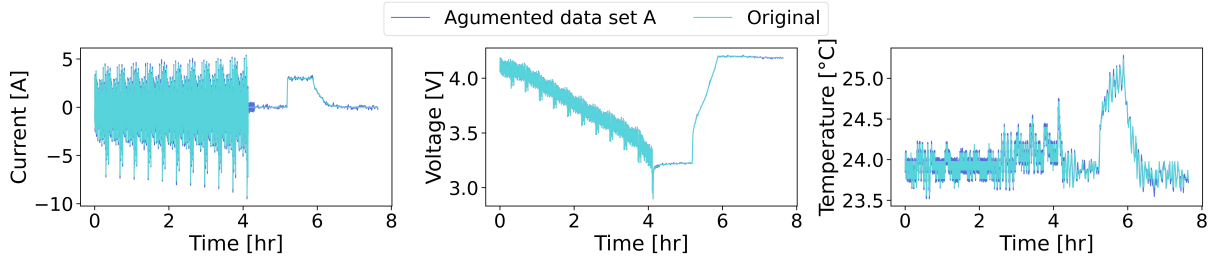


Figure 6.5: Plot of LA92 at 25 °C of the original and the augmented cycle. The cycle plotted is a part of the augmented data set A as stated in Table 6.2, implying noise with the highest standard deviation.

tion a method to improve the input of the thermal property is described.

An approximation of the heat generation in the battery cell is proposed as an additional parameter. It is approximated by the temperature difference between the temperature inside the heat chamber and the battery cell (ΔT). This was done because ΔT is associated with the heat flux generated from the internal losses of the LiB. As described in Section 2.2, both the reversible and irreversible heat losses in the cell change its temperature, and the magnitude and sign vary with SoC. Since the battery is in a heat chamber with a set temperature, the heat flux can be approximated from Newton's law of cooling, stating that the rate of heat loss of a body is proportional to the difference in temperatures between the body and its surroundings. Mathematically Newton's law is stated by:

$$Q = h \cdot A \cdot \Delta T(t) \quad (6.3)$$

where h [$W/m^2 \cdot K$] is the heat transfer coefficient, A [m^2] is the battery's surface area, $\Delta T(t) = T_{battery}(t) - T_{ambient}$ [K] is the time-dependent temperature difference between the environment and the battery. h and A are constants, so the heat transfer is therefore only changing with ΔT , which was to be used as an additional input feature.

A challenge is that no ambient temperature measurements exist. However, it is stated in the "read me" file of the battery data that the drive cycles and charging cycles are of a given, constant temperature. The battery cycling with drive cycles is done at $-20^\circ C$, $-10^\circ C$, $0^\circ C$, $10^\circ C$, $25^\circ C$, or $45^\circ C$ while charging is always done at $25^\circ C$ ⁶. Therefore, it was approximated that the ambient temperature was constant at either one of these six temperatures. During the pauses (zero current) in the charge file between the drive cycles and the charge cycles, the battery temperature is rising towards charging temperature ($25^\circ C$), indicating a rising ambient temperature. The opposite is observed at the end of the charge file where the battery temperature drops, indicating a decreased ambient temperature. The ambient temperature could have been approximated as linearly falling and rising, but this would yield a pretty rough estimation, and to reduce the uncertainty, the pauses were taken away for this experiment⁷.

⁶Kollmayer *et al.* stated that the battery temperature was $25^\circ C$. This fact was checked, and the mean charging battery temperature was $24.36^\circ C \pm 1^\circ C$ (inaccuracy of the thermocouples). Hence, it was found accurate enough to estimate the ambient temperature as $25^\circ C$

⁷There are pauses in the start and end of the drive cycle file as well, but here the battery temperature is

Four main experiments were conducted:

1. Benchmark case with transfer learning
2. Benchmark case with base structure
3. Additional input feature: ΔT
4. Additional input feature: ΔT averaged over 8 minutes

Two benchmark cases were obtained, similar to the base network. The reason why the base network was not used directly as the benchmark case is that the pauses were taken out of the data set. Therefore, the base network (with transfer learning) and base structure (with randomly initiated weights) were trained without pauses and used as benchmark cases. Both were trained with a learning rate of 0.001, and no frozen layers.

For the two last experiments, the additional input parameter was added to the base features. This makes up for a total of six input features. The input layer changes accordingly, and a new hyperparameter tuning was done with random search. It was conducted with 20 different combinations and three trials per combination. The search space is identical to the one stated in Table 7.1.

Table 6.3: Reduced search space used for random search. It was established after the first two searches.

Search space parameters	Search space values
Number of hidden layers	1-4
Number of nodes per hidden layer	4-36
Activation function	ReLU
Dropout rates	0, 0.1, 0.2, 0.3

The additional input feature for experiment number 3 is ΔT , implying the temperature difference at the same time instance that the SoC is predicted. In experiment 3, the average of ΔT during a limited time period of 8 minutes (equal to the averaging period of the voltage and current measurements), is the additional input feature. This averaged temperature difference between the ambient temperature and the battery temperature will be denoted $\overline{\Delta T}$. The reason for conducting this experiment was to check if the trend in temperature difference correlates with SoC.

6.7 Optimizing the RNN Architecture

Only three input variables were used: current, voltage, and battery temperature. The Panasonic data set [108] was utilized to train and test the RNN. The model was not tested on CCCV charging cycles, only on drive cycles.

relatively constant. The pauses in this file are therefore kept.

Several considerations have to be done before inputting the data to the RNN. Firstly, the RNN have a memory and therefore the data cannot be shuffled randomly. To elaborate, time-step $k+1$ have to be inputted after time-step k , which again have to be inputted after $k-1$. Else, there would not be any connections between the time-step, and the RNN would not be able to learn the interrelationships. Secondly, the model was set to stateful. A stateful model implies that the state after of the current epoch is sent to the next epoch. In theory, this means the model could keep the first time step in its memory all the way until the last time step. Thirdly, the memory of the RNN should only go back to the start of a cycle. Therefore, when a new cycle is inputted to the model, the state is reset. This is enabled by not feeding all cycles into the network at once, but manually looping though all the cycles and saving the history for each cycle (training and validation scores). To summarize, shuffling is disabled, stateful is set true, and a for loop with each data set is used.

Results and Discussion

In this chapter, the models are constructed as outlined in Chapter 6 with the preprocessed data as described in Chapter 5.

7.1 Preliminary Studies

Two preliminary studies were conducted before optimizing the network's hyperparameters. The first one was to research which data sets to choose as training, validation, and test sets. The second study was to research new input features.

7.1.1 Selection of Train, Validation, and Test Data

As discussed in the previous work chapter in Section 4.6, the train, validation, and test split is essential for constructing a robust and accurate model. Therefore, a study on splitting the data was conducted on the Panasonic data set.

A relationship between the energy consumption of each drive cycle and the difficulty in estimating the SoC of the data set is suggested. To check this out, 8 models constructed by similar network architecture as found from literature (16, 32 with ReLU as activation function, standard scaler) were made and tested on one drive cycle, for instance the drive cycle LA92. Each time a new model was trained, one validation set was put aside. This was repeated for the drive cycles US06, HWFET, and UDDS. In total 32 models ($8 \cdot 4$) were trained and tested. The average RMSE was calculated by summing the RMSE from each model. The result is found in Figure 7.1. It is clearly seen that the drive cycle US06 is hard to estimate. The assumed correlation between the energy consumption and the difficulty of estimating the SoC is however not evident, and further research is required before drawing a conclusion.

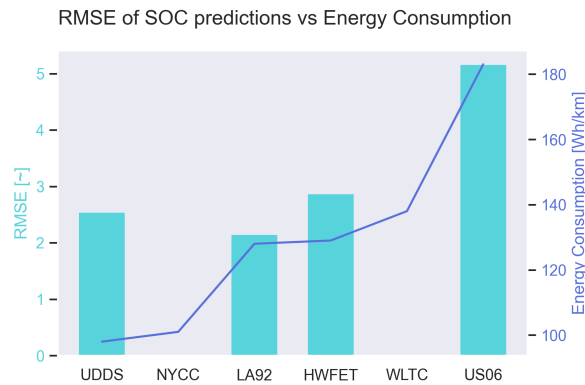


Figure 7.1: Plot illustrating the variation in difficulty for an FNN in predicting SoC for different test sets.

Based on these findings, it was concluded to use HWFET and US06 as validation data when optimizing the hyperparameters of the model. The reason for this choice is their high RMSE prediction scores, and thus an increased ability to stop the training at a stage where overfitting is not present. However, the downside is that the model is not trained on the most challenging data, but this was found less important since the model is not meant for real-world applications, but constructed for critical research.

The impact of temperature was also investigated. Lower temperatures typically yield a higher prediction score, with some exceptions.

7.1.2 Feature Selection by Boruta Shap

Before training the network, a preliminary effort to find parameters that are highest correlated to SoC was done with the ML algorithm Boruta Shap. Power was extracted as having a high correlation to SoC than current. Therefore an attempt to use power instead of current as an input feature to the FNN was done. This resulted in higher error scores. Unfortunately, little information was obtained through this method. The reason could be that the method is suited for a larger amount of unknown features. However, features from previous work were confirmed to be efficient to estimate the SoC. In addition, averaging windows of 1000 samples for the current and voltage was found as better input features than 100 samples. The results are found in Appendix E.

7.2 Hyperparameter Tuning and Feature Selection

The first method that was tried for obtaining the best hyperparameters was manually changing one by one parameter for each tuning. Then the parameters used for the tuning were noted down, with their corresponding error. This was done as an iterative process to reduce the error. However, there is an enormous range of adjustments that can be done on the hyperparameters. In addition, adjusting one parameter influence another. For example, if the optimal number of nodes for one hidden layer was found, then, if adding a new hidden layer, the number of nodes in

the layers has to be optimized once more. It is evident that this method is time-consuming, and, more importantly, it did not result in the low errors as previous work obtained, stated in Table 4.1. Therefore, the optimization algorithm random search was used to optimize the network.

7.3 Random Search

To reach the goal of obtaining a satisfying low error score, more than 450 different combinations of hyperparameters were tried. In total, this was a result of 24 tunings, where one tuning consists of changing the search space, and/or using different features. In this section, the majority of the results are graphically presented. A full overview of the search space and the corresponding results is found in Appendix C.0.1.

7.3.1 First Tuning Results

It was quickly established to reduce the search space for the hyperparameter tuning. Of the output activation functions (ReLU, Linear and Sigmoid), ReLU resulted in the lowest RMSE. There were some exceptions, but based on previous research and these tuning results, it was concluded to only use ReLU for further searches. Another reduction in the search space was from 1-7 hidden layers to 1-4 hidden layers. Moreover, the search space of the nodes was reduced from 4-68 nodes per layer to 2-36. This was based on the fact that the models with the lowest error scores had less than 4 hidden layers and less than 36 nodes. In addition, no models with dropout rates of 50% yielded low error scores and therefore neglected from the search space after the first tuning. These reductions resulted in a narrowed search space stated in Table 7.1.

Table 7.1: Reduced search space used for random search. It was established after the first two searches.

Search space parameters	Search space values
Number of hidden layers	1-4
Number of nodes per hidden layer	4-36
Activation function	ReLU
Dropout rates	0, 0.1, 0.2, 0.3

7.3.2 Window Size for Averaging Current and Voltage

Research on the window size optimal for averaging the voltage and current were conducted. In this research, the search space stated in Table 7.1 is used, and no adjustments were made. This was to reduce the bias from adjusting many hyperparameters, but to still have enough combinations to get a result reflecting the optimal input features.

To begin with, the current and voltage were only averaged over a window size between 300 to 4800 steps. Although the error kept on decreasing when increasing the window size, there were three reasons why larger time-steps were assumed less optimal and therefore not research at

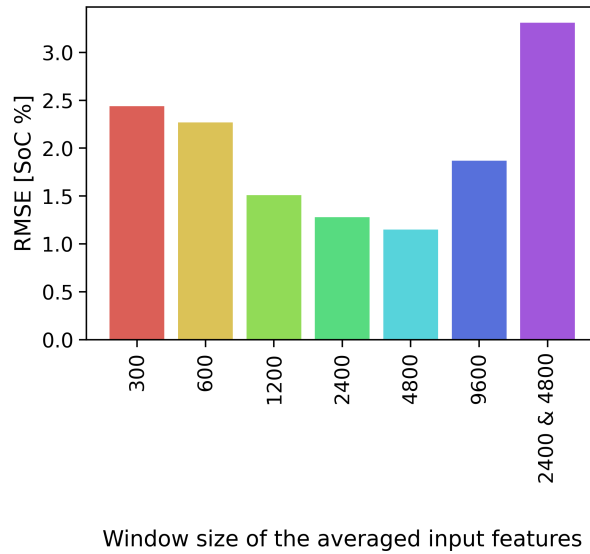


Figure 7.2: The best result after 20 random searches per window size. The rightmost bar have two window sizes; the voltage and current is averaged over 2400 and 4800 samples, respectively. because the voltage was average over a shorter time frame than the current.

first. Firstly, when the time window of averaging increases, the data set becomes smaller due to the cutting of the time-steps at the start. The consequence is neither training nor testing on the battery dynamics at high SoC. Secondly, it was assumed that the larger time the current is averaged over, the more of the target value is included in the input. This was reasoned by that the target value is calculated by coulomb counting, where the current is integrated over time. Lastly, only one paper was found that stated the window size used, namely [67], where 400 time-steps were used. It was incorrectly assumed that 400 time-steps meant 400 samples, or 40 seconds since it was the Panasonic data set that was used with a sampling frequency of 0.1 Hz¹.

A window size of 1200 samples was chosen for averaging both current and voltage, corresponding to 120 seconds. This was due to a drop in RMSE found from 600 to 1200 steps as seen in Figure 7.2. Again, the results with the averaging of 9600 which is displayed in the figure, were not re-search until a later stage. So, when researching the window size between 300 to 4800 steps, it was found that doubling the window size of 1200, only yielded a minor reduction in error, compared to the assumed disadvantages discussed above.

Another averaging tried was using a different window for averaging the current and voltage measurements. The voltage and current were averaged by 2400 and 4800 samples, respectively. However, this did not reduce the error.

On a later stage in this research, a recent paper was discovered that stats the window size,

¹To further explain this confusion, the averaging window was also referred to as the number of "historical data points". It was therefore assumed that one time-step or one data point corresponded to one sample.

[109], by the same research group as the previously discussed paper, [67]. In this new paper, they explicitly state that one step is 1 second (and not 0.1 seconds as the sampling was done at) and that they used an averaging window of 500 seconds, implying 5000 samples². With this new information, a new random search was conducted with an even larger window size of 9600, to closely examine the assumption that the larger window, the lower the prediction error. It resulted in an increased error, indicating that the error does not have a monotone decreasing trend with an increased current window, as first assumed. This finding resulted in redoing the decision of an optimal window size of 1200, to a new optimal window size of 4800 steps. The new window size was used for all experiments after this point but before this finding was done, 14 searches had been conducted with a window size of 1200. Despite training and testing on a different window size than the optimal one, the information from these searches was not abandoned but neither fully utilized and trusted. They were seen in combination with the results from the experiments with a window size of 4800.

To summarize and discuss this section, the optimal window size for averaging samples was found to be 4800 samples because it resulted in the lowest error. This window size corresponds to eight minutes. A disadvantage of averaging over this large window size is that the first 4800 steps of the data is removed. This could however be fixed if a coupling between each charge cycle and discharge cycle was made. This requires a thorough understanding of the data set such that the right profiles are coupled. Due to the time limitation this was not done. Moreover, there are a few uncertainties if 4800 samples is optimal where one aspect is that not all combinations of all hyperparameters were tried. However, 20 combinations per window size were assumed to yield a good indication. Another aspect is that, if assumed random search yielded decent hyperparameters, the absolute optimal window size could exist anywhere between 1200 and 9600. However, previous work have also stated that averaging over 4000-5000 steps is optimal and it was concluded to use 8 minutes as the averaging window size.

7.3.3 Number of Hidden Layers, Nodes, and Input Features

When developing a NN there is a sweet balance between complexity and overfitting. A complex model with respect to the number of hidden layers and the number of nodes per layer takes more time to train and predict. In addition, if becoming too complex, the network could in theory have one mapping between each input and output value, instead of generalizing. In this thesis, no specific sweet spot of this balance was found. However, a decent network was obtained, with a structure of three hidden layers and 4, 28, and 32 nodes per layer.

In parallel with the search for the optimal number of hidden layers, a search of input features was done. More specifically, a search on whether or not to including the instant time measures was done in parallel. The reason for parallelizing the searches is that the number of features inputted influences the optimal structure. It was found that including the instant time-steps is beneficial.

This result was obtained by random search as above but the number of layers for each search

²To exclude any confusion on the steps size, in this thesis the step size is always 0.1 second, based on the sampling frequency. This implies that one sample is one step.

was held constant. The majority of the searches to find the optimal number of layers were done with a window size of 1200. The validation data changes during the searches, as described in Section 6.4.1.

The results from tuning for an optimal number of hidden layers are stated in Table 7.2. In addition, the RMSE is plotted for each search in Figure 7.3.

The table is somewhat complex, and a brief explanation is useful. To ease the interpretation of the results stated in the table and figure, there are color codes. Equal colors indicate that the same validation data is used and that the error scores are directly comparable. One exception is made. The green, light blue, and dark blue colors are all validated by the US06 data set. However, for the blue shades (search number 20-23) a CCCV charging is included in the validation data. Moreover, for the search marked with the dark blue color (search number 22 and 23), the averaging window of the hyperparameter is 4800 instead of 1200.

Table 7.2: Search conducted for determining the number of layers, the number of nodes per layer, and whether or not to only use averaged current and voltage measurements, or include the voltage and current measurements of an instant time-steps. All scores are from predicting on the validation data. The colors are to separate the searches whenever the data set used for validation is changed, CCCV charging data is included, or the window size is changed. Be cautious when comparing results between the colors.

Tuning No.	Best MAE SoC[%]	Best RMSE SoC[%]	CCCV	Input	Hidden layers	Validation data used for error scores	Window Size
5	1.13	1.51	No	i_1200, V_1200, T	1-4	HWFET	1200
8	1.20	1.60	No	i_1200, V_1200, V, T	1-4	HWFET	1200
9	1.11	1.51	No	i_1200, V_1200, i, T	1-4	HWFET	1200
10	1.11	1.49	No	i_1200, V_1200, i, V, T	1-4	HWFET	1200
11	1.78	2.53	No	i_1200, V_1200, T	4	HWFET, US06	1200
12	1.55	2.23	No	i_1200, V_1200, T	3	HWFET, US06	1200
13	1.65	2.23	No	i_1200, V_1200, i, V, T	1	HWFET, US06	1200
14	1.65	2.26	No	i_1200, V_1200, i, V, T	2	HWFET, US06	1200
15	1.62	2.26	No	i_1200, V_1200, i, V, T	3	HWFET, US06	1200
16	1.89	2.60	No	i_1200, V_1200, i, V, T	4	HWFET, US06	1200
17	1.92	2.65	No	i_1200, V_1200, i, V, T	3	US06	1200
18	1.95	2.78	No	i_1200, V_1200, i, V, T	2	US06	1200
19	2.04	2.91	No	i_1200, V_1200, T	3	US06	1200
20	2.04	2.83	Yes	i_1200, V_1200, i, V, T	3	US06	1200
21	2.27	3.10	Yes	i_1200, V_1200, T	3	US06	1200
22	1.16	1.43	Yes	i_4800, V_4800, i	3	US06	4800
23	1.01	1.26	Yes	i_4800, V_4800, i, V, T	3	US06	4800

The results will be discussed in the same order as the tuning was conducted:

- Search number 5, 8 - 10: An exploration if inputting current and/or voltage at an instant time-steps (i and V , respectively) is beneficial to solely using i_{avr} , V_{avr} and T . Similar error scores were recorded and more experiments were required.
- Search number 11 - 16: Each search was conducted with a constant number of hidden layers. The results indicate similar error scores (RMSE and MAE) for search number 12-15 where the hidden layers range from 1-3.
- Search number 17 - 19: Three hidden layers yield lower error than two. Therefore three hidden layers are chosen.

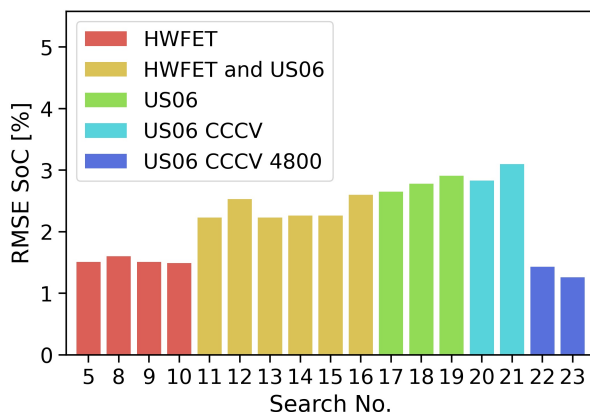
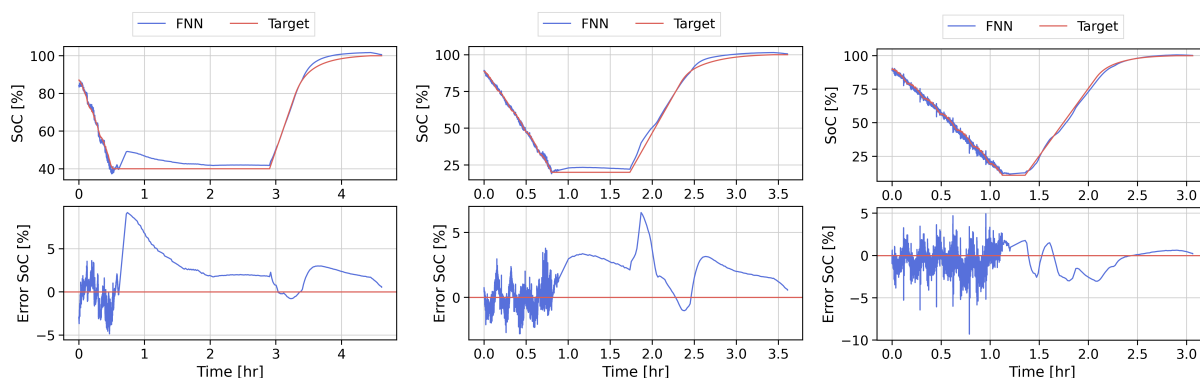


Figure 7.3: The legend holds information about the validation cycle used if CCCV charging data was included, and the two searches where the window size was increased to 4800, instead of 1200. More detailed information about each search number is found in Table 7.2.

- Search number 20 - 23: Current and voltage at an instant time-steps are found beneficial to include as input features. The RMSE drooped with a 9-12% when including instant time-steps. In addition, averaging over 8 minutes reduces the error by 50%, and was therefore used as discussed previously.

It should be stated that not all combinations are tried for each search. Therefore the best networks found within a search does not necessarily mean that the best network within the search space is found but is used as an indication of decent combinations.



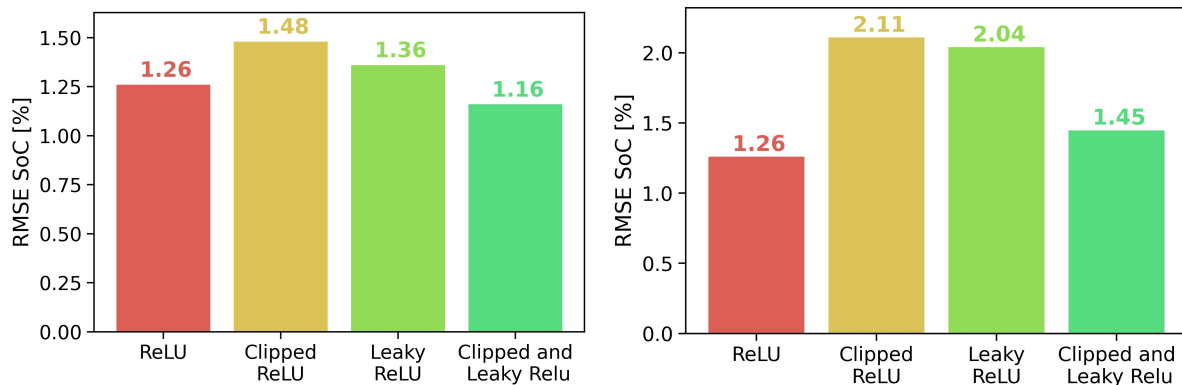
(a) -20°C , RMSE: 1.73, MAE: 1.37, **(b)** 0°C , RMSE: 0.92, MAE: 0.75, **(c)** 25°C , RMSE: 1.16, MAE: 0.95, MAX: 9.19

Figure 7.4: The model with the lowest RMSE from random search, number 23, is validated on the US06 drive cycle. The overall RMSE for all five temperatures is 1.26 SoC [%], as stated in Table 7.2. The error scores for each temperature are stated in the caption of each plot, with the units SoC [%].

In ML a lot of research is put into optimizing hyperparameters [133–135]. Despite this being an important topic, there was no time to research the hyperparameter tuning any further in this thesis. Unfortunately, no conclusion of the optimal number of layers for estimating SoC with FNN can be drawn. However, since the error scores of using three hidden layers yielded a slightly overall lower error than two hidden layers, and, to have time to research other important aspects of SoC estimation with FNN, it was determined to use the best network from search number 23. As stated at the beginning of this section, this was a network with three hidden layers with 4, 28, and 32 nodes per layer, and with a drop out of 20% between the last layer and the output layer. It was concluded to continue further work with these hyperparameters. In addition, it was decided to include the instant time measurements as input features yielding a total of five input features; V_{4800} , i_{4800} , V , i and T .

To summarize the results of this section, Figure 7.4 displays the predicted and target SoC plotted against time. The error is displayed underneath. It is clear that the FNN have the ability to follow the target curve, but there are a few high errors. Therefore, the next sections investigate techniques to improve the FNNs robustness.

7.3.4 Clipped ReLU and Leaky ReLU



(a) Optimized by backpropagation with Adam optimizer.

(b) Optimized the FNNs that correspond to three right-most scores by backpropagation with stochastic gradient descent.

The activation functions were not a focus in the random search where ReLU was quickly established to be preferred over linear and sigmoid activation function. In this section, two other activation functions are explored, clipped and leaky ReLU. Using both at once yielded a lowered RMSE as can be seen in Figure 7.5a. However, when only implementing one of them, the error is higher than the benchmark case where only ReLU is used as the activation function. The reason for this is not understood. In addition, to optimize with the backpropagation algorithm stochastic gradient decent³ was tried for the new activation functions. This resulted in an increased RMSE. Interesting, the trend in error shows the same tendency; that clipped and leaky ReLU together

³Default Keras parameters was used for stochastic gradient decent.

lowers the RMSE⁴.

The clipped ReLU was applied as an effort to stop the network from predicting a higher SoC than 100%. As seen from Figure 7.4, the network overshoots at the end of the charging cycle for -20°C and 0°C . The ReLU is therefore clipped at 1. In Figure 7.6, the effect of clipped ReLU is seen; when the network tries predicting higher than 100% SoC it is clipped. This yields a reduced error at the very end of the cycle (right before 4 hours). The downside is that it predicts a straight line of 100% SoC from 3.5 hours and until the end of the predictions. 3.5 hours is approximately the point where the current is lowered due to v_h being reached (CCCV charging). To have a SoC estimator in an EV that predicts a fully charged battery when in reality it is nearly 10% from fully charged, would be a catastrophe for the battery management system (the battery would be happy though since it would never be fully charged and therefore have a prolonged lifetime, if not the consequence is that the user always discharges it fully instead).

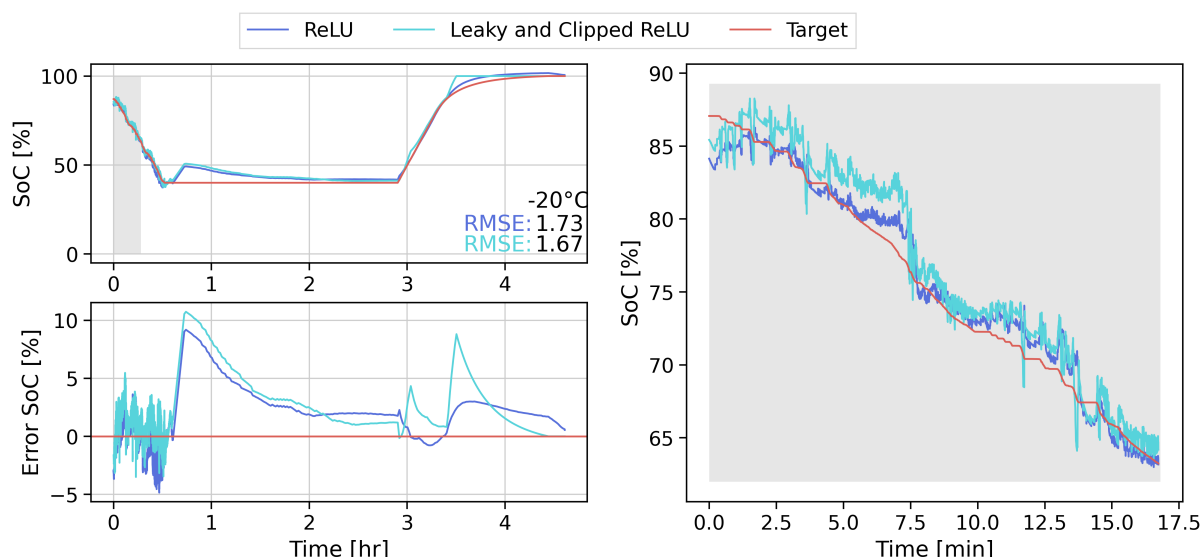


Figure 7.6: US06 -20°C is plotted. This is the cycle with highest RMSE, and have a MAX error of nearly 10.

Despite the high error at around 3.5 hours, the total RMSE is lowered by using clipped and leaky Relu in combination. In addition, at the start of the pause where the error is high for -20°C , the clipped and leaky ReLU together reduces the error by around 20%. Therefore, it was chosen to proceed with this configuration. This model will from this point onward be called the base network, and the model, but with randomly initiated weights is called the base structure.

⁴The benchmark case was not optimized with stochastic gradient decent.

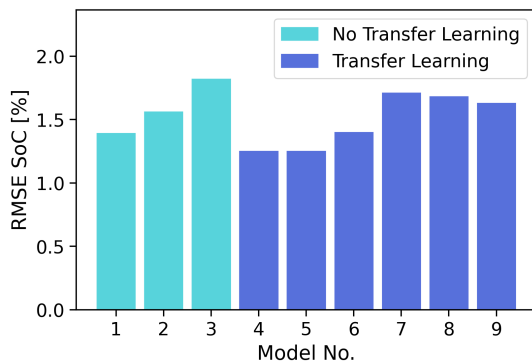
7.4 Transfer Learning

Transfer learning was done on the base model, yielding lowered RMSE results for all the three new data sets (LG, Samsung, and Turnigy).

Despite the lowered RMSE, the MAE is increased by transfer learning compared to randomly initiating the network. This indicates that when using transfer learning, the number of large errors is reduced, but on average the offset from the target SoC is larger. This cannot be observed directly from the plotting of the results as done in Figure 7.11, but it is the only possible explanation. It was not understood why the MAE is higher, and RMSE is lower, but it is a tendency for all the data sets. It is interesting to note that model six has the same learning rate, and no frozen layers, implying that the only difference between this network and models 1-3 is the initialization.

It was chosen to go forward with training by transfer learning. This was based on the choices to use RMSE as the most important metric, and transfer learning lowered this metric. Of all the models, model number five yielded the lowest RMSE. It was lowered by 10%, 23%, and 14% for LG, Samsung, and Turnigy, respectively. Moreover, it had the lowest MAE of the networks where transfer learning was utilized, and one of the lowest MAX errors. However, model number five also had a significantly higher MAE than the models trained on randomly initiated weights (34%, 19%, and 45% for LG, Samsung, and Turnigy, respectively).

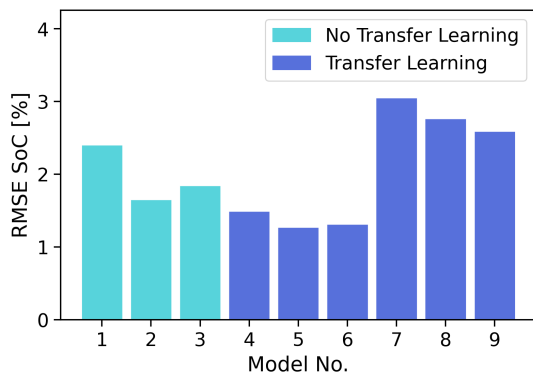
To start with, it was tried to train the network by freezing two hidden layers. It resulted in high errors, and these results are not included in the thesis. However, an interesting approach would have been to unfreeze the layers, to further train with a lower learning rate.



Model No.	LR	Freezed layers	RMSE SoC[%]	MAE SoC[%]	MAX SoC[%]	Val RMSE
1	1e-3	NA	1.39	1.06	10.40	1.15
2	1e-3	NA	1.56	1.22	9.21	1.25
3	1e-3	NA	1.82	1.40	13.86	1.22
4	1e-5	0	1.25	1.64	10.53	1.16
5	1e-4	0	1.25	1.60	11.55	1.19
6	1e-3	0	1.40	1.78	12.57	1.21
7	1e-5	1	1.71	2.48	22.35	1.68
8	1e-4	1	1.68	2.46	22.38	1.67
9	1e-3	1	1.63	2.37	21.88	1.69

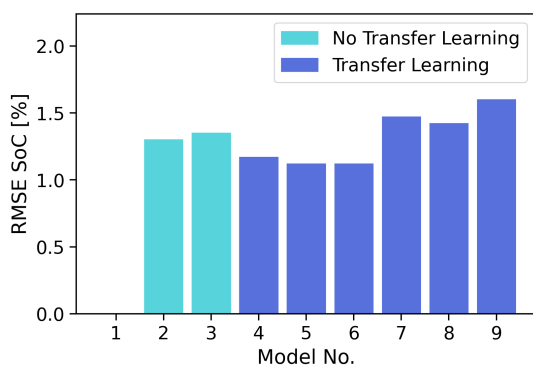
Figure 7.7 & Table 7.3: LG battery. LR = Learning Rate.

It should be noted that there was a large MAX error the first time predicting with the test data of the LG and Turingy batteries. It was the same MAX error (80% and 91%) for all models. It was concluded that the networks predicted a 0% SoC due to faulty voltage measurements, while the actual SoC was high. When this large error was noticed, the MAX error when predicting with the validation and training data was also calculated, yielding excessive errors. Therefore, further cleaning of all the data was inevitable. A complete explanation of the re-cleaning is found



Model No.	LR	Freezed Layers	RMSE SoC[%]	MAE SoC[%]	MAX SoC[%]	Val RMSE
1	1e-3	NA	2.39	1.67	16.88	1.37
2	1e-3	NA	1.64	1.16	14.8	1.32
3	1e-3	NA	1.83	1.41	10.88	49.42*
4	1e-5	0	1.48	2.09	20.07	1.75
5	1e-4	0	1.26	1.76	13.32	1.70
6	1e-3	0	1.30	1.81	13.49	1.51
7	1e-5	1	3.04	4.07	33.74	3.50
8	1e-4	1	2.75	3.69	32.53	3.29
9	1e-3	1	2.58	3.60	23.45	2.97

Figure 7.9 & Table 7.3: Samsung. LR = Learning Rate. *Do not know why this score is 50 times higher than the rest. Most likely a human error in the saving process.



Model No.	LR	Freezed Layers	RMSE SoC[%]	MAE SoC[%]	MAX SoC[%]	Val RMSE
1	1e-3	NA	60.19	54.53	100.0	0.89
2	1e-3	NA	1.30	0.93	18.91	0.78
3	1e-3	NA	1.35	0.90	28.63	0.83
4	1e-5	0	1.17	1.69	23.66	1.05
5	1e-4	0	1.12	1.64	23.23	0.96
6	1e-3	0	1.12	1.65	25.61	1.03
7	1e-5	1	1.47	1.94	22.43	1.38
8	1e-4	1	1.42	1.90	22.25	1.28
9	1e-3	1	1.60	2.10	22.17	1.31

labelformat=andtable

Figure 7.10: Turingy. LR = Learning Rate. As seen in the table, model number one did not converge, and hence the RMSE is not plotted.

in Section 5.2.2. Due to time limitations, the models were not retrained with the rewashed data even though it could have been beneficial. However, it was much less than 1% of the points that were off, and the model learned the battery parameters, even when training on the faulty data. Since the model of the best epoch for each network was saved, it was reloaded and the test error scores recalculated as stated in Table 7.7-7.10.

On the bright side, the fact that the network yielded these large errors at the anomalies, confirms that the network is capable of learning the battery dynamics to a large extent.

Moving back from this sidetrack, transfer learning has potential. However, more research should be done on why the MAE becomes large. Previously, transfer learning for a better SoC estimation has been researched in a couple of papers; in [136] they implemented transfer learning for SoC estimation with time convolutional neural networks and in [112] for RNN with LSTM cells. Both works state an increased prediction accuracy with transfer learning, and the work in this thesis supports that transfer learning can be an efficient technique in the field of estimating SoC by

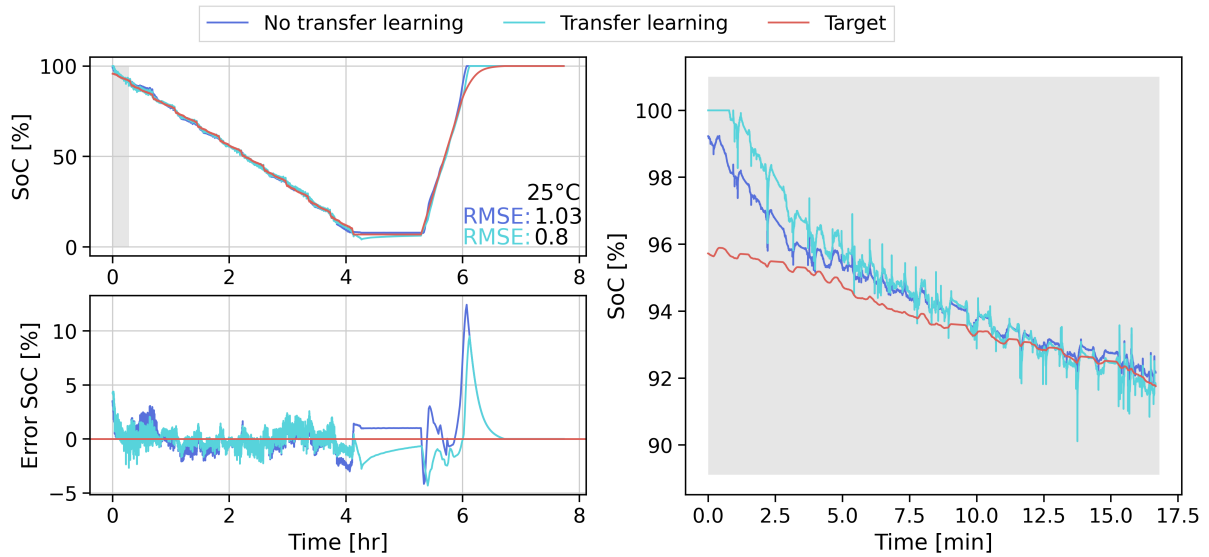


Figure 7.11: Predicted SoC on drive cycle yielding the lowest RMSE. UDDS LG battery. The right plot is a zoomed in plot of the gray shaded area in the upper left plot.

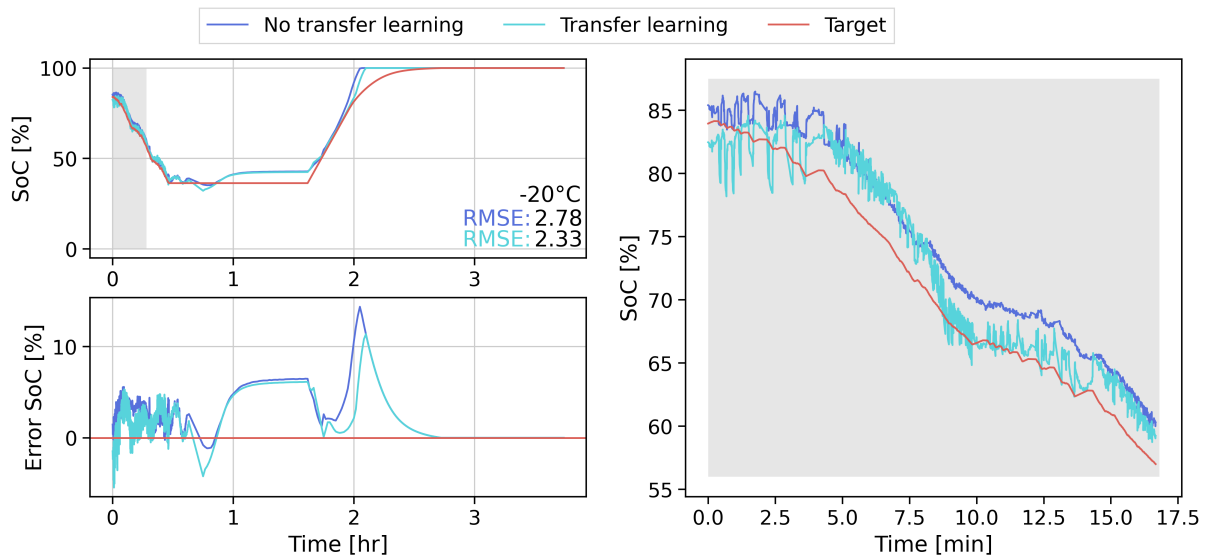


Figure 7.12: LG battery -20°C US06.

NNs.

7.5 Augmented data

The test scores after trained and validated on augmented data did not result in decreased error scores (RMSE > 1.5%). However, the validation RMSE was decreased. This could indicate that the reason for the higher test score is an overfitted network and that it was overfitted on the noisy data. The validation data was also used with the augmented data, while the test data had no noise. Therefore, an attempt was made to retrain the model on augmented data, but by using validation data without any noise to check if the model stops at a different epoch. However, this resulted in even larger error measurements.

As written in Section 6.5.1 and Section 4.2, two other papers, [109] and [67], state that adding noise to the data is beneficial. This thesis does not disprove the effect of using augmented data, nor can it confirm any benefits. It might be because the network is too complex and that it overfits on the noise. However, in [67] which was also trained on Gaussian noise, the network is of similar complexity as in this thesis⁵. Another reason could be that the noise was too large. By visually inspecting the augmented data depicted in Section 6.5.1, in addition to comparing the noise inserted to the data for this thesis to previous work, too large noise for the voltage and temperature does not seem to be the cause. However, the noise of the current data had a standard deviation of 0.03 - 0.41 depending on the cycle. This would yield Gaussian noise with a magnitude up to ten times larger than in [67], and is most likely the cause of the poor result. More research is required to obtain what type of augmented data is beneficial for estimating SoC with FNN. A recommendation for further studies is to lower the noise of the current from what was used in this work.

7.6 Temperature Difference as Input Parameter

The results from using the temperature difference (ΔT) between the ambient temperature and the battery temperature as an additional input to the base features were promising. The best result was obtained when averaging this temperature difference ($\overline{\Delta T}$) where the RMSE and MAE decreased by 16 % and 7%, respectively. However, the most significant observation was the decrease in MAX error by 27%. The percentage is calculated based on the benchmark case with the lowest error (benchmark case with transfer learning). The lowest error from each experiment is stated in Table 7.13 and the RMSE is illustrated by a bar plot in Figure 7.13.

The best models from the random search with ΔT was a model trained with three hidden layers of 34, 42, and 22. A similar network architecture was obtained after random search with the $\overline{\Delta T}$ with 26, 28, and 16 nodes per hidden layer. None of the networks were trained with dropout layers. The result yields a reduced MAE and RMSE for both ΔT and $\overline{\Delta T}$.

Two biases are present and are discussed:

1. The benchmark case was not optimized by random search as ΔT and $\overline{\Delta T}$ were. Since the

⁵In [67] they used FNN with three hidden layers with 8, 16, and 32 nodes per layer. In comparison, this thesis use three hidden layers with 4, 28, and 32 nodes per layer.

pauses in the training and test data were taken away, different hyperparameters than the base network (that was trained with pauses) could have been optimal.

2. The benchmark cases are constructed with leaky ReLU and clipped ReLU, while ΔT and $\overline{\Delta T}$ are with simple ReLU. However, since implementing leaky ReLU and clipped ReLU showed better results than only ReLU, this should favor the benchmark for obtaining a low error. Consequently, the higher error of the benchmark cases indicates that the ΔT and $\overline{\Delta T}$ are adequate input features.

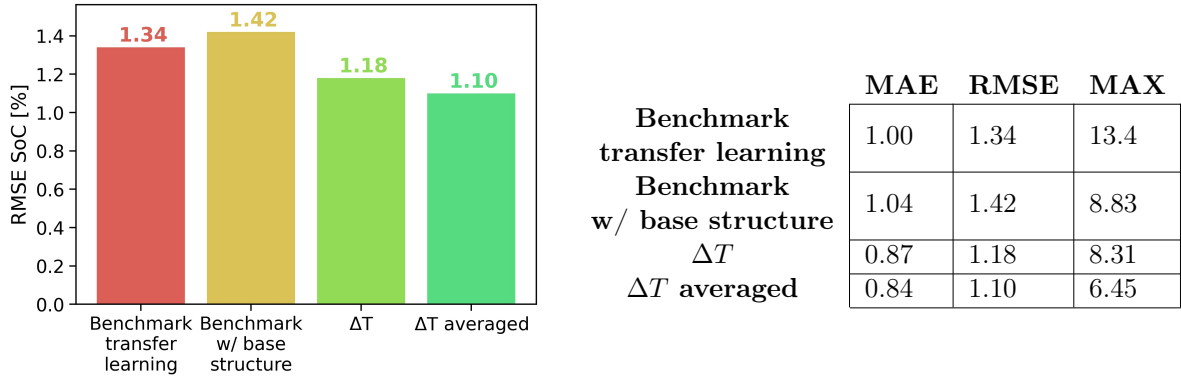


Figure 7.13 & Table 7.4: Error scores for the two benchmark cases (transfer learning and base structure), and the two models with the lowest RMSE after optimizing by random search with ΔT and $\overline{\Delta T}$ used as additional input features.

To evaluate the effect of the new input feature $\overline{\Delta T}$, the case yielding the highest RMSE (3.00% SoC) is analysed individually. It is the drive cycle US06 cycled at -20°C that has the highest RMSE and is plotted in Figure 7.14. Even though the benchmark case have an overall lower RMSE, the MAX error is significantly higher. From the figure it can be observed that the highest error for the $\overline{\Delta T}$ network occurs when relaxation of the battery, and the voltage increase despite no current input due to the overpotentials. As previously written, the majority of this pause was removed⁶. Therefore, if trained on more data with pauses, the network might learn the relaxation dynamics of the battery which would reduce this error. Another high error is situated in the end of the cycle where the network overshoots. Implementing clipped ReLU might fix this problem.

As a comparison to the case of the highest RMSE, also the case yielding the lowest error was plotted. The benchmark case have MAX errors four times as large as the $\overline{\Delta T}$ case, and a doubled RMSE. The RMSE and MAE of all temperatures, all cycles, and for both benchmark cases and the ΔT -network can be found in Appendix C.0.2.

The last figure that is exceptionally noteworthy is of the SoC curve of the benchmark case trained with the base structure (*e.g.*, randomly initiated weights). It has a slightly higher MAE and RMSE, but a large decrease in the MAX error score. When plotted, it is clearly seen that the MAX error score is reduced due to its ability to follow the charging curve when moving towards

⁶The straight line in Figure 7.14 between 0.7 hr and 1.7 hr is where the pause data was removed. The line is simply a line between two data points that are an hour apart in time.

a high SoC.

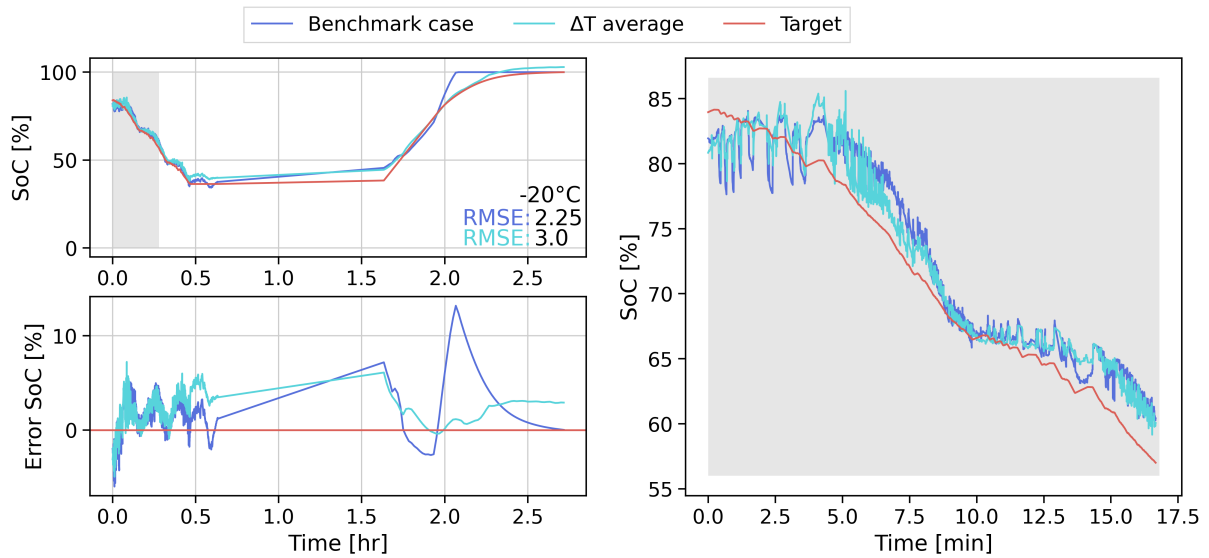


Figure 7.14: Highest RMSE of the $\overline{\Delta T}$ case was found for -20°C US06, which is the cycle that is plotted.

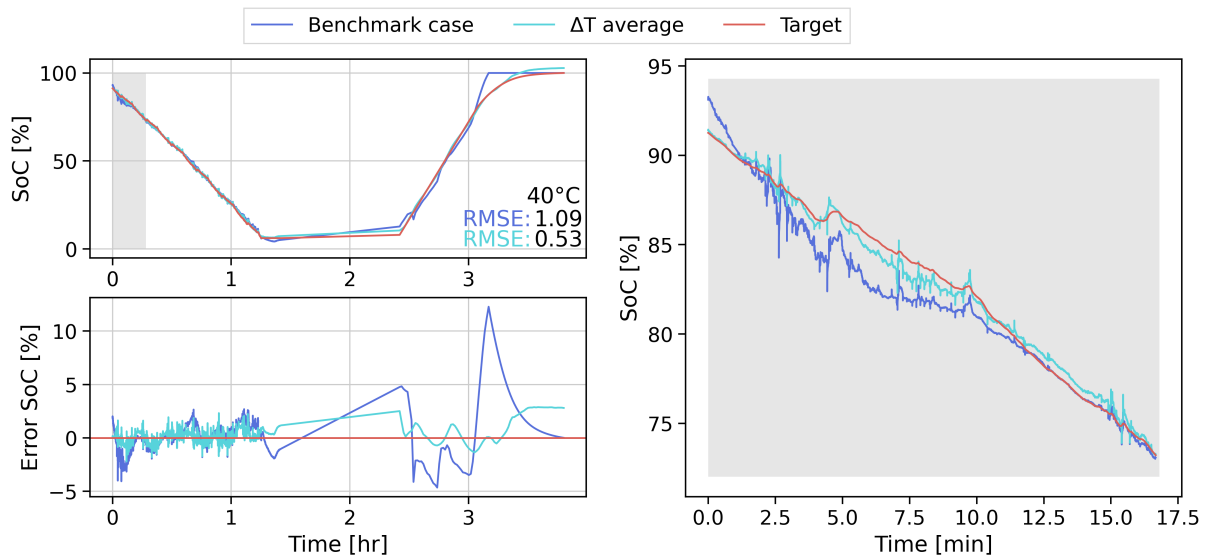


Figure 7.15: Case with the lowest RMSE for $\overline{\Delta T}$ is plotted. It is the LG battery at 40°C cycled by the drive cycle HWFET. The benchmark case is from transfer learning.

A note should be made on the sampling time. In the graphs, the SoC is plotted against time. However, it should be noted that the sampling frequency of the charging cycles is 600 times higher than for the drive cycles, implying that the amount of data trained on for charging cycles is drastically lower than for the drive cycles. This might be the reason for the high error scores of the charge cycle. If trained on more charging data, the error might reduce. One simple way is reusing the charging data that already exist since the amount of charging data is disproportional to the discharging data.

From visually observing the graphs, the benchmark case constructed from the base structure seems to be more accurate. However, the overall RMSE is less, while the MAX error is higher. This is probably due to the sampling frequency being lower at the charge state. To obtain an RMSE that takes this into account can be done by scaling each charging RMSE by 600. Then, the randomly initialized network would most likely yield the lowest error scores.

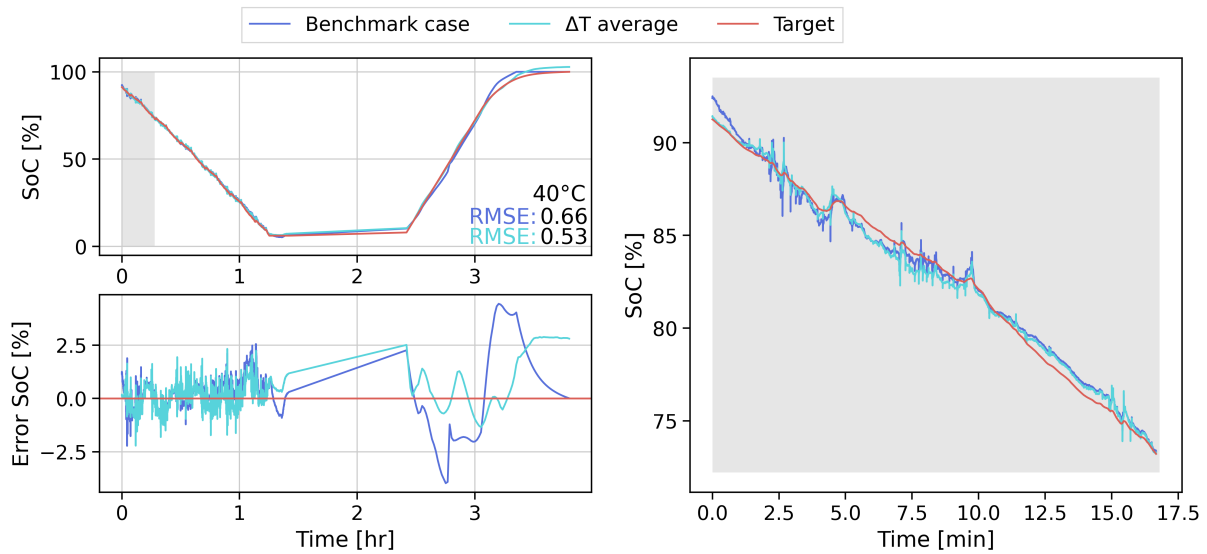


Figure 7.16: SoC estimated for 40 °C HWFET. The benchmark case have Randomly initiated weights

7.7 RNN Results

An RNN was constructed to test if it could be more beneficial than an FNN. However, the hyperparameters were not optimized correctly and the results are unsatisfactory.

As seen from Figure 7.17a and 7.17b, the network overshoots the estimation at high SoC. There-

fore, a clipped ReLU was tried. Unfortunately, the error of the loss function diverged when a ReLU clipped at one was implemented. The only time the network converged, was when setting the number of units to one. However, the model with this configuration only predicted a constant value. Other configurations that were tried with the standard ReLU was constructing two layers with 5-50 nodes per layer and one layer with 1-200 nodes with a dropout ranging from 0-20%. The models either did not converge, or yielded high error scores.

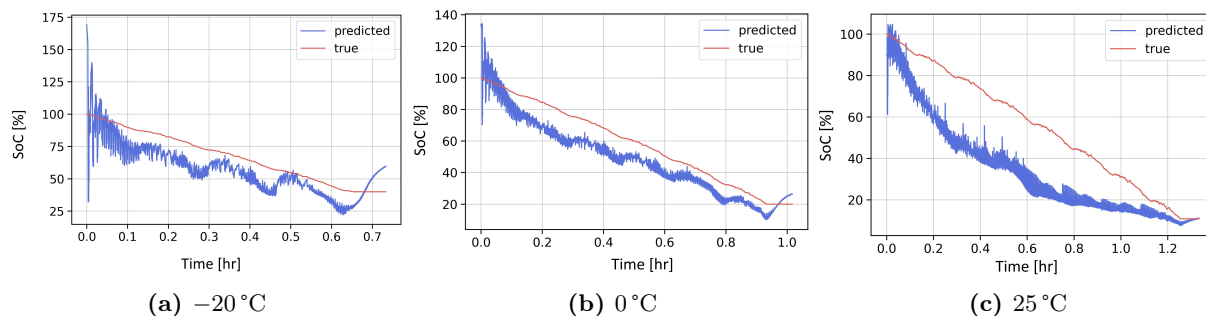


Figure 7.17: Results from predicting with an RNN with the Panasonic data set on the drive cycle .

To improve these results, several adjustments could be made. It can be seen in Figure 7.17c, the SoC prediction have a flat profile towards the end. The reason could be that the model have learned that the last time step is very similar to the current time step. Therefore, the model would have a strong relation to its last prediction rather than a strong relation to the battery data of the current time step. An adjustment that would be beneficial to avoid this reliance on the last state is to not use a stateful model. By setting stateful to False, the state would not be passed forward to the next epoch, and the models memory would be limited in time. Then, the model would be forced to rely on less data from the past, and more data from the current time step. For instance, a memory of 8 minutes would be interesting to use due to the research done on averaging windows for FNNs.

As stated above, another challenge with this RNN was that it frequently yielded very large errors ("NaN") after training on a few drive cycles. In Section 3.3.6 it was written that RNNs are prone to the vanishing gradient problem. Since the RNN made in this thesis had a long-term relationship from the start of an cycle to the end of it (many hundred thousand points), reducing the input sequence could help. For instance, by drastically shortening the memory to only contain the last 8 minutes, the large error most likely caused by the vanishing gradient problem might be fixed.

Finally, hyperparameter tuning by random search or another search algorithm could have been beneficial.

Conclusion

As in many other fields of research, it is difficult to compare results. This includes both the results obtained within this thesis and the results stated in other papers. The former is bottoming in two challenges. One concerning the random nature of Neural Networks (NNs), and the second concerning the network-tuning. The knowledge about whether or not the best network hyperparameters were found and the optimal initialization, remains unknown to this date. The latter difficulty of comparing papers is the many factors influence the result. For instance, the battery chemistry used, the way data preprocessing was done, the calculation of target State of Charge (SoC), whether or not both charging and discharging is included in the data set, and the error metrics are some of the factors.

The aim of this thesis was to research different techniques and input features that could improve SoC estimation by NNs. Three techniques have been investigated; random search for optimizing hyperparameters, transfer learning and augmented data to improve accuracy and robustness. Tuning the network by random search proved to be effective in comparison to systematically change one by one parameter. Other optimization algorithms as grid search could with advantage be tried to narrow the search further. The second technique, transfer learning, showed promising results with a decreased Root Mean Square Error (RMSE). However, the Mean Average Error (MAE) increased and research is required to understand this mechanism and if the trend observed in this thesis is a general trend. The last technique tried was adding Gaussian noise to the data. This increased the error and it is difficult to arrive at any conclusions concerning the benefit of augmented data. Previous work state it is effective, and it is assumed that the Gaussian noise added to the current data of this thesis was too large.

Moving onto the work on the input features, the averaging window size for current and voltage was researched. Moreover, it was research if using the heat generated by the battery as an input feature is beneficial. For the former, it was found that averaging over 8 minutes yielded the lowest error. This number is both based on this thesis and previous work. The latter research on input features was to improve the method to couple electrical and thermal properties. It was done by inputting the temperature difference between the battery and its surroundings yielded an approximation of the heat generation in the cell. To the authors knowledge, this is

the first time heat generation as an input feature have been used for SoC estimation with NNs. The results were promising and the lowest error was obtained when averaging the temperature difference. The overall MAE, RMSE, and MAX error dropped by 16%, 7%, and 27%, respectively. Due to the uncertainty of an ideal structure of the benchmark case for this study, more experiments with this feature should be conducted to validate the result. However, this thesis indicates that the heat generated in the battery has a correlation to SoC which the FNN captures.

Concerning the network's ability to learn battery dynamics, the network learned them to a large extent. The results indicate that the greatest challenge is to learn the dynamics of the LiB during relaxation, especially for low temperatures. To improve this issue it is suggested to training specifically on more data with pauses. Another challenging part to estimate was at the end of the charging cycle where the voltage has reached its maximum voltage and charging by a descending current. Implementing a clipped ReLU diminish this error. However, it did not solve the problem completely. Despite these remaining challenges, a confirmation that the network had learned the battery dynamics was obtained when predicting with anomalies that the author had not observed in the data.

Despite the network learning the general battery dynamics, there is still a way to go for optimizing the FNN. The lowest MAX error obtained in this thesis is 6.45% SoC. Relating this work to the practical SoC estimator in EVs, it can be critical for the user to know the exact SoC, especially when the SoC is low. Therefore, further research to reduce the MAX error is required.

Going off from the scientific perspective to the sake of research and learning, another conclusion is drawn. This thesis has resulted in an increased knowledge basis of the author. However, the research could advantageously be more tapered. For instance, the research could have been limited to a specific topic as transfer learning, SoC estimation on aged cells, optimal input features, or estimating with only one neural network type. It is a delicate balance since the results on the temperature difference were obtained as the last try to contribute to research, which was found to be the most promising result of this research.

Further Work

The results from this thesis show that FNNs are able to learn the major battery dynamics needed to predict the SoC. However, the MAX error is still significant and further investigations should focus on techniques to reduce this error. The work includes determining what type of augmented data is efficient, the optimal procedure to do transfer learning, and coupling electrical, thermal and aging properties.

A list of specific implementations which could have improved this work are:

- Training the NN on a relatively larger amount of charging and pause data to decrease the MAX error which typically occurred in these phases (in this thesis the amount of discharging data had two orders of magnitude more samples than the charging data).
- To reduce the MAX errors in general, this metric could have been included in the loss function.
- When creating augmented data, decreasing the amount of noise put on the current by a factor of 10 from what was done in this work. In addition, further studies on constant off-sets, constant gain, and random noise on the training data could be beneficial.
- Testing the networks robustness for real-world driving conditions containing noisy measurements would be beneficial. This could be done by adding noise to the test data.
- When training with transfer learning were some layers are frozen, an interesting approach would be to unfreeze the layers and continue training with a reduced learning rate. In addition, training each layer with different learning rates could improve the transfer learning. For instance, for a network of three hidden layers a learning rate of 10^6 , 10^5 , and 10^3 per layer would be interesting
- For predicting the SoC on aged batteries (*e.g.*, the LCO battery data from this thesis), the internal resistance would be an interesting input feature.
- Build an RNN that is not stateful and with input sequences of around 8 minutes.
- Research the pros and cons of GRU *vs.* LSTM, or other RNNs.
- To provide a more user-friendly SoC estimate, a simple and pragmatic way is using the Wh as a capacity measurement instead of the Ah. The SoC could be defined as the ratio between the Wh drawn/inputted and the total Wh before the LiB reached its cut-off voltage.

Moving on to more extensive improvements, increasing the accuracy of the target SoC is an important research topic. Even though the accuracy of the measurement equipment is high, the charge/discharge efficiency is not easily estimated. In supervised machine learning, the model will not be better than its target value. This implies that training NNs for SoC predictions, an accurate target SoC is essential. In this thesis, as well as the work stated in the literature review in Chapter 4, it is shown that the RMSE and MAE have an order of magnitude of 1 compared to the target value. Therefore, the error of Coulomb counting contributes to an inaccuracy in the stated NN prediction error. For instance Titan Advanced Energy Solutions states to have a device that can measure the SoC with an accuracy of 99.9% SoC[137]¹. Using these type of devices to obtain the target SoC would be beneficial.

For a user-friendly SoC estimator in an EV, the estimation of the km-range should be research. A classic conversion could be made between the SoC and the km-range. However, since estimating the km-range increases the complexity of SoC estimation, the advantages of ML might be more prominent. Utilizing ML models in parallel where for instance one model predicts the SoC and another model predicts the future power consumption of the car, and then combining the models in the end to estimate the km-range, would be an interesting research. Then parameters like the weather forecast (*e.g.*, wind, temperature, sun, and precipitation), the topography of the road the driver is about to take, the drivers profile (*e.g.*, learned the drivers earlier driving habits), and satellite photos of the traffic could be inputted. When cars become fully autonomous, the SoC estimation might become slightly simplified since influence of the drivers driving habits can be neglected.

Heat generation in the battery as an additional input feature might prove an important area for further research. Today, there are ongoing research to improve temperature measurements inside the battery cell. For instance optic fiber sensing [138] and research on heat generation through battery cells are investigated. To include these type of measurements could increase the robustness and accuracy of the SoC estimator.

To summarize, the list of further work is long, and the research field of SoC estimation is far from reaching its finish line. Along with machine learning algorithms and the battery research for understanding the operational mechanisms constantly improving, SoC estimation have an exciting journey ahead!

¹A mail was sent questioning the manufacturer what the accuracy score was based on and how it was calculated, but no response was gotten.

Bibliography

- [1] H. Ritchie and M. Roser, *Renewable energy*, Dec. 2017. [Online]. Available: <https://ourworldindata.org/renewable-energy>.
- [2] Z. Jalili, K. W. Krakhella, K. E. Einarsrud and O. S. Burheim, 'Energy generation and storage by salinity gradient power: A model-based assessment,' *Journal of Energy Storage*, vol. 24, p. 100755, 2019.
- [3] Iea, *Renewables – global energy review 2020 – analysis*, Apr. 2020. [Online]. Available: <https://www.iea.org/reports/global-energy-review-2020/renewables>.
- [4] I. E. A. International Energy Agency and W. Bank, *Sustainable Energy for All 2013-2014: Global Tracking Framework Report*. The World Bank, 2014, p. 194.
- [5] *Renewable energy*, Apr. 2020. [Online]. Available: <https://www.c2es.org/content/renewable-energy/>.
- [6] *Barriers to renewable energy technologies*, Jun. 2014. [Online]. Available: <https://www.ucsusa.org/resources/barriers-renewable-energy-technologies>.
- [7] *A designer's guide to lithium (li-ion) battery charging*, Sep. 2016. [Online]. Available: <https://www.digikey.no/no/articles/a-designer-guide-fast-lithium-ion-battery-charging?utm-adgroup=Battery+Products>.
- [8] M. A. Hannan, M. H. Lipu, A. Hussain and A. Mohamed, 'A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations,' *Renewable and Sustainable Energy Reviews*, vol. 78, pp. 834–854, 2017.
- [9] B. F. Heyer, *One meter battery tester*, US Patent 2,225,051, Dec. 1940.
- [10] D. N. How, M. Hannan, M. H. Lipu and P. J. Ker, 'State of charge estimation for lithium-ion batteries using model-based and data-driven methods: A review,' *IEEE Access*, vol. 7, pp. 136116–136136, 2019.
- [11] M. A. Hannan, M. M. Hoque, A. Hussain, Y. Yusof and P. J. Ker, 'State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations,' *Ieee Access*, vol. 6, pp. 19362–19378, 2018.

- [12] A. F. Gunnarshaug, P. J. S. Vie and S. Kjelstrup, ‘Reversible heat effects in cells relevant for lithium-ion batteries,’ *Journal of The Electrochemical Society*, 2021.
- [13] F. Richter, A. Gunnarshaug, O. S. Burheim, P. J. Vie and S. Kjelstrup, ‘Single electrode entropy change for LiCoO_2 electrodes,’ *ECS Transactions*, vol. 80, no. 10, p. 219, 2017.
- [14] D. Kim, K. Koo, J. Jeong, T. Goh and S. W. Kim, ‘Second-order discrete-time sliding mode observer for state of charge determination based on a dynamic resistance li-ion battery model,’ *Energies*, vol. 6, pp. 5538–5551, Oct. 2013. DOI: 10.3390/en6105538.
- [15] *Chemical equilibrium*, Apr. 2021. [Online]. Available: <https://www.britannica.com/science/chemical-equilibrium>.
- [16] H. Abdi, R. Rasouli Nezhad and M. Salehemaleh, ‘Chapter 5 - fuel cells,’ in *Distributed Generation Systems*, G. Gharehpetian and S. M. Mousavi Agah, Eds., Butterworth-Heinemann, 2017, pp. 221–300, ISBN: 978-0-12-804208-3. DOI: <https://doi.org/10.1016/B978-0-12-804208-3.00005-4>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9780128042083000054>.
- [17] C. Honsberg and S. Bowden, *Battery capacity*, 2020. [Online]. Available: <https://www.pveducation.org/pvcdrom/battery-characteristics/battery-capacity>.
- [18] J. T. Warner, *Lithium-Ion Battery Chemistries: A Primer*. Elsevier, 2019.
- [19] B. University, *Types of lithium-ion batteries*, 2020. [Online]. Available: <https://batteryuniversity.com/learn/article/types-of-lithium-ion>.
- [20] U. of Maryland, *Calce battery research group*, 2017. [Online]. Available: <https://web.calce.umd.edu/batteries/data.htm>.
- [21] Z. Salameh and B. Kim, ‘Advanced lithium polymer batteries,’ in *2009 IEEE Power & Energy Society General Meeting*, IEEE, 2009, pp. 1–5.
- [22] *Lithium-ion vs lithium-polymer: What’s the difference?* Oct. 2020. [Online]. Available: <https://www.androidauthority.com/lithium-ion-vs-lithium-polymer-whats-the-difference-27608/>.
- [23] B. Scrosati, F. Croce and S. Panero, ‘Progress in lithium polymer battery r&d,’ *Journal of power sources*, vol. 100, no. 1-2, pp. 93–100, 2001.
- [24] G. L. Plett, *3.1.3: How do we define soc carefully*, 2020. [Online]. Available: <https://www.coursera.org/learn/battery-state-of-charge>.
- [25] O. S. Burheim, ‘Engineering energy storage,’ in. Academic press, 2017, pp. 111–145.
- [26] V. J. Ovejas and A. Cuadras, ‘Effects of cycling on lithium-ion battery hysteresis and overvoltage,’ *Scientific Reports*, vol. 9, no. 1, 2019. DOI: 10.1038/s41598-019-51474-5.
- [27] N. Watrin, B. Blunier and A. Miraoui, ‘Review of adaptive systems for lithium batteries state-of-charge and state-of-health estimation,’ in *2012 IEEE Transportation Electrification Conference and Expo (ITEC)*, IEEE, 2012, pp. 1–6.
- [28] M. Alamgir and A. M. Sastry, ‘Efficient batteries for transportation applications,’ SAE Technical Paper, Tech. Rep., 2008.

- [29] F. Huet, 'A review of impedance measurements for determination of the state-of-charge or state-of-health of secondary batteries,' *Journal of power sources*, vol. 70, no. 1, pp. 59–69, 1998.
- [30] V. Agubra and J. Fergus, 'Lithium ion battery anode aging mechanisms,' *Materials*, vol. 6, no. 4, pp. 1310–1325, 2013.
- [31] M. T. Lawder, P. W. Northrop and V. R. Subramanian, 'Model-based sei layer growth and capacity fade analysis for ev and phev batteries and drive cycles,' *Journal of The Electrochemical Society*, vol. 161, no. 14, A2099, 2014.
- [32] M. Petzl, M. Kasper and M. A. Danzer, 'Lithium plating in a commercial lithium-ion battery—a low-temperature aging study,' *Journal of Power Sources*, vol. 275, pp. 799–807, 2015.
- [33] Q. Liu, C. Du, B. Shen, P. Zuo, X. Cheng, Y. Ma, G. Yin and Y. Gao, 'Understanding undesirable anode lithium plating issues in lithium-ion batteries,' *RSC advances*, vol. 6, no. 91, pp. 88 683–88 700, 2016.
- [34] N. Omar, M. A. Monem, Y. Firouz, J. Salminen, J. Smekens, O. Hegazy, H. Gaulous, G. Mulder, P. Van den Bossche, T. Coosemans *et al.*, 'Lithium iron phosphate based battery—assessment of the aging parameters and development of cycle life model,' *Applied Energy*, vol. 113, pp. 1575–1585, 2014.
- [35] *What is battery self-discharge? and how do you counter it?* [Online]. Available: <https://www.panasonic-eneloop.eu/en/news/what-battery-self-discharge-and-how-do-you-counter-it>.
- [36] P. Kollmeyer, 'Samsung inr21700 30t 3ah li-ion battery data,' *Mendeley Data*, vol. 1, 2020. DOI: 10.17632/9xyvy2nj3.1.
- [37] S. Zhang, K. Xu and T. Jow, 'Low temperature performance of graphite electrode in li-ion cells,' *Electrochimica acta*, vol. 48, no. 3, pp. 241–246, 2002.
- [38] R. Bugga, M. Smart, J. Whitacre and W. West, 'Lithium ion batteries for space applications,' in *2007 IEEE Aerospace Conference*, 2007.
- [39] C. Vidal, O. Gross, R. Gu, P. Kollmeyer and A. Emadi, 'Xev li-ion battery low-temperature effects,' *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4560–4572, 2019.
- [40] M. Smart, B. Ratnakumar and L. Whitcanack, 'Performance of low temperature electrolytes in experimental and prototype li-ion cells,' in *5th International Energy Conversion Engineering Conference and Exhibit (IECEC)*, 2007, p. 4744.
- [41] J. Jaguemont, L. Boulon and Y. Dubé, 'A comprehensive review of lithium-ion batteries used in hybrid and electric vehicles at cold temperatures,' *Applied Energy*, vol. 164, pp. 99–114, 2016.
- [42] Z. Li, J. Huang, B. Y. Liaw and J. Zhang, 'On state-of-charge determination for lithium-ion batteries,' *Journal of Power Sources*, vol. 348, pp. 281–301, 2017.
- [43] J. T. Warner, *The handbook of lithium-ion battery pack design: chemistry, components, types and terminology*. Elsevier, 2015.
- [44] 'Arbin instruments mitspro8.0 users manual,' English, version 8, March. 2019.

- [45] B. Arcus, *Tesla model 3 amp; chevy bolt battery packs examined*, Sep. 2019. [Online]. Available: <https://cleantechnica.com/2018/07/08/tesla-model-3-chevy-bolt-battery-packs-examined/>.
- [46] S.-M. Bak, Z. Shadike, R. Lin, X. Yu and X.-Q. Yang, ‘In situ/operando synchrotron-based x-ray techniques for lithium-ion battery research,’ *NPG Asia Materials*, vol. 10, no. 7, pp. 563–580, 2018.
- [47] W.-Y. Chang, ‘The state of charge estimating methods for battery: A review,’ *International Scholarly Research Notices*, vol. 2013, 2013.
- [48] C. Weng, J. Sun and H. Peng, ‘A unified open-circuit-voltage model of lithium-ion batteries for state-of-charge estimation and state-of-health monitoring,’ *Journal of power Sources*, vol. 258, pp. 228–237, 2014.
- [49] M. Coleman, C. K. Lee, C. Zhu and W. G. Hurley, ‘State-of-charge determination from emf voltage estimation: Using impedance, terminal voltage, and current for lead-acid and lithium-ion batteries,’ *IEEE Transactions on industrial electronics*, vol. 54, no. 5, pp. 2550–2557, 2007.
- [50] W. Waag, C. Fleischer and D. U. Sauer, ‘Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles,’ *Journal of Power Sources*, vol. 258, pp. 321–339, 2014.
- [51] S. Lee, J. Kim, J. Lee and B. H. Cho, ‘State-of-charge and capacity estimation of lithium-ion battery using a new open-circuit voltage versus state-of-charge,’ *Journal of power sources*, vol. 185, no. 2, pp. 1367–1373, 2008.
- [52] J. Xu, C. C. Mi, B. Cao, J. Deng, Z. Chen and S. Li, ‘The state of charge estimation of lithium-ion batteries based on a proportional-integral observer,’ *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1614–1621, 2013.
- [53] F. Feng, R. Lu and C. Zhu, ‘A combined state of charge estimation method for lithium-ion batteries used in a wide ambient temperature range,’ *Energies*, vol. 7, no. 5, pp. 3004–3032, 2014.
- [54] J. Wang, B. Cao, Q. Chen and F. Wang, ‘Combined state of charge estimator for electric vehicle battery pack,’ *Control Engineering Practice*, vol. 15, no. 12, pp. 1569–1576, 2007.
- [55] K. S. Ng, C.-S. Moo, Y.-P. Chen and Y.-C. Hsieh, ‘Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries,’ *Applied energy*, vol. 86, no. 9, pp. 1506–1511, 2009.
- [56] K.-Y. Oh, N. A. Samad, Y. Kim, J. B. Siegel, A. G. Stefanopoulou and B. I. Epureanu, ‘A novel phenomenological multi-physics model of li-ion battery cells,’ *Journal of Power Sources*, vol. 326, pp. 447–458, 2016.
- [57] R. Zhang, B. Xia, B. Li, L. Cao, Y. Lai, W. Zheng, H. Wang and W. Wang, ‘State of the art of lithium-ion battery soc estimation for electrical vehicles,’ *Energies*, vol. 11, no. 7, p. 1820, 2018.

- [58] C. Fleischer, W. Waag, H.-M. Heyn and D. U. Sauer, 'On-line adaptive battery impedance parameter and state estimation considering physical principles in reduced order equivalent circuit battery models part 2. parameter and state estimation,' *Journal of Power Sources*, vol. 262, pp. 457–482, 2014.
- [59] R. Klein, N. A. Chaturvedi, J. Christensen, J. Ahmed, R. Findeisen and A. Kojic, 'Electrochemical model based observer design for a lithium-ion battery,' *IEEE Transactions on Control Systems Technology*, vol. 21, no. 2, pp. 289–301, 2012.
- [60] X. Hu, S. Li and H. Peng, 'A comparative study of equivalent circuit models for li-ion batteries,' *Journal of Power Sources*, vol. 198, pp. 359–367, 2012.
- [61] S. Yuan, H. Wu and C. Yin, 'State of charge estimation using the extended kalman filter for battery management systems based on the arx battery model,' *Energies*, vol. 6, no. 1, pp. 444–470, 2013.
- [62] Q. Zhu, M. Xu, W. Liu and M. Zheng, 'A state of charge estimation method for lithium-ion batteries based on fractional order adaptive extended kalman filter,' *Energy*, vol. 187, p. 115 880, 2019.
- [63] S. Sepasi, R. Ghorbani and B. Y. Liaw, 'A novel on-board state-of-charge estimation method for aged li-ion batteries based on model adaptive extended kalman filter,' *Journal of Power Sources*, vol. 245, pp. 337–344, 2014.
- [64] M. Zeng, P. Zhang, Y. Yang, C. Xie and Y. Shi, 'Soc and soh joint estimation of the power batteries based on fuzzy unscented kalman filtering algorithm,' *Energies*, vol. 12, no. 16, p. 3122, 2019.
- [65] G. L. Plett, 'Extended kalman filtering for battery management systems of lipb-based hev battery packs: Part 3. state and parameter estimation,' *Journal of Power sources*, vol. 134, no. 2, pp. 277–292, 2004.
- [66] Z. Chen, Y. Fu and C. C. Mi, 'State of charge estimation of lithium-ion batteries in electric drive vehicles using extended kalman filtering,' *IEEE Transactions on Vehicular Technology*, vol. 62, no. 3, pp. 1020–1030, 2012.
- [67] E. Chemali, P. J. Kollmeyer, M. Preindl and A. Emadi, 'State-of-charge estimation of li-ion batteries using deep neural networks: A machine learning approach,' *Journal of Power Sources*, vol. 400, pp. 242–255, 2018.
- [68] M. A. Hannan, M. S. H. Lipu, A. Hussain, M. H. Saad and A. Ayob, 'Neural network approach for estimating state of charge of lithium-ion battery using backtracking search algorithm,' *Ieee Access*, vol. 6, pp. 10 069–10 079, 2018.
- [69] H. He, R. Xiong, X. Zhang, F. Sun and J. Fan, 'State-of-charge estimation of the lithium-ion battery using an adaptive extended kalman filter based on an improved thevenin model,' *IEEE Transactions on vehicular technology*, vol. 60, no. 4, pp. 1461–1469, 2011.
- [70] J. Brownlee, *What is the difference between a parameter and a hyperparameter*, 2018.
- [71] C. Vidal, P. Malysz, P. Kollmeyer and A. Emadi, 'Machine learning applied to electrified vehicle battery state of charge and state of health estimation: State-of-the-art,' *IEEE Access*, vol. 8, pp. 52 796–52 814, 2020.

- [72] M. van Gerven, *Modeling human brain function with artificial neural networks*. [Online]. Available: <https://natmeg.se/onewebmedia/Marcel%20van%20Gerven%20-%20Machine%20learning%20-%20Karolinska.pdf>.
- [73] S. Sharma, *Epoch vs batch size vs iterations*, Mar. 2019. [Online]. Available: <https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9>.
- [74] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, ‘Dropout: A simple way to prevent neural networks from overfitting,’ *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [75] G. Dreyfus, *Neural networks: methodology and applications*. Springer Science & Business Media, 2005.
- [76] I. V. Tetko, D. J. Livingstone and A. I. Luik, ‘Neural network studies. 1. comparison of overfitting and overtraining,’ *Journal of chemical information and computer sciences*, vol. 35, no. 5, pp. 826–833, 1995.
- [77] S. Learn, *Cross-validation: Evaluating estimator performance*, 2017.
- [78] G. Ian and C. Aaron, *Deep learning (adaptive computation and machine learning series)*, 2016.
- [79] V. Nair and G. E. Hinton, ‘Rectified linear units improve restricted boltzmann machines,’ in *ICML*, 2010.
- [80] J. Brownlee, ‘A gentle introduction to the rectified linear unit (relu),’ *Machine Learning Mastery*. <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks>, 2019.
- [81] A. L. Maas, A. Y. Hannun and A. Y. Ng, ‘Rectifier nonlinearities improve neural network acoustic models,’ in *Proc. icml*, Citeseer, vol. 30, 2013, p. 3.
- [82] C. McDonald, ‘Machine learning fundamentals (i): Cost functions and gradient descent,’ *Towards Data Science*, vol. 27, 2017.
- [83] A. Shrestha and A. Mahmood, ‘Review of deep learning algorithms and architectures,’ *IEEE Access*, vol. 7, pp. 53 040–53 065, 2019.
- [84] C. G. Broyden, ‘The convergence of a class of double-rank minimization algorithms: 2. the new algorithm,’ *IMA journal of applied mathematics*, vol. 6, no. 3, pp. 222–231, 1970.
- [85] K. Levenberg, ‘A method for the solution of certain non-linear problems in least squares,’ *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.
- [86] D. W. Marquardt, ‘An algorithm for least-squares estimation of nonlinear parameters,’ *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.
- [87] *Feedforward neural networks*, Brilliant, 2020. [Online]. Available: <https://brilliant.org/wiki/feedforward-neural-networks/>.
- [88] K. Grzegorzcyk, M. Kurdziel and P. I. Wójcik, ‘Effects of sparse initialization in deep belief networks,’ *Computer Science*, vol. 16, 2015.

- [89] S. YADAV, 'Weight initialization techniques in neural networks,' *Towards Data Science*, 2018.
- [90] cs231n, 2020. [Online]. Available: <https://cs231n.github.io/neural-networks-2/>.
- [91] M. A. Hannan, M. H. Lipu, A. Hussain, P. J. Ker, T. Mahlia, M. Mansor, A. Ayob, M. H. Saad and Z. Dong, 'Toward enhanced state of charge estimation of lithium-ion batteries using optimized machine learning techniques,' *Scientific reports*, vol. 10, no. 1, pp. 1–15, 2020.
- [92] L. Kang, X. Zhao and J. Ma, 'A new neural network model for the state-of-charge estimation in the battery degradation process,' *Applied Energy*, vol. 121, pp. 20–27, 2014.
- [93] C. Li, F. Xiao and Y. Fan, 'An approach to state of charge estimation of lithium-ion batteries based on recurrent neural networks with gated recurrent unit,' *Energies*, vol. 12, no. 9, p. 1592, 2019.
- [94] C. Vidal, M. Haußmann, D. Barroso, P. M. Shamsabadi, A. Biswas, E. Chemali, R. Ahmed and A. Emadi, 'Hybrid energy storage system state-of-charge estimation using artificial neural network for micro-hybrid applications,' in *2018 IEEE Transportation Electrification Conference and Expo (ITEC)*, IEEE, 2018, pp. 1075–1081.
- [95] Z. Hou, P. Xie, J. Hou *et al.*, 'The state of charge estimation of power lithium battery based on rbf neural network optimized by particle swarm optimization,' *Journal of Applied Science and Engineering*, vol. 20, no. 4, pp. 483–490, 2017.
- [96] R. Pascanu, T. Mikolov and Y. Bengio, 'On the difficulty of training recurrent neural networks,' in *International conference on machine learning*, 2013, pp. 1310–1318.
- [97] M. Nguyen, 'Illustrated guide to lstm's and gru's: A step by step explanation,' *Towards data*, 2018.
- [98] R. Zemouri and J. M. Faure, 'Comparative study between the timed automata and the recurrent radial basis function for discrete event system diagnosis,' *IFAC Proceedings Volumes*, vol. 39, no. 13, pp. 1455–1460, 2006.
- [99] H. Faris, I. Aljarah and S. Mirjalili, 'Evolving radial basis function networks using moth-flame optimizer,' in *Handbook of neural computation*, Elsevier, 2017, pp. 537–550.
- [100] H. Yu, T. Xie, S. Paszchynski and B. M. Wilamowski, 'Advantages of radial basis function networks for dynamic system design,' *IEEE Transactions on Industrial Electronics*, vol. 58, no. 12, pp. 5438–5450, 2011.
- [101] K. -L. Du and M. Swamy, 'Radial basis function networks,' *Neural networks in a soft-computing framework*, pp. 251–294, 2006.
- [102] J. Bergstra and Y. Bengio, 'Random search for hyper-parameter optimization.,' *Journal of machine learning research*, vol. 13, no. 2, 2012.
- [103] K. Team, *Keras documentation: Transfer learning amp; fine-tuning*. [Online]. Available: https://keras.io/guides/transfer_learning/.
- [104] S. J. Pan and Q. Yang, 'A survey on transfer learning,' *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

- [105] R. Nicolas, ‘The different driving cycles,’ *Car-Engineer. com*, accessed Aug, vol. 31, p. 2015, 2013.
- [106] Apr. 2017. [Online]. Available: <https://www.epa.gov/emission-standards-reference-guide/all-epa-emission-standards>.
- [107] *Homepage*. [Online]. Available: <https://unece.org/>.
- [108] P. Kollmeyer, ‘Panasonic 18650pf li-ion battery data,’ *Mendeley Data*, vol. 1, 2018. DOI: 10.17632/wykht8y7tg.1.
- [109] C. Vidal, P. Kollmeyer, M. Naguib, P. Malysz, O. Gross and A. Emadi, ‘Robust xev battery state-of-charge estimator design using a feedforward deep neural network,’ *SAE International Journal of Advances and Current Practices in Mobility*, vol. 2, no. 2020-01-1181, pp. 2872–2880, 2020.
- [110] A. Barbosa de Lima, M. B. Salles and J. R. Cardoso, ‘State-of-charge estimation of a li-ion battery using deep forward neural networks,’ *arXiv e-prints*, arXiv–2009, 2020.
- [111] E. Chemali, P. J. Kollmeyer, M. Preindl, R. Ahmed and A. Emadi, ‘Long short-term memory networks for accurate state-of-charge estimation of li-ion batteries,’ *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6730–6739, 2017.
- [112] C. Vidal, P. Kollmeyer, E. Chemali and A. Emadi, ‘Li-ion battery state of charge estimation using long short-term memory recurrent neural network with transfer learning,’ in *2019 IEEE Transportation Electrification Conference and Expo (ITEC)*, 2019, pp. 1–6. DOI: 10.1109/ITEC.2019.8790543.
- [113] X. Chen, W. Shen, M. Dai, Z. Cao, J. Jin and A. Kapoor, ‘Robust adaptive sliding-mode observer using rbf neural network for lithium-ion battery state of charge estimation in electric vehicles,’ *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 1936–1947, 2015.
- [114] H. Lipu, M. Hannan, A. Hussain, A. Ayob, M. H. Saad and K. M. Muttaqi, ‘State of charge estimation in lithium-ion batteries: A neural network optimization approach,’ *Electronics*, vol. 9, no. 9, p. 1546, 2020.
- [115] P. Kollmeyer, C. Vidal, M. Naguib and M. Skells, ‘Lg 18650hg2 li-ion battery data,’ *Mendeley Data*, vol. 2, 2020. DOI: 10.17632/b5mj79w5w9.2.
- [116] D. Corbin, *Too much or not enough: Sample sizes and statistical analysis*, 2020. [Online]. Available: <https://blog.minitab.com/blog/too-much-or-not-enough-sample-sizes-and-statistical-analysis>.
- [117] F. Pedregosa, G. Varoquaux, A. Gramfort, B. Michel V. and Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, ‘Scikit-learn: Machine learning in Python,’ *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [118] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009, ISBN: 1441412697.
- [119] Y. Zhang and C.-Y. Wang, ‘Cycle-life characterization of automotive lithium-ion batteries with linio2 cathode,’ *Journal of the Electrochemical Society*, vol. 156, no. 7, A527, 2009.

- [120] T. Yoshida, M. Takahashi, S. Morikawa, C. Ihara, H. Katsukawa, T. Shiratsuchi and J.-i. Yamaki, 'Degradation mechanism and life prediction of lithium-ion batteries,' *Journal of The Electrochemical Society*, vol. 153, no. 3, A576, 2006.
- [121] *Share and discover datasets*. [Online]. Available: <https://data.mendeley.com/>.
- [122] P. Kollmeyer, 'Turnigy graphene 5000mah 65c li-ion battery data,' *Mendeley Data*, vol. 1, 2020. DOI: [10.17632/4fx8cjprxm.1](https://doi.org/10.17632/4fx8cjprxm.1).
- [123] *100% electric edumper - 123t*, Sep. 2020. [Online]. Available: <https://lithiumsystem.ch/projects/edumper/?lang=en>.
- [124] *Bu-808c: Coulombic and energy efficiency with the battery*. [Online]. Available: https://batteryuniversity.com/learn/article/bu_808c_coulombic_and_energy_efficiency_with_the_battery.
- [125] F. Feng, X. Hu, L. Hu, F. Hu, Y. Li and L. Zhang, 'Propagation mechanisms and diagnosis of parameter inconsistency within li-ion battery packs,' *Renewable and Sustainable Energy Reviews*, vol. 112, pp. 102–113, 2019.
- [126] E. Keany, *BorutaShap : A wrapper feature selection method which combines the Boruta feature selection algorithm with Shapley values*. Version 1.1, Nov. 2020. DOI: [10.5281/zenodo.4247618](https://doi.org/10.5281/zenodo.4247618). [Online]. Available: <https://doi.org/10.5281/zenodo.4247618>.
- [127] F. Chollet *et al.* (2015). 'Keras,' [Online]. Available: <https://github.com/fchollet/keras>.
- [128] D. P. Kingma and J. Ba, 'Adam: A method for stochastic optimization,' *arXiv preprint arXiv:1412.6980*, 2014.
- [129] B. Roy, *All about feature scaling*, Apr. 2020. [Online]. Available: <https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35>.
- [130] *6.3. preprocessing data*. [Online]. Available: <https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-scaler>.
- [131] J. Brownlee, *Hyperparameter optimization with random search and grid search*, Sep. 2020. [Online]. Available: <https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/>.
- [132] *Tuners*. [Online]. Available: <https://keras-team.github.io/keras-tuner/documentation/tuners/#randomsearch-class>.
- [133] T. Akiba, S. Sano, T. Yanase, T. Ohta and M. Koyama, 'Optuna: A next-generation hyperparameter optimization framework,' in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.
- [134] M. Feurer and F. Hutter, 'Hyperparameter optimization,' in *Automated Machine Learning*, Springer, Cham, 2019, pp. 3–33.
- [135] J. Rijdsdijk, L. Wu, G. Perin and S. Picek, 'Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis.,' *IACR Cryptol. ePrint Arch.*, vol. 2021, p. 71, 2021.

- [136] Y. Liu, J. Li, G. Zhang, B. Hua and N. Xiong, 'State of charge estimation of lithium-ion batteries based on temporal convolutional network and transfer learning,' *IEEE Access*, vol. 9, pp. 34177–34187, 2021. DOI: [10.1109/ACCESS.2021.3057371](https://doi.org/10.1109/ACCESS.2021.3057371).
- [137] Jun. 2021. [Online]. Available: <https://www.titanaes.com/>.
- [138] I. B. Espedal, A. Jinasena, O. S. Burheim and J. J. Lamb, 'Current trends for state-of-charge (soc) estimation in lithium-ion battery electric vehicles,' *Energies*, vol. 14, no. 11, pp. 1–24, 2021.

Paper I

I. B. Espedal, A. Jinasena, O. S. Burheim and J. J. Lamb, 'Current trends for state-of-charge (soc) estimation in lithium-ion battery electric vehicles,' *Energies*, vol. 14, no. 11, pp. 1–24, 2021

Review

Current Trends for State-of-Charge (SoC) Estimation in Lithium-Ion Battery Electric Vehicles

Ingvild B. Espedal ¹, Asanthi Jinasena ¹, Odne S. Burheim ¹ and Jacob J. Lamb ^{1,2,*}

¹ Department of Energy and Process Engineering & ENERSENSE, NTNU, 7491 Trondheim, Norway; ingvibes@stud.ntnu.no (I.B.E.); asanthi.jinasena@ntnu.no (A.J.); odne.s.burheim@ntnu.no (O.S.B.)

² Department of Electronic Systems & ENERSENSE, NTNU, 7491 Trondheim, Norway

* Correspondence: jacob.j.lamb@ntnu.no

Abstract: Energy storage systems (ESSs) are critically important for the future of electric vehicles. Despite this, the safety and management of ESSs require improvement. Battery management systems (BMSs) are vital components in ESS systems for Lithium-ion batteries (LIBs). One parameter that is included in the BMS is the state-of-charge (SoC) of the battery. SoC has become an active research area in recent years for battery electric vehicle (BEV) LIBs, yet there are some challenges: the LIB configuration is nonlinear, making it hard to model correctly; it is difficult to assess internal environments of a LIB (and this can be different in laboratory conditions compared to real-world conditions); and these discrepancies can lead to raising the instability of the LIB. Therefore, further advancement is required in order to have higher accuracy in SoC estimation in BEV LIBs. SoC estimation is a key BMS feature, and precise modeling and state estimation will improve stable operation. This review discusses current methods use in BEV LIB SoC modelling and estimation. The review culminates in a brief discussion of challenges in BEV LIB SoC prediction analysis.



Citation: Espedal, I.B.; Jinasena, A.; Burheim, O.S.; Lamb, J.J. Current Trends for State-of-Charge (SoC) Estimation in Lithium-Ion Battery Electric Vehicles. *Energies* **2021**, *14*, 3284. <https://doi.org/10.3390/en14113284>

Academic Editor: Woojin Choi

Received: 14 April 2021

Accepted: 1 June 2021

Published: 4 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: lithium-ion battery; state-of-charge; modelling; battery management systems

1. Introduction

Energy storage systems (ESSs) are important technologies for future electric cars and smart grids [1–4]. Lithium-ion batteries (LIBs) are the fastest growing ESS technology [5]. Despite this, the safety and management of LIBs are vital areas that still require further development [6]. Therefore, LIB management systems (BMSs) are critical for the electrification of battery electric vehicles (BEVs) and encompass a variety of features to ensure optimal operation (Figure 1).

The developing advanced and smart state-of-charge (SoC) estimators for LIBs have become an active research topic in recent years. The key technological challenges limiting the advancement of SoC can be gathered into three aspects. The first is that the LIB structure is nonlinear, making it challenging to model accurately. This is due to the multi-scale nature (e.g., active materials, cells, and battery packs are all in different spatial scales), and time scale aspects (e.g., aging). Second, the internal environment is difficult to determine and is susceptible to fluctuations of the external environment. Scaling up LIBs from laboratory- to industrial-level production decreases the correlation between calculated values and actual values, rendering it difficult to reliably determine the battery's internal states. Finally, the LIBs discrepancies directly affect the performance of the LIB pack, raising the instability of the LIB. Estimation measures designed for smaller LIBs are redundant on large-scale LIBs (i.e., BEV LIBs), and reliable and precise LIB SoC estimation is difficult. Therefore, advanced SoC methods are urgently required to overcome these challenges [7,8].

Battery state estimation is a key advanced BMS feature in BEVs. Precise modeling and state estimation will allow stable operation, facilitate optimal battery operation, and provide the fundamentals for security supervision [9]. This review discusses BEV LIB

SoC modelling, estimation and methods. The review culminates in a brief discussion of challenges in BEV LIB SoC prediction analysis.

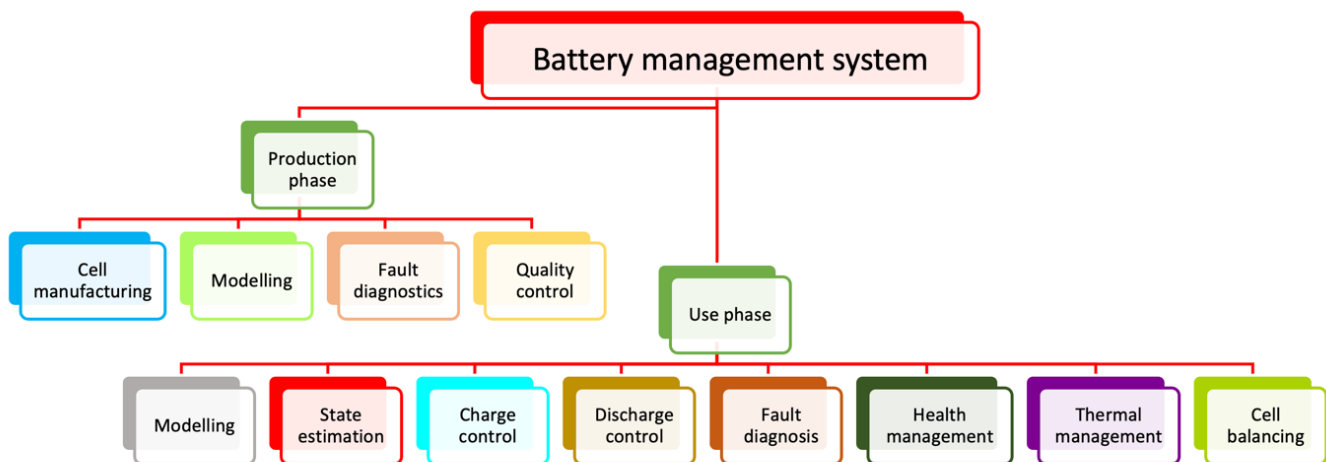


Figure 1. Battery management system (BMS) functional features for battery electric vehicle lithium-ion batteries.

2. Overview of BEV LIB SoC Modelling Approaches

The battery models are useful in model-based SoC estimation and can be characterized as physical electrochemical models [10,11], electrical equivalent circuit models [12,13] and data-driven models [14,15], with the two latter being routinely used in BEV SoC estimation. This section introduces these models with emphasis on data-driven models and electrical equivalent circuit models, and more information about their application and comparisons between the models is given in Section 4.

2.1. Physical Electrochemical Models

Single-particle models are the simplest established model for physics-based electrochemical analysis [16]. Here, a single particle reflects the Li-ion concentration distribution in the electrode. It can be used to analyze primary output and electrode solid-phase diffusion effect; however, the precision is poor. To improve the precision, a model has been developed that considers the electrolyte's effect on the output potential, suggesting a partial differential equation to conserve the liquid electrolyte material [17].

A pseudo-two-dimensional model has been developed that considers that the cell anode and cathode as porous consisting of ball-like particles with the electrolyte filling the gaps in-between [18]. As the pseudo-two-dimensional model includes multiple coupled partial differential equations, it must be condensed from the engineering perspective [19,20].

A key intention why physics-based models are hard to implement is that a huge amount of unspecified variables are required to be defined using approaches such as global optimization. Unsurprisingly, they can face overfitting or local optimization issues. Without correct and comprehensive parameters, the open loop simulations of electrochemical models based on physics are not optimal for SoC calculations. High-resolution detailed models usually contain numerous nonlinear partial differential equations, which make the model solving complex and computationally expensive, and is not suitable for estimation online, despite the high accuracy. There are reduced-order models that are less complex and take less computational time than the full-order models and can be used in online estimation applications [21]. However, these advantages come with the cost of increased estimation errors.

2.2. Electrical Equivalent Circuit Models

Electrical equivalent circuit models have grown in popularity due to their simpler construction, allowing implementation into real-time applications. Models of electronic

counterparts use electrical components to imitate LIB behavior. These can be classified into integral-order and fractional-order models.

2.2.1. Integral-Order Models

Most of the commonly used equivalent circuit models are integral-order models. Of these, the R_{int} model is the most used integral-order model [22]. The R_{int} model structure is straightforward; however, it does not consider the polarization and diffusion dynamics. Liaw et al. [23] introduced the resistor–capacitor model, which is a first-order model capable of mimicking LIB charging and discharging behavior using one resistor–capacitor network. Additionally, the open-circuit voltage hysteresis behavior can be considered to improve model accuracy [24].

The LIBs input/output relationship is simple to infer for integral-order analogous circuit models. Therefore, the most frequently used technique for online parameter detection is the least-squares recursive algorithm. A co-estimator has also been recommended to predict model parameters and battery status [25]. This can be in the form of identification of parameters while considering the electrochemical properties [26], or a multi-objective genetic algorithm [27].

2.2.2. Fractional-Order Models

Fractional-order calculus-based models are also common in modeling due to their fractional characteristics. Electrochemical impedance spectroscopy (EIS) is an accurate method for modeling electrochemical reactions of lithium ions. However, it is quite difficult to estimate SoC using only EIS; therefore, many fractional-order models are used together with EIS for a better estimation [28]. There are some studies that use Bode plots [29] to aid the circuit models for improved estimation. However, in order to use electrical equivalent circuit models for efficient online SoC estimation, a proper parameter identification method (either online or offline) such as curve fitting, recursive least square, particle swarm, or genetic algorithm must be used.

Constant phase elements can also be utilized in resistor–capacitor networks instead of capacitors [30]. Using simplified fractional-order impedance, a genetic algorithm can be applied to classify model constraints obtaining an error of 0.5% [31]. Alternatively, non-integer model derivatives with a particle swarm optimization algorithm can be used to classify model constraints with strong accuracy and robustness [32]. Fractional-order equivalent circuit models with alternatives of Kalman filter are commonly used to estimate SoC in LIBs [33,34].

2.3. Data-Driven Models

Data-driven approaches are commonly used to design LIB models. Most used data-driven methods for online SoC estimation are based on machine learning techniques (e.g., neural networks, support vector machines, and fuzzy logics). LIB testing experiments have confirmed the successful solution proposed using a neural-based thermal-electric-coupled model [35]. It is also possible to perform data estimation by integrating the neural network with particle filtering [36]. Dynamic simulation technology is commonly used in LIB modelling, and a recent Monte Carlo 3D model has been used to show the structural evolution of solid sulfur and lithium sulfide dissolution/precipitation during lithium-sulfur battery discharge [37]. The kinetic model can estimate battery subtleties over longer scales of time. While data-driven techniques work well on nonlinear issues, they can be affected by the datasets and methods used for training. Further, the requirement of large data set to cover all the possible operating conditions, demand high overall computational costs.

3. State Estimation in BEV LIBs

A BEV LIB is a dynamic, nonlinear device of several state variables, so correct state prediction is the secret to controlling and the foundation for LIB power. This section

defines SoC and parameters routinely used in BEVs for SoC estimation and summarizes the commonly used LIB state estimation methods.

3.1. SoC in LIBs

SoC is the measure of the remaining energy in a battery (Figure 2). Physically, the amount of energy left in a battery (ψ) is defined as the average concentration of lithium-ions in the cathode ($C_{s,avg}$) over the maximum possible concentration:

$$\psi = \frac{C_{s,avg}}{C_{s,max}} \quad (1)$$

Theoretically, $\psi = 0$ and $\psi = 100$ is possible; however, it is not feasible as the removal of too many lithium-ions from the cathode will damage the structure and increase degradation. Therefore, a ψ window is defined for LiBs where $\psi_{0\%} > 0$ and $\psi_{100\%} < 1$. The SoC can now be defined by a ratio of the defined ψ windows:

$$\text{SoC} = \frac{\psi_k - \psi_{0\%}}{\psi_{100\%} - \psi_{0\%}} \quad (2)$$

where ψ_k is the amount of energy left in the battery at time k . This definition of SoC is theoretically correct; however, it is not practically possible for a BMS to calculate ψ as the concentration of lithium-ions cannot be measured directly. As a consequence, a definition of SoC that is not directly based on the lithium-ion concentration is required. SoC can therefore be explained as the LIBs residual power ratio (C_r) to maximum available capacity (C_a) (Equation (3)) [38,39]. C_r is the residual energy that can be drained from the LIB over time. C_r is affected by cycling and deterioration of the LIBs electrochemical aspects. C_a is the highest possible load capacity that can be collected during the initial cycle period under diverse conditions. C_a is unable to reach the rated power (C_{rated}) (i.e., the manufacturer's LIB capacity for regular operation). The importance of a LIBs C_a relies on the state-of-health (SoH) and current discharge volume. The SoC is expressed in percentage between 100% (fully charged) and 0% (fully discharged; Figure 2).

$$\text{SoC}(t) = \left(\frac{C_r}{C_a} \right) \cdot 100\% \quad (3)$$

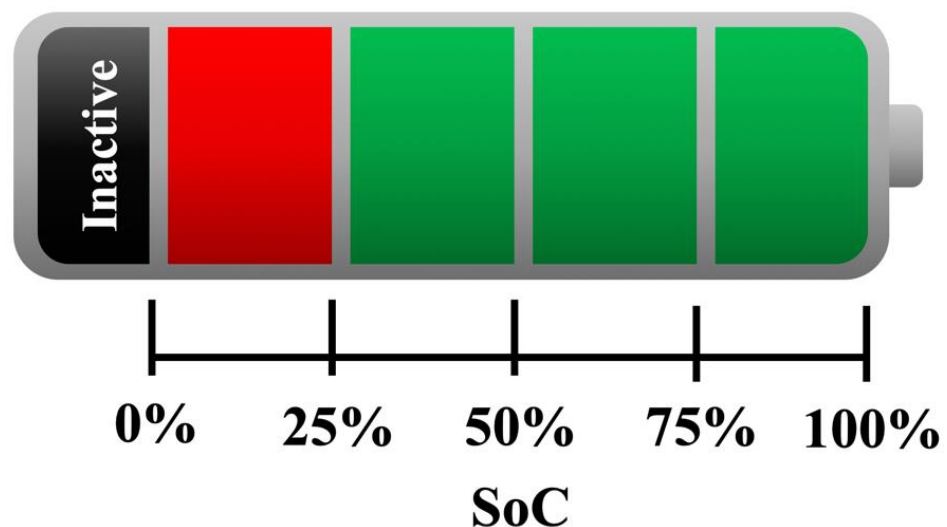


Figure 2. Stored energy status of a lithium-ion battery related to state-of-charge.

3.1.1. Coulomb Counting

Coulomb counting is a simple classical SoC estimation method. Coulomb counting is utilized to find the connection between the SoC and the cycling of the LIB. The SoC(t) can usually be determined through this method (Equation (4)), where the coulombic efficiency is denoted η_i , the initial SoC is denoted SoC(0), and the battery charging/discharging current is denoted as $I(t)$. Commonly, η_i represents the ratio between the consumed and available electrons, while cycling the LIB (a ratio of 0.9 during charging, and 1 during discharging) [40]. Despite this, the coulombic charging efficiency ranges from 0.9 to 1 with changes in the conditions of operation (e.g., the current rate and temperature) [41].

$$\text{SoC}(t) = \text{SoC}(0) + \int_0^t \frac{\eta_i I(t)}{C_a} \cdot \Delta t \quad (4)$$

Large errors can occur in this method due to accumulation of terminal measurements, which require frequent calibrations [42]. Despite the easy implementation, error accumulation and the initial SoC value requirement [43] make this approach not ideal for online SoC estimation.

3.1.2. Open Circuit Voltage

The open circuit voltage (OCV) is a significant value that is routinely used to characterize electrochemical cells (i.e., batteries). Despite this, there is no common description for it [44]. In general, it is described as the voltage determined by a potentiometer between the cathodic and anodic poles of the electrochemical cell when the cell has reached thermodynamic equilibrium [45]. As the status of whether the cell is in equilibrium is not straightforwardly known, this definition can be unclear, and the general approach is to leave the cell uninterrupted for significant time to achieve thermodynamic equilibrium. However, the cell is often not left to rest to full thermodynamic equilibrium during use while calculating the OCV [45]. In practice, the OCV is calculated from the cell potential once a plateau occurs in the cell potential over time.

Alternatively, the OCV can be calculated from the activity of the electrochemically active components within the cell. This can allow the OCV dependency on behavior to be substituted with their molarities or concentrations. This approach requires the use of activity coefficients for the active components within the cell, which are generally found by simplified assumptions of ideal active components due to the lack of readily available data from a functioning cell. This simplification may in some cases lead to OCV estimation errors [46]. In terms of LIBs in electric vehicles, the only parameters measured are generally the current, voltage and temperature of the cell; therefore, the method used for OCV determination is based on the former method requiring thermodynamic equilibrium.

Using the OCV is highly precise and simple method for SoC estimation. The connection between SoC and OCV is extracted from the stepwise calculation of OCV for various SoC values in a LIB. An example of this connection is given in Figure 3. Despite this, the OCV and SoC relationship for each LIB varies over the battery life and the related hysteresis effect, while they are of the same rating, content, and structure [47,48]. The value of OCVs is not the same when charging and discharging due to hysteresis characteristics, despite having the same SoC value [49]. The main drawback is it takes time to reach thermodynamic equilibrium conditions, which makes it only applicable when the vehicle is at rest, instead of in driving mode. It has been shown that it is possible to estimate the OCV of a redox flow battery without waiting to reach equilibrium conditions [44]; however, this has not yet been shown in LIBs. In previous years, the literature has suggested numerous updated OCV-based methods to improve processing time and accuracy [50,51]. However, this approach is inadequate for SoC estimation due to its strong reliance on the OCV, the requirement for equilibrium conditions and precision calculation of charging/discharging voltages [47].

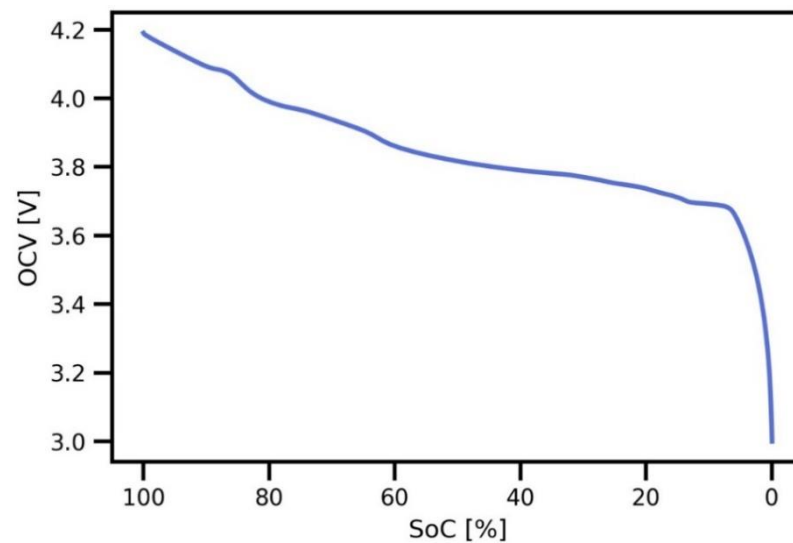


Figure 3. The relationship of the open circuit voltage with the state-of-charge. The OCV of a Lithium cobalt oxide (LCO) battery produced by Melasta Battery (Model No. SLPBB042126HN) with a capacity of 6550 mAh. The battery was cycled in an Arbin Instruments cyler at 25 °C. The OCV was estimated from cycling at a c-rate of $c/20$, and the SoC was estimated using the coulomb counting method.

3.1.3. State of Health

Generally, when calculating the SoC, the rated, real, or cycle capacity is used for C_a (Equation (4)) [52–54]. Therefore, defining the established battery capacity is not consistent [55]. In the theoretical analysis, capacity is used as the equation denominator, treating C_a as a fixed number, and the SoC is calculated by deducting the charge sum from C_a [56]. An important parameter to consider then is the SoH, which is the ratio of maximum residual capacity at the time (C_m) to the maximum available capacity of the battery when new (Equation (5)).

$$\text{SoH}(t) = \left(\frac{C_m}{C_a} \right) \cdot 100\% \quad (5)$$

SoH allows the inclusion of BEV LIB ageing and is affected by different causes (e.g., driving style and ambient temperature). Owing to the effect of various LIB fluctuations (e.g., temperature), the inconsistencies of battery voltage, internal resistance, and power can increase to some extent, influencing the value of SoH. Therefore, it is vital to consider the relationship between SoH, SoC, and depth-of-discharge (DoD).

$$\text{SoC}(t) = \text{SoH}(t) - \text{DoD}(t) \quad (6)$$

Here, DoD is expressed as the percentage of capacity that can be discharged relative to the available capacity (i.e., $\left(\frac{C_m - C_r}{C_a} \right) \cdot 100\%$).

3.2. Estimation of Energy and Remaining Capacity

Physically, a LIB discharge transfers Li from the anode to the cathode, with the opposite occurring during charging. Electrochemically, the SoC is positively associated to the Li concentration in the anode and negatively to the Li concentration in the cathode. The cell potential depends on the concentration of the electrode surface, where SoC depends on average concentration of particles, making SoC estimation using the potential challenging. Another aspect is determining the amount energy in a LIB pack. This is essential for battery electric vehicles, as the calculated vehicle range is entirely dependent on this. Therefore, the state-of-energy (SoE; residual energy) calculation is vital.

3.2.1. Look-Up Tables

Look-up tables are useful and forthright, allowing SoC estimation through mapping the connection between characteristic parameters of the LIB and the SoC [57]. The primary drawback of look-up tables is that the LIB must be resting for a long time for reliability of the internal electrochemistry, so that measurements can be precise. SoC calculation efficiency also relies dramatically on the precision of the table used. Therefore, the approach is not suitable for online and precise estimation of SoC.

Open-Circuit Voltage

The OCV is the LIB potential after a long, load-free rest, and has a nonlinear relationship with SoC [58,59]. Comprehensive methods have been presented where the OCV is calculated for various SoC levels producing extensive OCV-SoC tables [58,59]. The process is straightforward; however, the LIB must rest for significant time to ensure accurate OCV measurement. Considering the effect of temperature, substance, driving style, and aging on the OCV-SoC relationship, these considerations must be applied to the OCV-SoC table [60]. An OCV-SoC-temperature table has been developed to infer SoC battery [61]. The hysteresis effect is also an important factor impacting the accuracy [62–64]. An incorrect OCV-SoC correlation deprived of understanding of the hysteresis effect will lead to incorrect SoC estimations.

Impedance

A relationship concerning impedance and SoC also exists. By adding a current frequency to the battery, the nonlinear fitting or parameter recognition algorithm can determine multiple SoC-related parameters (e.g., internal ohmic resistance), and then an impedance look-up table method can be used [65,66]. However, battery ageing can influence the precision of the SoC look-up table process and can cause substantial predictive errors [67]. Moreover, current and ambient temperature can contribute to non-linear impedance and SoC changes [68].

3.2.2. Ampere-Hour Integral

Ampere-hour integral is a simpler method where current integration is used for SoC estimation [69]; however, this is at the cost of some clear disadvantages. Sensor error can occur due to the open-loop estimation, which will produce large errors in the SoC estimation. Additionally, deviations in Coulomb performance can materialize from temperature and aging, also affecting the accuracy of SoC estimation. Furthermore, the initial SoC is estimated using look-up tables, so preliminary errors will propagate through the entire estimation process. This approach is typically paired with model-based or data-driven methods to improve its sturdiness.

3.2.3. Filter-Based

The model-based estimators can usually be loosely clustered into Bayesian estimators (filters), deterministic observers, and learning algorithms. The compared performances of these estimation methods in this section mainly depend on the individual models that are used with the estimation method rather than the standalone estimation method itself.

Most common Bayesian estimators are the variants of Kalman filters. Recently, a number of alternatives are used to estimate SoC in LIBs with Kalman filters (Figure 4).

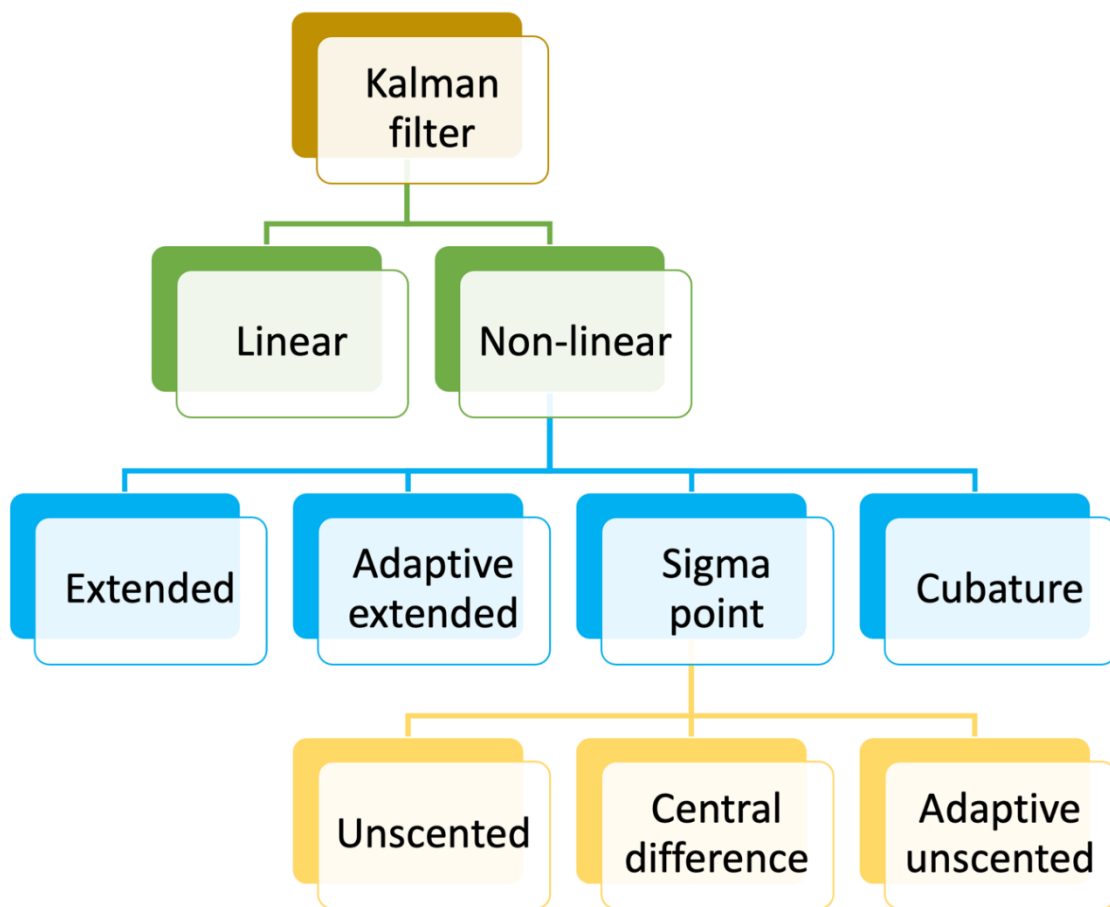


Figure 4. The Kalman filter family of algorithms that have been used for state-of-charge estimation in battery electric vehicle Lithium-ion batteries.

Linear Kalman Filter

The Kalman filter has been commonly used for BEV LIB SoC estimation [70]. The general estimation algorithm can be regarded as a recursive mechanism involving first the estimating the system states, and second updating the system states based on the feedback errors [71]. Linear Kalman filter is optimal for linear systems. As the OCV is nonlinear, Kalman filters cannot be used explicitly for OCV-based estimations [72]. The OCV function can be linearized with local linearization, optimizing Kalman filters for SoC estimation [73].

Extended Kalman Filter

The theory of the extended Kalman filter is to linearize the nonlinear system at each time step [74]. When used with OCV-based models, it extends the nonlinear OCV function with partial derivatives based on the nonlinear function linearization principle [75]. As model parameters are easily modified due to nonlinear behavior, an SoC estimation approach based on a reduced-order LIB model and an extended Kalman filter has been proposed with errors below 2% [76]. Alternatively, SoC estimation of a LIB based on a dual-time scale extended Kalman filter has been developed, where the average SoC was measured for all cells, then the individual cell SoC was calculated by the discrepancy between the mean and each individual cell [77]. This yielded an SoC error below 2%. Overall, the extended Kalman filter's accuracy depends not only on OCV function linearization, but on the model parameters also [78]; as a consequence, improved approaches can be classified into either model improvement or algorithm improvement [10].

Adaptive Extended Kalman Filter

When the parameters or the noise covariances of any Kalman filter are adapted according to the measurements, it is called an adaptive Kalman filter. He et al. used an adaptive extended Kalman filter for estimating SoC, where the process and measurement noise covariances are adapted [79]. This helps the adaptive extended Kalman filter effectively prevent algorithm divergence or bias [80,81]. Based on a closed-loop feedback system with multiparameter input, an adaptive extended Kalman filter calculated the correct SoC estimate with a median SoC error of 3% [82]; however, this did not take LIB aging into account. To upgrade the battery-aging model, a basic optimization algorithm can be applied, and an adaptive extended Kalman filter can calculate the SoC in aging LIBs with an error of less than 4% [83]. As the fractional-order model provides a clearer explanation of LIB behavior, the adaptive extended Kalman filter estimates the SoC using the fractional-order model [84].

Sigma-Point Kalman Filter

A sigma-point Kalman filter supplements the extended Kalman filter where the nonlinear component is overlooked. Therefore, sigma-point Kalman filter SoC estimation has been suggested [85], and an electrochemical model-based SoC estimation approach was proposed using an adaptive square root sigma point Kalman filter with equality constraints observing a 30% improvement compared to an adaptive extended Kalman filter [86].

Unscented Kalman Filter

The unscented Kalman filter is a nonlinear estimator and is developed from traceless transformation. Nonlinear machine formulas can be extended to regular Kalman filter by unscented transformation [87]. The unscented Kalman filters transformation does not neglect higher-order terms, but it has high precision estimates [88]. To improve conventional unscented Kalman filter performance, several improved algorithms have been produced [89]. To minimize computational requirements for unscented transformation in the unscented Kalman filter, the square root unscented Kalman spherical transform filter was developed to estimate battery SoC [90]; however, the maximum error increased by 37% compared to extended Kalman filters. The Fuzzy Inference System was used to improve the robustness and simplify the unscented Kalman filter [91], where the SoC error was reported to be below 1.76%.

Adaptive Unscented Kalman Filter

Sun et al. have estimated SoC of a LIB using an adaptive unscented Kalman filters where the noise covariance is auto tuned [92]. A recursive least-square approach defined on-line model parameters and modified the adaptive unscented Kalman filter state model in real-time, minimizing the SoC estimate error [93]. When comparing adaptive unscented, adaptive extended, unscented, and extended Kalman filters performance in SoC estimates based on a second order resistor-capacitor model, the adaptive unscented Kalman filters achieved the best results [94]. Other experimental results have also found that SoC prediction accuracy is better with adaptive unscented Kalman filters when compared to unscented Kalman filters and extended Kalman filters [95,96].

Central Difference Kalman Filter

This method uses the Sterling interpolation formula to generate the Sigma points, which then finds the posterior distribution of the states [97]. It has been shown that the central different Kalman filter was efficient than the extended Kalman filter at determining the SoC [98,99].

Cubature Difference Kalman Filter

The cubature difference Kalman filter uses the spherical-radial cubature rule to calculate the multivariate moment integrals. It is centered on the radial volume criterion of the

third-order, where the cubature difference Kalman filter approximates the average value of the nonlinear state using a sequence of volume coordinates that can efficiently resolve the difficulty of nonlinear state estimation [100]. Two important steps are to turn the integral form into the spherical integral form and the spherical radial third-order criterion. It has been observed that the cubature difference Kalman filter error in SoC estimation is 1.78% lower than that of extended Kalman filter approaches [101]; however, the estimation time is significantly extended.

Particle Filter

Unlike the Kalman filter variants described earlier, the particle filter is based on probability densities, thus can handle non-Gaussian systems. When designing a particle filter, the main difficulty is to select the proper proposal distributions that can approximate the posterior distributions. The most common method is to use the transition prior for this. The particle filters central concept is to produce a series of independent random sampling points in the state space according to the system state vector's scalar distribution, then change the particle location and state to the detected values. The maximal particle state is then estimated by changing particle sets [102,103]. Results have demonstrated the particle filters are much better than extended Kalman filters in estimating the SoC [104–107].

Unscented Particle Filter

Using the transition prior as a proposal distribution in particle filters can be inefficient, if new measurements appear at the tail of the prior (when there are abrupt changes in measurement data) or if the likelihood is narrow and high compared to the prior (when sensors are highly precise). To avoid this problem, usually extended Kalman filter approximation or unscented Kalman filter approximation is used as the proposal distribution of the particle filter. When the unscented Kalman filter approximation is used, the particle filter is called an unscented particle filter. The unscented particle filter is an improved particle filter, where the key concept is to boost the particle filter sampling. Calculation results provide online observation detail, improving the particle filter sampling effect [108]. The unscented particle filter has been widely used in SoC estimation due to its superior efficiency [109,110], with SoC estimation errors below 2% at a high computational efficiency [111].

Cubature Particle Filter

Cubature particle filter is based on spherical-radial cubature rule of the third degree and generally used in high-dimensional state estimations. The third-degree cubature rule is a special form of the unscented transformation but with a higher numerical stability. However, both the unscented and cubature particle filters have limitations in both accuracy and robustness. Cubature particle filters measure the average and discrepancy of a nonlinear random function by the volume method explicitly and produces the proposed density function for weighted particles. By then measuring the particle mean, the minimum mean square error approximation is obtained [112]. The cubature particle filter algorithm uses the new measurement knowledge when producing particles, improving the degree of approximation of the later likelihood of machine state [113]. The cubature particle filter has been observed to outperform the extended particle filter in terms of errors; however, the computational efficiency is low [114]. By using an adaptive weighted cubature particle filter, the computational efficiency was improved and the error in SoC estimation was reduced to 1% [115].

3.2.4. Observer-Based

Although the observer is a common term that is used for any type of estimator including the Bayesian filters or learning algorithms, the observer mentioned in this section refers only to the deterministic observers. This includes classical Luenberger type observers, finite-dimensional observers, and combined observers. To realize state input or

other control system requirements, state observer definition and construction processes have been suggested [116]. Recently, observer-based approaches (e.g., the Luenberger observer) have been commonly used to estimate the necessary states of LIBs.

Luenberger

The Luenberger observer is used extensively in linear, nonlinear, and time-varying systems [117]. Hu et al. [118] suggested an adaptive Luenberger observer system for online LIB pack SoC estimation, where the observer gain was adapted using a stochastic gradient approach. A nonlinear fractional LIB model-based SoC estimation has also been developed using a Luenberger observer [119], where the global asymptotic stability is ensured using a direct Lyapunov method.

Sliding Mode

Sliding mode observers are mainly used in LIB estimation where disturbances and measurement noise have sufficiently large switching gains [120]. Built from the sliding mode controller, the sliding mode observer retains robust tracking efficiency under model instabilities and environmental interferences [120]. A second-order discrete-time sliding mode observer has been implemented to eradicate what is known as the chattering phenomenon [121]. Furthermore, an adaptive gain sliding mode observer was proposed to minimize chattering levels and the compensate for errors in the model [122,123]. Moreover, a Gray prediction-based fuzzy sliding mode observer, which is combined with fuzzy logic, has been suggested to both minimize chattering and overcome overestimation [124]. Here, the battery voltage value was calculated by grey estimation and the gains on the sliding mode observer were calibrated through the fuzzy inference system to forecast the error. A combined neural network-based robust sliding mode observer has also been proposed with an adaptive switching gain to mitigate chattering effects in a lithium polymer battery [122,123]. Here, a radial-based neural network is integrated to the observer to ensure the convergence of SoC estimation errors [125].

Proportional Integral

A proportional-integral observer is a combined observer of proportional and integral observers and is an effective way of calculating the states with uncertain input disturbance. A parameter-normalized proportional integral observer was devised to manage the power error and initial error in LIB measurements, and the new integrator was to confine the drifting current effect. It was observed that this approach had lower computational complexity but high precision without matrix operation, even though the original SoC was uncertain [126].

H-Infinity

The H-infinity observer will ensure robustness of the incorrect initial state and uncertain disruption when imprecise or unidentified statistical characteristics from modelling and calculating errors are present. When combining an H-infinity observer with a hysteresis model, it may be possible to accommodate model ambiguities from temperature, aging and current [127]. A dynamic gain H-infinity observer has also been developed for battery pack SoC estimation, which can decrease the adverse impact of non-Gaussian models and measurement errors [128,129].

3.2.5. Data-Driven

Data-driven approaches presume the LIB is a black box model and learn internal dynamics by vast quantities of measured data. The main data-driven estimation methods used in LIB estimation include genetic algorithms, support vector machines, and neural networks and are usually used together with another estimation or inference method. For an extensive overview of data-driven methods that can be applied to SoC estimation in LIBs, the authors suggest the recent articles by Lipu et al. [130] and Vidal et al. [131].

Genetic Algorithm

Usually, a genetic algorithm is used to define the parameters of a LIB for further SoC estimation [132]. A novel SoC estimation approach focused on a sliding mode Gray model which is combined with genetic algorithm has been developed, where the use of genetic algorithm introduced greater precision and repeatability [133].

Support Vector Machines

A support vector machine is a cluster of supervised learning methods that can use kernels for various number of learning tasks [133,134]. As support vector machines are based on the structural risk minimization principle, it can perform better than the conventional neural networks. However, the increasing modelling size and the single output structure are disadvantages of this method [132]. An improved support vector machine for regression-based SoC estimation process has been suggested, where the observed results suggested it was a straightforward and reliable method compared to artificial neural networks [135]. Alternatively, adaptive unscented Kalman filters and least-square support vector machines were integrated to approximate battery SoC, where, even with minimal training samples, the LIB model can be correctly developed and modified [136].

Artificial Neural Networks

The neural network approach (Figure 5) has outstanding potential to form a nonlinear map showing an intricate nonlinear model [137]. An OCV-based SoC calculation based on the dual neural network fusion LIB model, where the linear neural network LIB model was used to define first-order or second-order electrochemical model parameters, and the second back-propagation neural network was used to take the OCV-to-SoC relationship [138]. A further development in back-propagation neural networks resulted in a strategy that utilizes principal component analysis and particle swarm optimization allowing improved accuracy and robustness of the estimation [139].

A radial basis function neural network-based uncertainty quantification algorithm has also been developed to build a response surface approximate model to estimate a multi-cell LIB pack SoC, as used in BEVs [140]. A novel approach based on a deep feed-forward neural network was used for LIB SoC estimation, by directly mapping the measurements to the SoC [141]. By establishing a load-classifying neural network model to estimate SoC, overfitting of the model was suppressed as a result of the model structure and the postprocessing [137]. A nonlinear observer based on a radial basis function neural network has also been proposed [142], which used an inclusive equivalent circuit model.

Lipu et al. [143] established a nonlinear autoregressive with exogenous input-based neural network that was optimized for the field of SoC estimation, where the lightning search algorithm allowed the identification for the optimal input delays, feedback delays and hidden layer neurons. Recently, Hannan et al. [144] presented a recurrent nonlinear autoregressive with exogenous inputs neural network with using the lightning search algorithm to increase SoC estimation. This further developed the approach by allowing accurate SoC estimation under different LIB conditions. By using the Levenberg–Marquardt-optimized multi-hidden layer wavelet neural network model proposed by Xia et al. [145], SoC estimation was possible through particle swarm optimization.

A stacked long short-term memory recurrent neural network was established to model LIB dynamics and estimate the SoC, with the approach providing rapid estimation to the true SoC, despite the original SoC being incorrect [146]. A similar approach was recently published where transfer learning was exploited to accelerate the neural network training, with a rolling learning method developed to implement SoH influence [147]. This method yielded precise estimation under different conditions, allowing a well-trained model to be straightforwardly transferred to similar battery chemistries. Further advancements were made by Chen et al. [148], where an autoregressive long short-term memory network and moving horizon estimation were used for SoC estimation. This method was investigated to

assess whether it was a suitable approach for SoC estimation when there is uncertainty or large deviations from the initial SoC.

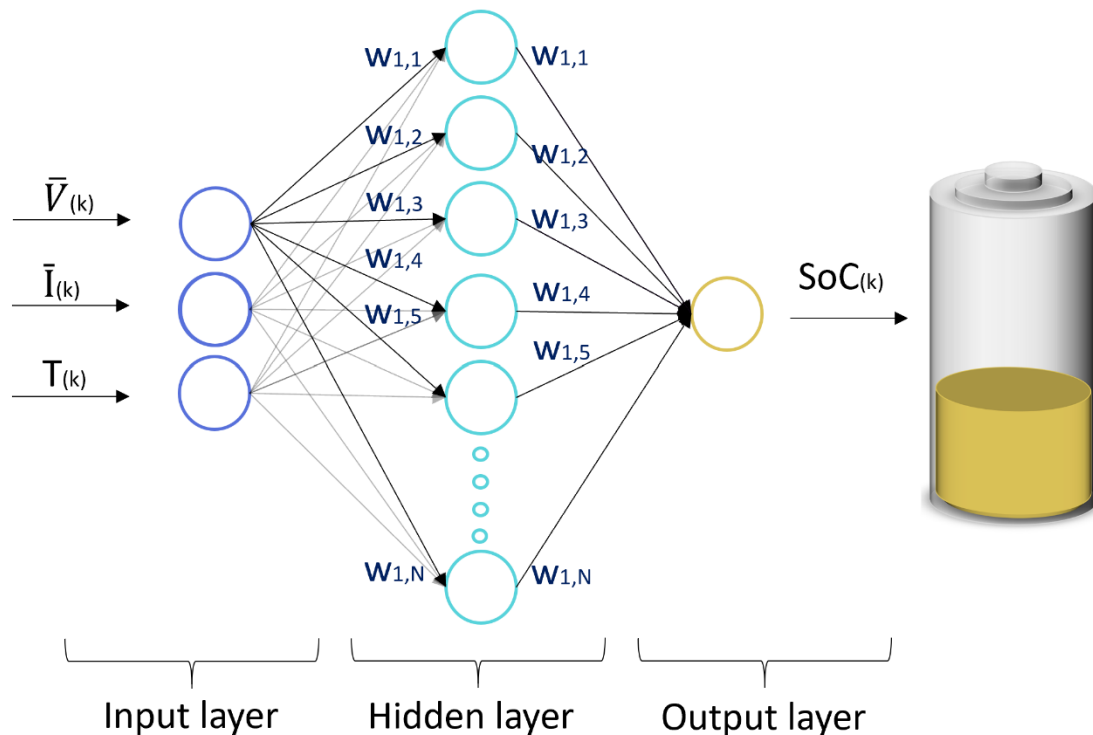


Figure 5. Architecture of a generalized feedforward neural network of voltage (V), current (I), and time (T) inputs, a double layer and nodes where W is the weights, and the state-of-charge (SoC) output.

3.3. Comparison of Approaches

The look-up table approach is straightforward and can map the nonlinear relationship between the SoC and OCV; however, it cannot meet real-time specifications due to the variability of the OCV during operation and is vulnerable to sensor imprecision. The ampere-hour integral approach is also rather straightforward and does not require high-cost computational resources; however, the precision depends largely on the determination of the original SoC and sensor accuracy. The ampere-hour integral approach, together with OCV look-up table or model-based methods, are promising approaches to acquire improved accuracy of SoC estimation in LIBs.

Filter-based and observer-based methods (examples of model-based approaches) can achieve high precision SoC estimation. Despite this, the accuracy of model-based methods depends on the accuracy of the model itself; therefore, extensive computational resources are required as the algorithms within such approaches are complex. Similarly, the type of estimator that can be used also depends on the type of model and the state space system that is used, which makes the comparison of estimators difficult, therefore general remarks are used.

Kalman filter-based approaches are commonly used in calculating SoC in LIBs. The linear Kalman filter-based approach only suits linear structures; however, the extended Kalman filter approach can overcome this as it applies a linearization approximation to nonlinear systems to address this. Despite this, the extended Kalman filter approaches' performance is unstable in terms of linearization error, making it unfit for high-order nonlinear processes as seen in SoC in BEV LIBs. The sigma-point and unscented Kalman filter methods are used for SoC estimation in order to avoid complicated Jacobian matrix and Gaussian noise computation; however, significant computational resources are required to perform such approaches when the number of states/parameters are high. In order to

overcome this, adaptive methods (e.g., adaptive extended and adaptive unscented Kalman filters) have been developed to improve the robustness of the method against measurement and process noise. The particle filter approach is proposed to manage non-Gaussian systems and has shown high precision. More specifically, the unscented and cubature Kalman filter approaches are designed to enhance the process of particle sampling and estimate the likelihood of the state; however, in such approaches, the convergence rate remains a problem.

Observer-based approaches can also achieve adequate estimate precision against unprecise sensor data or initial state errors; however, problems still exist for the determination of the appropriate gain. Data-driven models are dissimilar to model-based approaches and are insensitive to the performance of the model and external environmental factors. The key pitfalls are that these approaches typically entail high computational cost and long processing time, and the precision depends largely on the training data used. Table 1 gives an overview of the mentioned methods for estimating SoC in LIBs.

Table 1. The comparison of SoC estimation methods for LIBs.

	Method	Maximum Error ($\leq \pm$)
Look-up tables	OCV	1.2% [58]
	Impedance	1.4% [66]
Ampere-hour integral	Current integration	4% [69]
	Linear Kalman	2% [72]
Filter	Extended Kalman	1.4% [74]
	Adaptive Kalman	2% [80]
	Sigma-point Kalman	1.2% [85]
	Unscented Kalman	0.12% [87]
	Adaptive unscented Kalman	0.1% [94]
	Central difference Kalman	1.4% [98]
	Cubature Kalman	2.7% [101]
	Particle	0.86% [106]
	Unscented particle	0.9% [114]
	Cubature particle	1.1% [114]
Observer	Luenberger	0.88% [118]
	Sliding mode	2% [120]
	Proportional integral	2.5% [126]
	H-infinity	3.36% [128]
Data-driven	Genetic algorithm	2.98% [33]
	Support vector machine	6% [134]
	Neural network	3.8% [137]

3.4. Errors in Modeling

When implementing model-based SoC estimation, the errors caused by numerous sources and their consequence on estimation should be contemplated. Error amassing directly affects precision and convergence rate of SoC estimation. There are six types of errors that can be caused, such as ampere-hour counting and voltage-based correction when SoC is estimated using a nonlinear observer for a second-order equivalent circuit model (Figure 6) [146]. Hardware and software selection plays a crucial role in SoC estimation and should include high-precision sensors, good power approximation algorithms, and a highly accurate LIB modeling approach to reduce the impact of accumulating various errors.

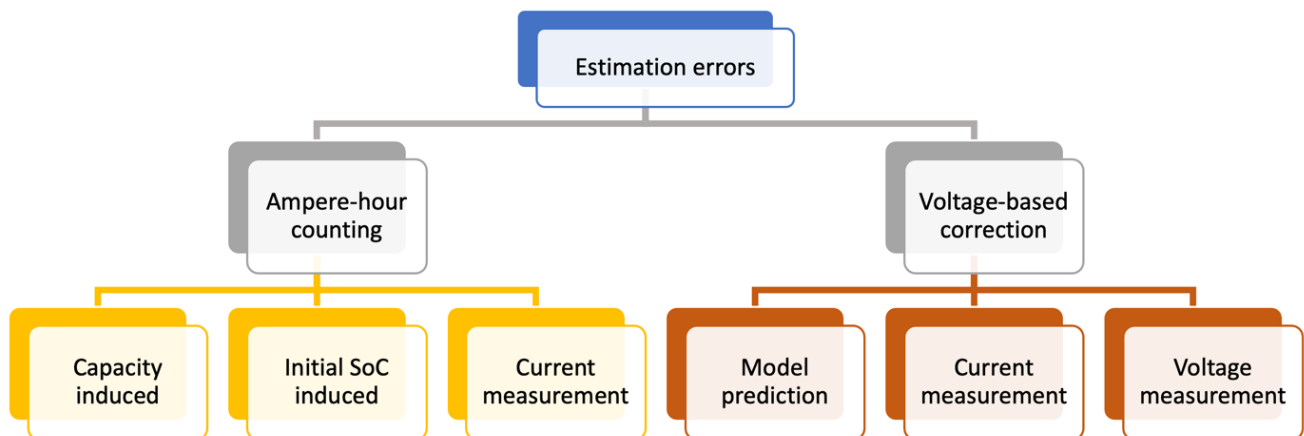


Figure 6. Sources of errors in model-based approaches to state-of-charge estimation in battery electric vehicle Lithium-ion batteries.

3.4.1. Capacity Induced Errors

The decline in capacity observed in aging LIBs is responsible for capacity induced errors in SoC modeling; however, this can be minimized during estimation by updating the LIB capacity within the model. For example, use of Kalman filter algorithms for SoC estimation can significantly reduce the effect of capacity induced errors during dynamic loading [149]. Furthermore, when a highly accurate battery model is used, capacity induced SoC errors can be reduced [150].

3.4.2. Initial SoC Error

Initial SoC error is the discrepancy between the actual initial SoC and the initial SoC estimate. To initialize the calculation, the first iteration requires initial SoC error to provide the initial SoC. The initial SoC value can be acquired from the look-up table or the current saved SoC value. Due to the recursive update of Kalman filter gain during estimation [149], the initial SoC error value is decreased and almost removed within the initial iterations. The initial SoC error converges rapidly to zero with significant observer gain [150].

3.4.3. Current Measurement Error

Measured current is fed to the ampere-hour counting and voltage-based rectification, both potentially developing current measurement error. Current measurement error is largely triggered by low current sensor accuracy, bias error, and noise produced during measurement. The noise impact on SoC errors is marginal due to integration in ampere-hour counting methods. Typically, the current measurement error produced in ampere-hour counting is unidirectional and can expand in the first iterations of the model [149]. The current measurement error produced in voltage-based correction is contrary to the current measurement error caused in ampere-hour counting [150]. Therefore, due to the current measurement error created by ampere-hour counting, the observer gain escalates and the comprehensive eradication of the error could be achievable within numerous iterations of the model [150].

3.4.4. Model Prediction Error

Model prediction error indicates to the variance among the estimated model output (potential in the model) and actual output (actual potential in the model) and has a significant contribution to SoC estimation error. Model prediction error can be separated into two segments, bias error, and noise. Model potential noise does not adversely disturb SoC estimation and accuracy [149]; however, the model prediction error's high dependence on the OCV-to-SoC association is a perplexing problem for SoC estimation using Kalman filter algorithms [150].

3.4.5. Voltage Measurement Error

Voltage measurement error can also occur as a consequence of bias and noise error. The noise error depends on the environment and is difficult to estimate; however, the most estimation algorithms can proficiently quash noise [150]. Bias error can be produced by the discrepancies in actual voltage and measured voltage of the LIB. To counteract this bias defect, the model can fabricate voltage divergence processes [149].

4. Challenges in SoC Estimation in LIBs

With the popularization of BEV LIB storage technologies, innovative control technology has grown in interest. It is difficult to correctly model the LIB structure and predict the condition, which significantly affects BMS efficiency and effectiveness. Research in specific areas is still required to obtain advanced BMS systems.

4.1. Advanced Optical Fiber Sensing

The internal states and parameters are hard to calculate directly within a LIB using traditional sensor methods. Although LIB state estimation algorithms exist, there is still room for calculation and parameter recognition errors. Therefore, contemporary sensor technologies are required to directly measure internal parameters of the BEV LIB. Optical fiber technology is one such method for determining internal measurements of LIB parameters [151–161]. These can be directly embedded within the LIB to measure temperature and strain [162–164], where the latter could improve SoC estimation. Optical fiber sensor technology is expected to enhance battery management, with future research aimed to develop real-time, reliable, and stable sensors merged with multisensor data fusion tools to achieve better knowledge of the LIBs internal state.

4.2. Multi-State Estimation

The single-state estimation approach is well researched; however, there is potential for the BEV LIB state to be nonuniform, which leads to errors in SoC estimation with the current approach. Multi-state estimation may provide a useful approach to account for multiple states within the BEV LIB [165–167]. These approaches have dramatically increased the SoC, voltage and power estimation precision. This area of research in BEV LIB SoC is still in its early stages of development, but it is a promising path for future progress in SoC estimation.

4.3. Battery Model Selection and Estimated Parameter Accuracy

Different LIB types retain their features in differing circumstances; however, owing to the intricate electrochemical behavior in complex conditions, model choice for SoC estimation becomes difficult. Model complexity is also an important aspect, largely based on the amount of model factors that are required to be defined. For selecting the ideal SoC estimation algorithm, the compromise amongst simplicity and accuracy is a vital parameter.

4.4. Operating Conditions

Model-based SoC estimation accuracy is significantly affected by the heterogeneity of BEV LIB model features, and the relationship between OCV and SoC induced by temperature, c-rate, and SoC range [168–170]. It has been observed that the internal resistance of a LIB is impartial to the effective SoC range when using various external temperature environments. Model parameter values are extremely temperature sensitive, but affected by the c-rate and SoC range to a lesser extent. Alternatively, the OCV-to-SoC association remains consistent with varying external working conditions (Figure 7). Despite this, when the SoC is below 10%, the OCV will lead to significant errors in SoC estimation. For BEV LIBs used in varying operating conditions, changes in temperature and c-rate (e.g., driving style and charging rate) are high, making the variation of the model parameters more significant. For reliable SoC prediction in such applications, the model parameters must also be revised online [171–173].

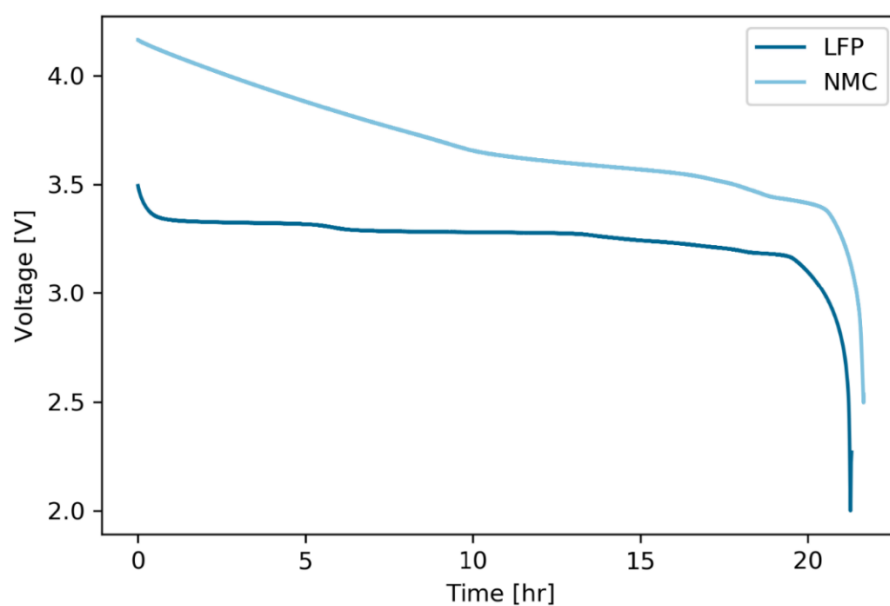


Figure 7. The decreasing voltage of two common LIB chemistries during discharge. The battery data were obtained from in [174] using low C-rate discharge ($C/22$), allowing the terminal voltage to be used to approximate the OCV.

5. Conclusions

This review has presented the physical properties that outline SoC in BEV LIBs. Current and voltage are the main physical parameters used in order to estimate the SoC in BEV LIBs. These can be straightforwardly used with physical electrochemical, data-driven models, and electrical equivalent models in order to estimate the SoC, with the latter two models being routinely used in LIB SoC estimation in BEVs. However, the accuracy of these approaches is low, and this can become a concern in optimal monitoring of the LIB state.

The alternative is to use state estimation methods based on look-up tables, ampere hour integrals, filter-based, observer-based, or data-driven estimation algorithms to determine the state. These methods allow a higher accuracy in state estimation for LIBs, but still produce some error. They also can require significant computational power, resulting in significant time in order to acquire the estimated state.

In order to improve SoC estimation, research should focus on optimization of estimation approaches that allow SoC estimation without significant computational power. Additionally, focus should be directed towards reducing the error accumulation by improving the fundamental understanding of battery function. By focusing on capacity induced error, initial SoC error, current measurement error, and voltage measurement error, the error in the model estimation should be minimized. This could allow the use of less demanding algorithms with smaller errors. In order to achieve this, many challenges are required to be overcome. Such challenges could be alleviated with use of internal optical fiber sensing, multi-state estimation, estimated model parameter accuracy, and operating conditions. Overall, there is still significant progress to be made in this area of research, but the convergence point is getting closer.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors acknowledge the support from the ENERSENSE research initiative, NTNU, Norway. The authors also acknowledge Freyr Battery AS, NTNU (Project No. 90492503) and EIT Innoenergy SE (Project No. 02-2019-IP172-FREYR).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rahimi-Eichi, H.; Ojha, U.; Baronti, F.; Chow, M.-Y. Battery management system: An overview of its application in the smart grid and electric vehicles. *IEEE Ind. Electron. Mag.* **2013**, *7*, 4–16. [[CrossRef](#)]
2. Felius, L.C.; Lamb, J.J.; Hrynyszyn, B.D.; Dessen, F. Smart components and systems. In *Energy-Smart Buildings: Design, Construction and Monitoring of Buildings for Improved Energy Efficiency*; IOP Publishing: Bristol, UK, 2020; Volume 1, pp. 1–16.
3. Lamb, J.J.; Pollet, B.G.; Burheim, O.S. Energy storage. In *Energy-Smart Buildings Design: Construction and Monitoring of Buildings for Improved Energy Efficiency*; IOP Publishing: Bristol, UK, 2020; Volume 1, pp. 1–14.
4. Hamre, B.; Bracchi, T.; Felius, L.C.; Burheim, O.S.; Pollet, B.G.; Lamb, J.J. Energy production in buildings. In *Energy-Smart Buildings: Design, Construction and Monitoring of Buildings for Improved Energy Efficiency*; IOP Publishing: Bristol, UK, 2020; Volume 1, pp. 1–13.
5. Aaldering, L.J.; Leker, J.; Song, C.H. Analysis of technological knowledge stock and prediction of its future development potential: The case of lithium-ion batteries. *J. Clean. Prod.* **2019**, *223*, 301–311. [[CrossRef](#)]
6. Feng, X.; Ouyang, M.; Liu, X.; Lu, L.; Xia, Y.; He, X. Thermal runaway mechanism of lithium ion battery for electric vehicles: A review. *Energy Storage Mater.* **2018**, *10*, 246–267. [[CrossRef](#)]
7. Hu, X.; Cao, D.; Egardt, B. Condition monitoring in advanced battery management systems: Moving horizon estimation using a reduced electrochemical model. *IEEE/ASME Trans. Mechatron.* **2017**, *23*, 167–178. [[CrossRef](#)]
8. Chaturvedi, N.A.; Klein, R.; Christensen, J.; Ahmed, J.; Kojic, A. Algorithms for advanced battery-management systems. *IEEE Control Syst. Mag.* **2010**, *30*, 49–68.
9. Wang, Y.; Chen, Z.; Zhang, C. On-line remaining energy prediction: A case study in embedded battery management system. *Appl. Energy* **2017**, *194*, 688–695. [[CrossRef](#)]
10. Wang, Y.; Tian, J.; Sun, Z.; Wang, L.; Xu, R.; Li, M.; Chen, Z. A comprehensive review of battery modeling and state estimation approaches for advanced battery management systems. *Renew. Sustain. Energy Rev.* **2020**, *131*, 110015. [[CrossRef](#)]
11. Zhang, Q.; Wang, D.; Yang, B.; Cui, X.; Li, X. Electrochemical model of lithium-ion battery for wide frequency range applications. *Electrochim. Acta* **2020**, *34*, 136094. [[CrossRef](#)]
12. Nejad, S.; Gladwin, D.T.; Stone, D.A. A systematic review of lumped-parameter equivalent circuit models for real-time estimation of lithium-ion battery states. *J. Power Sources* **2016**, *316*, 183–196. [[CrossRef](#)]
13. Seaman, A.; Dao, T.-S.; McPhee, J. A survey of mathematics-based equivalent-circuit and electrochemical battery models for hybrid and electric vehicle simulation. *J. Power Sources* **2014**, *256*, 410–423. [[CrossRef](#)]
14. Wei, J.; Dong, G.; Chen, Z. Remaining useful life prediction and state of health diagnosis for lithium-ion batteries using particle filter and support vector regression. *IEEE Trans. Ind. Electron.* **2017**, *65*, 5634–5643. [[CrossRef](#)]
15. Wang, Y.; Yang, D.; Zhang, X.; Chen, Z. Probability based remaining capacity estimation using data-driven and neural network model. *J. Power Sources* **2016**, *315*, 199–208. [[CrossRef](#)]
16. Romero-Becerril, A.; Alvarez-Icaza, L. Comparison of discretization methods applied to the single-particle model of lithium-ion batteries. *J. Power Sources* **2011**, *196*, 10267–10279. [[CrossRef](#)]
17. Grandjean, T.R.B.; Li, L.; Odio, M.X.; Widanage, W.D. Global Sensitivity Analysis of the Single Particle Lithium-Ion Battery Model with Electrolyte. In Proceedings of the 2019 IEEE Vehicle Power and Propulsion Conference (VPPC), Hanoi, Vietnam, 14–17 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–7.
18. Doyle, M.; Fuller, T.F.; Newman, J. Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell. *J. Electrochem. Soc.* **1993**, *140*, 1526. [[CrossRef](#)]
19. Dao, T.-S.; Vyasrayani, C.P.; McPhee, J. Simplification and order reduction of lithium-ion battery model based on porous-electrode theory. *J. Power Sources* **2012**, *198*, 329–337. [[CrossRef](#)]
20. Han, X.; Ouyang, M.; Lu, L.; Li, J. Simplification of physics-based electrochemical model for lithium ion battery on electric vehicle. Part II: Pseudo-two-dimensional model simplification and state of charge estimation. *J. Power Sources* **2015**, *278*, 814–825. [[CrossRef](#)]
21. Guo, X.; Kang, L.; Yao, Y.; Huang, Z.; Li, W. Joint estimation of the electric vehicle power battery state of charge based on the least squares method and the Kalman filter algorithm. *Energies* **2016**, *9*, 100. [[CrossRef](#)]
22. Johnson, V.H. Battery performance models in ADVISOR. *J. Power Sources* **2002**, *110*, 321–329. [[CrossRef](#)]
23. Liaw, B.Y.; Nagasubramanian, G.; Jungst, R.G.; Doughty, D.H. Modeling of lithium ion cells—A simple equivalent-circuit model approach. *Solid State Ion.* **2004**, *175*, 835–839.
24. Verbrugge, M.; Tate, E. Adaptive state of charge algorithm for nickel metal hydride batteries including hysteresis phenomena. *J. Power Sources* **2004**, *126*, 236–249. [[CrossRef](#)]
25. Wang, Y.; Liu, C.; Pan, R.; Chen, Z. Modeling and state-of-charge prediction of lithium-ion battery and ultracapacitor hybrids with a co-estimator. *Energy* **2017**, *121*, 739–750. [[CrossRef](#)]

26. Zhang, X.; Lu, J.; Yuan, S.; Yang, J.; Zhou, X. A novel method for identification of lithium-ion battery equivalent circuit model parameters considering electrochemical properties. *J. Power Sources* **2017**, *345*, 21–29. [[CrossRef](#)]
27. Brand, J.; Zhang, Z.; Agarwal, R.K. Extraction of battery parameters of the equivalent circuit model using a multi-objective genetic algorithm. *J. Power Sources* **2014**, *247*, 729–737. [[CrossRef](#)]
28. Gao, P.; Zhang, C.; Wen, G. Equivalent circuit model analysis on electrochemical impedance spectroscopy of lithium metal batteries. *J. Power Sources* **2015**, *294*, 67–74. [[CrossRef](#)]
29. Jang, J.; Yoo, J. Equivalent circuit evaluation method of lithium polymer battery using bode plot and numerical analysis. *IEEE Trans. Energy Convers.* **2011**, *26*, 290–298. [[CrossRef](#)]
30. Freeborn, T.J.; Maundy, B.; Elwakil, A.S. Fractional-order models of supercapacitors, batteries and fuel cells: A survey. *Mater. Renew. Sustain. Energy* **2015**, *4*, 9. [[CrossRef](#)]
31. Yang, Q.; Xu, J.; Cao, B.; Li, X. A simplified fractional order impedance model and parameter identification method for lithium-ion batteries. *PLoS ONE* **2017**, *12*, e0172424. [[CrossRef](#)]
32. Zou, Y.; Li, S.E.; Shao, B.; Wang, B. State-space model with non-integer order derivatives for lithium-ion battery. *Appl. Energy* **2016**, *161*, 330–336. [[CrossRef](#)]
33. Mu, H.; Xiong, R.; Zheng, H.; Chang, Y.; Chen, Z. A novel fractional order model based state-of-charge estimation method for lithium-ion battery. *Appl. Energy* **2017**, *207*, 384–393. [[CrossRef](#)]
34. Xiong, R.; Tian, J.; Shen, W.; Sun, F. A novel fractional order model for state of charge estimation in lithium ion batteries. *IEEE Trans. Veh. Technol.* **2018**, *68*, 4130–4139. [[CrossRef](#)]
35. Wang, Q.-K.; He, Y.-J.; Shen, J.-N.; Ma, Z.-F.; Zhong, G.-B. A unified modeling framework for lithium-ion batteries: An artificial neural network based thermal coupled equivalent circuit model approach. *Energy* **2017**, *138*, 118–132. [[CrossRef](#)]
36. Zhang, C.; Zhu, Y.; Dong, G.; Wei, J. Data-driven lithium-ion battery states estimation using neural networks and particle filtering. *Int. J. Energy Res.* **2019**, *43*, 8230–8241. [[CrossRef](#)]
37. Thangavel, V.; Guerrero, O.X.; Quiroga, M.; Mikala, A.M.; Rucci, A.; Franco, A.A. A three dimensional kinetic Monte Carlo model for simulating the carbon/sulfur mesostructural evolutions of discharging lithium sulfur batteries. *Energy Storage Mater.* **2020**, *24*, 472–485. [[CrossRef](#)]
38. Shrivastava, P.; Soon, T.K.; Idris, M.Y.I.B.; Mekhilef, S. Overview of model-based online state-of-charge estimation using Kalman filter family for lithium-ion batteries. *Renew. Sustain. Energy Rev.* **2019**, *113*, 109233. [[CrossRef](#)]
39. Kim, I.-S. Nonlinear state of charge estimator for hybrid electric vehicle battery. *IEEE Trans. Power Electron.* **2008**, *23*, 2027–2034.
40. Plett, G.L. Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 3. State and parameter estimation. *J. Power Sources* **2004**, *134*, 277–292. [[CrossRef](#)]
41. Feng, F.; Lu, R.; Zhu, C. A combined state of charge estimation method for lithium-ion batteries used in a wide ambient temperature range. *Energies* **2014**, *7*, 3004–3032. [[CrossRef](#)]
42. Meissner, E.; Richter, G. Battery monitoring and electrical energy management: Precondition for future vehicle electric power systems. *J. Power Sources* **2003**, *116*, 79–98. [[CrossRef](#)]
43. Zhang, C.; Jiang, J.; Zhang, L.; Liu, S.; Wang, L.; Loh, P.C. A generalized SOC-OCV model for lithium-ion batteries and the SOC estimation for LNMCO battery. *Energies* **2016**, *9*, 900. [[CrossRef](#)]
44. Del Olmo, D.; Pavelka, M.; Kosek, J. Open-Circuit Voltage Comes from Non-Equilibrium Thermodynamics. *J. Non-Equilib. Thermodyn.* **2020**, *46*, 91–108. [[CrossRef](#)]
45. Vágner, P.; Kodým, R.; Bouzek, K. Thermodynamic analysis of high temperature steam and carbon dioxide systems in solid oxide cells. *Sustain. Energy Fuels* **2019**, *3*, 2076–2086. [[CrossRef](#)]
46. Pavelka, M.; Wandschneider, F.; Mazur, P. Thermodynamic derivation of open circuit voltage in vanadium redox flow batteries. *J. Power Sources* **2015**, *293*, 400–408. [[CrossRef](#)]
47. Lee, S.; Kim, J.; Lee, J.; Cho, B.H. State-of-charge and capacity estimation of lithium-ion battery using a new open-circuit voltage versus state-of-charge. *J. Power Sources* **2008**, *185*, 1367–1373. [[CrossRef](#)]
48. Cuma, M.U.; Koroglu, T. A comprehensive review on estimation strategies used in hybrid and battery electric vehicles. *Renew. Sustain. Energy Rev.* **2015**, *42*, 517–531. [[CrossRef](#)]
49. Hannan, M.A.; Lipu, M.S.H.; Hussain, A.; Mohamed, A. A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations. *Renew. Sustain. Energy Rev.* **2017**, *78*, 834–854. [[CrossRef](#)]
50. Waag, W.; Sauer, D.U. Adaptive estimation of the electromotive force of the lithium-ion battery after current interruption for an accurate state-of-charge and capacity determination. *Appl. Energy* **2013**, *111*, 416–427. [[CrossRef](#)]
51. Leng, F.; Tan, C.M.; Yazami, R.; Le, M.D. A practical framework of electrical based online state-of-charge estimation of lithium ion batteries. *J. Power Sources* **2014**, *255*, 423–430. [[CrossRef](#)]
52. Dincer, I.; Hamut, H.S.; Javani, N. *Thermal Management of Electric Vehicle Battery Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2016; ISBN 1118900219.
53. Andrea, D. *Battery Management Systems for Large Lithium Ion Battery Packs*; Artech House: Norwood, MA, USA, 2010; ISBN 1608071057.
54. Rahn, C.D.; Wang, C.-Y. *Battery Systems Engineering*; John Wiley & Sons: Hoboken, NJ, USA, 2013; ISBN 1118517059.

55. Lillehei, C.W.; Cruz, A.B.; Johnsrude, I.; Sellers, R.D. A new method of assessing the state of charge of implanted cardiac pacemaker batteries. *Am. J. Cardiol.* **1965**, *16*, 717–721. [[CrossRef](#)]
56. Tan, X. *Electric Vehicle Power Battery Management System Design*; Sun Yat-sen University Press: Guangzhou, China, 2011.
57. How, D.N.T.; Hannan, M.A.; Lipu, M.S.H.; Ker, P.J. State of charge estimation for lithium-ion batteries using model-based and data-driven methods: A review. *IEEE Access* **2019**, *7*, 136116–136136. [[CrossRef](#)]
58. Pop, V.; Bergveld, H.J.; het Veld, J.H.G.O.; Regtien, P.P.L.; Danilov, D.; Notten, P.H.L. Modeling battery behavior for accurate state-of-charge indication. *J. Electrochem. Soc.* **2006**, *153*, A2013. [[CrossRef](#)]
59. Ali, M.U.; Zafar, A.; Nengroo, S.H.; Hussain, S.; Junaid Alvi, M.; Kim, H.-J. Towards a smarter battery management system for electric vehicle applications: A critical review of lithium-ion battery state of charge estimation. *Energies* **2019**, *12*, 446. [[CrossRef](#)]
60. Klintberg, A.; Zou, C.; Fridholm, B.; Wik, T. Kalman filter for adaptive learning of two-dimensional look-up tables applied to OCV-curves for aged battery cells. *Control Eng. Pract.* **2019**, *84*, 230–237. [[CrossRef](#)]
61. Xing, Y.; He, W.; Pecht, M.; Tsui, K.L. State of charge estimation of lithium-ion batteries using the open-circuit voltage at various ambient temperatures. *Appl. Energy* **2014**, *113*, 106–115. [[CrossRef](#)]
62. Dong, G.; Wei, J.; Zhang, C.; Chen, Z. Online state of charge estimation and open circuit voltage hysteresis modeling of LiFePO₄ battery using invariant imbedding method. *Appl. Energy* **2016**, *162*, 163–171. [[CrossRef](#)]
63. Ovejas, V.J.; Cuadras, A. Effects of cycling on lithium-ion battery hysteresis and overvoltage. *Sci. Rep.* **2019**, *9*, 1–9. [[CrossRef](#)] [[PubMed](#)]
64. Rashid, M.; Pathan, T.S.; McGordon, A.; Kendrick, E.; Widanage, W.D. Investigation of hysteresis and relaxation behaviour in graphite and LiNi_{0.33}Mn_{0.33}Co_{0.33}O₂ electrodes. *J. Power Sources* **2019**, *440*, 227153. [[CrossRef](#)]
65. Zheng, L.; Zhang, L.; Zhu, J.; Wang, G.; Jiang, J. Co-estimation of state-of-charge, capacity and resistance for lithium-ion batteries based on a high-fidelity electrochemical model. *Appl. Energy* **2016**, *180*, 424–434. [[CrossRef](#)]
66. Bao, Y.; Dong, W.; Wang, D. Online internal resistance measurement application in lithium ion battery capacity and state of charge estimation. *Energies* **2018**, *11*, 1073. [[CrossRef](#)]
67. Huet, F.; Nogueira, R.P.; Lailler, P.; Torcheux, L. Investigation of the high-frequency resistance of a lead-acid battery. *J. Power Sources* **2006**, *158*, 1012–1018. [[CrossRef](#)]
68. Huang, W.; Qahouq, J.A.A. An online battery impedance measurement method using DC–DC power converter control. *IEEE Trans. Ind. Electron.* **2014**, *61*, 5987–5995. [[CrossRef](#)]
69. Zhang, Y.; Song, W.; Lin, S.; Feng, Z. A novel model of the initial state of charge estimation for LiFePO₄ batteries. *J. Power Sources* **2014**, *248*, 1028–1033. [[CrossRef](#)]
70. Xu, L.; Wang, J.; Chen, Q. Kalman filtering state of charge estimation for battery management system based on a stochastic fuzzy neural network battery model. *Energy Convers. Manag.* **2012**, *53*, 33–39. [[CrossRef](#)]
71. Mastali, M.; Vazquez-Arenas, J.; Fraser, R.; Fowler, M.; Afshar, S.; Stevens, M. Battery state of the charge estimation using Kalman filtering. *J. Power Sources* **2013**, *239*, 294–307. [[CrossRef](#)]
72. Dong, G.; Wei, J.; Chen, Z. Kalman filter for onboard state of charge estimation and peak power capability analysis of lithium-ion batteries. *J. Power Sources* **2016**, *328*, 615–626. [[CrossRef](#)]
73. Yu, Z.; Huai, R.; Xiao, L. State-of-charge estimation for lithium-ion batteries using a kalman filter based on local linearization. *Energies* **2015**, *8*, 7854–7873. [[CrossRef](#)]
74. Xiong, R.; Sun, F.; Chen, Z.; He, H. A data-driven multi-scale extended Kalman filtering based parameter and state estimation approach of lithium-ion polymer battery in electric vehicles. *Appl. Energy* **2014**, *113*, 463–476. [[CrossRef](#)]
75. Chen, Z.; Fu, Y.; Mi, C.C. State of charge estimation of lithium-ion batteries in electric drive vehicles using extended Kalman filtering. *IEEE Trans. Veh. Technol.* **2012**, *62*, 1020–1030. [[CrossRef](#)]
76. Lee, J.; Nam, O.; Cho, B.H. Li-ion battery SOC estimation method based on the reduced order extended Kalman filtering. *J. Power Sources* **2007**, *174*, 9–15. [[CrossRef](#)]
77. Dai, H.; Wei, X.; Sun, Z.; Wang, J.; Gu, W. Online cell SOC estimation of Li-ion battery packs using a dual time-scale Kalman filtering for EV applications. *Appl. Energy* **2012**, *95*, 227–237. [[CrossRef](#)]
78. Lee, K.-T.; Dai, M.-J.; Chuang, C.-C. Temperature-compensated model for lithium-ion polymer batteries with extended Kalman filter state-of-charge estimation for an implantable charger. *IEEE Trans. Ind. Electron.* **2017**, *65*, 589–596. [[CrossRef](#)]
79. He, H.; Xiong, R.; Zhang, X.; Sun, F.; Fan, J. State-of-charge estimation of the lithium-ion battery using an adaptive extended Kalman filter based on an improved Thevenin model. *IEEE Trans. Veh. Technol.* **2011**, *60*, 1461–1469.
80. Xiong, R.; He, H.; Sun, F.; Zhao, K. Evaluation on state of charge estimation of batteries with adaptive extended Kalman filter by experiment approach. *IEEE Trans. Veh. Technol.* **2012**, *62*, 108–117. [[CrossRef](#)]
81. He, H.; Xiong, R.; Guo, H. Online estimation of model parameters and state-of-charge of LiFePO₄ batteries in electric vehicles. *Appl. Energy* **2012**, *89*, 413–420. [[CrossRef](#)]
82. Xiong, R.; Gong, X.; Mi, C.C.; Sun, F. A robust state-of-charge estimator for multiple types of lithium-ion batteries using adaptive extended Kalman filter. *J. Power Sources* **2013**, *243*, 805–816. [[CrossRef](#)]
83. Sepasi, S.; Ghorbani, R.; Liaw, B.Y. A novel on-board state-of-charge estimation method for aged Li-ion batteries based on model adaptive extended Kalman filter. *J. Power Sources* **2014**, *245*, 337–344. [[CrossRef](#)]
84. Zhu, Q.; Xu, M.; Liu, W.; Zheng, M. A state of charge estimation method for lithium-ion batteries based on fractional order adaptive extended kalman filter. *Energy* **2019**, *187*, 115880. [[CrossRef](#)]

85. Seo, B.-H.; Nguyen, T.H.; Lee, D.-C.; Lee, K.-B.; Kim, J.-M. Condition monitoring of lithium polymer batteries based on a sigma-point Kalman filter. *J. Power Electron.* **2012**, *12*, 778–786. [[CrossRef](#)]
86. Bi, Y.; Choe, S.-Y. An adaptive sigma-point Kalman filter with state equality constraints for online state-of-charge estimation of a Li (NiMnCo) O₂/Carbon battery using a reduced-order electrochemical model. *Appl. Energy* **2020**, *258*, 113925. [[CrossRef](#)]
87. Zhang, J.; Xia, C. State-of-charge estimation of valve regulated lead acid battery based on multi-state Unscented Kalman Filter. *Int. J. Electr. Power Energy Syst.* **2011**, *33*, 472–476. [[CrossRef](#)]
88. Chen, Z.; Yang, L.; Zhao, X.; Wang, Y.; He, Z. Online state of charge estimation of Li-ion battery based on an improved unscented Kalman filter approach. *Appl. Math. Model.* **2019**, *70*, 532–544. [[CrossRef](#)]
89. Gholizade-Narm, H.; Charkhgard, M. Lithium-ion battery state of charge estimation based on square-root unscented Kalman filter. *IET Power Electron.* **2013**, *6*, 1833–1841. [[CrossRef](#)]
90. Aung, H.; Low, K.S.; Goh, S.T. State-of-charge estimation of lithium-ion battery using square root spherical unscented Kalman filter (Sqrt-UKFST) in nanosatellite. *IEEE Trans. Power Electron.* **2014**, *30*, 4774–4783. [[CrossRef](#)]
91. Zeng, M.; Zhang, P.; Yang, Y.; Xie, C.; Shi, Y. SOC and SOH Joint Estimation of the Power Batteries Based on Fuzzy Unscented Kalman Filtering Algorithm. *Energies* **2019**, *12*, 3122. [[CrossRef](#)]
92. Sun, F.; Hu, X.; Zou, Y.; Li, S. Adaptive unscented Kalman filtering for state of charge estimation of a lithium-ion battery for electric vehicles. *Energy* **2011**, *36*, 3531–3540. [[CrossRef](#)]
93. Liu, G.; Xu, C.; Li, H.; Jiang, K.; Wang, K. State of charge and online model parameters co-estimation for liquid metal batteries. *Appl. Energy* **2019**, *250*, 677–684. [[CrossRef](#)]
94. Partovibakhsh, M.; Liu, G. An adaptive unscented Kalman filtering approach for online estimation of model parameters and state-of-charge of lithium-ion batteries for autonomous mobile robots. *IEEE Trans. Control Syst. Technol.* **2014**, *23*, 357–363. [[CrossRef](#)]
95. Peng, S.; Chen, C.; Shi, H.; Yao, Z. State of charge estimation of battery energy storage systems based on adaptive unscented Kalman filter with a noise statistics estimator. *IEEE Access* **2017**, *5*, 13202–13212. [[CrossRef](#)]
96. Qiu, X.; Guo, Y.; Zhang, J.; Zhao, H.; Peng, X.; Wu, Z.; Tian, R.; Yang, J. State of Charge Estimation of Lithium Battery Energy Storage Systems Based on Adaptive Correntropy Unscented Kalman Filter. In Proceedings of the 2020 5th Asia Conference on Power and Electrical Engineering (ACPEE), Chengdu, China, 4–7 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 851–857.
97. Lim, J. CDKF approach for estimating a static parameter of carrier frequency offset based on nonlinear measurement equations in OFDM systems. *Nonlinear Dyn.* **2014**, *78*, 703–711. [[CrossRef](#)]
98. Xuan, D.-J.; Shi, Z.; Chen, J.; Zhang, C.; Wang, Y.-X. Real-time estimation of state-of-charge in lithium-ion batteries using improved central difference transform method. *J. Clean. Prod.* **2020**, *252*, 119787. [[CrossRef](#)]
99. Sangwan, V.; Kumar, R.; Rathore, A.K. State-of-charge estimation for li-ion battery using extended Kalman filter (EKF) and central difference Kalman filter (CDKF). In Proceedings of the 2017 IEEE Industry Applications Society Annual Meeting, Cincinnati, OH, USA, 14 March 2018; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
100. Arasaratnam, I.; Haykin, S. Cubature kalman filters. *IEEE Trans. Automat. Contr.* **2009**, *54*, 1254–1269. [[CrossRef](#)]
101. Peng, J.; Luo, J.; He, H.; Lu, B. An improved state of charge estimation method based on cubature Kalman filter for lithium-ion batteries. *Appl. Energy* **2019**, *253*, 113520. [[CrossRef](#)]
102. Nummiaro, K.; Koller-Meier, E.; Van Gool, L. An adaptive color-based particle filter. *Image Vis. Comput.* **2003**, *21*, 99–110. [[CrossRef](#)]
103. Tulsyan, A.; Tsai, Y.; Gopaluni, R.B.; Braatz, R.D. State-of-charge estimation in lithium-ion batteries: A particle filter approach. *J. Power Sources* **2016**, *331*, 208–223. [[CrossRef](#)]
104. Chen, Z.; Sun, H.; Dong, G.; Wei, J.; Wu, J.I. Particle filter-based state-of-charge estimation and remaining-dischargeable-time prediction method for lithium-ion batteries. *J. Power Sources* **2019**, *414*, 158–166. [[CrossRef](#)]
105. Schwunk, S.; Armbruster, N.; Straub, S.; Kehl, J.; Vetter, M. Particle filter for state of charge and state of health estimation for lithium-iron phosphate batteries. *J. Power Sources* **2013**, *239*, 705–710. [[CrossRef](#)]
106. Wang, Y.; Zhang, C.; Chen, Z. A method for state-of-charge estimation of LiFePO₄ batteries at dynamic currents and temperatures using particle filter. *J. Power Sources* **2015**, *279*, 306–311. [[CrossRef](#)]
107. Liu, X.; Chen, Z.; Zhang, C.; Wu, J. A novel temperature-compensated model for power Li-ion batteries with dual-particle-filter state of charge estimation. *Appl. Energy* **2014**, *123*, 263–272. [[CrossRef](#)]
108. Van Der Merwe, R.; Doucet, A.; De Freitas, N.; Wan, E. The unscented particle filter. *Adv. Neural Inf. Process. Syst.* **2000**, *13*, 584–590.
109. He, Y.; Liu, X.; Zhang, C.; Chen, Z. A new model for State-of-Charge (SOC) estimation for high-power Li-ion batteries. *Appl. Energy* **2013**, *101*, 808–814. [[CrossRef](#)]
110. Wang, Y.; Chen, Z. A framework for state-of-charge and remaining discharge time prediction using unscented particle filter. *Appl. Energy* **2020**, *260*, 114324. [[CrossRef](#)]
111. Shen, Y. Hybrid unscented particle filter based state-of-charge determination for lead-acid batteries. *Energy* **2014**, *74*, 795–803. [[CrossRef](#)]
112. Wang, D.; Yang, F.; Tsui, K.-L.; Zhou, Q.; Bae, S.J. Remaining useful life prediction of lithium-ion batteries based on spherical cubature particle filter. *IEEE Trans. Instrum. Meas.* **2016**, *65*, 1282–1291. [[CrossRef](#)]

113. Guo, R.; Gan, Q.; Zhang, J.; Guo, K.; Dong, J. Huber cubature particle filter and online state estimation. *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.* **2017**, *231*, 158–167. [[CrossRef](#)]
114. Xia, B.; Sun, Z.; Zhang, R.; Cui, D.; Lao, Z.; Wang, W.; Sun, W.; Lai, Y.; Wang, M. A comparative study of three improved algorithms based on particle filter algorithms in soc estimation of lithium ion batteries. *Energies* **2017**, *10*, 1149. [[CrossRef](#)]
115. Zhang, K.; Ma, J.; Zhao, X.; Zhang, D.; He, Y. State of Charge Estimation for Lithium Battery Based on Adaptively Weighting Cubature Particle Filter. *IEEE Access* **2019**, *7*, 166657–166666. [[CrossRef](#)]
116. Luenberger, D. An introduction to observers. *IEEE Trans. Automat. Contr.* **1971**, *16*, 596–602. [[CrossRef](#)]
117. Luenberger, D. Observers for multivariable systems. *IEEE Trans. Automat. Contr.* **1966**, *11*, 190–197. [[CrossRef](#)]
118. Hu, X.; Sun, F.; Zou, Y. Estimation of state of charge of a lithium-ion battery pack for electric vehicles using an adaptive Luenberger observer. *Energies* **2010**, *3*, 1586–1603. [[CrossRef](#)]
119. Wang, B.; Liu, Z.; Li, S.E.; Moura, S.J.; Peng, H. State-of-charge estimation for lithium-ion batteries based on a nonlinear fractional model. *IEEE Trans. Control Syst. Technol.* **2016**, *25*, 3–11. [[CrossRef](#)]
120. Huangfu, Y.; Xu, J.; Zhao, D.; Liu, Y.; Gao, F. A novel battery state of charge estimation method based on a super-twisting sliding mode observer. *Energies* **2018**, *11*, 1211. [[CrossRef](#)]
121. Kim, D.; Koo, K.; Jeong, J.J.; Goh, T.; Kim, S.W. Second-order discrete-time sliding mode observer for state of charge determination based on a dynamic resistance li-ion battery model. *Energies* **2013**, *6*, 5538–5551. [[CrossRef](#)]
122. Chen, X.; Shen, W.; Cao, Z.; Kapoor, A. Adaptive gain sliding mode observer for state of charge estimation based on combined battery equivalent circuit model. *Comput. Chem. Eng.* **2014**, *64*, 114–123. [[CrossRef](#)]
123. Chen, X.; Shen, W.; Cao, Z.; Kapoor, A. A novel approach for state of charge estimation based on adaptive switching gain sliding mode observer in electric vehicles. *J. Power Sources* **2014**, *246*, 667–678. [[CrossRef](#)]
124. Kim, D.; Goh, T.; Park, M.; Kim, S.W. Fuzzy sliding mode observer with grey prediction for the estimation of the state-of-charge of a lithium-ion battery. *Energies* **2015**, *8*, 12409–12428. [[CrossRef](#)]
125. Chen, X.; Shen, W.; Dai, M.; Cao, Z.; Jin, J.; Kapoor, A. Robust adaptive sliding-mode observer using RBF neural network for lithium-ion battery state of charge estimation in electric vehicles. *IEEE Trans. Veh. Technol.* **2015**, *65*, 1936–1947. [[CrossRef](#)]
126. Tang, X.; Wang, Y.; Chen, Z. A method for state-of-charge estimation of LiFePO₄ batteries based on a dual-circuit state observer. *J. Power Sources* **2015**, *296*, 23–29. [[CrossRef](#)]
127. Xie, J.; Ma, J.; Sun, Y.; Li, Z. Estimating the state-of-charge of lithium-ion batteries using an H-infinity observer with consideration of the hysteresis characteristic. *J. Power Electron.* **2016**, *16*, 643–653. [[CrossRef](#)]
128. Zhu, Q.; Xiong, N.; Yang, M.-L.; Huang, R.-S.; Hu, G.-D. State of charge estimation for lithium-ion battery based on nonlinear observer: An H_∞ method. *Energies* **2017**, *10*, 679. [[CrossRef](#)]
129. Zhu, Q.; Li, L.; Hu, X.; Xiong, N.; Hu, G.-D. H_∞-Based Nonlinear Observer Design for State of Charge Estimation of Lithium-Ion Battery With Polynomial Parameters. *IEEE Trans. Veh. Technol.* **2017**, *66*, 10853–10865. [[CrossRef](#)]
130. Lipu, M.S.H.; Hannan, M.A.; Hussain, A.; Ayob, A.; Saad, M.H.M.; Karim, T.F.; How, D.N.T. Data-driven state of charge estimation of lithium-ion batteries: Algorithms, implementation factors, limitations and future trends. *J. Clean. Prod.* **2020**, *277*, 124110. [[CrossRef](#)]
131. Vidal, C.; Malysz, P.; Kollmeyer, P.; Emadi, A. Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation: State-of-the-Art. *IEEE Access* **2020**, *8*, 52796–52814. [[CrossRef](#)]
132. Ting, T.O.; Man, K.L.; Lim, E.G.; Leach, M. Tuning of Kalman filter parameters via genetic algorithm for state-of-charge estimation in battery management system. *Sci. World J.* **2014**, *2014*, 176052. [[CrossRef](#)] [[PubMed](#)]
133. Chen, L.; Wang, Z.; Lü, Z.; Li, J.; Ji, B.; Wei, H.; Pan, H. A novel state-of-charge estimation method of lithium-ion batteries combining the grey model and genetic algorithms. *IEEE Trans. Power Electron.* **2017**, *33*, 8797–8807. [[CrossRef](#)]
134. Anton, J.C.A.; Nieto, P.J.G.; Viejo, C.B.; Vilán, J.A.V. Support vector machines used to estimate the battery state of charge. *IEEE Trans. Power Electron.* **2013**, *28*, 5919–5926. [[CrossRef](#)]
135. Hu, J.N.; Hu, J.J.; Lin, H.B.; Li, X.P.; Jiang, C.L.; Qiu, X.H.; Li, W.S. State-of-charge estimation for battery management system using optimized support vector machine for regression. *J. Power Sources* **2014**, *269*, 682–693. [[CrossRef](#)]
136. Meng, J.; Luo, G.; Gao, F. Lithium polymer battery state-of-charge estimation based on adaptive unscented Kalman filter and support vector machine. *IEEE Trans. Power Electron.* **2015**, *31*, 2226–2238. [[CrossRef](#)]
137. Tong, S.; Lacap, J.H.; Park, J.W. Battery state of charge estimation using a load-classifying neural network. *J. Energy Storage* **2016**, *7*, 236–243. [[CrossRef](#)]
138. Dang, X.; Yan, L.; Xu, K.; Wu, X.; Jiang, H.; Sun, H. Open-circuit voltage-based state of charge estimation of lithium-ion battery using dual neural network fusion battery model. *Electrochim. Acta* **2016**, *188*, 356–366. [[CrossRef](#)]
139. Hossain Lipu, M.S.; Hannan, M.A.; Hussain, A.; Saad, M.H.M. Optimal BP neural network algorithm for state of charge estimation of lithium-ion battery using PSO with PCA feature selection. *J. Renew. Sustain. Energy* **2017**, *9*, 64102. [[CrossRef](#)]
140. Sun, F.; Xiong, R.; He, H. A systematic state-of-charge estimation framework for multi-cell battery pack in electric vehicles using bias correction technique. *Appl. Energy* **2016**, *162*, 1399–1409. [[CrossRef](#)]
141. Chemali, E.; Kollmeyer, P.J.; Preindl, M.; Emadi, A. State-of-charge estimation of Li-ion batteries using deep neural networks: A machine learning approach. *J. Power Sources* **2018**, *400*, 242–255. [[CrossRef](#)]
142. Chen, J.; Ouyang, Q.; Xu, C.; Su, H. Neural network-based state of charge observer design for lithium-ion batteries. *IEEE Trans. Control Syst. Technol.* **2017**, *26*, 313–320. [[CrossRef](#)]

143. Lipu, M.S.H.; Hannan, M.A.; Hussain, A.; Saad, M.H.M.; Ayob, A.; Blaabjerg, F. State of charge estimation for lithium-ion battery using recurrent NARX neural network model based lightning search algorithm. *IEEE Access* **2018**, *6*, 28150–28161. [[CrossRef](#)]
144. Hannan, M.A.; Lipu, M.S.H.; Hussain, A.; Ker, P.J.; Mahlia, T.M.I.; Mansor, M.; Ayob, A.; Saad, M.H.; Dong, Z.Y. Toward enhanced State of charge estimation of Lithium-ion Batteries Using optimized Machine Learning techniques. *Sci. Rep.* **2020**, *10*, 1–15.
145. Xia, B.; Cui, D.; Sun, Z.; Lao, Z.; Zhang, R.; Wang, W.; Sun, W.; Lai, Y.; Wang, M. State of charge estimation of lithium-ion batteries using optimized Levenberg-Marquardt wavelet neural network. *Energy* **2018**, *153*, 694–705. [[CrossRef](#)]
146. Yang, F.; Song, X.; Xu, F.; Tsui, K.-L. State-of-charge estimation of lithium-ion batteries via long short-term memory network. *IEEE Access* **2019**, *7*, 53792–53799. [[CrossRef](#)]
147. Liu, Y.; Shu, X.; Yu, H.; Shen, J.; Zhang, Y.; Liu, Y.; Chen, Z. State of charge prediction framework for lithium-ion batteries incorporating long short-term memory network and transfer learning. *J. Energy Storage* **2021**, *37*, 102494. [[CrossRef](#)]
148. Chen, Y.; Li, C.; Chen, S.; Ren, H.; Gao, Z. A combined robust approach based on auto-regressive long short-term memory network and moving horizon estimation for state-of-charge estimation of lithium-ion batteries. *Int. J. Energy Res.* **2021**. [[CrossRef](#)]
149. Zheng, Y.; Ouyang, M.; Han, X.; Lu, L.; Li, J. Investigating the error sources of the online state of charge estimation methods for lithium-ion batteries in electric vehicles. *J. Power Sources* **2018**, *377*, 161–188. [[CrossRef](#)]
150. Shen, P.; Ouyang, M.; Han, X.; Feng, X.; Lu, L.; Li, J. Error analysis of the model-based state-of-charge observer for lithium-ion batteries. *IEEE Trans. Veh. Technol.* **2018**, *67*, 8055–8064. [[CrossRef](#)]
151. Wahl, M.S.; Lamb, J.J.; Muri, H.I.; Snilsberg, R.K.; Hjelme, D.R. Light properties and sensors. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer: Berlin/Heidelberg, Germany, 2020.
152. Muri, H.I.; Wahl, M.S.; Lamb, J.J.; Snilsberg, R.K.; Hjelme, D.R. Sensor fusion. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer International Publishing: New York, NY, USA, 2020.
153. Spitthoff, L.; Lamb, J.J.; Pollet, B.; Burheim, O.S. Lifetime expectancy of lithium-ion batteries. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer: Berlin/Heidelberg, Germany, 2020.
154. Spitthoff, L.; Øyre, E.S.; Muri, H.I.; Wahl, M.S.; Gunnarshaug, A.F.; Pollet, B.; Lamb, J.J.; Burheim, O.S. Thermal management of Lithium ion batteries. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer International Publishing: New York, NY, USA, 2020.
155. Wahl, M.S.; Muri, H.I.; Snilsberg, R.K.; Lamb, J.J.; Hjelme, D.R. Temperature and humidity measurements. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer International Publishing: New York, NY, USA, 2020.
156. Lamb, J.J.; Burheim, O.S.; Pollet, B. Hydrogen fuel cells and water electrolyzers. In *Micro-Optics and Energy: Sensors for Energy Devices*; Springer: Berlin/Heidelberg, Germany, 2020.
157. Yang, G.; Leitão, C.; Li, Y.; Pinto, J.; Jiang, X. Real-time temperature measurement with fiber Bragg sensors in lithium batteries for safety usage. *Measurement* **2013**, *46*, 3166–3172. [[CrossRef](#)]
158. David, N.A.; Wild, P.M.; Hu, J.; Djilali, N. In-fibre Bragg grating sensors for distributed temperature measurement in a polymer electrolyte membrane fuel cell. *J. Power Sources* **2009**, *192*, 376–380. [[CrossRef](#)]
159. Nascimento, M.; Novais, S.; Leitão, C.; Domingues, M.F.; Alberto, N.; Antunes, P.; Pinto, J.L. Lithium batteries temperature and strain fiber monitoring. In Proceedings of the 24th International Conference on Optical Fibre Sensors, Curitiba, Brazil, 28 September–2 October 2015; International Society for Optics and Photonics: Bellingham, WA, USA, 2015; Volume 9634, p. 96347V.
160. Sommer, L.W.; Raghavan, A.; Kiesel, P.; Saha, B.; Schwartz, J.; Lochbaum, A.; Ganguli, A.; Bae, C.-J.; Alamgir, M. Monitoring of intercalation stages in lithium-ion cells over charge-discharge cycles with fiber optic sensors. *J. Electrochem. Soc.* **2015**, *162*, A2664. [[CrossRef](#)]
161. Sommer, L.W.; Kiesel, P.; Ganguli, A.; Lochbaum, A.; Saha, B.; Schwartz, J.; Bae, C.-J.; Alamgir, M.; Raghavan, A. Fast and slow ion diffusion processes in lithium ion pouch cells during cycling observed with fiber optic strain sensors. *J. Power Sources* **2015**, *296*, 46–52. [[CrossRef](#)]
162. Bae, C.; Manandhar, A.; Kiesel, P.; Raghavan, A. Monitoring the strain evolution of lithium-ion battery electrodes using an optical fiber Bragg grating sensor. *Energy Technol.* **2016**, *4*, 851–855. [[CrossRef](#)]
163. Raghavan, A.; Kiesel, P.; Sommer, L.W.; Schwartz, J.; Lochbaum, A.; Hegyi, A.; Schuh, A.; Arakaki, K.; Saha, B.; Ganguli, A. Embedded fiber-optic sensing for accurate internal monitoring of cell state in advanced battery management systems part 1: Cell embedding method and performance. *J. Power Sources* **2017**, *341*, 466–473. [[CrossRef](#)]
164. Ganguli, A.; Saha, B.; Raghavan, A.; Kiesel, P.; Arakaki, K.; Schuh, A.; Schwartz, J.; Hegyi, A.; Sommer, L.W.; Lochbaum, A. Embedded fiber-optic sensing for accurate internal monitoring of cell state in advanced battery management systems part 2: Internal cell signals and utility for state estimation. *J. Power Sources* **2017**, *341*, 474–482. [[CrossRef](#)]
165. Song, Y.; Liu, D.; Liao, H.; Peng, Y. A hybrid statistical data-driven method for on-line joint state estimation of lithium-ion batteries. *Appl. Energy* **2020**, *261*, 114408. [[CrossRef](#)]
166. Feng, F.; Teng, S.; Liu, K.; Xie, J.; Xie, Y.; Liu, B.; Li, K. Co-estimation of lithium-ion battery state of charge and state of temperature based on a hybrid electrochemical-thermal-neural-network model. *J. Power Sources* **2020**, *455*, 227935. [[CrossRef](#)]
167. Hu, X.; Jiang, H.; Feng, F.; Liu, B. An enhanced multi-state estimation hierarchy for advanced lithium-ion battery management. *Appl. Energy* **2020**, *257*, 114019. [[CrossRef](#)]
168. Huang, C.; Wang, Z.; Zhao, Z.; Wang, L.; Lai, C.S.; Wang, D. Robustness evaluation of extended and unscented Kalman filter for battery state of charge estimation. *IEEE Access* **2018**, *6*, 27617–27628. [[CrossRef](#)]

169. Yang, S.; Deng, C.; Zhang, Y.; He, Y. State of charge estimation for lithium-ion battery with a temperature-compensated model. *Energies* **2017**, *10*, 1560. [[CrossRef](#)]
170. Zhang, Y.; Shang, Y.; Cui, N.; Zhang, C. Parameters identification and sensitive characteristics analysis for lithium-ion batteries of electric vehicles. *Energies* **2018**, *11*, 19. [[CrossRef](#)]
171. Shen, P.; Ouyang, M.; Lu, L.; Li, J.; Feng, X. The co-estimation of state of charge, state of health, and state of function for lithium-ion batteries in electric vehicles. *IEEE Trans. Veh. Technol.* **2017**, *67*, 92–103. [[CrossRef](#)]
172. Yu, Q.; Xiong, R.; Lin, C.; Shen, W.; Deng, J. Lithium-ion battery parameters and state-of-charge joint estimation based on H-infinity and unscented Kalman filters. *IEEE Trans. Veh. Technol.* **2017**, *66*, 8693–8701. [[CrossRef](#)]
173. Wang, Q.; Kang, J.; Tan, Z.; Luo, M. An online method to simultaneously identify the parameters and estimate states for lithium ion batteries. *Electrochim. Acta* **2018**, *289*, 376–388. [[CrossRef](#)]
174. CALCE Battery Research Group. CALCE Battery Group. Available online: <https://web.calce.umd.edu/batteries/index.html#> (accessed on 3 May 2021).

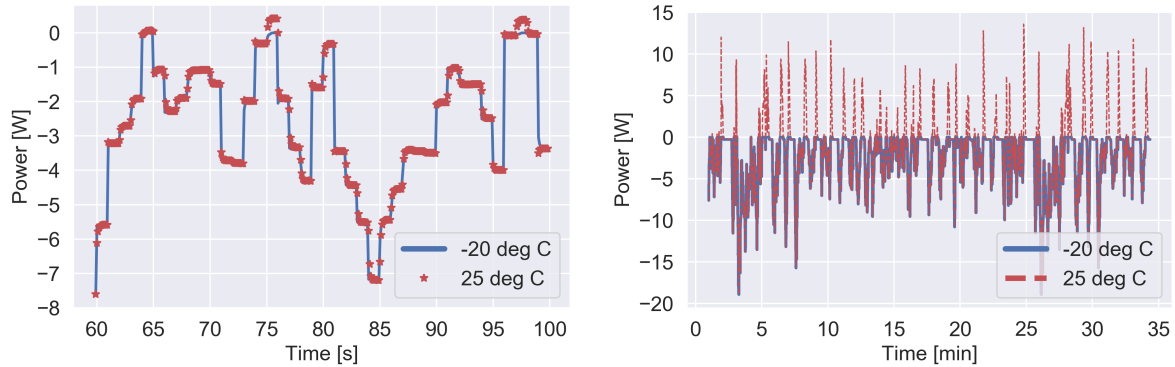
Appendix A

Driving Cycles

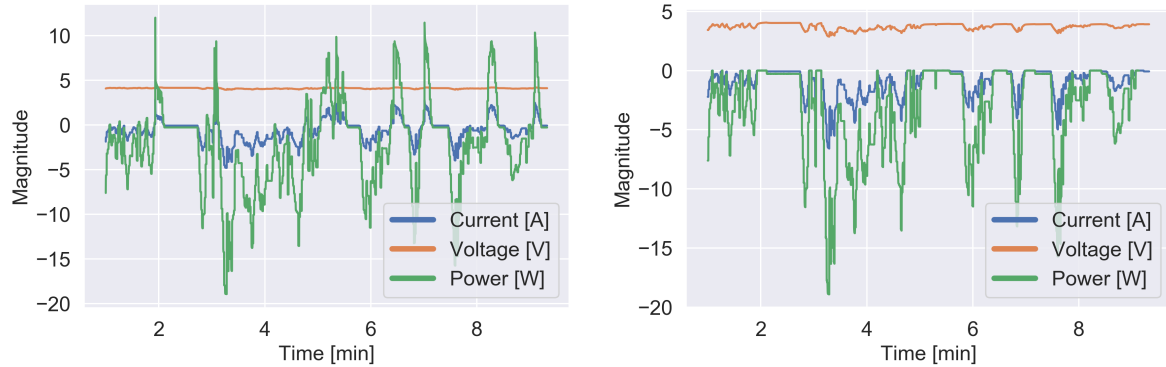
A.1 Plotting Power, Current and Voltage Relations

To understand the driving cycles, plots were created of the power consumption over a time frame. From Figure A.1a and A.1b it can be observed that the power curves plotted over a time frame match for negative values of the power, representing power drawn from the LiB. When the power has positive values, the LiB is charged. Since no regenerating is done for -20°C , no power is neither consumed, nor generated. This is why the curves do not match for positive values of the power.

Figure A.1c and Figure A.1d indicate the relationship between current, voltage and power for 25°C and -20°C respectively. Several interesting facts can be observed. Firstly, the voltage (orange) is distinctly more stable for the higher temperature. Secondly, the current (blue) drawn is, in absolute value, higher for lower temperatures. This is clearly seen right after three minutes where the current drops below -5 only when cycling at -20°C .



(a) UDSS profile of the power output where 25 °C is plotted on top of -20 °C for less than one minute. (b) The same plot as (a) only for a longer time of approximately 35 minutes.



(c) The first ten minutes of the UDSS profile at 25 °C. (d) The first ten minutes of the UDSS profile at -20 °C.

Figure A.1: UDSS driving cycle. The cycle is one of the standards from the United States Environmental Protection Agency (EPA) [106]. The data used for plotting was created by [108]. Subplot (a) and (b) represents a time window of the power profile. Subplot (c) and (d) displays the current, voltage and power profiles of 25 °C and -20 °C respectively.

A.1.1 Experiment

A.1.1.1 Introduction to the Experiment

An essential part of ensuring that the SoC estimator can be used for real-world applications is testing it against new data sets. In some of the works that were reviewed in Chapter , this was not satisfactorily done. To provide insight into these gaps in the literature, a forward neural network was systematically trained and tested for various drive cycles and network structures.

The purpose of the experiment was to evaluate how different partitioning of the data can affect the final accuracy result. Therefore, an FNN was constructed using the library Scikitlearn [117] from Python [118]. The public data set Panasonic NCR18650PF [108] was used. It was tested for three different data partitions. More explicitly, it was tested on a random data partition with a 70:30 split ratio for training and testing, respectively. Moreover, picking out intentionally chosen

cycles for training and testing was done.

An effort was put into conducting a fair experiment. As an example, training on 90% of the data would in general yield a more robust model than only training on 50% of the data. Therefore both 50% of the data sets were left out, as well as 90% of the data sets. The experiments can be categorized into three groups by how the data was partitioned:

1. 30:70 training testing ratio where the data in each set was randomly chosen.
2. Half of the cycles were used for training and the other half was used for testing; Cycle 1-4 and NN were used for training, UDDS, US06, LA92, and two HWFET cycles were used for testing.
3. All but one of the cycles were used for training, the last one was used for testing. The cycles that were picked out, one after one, for testing were UDDS, US06, LA92, and two HWFET cycles.

The HWFET drive cycle was recorded twice, and the data sets were given the name HWFETa and HWFETb. Since HWFETa was used for training when testing with HWFETb, and visa versa, the testing results with these two cycles are biased. Still, they are included in the result section.

Other specifications are that testing and training on one temperature, 25 °C. Moreover, different network structures were constructed in terms of the number of layers and number of nodes. Another factor that impacts accuracy is the number of epochs. Therefore, a range of different epochs was trained on, specifically 1, 20, 100, 200, 300, and 400 epochs. Furthermore, different numbers of layers and nodes per layer were examined, such that not an arbitrary choice of these hyperparameters could majorly affect the result.

No validation set was created. However, none of these models are meant to be a suggestion of an enhanced ML SoC estimator. On the contrary, if one of the models were used, there is a major possibility that they are overfitted or underfitted (training the network too few times) and would yield a large error if tested for real-world applications. The model is only intended to broadly outline how the choice of testing set and model complexity could affect the accuracy result.

A.1.1.2 Method

With the intention of reproducible results, the choice of hyperparameters and functions will be stated below. The error function used was MAE given by Equation 3.5b. For a fair comparison to the models in the literature study, both the RMSE and the MAE should have been recorded. However, since the purpose was only to compare the models of this report against another, only the MAE is stated. The activation function used was ReLU given by Equation 3.3c (there is one exception where hyperbolic tangent Equation 3.3b, was used). The learning rate was initially set to $\alpha = 0.001$, but automatically reduced as the network converged. This is a form of regularization, where the learning rate limit was set to $\alpha = 0.0001$.

In Scikitlearn, there is a possibility to initialize the network in the same way each time the network

is trained. This enables the possibility to create the exact same network several times, implying that the exact results gotten in this report can be reproduced. The choice of initialization can in Scikitlearn be done by giving the parameter "random_state" a value. In this paper, "random_state" was set to 42. This choice was arbitrarily made and only done to yield reproducible results.

The input features are chosen based on the most promising results obtained in the literature of FNNs. Therefore the voltage and the battery temperature at the current time step are used, as well as the mean voltage and mean current over the last 300 time steps. Other time steps that were tested out were 600 and 100, but 300 yielded the best accuracy when the data was trained and tested with a 30:70 split ratio. The input features were scaled by standardization defined by Equation 3.8.

The output is the SoC at a timestamp k . The target SoC was obtained by using Coulomb counting given by Equation 2.14, where the manufactures stated the total capacity of a new battery was used as the total capacity. As previously discussed, this is not the best practice, but with the purpose of a rough comparison of different training and testing strategies, it was found decent.

A.1.1.3 Results of the Experiments

The results stated in Table A.1 indicates a general relationship between the error and the data partitioning. A higher accuracy can be observed for the 30:70 split ratio, than when testing with many specific cycles (yielding $\sim 50\%$ training data) or testing a specific (yielding $\sim 90\%$ training data). The later test is interesting since in general a large training set is preferred and yields typically a high accuracy due to good chances of generalization. However, even with 90% of the data is used for training, none of the test results from the drive cycles were close to as good as from the 30:70 split ratio experiment. This indicates a lack of the 30% unseen data to test the robustness to the extent that only 10% of a new driving set does.

The reason why the split of 30:70 ratio could yield a similar data set for training and testing even though the model had never seen 30% of the data, could be that the sampling frequency of the data set is 0.1 Hz. Consequently, the data points that are sampled right after another have very similar values. Therefore, by only taking out randomly 30% of the data for testing, the test set is similar to the training set.

A trend of overtraining and overfitting is observed for the 30:70 split ratio. This statement is reasoned with the error decreasing with the increased number of training epochs. Additionally, the error decreases with increased complexity in terms of the number of hidden layers and nodes per hidden layer. As previously stated, the best practice in ML is to have a validation set. Then both the error of the training and the validation set could be traced. Since overtraining happens when the training error starts to converge, while the validation error increases, an estimate of the number of epochs where overtraining happens could be made if these two errors were recorded. However, the error of the training data was not recorded, but based on the fact that the ML

algorithm is constructed to minimize the error of the training data, an assumption that the training data error converges with the number of epochs is made. This implies that both the training data and the test data converge. It is suspicious that the error of the test data never starts to increase. One reason could be that the model has become better with each epoch. However, when testing on an unseen drive cycle (row index 12-16), the error is lowest after a certain number of epochs and increases when trained after this point. The exception is LA92, which seems to oscillate. This should be investigated further.

When testing the data for unseen drive cycles, the network with only one hidden layer of 16 nodes performs better than the more complex structure (two hidden layers with 128 and 64 nodes). Moreover, the error for the complex structure goes rapidly up with prolonged training, as seen in Figure A.2 for the one-layered NN. This is the true opposite case of the 30:70 split ratio. This indicates that the complex network structure is overfitted, and when tested on similar data as the training data, the accuracy is low, while the opposite is observed for testing with new drive cycles. This reasoning strengthens the suggestion of the 30% randomly chosen data's lack to test the robustness.

Another observation is that the drive cycles are of different difficulty. Two cycles stand out to be particularly challenging, namely the US06 and UDDS where the lowest MAE is around four. Therefore, testing on different drive cycles is essential to obtain a reliable accuracy result.

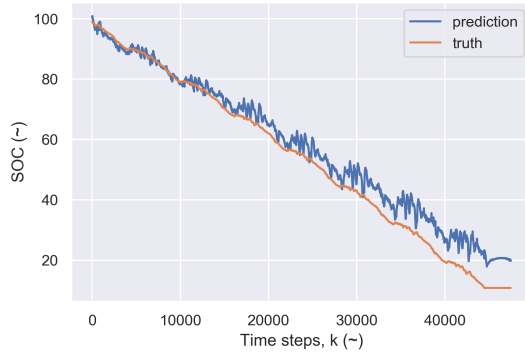
Table A.1: Results of the validation data (no test data used). The experiment consisted of different partitioning of the data into test and training sets. In addition, the number of layers and the number of nodes per layer are differing for some of the experiments, to check what configurations that are easiest overfitted.

Row Index	Test Data	Training Data	Neurons per Layer	MAE 1 Epoch	MAE 20 Epochs	MAE 100 Epochs	MAE 200 Epochs	MAE 300 Epochs	MAE 400 Epochs	Activation Function
1	30 % of 25 deg C	70 % of 25 deg C	16, 16, 32	0.7	0.40, (54 ep)**	0.36 (154 ep)**	0.3 (254 ep)**	0.28 (354 ep)**	*	ReLU
2	30 % of 25 deg C	70 % of 25 deg C	16	2.24	0.69	0.68	0.68	0.68	0.68	ReLU
3	30 % of 25 deg C	70 % of 25 deg C	32, 16	0.81	0.43	0.36	0.34	0.33	0.31	ReLU
4	30 % of 25 deg C	70 % of 25 deg C	16, 32	0.86	0.47	0.39 (120 ep)**	0.37	0.35	0.34	ReLU
5	30 % of 25 deg C	70 % of 25 deg C	64, 32	0.71	0.34	0.25	0.23	0.23	*	ReLU
6	30 % of 25 deg C	70 % of 25 deg C	128, 64	0.60	0.32	0.25	0.22	0.17	0.17	ReLU
7	LA92, UDDS, HWFET, US06	Cyle1-4 + NN	128, 64	[1.74 - 5.48]	[3.12 - 8.37]	*	*	*	[4.10 - 15.92]	ReLU
8	LA92, UDDS, HWFET, US06	Cyle1-4 + NN	64,32	[2.36 - 5.12]	[2.19 - 6.00]	*	*	*	[4.11-18.12]	ReLU
9	LA92, UDDS, HWFET, US06	Cyle1-4 + NN	32, 16	[2.29 - 9.76]	[2.41-10.42]	[3.86 - 12.15]	*	*	[3.22 - 16.41]	ReLU
10	LA92, UDDS, HWFET, US06	Cyle1-4 + NN	16, 32	[2.07 - 5.96]	[2.41-10.42]	*	*	*	[3.25 - 40.45]	ReLU
11	LA92, UDDS, HWFET, US06	Cyle1-4 + NN	16	[2.83 - 9.54]	[2.34-9.82]	[2.06-21.84]	*	*	[1.45 - 13.45]	ReLU
12	US06	Cyle1-4 + NN, LA92, UDDS, HWFET	16	19.25	3.65	2.47	*	*	2.88	Tanh
13	US06	Cyle1-4 + NN, LA92, UDDS, HWFET	16	6.97	4.52	5.32	7.10	8.34	8.70	ReLU
14	UDDS	Cyle1-4 + NN, LA92, HWFET, US06	16	9.54	3.69	5.36	11.21	13.19	13.48	ReLU
15	LA92	Cyle1-4 + NN, LA92, UDDS, US06, HWFET	16	5.74	1.68	1.53	1.47	1.50	1.45	ReLU
16	HWFETa***	Cyle1-4 + NN, LA92, UDDS, US06, HWFETb	16	2.84	1.56	2.66	3.63	4.39	4.90	ReLU
17	HWFETb***	Cyle1-4 + NN, LA92, UDDS, HWFETa, US06	16	2.83	1.45	2.54	3.52	4.3	4.81	ReLU
16	LA92	Cyle1-4 + NN, UDDS, US06, HWFET	128, 64	1.74	2.20	2.44	3.03	3.57	4.10	ReLU
17	UDDS	Cyle1-4 + NN, LA92, US06, HWFET	128, 64	4.08	9.15	9.54	22.23	13.61	9.92	ReLU

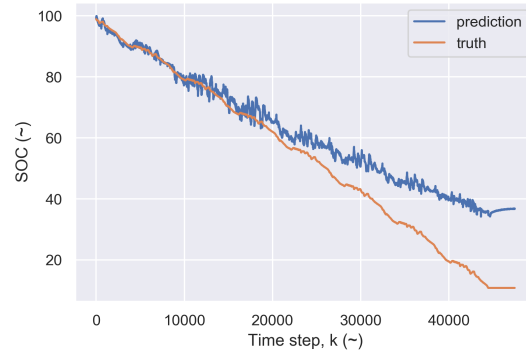
* No recordings were done.

** The number inside the brackets indicates the number of epochs (ep) executed. The experiments were done before a system was made to test the model after a given number of epochs.

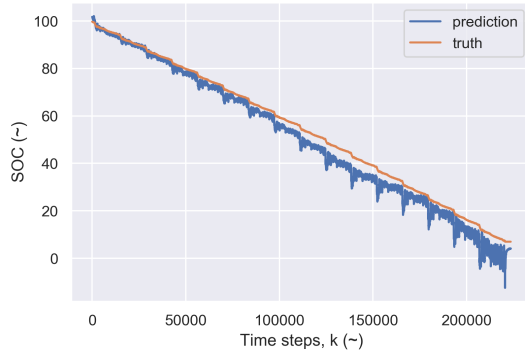
*** HWFETa was used to train with when testing with HWFETb, and visa versus. The results stated for HWFETa and HWFETb are therefore biased.



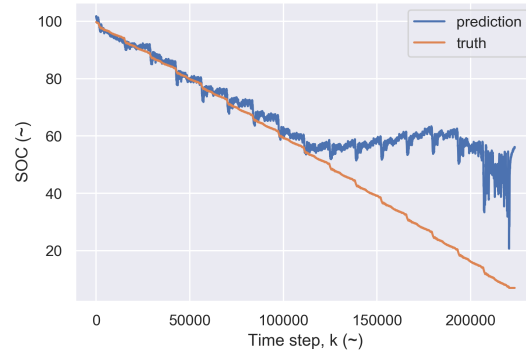
(a) US06 20 epochs, MAE = 4.52.



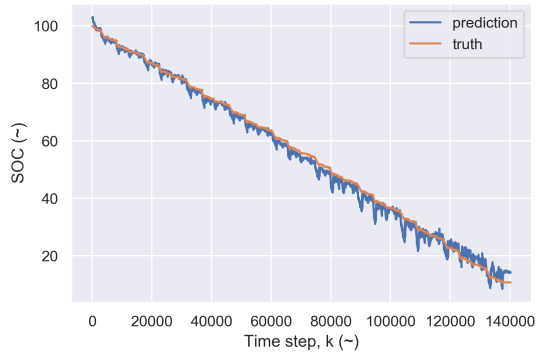
(b) US06 400 epochs, MAE = 8.7.



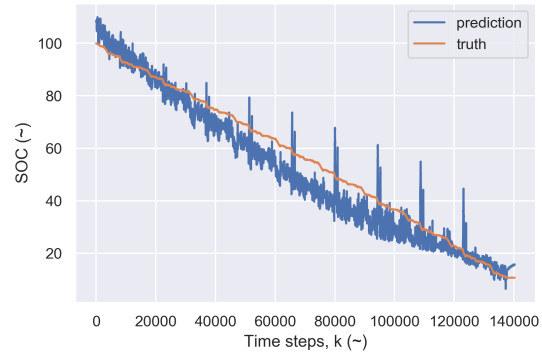
(c) UDDS 20 epochs. MAE = 3.69.



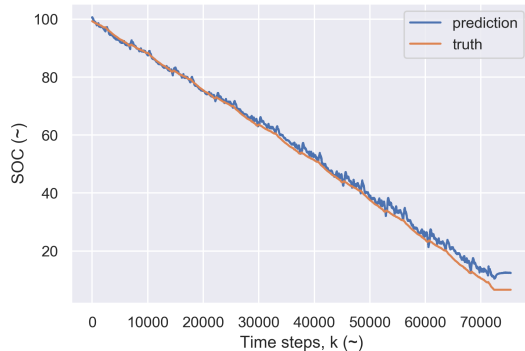
(d) UDDS 400 epochs. MAE = 13.48.



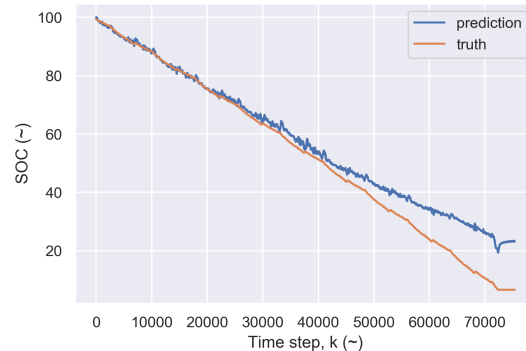
(e) LA92 400 epochs. MAE = 1.45.



(f) LA92 1 epochs. MAE = 5.74.



(g) HWFETa 20 epochs. MAE = 1.45.



(h) HWFETa 400 epochs. MAE = 4.90.

Figure A.2: The predicted results from the FNN when using 9 cycles for training and one for testing. On the left and right side the results with the lowest and highest MAE are displayed respectively. With respect to Table A.1, the graphs corresponds to the results with the row index 13-16. Please remember that HWFETa is somewhat biased since HWFETb was used for training.

Appendix **B**

Data Acquisition for the LCO cell

B.1 Battery Set-up

Set-up of the battery when connected to the cycler. It is the simple configuration with only one channel that was used in the end, because it was not possible to parallel connect the channel with transient profiles with rapid charge and discharging present. In the manual of the Arbin cycler, it states that there is a risk of short-circuiting the battery with rapid changes from charge to discharge, and reversely, when channels are connected in parallel, and therefore it was set a software limitation to prevent this happening. However, with this configuration it is possible to draw higher currents, and a description is therefore kept in this appendix, in case it can be useful in other studies.

The maximum current that can be drawn by one channel of the Arbin cycler is ± 30 A, while the maximum current of the power cycles is above 90 A. However, channels can be connected in parallel to draw more current. A design of connecting channels in parallel was made and can be found in Appendix B. Unfortunately, highly transient profiles with charging and discharging like the power cycles in Figure 5.6, is not possible to cycle with parallel channels using the Arbin cycler due to the risk of short-circuiting the battery.

B.1.1 Simple Configuration

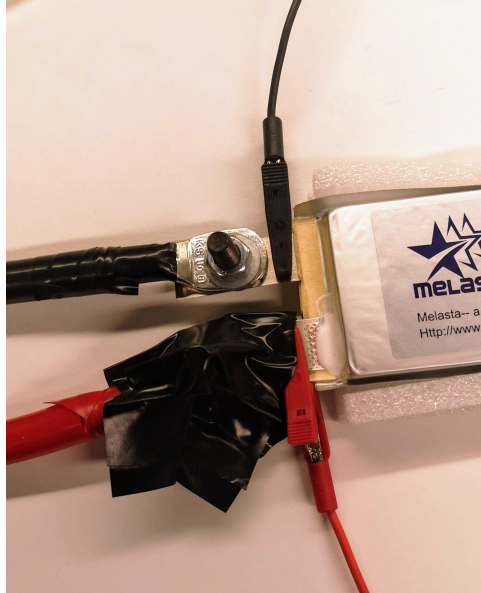


Figure B.1: Simple configuration for connecting one channel to each battery tab. The tab to the cathode is taped to prevent short-circuiting the cell. For the Arbin cycler, a maximum of 30 A can be drawn with this configuration. This is the configuration that was used in this project.

B.1.2 Tab Clamps

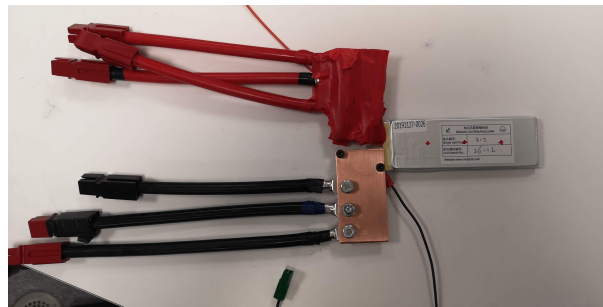


Figure B.2: Back part of the tab clamps.

B.1.3 Internal Resistance Measurements

The internal resistance had to be found in order to cycle the batteries with a CCCV charge so that the Arbin cycler could adjust the current rate accordingly to the internal resistance to obtain a constant voltage. In addition, the internal resistance indicates the SoH of the battery, and was a potential feature for the SoC estimation.

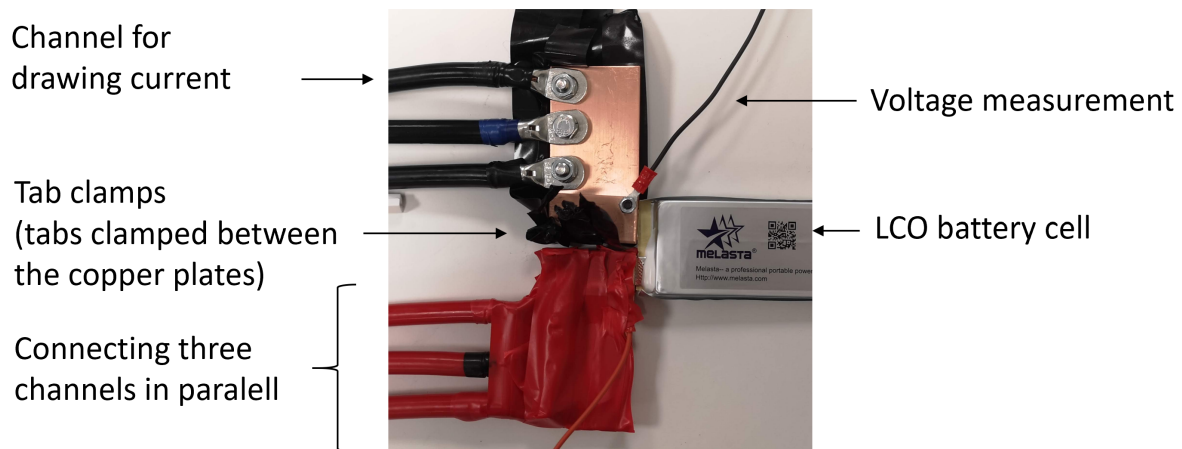


Figure B.3: Tab clamps where three channels can be connected to the battery in parallel, and thereby 90 A can be drawn. It could not be used in this project due to software/hardware issues with transient power profiles switching rapidly between discharge and charge

The internal resistance was calculated based on hybrid pulse power characterization. All measurements were done at 4 V, where a pause of at least 1 minute was present before a new measurement was done. Ideally, the pause should have been longer since the relaxation time of LiBs can be up to several hours or days, due to the resistance of polarization and concentration over-potential are still present when conducting experiments right after one another, and one wish to only extract the Ohmic resistance. However, due to limited time of the thesis and the fact that a lot of tests had to be conducted (>20 per battery) to find the internal resistance, it was not time for a long pause. Therefore, it should be noted that the internal resistance stated, contain a small error.

This cycling and the calculations was obtain directly by the Arbin cyler, and a detailed descriptions are therefore not included here. The parameters that could be adjusted were the amplitude of each pulse, the mean current and the time step between each pulse. The amplitude of each pulse and the mean current were kept constant, to 0.1 A and 0.2 A, respectively, which is the same as the suggestion in the Arbin manual. The time step was set as small as possible, in order to only measure the Ohmic resistance and not the resistance of the overpotenital and concentration potential.

The results from the internal resistance measurements and calculations are found by scanning the QR code of Table 5.4. It is not understood why the internal resistance of the 100% SoH battery has such a high internal resistance. The tests were done after all the cycling were done, so some increase in internal resistance is expected. However, the battery with 100% SoH needed 6 ms to obtain the results, compared to 2-3 for the two other batteries. It was tried to take the test after resting for 24 hours. This did however not help.

Appendix C

Results

Detailed information on the scores obtained by FNN are found bellow.

C.0.1 FNN - Random Search Panasonic Data Set

Table C.1, displays the different search space of the random search and the corresponding error measurements. The rows of this table, is every time the search space was changed, new input features were trained with, or new data sets were tested and validated with. As can be seen from this table, more than one parameters were changed between each time. This was done based on the network that performed the best. For example, if the best five networks consisted of 2-3 layers, and the search space was between 1-6 layers, then the next tuning was done with a tapered search space of 1-4 layers.

Table C.1: Sores from the best model of each tuning, with the search space. To limit the search space, the algorithm could only pick even values for the number of nodes per layer, meaning the range had a step size of two. The exception is the first tuning where the step size is 8.

Tuning No.	Best MAE SoC[%]	Best RMSE SoC[%]	Avr 5 best RMSE SoC[%]	CCCV	Input	First layer (min, max No. nodes)	Additional hidden layers	Min No. nodes	Max No. nodes (step)	Activation function	Dropout rate	Output activation function	Max combi	No. Executions per trial	No. epochs	Validation data used for error scores	Window Size	
1	6.17	8.30	8.50	No	i_{300}, V_{300}, T	NA	1-6	4	68	8	relu, linear	0, 0.1, 0.2, 0.3, 0.5	relu, linear, sigmoid	20	5	20	US06	300
2	2.73	3.61	4.90	No	i_{300}, V_{300}, T	NA	1-4	2	36	2	relu, linear	0, 0.1, 0.2, 0.3	relu, linear, sigmoid	20	2	20	US06	300
3	1.94	2.44	NA	No	i_{300}, V_{300}, T	4-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	5	2	5	US06	300
4	1.80	2.27	2.40	No	i_{400}, V_{400}, T	4-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	600
5	1.13	1.51	1.70	No	i_{1200}, V_{1200}, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
6	0.95	1.28	1.41	No	i_{2400}, V_{2400}, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	2400
7	0.85	1.15	1.26	No	i_{4800}, V_{4800}, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	4800
22	1.18	1.87	2.23	No	i_{9600}, V_{9600}, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	9600
23	2.27	3.31	3.33	No	i_{4800}, V_{4800}, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	4800/2400
24	1.20	1.60	1.85	No	i_{1200}, V_{1200}, V, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
8	1.11	1.51	1.20	No	i_{1200}, V_{1200}, i, T	2-32	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
9	1.11	1.49	1.60	No	$i_{1200}, V_{1200}, i, V, T$	2-36	1-3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
10	1.78	2.53	4.50	No	i_{1200}, V_{1200}, T	2-36	3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
11	1.55	2.23	2.45	No	i_{1200}, V_{1200}, T	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
12	1.65	2.23	1.89	No	$i_{1200}, V_{1200}, i, V, T$	2-36	0	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
13	1.05	2.26	2.60	No	$i_{1200}, V_{1200}, i, V, T$	2-36	1	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
14	1.62	2.26	2.34	No	$i_{1200}, V_{1200}, i, V, T$	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
15	1.89	2.60	4.30	No	$i_{1200}, V_{1200}, i, V, T$	2-36	3	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
16	1.92	2.65	3.00	No	$i_{1200}, V_{1200}, i, V, T$	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
17	1.95	2.78	3.17	No	$i_{1200}, V_{1200}, i, V, T$	2-36	1	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
18	2.04	2.91	3.23	No	i_{1200}, V_{1200}, T	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
19	2.04	2.83	3.09	Yes	$i_{1200}, V_{1200}, i, V, T$	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
20	2.27	3.10	3.26	Yes	i_{1200}, V_{1200}, T	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	1200
21	1.16	1.43	1.51	Yes**	i_{4800}, V_{4800}, T	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	4800
22	1.01	1.20	1.69	Yes**	$i_{4800}, V_{4800}, i, V, T$	2-36	2	2	36	2	relu	0, 0.1, 0.2, 0.3	relu	20	2	5	US06	4800

* Data set at -20°C was not used for training and validation.

** Validated without CCCV for a fail comparison

C.0.2 ΔT Average scores - LG Data Set

Table C.2: Scores of all test cycles, all temperatures for the two benchmark cases and the averaged temperature difference.

Drive cycle	Temperatures deg C	Benchmark Transfer learning		Benchmark w/ base structure		Delta T average	
		MAE SoC [%]	RMSE SoC [%]	MAE SoC [%]	RMSE SoC [%]	MAE SoC [%]	RMSE SoC [%]
HWFET	-20	1.25	2.14	1.93	2.86	1.61	2.22
	-10	1.28	1.84	1.53	1.97	1.24	1.56
	0	1.25	1.68	1.33	1.65	0.89	1.21
	10	0.60	0.78	0.73	0.95	0.50	0.66
	25	0.60	0.78	0.73	0.95	0.50	0.66
	40	0.52	0.66	0.84	1.09	0.43	0.53
LA92	-20	1.79	1.94	1.27	1.50	1.62	1.90
	-10	1.29	1.48	1.18	1.47	1.28	1.51
	0	0.77	0.96	0.77	0.96	0.75	0.93
	10	0.73	0.93	0.95	1.16	0.81	1.10
	25	0.80	1.03	0.82	1.13	0.69	0.86
	40	0.64	0.82	0.83	1.14	0.58	0.78
UDDS	-20	2.46	2.77	1.74	2.03	2.26	2.60
	-10	2.01	2.31	1.19	1.57	1.96	2.23
	0	1.10	1.35	1.12	1.33	1.08	1.37
	10	0.92	1.11	0.76	0.97	0.79	0.94
	25	0.60	0.79	0.63	0.82	0.56	0.69
	40	0.53	0.72	0.65	0.92	0.58	0.91
US06	-20	2.53	2.76	1.89	2.25	2.61	3.00
	-10	1.19	1.40	1.02	1.29	1.26	1.75
	0	1.17	1.54	1.36	1.66	0.90	1.23
	10	0.89	1.10	1.13	1.42	0.89	1.25
	25	0.75	0.94	0.86	1.07	0.73	0.92
	40	0.80	0.98	1.15	1.38	0.73	0.90

Appendix **D**

Standard Deviation of Gaussian Noise

This appendix is a table with the largest standard deviations of the Gaussian noise of all the drive cycles and charge cycles at all temperatures. The standard deviation corresponds to the augmented data set A as stated in Table 6.2.

Table D.1: Standard deviation of the largest noise for current, voltage and battery temperature. Table with the standard deviations for the noise for each drive cycle at each temperature.

Temperature	Name of drive cycle	Drive cycle (discharge)			corresponding CCCV charging cycle		
		std i	V	T	std i	V	T
0 °C	Mixed2	0.227	0.003	0.06	0.115	0.004	1.076
	Mixed4	0.241	0.003	0.073	0.115	0.003	1.075
	Mixed5	0.233	0.003	0.057	0.115	0.004	1.071
	Mixed6	0.262	0.003	0.09	0.115	0.004	1.07
	Mixed7	0.26	0.003	0.074	0.115	0.004	1.069
	Mixed8	0.233	0.003	0.068	0.115	0.004	1.068
	UDDS	0.152	0.002	0.019	0.115	0.004	1.071
	HWFET	0.154	0.003	0.038	0.115	0.004	1.074
	LA92	0.229	0.003	0.033	0.115	0.004	1.068
	US06	0.39	0.003	0.095	0.116	0.004	1.064
Mixed1	0.231	0.003	0.068	0.0	0.0	0.0	
10 °C	US06	0.414	0.003	0.067	0.118	0.004	0.66
	Mixed1	0.257	0.003	0.07	0.118	0.004	0.653
	Mixed2	0.241	0.003	0.047	0.118	0.004	0.66
	Mixed4	0.256	0.003	0.056	0.116	0.004	0.655
	Mixed5	0.253	0.002	0.051	0.116	0.004	0.658
	Mixed6	0.253	0.003	0.052	0.116	0.004	0.656
	Mixed7	0.262	0.002	0.043	0.116	0.004	0.657
	Mixed8	0.234	0.003	0.04	0.116	0.004	0.658
	UDDS	0.155	0.002	0.014	0.117	0.004	0.658
	HWFET	0.16	0.003	0.03	0.116	0.004	0.654
LA92	0.246	0.003	0.027	0.0	0.0	0.0	
25 °C	UDDS	0.155	0.003	0.016	0.12	0.004	0.042
	LA92	0.241	0.003	0.022	0.12	0.004	0.043
	US06	0.405	0.003	0.053	0.12	0.004	0.042
	Mixed1	0.252	0.003	0.047	0.12	0.004	0.04
	Mixed2	0.245	0.003	0.044	0.12	0.004	0.04
	Mixed3	0.265	0.003	0.041	0.119	0.004	0.043
	Mixed4	0.256	0.003	0.042	0.119	0.004	0.042
	Mixed5	0.263	0.003	0.05	0.119	0.004	0.045
	Mixed6	0.244	0.003	0.035	0.119	0.004	0.041
	Mixed7	0.253	0.003	0.031	0.119	0.004	0.043
Mixed8	0.233	0.003	0.032	0.119	0.004	0.041	
40 °C	UDDS	0.153	0.003	0.011	0.119	0.004	0.62
	HWFET	0.158	0.003	0.02	0.118	0.004	0.621
	LA92	0.24	0.003	0.017	0.119	0.004	0.619
	US06	0.401	0.003	0.043	0.118	0.004	0.618
	Mixed1	0.25	0.003	0.036	0.119	0.004	0.62
	Mixed2	0.242	0.003	0.031	0.118	0.004	0.616
	Mixed3	0.261	0.002	0.03	0.0	0.0	0.0
	Mixed4	0.133	0.0	0.008	0.031	0.0	0.022
	Mixed5	0.178	0.001	0.017	0.031	0.0	0.045
	Mixed6	0.212	0.001	0.054	0.03	0.0	0.027
Mixed7	0.207	0.001	0.033	0.03	0.0	0.02	
Mixed8	0.169	0.0	0.027	0.03	0.0	0.019	
-10 °C	UDDS	0.156	0.002	0.028	0.112	0.003	1.463
	HWFET	0.161	0.003	0.059	0.111	0.003	1.474
	LA92	0.233	0.003	0.05	0.112	0.003	1.465
	US06	0.381	0.003	0.118	0.111	0.003	1.46
	Mixed1	0.222	0.003	0.08	0.111	0.003	1.465
	Mixed2	0.232	0.003	0.09	0.111	0.003	1.473
	Mixed4	0.234	0.003	0.068	0.111	0.003	1.462
	Mixed5	0.238	0.003	0.079	0.111	0.003	1.464
	Mixed3	0.244	0.003	0.082	0.11	0.003	1.47
	Mixed6	0.251	0.003	0.092	0.111	0.003	1.468
Mixed7	0.254	0.003	0.097	0.111	0.003	1.455	
Mixed8	0.229	0.003	0.093	0.111	0.003	1.464	
-20 °C	UDDS	0.144	0.002	0.043	0.099	0.002	1.854
	HWFET	0.158	0.003	0.08	0.099	0.002	1.86
	LA92	0.201	0.003	0.062	0.0	0.0	0.0
	US06	0.269	0.003	0.125	0.099	0.002	1.864
	Mixed1	0.198	0.003	0.105	0.098	0.002	1.86
	Mixed2	0.2	0.003	0.118	0.098	0.002	1.867
	Mixed3	0.203	0.002	0.093	0.097	0.002	1.869
	Mixed4	0.201	0.003	0.086	0.097	0.002	1.865
	Mixed5	0.197	0.003	0.111	0.097	0.002	1.869
	Mixed6	0.195	0.003	0.096	0.097	0.002	1.869
Mixed7	0.204	0.003	0.088	0.097	0.002	1.867	
Mixed8	0.19	0.003	0.093	0.097	0.002	1.871	

Appendix E

Boruta Shap Feature Extraction

First, all the parameters of the data set was inputted. The boruta shap algorithm noticed the correlation between SoC and Ah. Since the target SoC is calculated by Coulomb counting, this was however not very informative, and the Ah cannot be used as an input parameter, since it is too closely related to the target value.

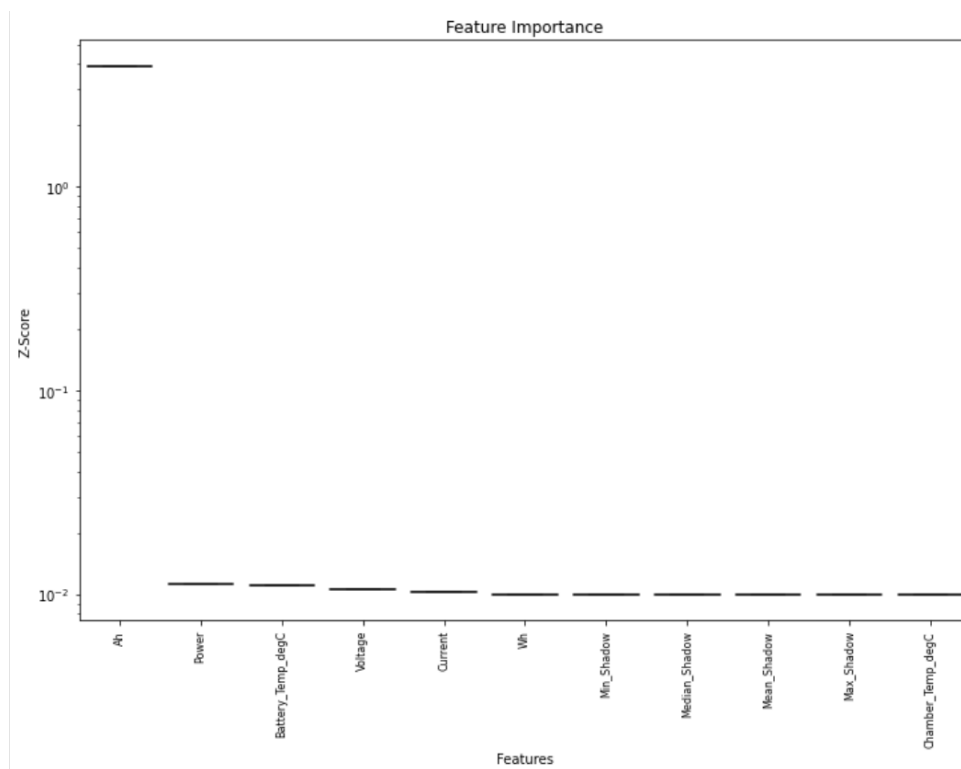


Figure E.1: First attempt to extract features with boruta shap.

In the next trial with boruta shap, the Ah were taken away.

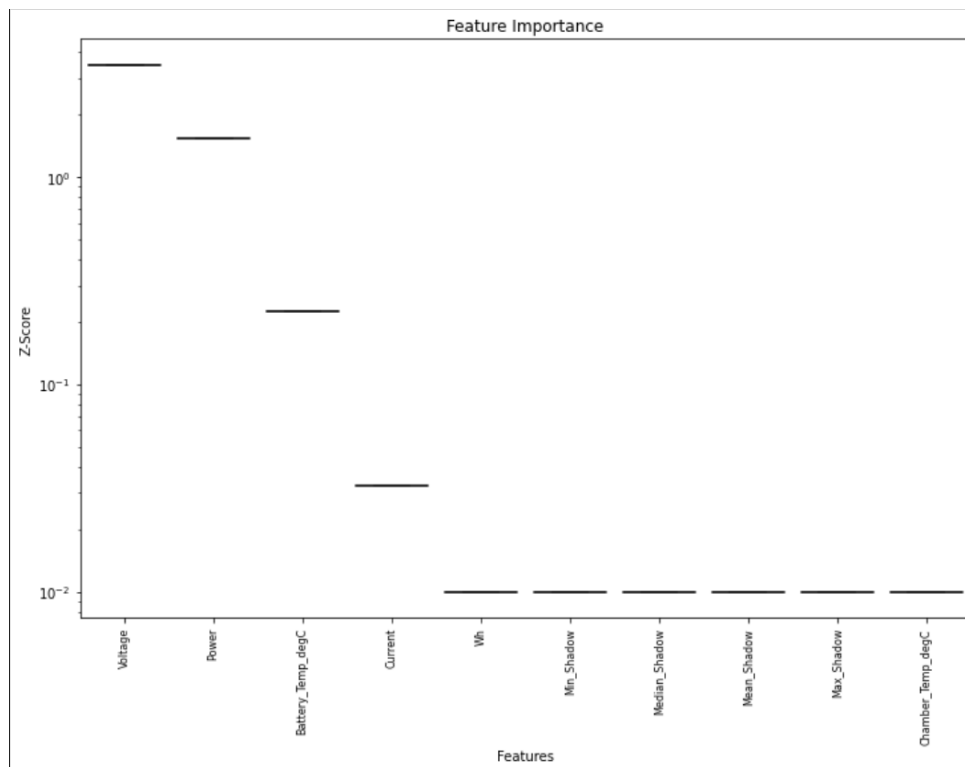


Figure E.2: Second attempt of Boruta shap. The Ah were taken away to find other correlations than between Ah and SoC.

In the last trial, averaged parameters were utilized. It was seen that the parameters averaged over a large time frame, correlated most with the SoC. A concern was drawn that a large window for averaging the current, would be too similar to the Ah.

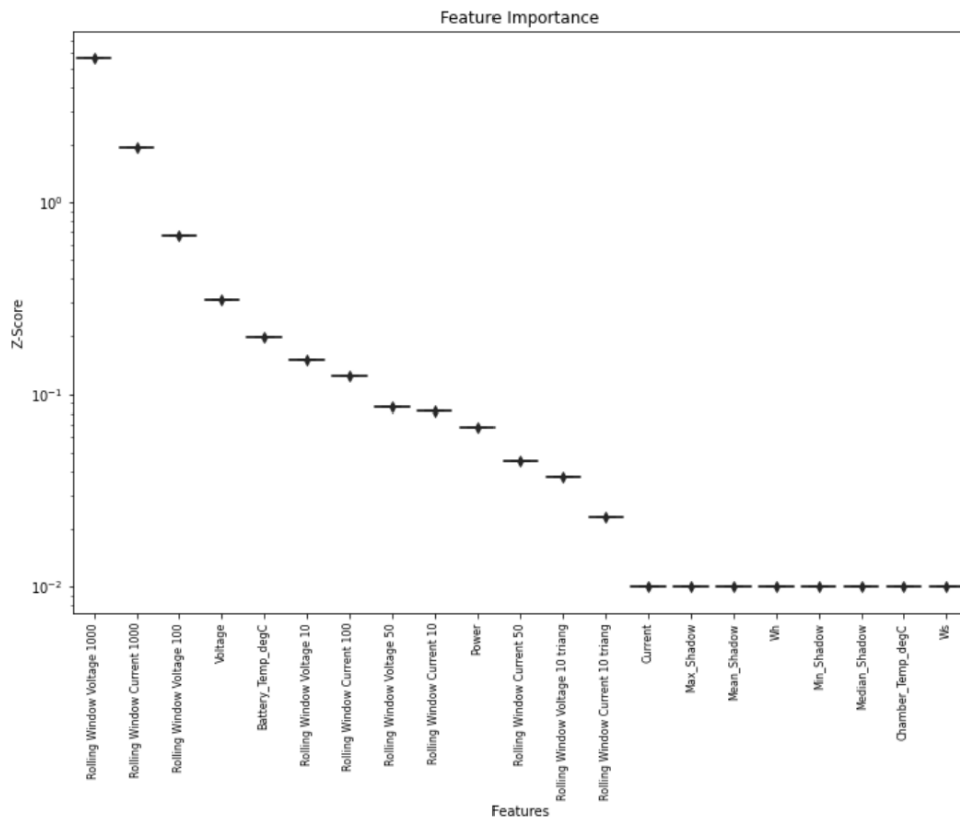


Figure E.3: Third attempt of Boruta shap. Averaging values were included.

Appendix **F**

Risk Assessment

NTNU  HMS	Risikovurdering	Utarbeidet av	Nummer	Dato	
		HMS-avd.	HMSRV2603	22.03.2011	
		Godkjent av		Erstatter	
		Rektor		01.12.2006	

Sannsynlighet vurderes etter følgende kriterier:

Svært liten 1	Liten 2	Middels 3	Stor 4	Svært stor 5
1 gang pr 50 år eller sjeldnere	1 gang pr 10 år eller sjeldnere	1 gang pr år eller sjeldnere	1 gang pr måned eller sjeldnere	Skjer ukentlig

Konsekvens vurderes etter følgende kriterier:



Gradering	Menneske	Ytre miljø Vann, jord og luft	Øk/materiell	Omdømme
E Svært Alvorlig	Død	Svært langvarig og ikke reversibel skade	Drifts- eller aktivitetsstans >1 år.	Troverdighet og respekt betydelig og varig svekket
D Alvorlig	Alvorlig personskade. Mulig uførhet.	Langvarig skade. Lang restitusjonstid	Driftsstans > ½ år Aktivitetsstans i opp til 1 år	Troverdighet og respekt betydelig svekket
C Moderat	Alvorlig personskade.	Mindre skade og lang restitusjonstid	Drifts- eller aktivitetsstans < 1 mnd	Troverdighet og respekt svekket
B Liten	Skade som krever medisinsk behandling	Mindre skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1 uke	Negativ påvirkning på troverdighet og respekt
A Svært liten	Skade som krever førstehjelp	Ubetydelig skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1 dag	Liten påvirkning på troverdighet og respekt

Risikoverdi = Sannsynlighet x Konsekvens

Beregn risikoverdi for Menneske. Enheten vurderer selv om de i tillegg vil beregne risikoverdi for Ytre miljø, Økonomi/materiell og Omdømme. I så fall beregnes disse hver for seg.

Til kolonnen "Kommentarer/status, forslag til forebyggende og korrigerende tiltak":

Tiltak kan påvirke både sannsynlighet og konsekvens. Prioriter tiltak som kan forhindre at hendelsen inntreffer, dvs. sannsynlighetsreducerende tiltak foran skjerpet beredskap, dvs. konsekvensreducerende tiltak.

NTNU	Risikomatrise	utarbeidet av	Nummer	Dato	
		HMS-avd.	HMSRV2604	08.03.2010	
HMS/KS		godkjent av		Erstatter	
		Rektor		09.02.2010	

MATRISSE FOR RISIKOVURDERINGER ved NTNU

KONSEKVENSENS	Svært alvorlig	E1	E2	E3	E4	E5
	Alvorlig	D1	D2	D3	D4	D5
	Moderat	C1	C2	C3	C4	C5
	Liten	B1	B2	B3	B4	B5
	Svært liten	A1	A2	A3	A4	A5
		Svært liten	Liten	Middels	Stor	Svært stor
		SANNSYNLIGHET				

Prinsipp over akseptkriterium. Forklaring av fargene som er brukt i risikomatrisen.

Farge	Beskrivelse
Rød	Uakseptabel risiko. Tiltak skal gjennomføres for å redusere risikoen.
Gul	Vurderingsområde. Tiltak skal vurderes.
Grønn	Akseptabel risiko. Tiltak kan vurderes ut fra andre hensyn.