A Data-Driven Lithium-Ion Concentration Estimator for Electrode Solid Phase in Battery Models.

by

Le Cai

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
August 5, 2023

Keywords: LSTM, Lithium-ion Battery, Concentration

Approved by

Xiao Qin, Chair, Alumni Professor of Computer Science and Software Engineering
Song-Yul Choe, Co-Chair, William B. and Elizabeth Reed Professor of Mechanical Engineering
Cheryl Seals, Charles W. Barkley Professor of Computer Science and Software Engineering

Abstract

This thesis introduces a data-driven approach for lithium-ion concentration estimation in battery cells, leveraging a deep learning model based on Long Short-Term Memory (LSTM) networks. As lithium-ion batteries gain widespread adoption, particularly in electric vehicles and renewable energy storage, accurate and efficient estimation of lithium-ion concentration is pivotal. Existing methods like the polynomial Reduced Order Models (ROMs), although popular, face trade-offs between computational efficiency and accuracy.

In this work, we propose an LSTM-based estimator designed to achieve high accuracy without compromising computational speed. The LSTM architecture's depth allows it to capture intricate relationships within the cell. The estimator's performance was rigorously evaluated and compared against a polynomial ROM model. Our estimator exhibited remarkable accuracy in short term estimation, achieving a Mean Absolute Percentage Error (MAPE) of 1.09% at 1000s and 3.27% at 2000s for estimation of surface concentration for single particle anode. For estimation of average concentration for single particle anode, the MAPE was 1.27% at 1000s and 2.55% at 2000s. The estimator's swift computation time provides potential for real-time applications.

Overall, this thesis provides a comprehensive exploration of LSTM-based lithium-ion concentration estimation, from model design and implementation to performance evaluation. This work underscores the potential of data-driven approaches in battery state estimation, opening up new possibilities for real-time applications and improved battery management systems.

Acknowledgments

I am profoundly grateful to my advisor, Dr. Song-Yul Choe, for presenting me with this research opportunity. His invaluable guidance, unwavering support, and constructive suggestions have been instrumental in shaping and steering my research to fruition. His meticulous attention to detail and dedication in reviewing and refining my work have been paramount in completing my research program.

I wish to extend my sincere appreciation to Dr. Xiao Qin and Dr. Cheryl Seals for their participation in my committee. Their invaluable insights, suggestions, and rigorous critique have greatly enriched my work, and I am grateful for their time and expertise.

My gratitude also extends to my colleagues Xiaoniu Du and Kyunjin Yu. Their assistance, guidance, and shared wisdom throughout my thesis journey have been a source of both learning and encouragement.

I would also like to thank my friends, Xiaoyun Yang, Xingxing Zhang, Chi Xu, Yaoxuan Luan, Lingxiao Wang for their help and suggestions.

Finally, I would like to express my heartfelt appreciation to my partner Yuxin Cai, my dogs Offer Cai, Roubaix Cai, and my family. Their continuous support, encouragement, and belief in my abilities have been a beacon of light in my academic journey, providing the motivation needed to surmount any challenges that came my way. This accomplishment would not have been possible without them.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BMS	Battery Management System
FOM	Full Order Model
OCV	Open Circuit Voltage
ROM	Reduce Order Model
SOC	State of Charge
SOH	State of Health

Greek symbols

α	Transfer coefficient of reaction
δ	Thickness (mm)
η	Reaction overpotential of electrode (V)
κ	Ionic conductivity (S $cm^{-}1$)
σ	Conductivity (S $cm^{-}1$)
ε	Volume fraction of a porous medium or strain
φ	Electric potential (V)

Electrochemical Model Subscripts and Superscripts

$+$	Positive electrode (cathode)

$-$	Negative electrode (anode)
a	Anodic
avg	Average value
c	Cathodic
e	Electrolyte phase
eff	Effective
eq	Equilibrium
Li	Lithium ion
LiP	lithium plating
max	Maximum
s	Solid phase
$surf$	Electrode particle surface

Electrochemical Model

a_a	Anodic intercalation factor
a_c	Cathodic intercalation factor
a_s, Li	Specific surface area of lithium deposition reaction at electrode (cm^-1)
$a_s, side$	Specific surface area of side reaction at electrode (cm^-1)
a_s	Surface area at electrode
C_e	Ion concentration in electrolyte
C_s	Ion concentration in electrode

D_e	Diffusion coefficient in electrolyte
D_s	Diffusion coefficient in electrode
F	Faraday constant
I	Current of the cell (A)
i_0	Exchange current density (A cm^2)
j	Reaction rate
L	Thickness of microcell (cm)
l	Coordinate along the thickness of microcell
R	Resistance (Ω cm^2) or universal gas constant (8.3143 J mol^-1 K^-1)
r	Coordinate along the radius of electrode particle
R_s	Radius of spherical electrode particle (cm)
T	Cell temperature (K)
t	Time
V_t	Terminal voltage of cell (V)
x	Stoichiometric number of the anode
y	Stoichiometric number of the cathode

Chapter 1

Introduction

1.1 Background

Lithium-ion Batteries (LIBs) have emerged as a dominant form of energy storage, it facilitated a modern societal paradigm shift towards a more sustainable energy and technology integrated future. The explosive growth of the battery market fueled by the urgent demand for energy storage in various applications have primarily been powered by advancements in LIB technology.

LIBs power a wide variety of applications that are integral to our modern lives. The personal electronics sector, which includes smartphones, laptops, tablets, and wearable devices, is perhaps the most recognizable user of the technology. LIBs have also catalyzed the EV revolution, solving range and weight issues that troubled earlier implementation of electric vehicles. Furthermore, LIBs play an increasingly critical role in large-scale grid energy storage systems, stabilizing power grids by storing excess electricity during off-peak periods for use during periods of high demand. This capability has also facilitated the integration of renewable energy sources into the grid, which often suffer from intermittency issues.

As the popularity of LIBs increasing, it becomes apparent that the management of the energy storage systems is equally crucial. Battery Management Systems (BMS) are essential in maximizing the performance, lifespan and safety of LIBs, a BMS is an embedded system comprises of purpose-built electronics and custom designed algorithms. BMS monitor and manage various parameters during operation such as voltage, current and temperature and estimates battery states such as state-of-charge (SOC) and state-of-health(SOH).

BMS is designed with several objectives in mind. First, it need to ensure the safety of the user by constantly monitor the operating conditions of battery and respond if unsafe

operating conditions are detected. Secondly, BMS should protect the cells of battery from data in the case of failure or abuse. Thirdly, BMS with its internal algorithms should regulate and prolong the lifespan of the battery and maintain it in a state which can fulfill the designed requirements. Lastly, BMS can communicate with other devices in the application to provide the capability of the battery.

1.1.1 NMC Lithium-ion Cell

A lithium-ion cell typically consists of several key components; positive electrode, negative electrode, electrolyte, separator, and current collectors.

Positive electrode is one of the two electrodes of a Li-ion cell, it acts as the source of lithium-ions and the composition of its compound determines the capacity and voltage of the battery. In 1980, John B. Goodenough and Koichi Mizushima discovered the potential to use Lithium Cobalt Oxide ($LiCoO_2$) as an intercalation electrode [4]. Lithium Cobalt Oxide, also known as LCO, is commonly used in portable electronic devices, but also suffers from some issues; the structure of LCO contains cobalt, which is rare, expensive, and toxic, LCO also have a shorter lifespan of the cell. Advancements were made and a combination of Nickel, Manganese, Cobalt (NMC) has become one the of most successful Li-ion systems. Nickel offers high specific energy but have poor stability, manganese can form a cubic spinel structure to achieve low internal resistance but offers low specific energy, cobalt stabilizes nickel but is in limited supply, by combining the metals, the strengths of the metals are enhanced. NMC electrode materials are composed of spherical particles, resulting in a powder form. This characteristic is advantageous because it maximizes the available surface area, thereby enhancing the potential for electrochemical reactions.

The other electrode of a Li-ion cell is the negative electrode, graphite (C_6) is used in the majority of commercial Li-ion batteries for almost 30 years[5]. Graphite has graphene layers of C_6 structures that are tightly bonded, the graphene layers are loosely stacked which allowed lithium to intercalate between layers.

Electrolyte in Li-ion battery are made with non-aqueous organic solvents, additives, and lithium salt. It acts as the media that conducts lithium-ion between electrodes. The most used salt is $LiPF_6$ and some common solvents include ethylene carbonate (EC), dimethyl carbonate (DMC), ethyl methyl carbonate (EMC), and diethyl carbonate (DEC). Apart from electrolyte, another important component of Li-ion battery is separator.

The separator is a permeable membrane designed with appropriately sized holes to facilitate the passage of lithium-ions while preventing direct contact between the negative and positive electrode particles. Additionally, it serves as an electronic insulator, ensuring that electrical currents flow only through the desired paths within the battery.

The current collectors are the interfaces between electrodes and the external circuit. Positive electrode (high potential) uses aluminum foil, and the negative electrode (low potential) uses copper foil. Modern Li-ion batteries coats active electrode materials on both sides of metallic current collector foils which conducts the current into and out of the cell.

Lithium-ion battery rely on a key process known as intercalation. Lithium is stored within the electrodes, during discharge, lithium exits the surface of negative electrode particles, gives up an electron and becoming Li^+ in the electrolyte. Lithium will diffuse outward from center of negative electrode particles to equalize concentration level with in electrode particle. The electron travels from current collector, through external circuit and positive electrode current collector into cathode, where Li^+ joins with the electron, becoming lithium and inserts into the positive electrode particle from its surface, and in both electrodes, diffusion equalizes the internal concentration of lithium over time.

As described above, during discharging, the lithium-ions flow from negative electrode through electrolyte and intercalate into positive electrode[6], this process is completely reversible, during charging lithium-ion flow from positive electrode and intercalate into negative electrode. This is also depicted in Figure 1.1. The internal chemical reaction of NMC lithium cell can be described as follows:

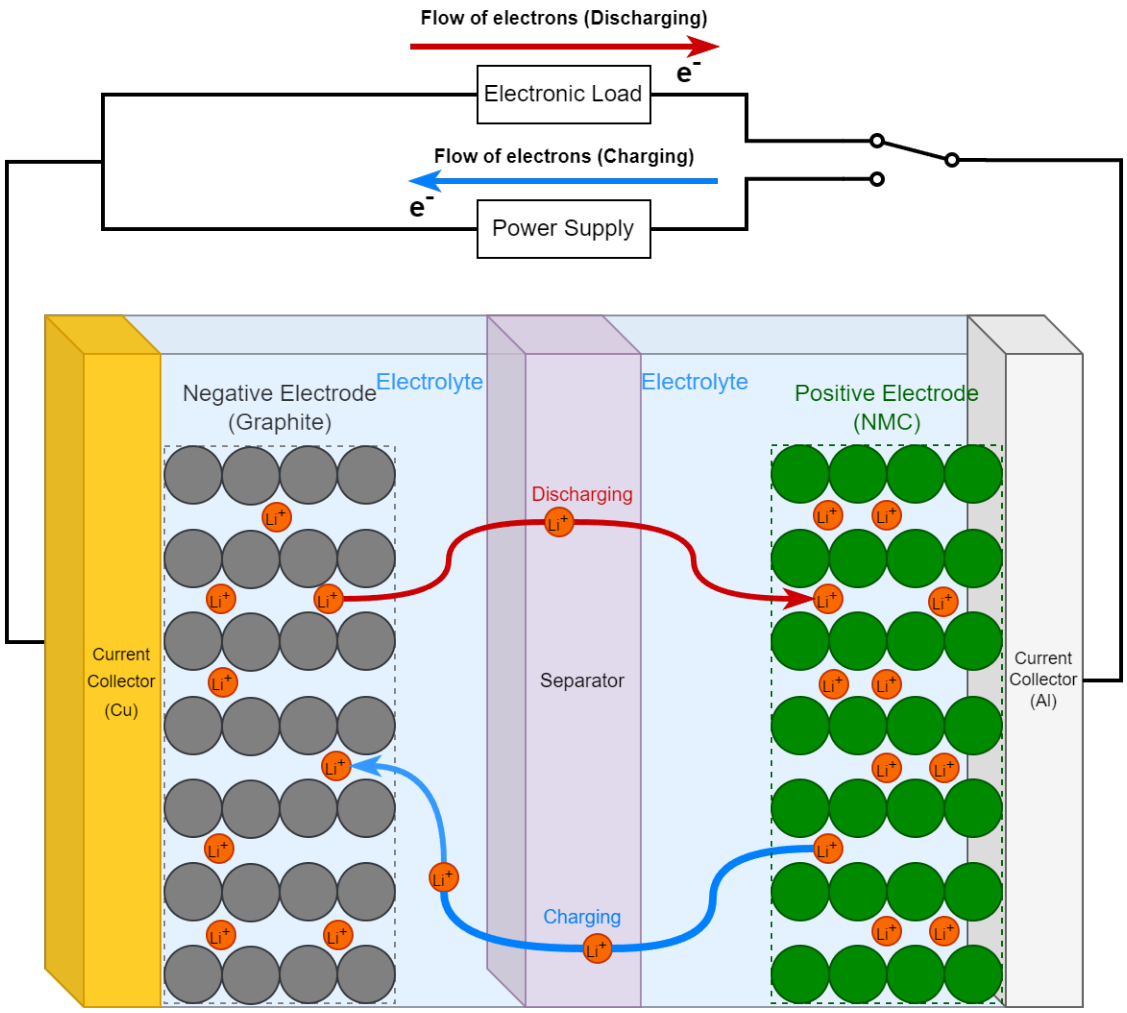

Figure 1.1: Schematic diagram of a Lithium-ion cell.

- Negative Electrode Reaction

$$Li_xC_6 \rightleftharpoons C_6 + xLi^+ + xe^- \tag{1.1}$$

- Positive Electrode Reaction

$$Li_{(1-x)}MO_2 + xLi^+ + xe^- \rightleftharpoons LiMO_2 \tag{1.2}$$

- Overall Reaction

$$Li_xC_6 + Li_{(1-x)}MO_2 \rightleftharpoons C_6 + LiMO_2 \tag{1.3}$$

1.1.2 Literature Review

Lithium-ion batteries with their high energy density and long cycle life, have become the power source of choice in numerous applications ranging from portable electronics to electric vehicles and grid storage. However, their complex behavior and the need for efficient and safe operation presents a variety of challenges that need to be tackled.

Traditionally, battery management systems (BMS) have employed physics-based models such as the equivalent circuit model (ECM) and electrochemical models to predict battery behavior. However, these models have their limitations. ECMs, for instance, do not adequately capture the aging phenomena and have limited accuracy under dynamic conditions. Electrochemical models, while detailed, are computationally intensive and often impractical for real-time applications[7].

Recently, machine learning (ML) techniques have emerged as a promising tool for tackling these challenges. Machine learning can model the nonlinear relationships and capture the temporal dynamics in the battery behavior, making them an attractive option for battery state estimation and health prognosis.

Many efforts also into using machine learning methods to approximate lithium-ion battery dynamics. Physics based electrochemical model is often used to describe the dynamic of lithium-ion battery but it is computationally expensive.

Mckay et al.[8] proposed a surrogate modeling approach that uses synthetic data generated by SPM model to train a deep neural network (DNN) to approximate the battery dynamics. The DNN model is trained to predict future battery behavior over 100-second horizon, the prediction consists of terminal voltage, concentration profiles in electrodes and whether the battery failed during time window. The DNN achieved a MAPE of 1.28% in testing dataset, and when used in multi step prediction, the model showed signs of error accumulation.

Chun et al.[9] proposed a real-time parameter estimation of an electrochemical lithium-ion battery model using LSTM. Different neural networks were tested and LSTM was selected

based on accuracy. The model is first trained on synthetic data to learn the pseudo-two-dimensional model dynamics, the model is given voltage, current, temperature and state of charge to estimate cathode and anode surface areas, cathode and anode conductivities, SEI layer thickness and normalized available capacity. The model perform estimation based on 10-seconds data segments. The proposed method achieved a RMSE of 0.430% on testing dataset, which is considered an accurate estimation of parameters.

Li et al.[10] proposed a hybrid state estimation model, where an electrochemical-thermal model is developed and experimentally verified and used to generate a large quantity of data under various operating conditions. The generated data is used to train a deep neural network to perform state estimations of li-ion battery. The model achieved a maximum error of 2.93% for all estimated states under new ambient temperatures.

One of the key applications of ML in battery management has been in state-of-charge (SOC) estimation. For instance, How et al.[11] utilized a deep neural network (DNN) approach to estimate SOC under different EV drive cycles, with sufficient number of hidden layers the DNN was able to generalize the trend of SOC estimation from one drive cycle to another.

However, How et al. did not consider the temporal relations of input data to the DNN, each point was treated as an independent individual. Other networks, such as recurrent neural network (RNN) along with its variants Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) have memory information throughout the neural network to retain the previous information are also used on this application. Lipu et al.[12] and Yang el al.[13] used deep RNN and RNN-GRU to predict the SOC of lithium-ion batteries with excellent results.

Similarly, convolutional neural network (CNN), often used in computer vision applications can be adapted to perform SOC estimation. CNNs consists of convolutional layer, pooling layer and fully connected layer, where convolutional layer extracts feature from input, followed by pooling layer to down sample the data, the data is then processed by the fully

connected layer to produce final result. Bhattacharjee et al.[14] proposed a one-dimensional CNN based SOC estimation algorithm with transfer learning mechanism to improve generalization capability and yield acceptable accuracy estimations.

Machine learning methods have also been applied in predicting the state-of-health (SOH) and Remaining Useful Life (RUL) of batteries.

Feed-forward Neural Network (FNN) is a type of artificial neural network (ANN), the information only travel in one direction from its input layer to hidden layer to output layer. FNN has been implemented to estimate battery SOH in various works. Kashkooli et al.[15] used calendar aging data to estimate battery SOH and achieved a RMSE of 1.17% in his work.

Recurrent neural networks (RNNs) have the ability to process and store past information, which is crucial to identify the correlation between battery features. As mentioned above, LSTM, a promising variant of RNN that excels at capture long term and short term temporal relations within data is proven to be useful in SOH estimation. Choi et al.[16] proposed a capacity estimation framework for Li-ion battery based on FNN, CNN and LSTM and validated their performance with NASA lithium-ion battery dataset, the results shown LSTM has very accurate capacity estimations even when abnormal deviations of charging profile is present due to its internal memory cell storing long term information, where CNN and FNN fails and delivered low estimation results based on abnormal deviations. In addition, they demonstrated that by using multi-channel data instead of single-channel data, the MAPE improved by 25% for LSTM.

Like SOC estimation applications, convolutional neural network (CNN) is also used in SOH estimation applications due to its ability to extract and select feature from data. Shen et al.[17] proposed a deep convolutional neural network model with 5 convolution layers and 3 fully connected layers. The proposed model achieved a MAPE of 1.477% overall based on NASA datasets. In the work of Crocioni et al.[18] A CNN GRU model is selected based on its low estimation error using NASA dataset, the model is then adapted onto microcontrollers

via STM32Cube.AI software. The comparison detected no change in model's performance, which proved the feasibility of CNN in online SOH estimation applications.

A summary of machine learning lithium-ion battery applications is provided in Table 1.1.

Year	Journal	Method	Application	Remarks	Reference
Application: Battery Parameter Estimation					
2020	IEEE Access	LSTM	Real-time parameter estimation	Estimations include two solid particle surface areas (cathode and anode), two solid particle conductivities (cathode and anode), SEI layer thickness, and normalized available capacity.	[9]
2021	Journal of Power Sources	LSTM	Battery state estimation	2-GLSTM (Variant of LSTM) is used to estimate battery states based on data generated by physics-based model	[10]
2021	IFAC-PapersOnLine	DNN	Battery states and failure estimation	DNN considers a 100s prediction horizon, recursive multi-step prediction show increase in error	[8]
Application: SOC (SOC)					
2020	IEEE Transactions on Industry Applications	DNN	SOC estimation	Model with different hidden layer number is trained and tested on DST drive cycle, the best performing model is tested on FUDS, BJDST and US06 drive cycle. No temporal relations is considered in this work.	[11]
2019	Journal of The Electrochemical Society	FNN	SOC estimation of calendar aged li-ion cell	Trained 2 DNNs to account for charge/discharge asymmetry of li-ion cell. RMSE 1.17% and 1.81% for discharge and charge.	[15]
2019	2019 IEEE Industry Applications Society Annual Meeting	RNN	SOC estimation	Deep RNN with Firefly algorithm to find optimal hyperparameters of the network.	[12]
2019	Energy	GRU-RNN	SOC estimation	GRU-RNN SOC estimation results with different epoch numbers and different architecture were presented, the RMSE of SOC estimation were less than 2.5% for NMC cell and less than 3.5% for LFP battery.	[13]
2017	2017 American Control Conference (ACC)	RNN	SOC estimation based on data from SPM	2 Elman networks were implemented in this test, the more complex network consistently outperforms simpler network, the accuracy are comparable to SPMe.	[1]
2021	IEEE Transactions on Industry Applications	GRU-RNN	SOC estimation	Two-hidden layer stacked GRU model is used in this work. A performance comparison of RNN variants, CNN variants and MLP is also performed, RNN variants performed best followed by CNN variants and MLP came last.	[19]
2022	Applied Energy	LSTM	SOC estimation	Two LSTM were used in CDTL, source LSTM is trained on SOC data from historical cells while target LSTM is trained using limited available SOC data from target cell. This allowed transfer of knowledge from source LSTM to target LSTM using controllable multiple domain adaption with adaptive regularization.	[20]
2021	IEEE Transactions on Vehicular Technology	CNN	SOC estimation with 1D CNN with transfer learning	CNN to be trained on LG dataset (target) is initialized with weights from a trained CNN trained with similar conditioned Panasonic data. Window size for CNN is 500s.	[14]
2021	IEEE Transactions on Power Electronics	CNN	SOC estimation	FCN proposed in this work apply convolution operations across time axis t, The model scored a RMSE of 0.85%, out performing FCN without lr optimization, GRU, LSTM and CNN.	[21]
2020	Journal of Power Sources	Hybrid	Co-estimation of SOC and state of temperature	Hybrid model contains an electrochemical thermal sub-model consists of a simplified SP model with a lumped thermal model. FNN is used to increase accuracy of ETSM in high C and extreme temperature environment, final SOC and SOT is calculated by an unscented Kalman filter.	[22]
Application: SOH (SOH) and RUL (RUL)					
2019	Journal of The Electrochemical Society	FNN	FNN SOH estimation of calendar aged li-ion cell	DNN takes in time in months, storage temperature, storage condition (fully charged/discharged) and output SOH over time. RMSE of 1.67% for SOH estimation.	[15]
2016	Applied Energy	FNN	Real-time SOH estimation for EV	Offline learning is used in this work where data gathered from cycler is used to train model on a server and apply optimized parameters to online SOH estimation applications (BMS)	[23]
2017	Energy Procedia	FNN	SOH estimation of li-ion battery in EVs	The input for FNN model in this work are SOC, ohmic resistance R_0, polarization resistance R_p and polarization capacity C_p. The use of first-order ECM parameters is different from other implementations, this helped with the interpretability of NN.	[24]
2019	Energies	DNN	SOH and RUL estimation by data driven DNN	A three layer DNN is implemented in this work to estimate SOH and RUL, a comparison is done against k_NN, LR, SVM and ANN, DNN had the best performance, but other networks like CNN and RNN are excluded from this comparison.	[25]
2022	Applied Energy	FNN	Health-Conscious vehicle battery state estimation	A cloud-based health-conscious transfer learning battery model is proposed, vehicle operation data is collected by BMS and uploaded to cloud used for training. The optimized parameters is then transferred to BMS via communication network. This model can reduce prediction error and allow deployment at early stages where large scale data is not available.	[26]
2021	International Journal of Energy Research	LSTM	Capacity estimation	FNN, CNN and LSTM models are trained and compared in this work, the best performance over test sets is obtained by LSTM followed by RNN and CNN in order.	[27]
2019	IEEE Access	LSTM	Capacity estimation	A baseline FNN consists of 10 neurons and a more complex FNN-2 with 40 neurons are implemented and tested, FNN-2 Consistently outperformed baseline FNN. FNN-2 also outperformed CNN-1 and CNN-2 in this work's comparison. LSTM has the best performance out of all models.	[16]
2018	IEEE Transactions on Vehicular Technology	LSTM	RUL prediction of Li-ion batteries	A LSTM is constructed and enhanced with resilient mean square back-propagation technique which is designed for mini-batch training, drop out to reduce overfitting and Monte Carlo simulation to generate prediction uncertainties. This allowed the LSTM model to predict RUL independent of offline training data.	[28]
2021	Journal of Energy Storage	LSTM	SOH Forecasting	This work proposed a Variational LSTM with transfer learning. Model is first trained on source training dataset with a window size of i over K cycles, the trained weights is then transferred to target training set with window size of j over L cycles, the test also uses j and L as its window size and cycle. The model may not be ideal due to large size and high computational cost.	[29]
2021	Journal of Power Sources	LSTM	Battery degradation trajectory prediction	A sequence to sequence LSTM is used to predict battery degradation trajectory based on 100 cycles of data in one shot, the model is also compared with iterative approach LSTM, the sequence to sequence LSTM model outperformed iterative LSTM in both accuracy and computational efficiency.	[30]
2021	Journal of Cleaner Production	RNN	Online SOC estimation for EVs	Variable length LSTM along with L1, L2 and dropout is used in this work. Variable length input is done by masking layer adding filter characters (0) to sequences less than sequence length. L1, L2 and dropout are all regularization techniques to prevent model overfitting.	[31]
2020	Journal of Energy Storage	GRU-CNN	SOH estimation	GRU-CNN model described in this work is essentially a GRU and a CNN receives same input, the output of both model gets concatenated and project as SOH estimation. The training time is significantly long, it took 60193.27s to train the model on the NASA dataset.	[32]
2020	Applied Energy	CNN	RUL prediction	A dilated CNN based architecture is proposed, the architecture consists of 5 layers of dilated CNN, followed by a 1D-CNN with single output channel and 3 fully connected layers. The performance is compared with Shallow MLP, MLP, CNN and CNN+LSTM.	[33]
2020	Applied Energy	CNN	Capacity estimation	This study applied both transfer learning and ensemble learning to CNN. A source Deep CNN is trained with source dataset and transferred its weights to a target DCNN to perform capacity estimation on target dataset. In addition, 3 target DCNNs are used to generate 3 outputs, after ensemble a final capacity estimation is generated. DCNN-ETL performed batter than RF regression, GP regression, DCNN, DCNN-TL and DCNN-EL.	[34]
2019	Journal of Energy Storage	CNN	Online capacity estimation	A deep CNN consists of 5 convolutional layers and 3 fully connected layers is used in this work. DCNN had higher accuracy compared to RVM and ANN on NASA datasets.	[17]
2020	IEEE Access	CNN-GRU	SOH estimation on IoT microcontrollers	The CNN GRU architecture uses 1D convolutional layer, a max pooling layer, a GRU layer and a dense output layer. A performance evaluation is performed and CNN GRU outperformed baseline models RF and SVR, and DL models CNN, LSTM, CNN LSTM and GRU. Model is then incorporated into ARM Cortex-M4 MCU via STM32Cube.AI tool, the model's performance remains unchanged.	[18]

Table 1.1: Summary of ML battery applications

Items		Specification
Chemistry		NMC // Graphite
Capacity		55.6 Ah
Nominal Voltage		3.65 V
Dimension	Width (W)	354.00 mm
	Length (L)	99.05 mm
	Thickness (T)	9.5 mm
Weight		748 g
Volume		0.33 L
Energy Density	Weight	271 Wh/kg
	Volume	609 Wh/L
Voltage range @ 25°C		2.5 ∼4.2 V
Cell type		Pouch

Table 1.2: Specification of the NMC cell

Components	Max voltage	Max current
Power Supply	8V	125A
Electronic Load	120V	200A
Bipolar Power Supply	72V	6A

Table 1.3: Specification of test station's components.

1.2 Experimental Setup

For this work, experiments were conducted on a NMC pouch cell to gather experiment data, the specifications of the cell are summarized in Table 1.2.

A battery test station is constructed that consists of a programmable power supply and an electronic load and National Instruments data acquisition (NI-DAQ) system along with the LabVIEW software. In addition, a thermal chamber and calorimeter are used to keep the constant work temperature of a battery cell. LabVIEW controls all equipment and collects data from the battery cell in addition to protection of overcharge and discharge of the cell. The data includes terminal voltage, current and temperature, which is further processed to obtain the capacity, power and SOC. The block diagram for the test station is depicted in Figure 1.2, and the specification of test station's components are summarized in Table 1.3.

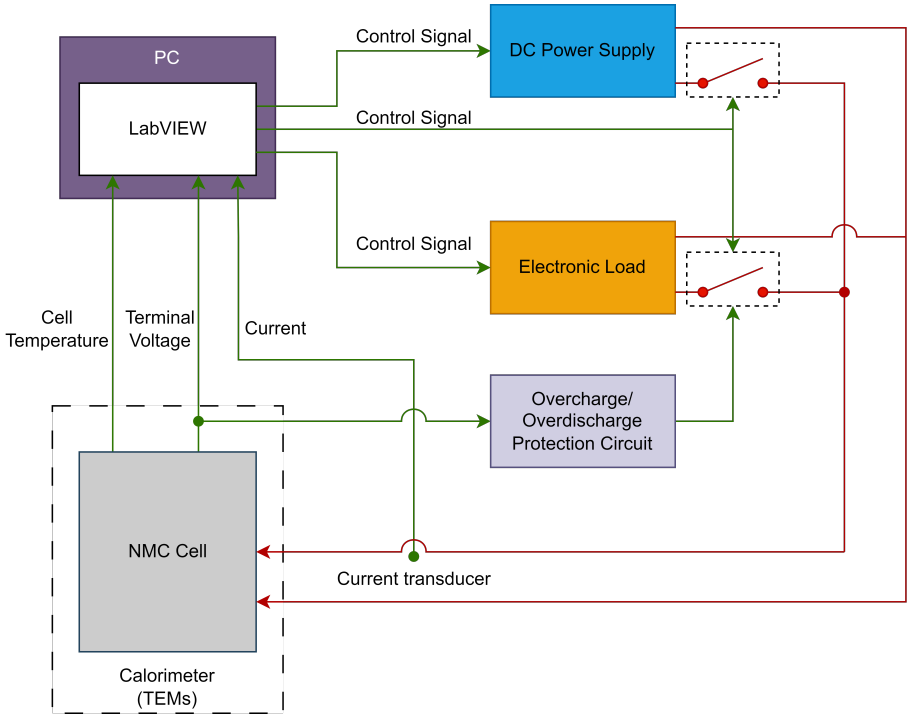

Figure 1.2: Block diagram of test station.

1.3 Motivation and objectives

Direct measurement of battery states and parameters during system operation is not feasible, necessitating their estimation. In order to accurately operate the Battery Management Systems (BMS), it's imperative to consistently estimate these states, prompting extensive research into advanced algorithms. Much of this research is concentrated on estimating individual states, such as the State of Charge (SoC) or State of Health (SoH) of the battery. This work explores the estimation of lithium-ion concentration, a specific facet within the intricate system of battery internals. These estimations could contribute to applications like State of Charge (SoC), State of Power (SoP)[35], and Terminal Voltage estimations.

The process of monitoring batteries presents significant challenges due to the dependency of their states on internal parameters. These parameters have a nonlinear relationship with a host of external operating conditions like temperature and load profiles. Furthermore, as the cell ages, these parameters undergo considerable changes, complicating the prediction of battery behavior. Consequently, sophisticated battery algorithms are essential for Battery

Management Systems (BMS) that can estimate these parameters throughout the battery life cycle while also factoring in external operating conditions.

However, several constraints must be kept in mind during the development of BMS algorithms. First, these algorithms are typically implemented using micro controllers, which have limited computational capabilities. Therefore, simpler models such as the equivalent circuit model (ECM) are generally preferred. Second, micro controllers have restricted memory capacity, so recursive methods, which eliminate the need for extensive data storage as they update with each sampling instance, are often favored. Lastly, the results derived from these estimation methods need to be both highly accurate and reliable. As a result, the application of advanced estimation techniques is crucial to meet the desired performance standards.

Building on the preceding analysis, this thesis introduces a data-driven method designed for fast and accurate estimation of the electrode surface concentration and the electrode average concentration in lithium-ion cells. This research concentrates on the following three principal areas:

- Development and optimization of a data-driven estimator using deep neural networks, fine-tuned via empirical methodologies.
- Estimation of electrode surface lithium-ion concentration $C_{s_{surf}}$ and average electrode lithium-ion concentration $C_{s_{avg}}$.
- Exploration of potential applications of the data-driven estimator, including State of Health (SoH), State of Power (SoP), and integration into Hybrid Models.

1.4 Thesis Outline

This thesis is structured as follows:

1. Introduction

This chapter provides the context for the thesis by discussing the background, the experimental setup, as well as the motivations and objectives of this research. It includes a brief introduction to the basic structure and operating principle of NMC lithium-ion cells and a literature review focused on machine learning methods and applications used in battery.

2. Battery Modeling

   This chapter offers a detailed exploration of the different types of battery models introduced in the first chapter. It delves into the specifics of physics-based models and Equivalent Circuit Models (ECM), as well as the commonly used data-driven models, highlighting their respective characteristics. A comprehensive comparison leads to the selection and implementation of a variant of Recurrent Neural Network (RNN), specifically a Long Short-Term Memory (LSTM) model, in the proposed data-driven estimator.

3. Lithium-ion Concentration Estimation

   Serving as the core of the thesis, this chapter presents the approach undertaken to address the research problem. It provides an overview of the problem and then elaborates on the inner workings of the LSTM cell. This is followed by a detailed inspection of the estimator's framework, outlining the methodology and process of implementation. Finally, the chapter concludes with the evaluation and discussion of the results derived from the estimator along with analysis of results and potential applications.

4. Conclusion and future work

   This final chapter summarizes the findings of the research as well as proposes potential avenues for future research.

Chapter 2

Battery Modeling

2.1 Modern Li-ion Battery Models

As we delve deeper into the complexity of lithium-ion batteries, it becomes apparent that the management of the energy storage systems is equally crucial. Battery Management Systems (BMS) are essential in maximizing the performance, lifespan and safety of LIBs. BMS monitor and manage various operational parameters, such as state-of-charge (SOC), state-of-health (SOH) and temperature. These systems are especially critical in applications like electric vehicles and grid energy storage systems, where battery performance can significantly impact the overall system's efficiency, performance and safety.

Battery models are used to simulate the behavior of batteries under different operating conditions, they play an essential role in the design of BMS and help optimize the performance and longevity of LIBs. Over the years, several battery models have been developed, ranging from simple empirical models to complex electrochemical models and more recently proposed data-driven models due to increased popularity of artificial intelligence.

Three categories of battery models are introduced and compared in this literature:

1. Physics-based Electrochemical Model
2. Equivalent Circuit Model
3. Data-driven Model

2.2 Physics-based electrochemical models

Electrochemical models, being physics-based, portray the intricate electrochemical processes happening within a cell. These processes are directed by physical laws, including

electrochemical kinetics, mass, charge, and energy balance, as well as potential theory. This set of principles generates a series of coupled, nonlinear partial differential equations (PDEs), defining the behavior of the model. These models enable the exploration of key battery behaviors at a microscopic level. Moreover, they offer a unique advantage of making all internal states fully observable, thus permitting 'virtual measurements' of quantities that are otherwise impossible to measure directly. Such a capability proves invaluable for comprehensive analysis and research.

2.2.1 Single Particle Model (SPM)

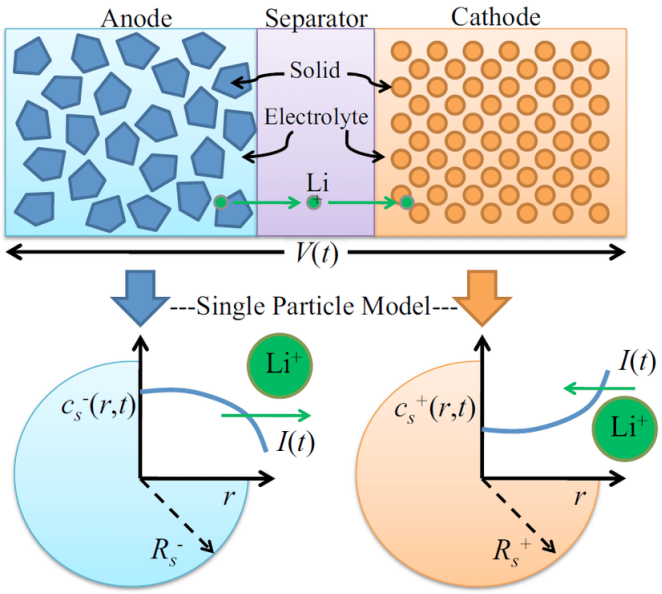

Figure 2.1: Diagram of SPM Model [1]

The Single Particle Model (SPM) treats each electrode in a battery as a single spherical particle as depicted in Figure 2.1. The model's primary objective is to calculate lithium-ion concentration within these particles and the cell's potential under various operating conditions[6]. The key assumption is that the lithium-ion concentration is uniform across each electrode, and diffusion is the only mechanism that lithium ions use to move in and out of the electrode particles.

The SPM provides a good balance between complexity and accuracy for many applications. However, due to its simplifying assumptions, it may not accurately represent the battery's behavior under all conditions. For example, it neglects the effects of electrode/electrolyte interactions and variations in temperature.

2.2.2 Pseudo-Two-Dimensional (P2D) Model

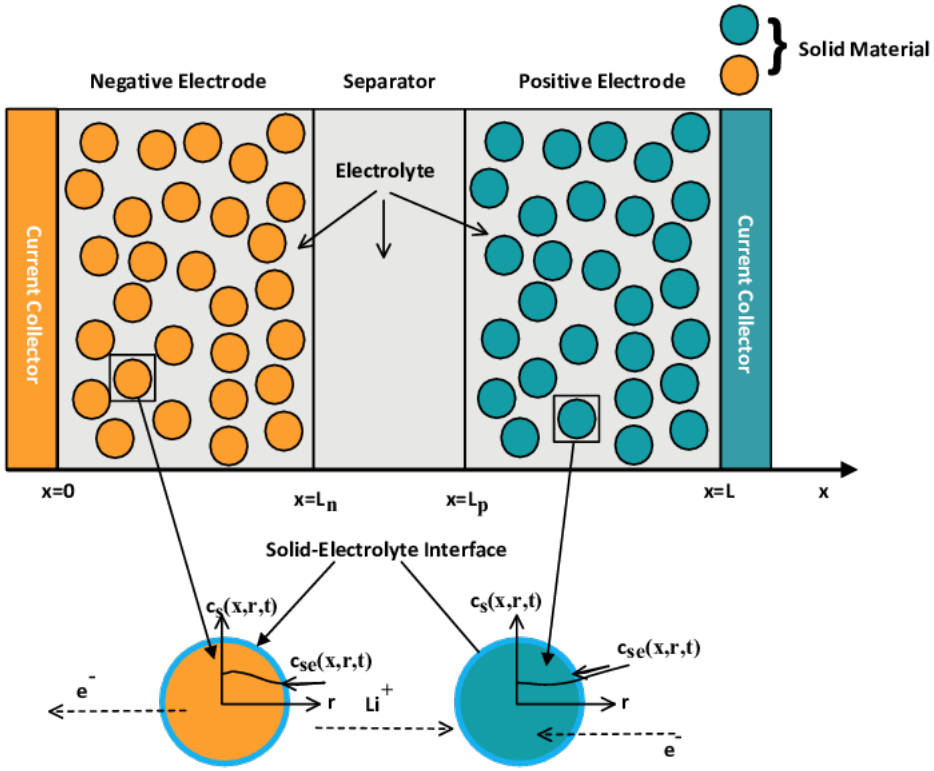

Figure 2.2: Schematic of pseudo-two-dimensional model for lithium-ion battery.[2]

The P2D Model, also known as the Doyle-Fuller-Newman (DFN) Model, proposed by Doyle et al.[36], is a more detailed electrochemical model that takes into account the transport and reaction process in both the solid electrode particles and the electrolyte, The model is a set of nonlinear partial differential and algebraic equations (PDAEs). Unlike the SPM, P2D model treats electrodes of the cell as porous electrodes composed of numerous spherical particles with electrolytes filled between the particles as depicted in Figure 2.2.

The P2D model can describe the lithium-ion concentration distribution in the solids of electrodes and the liquid phase of electrolytes with in the cell. However, the P2M model suffers from its computational complexity, it can be computationally intensive to solve a set of PDAEs. Therefore, the P2D model may not be suitable for real-time applications, to overcome this issue, model-order reduction methods were used to transform the PDAEs into ordinary differential equations (ODEs), this technique simplified the rigorous P2D model which increased the computational efficiency with acceptable accuracy.

The P2D model have following components:

1. Mass conservation in electrode:

   This represents the diffusion of lithium-ions within the solid electrode particles.

   Governing equation:

$$\frac{\partial C_s}{\partial t} = \frac{D_s}{r^2}\frac{\partial}{\partial r}(r^2\frac{\partial C_s}{\partial r}) \tag{2.1}$$

   Boundary and initial conditions:

$$\left.\frac{\partial C_s}{\partial r}\right|_{r=0} = 0,$$

$$\left.\frac{\partial C_s}{\partial r}\right|_{r=R_s} = -\frac{1}{D_s F a_s} j^{Li}$$

2. Mass conservation in electrolyte:

   This equation represents the diffusion and migration of lithium ions in the electrolyte:

   Governing equation:

$$\frac{\partial(\varepsilon_e C_e)}{\partial t} = \frac{\partial}{\partial x}\left(D_e^{eff}\frac{\partial}{\partial x}C_e\right) + \frac{1-t_+^0}{F} j^{Li} \tag{2.2}$$

   Boundary and initial conditions:

$$\left.\frac{\partial C_e}{\partial x}\right|_{x=0} = \left.\frac{\partial C_e}{\partial x}\right|_{x=L} = 0$$

3. Charge conservation in electrode:

   This represents the movement of electrons within the solid electrode:

   Governing equation:

$$\frac{\partial}{\partial x}\left(\sigma^{eff}\frac{\partial}{\partial x}\phi_s\right) - j^{Li} = 0 \tag{2.3}$$

   Boundary and initial conditions:

$$-\sigma^{eff}\frac{\partial}{\partial x}\phi_s\bigg|_{x=0} = -\sigma^{eff}\frac{\partial}{\partial x}\phi_s\bigg|_{x=L} = \frac{I}{A}$$

$$\frac{\partial}{\partial x}\phi_s\bigg|_{x=\sigma_-} = \frac{\partial}{\partial x}\phi_s\bigg|_{x=\sigma_-+\sigma_{sep}} = 0$$

4. Charge conservation in electrolyte:

   This represents the movement of ions within the electrolyte:

   Governing equation:

$$\frac{\partial}{\partial x}\left(\kappa^{eff}\frac{\partial}{\partial x}\phi_e\right) + \frac{\partial}{\partial x}\left(\kappa_D^{eff}\frac{\partial}{\partial x}\ln\phi_e\right) + j^{Li} = 0 \tag{2.4}$$

   Boundary and initial conditions:

$$\frac{\partial\phi_e}{\partial x}\bigg|_{x=0} = \frac{\partial\phi_e}{\partial x}\bigg|_{x=L} = 0$$

5. Electrochemical kinetics

   This describes the rate of the charge-transfer reaction at the electrode/electrolyte interface:

   Governing equation:

$$j^{Li} = a_s i_0\left\{exp\left[\frac{\alpha_a nF}{RT}\eta\right] - exp\left[-\frac{\alpha_c nF}{RT}\eta\right]\right\} \tag{2.5}$$

2.2.3 Reduced Order Model (ROM)

Reduced Order Models (ROMs) are simplified representations of high-fidelity simulation models that retain the essential behavior and predictive capability of the original model but at a significantly reduced computational cost[37][38].

A polynomial ROM utilizes a polynomial equation to approximate the behavior of a complex system, it is typically derived from a full-order model, by retaining the most important terms/features and discarding less significant ones[39].

The strength of Polynomial ROMs lies in their computational efficiency, which is attributed to their simplification compared to comprehensive models. Furthermore, the interpretability of these models is notably straightforward, with the coefficients of polynomial terms offering clear insights, unlike more complex models such as neural networks.

However, the main challenge with Polynomial ROMs lies in defining the appropriate polynomial order. While higher orders can enhance accuracy, they also introduce the risk of overfitting. Therefore, choosing the polynomial order requires careful consideration to strike a balance between model complexity and predictive performance.

2.3 Equivalent Circuit Model (ECM)

Equivalent circuit models (ECMs) are often used to describe the dynamic behavior of batteries due to their relative simplicity and the ease of obtaining their parameters through simple experiments[40][41].

A battery ECM is constructed with a combination of basic electrical components like resistors, capacitors, and voltage sources. The correlation with battery dynamics is preserved by adding capacitors into the circuit. In this section, the "R_{int}" model and Thevenin Model will be covered.

1. R_{int} Model:

R_{int} Model is constructed with a resistor R_0 and a voltage source (OCV) to represent the internal resistance of the battery, as shown in Figure 2.3. Because the model dos not represent transient behavior, it is unable to estimate battery states under non-constant load accurately.

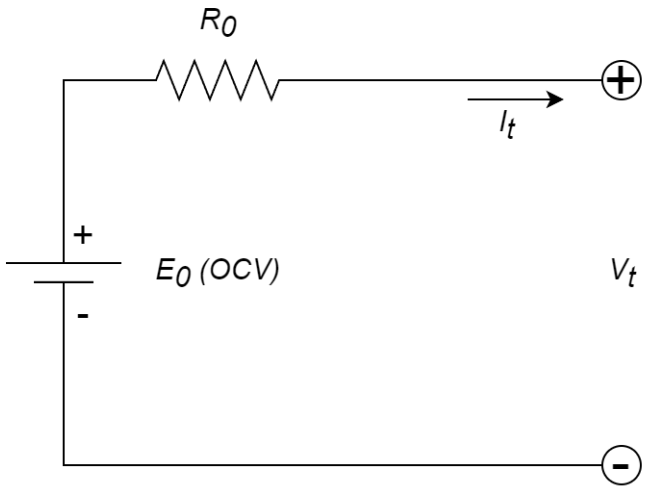

Figure 2.3: Diagram of R_{int} Model.

The R_{int} model can be described by Equation 2.6.

$$V_t = E_0 - R_0 \cdot I_t \tag{2.6}$$

Where V_t is the terminal voltage, E_0 is the open-circuit voltage (OCV), R_0 is the internal resistance and I_t is the current.

2. The Thevenin Model:

   The Thevenin model adds a capacitor and another resistor to the R_{int} model to account for the transient behavior of the battery. The structure of the model is shown in Figure 2.4.

   The Thevenin model can be represented by the following equations:

$$V_t = E_0 - R_0 \cdot I_t - V_{C_1} \tag{2.7}$$

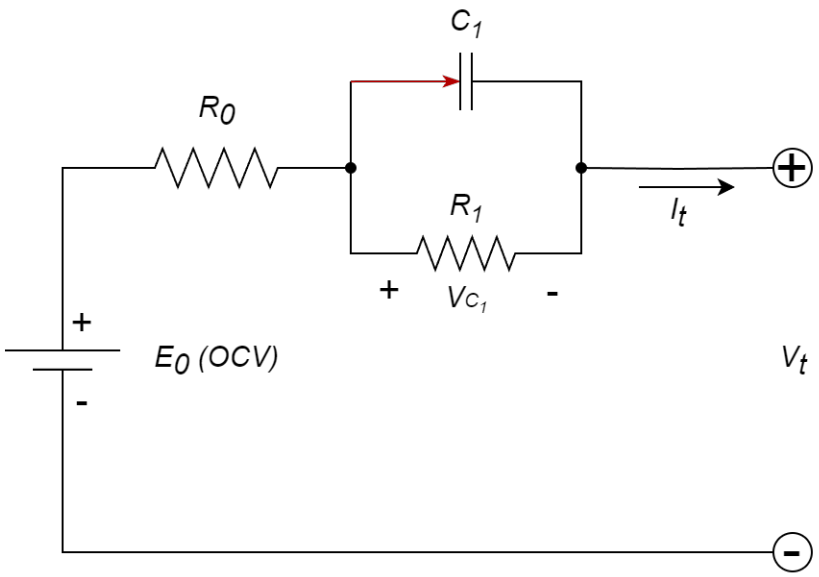

Figure 2.4: Diagram of the Thevenin Model.

$$\frac{dV_{C_1}}{dt} = -\frac{V_{C_1}}{R_1 C_1} + \frac{I_t}{R_1} \tag{2.8}$$

Where R_1 and C_1 represent the polarization resistance and capacitance, V_{C_1} is the voltage across C_1.

2.4 Data-Driven Models

In addition to Physics-based electrochemical models, data-driven models have recently gained popularity in battery modeling due to advancements in computational power and machine learning. Data-Driven Models are not explicitly designed to understand the underlying complex internal physical and chemical processes within the battery, they learn the relationships between the battery's inputs and its outputs from experimental data. Various machine learning techniques including regression, artificial neural networks and more, are widely used in data-driven models[42]. In this section, a few popular algorithms will be introduced and compared.

2.4.1 Feed-Forward Neural Network (FNN)

A Feed Forward Neural Network (FNN) is a type of artificial neural network where information flows in only one direction, from the input layer to the output layer. It is called "feed forward" because there are no loops or cycles in the network's connections[23][24][11][25]. FNNs are composed of multiple layers of interconnected neurons, which are computational units that perform simple mathematical operations. The network learns from training data by adjusting the weights and biases associated with these connections. A diagram of FNN in shown in Figure 2.5.

The architecture of an FNN consists of three main components: the input layer, one or more hidden layers, and the output layer. The input layer receives the input data and passes them to the next layer, the number of neurons in the input layer corresponds to the number of features input possesses. The hidden layer is located between the input and output layers, there can be more than one hidden layers and each hidden layer can have varying number of neurons, allowing the network to understand complex relationships and patterns behind the data. The output layer produces the network's final predictions, the number of neurons in the output layer corresponds to expected output size, which depends on the specific problem been addressed by the network.

Feed-Forward Neural Network is very versatile and has various variants and applications including battery modeling, SOC estimation and SOH&RUL estimation.

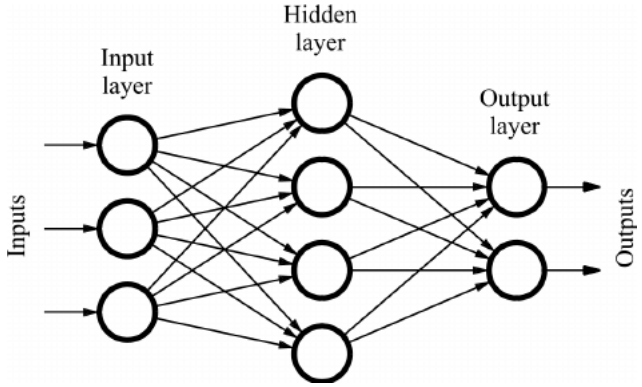

Figure 2.5: Schematic of Feed-Forward Neural Network[3]

Activation functions introduces non-linearity into the FNN, enabling it to model complex relationships between input features and output predictions, some of the most popular activation functions include:

- Sigmoid: Logistic Sigmoid Function takes any real valued input and outputs a value within the range of 0 to 1, the mathematical representation is as follows:

$$f(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \tag{2.9}$$

  Sigmoid function is very commonly used for probability predictions because the range of Sigmoid function matches with the range of probability which only exists between 0 and 1.

- Tanh: Hyperbolic Tangent Function takes any real valued input and maps to a value within the range of -1 to 1, the mathematical representation is as follows:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.10}$$

  Since the output of the tanh activation function is zero centered, we can map the output values as strongly negative, neutral, or strongly positive.

- ReLU: Rectified Linear Unit or ReLU allows positive values to flow through as they are and set the output to 0 for negative inputs, the mathematical representation is as follows:

$$f(x) = max(0, x) \tag{2.11}$$

  Compared with Sigmoid and Tanh activation function mentioned above, ReLU is computationally efficient as only certain number of neurons are activated, in addition, for positive x, the ReLU function has a constant gradient of 1, whereas a Sigmoid function has a gradient that rapidly converges towards 0, making the neural network hard to

train, this is known as the vanishing gradient problem, and by using ReLU as activation function, this problem is alleviated.

2.4.2 Recurrent Neural Network (RNN)

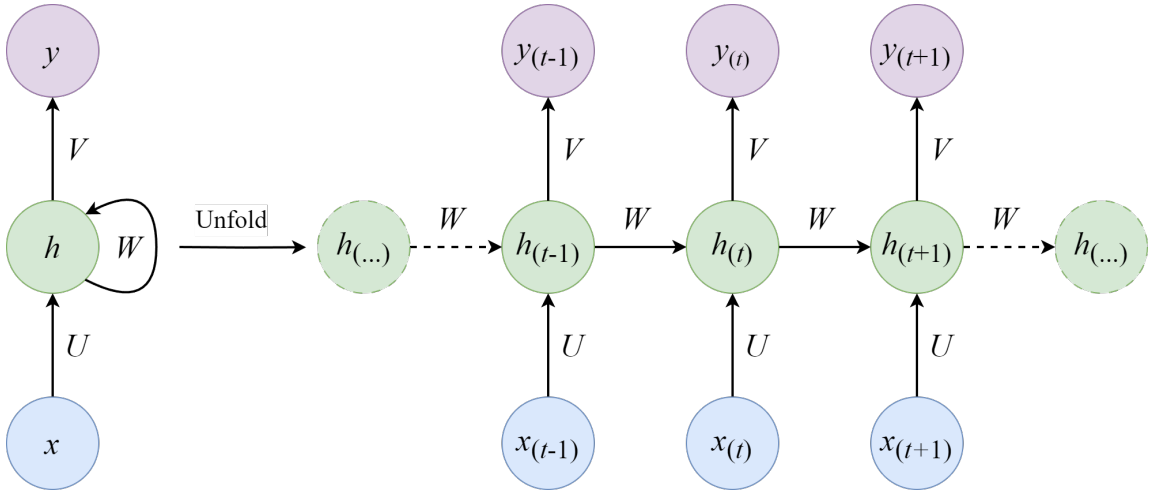

Figure 2.6: Schematic of RNN and unrolled RNN.

A Recurrent Neural Network (RNN) is a type of artificial neural network designed to process sequential data, where the order and temporal dependencies of the data are crucial[43]. Recurrent Neural Networks use learns from the training data just like Feed-forward Neural Networks, unlike FNNs, RNNs have can use its "memory" to take information from prior inputs to influence the current input and output.

A schematic of RNN and unrolled RNN is shown in the Figure 2.6. The simple RNN is shown on the left and on the right side is an RNN being unfolded into a full network, where x is the input into the network, and x_t is the input into the network at time step t, similarly y is the output of the network and y_t is the output of the network at time step t. Hidden state is represented as h and the hidden state at time step t is h_t which acts as the "memory" of the network, h_t is calculated based on the current input and the previous time step's hidden state. m is the number of hidden units in this network.

In this section, the following notations are used to explain RNN:

- $x_t \in \mathbb{R}$ is the input at time step t.
- $y_t \in \mathbb{R}$ is the output of the network at time step t.
- $h_t \in \mathbb{R}^m$ is the hidden state at time t.
- $U \in \mathbb{R}^m$ is the weight matrix for input to hidden connections.
- $W \in \mathbb{R}^{m \times m}$ is the weight matrix for hidden to hidden connections.
- $V \in \mathbb{R}^m$ is the weight matrix for hidden to output connections.
- $b_h \in \mathbb{R}^m$ is the bias for recurrent layer.
- $b_y \in \mathbb{R}$ is the bias for output layer.

At time step t - 1, we can calculate the hidden state of time step t with activation function f by the equation below:

$$h_t = f(Ux_{t-1} + Wh_{t-1} + b_h) \tag{2.12}$$

The output y at time step t is computed as:

$$y_t = f(w_y \cdot h_t + b_y) \tag{2.13}$$

The forward pass of an RNN starts with initialization, where the initial hidden state is set to a default value or initialized randomly, then at each time step t in the input sequence, input x_t is received and combined with hidden state from last time step h_{t-1} to calculate the hidden state h_t which can be used to calculate the output y_t, once output is calculated, hidden state h_t becomes the previous time step hidden state for next time step. The above process is repeated for each time step in the sequence, once the entire sequence has been processed, the RNN can produce a final output.

After the forward pass, the error of the output and the target output is calculated using a loss function. The error gradients are propagated backward through time to update the weights of the RNN, this process starts from the final time step and goes back to the first time step. At each time step during the backward pass, the gradients of the model's parameters, including the weights and biases, are computed with respect to the loss function using the chain rule of calculus, the gradients are accumulated and stored for each time step. Once the gradients have been calculated for all time steps, the accumulated gradients are used to update the model's weights and biases using an optimization algorithm such as gradient descent, the weights are adjusted in the direction that minimizes the loss function.

This iterative process of forward pass, backward pass, and weight update is repeated for multiple iterations or epochs to improve the RNN's performance and reduce the loss.

Just like feed-forward neural networks, various activation functions can be used on recurrent neural networks to introduce non-linearity and enable the network to model complex relationships, some commonly used functions include the Logistic Sigmoid Function, the Hyperbolic Tangent Function and the Rectified Linear Unit Activation Function, they have been introduced in detail in previous sections.

Recurrent Neural Networks is effective for analyzing time series data, they can capture long-term dependencies, trends and patterns in the data and provide accurate predictions and insightful analysis. However, RNN also has its shortcomings, the backward pass in RNNs can be prone to the exploding and vanishing gradient problem, particularly for long sequences. The issue arises when the gradients increase or decrease exponentially as it propagates through the model and eventually explode or diminishes, making it challenging to capture long-term dependencies.

RNNs have several applications in battery modeling tasks, with its ability to handle time series data and sequence input, some of applications include Lithium-ion Battery State of Charge (SoC) estimation[12][13][1][19][20], Remaining Useful Life (RUL) and State of Health (SOH) estimation[28][44][16][29][27][30][31].

2.4.3 Convolutional Neural Network (CNN)

Convolutional Neural Networks are a class of deep learning models widely used for tasks involving image and signal processing. In 1990 LeCun et al. introduced the modern framework of CNN[45], later in 1998, his work is improved and the application was presented[46]. They are specifically designed to automatically extract and learn hierarchical representations from data. CNNs have revolutionized computer vision tasks, achieving remarkable performance in tasks such as image classification, object detection and image segmentation.

The architecture of a CNN typically consists of three main types of layers: convolutional layers, pooling layers, and fully connected layers.

- Convolutional Layers: Convolutional layer is responsible for feature extraction in a CNN, and it is also where most of the computation happen. Convolutional layer will apply a set of learnable filters, also known as kernels to the input data, the filter performs convolution operations to capture local patterns and extract relevant features. It is possible to layer another convolutional layer following the initial convolutional layer, enabling the network to learn and detect lower level patterns and higher level patterns of the input data.

- Pooling Layers: Pooling layers reduce the spatial dimensionality of the input by downsampling the features. The pooling layer performs a pooling operation by sweeps a filter across the entire input, the output depends on the specific pooling operation, two main types of pooling operation are max pooling and average pooling, max pooling selects the maximum value as output and average pooling calculates the average value within the receptive field as output.

- Fully Connected Layers: Fully connected layers, typically placed at the end of the network, transform the extracted features into a suitable format for the final classification or regression. These layers connect every neuron from the previous layer to every neuron in the current layer.

During the forward pass of a CNN, the input data is passed through the convolutional layers, the learnable filter performs a dot production operation between its weights and the input, capturing spatial features, this dot product is then fed into an output array. The filter slide across the input, repeating the process until the entire input has been covered. The final output array consists of dot products from the input and weights is known as a feature map.

After the convolution operation, an activation function is applied element-wise to the feature maps, the most commonly used activation function in CNNs is the Rectified Linear Unit (ReLU) mentioned above, ReLU introduces non-linearity to the model.

Next, a pooling layer is applied to down-sample the feature maps, max pooling and average pooling introduced above selects the maximum value or average value within the receptive field of the input. By reducing spatial dimension and number of parameters in the network through pooling, the complexity of model is reduced allowing efficient computation to be carried out.

After pooling layer, its output is flattened into a one-dimensional vector and passed into fully connected layers, the fully connected layers transform the extracted features into a suitable format for the final classification or regression task.

The output of fully connected layers is fed into another activation function to produce the final output. If the model is tasked to handle regression, linear activation function can produce continuous values as the output, if the model is tasked to handle classification, softmax is often used as the softmax function outputs a probability distribution over multiple classes.

Similar to the neural networks introduced above, the backward pass in a CNN involves the calculation of gradients using the backpropagation algorithm. The error gradients flow from the output layer to the input layer, updating model's weights and biases through gradient descent optimization. The gradients are computed using the chain rule of calculus, and the weights are adjusted to minimize the difference between predicted and actual values.

In summary, convolutional neural network is very good at capturing spatial hierarchies and meaningful patterns from data, this has been proved by its popularity and success in the area of computer vision. However, CNNs require large training datasets, it is hard to interpret and understand how decisions were made, especially for deep networks with many layers, the increase in depth of the model also means it requires more computational resources for training and evaluating.

Although convolutional neural network is mainly used for image processing tasks[47], it is proven that CNN is capable of several applications in the field of battery modeling, some of the applications include SOC estimation and SOH&RUL estimation.

Chapter 3

Lithium-ion Concentration Estimation

3.1 Overview

In this work, a data-driven battery state estimator is implemented and its estimations are validated against results calculated by Full Order Model. The Full Order Model's result is referring to a single particle in anode.

For this estimator, the expected input vector, x contains the following states:

$$x = [I^T, V^T, T^T, C^{T-1}_{s_{surf}}, C^{T-1}_{s_{avg}}]$$

where I^T, V^T, T^T denotes the current, voltage and temperature measured at current time step T, and $C^{T-1}_{s_{surf}}, C^{T-1}_{s_{avg}}$ denotes the lithium-ion concentration at electrode surface and average lithium-ion concentration of electrode at time step T-1.

And the output vector y contains the following states:

$$y = [C^T_{s_{surf}}, C^T_{s_{avg}}]$$

where $y = [C^T_{s_{surf}}, C^T_{s_{avg}}]$ denotes the lithium-ion concentration at electrode surface and average lithium-ion concentration of single particle in Anode at time step T respectively.

Since no sensor is available to measure $C_{s_{surf}}$ and $C_{s_{avg}}$ within the cell, fast and accurate estimation of lithium-ion concentration have many applications like cell SoC calculation, potential calculation and state of power calculation[35].

3.2 Problem Framing

The problem that motivated this work is cell internal state $C_{s_{surf}}$ and $C_{s_{avg}}$ can not be measured, accurate estimation of $C_{s_{surf}}$ and $C_{s_{avg}}$ can enable the calculation of battery states such as state of charge (SOC) and terminal voltage.

Current existing non-ML solution (P2D) can accurately describe internal states of cell but too computationally expensive, limited its application in real-time scenarios. Another current non-ML solution (ROM) is computational efficient but not as accurate compared to P2D.

The ideal outcome is to develop a data-driven model that can estimate cell internal state $C_{s_{surf}}$ and $C_{s_{avg}}$ with a high level of accuracy and computational efficiency, with potential for real-time applications and improving over time with access to more data.

Therefore the model's goal is to predict $C_{s_{surf}}$ and $C_{s_{avg}}$ accurately and swiftly, the model's outputs are $C_{s_{surf}}$ and $C_{s_{avg}}$.

3.3 Approach

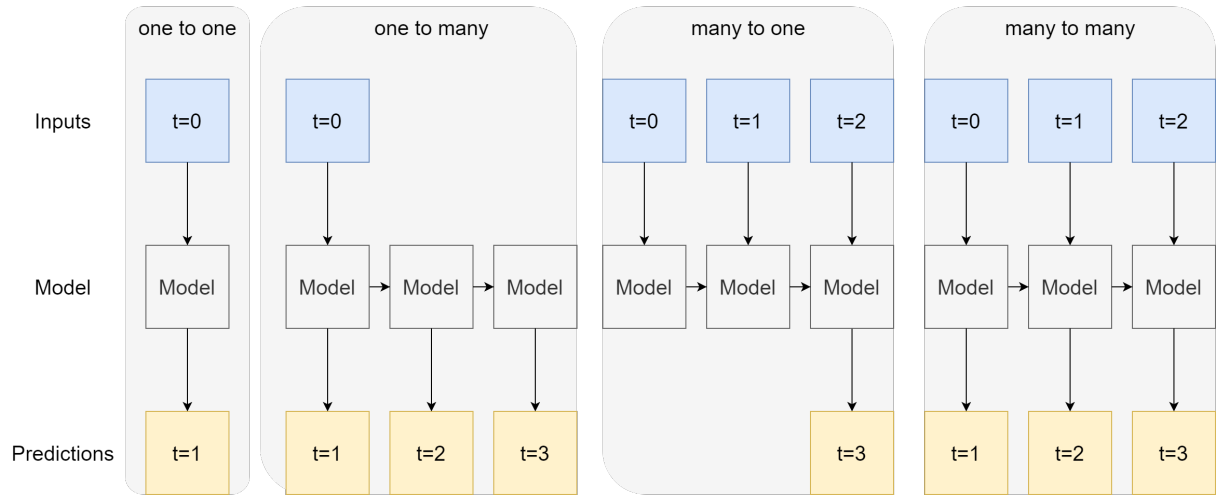

Figure 3.1: Different types of time series forecasting.

This problem can be categorized as a time series forecasting problem as the model is predicting future values based on historical time series data. time series forecasting have different types, as shown in Figure 3.1:

- One to one:

  One to one has a single input feed into the model and the model outputs one prediction based on the input.

- One to many:

  One to many has a single input feed into the model and the model outputs multiple predictions based on the input.

- Many to one:

  Many to one has a sequence of inputs feed into the model and the model outputs one prediction based on the sequence input.

- Many to many:

  Many to many has a sequence of inputs feed into the model and the model outputs multiple predictions based on the input.

For this work, one to one and one to many time series forecasting is eliminated because it makes predictions based on data from previous step only, neglecting data from near past, which has an influence of the concentration level within electrode.

Many to one is selected as the desired approach for this work, because it takes in a sequence of past data and predict a single step into future, which fits the ideal outcome: provide accurate and swift estimations with potential for real-time applications.

Many to many is also a viable approach, but considering the added computational cost of multiple predictions based on inputs, it is apparent that many to many is less efficient compared to many to one forecasting model.

In this work, a sliding window autoregressive many to one forecasting model is used, the idea is shown in Figure 3.2. The model starts with provided input colored in blue, and after each time step the prediction of model is used as input for the next time step, over time the model is making predictions purely based on its predictions.

This particular approach is selected based on the fact that the model may be used in real-time applications where the model is required to provide accurate estimations in short intervals while Longer interval fails to provide BMS accurate cell state and capability. If BMS request the state of cell during the interval, it either has to make a decision based on expired data that may not reflect the true state of cell at the moment or wait for next prediction, which could potentially lead to unsafe operation and slow response time.

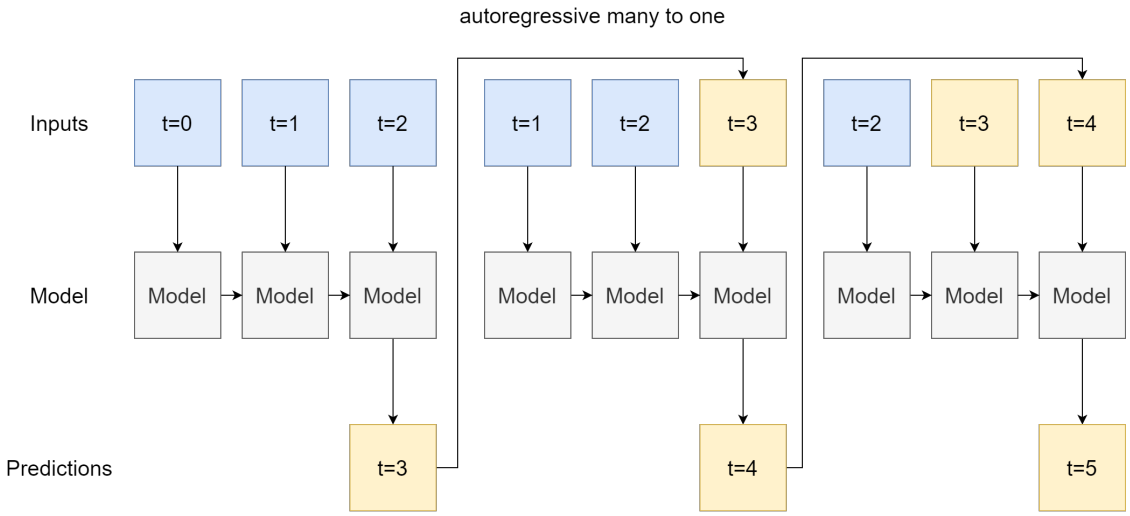

Figure 3.2: Autoregressive many to one forecasting model.

3.4 Model Selection

In this section, the reasoning for selecting LSTM as the model for this thesis is explained in detail.

In the paper by Chun et al.[9], a performance comparison of network architecture is included. A total of 6 networks were tested, the networks are MLP, VGG11, VGG16, ResNet18, GRU and LSTM. MLP is a fully connected class of feed forward neural network (FNN) and

VGG11, VGG16 and ResNet18 are all pretrained deep convolutional neural networks (CNN), GRU and LSTM are both variants of recurrent neural network (RNN). The results of performance comparison shown LSTM and GRU had the lowest estimation error with a RMSE% of 0.43 and 0.56 respectively, MLP scored a RMSE% of 1.13 while VGG11, VGG16 and ResNet18 performed worse with a RMSE% of 1.54, 1.68 and 1.63 respectively. Based on the results of performance comparison we can conclude RNNs performed the best, followed by FNNs and CNNs came in last, within RNN models, LSTM out performed GRU as the best network for the real-time parameter estimation task Chun et al. addressed.

Similar trend was observed in research conducted by Choi et al.[16], where a capacity estimation framework for lithium-ion battery is proposed, along with a performance evaluation of FNN, CNN and LSTM. Each model's structure is designed based by its performance of validation set and MAPE is considered as an representative error index. The capacity estimation results of FNN, CNN and LSTM shown that LSTM has the best accuracy among the models tested with a MAPE of 1.377%, the baseline FNN-1 with 10 neurons has a MAPE of 4.71% and FNN-2 with 40 neurons has a lower MAPE% of 3.65, the CNN-1 with 10 filters for first convolution layer and 5 filters for second convolution layer scored a MAPE of 4.002% while the CNN-2 with 30 filters for first convolution layer and 15 filters for second convolution layer scored a MAPE of 4.419%. It is also mentioned in paper that the LSTM was able to provide accurate estimation under abnormal deviation of the data due to cell states storing long-term information, suppressed the effect of weight of abnormal data. The results in this work proved again that LSTM is the choice for time series data regression tasks out of FNN, CNN and LSTM.

Kaur et al.[27] further reinforced this consensus in their comparative analysis of different deep learning networks for capacity estimation tasks. Similar to research conducted by Chun et al.[9] and Choi et al.[16], FNN, CNN and LSTM are considered for time series data estimation. The results shown LSTM is the best model for this task with an average RMSE and MAE over 6 tests of 0.0426 and 0.0216 respectively, followed by FNN with an

average RMSE and MAE of 0.0447 and 0.0183 respectively, CNN came last with an average RMSE and MAE of 0.0527 and 0.0311 respectively. The LSTM's average RMSE was 4.69% and 19.16% lower as compared to FNN and CNN in order. The study concluded LSTM performed better than FNN and CNN due to its ability to interpret variable sized time series battery data in a better manner.

In addition to the battery related studies mentioned above, a study carried out by Chandra et al. [48] evaluating deep learning models for multi-step ahead time series prediction where neural network models includes FNN-SGD, FNN-Adam, LSTM, BD-LSTM, ED-LSTM, RNN, and CNN is evaluated against multiple real world time series problems. The results over 7 different problems shown Bi-directional LSTM (BD-LSTM) has a mean-rank of 2.00, Encoder-decoder LSTM (ED-LSTM) has a mean-rank of 2.14 and LSTM has a mean-rank of 2.42. CNN achieved a mean-rank of 3.85, followed by RNN with a mean-rank of 4.85. FNN-Adam was able to outperform FNN-SGD with a mean-rank of 5.42 while FNN-SGD came last with a mean rank of 7, consistently outperformed by all other models in all 7 problems. This study once again confirmed LSTM along with its variants, offers the best performance in time series prediction problems.

Collectively, these studies demonstrated the supremacy of LSTM in both parameter and capacity estimation tasks due to its ability to maintain data within its cell states. This characteristic, inherent to LSTM's architecture, enables effective management of time series data, thus making LSTM an excellent choice for these estimation tasks.

Based on the compelling empirical evidence provided by these studies, LSTM was selected as the most appropriate model for this thesis, as it exhibits a unique combination of accuracy, robustness, and data retention capabilities in the context of time-series prediction problems. With its proven proficiency in handling time-series data and its superior performance in comparison with other deep learning models, LSTM demonstrates the necessary qualities to tackle the challenges posed in this thesis.

3.5 Long Short-Term Memory Neural Network

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network architecture specifically designed to address the vanishing gradient problem and capture long-term dependencies in sequential data. It was introduced by Hochreiter and Schmidhuber in 1997 [49].

The LSTM architecture consists of various gates and memory cell, also known as cell states. Cell states can store information over long periods of time, allowing the network to retain and utilize relevant information from earlier time steps, The key components of an LSTM cell are:

- Cell States (C_t): The cell state acts as the memory of the LSTM, It retains and carries information throughout the entire sequence of inputs, allowing the network to capture and maintain long-term dependencies. It is depicted as the top horizontal line in Figure 3.3. The cell state is modified using gates and activation functions to selectively add or remove information.

- Hidden State (H_t): The hidden state is derived from the cell state and carries the summarized information of the sequence to the current time step, it acts as the output of the LSTM cell.

- Input Gate (i): The input gate determines which parts of the input should be stored in the cell state. It combines information from the current input and the previous hidden state as inputs that passes through a sigmoid activation function and produces a vector with values ranges between 0 and 1, 1 represent to keep data and 0 means the data will not be kept for update.

- Forget Gate (f): The forget gate decides which information to discard from the cell state. It takes the current input and the previous hidden state as inputs that pass through a sigmoid activation function and outputs a vector with values range between

0 and 1 for each element in the cell state, 1 represent to keep data and 0 means the data will not be kept for update.

- Output Gate (o): The output gate controls the flow of information from the cell state to the output. It takes the current input and the previous hidden state as inputs that pass through a sigmoid activation function and outputs a vector with values range between 0 and 1 for each element in the cell state, 1 represent to keep data and 0 means the data will not be kept for output.

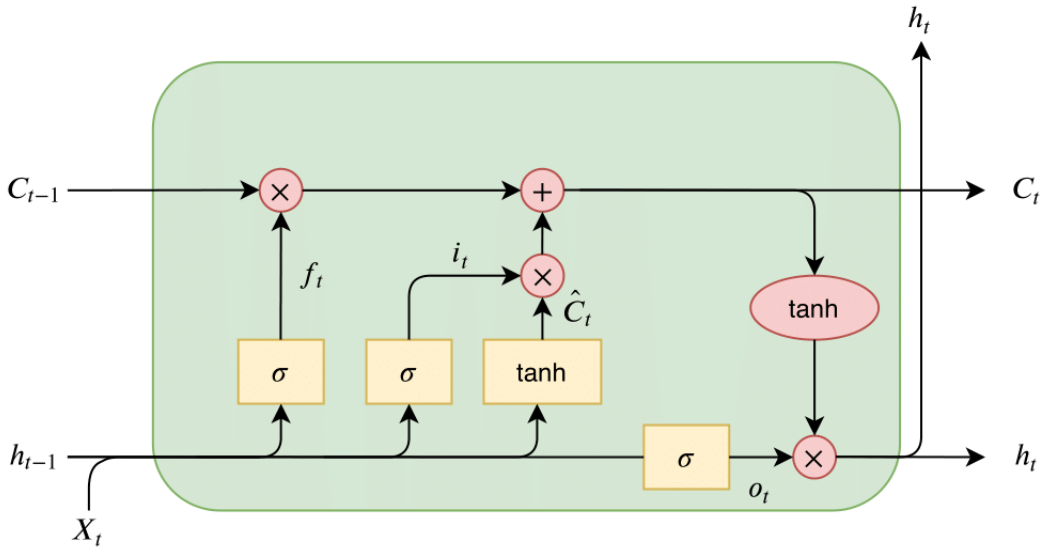

Figure 3.3: Diagram of a LSTM cell

In this part, we have a LSTM neural network with h hidden units and number of inputs is d. A forward pass of above model contains the following steps, during the forward pass, information flows through cell states C_t:

- Last time step's hidden state H_{t-1} and the current time step input X_t is concatenated and entering the data rail depicted as the bottom horizontal line in Figure 2.5, we will refer to it as combined input.

- The combined input is fed into the forget gate, which decides what information is going to be discarded from the cell state, a sigmoid layer within the gate will output a

number between 0 and 1 for each number in last time step's cell state $C_t - 1$, 0 means to discard that information and 1 means to keep the information. The formula for forget gate is as follows:

$$f_t = \sigma(W_{xf}X_t + W_{hf}H_{t-1} + b_f) \tag{3.1}$$

where $W_{xf} \in \mathbb{R}^{d\times h}$ and $W_{hf} \in \mathbb{R}^{h\times h}$ represents weight parameters between input layer to hidden layer and hidden layer to hidden layer and $b_f \in \mathbb{R}^{1\times h}$ represent bias parameter of the forget gate.

- The combined input is then fed into the input gate to decide what new information will be stored in cell state. Inside the input gate, a sigmoid layer acts as a filter of identifying which component of the new candidate vector will be updated in cell state. A tanh layer also takes in the combined input and output a vector of new candidate values $\tilde{C}_t$, every value in the vector ranges between -1 to 1. The formula for input gate is as follows:

$$i_t = \sigma(W_{xi}X_t + W_{hi}H_{t-1} + b_i) \tag{3.2}$$

$$\tilde{C}_t = tanh(W_{xc}X_t + W_{hc}H_{t-1} + b_C) \tag{3.3}$$

where $W_{xi}, W_{xc} \in \mathbb{R}^{d\times h}$, $W_{hi}, W_{hc} \in \mathbb{R}^{h\times h}$ are weight parameters and b_i, $b_c \in \mathbb{R}^{1\times h}$ are bias parameters for input gate.

- From the steps above we have output from forget gate f_t that dictate how much information within last time step cell state C_{t-1} is discarded, output from input gate i_t that determines how much of the combined input is to be updated into cell state C_t and new candidate values from the tanh layer $\tilde{C}_t$. To update the cell state C_t, first the last time step cell state C_{t-1} is multiplied with output of forget gate f_t using element-wise multiplication, in this step information is discarded from C_{t-1}, then a element-wise multiplication between i_t and $\tilde{C}_t$ is performed, as the filtered new candidate inputs

to the cell state C_t, addition is performed and the new cell state C_t is updated. The formula is as follows and $\odot$ represents element-wise multiplication:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{3.4}$$

- Next, the output of output gate o_t is calculated, the combined output is fed into the output gate, a sigmoid layer will output a number between 0 and 1 for each value in cell state C_t, if the number is 1 then the information will be included in the hidden state h_t to impact subsequent layers, if the number is 0 the information would be kept away from subsequent layers. The formula for output gate is as follows:

$$f_t = \sigma(W_{xo}X_t + W_{ho}H_{t-1} + b_o) \tag{3.5}$$

where $W_xo \in \mathbb{R}^{d\times h}$ and $W_ho \in \mathbb{R}^{h\times h}$ are weight parameters and $b_o \in \mathbb{R}^{1\times h}$ represent bias parameters for the output gate.

- The hidden state H_t is calculated by applying tanh to the cell state C_t, this step ensures the values within C_t are between -1 and 1, this vector is then multiplied with output of output gate o_t and the result is hidden state H_t. The formula for calculation of hidden state H_t is as follows:

$$H_t = o_t \odot tanh(C_t) \tag{3.6}$$

- Finally, the final estimation result y_t is calculated by passing the output hidden state of each time step H_t through a fully connected layer, the formula is as follows,

$$y_t = W_{fc}H_t + b_{fc} \tag{3.7}$$

where W_{fc} and b_{fc} are the weight parameters and bias parameters for the fully connected layer.

The backward pass of LSTM is similar to the backward pass of Recurrent Neural Network introduced above, the error gradients are calculated and propagated through time using Back Propagation Through Time (BPTT) algorithm, as the gradients flow through the LSTM layers, the weights are updated based on these gradients to minimize the error, the loss function used to calculate error gradients in this work is mean squared error (MSE), the formula is as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_t - y_t^*)^2 \tag{3.8}$$

where y_t is the output, y_t^* is the target and N is the batch size.

LSTM and recurrent neural network shares several advantages in their application to sequential data modeling. Both LSTM and RNN are capable of capturing temporal dependencies and modeling dynamic patterns in sequences, they can handle variable-length sequences and are well-suited for tasks such as time series analysis, natural language processing, battery parameter and state estimation[9][28], and speech recognition.

One key advantage of LSTM over traditional RNNs is it mitigated the vanishing and exploding gradient problem introduced in previous sections by utilizing gates. The input, forget, output gates can regulate the flow of information and gradients carefully, allowing LSTMs to selectively retain or forget information in the cell state. Despite the advantages we mentioned above, LSTMs has a few shortcomings, it is more computationally expensive than traditional RNNs due to its complex architectures and operations, it require significant computational power and longer time to train, especially for large datasets and complex structures, LSTM models are prone to overfitting, particularly when training data is limited. Careful regularization and hyper-parameter tuning are necessary to mitigate this issue and ensure generalization.

3.6 Framework

In this work, experiment data was gathered by a in-house built battery test station as described in Section 1.2, the lithium-ion battery cell used in the experiment is a Nickle-Manganese-Cobalt-Oxide(NMC) pouch cell, the specifications are presented in Table 1.2. The temperature of the cell is monitored and controlled with a calorimeter at 35°C and different charging and discharging current was applied to the cell. Experiment data including current, terminal voltage and temperature were measured and recorded.

Data for the estimator comes from two sources: experiment data (I, V, T) gathered from test station is used by full order model to calculate battery states $(C_{s_{surf}}, C_{s_{avg}})$. The experiment data and battery states is then combined and used as data for LSTM. A flow chart of the training framework is presented in Figure 3.4.

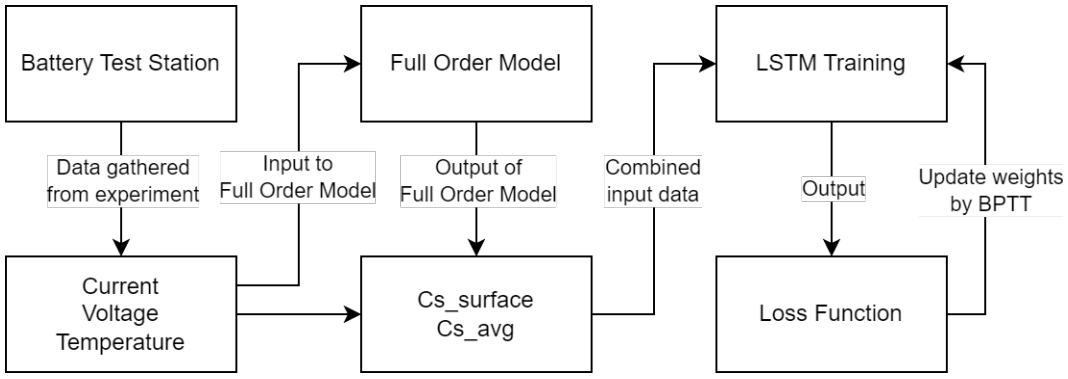

Figure 3.4: Diagram of training framework.

3.6.1 Data Preprocessing

This combination of experiment data and calculated battery states is then split into training, validation and testing datasets, two MinMaxScalers fit and transforms features by scaling each feature to a given range, in the case of this work, the range is carefully set to between -1 to 1 to prevent saturation of the activation functions within LSTM, the

transformation process of MinMaxScaler is as follows:

$$X_{std} = (X - X.min(axis = 0))/(X.max(axis = 0) - X.min(axis = 0)) \tag{3.9}$$

$$X_{scaled} = X_{std} * (max - min) + min \tag{3.10}$$

where $min, max = featurerange$ and X is the data used to compute the per-feature minimum and maximum used for later scaling along the features axis.

The LSTM model embedded within the estimator undergoes training using the training dataset, while the validation dataset is utilized to validate the hyper-parameters based on the model's performance. The results presented in this work are obtained from the testing dataset. It is crucial to exclude the validation set from the testing process to prevent bias in parameter selection. Including the validation set in testing would compromise the unbiased nature of the error evaluation. Thus, ensuring the use of an unseen test sample is essential to achieve accurate and unbiased results, as it ensures the model is evaluated on data it has not encountered during training or parameter selection.

Many precautions are taken to ensure the result is an accurate representation of the model. Data is split first before it is fitted to the training set and transformed by MinMaxScaler, this is to prevent any future information from inadvertently seeping into the model during the training process. The process of data preprocessing is depicted in Figure 3.5.

3.6.2 Design of neural network

In this work, a deep LSTM neural network is employed, this architecture with multiple layers and large hidden size provides the model with increased capacity to learn complex dependencies and capture intricate patterns in sequential data. The architecture of deep LSTM neural network used in this work is depicted in Figure 3.6. The deep LSTM neural network consists of 3 stacked LSTM to increase the depth of neural network.

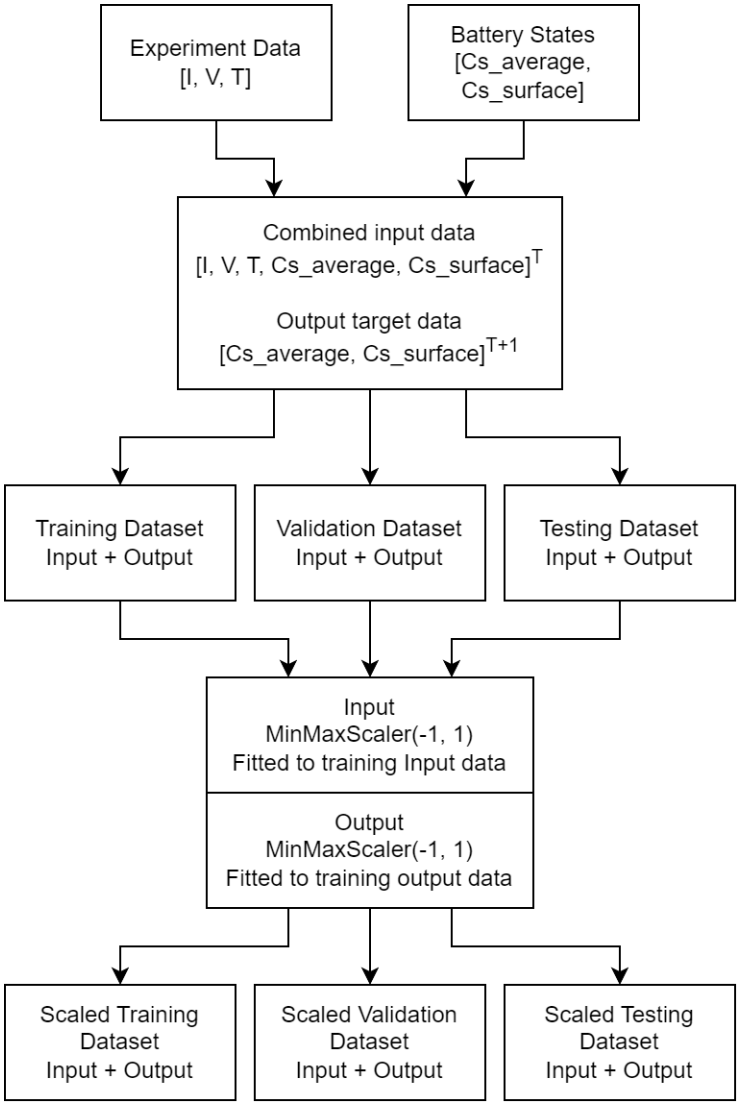

Figure 3.5: Process of data preprocessing.

The LSTM model described in this work has many hyper-parameters, they are determined empirically, three evaluation criteria were used to determine the performance of combination of hyper-parameters, and the hyper-perimeter used for results are listed in Table 3.1.

PyTorch, a machine learning framework [50], is used in this work to implement the neural network, the model is trained and tested on a desktop PC running windows 11 and python 3.11 with a Nvidia RTX 4090 GPU.

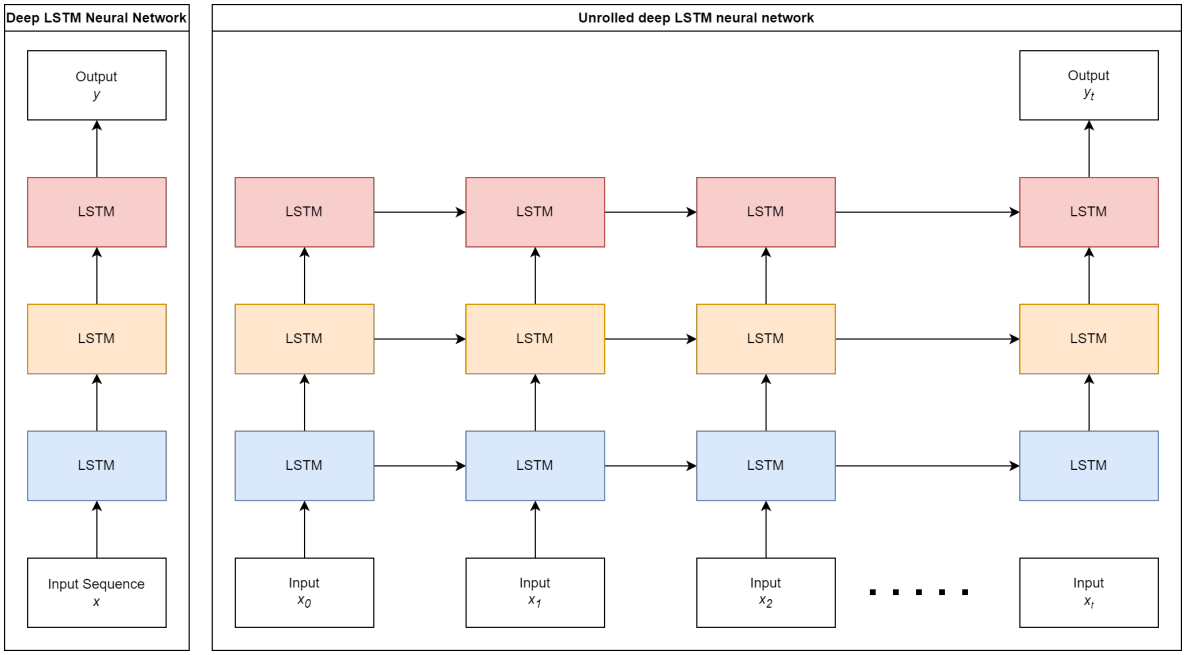

Figure 3.6: Diagram of Deep LSTM Neural Network

Hyper-parameter	Value
Loss function	MSE
Optimizer	Adam
Learning rate	0.0001
Max number of epochs	30000
Hidden size	512
Number of layers	3
Batch size	2048
Sequence length	100

Table 3.1: Hyper-parameters for LSTM

3.6.3 Model training and validation

During training and validation, each epoch consists of following steps:

- Custom DataLoader Class object generates Input data with and output target values for training. The Custom Data Loader class have batch training capability, can generate input sequences of variable length, and is able to shuffle at sequence level when sequence length is greater than 1.

- Batches of input sequences are fed into LSTM model to output predicted values, the predicted values are compared with target values through loss function (MSE), and the error gradients are used to update model weights and improve accuracy through back propagation through time (BPTT).

- After DataLoader exhausted all training data, the model moves onto validation. Validation is similar to training process where DataLoader generates input data sequences for model to perform prediction where it is compared with target value through loss function (MSE). Unlike training, the error gradient is not back propagated through time, instead the loss will be recorded as a metric to evaluate performance of model on unseen data.

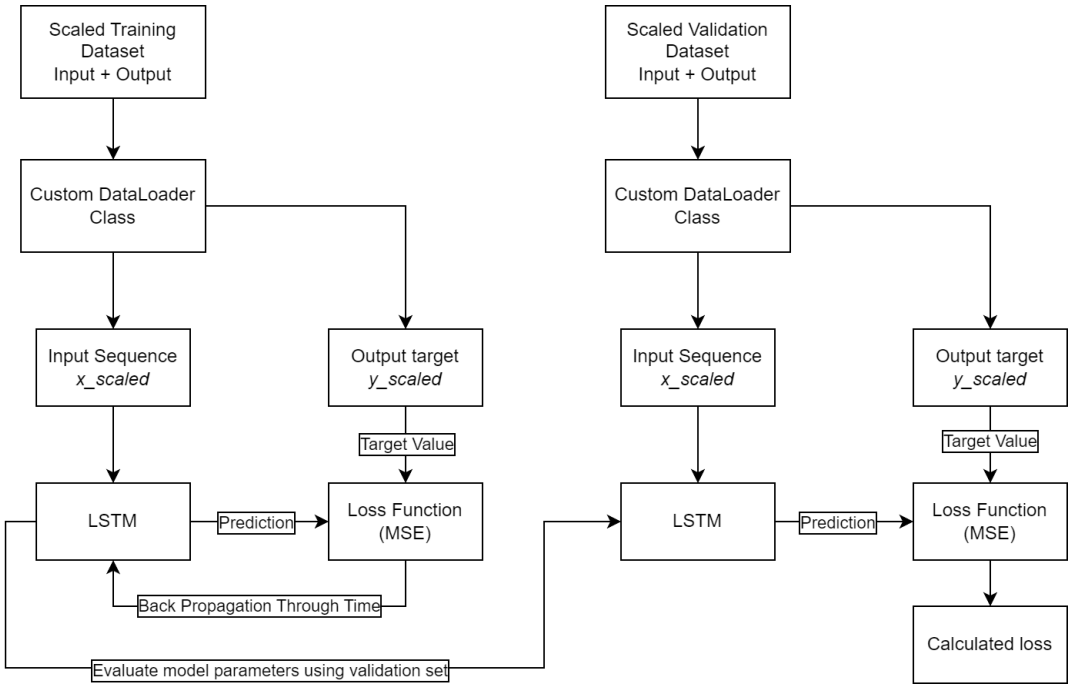

Figure 3.7: Process flow of training and validation.

The training and validation process is depicted in Figure 3.7. A technique known as Teacher Forcing is applied, where $C_{s_{surf}}^{T-1}$ and $C_{s_{avg}}^{T-1}$ from the input sequence are true target values instead of model's output, this prevented model to make predictions based on incorrect inputs, thus allowed the model to learn more effectively.

3.6.4 Model testing

During testing, the process is as follows:

- Scaled testing data is passed to a custom DataLoader class to generate input sequence for model.
- Trained LSTM model outputs predicted output. The predicted output is first used to calculate and record loss at each step with scaled target value. Then, the predicted output is inverse transformed to its original scale and used as input for next time step.
- The sequence of predictions will be plotted and compared with target value to test the performance of model when unseen data is encountered.

The testing process is depicted in Figure 3.8, note the output vector y is first inverse transformed from scaled values to original scale, then it is scaled by input scaler to be used as inputs for the next time step prediction. This process allows the model to generate a sequence of predictions over time, each one feeding into the next, all while ensuring the scale of the input data remains consistent with the scale used during training.

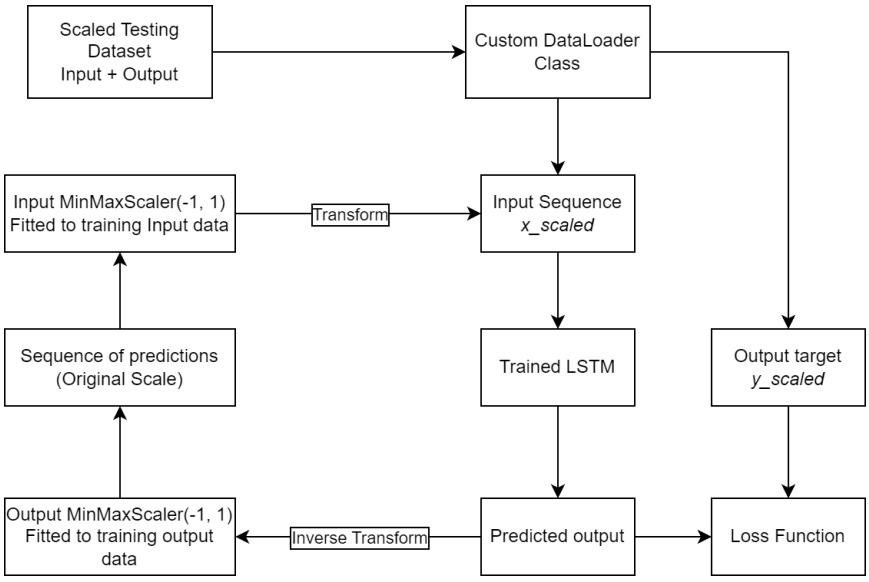

Figure 3.8: Process flow of Testing.

3.7 Results

The overall performance of data-driven estimator is evaluated in this section. A previous unseen set of driving cycle data containing both charging and discharging is measured in battery test station is used for this evaluation.

Current, voltage and temperature is measured and recorded during the experiment, the temperature of lithium-ion cell is monitored and controlled by a calorimeter at a constant 35°C throughout the experiment.

The experiment data is then used to calculate electrode surface concentration and average electrode concentration values by Full Order Model, where one particle in Anode is considered. The results of FOM is considered to be reference target value in this evaluation.

The data-driven estimator is also compared with the polynomial reduced order model (ROM), which is introduced in section 2.2.3.

The estimations made by data-driven estimator (LSTM) and polynomial ROM model is plotted against reference FOM, the errors between predicted value and reference value is calculated by four error metrics and results are also visualized and compared between data-driven estimator (LSTM) and polynomial ROM model.

The current and voltage data recorded during the experiment that's used in this evaluation is shown in Figure 3.9 and Figure 3.10 respectively.

3.7.1 Error metrics

Four error metrics are used in this evaluation to analyze the accuracy of data-driven estimator.

- Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \tag{3.11}$$

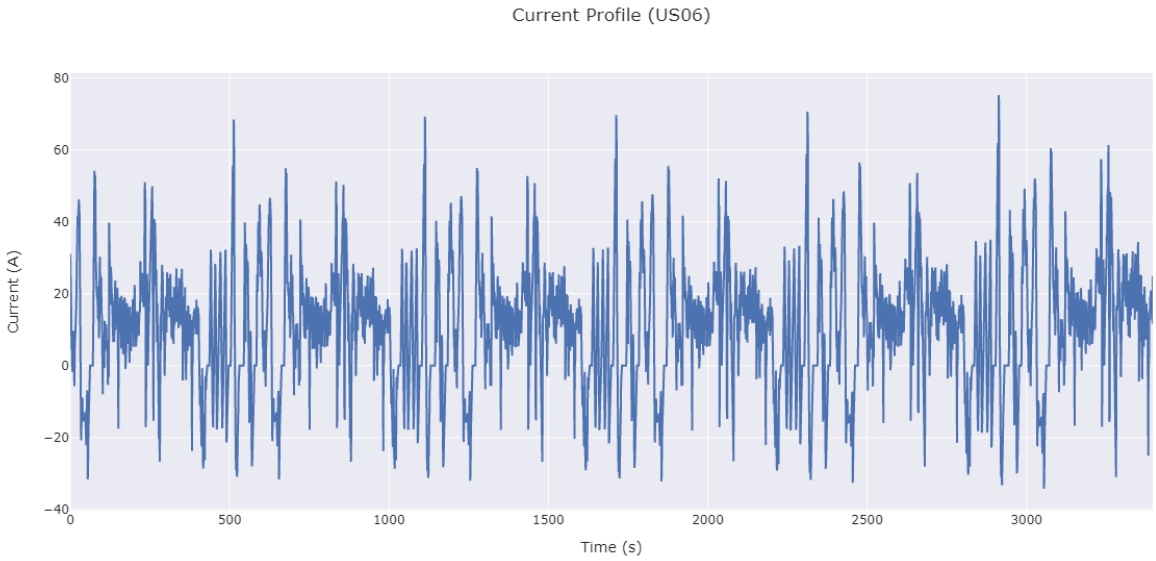

Figure 3.9: Current data used in evaluation.

where A_t is the actual target value and F_t is the estimated value, the sum of absolute percentage error of n items are divided by n.

- Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\sum_{t=1}^{n} \frac{(\hat{y}_t - y_t)^2}{n}} \tag{3.12}$$

where $\hat{y}_t$ is the estimated values and y_t are target values, n is the number of estimations made.

- Absolute Error (AE):

$$AE = |\hat{y}_t - y_t| \tag{3.13}$$

where $\hat{y}_t$ is the estimated values and y_t are target values.

- Mean Absolute Error (MAE):

$$MAE = \frac{\sum_{t=1}^{n} |\hat{y}_t - y_t|}{n} \tag{3.14}$$

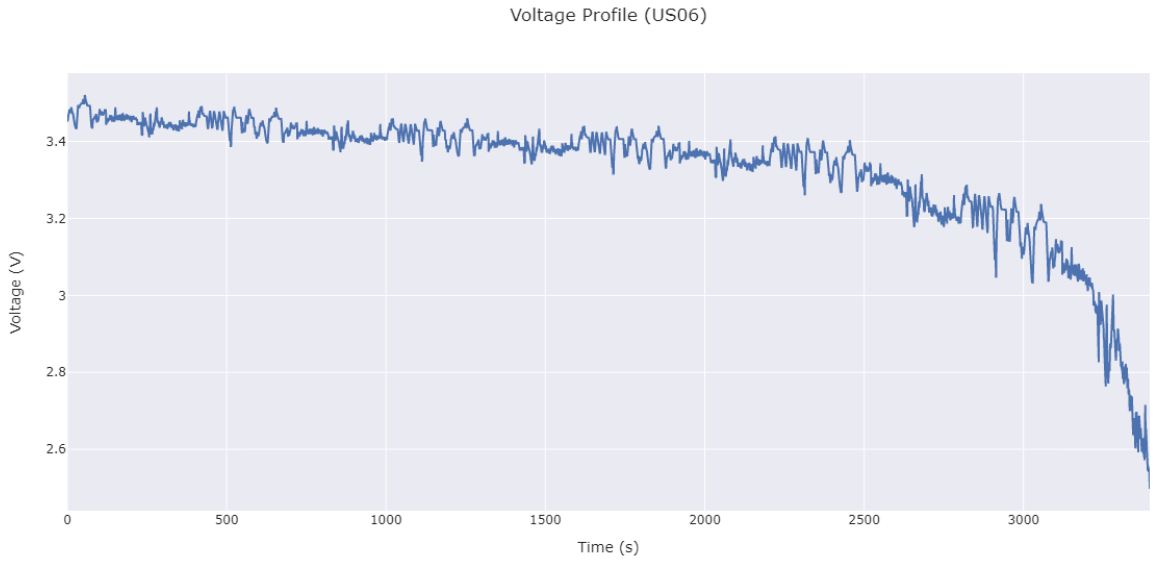

Figure 3.10: Voltage data used in evaluation.

where $\hat{y}_t$ is the estimated values and y_t are target values, n is the number of estimations made.

3.7.2 Evaluation for electrode surface concentration estimation

Estimation results of polynomial ROM for electrode surface concentration of single particle in anode during driving cycle is plotted against target values calculated by FOM, as shown in Figure 3.11.

From the plot we can observe that the estimations made by Polynomial ROM model is relatively consistent with the target values calculated by the FOM. Figure 3.12 shows the first 1000s of the estimations made by the polynomial ROM model. Detailed inspection of the plot shows that the Polynomial ROM model is not able to accurately estimate changes in surface concentration levels during charging where surface concentration increases.

The estimation results of data-driven LSTM model for electrode surface concentration of single particle in anode during driving cycle is also plotted against target values calculated by FOM, shown in Figure 3.13.

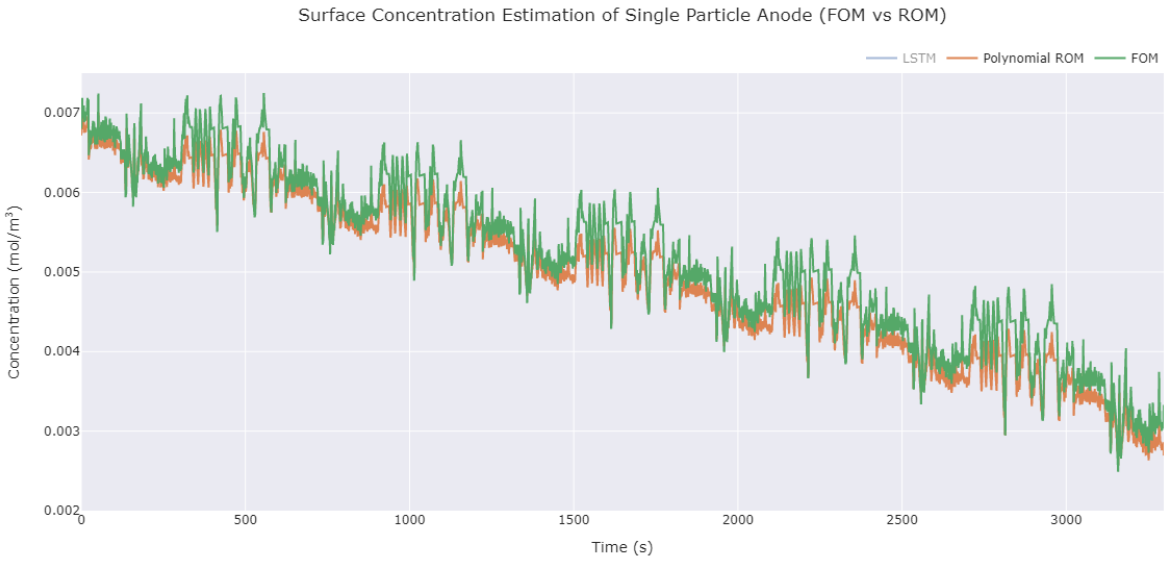

Figure 3.11: Estimations of electrode surface concentration of single particle anode by Polynomial ROM vs. FOM target values.

From the plot we can make the observation that the estimations made by data-driven LSTM model is also able to capture changes of electrode surface concentration. Figure 3.14, which shows the first 1000s of the estimations made by the data-driven LSTM model, proved that the model has superior performance in initial stages of estimation.

Figure 3.15 shows the surface concentration estimation with FOM, ROM and LSTM. We can observe the LSTM's result is higher than FOM while polynomial ROM is lower than FOM. The estimations made by LSTM is very accurate at start but starts to drift away from target value around 1500s mark.

Results of error metrics were shown in Figure 3.16, Figure 3.17, Figure 3.18 and Figure 3.19, following observations can be made.

1. Data-driven LSTM model has a initial advantage compared to polynomial ROM model. At the 1300s mark, data-driven LSTM model achieved a MAPE of around 1%, which is significantly better than the polynomial ROM model's 3.1% error at the same time point. This demonstrated the potential of data-driven LSTM model to offer precise predictions in the short term.

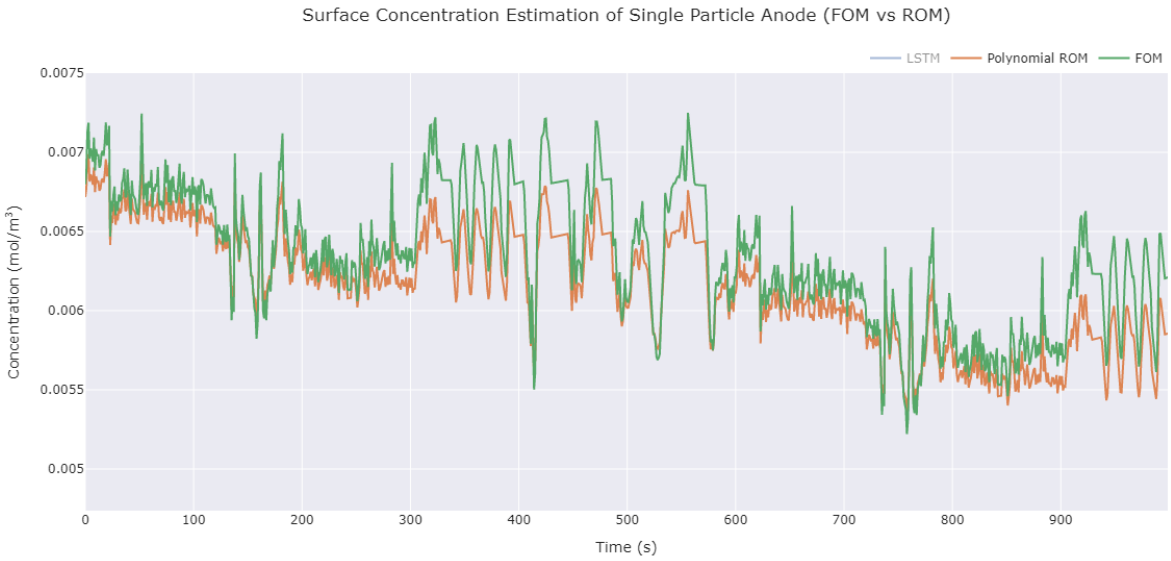

Figure 3.12: First 1000s of surface concentration estimations by Polynomial ROM vs. FOM target values.

2. The data-driven LSTM model's error increased over time. This issue is most likely caused by compounding error where the prediction of last time step is used as input for current time step. Despite the issue, data-driven LSTM model still outperforms the polynomial ROM model at the 2000s mark with a MAPE of 3.27% compared to polynomial ROM's 3.45%.

3. At the 3000s mark, the MAPE of data-driven LSTM model (6.99%) exceeds the polynomial ROM model's error (4.35%), and displayed a trend of continuous increasing as time increases.

The results of MAPE, RMSE, AE, and MAE showed the data-driven LSTM model have superior short-term performance, but as time increases, errors in prediction gets compounded and affects the accuracy of the model. On the other hand, polynomial ROM model have a slowly increasing but steady error curve, on a long term prediction scenario, polynomial ROM model will outperform data-driven LSTM model in accuracy.

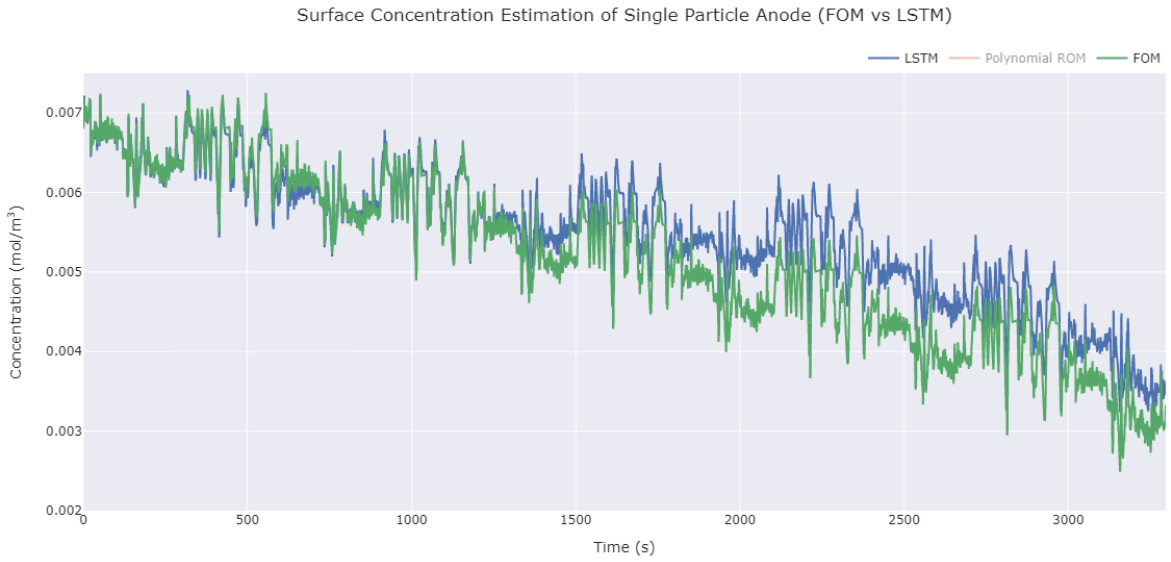

Figure 3.13: Estimations of electrode surface concentration of single particle anode by data-driven LSTM vs. FOM target values.

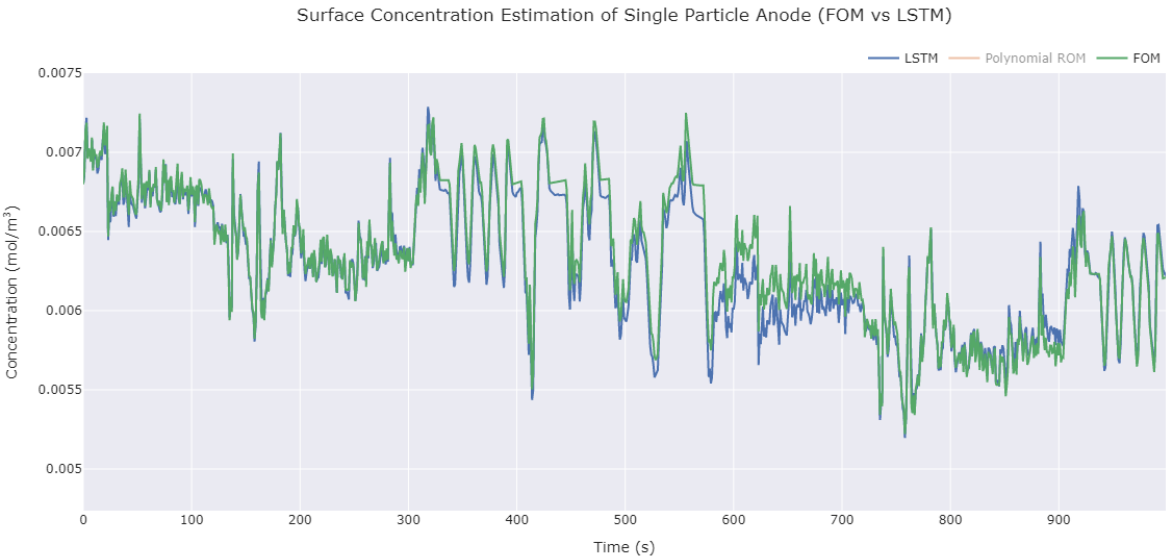

Figure 3.14: First 1000s of surface concentration estimations by data-driven LSTM vs. FOM target values.

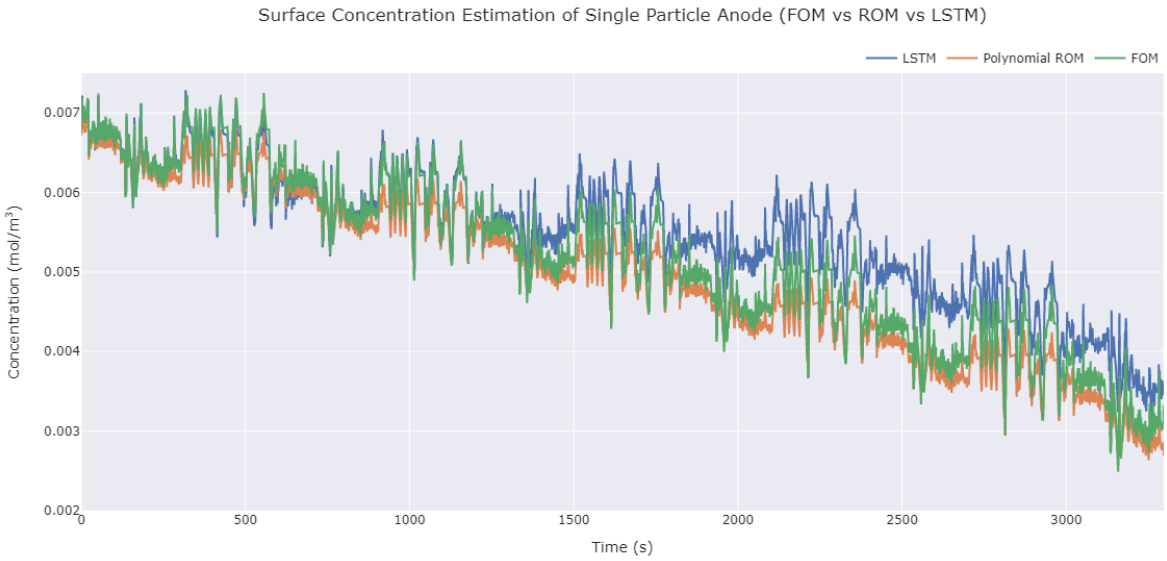

Figure 3.15: Surface concentration estimation of single particle anode by FOM vs. Polynomial ROM vs. LSTM.

3.7.3 Evaluation for average electrode concentration estimation

Estimation results of polynomial ROM for average electrode concentration of single particle in Anode during driving cycle is plotted against target values calculated by FOM, as shown in Figure 3.20.

The Polynomial ROM model's estimation have shown similar trends with the target values calculated by the FOM. Detailed inspection of the plot shows that the Polynomial ROM model have a very consistent error compared with target values, and it does not reflect minor changes in concentration well.

The estimation results of data-driven LSTM model for electrode surface concentration of single particle in Anode during driving cycle is also plotted against target values calculated by FOM, shown in Figure 3.21.

Figure 3.22 shows the surface concentration estimation with FOM, ROM and LSTM. We can observe the LSTM's result is higher than FOM while polynomial ROM is lower than FOM. The estimations made by LSTM is very accurate at start but starts to drift away from target value around 1500s mark.

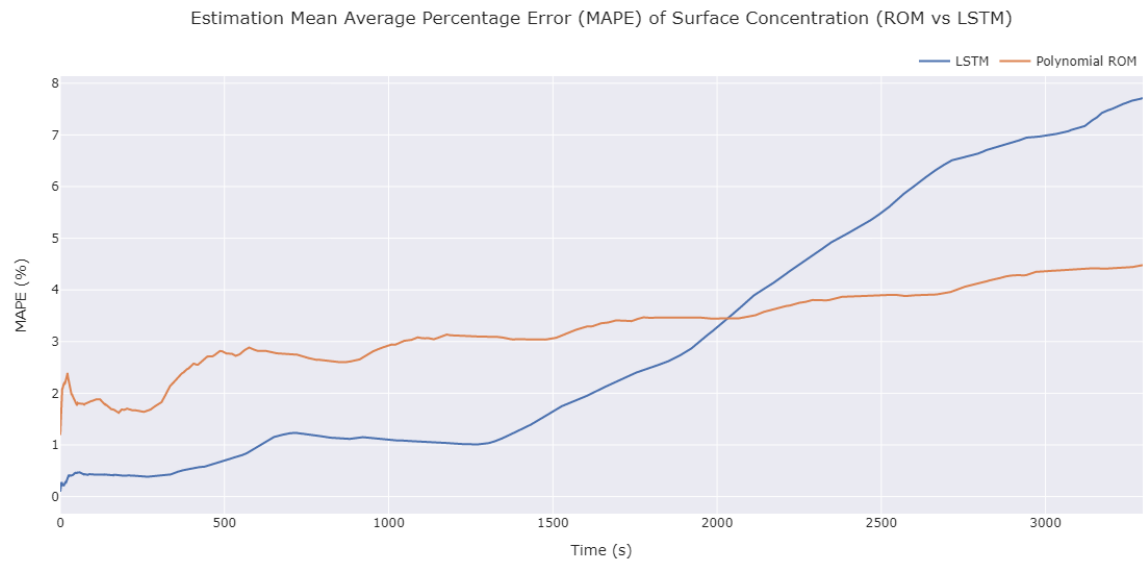

Figure 3.16: MAPE of surface concentration (ROM vs LSTM)

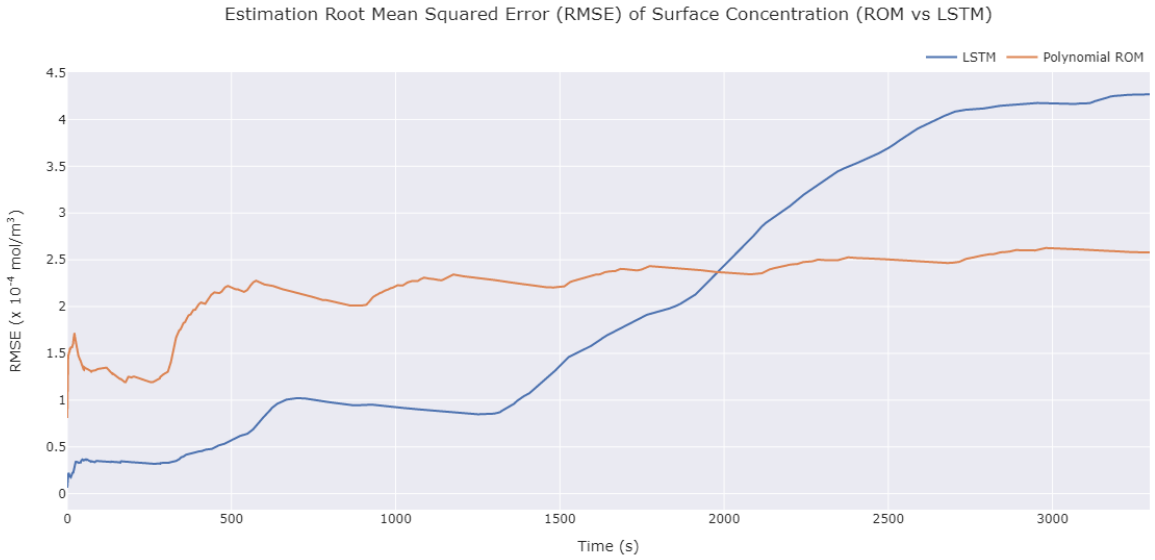

Figure 3.17: RMSE of surface concentration (ROM vs LSTM)

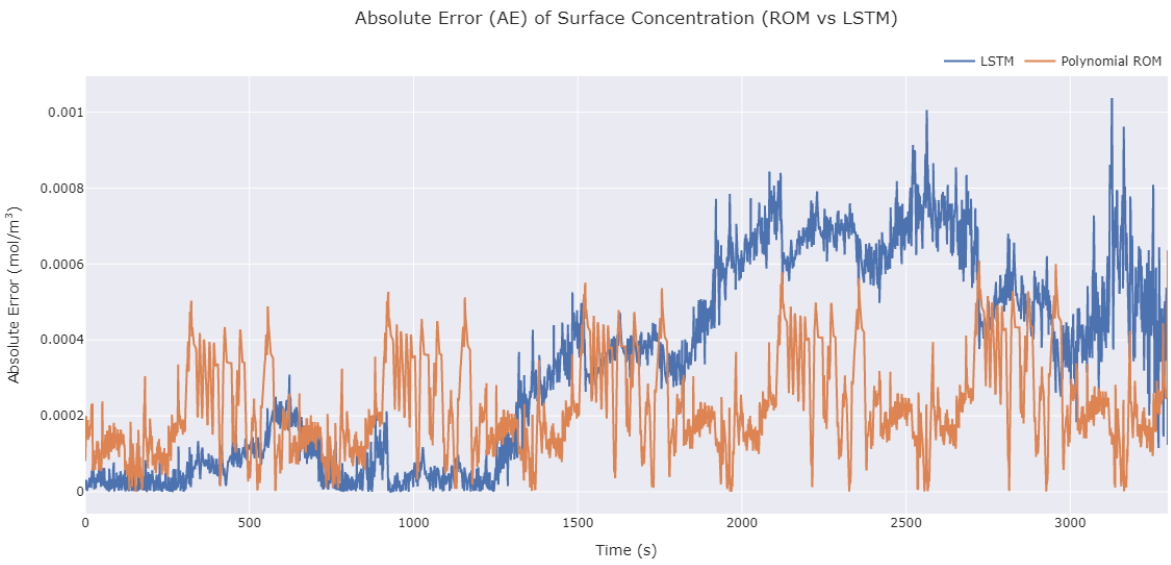

Figure 3.18: AE of surface concentration (ROM vs LSTM)

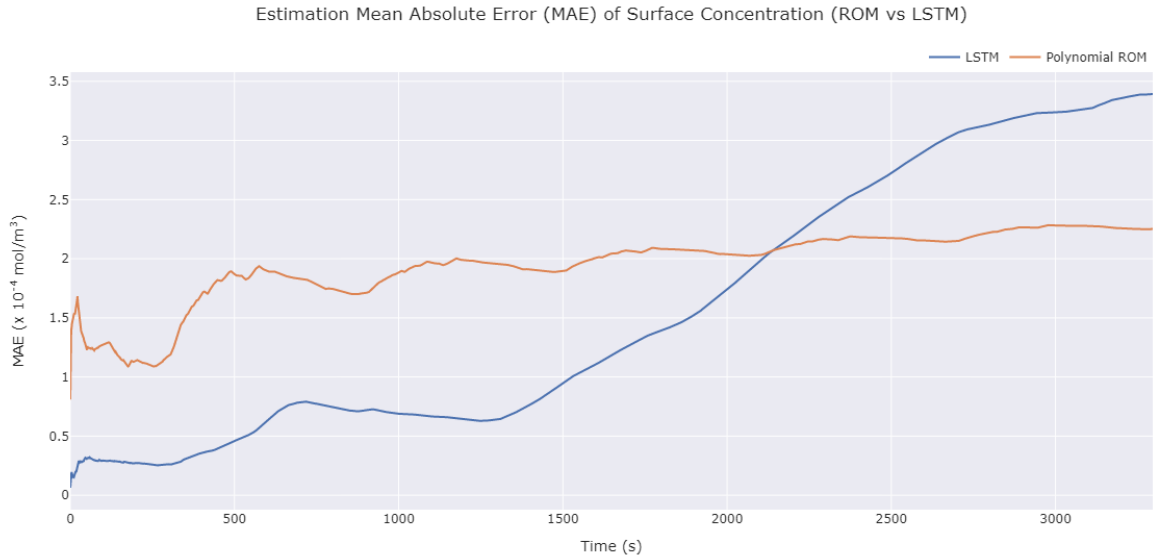

Figure 3.19: MAE of surface concentration (ROM vs LSTM)

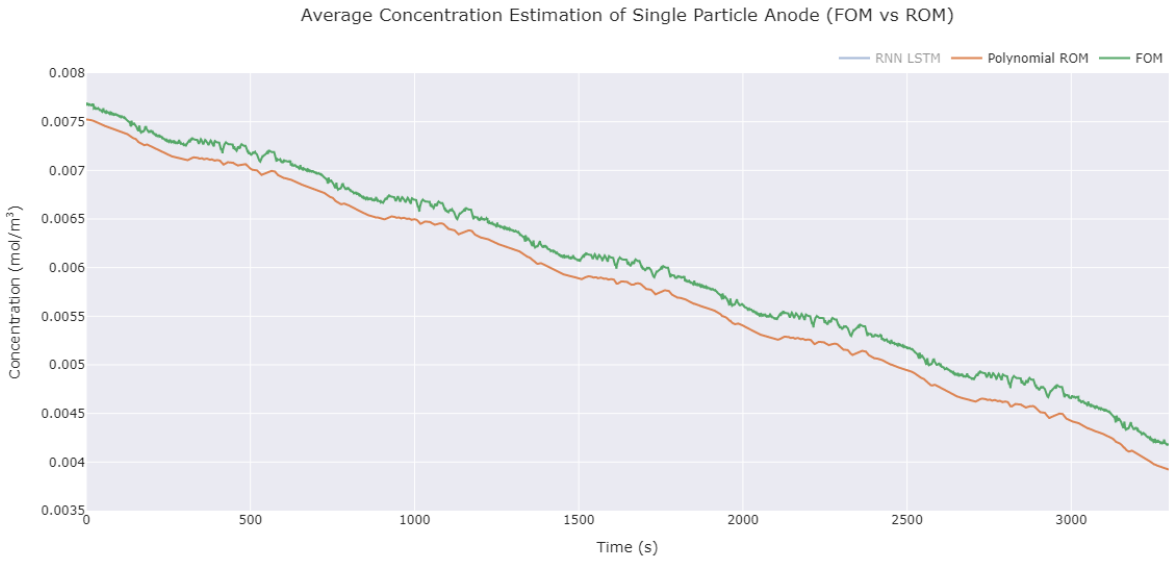

Figure 3.20: Estimations of average electrode concentration of single particle anode by Polynomial ROM vs. FOM target values.

Similar to surface concentration estimation, data-driven LSTM model have a great initial performance as the estimations are very close to the target value. As time progresses the estimations starts to shift away from target values but was able to reduce error near the 2800s mark.

Results of error metrics for average concentration prediction were shown in Figure 3.23, Figure 3.24, Figure 3.25 and Figure 3.26, following observations can be made.

1. Data-driven LSTM model has a initial advantage compared to polynomial ROM model. At the 730s mark, data-driven LSTM model has a MAPE of 1.5%, while polynomial ROM model has a MAPE of 2.2%.

2. At 1300s mark, data-driven LSTM model's MAPE decrease to 1.1%, which is out performed the polynomial ROM model's 2.48% error at the same time point.

3. At 2000s mark, data-driven LSTM model achieved a MAPE of 2.55%, outperforming the polynomial ROM model's MAPE of 2.8%.

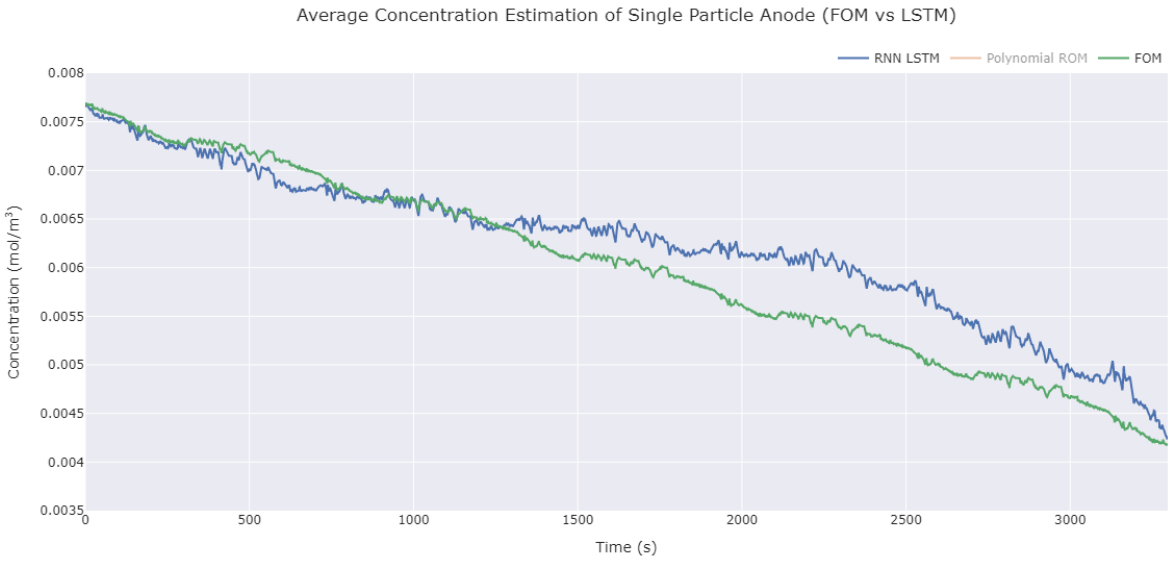

Figure 3.21: Estimations of average electrode concentration of single particle anode by LSTM vs. FOM target values.

4. At the 3000s mark, data-driven LSTM model has a MAPE of 5.16%, higher than polynomial ROM model's 3.4%.

Results from MAPE, RMSE, AE, and MAE further confirmed that the data-driven LSTM model is better at short-term prediction, initial predictions have a significantly lower error compared with the polynomial ROM model, while polynomial ROM model outperforms data-driven LSTM model in long term estimations. Both models' error increase over time, but polynomial ROM model accumulates error at a much slower rate compared with data-driven LSTM model.

3.7.4 Computation speed

During testing of data-driven LSTM model, the computation time of each time step is recorded and presented in Table 3.2. It is worth noting that the SPM model and polynomial ROM's computation time is not compared in this work due to different programming environments.

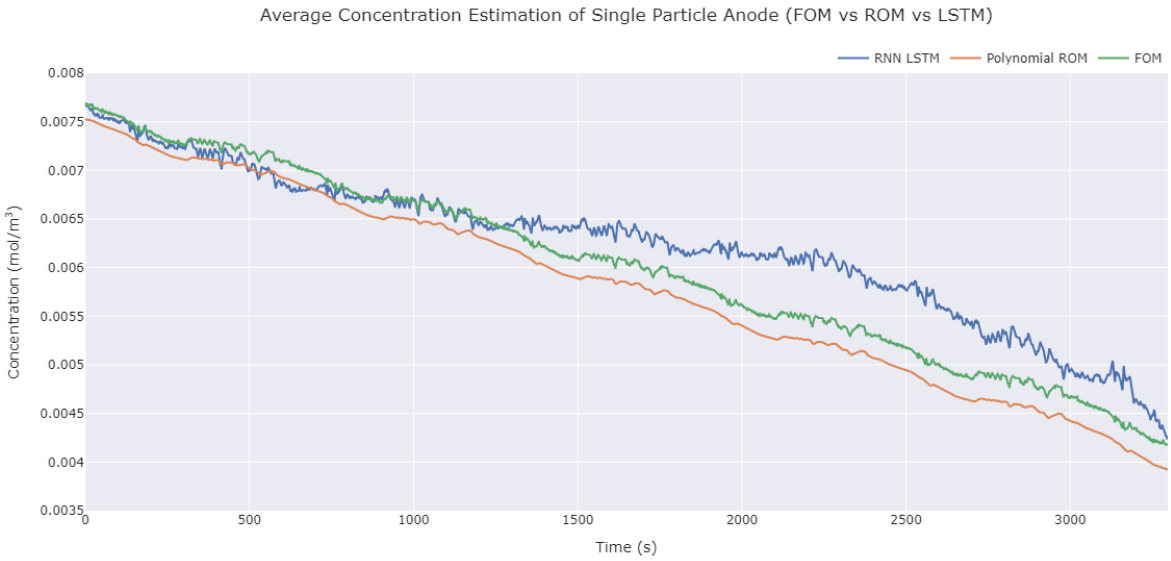

Figure 3.22: Average concentration estimation of single particle anode by FOM vs. Polynomial ROM vs. LSTM.

1. Prediction time:

   This is the amount of time taken for LSTM neural network to make a prediction based on input data.

2. Total time:

   This is the amount of time taken to inverse transform model output into its original scale plus the aforementioned prediction time.

The results presented is the average computation of 5 tests.

	Average	Max	Min
Prediction time	1.249ms	1.877ms	1.190ms
Total time	4.654ms	20.262ms	4.003ms

Table 3.2: Computation time for data-driven LSTM model.

It has shown that the computation time for data-driven LSTM model is very fast, capable of real-time applications.

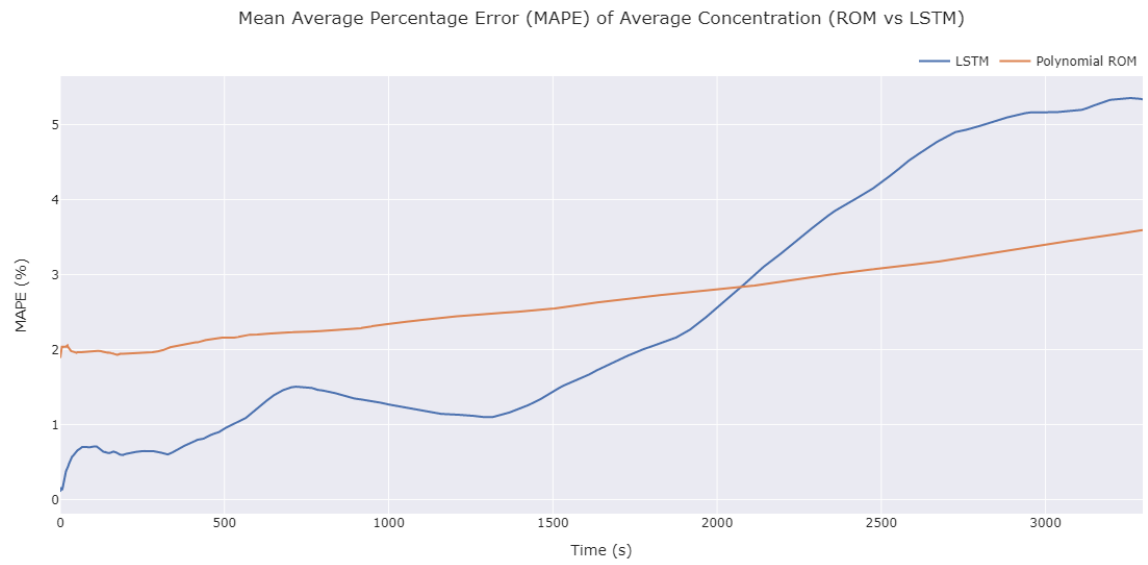

Figure 3.23: MAPE of average concentration (ROM vs LSTM)

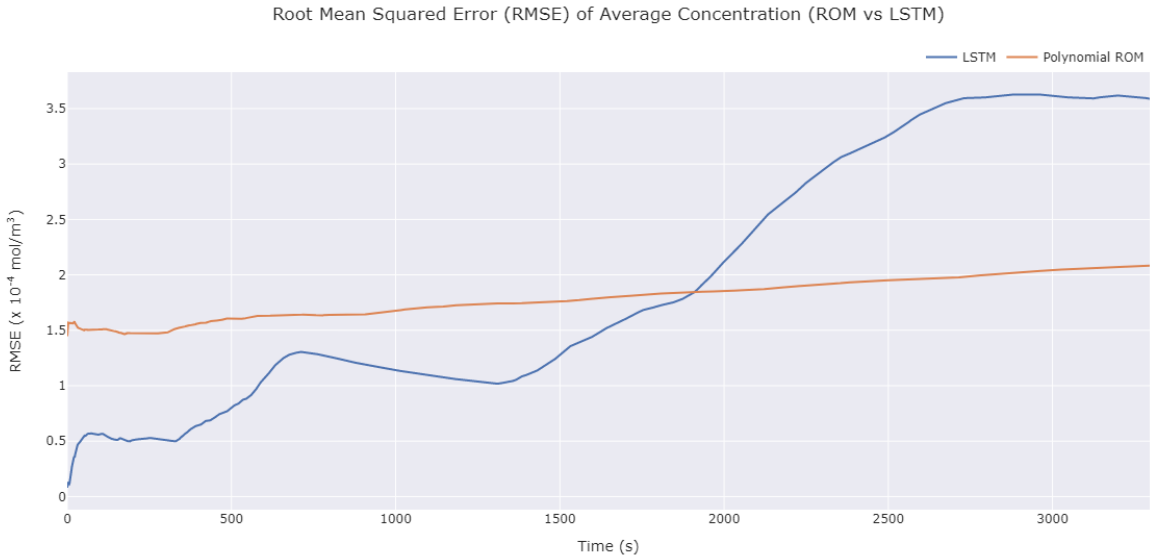

Figure 3.24: RMSE of average concentration (ROM vs LSTM)

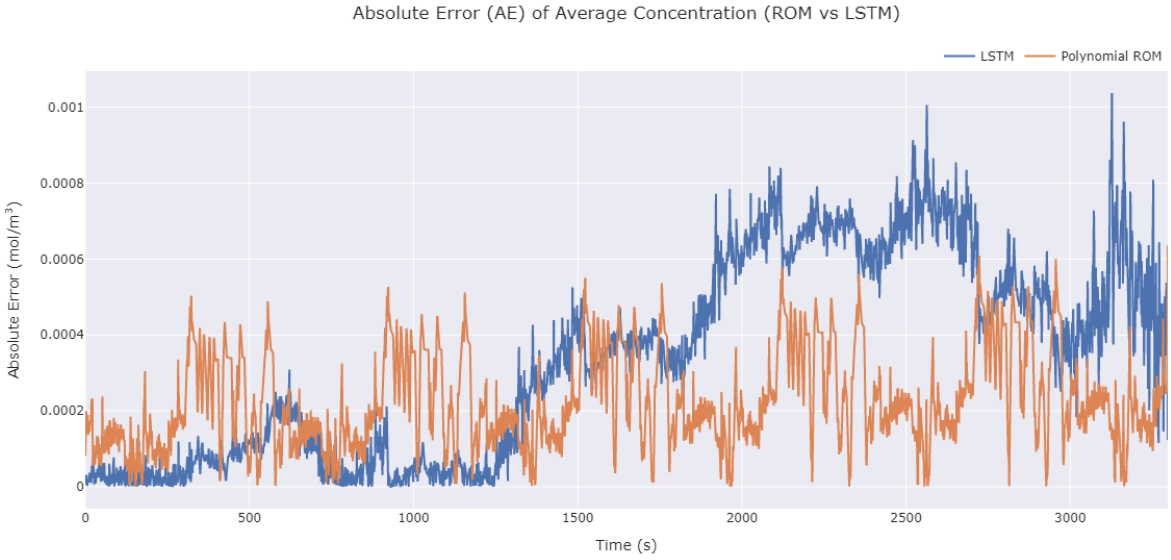

Figure 3.25: AE of average concentration (ROM vs LSTM)

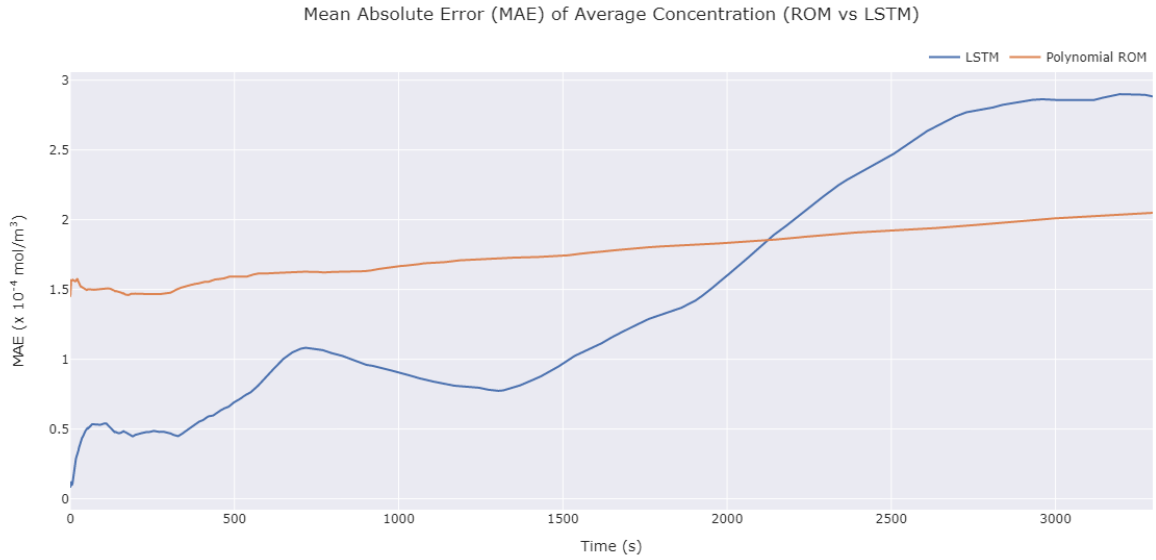

Figure 3.26: MAE of average concentration (ROM vs LSTM)

3.8 Summary

The first significant performance characteristic of the LSTM-based, data-driven model developed in this thesis was its initial superior accuracy. Specifically, the model achieved an impressive Mean Absolute Percentage Error (MAPE) of 1.09% at 1000s for estimation of surface concentration for single particle anode, and 1.27% at 1000s for estimation of average concentration for single particle anode, a notable improvement over the 2.93% and 2.34% error exhibited by the polynomial Reduced Order Model (ROM) at the same time point. This initial performance strongly demonstrated the potential of LSTM models for providing precise short-term predictions in the field of lithium-ion concentration estimation.

However, as time progressed, errors within the LSTM model began to increase. Despite this trend, it's crucial to note that the model still outperformed the polynomial ROM at the 2000-second mark, delivering an error rate of 3.27% against the ROM's 3.45% for surface concentration estimation and 2.55% against polynomial ROM's 2.80% for average concentration estimation. At the 3000-second mark, the LSTM model's error did exceed that of the ROM, registering at 6.99% compared to the ROM's 4.35% for surface concentration and 5.16% compared to the ROM's 3.40%. Although this indicates an area of improvement for the LSTM model in long-term prediction accuracy, the performance still highlighted the LSTM's competitive potential against traditional polynomial ROM models.

In summary, the LSTM-based, data-driven model developed in this work demonstrated robust performance in short-term predictions, with competitive long-term results against the polynomial ROM model. Its capacity for further performance improvement, coupled with its dynamic learning ability and flexibility, renders the LSTM model a promising approach for estimating average electrode concentration and surface concentration in lithium-ion cells.

3.8.1 Analysis of Prediction Error Increase and Potential Improvements

As the results shown, while LSTM based data-driven model excels at short-term predictions, the long-term accuracy is out performed by polynomial ROM model. In this subsection this issue is investigated, and potential improvement idea is proposed.

Recall in Section 3.3 where a many to one autoregressive forecasting model is selected based on the constraints of the problem. This is a common strategy for multi-step time series forecasting known as Recursive Strategy[51]. Because predictions are used in place of observations, the recursive strategy allows prediction errors to accumulate, leading to a rapid decrease in performance as the prediction time horizon increases.

Direct Strategy is another major approach to multi-step time series forecasting[48]. The strategy uses different forecasting models for each forecast horizon, because this strategy uses different models for prediction, there's no opportunity to model the dependencies between the predictions. This strategy also increases computational and maintenance burden as the time horizon increases. Considering the shorting comings of this strategy, it is not a viable option for this work.

Summing up, the model we proposed in this work is trained to predict one time step into the future, but the evaluation process considers the long term multi step accuracy of the model, which is not what the model is trained for. Various constraints, such as the lack of observations throughout the time horizon and the necessity for predictions to be made at relatively short intervals, also significantly impact its long-term accuracy.

One of the possible improvement idea is to find the optimal prediction horizon of recursive strategy, that is, the time range within which the model can deliver precise predictions. Once the time horizon has surpassed the optimal prediction horizon, the prediction should be restarted to ensure the model's accuracy. The drawback of this approach is the implementation may be difficult since accurate input data is required to restart prediction.

Another possible approach to improve the long term performance of the model is to utilize a technique known as residual learning. In this case we can develop a separate model

to predict residual error of the LSTM estimator and correct future forecasts, a high-level approach is as follows:

1. Estimator Training: LSTM estimator is trained with training data to perform recursive multi-step predictions, and the residual error (difference between model's prediction and actual target value) is calculated and kept as part of training data for residual model.

2. Residual Model Training: A second residual model is trained using the same input data used in LSTM estimator training, instead predicting target values, the residual model tries to predict the residual error recorded during LSTM estimator training process.

3. Prediction: During prediction, the estimator makes the prediction based on the input, this input is also used by the residual model to predict residual error of estimator model. The predicted residual error can be used to correct initial prediction made by estimator, results in potential improvement of model's accuracy for both short-term and long-term predictions.

In conclusion, while our LSTM-based model provides satisfactory short-term predictions, its long-term accuracy could be enhanced. The problem mainly arises from the inherent accumulation of errors in recursive forecasting. By leveraging techniques such as optimizing the prediction horizon or applying residual learning, we believe the model's long-term forecasting capabilities can be further improved.

3.9 Applications

In this section some potential applications of the LSTM estimator is proposed.

3.9.1 Short-term voltage prediction

Follow up on the concentration estimation, a potential hybrid model application is tested. The output of the LSTM estimator $C_{s_{surf}}$ and $C_{s_{avg}}$ is used in ROM model to

perform short-term voltage prediction, this application could be useful in battery control applications.

For this test, two input datasets were used: Driving Cycle (US06) data is used in testing and Constant Current Discharge at 0.3C over 11000s. The results are plotted against Experimental data, FOM calculated voltage and ROM estimated voltage.

The results of driving cycle data is shown in Figure 3.27. Estimations made with Polynomial ROM's concentration is in orange trace, estimations made with LSTM's concentration is in blue trace and FOM calculated terminal voltage is in green trace, the experimental terminal voltage is shown in dashed red line. Based on the plot we can observe that the results of ROM and LSTM is very close, but voltage estimation based on LSTM is consistently greater and closer to the experiment data and FOM result.

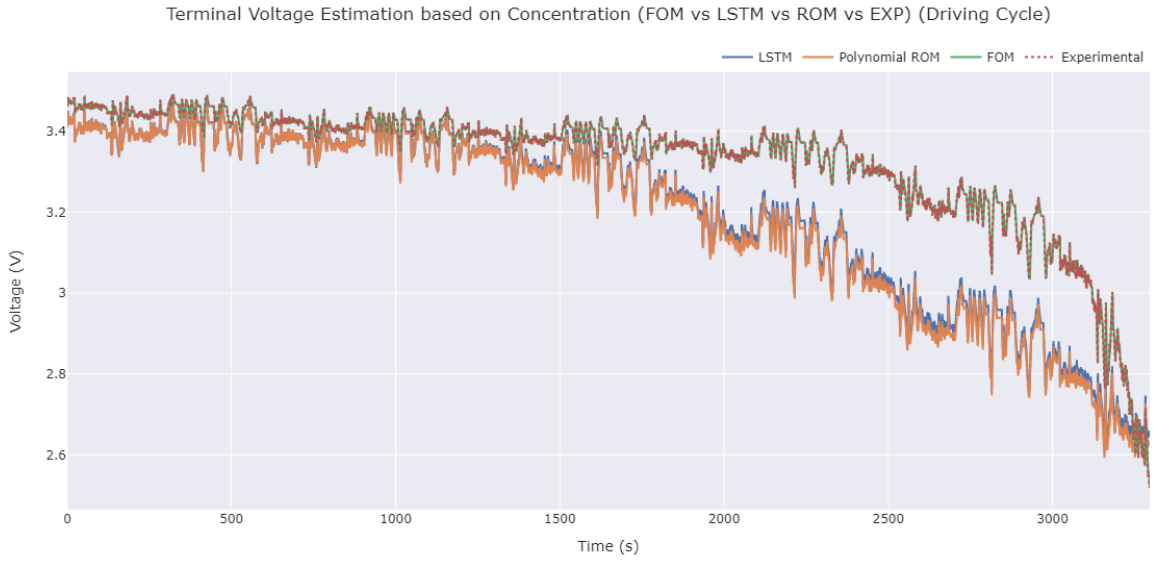

Figure 3.27: Terminal Voltage Estimation based on Concentration (Driving Cycle).

The results of CC discharging data is shown in Figure 3.28. Estimations made with Polynomial ROM's concentration is in orange trace, estimations made with LSTM's concentration is in blue trace and FOM calculated terminal voltage is in green trace, the experimental terminal voltage is shown in dashed red line. Similar observations to driving cycle data

can be made where estimations based on LSTM is close to estimations made with ROM, but it is closer to FOM and experimental results after 8000s.

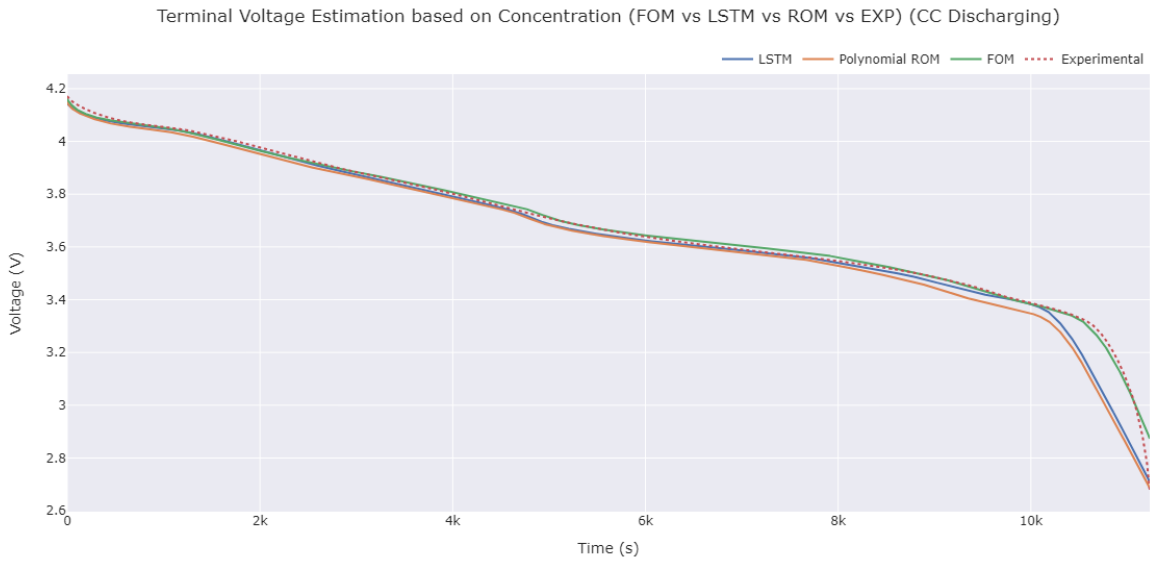

Figure 3.28: Terminal Voltage Estimation based on Concentration (CC Discharging).

Based on the results from datasets above, we can make the conclusion that the LSTM-ROM voltage estimation performed better than ROM model.

Recall in previous tests the ROM's results are consistently lower compared to FOM while LSTM is generally greater, this difference in prediction may contributed to the results shown in this test.

3.9.2 State of Charge Estimation

Another potential application is to estimate the cell SOC based on the output $C_{s_{avg}}$ of the LSTM estimator.

Consider the maximum theoretical concentration of lithium in electrode particle is $C_{s_{max}}$ and the theoretical concentration when the particle is completely empty of lithium is 0, with LSTM estimated average concentration of Li in particle at time k: $C_{s_{avg,k}}$, we can calculate

the lithium stoichiometry θ_k at time k:

$$\theta_k = \frac{C_{s_{avg,k}}}{C_{s_{max}}} \tag{3.15}$$

The lithium stoichiometry should be a value between 0 and 1, this reflects the amount of lithium present in the electrode and can be used to calculate cell SOC. The cell SOC often is within the range of the stoichiometry to prevent potential overcharging or overdischarging of the battery, which can be described by:

$$\theta_{0\%} > 0 \text{ and } \theta_{100\%} < 1 \tag{3.16}$$

where the $\theta_{0\%}$ and $\theta_{100\%}$ represents 0% and 100% state of charge respectively.

Therefore we can calculate cell SOC z_k as follows:

$$\begin{aligned} z_k &= \frac{\theta_k^{neg} - \theta_{0\%}^{neg}}{\theta_{100\%}^{neg} - \theta_{0\%}^{neg}} \\ &= \frac{\theta_k^{pos} - \theta_{0\%}^{pos}}{\theta_{100\%}^{pos} - \theta_{0\%}^{pos}} \end{aligned} \tag{3.17}$$

Note we can use both negative and positive electrode stoichiometry for state of charge calculation. For positive electrode stoichiometry, $\theta_{0\%}^{pos} > \theta_{100\%}^{pos}$. Therefore in the positive electrode part of Equation 3.17, both numerator and denominator of the equation would be negative values, resulting in positive positive state charge.

Chapter 4

Conclusion

This final chapter aims to consolidate the major findings of this thesis and highlight potential avenues for further exploration. The chapter is divided into two sections: Section 4.1 outlines the possible areas for enhancing and expanding the proposed LSTM-based lithium-ion concentration estimator. Section 4.2 summarizes the key achievements of this work and discusses their implications.

4.1 Future work

While the data-driven LSTM-based lithium-ion concentration estimator presented in this thesis has demonstrated commendable accuracy and computational efficiency in the short term prediction, it has also revealed certain areas where further enhancements can be pursued. The following items indicate the potential future directions:

1. The LSTM model proposed in this work excels at short-term estimations but suffers at long-term estimations. Different training techniques and regularization methods can be explored in future work to improve the long-term performance of the model.

2. Hybrid models that incorporate both data-driven and physics-based approaches could be explored to leverage the advantages of both paradigms.

3. Given its quick computation time, potential real-world applications for the LSTM model should be explored. This could include deployment within battery management systems for electric vehicles or renewable energy storage systems.

4.2 Concluding Remarks

In this thesis, we have developed, implemented, and validated a data-driven estimator for lithium-ion concentration. Comprehensive details regarding the design, implementation, and training process have been elaborated, along with the various steps undertaken to augment the accuracy and reliability of our test results. By employing a stacked LSTM architecture, we have endowed our neural network with an increased depth, thus enabling it to discern intricate relationships within the cell.

Our proposed estimator has been critically evaluated and contrasted with the Polynomial Reduced Order Model (ROM), a commonly employed approach in battery modeling. The results demonstrate that our model exhibits significant promise in delivering swift and precise concentration estimations. For instance, in surface concentration testing, our estimator attained a Mean Absolute Percentage Error (MAPE) of 1% at the 1000s mark, escalating to a still impressive 3.29% at the 2000s mark. The model also delivered commendable results in average concentration testing, with a MAPE of 1.27% and 2.56% at the 1000s and 2000s marks respectively. Furthermore, the rapid computation time of our model lends itself perfectly to real-time applications.

In conclusion, this thesis presents a comprehensive exploration of lithium-ion cell backgrounds and architectures, an in-depth review of existing lithium-ion battery modeling and state estimation techniques, and a systematic classification and comparison of contemporary battery models. Following the literature review, we propose a data-driven methodology for estimating lithium-ion concentration at the electrode surface and the average electrode concentration. The design, implementation, and training processes are meticulously explained and demonstrated. Finally, the proposed methodology is validated and compared with the polynomial ROM, offering promising prospects for future research and real-world applications.

Bibliography

[1] Saehong Park, Dong Zhang, and Scott Moura. Hybrid electrochemical modeling with recurrent neural networks for li-ion batteries. In *2017 American Control Conference (ACC)*, pages 3777–3782, 2017.

[2] Bharat Balagopal and Mo-Yuen Chow. Effect of anode conductivity degradation on the thevenin circuit model of lithium ion batteries. 10 2016.

[3] Ramon Quiza and J. Davim. *Computational Methods and Optimization*, pages 177–208. 01 2011.

[4] K. Mizushima, P.C. Jones, P.J. Wiseman, and J.B. Goodenough. Lixcoo2 (0¡x¡-1): A new cathode material for batteries of high energy density. *Materials Research Bulletin*, 15(6):783–789, 1980.

[5] Jakob Asenbauer, Tobias Eisenmann, Matthias Kuenzel, Arefeh Kazzazi, Zhen Chen, and Dominic Bresser. The success story of graphite as a lithium-ion anode material – fundamentals, remaining challenges, and recent developments including silicon (oxide) composites. *Sustainable Energy Fuels*, 4:5387–5416, 2020.

[6] Tanvir R. Tanim, Christopher D. Rahn, and Chao Yang Wang. State of charge estimation of a lithium ion cell based on a temperature dependent and electrolyte enhanced single particle model. *Energy*, 80:731–739, February 2015.

[7] Wei He, Michael Pecht, David Flynn, and Fateme Dinmohammadi. A physics-based electrochemical model for lithium-ion battery state-of-charge estimation solved by an optimised projection-based method and moving-window filtering. *Energies*, 11(8), 2018.

[8] Maricela Best Mckay, Brian Wetton, and R. Bhushan Gopaluni. Learning physics based models of lithium-ion batteries. *IFAC-PapersOnLine*, 54(3):97–102, 2021. 16th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2021.

[9] Huiyong Chun, Jungsoo Kim, Jungwook Yu, and Soohee Han. Real-time parameter estimation of an electrochemical lithium-ion battery model using a long short-term memory network. *IEEE Access*, 8:81789–81799, 2020.

[10] Weihan Li, Jiawei Zhang, Florian Ringbeck, Dominik Jöst, Lei Zhang, Zhongbao Wei, and Dirk Uwe Sauer. Physics-informed neural networks for electrode-level state estimation in lithium-ion batteries. *Journal of Power Sources*, 506:230034, 2021.

[11] Dickshon N. T. How, Mahammad A. Hannan, Molla S. Hossain Lipu, Khairul S. M. Sahari, Pin Jern Ker, and Kashem M. Muttaqi. State-of-charge estimation of li-ion battery in electric vehicles: A deep neural network approach. *IEEE Transactions on Industry Applications*, 56(5):5565–5574, 2020.

[12] M.S. Hossain Lipu, M. A. Hannan, Aini Hussain, M.H.M. Saad, A. Ayob, and K. M. Muttaqi. Lithium-ion battery state of charge estimation method using optimized deep recurrent neural network algorithm. In *2019 IEEE Industry Applications Society Annual Meeting*, pages 1–9, Sep. 2019.

[13] Fangfang Yang, Weihua Li, Chuan Li, and Qiang Miao. State-of-charge estimation of lithium-ion batteries based on gated recurrent neural network. *Energy*, 175:66–75, 2019.

[14] Arnab Bhattacharjee, Ashu Verma, Sukumar Mishra, and Tapan K. Saha. Estimating state of charge for xev batteries using 1d convolutional neural networks and transfer learning. *IEEE Transactions on Vehicular Technology*, 70(4):3123–3135, April 2021.

[15] Ali Ghorbani Kashkooli, Hamed Fathiannasab, Zhiyu Mao, and Zhongwei Chen. Application of artificial intelligence to state-of-charge and state-of-health estimation of calendar-aged lithium-ion pouch cells. *Journal of The Electrochemical Society*, 166(4):A605, feb 2019.

[16] Yohwan Choi, Seunghyuong Ryu, Kyungnam Park, and Hongseok Kim. Machine learning-based lithium-ion battery capacity estimation exploiting multi-channel charging profiles. *IEEE Access*, PP:1–1, 06 2019.

[17] Sheng Shen, Mohammadkazem Sadoughi, Xiangyi Chen, Mingyi Hong, and Chao Hu. A deep learning method for online capacity estimation of lithium-ion batteries. *Journal of Energy Storage*, 25:100817, 2019.

[18] Giulia Crocioni, Danilo Pau, Jean-Michel Delorme, and Giambattista Gruosso. Li-ion batteries parameter estimation with tiny neural networks embedded on intelligent iot microcontrollers. *IEEE Access*, 8:122135–122146, 2020.

[19] Mahammad A. Hannan, Dickson N. T. How, Muhamad Bin Mansor, Md S. Hossain Lipu, Pin Jern Ker, and Kashem M. Muttaqi. State-of-charge estimation of li-ion battery using gated recurrent unit with one-cycle learning rate policy. *IEEE Transactions on Industry Applications*, 57(3):2964–2971, May 2021.

[20] Isaiah Oyewole, Abdallah Chehade, and Youngki Kim. A controllable deep transfer learning network with multiple domain adaptation for battery state-of-charge estimation. *Applied Energy*, 312:118726, 2022.

[21] M. A. Hannan, D. N. T. How, M. S. Hossain Lipu, Pin Jern Ker, Z. Y. Dong, M. Mansur, and Frede Blaabjerg. Soc estimation of li-ion batteries with learning rate-optimized deep fully convolutional network. *IEEE Transactions on Power Electronics*, 36(7):7349–7353, July 2021.

[22] Fei Feng, Sangli Teng, Kailong Liu, Jiale Xie, Yi Xie, Bo Liu, and Kexin Li. Co-estimation of lithium-ion battery state of charge and state of temperature based on a hybrid electrochemical-thermal-neural-network model. *Journal of Power Sources*, 455:227935, 2020.

[23] Gae won You, Sangdo Park, and Dukjin Oh. Real-time state-of-health estimation for electric vehicle batteries: A data-driven approach. *Applied Energy*, 176:92–103, 2016.

[24] Duo Yang, Yujie Wang, Rui Pan, Ruiyang Chen, and Zonghai Chen. A neural network based state-of-health estimation of lithium-ion battery in electric vehicles. *Energy Procedia*, 105:2059–2064, 2017. 8th International Conference on Applied Energy, ICAE2016, 8-11 October 2016, Beijing, China.

[25] Phattara Khumprom and Nita Yodo. A data-driven predictive prognostic model for lithium-ion batteries based on a deep learning algorithm. *Energies*, 12:660, 02 2019.

[26] Shuangqi Li, Hongwen He, Pengfei Zhao, and Shuang Cheng. Health-conscious vehicle battery state estimation based on deep transfer learning. *Applied Energy*, 316:119120, 2022.

[27] Kirandeep Kaur, Akhil Garg, Xujian Cui, Surinder Singh, and Bijaya Ketan Panigrahi. Deep learning networks for capacity estimation for monitoring soh of li-ion batteries for electric vehicles. *International Journal of Energy Research*, 45(2):3113–3128, 2021.

[28] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G. Pecht. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology*, 67(7):5695–5705, 2018.

[29] Seongyoon Kim, Yun Young Choi, Ki Jae Kim, and Jung-Il Choi. Forecasting state-of-health of lithium-ion batteries using variational long short-term memory with transfer learning. *Journal of Energy Storage*, 41:102893, 2021.

[30] Weihan Li, Neil Sengupta, Philipp Dechent, David Howey, Anuradha Annaswamy, and Dirk Uwe Sauer. One-shot battery degradation trajectory prediction with deep learning. *Journal of Power Sources*, 506:230024, 2021.

[31] Jichao Hong, Zhenpo Wang, Wen Chen, Leyi Wang, Peng Lin, and Changhui Qu. Online accurate state of health estimation for battery systems on real-world electric vehicles with variable driving conditions considered. *Journal of Cleaner Production*, 294:125814, 2021.

[32] Yaxiang Fan, Fei Xiao, Chaoran Li, Guorun Yang, and Xin Tang. A novel deep learning framework for state of health estimation of lithium-ion battery. *Journal of Energy Storage*, 32:101741, 2020.

[33] Joonki Hong, Dongheon Lee, Eui-Rim Jeong, and Yung Yi. Towards the swift prediction of the remaining useful life of lithium-ion batteries with end-to-end deep learning. *Applied Energy*, 278:115646, 2020.

[34] Sheng Shen, Mohammadkazem Sadoughi, Meng Li, Zhengdao Wang, and Chao Hu. Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries. *Applied Energy*, 260:114296, 2020.

[35] Weihan Li, Decheng Cao, Dominik Jöst, Florian Ringbeck, Matthias Kuipers, Fabian Frie, and Dirk Uwe Sauer. Parameter sensitivity analysis of electrochemical model-based battery management systems for lithium-ion batteries. *Applied Energy*, 269:115104, 2020.

[36] Marc Doyle, Thomas F. Fuller, and John Newman. Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell. *Journal of The Electrochemical Society*, 140(6):1526, jun 1993.

[37] Xueyan Li and Song Choe. State-of-charge (soc) estimation based on a reduced order electrochemical thermal model and extended kalman filter. pages 1100–1105, 06 2013.

[38] James Marcicki, Marcello Canova, A. Conlisk, and Giorgio Rizzoni. Design and parametrization analysis of a reduced-order electrochemical model of graphite/lifepo4 cells for soc/soh estimation. *Journal of Power Sources*, 237:310–324, 09 2013.

[39] Venkat R. Subramanian, Vinten D. Diwakar, and Deepak Tapriyal. Efficient macro-micro scale coupled modeling of batteries. *Journal of The Electrochemical Society*, 152(10):A2002, aug 2005.

[40] Xiaosong Hu, Shengbo Li, and Huei Peng. A comparative study of equivalent circuit models for li-ion batteries. *Journal of Power Sources*, 198:359–367, 01 2012.

[41] Wladislaw Waag, Christian Fleischer, and Dirk Uwe Sauer. Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles. *Journal of Power Sources*, 258:321–339, 07 2014.

[42] Lyu Li, Yu Peng, Yuchen Song, and Datong Liu. Lithium-ion battery remaining useful life prognostics using data-driven deep learning algorithm. In *2018 Prognostics and System Health Management Conference (PHM-Chongqing)*, pages 1094–1100, 2018.

[43] Weijiang Feng, Naiyang Guan, Yuan Li, Xiang Zhang, and Zhigang Luo. Audio visual speech recognition with multimodal recurrent neural networks. pages 681–688, 05 2017.

[44] Amirthalakshmi Veeraraghavan, Viswa Adithya Balakrishnan, Ajinkya Bhave, and Shankar Akella. Battery aging estimation with deep learning. pages 1–4, 12 2017.

[45] Y. Le Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. *Handwritten Digit Recognition with a Back-Propagation Network*, page 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[46] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[47] Olatomiwa Badmos, Andreas Kopp, Timo Bernthaler, and Gerhard Schneider. Image-based defect detection in lithium-ion battery electrode using convolutional neural networks. *Journal of Intelligent Manufacturing*, 31, 04 2020.

[48] Rohitash Chandra, Shaurya Goyal, and Rishabh Gupta. Evaluation of deep learning models for multi-step ahead time series prediction. *IEEE Access*, PP:1–1, 05 2021.

[49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

[50] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[51] Gianluca Bontempi, Souhaib Ben Taieb, and Yann-Aël Le Borgne. *Machine Learning Strategies for Time Series Forecasting*, pages 62–77. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.