Green Storage System Performance Evaluation and Prediction Using Machine Learning

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By

Leo Guo, Janaarthana Harri Palanisamy, Gabrielle Viray

05/2023

APPROVED

Dr. Jerry Gao, Project Advisor

[Program Director's Name], Director, MS Software Engineering

[Department Chair's Name], Department Chair

ABSTRACT

Green Storage System Performance Evaluation and Prediction Using Machine Learning
By Leo Guo, Janaarthana Harri Palanisamy, Gabrielle Viray

In today's fast-paced world, energy is a quintessential need for numerous different private, commercial and industrial sectors to lead their day-to-day activities. As the energy demand grows there is a lot of excitement in the shift towards cleaner electricity. Battery Energy Storage Systems (BESS) are one such technology that enables renewable energy to be stored and released when needed. The battery energy storage market size is valued at $9.21 billion in 2021 and is estimated to grow from $10.88 billion in 2022 to $31.20 billion by 2029, exhibiting a CAGR of 16.3%.

Battery management systems play a crucial role in many present-day devices/applications and can be seen everyday with the increase of electric vehicles(EVs) and smart grids. These systems perform various functions such as measuring battery temperature, monitoring battery health, and also monitoring battery remaining charge. In addition, battery management systems prevent battery degradation caused by overcharging or over-discharging. Furthermore, accurately predicting State of Health (SoH), State of Charge (SoC), and Remaining Useful Life (RUL) remains a crucial yet arduous task in ensuring the batteries' performance, lifetime, and safety operations.

A web application that uses data driven machine learning methods to accurately predict different features of the battery is proposed in this project. Various public datasets are analyzed to learn useful information for the estimation of specific features of the battery and to evaluate the performance of different battery types. The web application also allows the user to explore the datasets of different battery types by using the preloaded datasets or by uploading their own dataset to the web app. A dashboard with the prediction result and other data will be displayed to the user as per their choice of dataset and feature they choose.

Acknowledgments

The authors are deeply indebted to Professor Jerry Gao for his invaluable comments and assistance in the preparation of this study.

Table of Contents

List of Figures

List of Tables

Chapter 1. Project Overview

1.1 Introduction

The global energy storage systems market is rapidly growing due to increased demand in numerous sectors like transportation, IT, and finance. The need for clean and renewable energy will likely propel the market further as it is an affordable and efficient energy resource alternative to fossil fuel. However, capital investment and maintenance restrain the battery energy storage systems market. On the flip side, the increase in the electric vehicle sector and recent investments in the power infrastructure network are expected to create tremendous opportunities in the coming years. The global energy storage systems market was valued at 211.24 GW in 2021 and is increasing at a CAGR of 11% from 2022 to 2030. The global market generated $188.5 billion in 2020 and is rising at a CAGR of 8.3%, estimated to reach $435.4 billion by 2030.

Energy is essential for everyday activities across numerous different sectors to function. The increasing demand for electricity and the rising ozone-harming greenhouse discharges has led organizations to adopt low-carbon emission frameworks while generating energy. Battery Energy Storage systems (BEES) are one such sector that significantly creates innovations that use battery chemistries to store energy to serve the demand. The battery market is segmented based on the type, lithium-ion batteries, lead-acid batteries, nickel batteries, flow batteries, and others. The battery energy storage market size is valued at $9.21 billion in 2021 and is estimated to grow from $10.88 billion in 2022 to $31.20 billion by 2029, exhibiting a CAGR of 16.3%. Based on geographical locations the battery market is categorized across four main regions, Asia Pacific, Europe, North America, and the Rest of the world. The Asia Pacific is leading the market with an estimated $4.16 billion in 2021 and has the potential to share the major share in terms of volume and value in the coming years. As Asia Pacific, America, and Europe dominate the battery market, ML and AI have emerged as promising tools to accelerate the rate of innovation. AI has been experimented with overcoming long evaluation periods, optimizing battery structure, and more effective materials discovery. Due to the increasing use of portable electronics, the consumer electronics industry is expected to proliferate in the coming years. AI in the renewable energy sector was valued at $8.67 billion in 2021 and is estimated to reach $75.82 billion by 2030, exhibiting a CAGR of 27.9%. With AI and ML playing a key role in the renewable energy sector the battery market will have to increase its focus on launching intelligent battery systems to improve battery performance and achieve a better ROI.

In this project, we will discuss improving the performance of green storage systems, focusing on lithium-ion batteries by estimating State-of-Health (SoH), State-of-Charge (SoC), and Remaining Useful Life (RUL) using different Machine Learning and Deep Learning techniques. With the battery's market into consideration accurately predicting SoH, SoC, and RUL helps in the smooth, safe, reliable, and uniform functioning of the battery management systems.

1.2 Proposed Areas of Study and Academic Contribution

The global energy demand is increasing rapidly due to the fast economic growth, large population, and innovative developments in the market. Fossil fuels are still the most dependable source of energy to meet the rising demand. But, Renewable energy consumption can be an alternative source to conserve fossil fuels and reduce global warming [4]. As the use of renewable energy is increasing, the need for better energy storage systems is required. Lithium-ion (Li-ion) batteries are recently have been one of the leading technologies for energy storage systems for applications like electric vehicles (EVs), electric ships, portable devices, and aerospace industries [4] [5]. Though the lithium-ion battery possesses high specific energy, high specific power, and long life it is generally an electrochemical system that comes with challenges and concerns. Li-ion batteries will degrade and deteriorate gradually during their use or shelving [5]. When the degradation reaches a specific capacity (70 - 80%), it can cause battery operation problems and even failure [4]. Studying and accurately estimating the lithium-ion batteries' State-of-Health (SoH), State-of-Charge (SoC) and Remaining Useful Life (RUL) can significantly improve the performance and reliability of the batteries. This section will briefly discuss the various promising attempts to accurately predict the SOH, SOC, and RUL of a li-ion battery using Machine Learning methods.

The State of Health is defined as the ratio of current capacity to its designated capacity; it is an estimate to quantify the health of the battery [2]. Some of the common methods to estimate the SoH are the Internal resistance or impedance method, Measuring energy on a full charge, Kalman filters, and Electrochemical methods [2]. However, these methods required human intervention to estimate the SoH. On the contrary, machine learning methods require minimal human intervention where different charging profiles (CPs) like the voltage, current, and temperature are measured over regular intervals to form a dataset. Using the dataset different ML models are trained to learn the underlying patterns of battery degradation and tested on new unseen data. Consequently, the machine learning-based estimation of SoH comes with its challenges. To tackle issues faced by conventional methods that cause incorrect battery health estimation, Khan et al [3]

proposed Multiple Channel CPs (MCCPs) based Battery Management System (BMS) to estimate the SoH through deep learning concepts. [3] investigates both machine learning and deep learning methods which include adaptive boosting, support vector regression, long short-term memory (LSTM), multi-layer perceptron, bi-direction LSTM, and conventional neural network to approach the battery's SoH estimation. ML models require large training data to achieve a good result. A flexible SoH prediction scheme for lithium-ion batteries using a predefined voltage range with LSTM and transfer learning (TL) to constitute a cell mean model with partial training data is discussed in [1]. A novel walk-forward algorithm is proposed using gradient boosting regression for SoH estimation which provided better results with little prior information about the batteries [6]. Huotari et al [6] compared seven different machine learning methods with multiple predictors for each method to derive the best model.

The State of Charge is defined as the amount of charge in a battery relative to its capacity. SoC estimation can help know the discharge capacity and plan the battery's usage [2]. Two traditional methods to estimate SoC are the terminal voltage method and the coulomb counting method [2]. However, these techniques have their limitations and are being displaced by more sophisticated methods that are generally computationally intensive [7]. A new method using LSTM-RNN is proposed that estimates the SoC accurately without any battery models, filters, or inference systems by learning from the datasets taken under different ambient conditions [7]. Though machine learning-based methods are adopted in the estimation of SoC, they sometimes cause the phenomenon of abrupt errors. [8] discusses a physics-constrained neural network; a combined data model that significantly reduces abrupt errors and also has the capability to integrate other ML-based methods. To tackle challenges in SoC estimation like complex parameter identification and complex internal structure of the battery, Zhao et al introduced a new input feature called voltage increment for the existing SoC estimation algorithms that could significantly improve the accuracy [9]. Huang et al [10] suggest deep learning approaches are well suited for SoC estimation as battery management systems are time-varying and non-linear [10]. [11] proposed a deep learning-based dual-stage attention mechanism with a gated recurrent unit-based encoder-decoder network that improves the accuracy and reduces the influence of noise on SoC estimation.

Lithium-ion batteries have been an integral part of our daily life, especially in the automotive sector due to their high energy and power capabilities. Yet, the precise lifetime prediction of the battery remains unresolved [12]. To estimate the lifespan of a battery, an artificial recurrent neural network architecture with LSTM is proposed considering each cell voltage, load voltage, temperature, and charge-discharge cycle [13].

With advances in deep learning, predicting RUL can be a crucial role in intelligent battery health management systems. Ren et al [14] proposed a deep learning approach that integrates autoencoder with deep neural networks (DNN) on the NASA dataset; the result showed significant improvement in RUL prediction. Early lifetime prediction of the lithium-ion battery will ensure safety, and reliability and accelerate the battery development lifecycle [15]. Fei et al proposed a machine-learning framework that analyses various parameters, selects optimal low-dimensional feature subsets, and feeds the features into six representative machine-learning models to predict the battery lifetime [15].

1.3 Current State of the Art

The SoH of a battery is evaluated by the deterioration in relation to a brand-new battery. In a BMS, SoH data is important in sustaining the efficiency and safety of a system (i.e. Vehicle) by modifying its controls within the specified ranges. SoH can be quantified by the decline in capacity and the rise in internal resistance[16]. There are several different approaches to predicting battery SoH in battery management systems (BMS). Various machine learning techniques have been surveyed for SoH estimation [16]. These algorithms include Feedforward Neural Network (FNN), Recurrent Neural Network (RNN), Radial Basis Function (RBF) Neural Network, Hamming Networks (HNN), Support Vector Machine (SVM), and Bayesian Network (BN). Based on the study, most of the suggested techniques are strong options for SoH estimation. Results showed a capacity estimation of less than 1% [16]. In addition, FNN w/ k-means and RNN (LSTM) provided the best results based on multi-temperature datasets [16].

In real-world applications of BMS, battery SoC is analogous to a fuel gauge in cars running on gasoline. It is important to carry out an indirect assessment of the SOC through estimation in order to ascertain the proportion of usable energy still present in the battery. This is accomplished using a wide range of approaches and methodologies that make use of observable signals such as battery terminal voltage, current, and temperature [16]. Several ML SoC estimation techniques have been surveyed: FNN, RBF, Extreme Learning Machine (ELM), SVM, and RNN. Results have shown RBFs having the highest error with FNNs and RNNS having lower average errors on multi-temperature datasets [16]. Thus, FNNs and RNNs are highlighted as being the most optimistic algorithms. More studies need to be done with ELM w/ AUKF and SVM since the datasets used to build the models are single temperature [16].

Forecasting the battery's remaining useful life is a crucial step in ensuring its dependability and safety. Battery capacity varies with time. When the capacity reaches the point of failure, a burst could occur, thus, RUL prediction is vital. In [17], machine learning techniques such as classic NN, modern NN, SVM, RVM, GPR, and model-based methods have been contrasted based on principle according to battery RUL. NNs are highly accurate and suitable for complex, non-linear behaviors. This method is effective for estimating battery remaining useful life due to the varying operating conditions of batteries. On the downside, NNs require more time consumption, large training set, and more storage for memory [17]. RVM has shown to be more effective than SVM for probabilistic forecasting and can further be enhanced using a gradual approach for online learning [17].

Chapter 2. Project Architecture

2.1 Introduction

The project will be implemented in the form of a web application. The project aims to provide a cloud training platform for users to use and receive immediate in-depth analysis of their data. The users are able to upload their battery data or select our preloaded data, and select an available model to make predictions and in-depth analysis of the battery. The data uploaded should be a csv file, with a specified format. The available model will depend on the wanted prediction (a selection from state of health, remaining life cycle, charge capacity, state of charge). Then the model will make a prediction. The data will be analyzed along with the prediction result, and fed back to the users.

2.2 Architecture Subsystems

2.2.1 Data Ingestion and Storage

Data ingestion and storage is needed for users to upload and choose battery datasets they would like to retrieve a performance analysis and prediction on. It is the process for importing and storing large data files into a cloud-based storage medium such as a database or data warehouse. For our project, we utilize MongoDB as our backend database for user login and verification. The user is able to upload their battery dataset through the front-end. The web application utilizes AWS SDK which provides API for interacting with AWS3 and enables uploading, downloading, and managing files in the S3 bucket.

The project uses Amazon web services(AWS) to run the machine learning models in the cloud. The application is also deployed using AWS.

- AWS S3 to store the uploaded data for analysis and estimation.
- AWS EFS to upload Machine learning models in the file system.
- AWS Lamba to trigger the ML model code.
- AWS SAM for deployment.

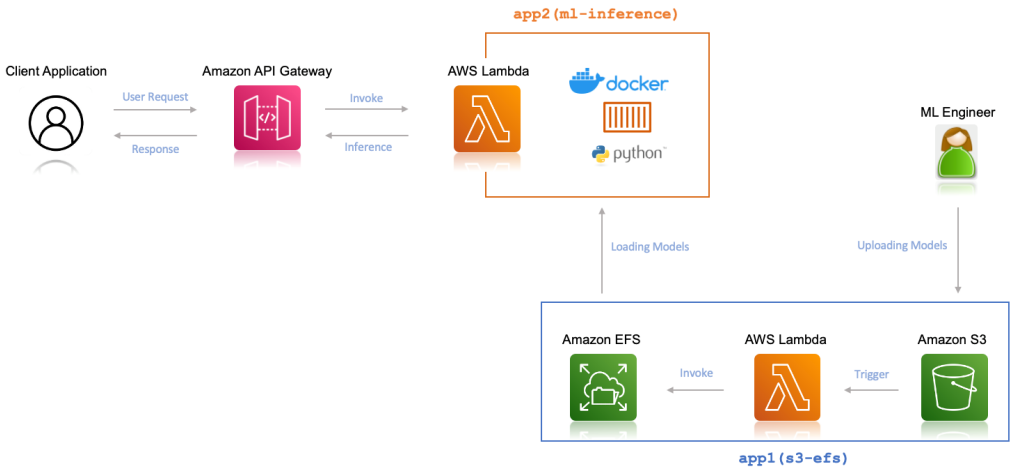

Figure 1. *Data Ingestion and Data Storage Diagram*

2.2.2 Machine Learning Model Training

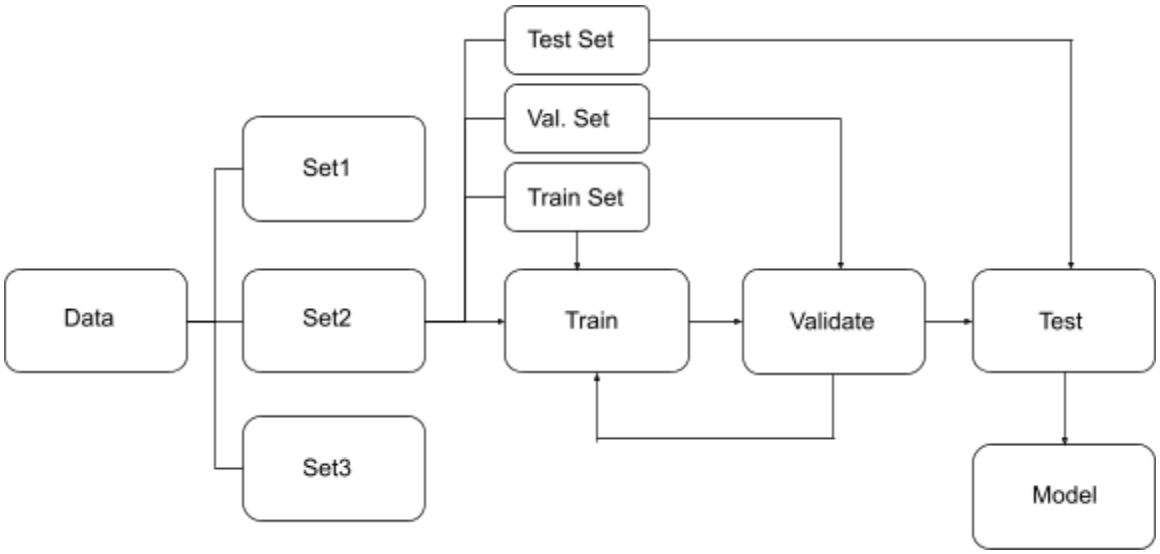

Figure 2. *Machine Learning Pipeline Diagram*

This project will have a main portion that focuses on data analysis and machine learning of energy storage systems. The diagram is then a data flow chart that illustrates the data processing and machine learning pipeline. There are various battery types and different recording methods among the existing data sets, so they cannot be used as a whole data set. Different types will be separated first before any processing happens. For

the same battery type, datasets can be merged; however, the datasets may have different features. As a result, feature selection is required in the preprocessing. To increase training efficiency and performance, statistical techniques can be used to obtain training samples. Overall, three data groups will be produced.

All collected data will each be analyzed on their purposes, features, min-max values, and distributions. The data may be separated and merged into groups for various purposes, mainly to predict different variables with either the same or different input feature combinations. The preprocessed data should also be analyzed to make sure it can statistically be used to represent the whole dataset population.

The training with each data group should be independent, which means three groups will train in parallel. Then within each group, the data will be split into training, validation, and testing sets for the standard purpose of the machine learning pipeline. The training sets will first be used to train multiple models to predict target features. Each model will be validated with the validation sets, and further improvements will be made. This can include techniques such as hyperparameter tuning and data augmentation. The training and validation process may repeat for multiple iterations until the desired model is acquired. Last, the testing dataset is used to evaluate the model performance. The model can only begin the testing process if it passes the validation phase. The final models will be integrated with the web application.

2.2.3 Machine Learning Model Deployment

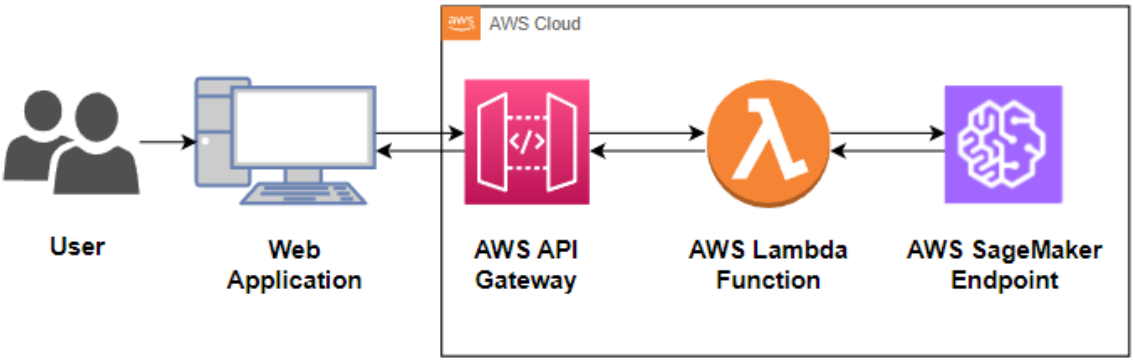

Figure 3. *Deployment Architecture Diagram*

The models can be integrated into the web application using AWS Machine Learning that runs in AWS cloud. The deployment architecture consists of the following components: AWS API Gateway, AWS Lambda Function, and AWS SageMaker Endpoint. Amazon SageMaker is a platform for deploying models to the cloud. Using this platform, developers can build and train their own models while efficiently deploying their models. Figure 2 shows an illustration of the deployment architecture for the models.

The models will be tested and built using a notebook inside AWS. In order to access these models, a connection needs to be established between the web application and AWS. An API gateway provided by AWS will act as a middleware between the client and the server. When the user accesses the web application through a web browser and wants to access the models, the client's application will send a REST-style request and connect to the API gateway. The API gateway will then connect to an AWS lambda function, which parses the information of the client's request and sends the information to the AWS SageMaker endpoint. The SageMaker endpoint, containing the code for our models, are used to process the data and perform a prediction. After a prediction is made, it is returned to the lambda function, which is then sent back to the API Gateway, and back to the web application for the user to view.

Chapter 3. Technology Descriptions

3.1 Web Application

The product of this project is an interactive web application. Nowadays web applications are commonly favored by both developers and customers, because of their convenience, accessibility, and functionality. Since web applications can be run on any browsers, users are able to access them anywhere with an internet connection. This project aims to enable users to access state-of-art machine learning analytics and predictions on battery storage. In this section, the use of technologies of major components is discussed. Specifically, the technologies for frontend, backend, and database will be talked about. Most of these should be quite familiar and commonly used by developers in industrial practices. The purpose and specific usage of each technology will be discussed below.

3.1.1 React

React is a popular Javascript library for building user interfaces. It has satisfactory performance for web page rendering, as it refreshes components as needed instead of the entire page. As the project may frequently update various data and information, React is the perfect choice. React is also very flexible and it has a modular architecture in the development. This makes the application scalable and allows frequent project updates, including adding more features, and testing various data. This could save time and effort in the long run, as the code is easy to read and implement by any developers.

React also has a large ecosystem of libraries, tools, and resources. There are many open source projects that can make the development easier and more efficient. The project also benefited from this. Specifically, Material UI is used for the major layout and dashboard design. This library provided many useful components, including menu bars, data tables, and plots.

3.1.2 Flask

Flask is a Python web framework, mainly used for backend communication in this project. It is used to receive and send RESTful API requests to frontend, and database. As it is a lightweight framework, it is easy to manage features and dependencies according to the needs of the project. Flask serves two purposes in this project. First, it provides the necessary data for the frontend to display. As the frontend may ask for display of various data, the backend server should provide the corresponding data upon requests. Second, the backend server should be able to retrieve from and update data in the database. The users have the privilege to manage their data, including uploading new files, requesting

data analysis reports, and machine learning training and predictions. The backend server should send requests to the database server.

3.1.3 Plotly library

Plotly library is an open source library used to build interactive data visualizations. A broad variety of visualizations (over 40 types of charts), including line plots, bar charts, scatter plots box plots, histograms, and heatmaps, and more, are among the many that can be generated. Plotly's capability to produce interactive visuals that users can explore and edit is one of its primary advantages. In order to display more information from a graph, the developer can, for instance, add hover over text. Sliders and dropdown menus can also be included to let users adjust the data that is displayed. Plotly can be utilized in a variety of contexts including data exploration, scientific visualization, and business analytics. It also interfaces with well-known Python data analysis tools like pandas and NumPy as well as web frameworks like Flask.

3.2 Model Building

3.2.1 Pandas library

The Pandas library is a powerful open source Python library for data analysis and manipulation. Our data is processed as data frames, and then can be fed into further machine learning pipelines, including data cleaning, data preprocessing, and feature engineering.

3.2.2 Numpy library

The NumPy library is also known as Numerical Python. It is a Python library used for numerical computation. It offers a robust array object and a large selection of array-operating mathematical functions. NumPy is widely used in data analysis, scientific computing, and machine learning. NumPy has several key features:

- Sophisticated array object called ndarray (n-dimensional array) that enables for the efficient storing and handling of huge arrays of homogenous data.
- Mathematical operations that may be performed on arrays such as matrix multiplication and eigendecomposition as well as universal functions including trigonometric, logarithmic, and exponential functions.
- Versatile indexing and slicing features for accessing array members depending on a variety of criteria, including index values, Boolean arrays, and slice ranges.
- Seamless integration with other Python libraries such as SciPy, Matplotlib, and Pandas

3.2.3 Scikit-learn library

The scikit-learn library is a Python library for building and evaluating machine learning models. It offers additional tools and methods like classification and regression. Scikit-learn can be used in conjunction with NumPy, SciPy, and matplotlib and is intended to function in tandem with these libraries. It provides a uniform interface for dealing with various machine learning techniques, making switching between models and comparing their performance simple. Scikit-learn has several major features, including:

- Easy-to-use and reliable API's for creating machine learning models.
- Broad range of built-in machine learning algorithms.
- Assistance with both supervised and unsupervised learning.
- Model selection and hyperparameter tweaking tools.
- Metrics for measuring model performance during evaluation.
- Scientific computing integration with other Python libraries.

Scikit-learn is widely used for a variety of tasks such as data analysis, picture recognition, natural language processing, and others.

3.3 Project Deployment

3.3.1 Amazon AWS EC2

Amazon Elastic Compute Cloud (EC2) is a web service provided by Amazon Web Services (AWS) that gives users the ability to deploy and execute applications to the cloud. Here are some salient characteristics that make it a well-liked option for project deployment:

- Scalability: EC2 enables you to quickly adjust computer resources as your project's demands change. As a result, you can quickly handle traffic surges or alter your resources as your project expands.
- Reliability: EC2 provides a highly dependable infrastructure, with numerous availability zones and automated failover features to guarantee that your application remains operational even if hardware or software fails.
- Flexibility: EC2 provides a wide selection of instance types and settings, allowing you to select the resources that best meet the needs of your project. EC2 also supports a wide range of operating systems, software packages, and computer languages.
- Security: In order to protect the security of the project and data, EC2 offers a variety of security measures including firewalls, encryptions, and access restrictions, to safeguard the safety of your project and data.

3.3.2 Amazon AWS S3

Amazon Simple Storage Service (S3) is an Amazon Web Service cloud-based object storage solution. Large volumes of data, including documents, photos, videos, and other kinds of files, may be stored and retrieved using this method. S3 provides a scalable, long-lasting, and secure data storage system, with characteristics such as:

- Durability: S3 is built to last for 99.999999999% of the time, guaranteeing that the data is always available when needed.
- Scalability: S3 is able to store a limitless amount of data and quickly scale up or down to match the needs of the application.
- Performance: S3 provides excellent read and write performance, with low latency and high throughput.
- Integration: S3 interfaces with other AWS services as well as a wide range of third-party tools and apps, making it simple to utilize in a variety of contexts.

3.4 Project Management

3.4.1 Github

Github is a web-based platform that allows software developers to collaborate on projects, manage code, and share software with others using a variety of tools and services. It is built on the Git version control system, which permits developers to track code changes and coordinate on coding projects with others.

Chapter 4. Project Design

The project will be presented in the form of a web application. The users are able to upload their battery data or select our preloaded data, and select an available model to make predictions and in-depth analysis of the battery. The data uploaded should be a csv file, with a specified format. The available model will depend on the wanted prediction(state of health, remaining life cycle, charge capacity, state of charge). Then the model will make a prediction. The data will be analyzed along with the prediction result, and fed back to the users.

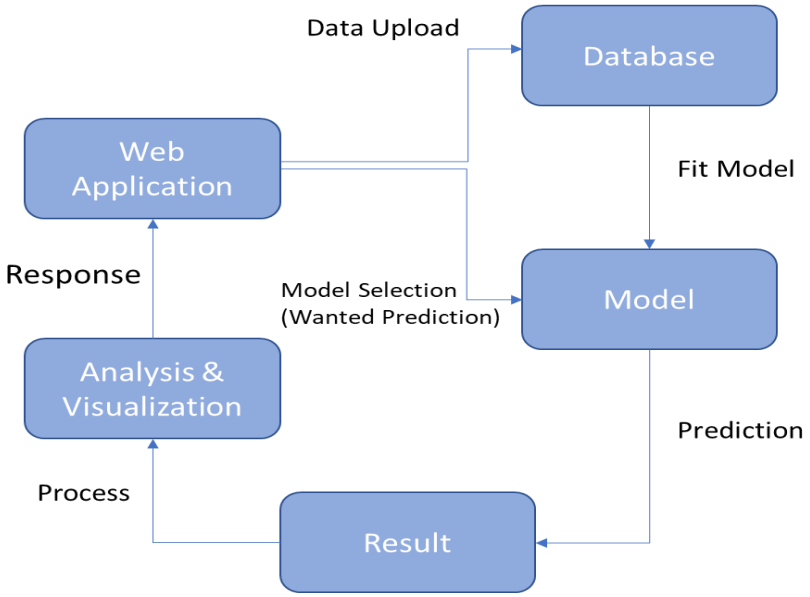

Figure 4. *Project Design Diagram*

4.1 User Interface (UI)

Our initial design was created using Figma. The sidebar shows two options for navigating to the dashboard and prediction page. The sidebar also shows the various datasets the user can click on to retrieve data analysis and predictions. On the dashboard page, three batter quality metrics are shown: state of health, state of charge, and remaining useful life. A data summary displaying statistics of the data are shown to the

user. These values may include temperature, voltage, current, capacity, etc. depending on the dataset. Finally, a graph will be displayed to the user showing data visualizations.

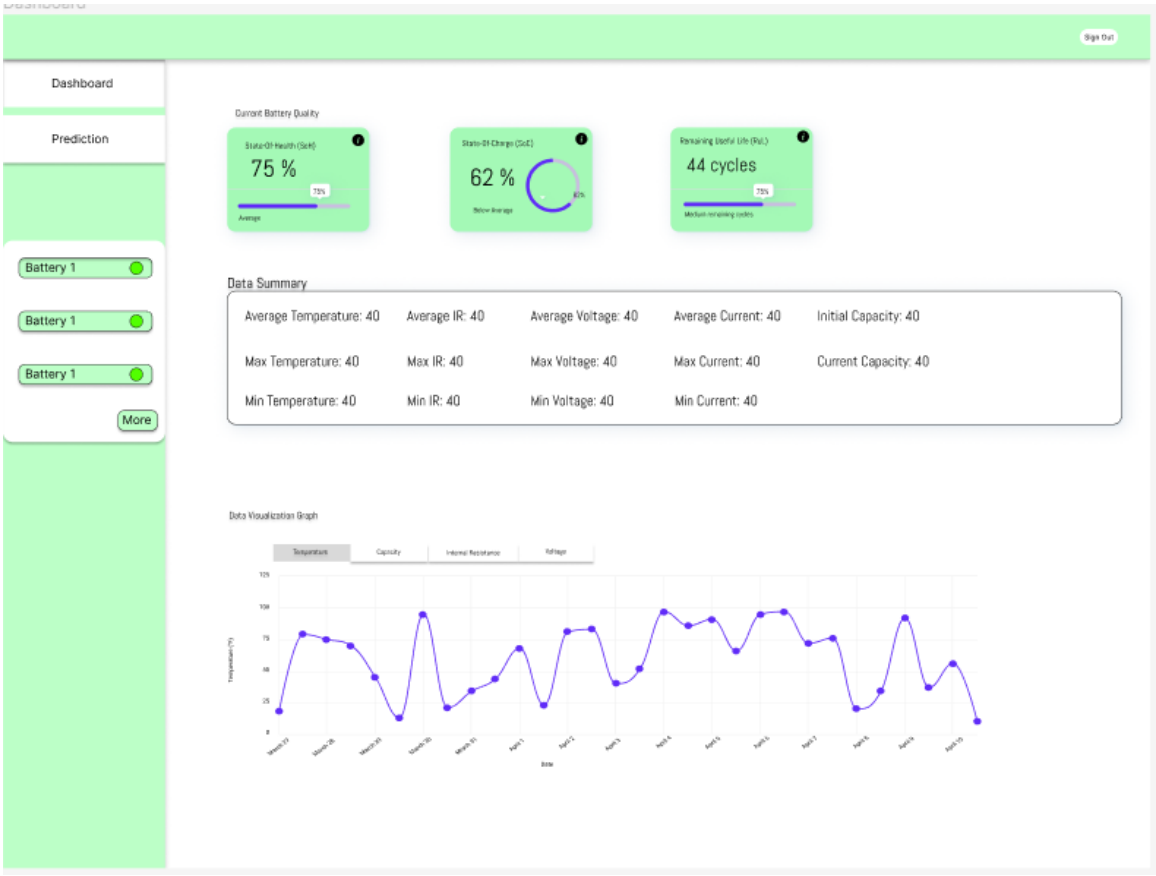

Figure 5. *Battery Data Analysis Dashboard*

4.2 Data Preprocessing

4.2.1 Data Cleaning

Data cleaning is the process of finding and fixing inconsistencies from the data. It is a critical step in machine learning since it guarantees that the data used to train the model is consistent and accurate. Cleansing the data improves the accuracy, reduces bias, enhances feature engineering, and improves model efficiency. When a user uploads their

battery dataset and selects their dataset to be analyzed on the user dashboard, the dataset is checked for missing values, duplicate values, and outliers.

4.2.2 Data Normalization

Data normalization is the process of preparing the data by changing values of the numerical columns to a standard scale without losing or distorting information. It is an important process for obtaining the best model performance.

4.2.3 Feature Engineering

Feature engineering is the transformation of raw data into features that can be useful for training machine learning models. Feature engineering entails choosing and manipulating data variables to create new features from already existing data or features. The accuracy and resilience of the model may increase, due to feature engineering, which leads to a better model performance.

4.3 Machine Learning Models

The Green Energy Storage System Evaluation and Prediction Application requires machine learning to be able to generate accurate performance results for batteries. The diagram below shows the project design for the system. Raw datasets will first be collected and cleaned. We will be using Python to process the data in order to select and generate attributes or features that are highly correlated to battery health. The object of feature selection is to enhance the performance of learning algorithms by getting rid of noise and irrelevant features in the data. Filter, wrapper, and embedded methods are common techniques used for feature selection. Feature processing will be applied to three different datasets containing three battery types. For each dataset, Exploratory Data Analysis will be conducted to better our understanding of the data, investigate patterns, analyze anomalies, and discover characteristics of the data. Tools such as clustering, univariate visualization, k-means, etc. can be useful for this analysis. Based on this information, graphs and plots will be generated to visualize the data. Common types of graphs include linear plots, scatter plots, multivariate charts. Some useful visualizations for battery datasets include plotting battery's capacity against life cycle, temperature against life cycle, or SoH against life cycle. One battery dataset will be fed into each of the three machine learning models: Linear Regression, Gradient Boosting, and Random Forest. The Ensemble Model will take and combine the results of the three models, assign weights to them, and make a final prediction.

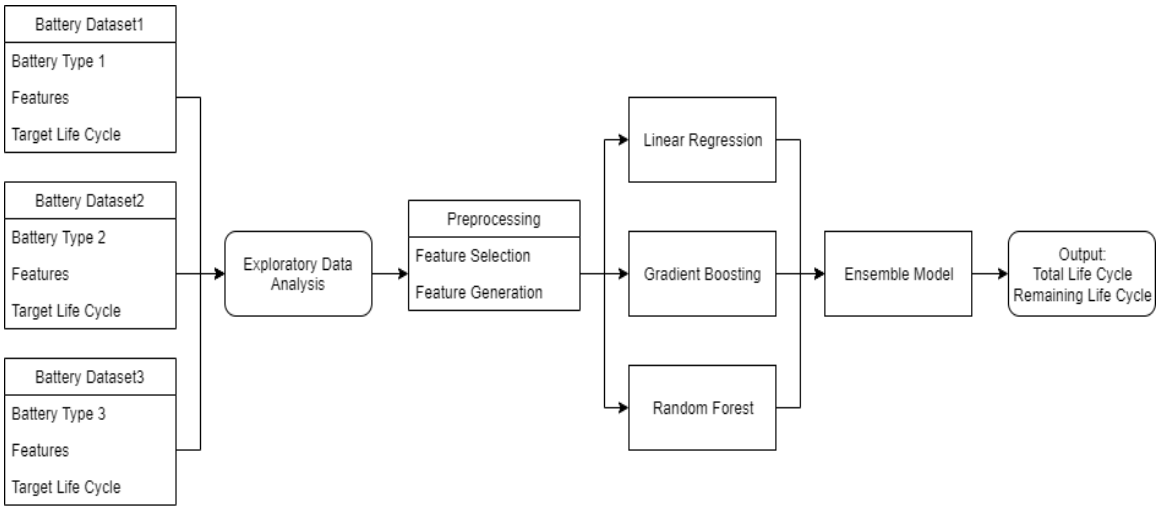

Figure 6. *Machine Learning Pipeline*

To achieve an accurate and timely life cycle predictor datasets are trained with common input features selected from three different groups of data. The battery charge/discharge cycle is linearly correlated with capacity degradation. End of Life (EOL), is when the remaining capacity of the battery reaches 70-80% of the initial capacity.

4.3.1 Classifiers

Classifiers are algorithms that have been trained to identify an input's category. It involves training on labeled input data. After adequate training, unlabeled data can be fed as inputs into the model to retrieve classification labels for each input. The classifier learns to find patterns and correlations in the input by evaluating its characteristics. This allows it to properly categorize subsequent inputs. Common classifiers in machine learning include: random forests, support vector machines (SVMs), Decision trees, Naive Bayes, Logistic regression, and neural networks. Classifiers can be applied in many contexts such as recommendation systems and image identification systems.

4.3.2 Regressors

Regressors are methods used in machine learning to forecast continuous values. It is dependent on a collection of input characteristics. Most common regression models include random forest regression, linear regression, support vector regression, and more.

Chapter 5. Project Implementation

5.1 Client-Tier Implementation

The client is a frontend application that is responsible for displaying the necessary information about the data that the user owns. The app is developed with React. There are a few steps to implementing the entire application. The first step is to install the npm environment, which enables the web application to be run on the browser during the process of development. The second step is to design the global elements, including the top bar and side menu of the application. These will be the main components that help navigate through different pages. Then a router is set up in the main App.js to lay out the overall structure of the website. Each specified path will be directed to the corresponding pages. This makes the project scalable as more pages can be added anytime to expand features. Finally, there are pages and components that need to be implemented. Pages are independent elements that serve various functions, including displaying information, access data summary, and perform machine learning related tasks. Each page can be implemented by different developers, and will not interfere with each other. Components are global elements that can be reused by all the pages. These can include graphical charts, data tables, and status boxes. Components are usually reused many times throughout the development process. Having components can save time and effort for developers.

5.1.1 Dashboard Page

The Dashboard Page has a sidebar for displaying multiple datasets the user has uploaded. The user can upload their battery dataset to the backend. The dataset must be formatted as a CSV file. The user is able to observe several insights about their battery data such as state of health, state of charge, and remaining useful life cycles. In addition to the battery statistics, the user is able to view a prediction of discharge capacity.

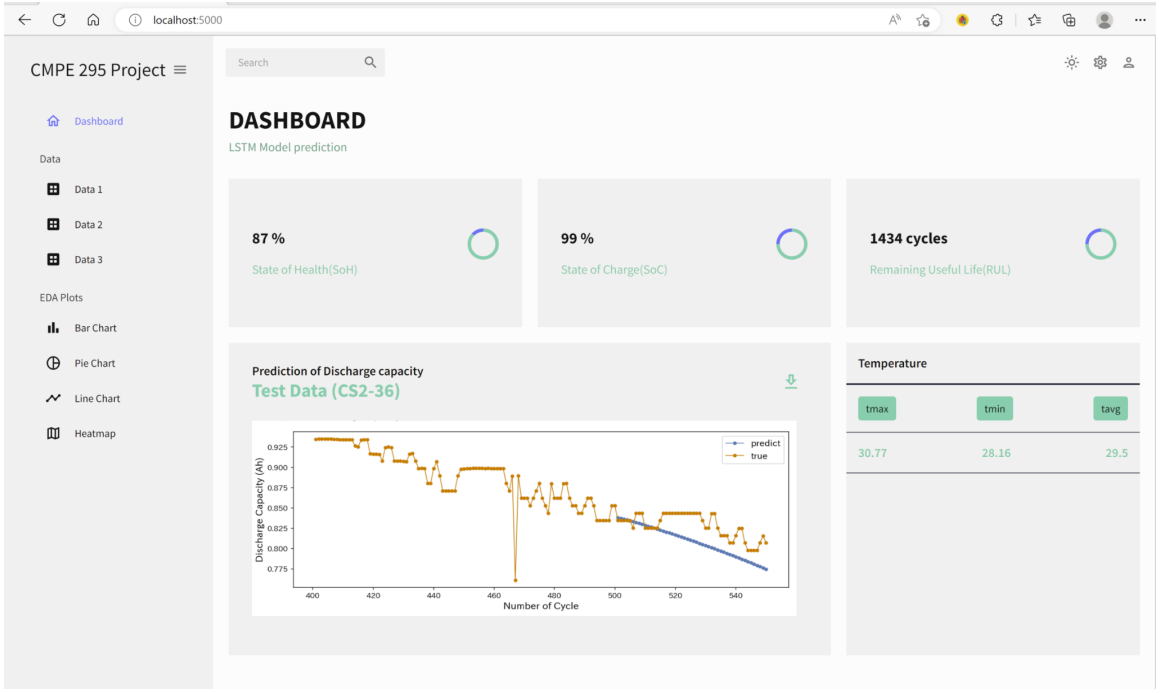

Figure 7. *Battery Dashboard Page*

The user can view and select several datasets on the sidebar. Upon clicking on a dataset, the user is able to view the information in that dataset. The user can cycle through the data using the bottom arrows to cycle through the pages of data.

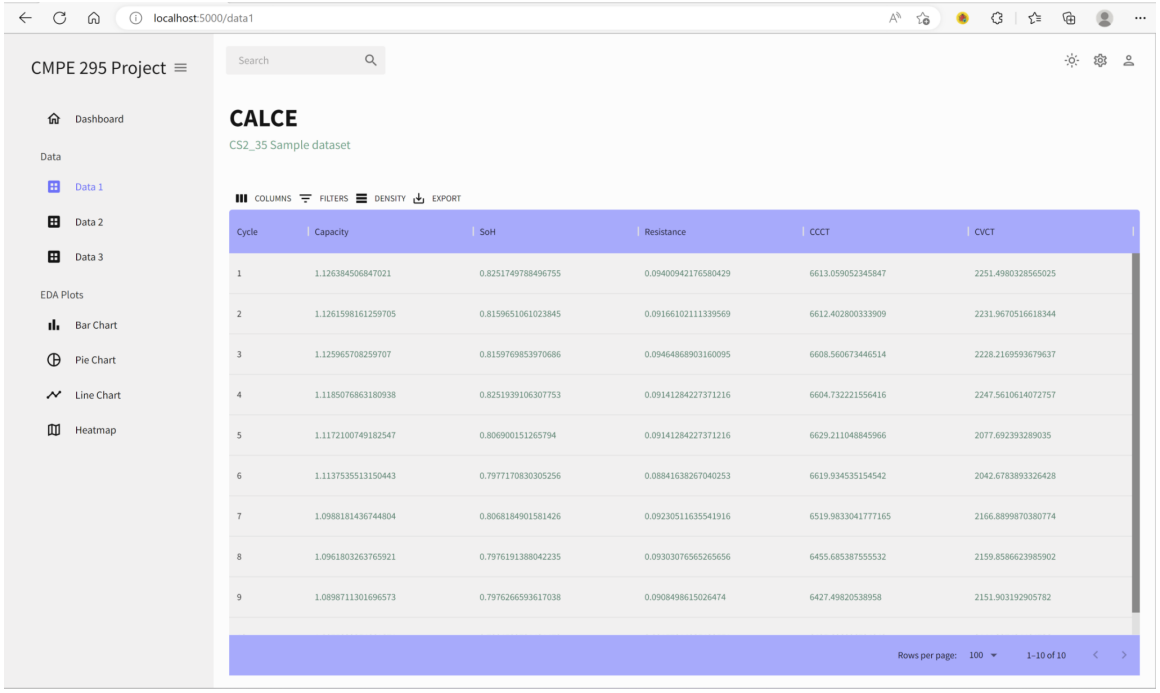

Figure 8. *Battery Dashboard Page after Battery Data Upload*

5.2 Middle-Tier Implementation

5.2.1 AWS Sagemaker

The AWS Sagemaker is used for the machine learning component. The Sagemaker contains our pipeline for preprocessing the data, making data analysis, and providing machine learning predictions. The Sagemaker can take input data directly from the backend server, and respond with a prediction result, along with other metadata.

5.2.2 AWS Lambda

Lambda is a serverless compute service that acts as the backend of the application. The backend code is written in Python, and is used to process the incoming API requests. Specifically, the requests include the GET requests that ask for the data analysis, and prediction results in JSON format.

5.2.3 API Gateway

The AWS API Gateway is a service used in combination with Lambda. This service defines RESTFul API for the application, and forwards the requests to Lambda.

5.3 Data-Tier Implementation

All the data for the application is stored in AWS S3 bucket for fast access. The data are written JSON format and named properly for the corresponding purpose. All the data are secured with the private setting of the S3 bucket. Only the account owner and the application API requests are able to access the data.

Chapter 6. Testing and Verification

6.1 Unit Testing

6.1.1 Data Processing

Verify data is being processed correctly before it is fed into the machine learning model.

6.1.2 Machine Learning Models

Verify that the machine learning models are working as expected by providing test data and checking the model's predictions against expected results.

6.1.3 Dashboard User Interface

Ensure the dashboard user interface is working correctly by checking that the dashboard elements are displayed correctly and user inputs are handled properly.

6.2 Integration Testing

6.2.1 Verify the Integration Between Different Components of the Web Application

This includes testing within Lambda and API Gateway. Simulated testing events are created, such as GET requests for the data analysis and prediction results.

6.2.2 Verify Web Application's Interaction with External Services

Verify the functionality and communication between the application and external services such as S3, Sagemaker, Lambda, and API Gateway to ensure seamless and reliable operations.

Chapter 7. Performance and Benchmarks

7.1 Model Performance Measurement

For regression models, RMSE and MSE are main measurements of the model performance. The error for the train set and test set is computed and monitored. The model should not overfit the data, as the error of the test set should not be too far from that of the train set.

For neural network, we monitor the loss curve of the train and validation set, and set up early stopping when the two converge.

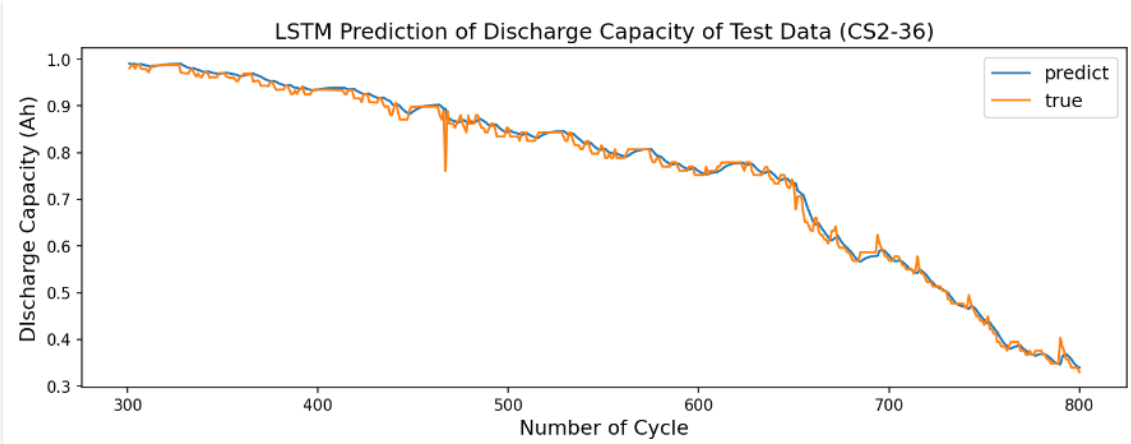

Figure 9. Predicting the discharge capacity of test data using LSTM

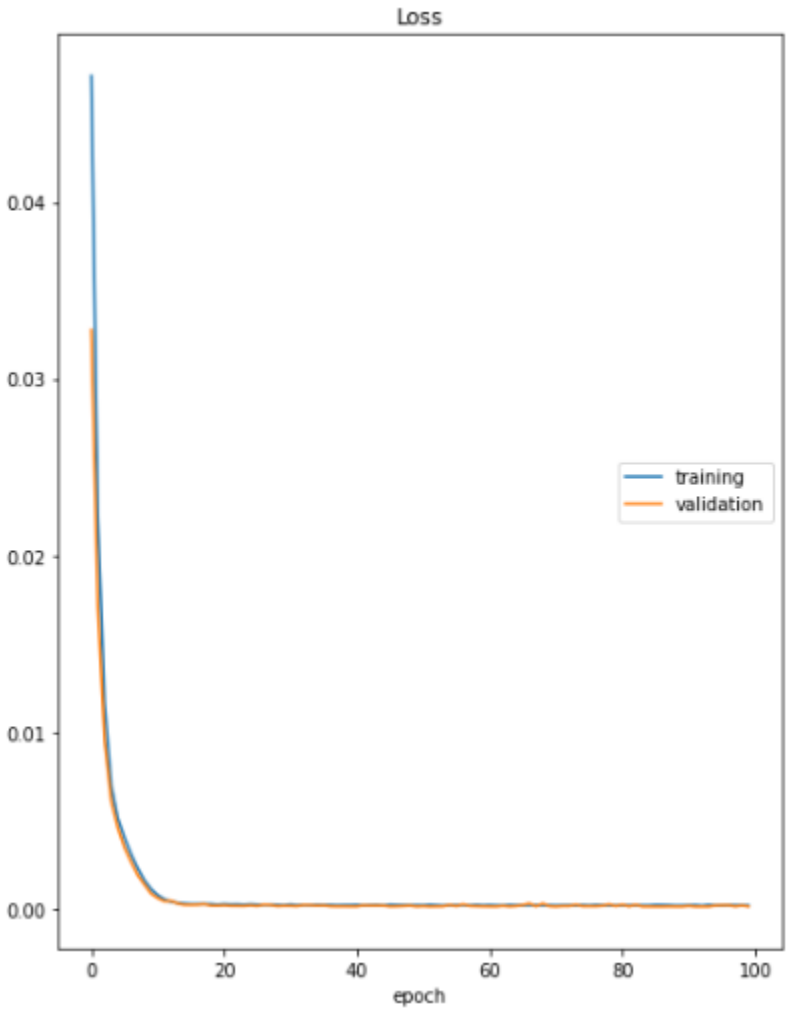

Figure 10. LSTM Model Performance

7.2 Function Benchmarks

Regression, random forest, and neural networks are the three models we analyze and compare for our model performance analysis. With the use of evaluation metric, RMSE, each model is trained and assessed. Results show in both our study and reference [1], the neural network model has the lowest RMSE. Our RF model generated an RMSE of 85, which is lower than RF model in [15]. However, GPR from [15] outperformed the LR model in our study.

Table 3. Model Comparison with Other Studies

Model	RMSE (cycles)	RMSE (cycles) - [Other studies]
Regression	268 (Linear)	119 (GPR) [15]
Random Forest	85	151 [15]
Neural Network	0.00017775749299095287	0.42 [1]

	Model	RMSE (cycles)
Our models	LR RF NN	268 85 0.00017775749299095287
Other studies	GPR RF NN	119 [15] 151 [15] 0.42 [1]

Chapter 8. Deployment, Operations, Maintenance

8.1 Deployment Strategies

The application client can be deployed on AWS EC2 instances. The proper environment needs to be set up for React. This includes the Node.js environment, and the npm install. The machine learning codes can be deployed on Sagemaker, and create an endpoint, which can respond to input data and make predictions. Lambda and API Gateway needs to be set up to provide communication between frontend and backend. An S3 bucket is also necessary for the data storage.

8.2 Operational Needs

If a different data format or prediction is required, the machine learning code needs to be modified to meet the needs.

8.3 Maintenance Required

The model will require maintenance as the data may change or extreme circumstances occur. The model needs to be re-trained if the prediction errors of the new data become too large.

Chapter 9. Summary, Conclusions, and Recommendations

9.1 Summary

The project's major goal is to provide users to obtain comprehensive analytics from battery datasets with a particular focus on accessing battery quality. This is accomplished by using advanced machine learning and deep learning approaches to forecast critical battery metrics such as SoH, SoC, and RuL. Machine learning and deep learning provide strong tools for analyzing large volumes of data and extracting relevant patterns and insights. Our application utilizes linear regression, random forest regression, and neural network models for predicting battery cycle life. In addition, a neural network classifier is used to further assess a battery's condition. These models learn from previous battery performance data, considering features such as internal resistance, qc, qd, current, temperature, and charge time.

9.2 Conclusions

Green energy storage systems are critical for maximizing the use of renewable energy sources. The performance and durability of batteries, which are critical components of these systems, can significantly impact their overall efficiency and lifespan. By accurately predicting battery SoH, SoC, and RuL, potential issues and degradation patterns can be discovered in advance, allowing for preventive maintenance and optimizing battery usage. This application and study intend to improve the efficiency and reliability of green energy storage systems by lowering maintenance costs, optimizing energy use, and improving battery management. Ultimately, the green energy storage performance evaluation and prediction application provides users with valuable insights and predictive capabilities for battery quality analysis.

9.3 Recommendations for Further Research

Further research could be made to compare and explore the use of actual real-world data to lab data for battery datasets. More precise forecasts and diagnostics could be made regarding the performance and behavior of batteries under real-world conditions. The data should encompass various battery chemistries, patterns of usage, and different operating circumstances. The gathering of extensive datasets from various geographical regions can also be facilitated. Real-world data can be integrated with current lab data, resulting in hybrid datasets that hold both real-world operational situations and controlled laboratory conditions.

Another recommendation could be a focus on feature engineering which is the extraction of relevant characteristics from real-world data that accurately reflects important factors that influence battery performance. Each feature can be evaluated by contrasting the performance of models utilizing various sets of features derived from lab and real-world data.

Glossary

State-of-Charge (SoC) - amount of charge or energy available in relation to its capacity

State-of-Health (SoH) - the percentage of the remaining battery capacity

Remaining Useful Life (RUL) - the number of charge/discharge cycles before the performance or SOH degrades to the point where it can no longer operate

References

1. X. Shu, J. Shen, G. Li, Y. Zhang, Z. Chen and Y. Liu, "A Flexible State-of-Health Prediction Scheme for Lithium-Ion Battery Packs With Long Short-Term Memory Network and Transfer Learning," in IEEE Transactions on Transportation Electrification, vol. 7, no. 4, pp. 2238-2248, Dec. 2021, doi: 10.1109/TTE.2021.3074638.

2. A. Varshney, A. Singh, A. A. Pradeep, A. Joseph and G. P, "Monitoring State of Health and State of Charge of Lithium-Ion Batteries Using Machine Learning Techniques," 2021 IEEE 5th International Conference on Condition Assessment Techniques in Electrical Systems (CATCON), 2021, pp. 022-027, doi: 10.1109/CATCON52335.2021.9670522.

3. N. Khan, F. U. M. Ullah, Afnan, A. Ullah, M. Y. Lee and S. W. Baik, "Batteries State of Health Estimation via Efficient Neural Networks With Multiple Channel Charging Profiles," in IEEE Access, vol. 9, pp. 7797-7813, 2021, doi: 10.1109/ACCESS.2020.3047732.

4. S. Bamati and H. Chaoui, "Lithium-Ion Batteries Long Horizon Health Prognostic Using Machine Learning," in IEEE Transactions on Energy Conversion, vol. 37, no. 2, pp. 1176-1186, June 2022, doi: 10.1109/TEC.2021.3111525.

5. Z. Deng, X. Hu, X. Lin, L. Xu, Y. Che and L. Hu, "General Discharge Voltage Information Enabled Health Evaluation for Lithium-Ion Batteries," in IEEE/ASME Transactions on Mechatronics, vol. 26, no. 3, pp. 1295-1306, June 2021, doi: 10.1109/TMECH.2020.3040010.

6. M. Huotari, S. Arora, Avleen Malhi and K. Framling, “Comparing seven methods of state-of-health time series prediction for lithium-ion battery packs of forklifts, ” *Journal of Applied Soft Computing,* 2021, vol. 111, pp. 107670, doi: https://doi.org/10.1016/j.asoc.2021.107670.

7. E. Chemali, P. J. Kollmeyer, M. Preindl, R. Ahmed and A. Emadi, "Long Short-Term Memory Networks for Accurate State-of-Charge Estimation of Li-ion Batteries," in IEEE Transactions on Industrial Electronics, vol. 65, no. 8, pp. 6730-6739, Aug. 2018, doi: 10.1109/TIE.2017.2787586.

8. Z. Ni and Y. Yang, "A Combined Data-Model Method for State-of-Charge Estimation of Lithium-Ion Batteries," in IEEE Transactions on Instrumentation and Measurement, vol. 71, pp. 1-11, 2022, Art no. 2503611, doi: 10.1109/TIM.2021.3137550.

9. B. Zhao et al., "Estimation of the SOC of Energy-Storage Lithium Batteries Based on the Voltage Increment," in IEEE Access, vol. 8, pp. 198706-198713, 2020, doi: 10.1109/ACCESS.2020.3031327.

10. Z. Huang, F. Yang, F. Xu, X. Song and K. -L. Tsui, "Convolutional Gated Recurrent Unit–Recurrent Neural Network for State-of-Charge Estimation of Lithium-Ion Batteries," in IEEE Access, vol. 7, pp. 93139-93149, 2019, doi: 10.1109/ACCESS.2019.2928037.

11. K. Yang, Y. Tang, S. Zhang, and Z. Zhang, “A deep learning approach to state of charge estimation of lithium-ion batteries based on dual-stage attention mechanism, ” *Journal of Energy,* vol. 244, pp. 123233, 2022, doi: https://doi.org/10.1016/j.energy.2022.123233.

12. M. Hosen et al., “Battery cycle life study through relaxation and forecasting the lifetime via machine learning, ” *Journal of Energy Storage*, vol. 40, pp. 102726, 2021, doi: https://doi.org/10.1016/j.est.2021.102726.

13. J. K. Thomas et al., “Battery monitoring system using machine learning, ” *Journal of Energy Storage”*, vol. 40, pp. 102741, 2021, doi: https://doi.org/10.1016/j.est.2021.102741.

14. L. Ren, L. Zhao, S. Hong, S. Zhao, H. Wang and L. Zhang, "Remaining Useful Life Prediction for Lithium-Ion Battery: A Deep Learning Approach," in IEEE Access, vol. 6, pp. 50587-50598, 2018, doi: 10.1109/ACCESS.2018.2858856.

15. Z. Fei et al., “Early prediction of battery lifetime via a machine learning based framework, “ *Journal of Energy Storage*, vol. 40, pp. 120205, 2021, doi: https://doi.org/10.1016/j.est.2021.120205.

16. C. Vidal, P. Malysz, P. Kollmeyer and A. Emadi, "Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation:

State-of-the-Art," in IEEE Access, vol. 8, pp. 52796-52814, 2020, doi: 10.1109/ACCESS.2020.2980961.

17. R. R. Ardeshiri, B. Balagopal, A. Alsabbagh, C. Ma and M. -Y. Chow, "Machine Learning Approaches in Battery Management Systems: State of the Art: Remaining useful life and fault detection," 2020 2nd IEEE International Conference on Industrial Electronics for Sustainable Energy Systems (IESES), 2020, pp. 61-66, doi: 10.1109/IESES45645.2020.9210642.

18. D. Mcintosh, “Li-ion Battery Aging Datasets”, *Dash Link*, Sep. 2010, https://c3.ndc.nasa.gov/dashlink/resources/133

19. F. Zheng, Y. Xing, J. Jiang, B. Sun, J. Kim, M. Pecht, “Influence of different open circuit voltage tests on state of charge online estimation for lithium-ion batteries”, *Applied Energy*, 183 (2016), pp. 513-525, 10.1016/j.apenergy.2016.09.010.
20. Y. Xing, *et al.*, “State of charge estimation of lithium-ion batteries using the open-circuit voltage at various ambient temperatures”, *Applied Energy*, 113 (2014), pp. 106-115.

21. W. He, N. Williard, M. Osterman., M. Pecht, “Prognostics of lithium-ion batteries based on Dempster–Shafer theory and the Bayesian Monte Carlo method”, *J. Power Sources*, 196 (2011), pp. 10314-10321.

22. S. Saxena, C. Hendricks, M. Pecht, “Cycle life testing and modeling of graphite/LiCoO2 cells under different state of charge ranges”, *J. Power Sources*, 327 (2016), pp. 394-400.

23. Severson *et al*. Data-driven prediction of battery cycle life before capacity degradation. *Nature Energy* volume 4, pages 383–391 (2019).

24. C. Birkl, D. Howey, “Oxford Battery Degradation Dataset 1”, *University of Oxford*, 2017, 10.5287/bodleian:KO2kdmYGg.

25. A. Aitio, D. Howey, “Predicting battery end of life from solar off-grid system field data using machine learning”, ArXiv, Jul. 2021, doi: arXiv:2107.13856

Appendices

Appendix A. Description of Implementation Repository

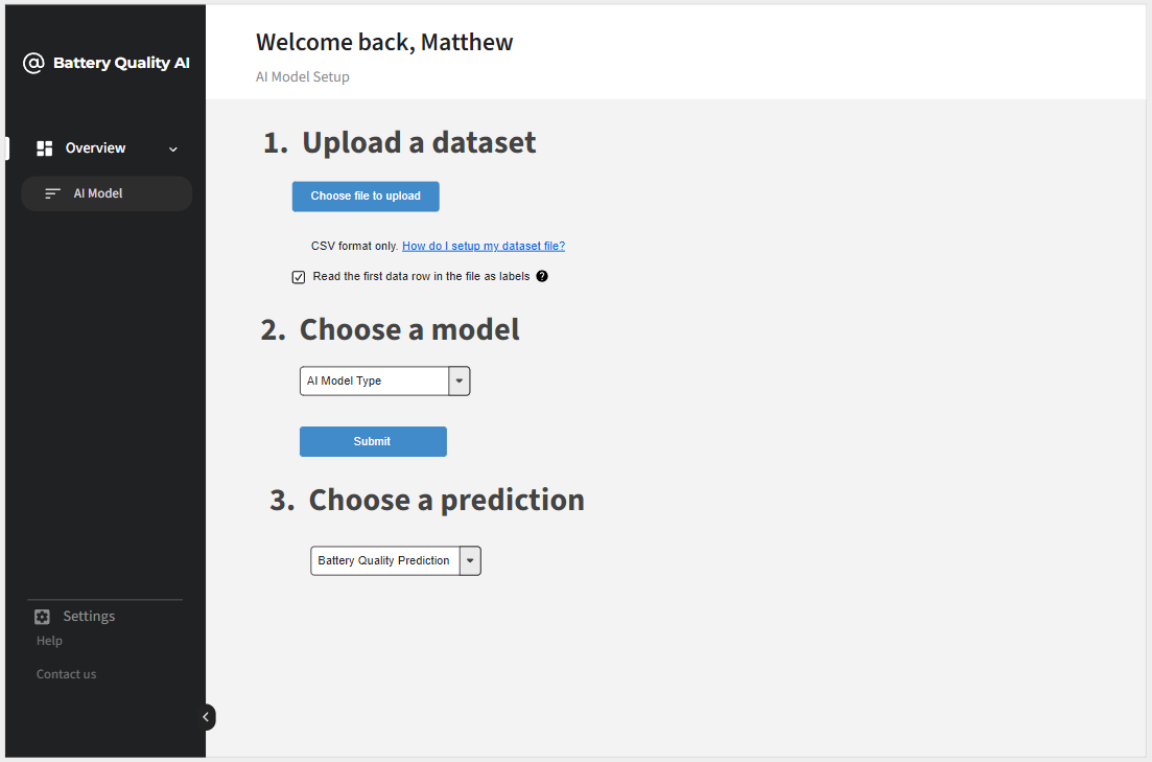

Figure 3. *Web UI design for user uploading their dataset*

Table 1. Review for battery SoH, SoC, lifetime, and RuL prediction

Paper ID	***Purpose***	***Datasets Used***	***Battery Type***	***Models***	***Evaluation Metrics***				
					RMSE	***MAE***	***MAPE***	***R2***	***Accuracy***
[1]	SoH Prediction	NCM data LFO data from CALCE LEP data from MIT-Stanford	Nickel cobalt Manganese Lithium cobalt oxide Lithium iron phosphate	LSTM with Transfer Learning	0.42	0.31	N/A	N/A	N/A

[2]	Monitoring SoH (With Savgol Filter)	LPF/graphite cells data from A123 Systems	Commercial Lithium iron phosphate	Linear Reg. PLS Reg. MLP Random Forest	1.07 1.54 1.96 1.53	N/A	N/A	0.93 0.85 0.68 0.83	N/A
[2]	Monitoring SoC	LPF/graphite cells data from A123 Systems	Commercial Lithium iron phosphate	Random Forest MLP	N/A	N/A	N/A	N/A	0.9759 0.9738
[3]	SoH Estimation	NASA (Battery #5)	Lithium-ion	LSTM BiLSTM	0.0249 0.0099	0.0156 0.0091	0.0359 0.0072	N/A	N/A
[4]	Long Horizon SoH Prediction	NASA and Oxford	Lithium-ion	NARXRNN (Cycle 100)	0.0157	0.0116	N/A	N/A	N/A
[5]	SoH Evaluation	MIT-Stanford CALCE VPSL	Lithium iron phosphate Lithium cobalt oxide Nickel cobalt Manganese	Linear Reg. SVM RVM GPR	0.0179 0.0132 0.0155 0.0127	0.0139 0.0102 0.0119 0.0098	N/A	N/A	N/A
[6]	SoH time series prediction	Dataset of 45 lithium-ion battery packs	Lithium-ion	Gradient Boosting	0.26	0.18	N/A	1.0	N/A
[7]	SoC Estimation	EV dataset with ambient temp increasing from 10 to 25 *C	Lithium-ion battery packs	LSTM-RNN (25 *C)	0.007	0.006	N/A	N/A	N/A
[8]	SoC Estimation	Panasonic NCR18650PF cell data	Nickle-cobalt-aluminum oxide	Physics Constrained NN	0.0036	0.0083	N/A	N/A	N/A
[9]	SoC Estimation based on Voltage	HV Packs Tester by Digatron	Lithium-ion batteries connected in parallel	MEA-BP Algo. Charging Discharging (0.2 *C)	N/A	N/A	0.0051 0.0063	N/A	N/A
[10]	SoC Estimation	Arbin BT2000 Tester – Battery Testing System	BAK 18650 battery	GRU CNN-GRU	0.0177 0.0167	0.0147 0.0133	N/A	N/A	N/A

[11]	SoC Estimation	18650 datasets from UMD	18650 batteries	ED DA-LSTM DA-GRU	1.3913 0.4918 0.3869	1.0671 0.3677 0.3209	N/A	0.9972 0.9996 0.9998	N/A
[12]	Lifetime Forecasting	Aging dataset 40 NMC cells	Nickel Cobalt Manganese	GPR	0.0159	0.0151	N/A	N/A	N/A
[13]	Battery Monitoring System - RUL	Battery Packs connected in series relation	Lithium-ion	LSTM	0.058	N/A	N/A	N/A	N/A
[14]	RUL Prediction	NASA	Lithium-ion	ADNN Bayesian Reg. Linear Reg. SVM	0.0666 0.1192 0.1200 0.1066	N/A	N/A	N/A	0.9334 0.8908 0.8800 0.8934
[15]	Early Lifetime Prediction	MIT dataset (first 100 cycle degradation data)	Lithium-ion	GPR SVM RF	(Cycles) 119 115 152	N/A	0.089 0.080 0.117	0.89 0.90 0.83	N/A

Table 2. Data Summary

Dataset	Purpose	Type	Profiles	Data Structure										Data Collection Interval	Data Source	Data Size	Format
				Cycle	Current	Voltage	Charge Capacity	Discharge Capacity	Temperature	Charge Energy	Discharge Energy	dv/dt	Internal Resistance				
Li-ion Battery Aging Datasets [18]	Battery health	18650	CC-CV EIS	Y	Y	Y	Y	Y	Y	Y	Y	N	N	16-20 seconds	NASA	60 MB	.mat
CALCE Battery Group [19-22]	Battery SoC	INR 18650-20R	DST FUDS US06 BJDST	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	10, 30 seconds	Uni. of Maryland	100-200MB	.xlsx
	SoC accuracy	A123	DST FUDS	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	30 seconds	Uni. of Maryland	100-200MB	.xlsx
	Capacity prediction	CS2	CC-CV	Y	Y	Y	Y	Y	Y	N	N	N	N	10 seconds	Uni. of Maryland	100-200MB	.txt
	Capacity prediction	CX2	CC-CV	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	30 seconds	Uni. of Maryland	100-200MB	.xlsx
	Partial charge	Pouch cells	CC-CV	Y	Y	Y	Y	Y	N	N	N	N	N	10-30 seconds	Uni. of Maryland	100-200MB	.mat
BEEP Structured Data [23]	Battery Cycle Life Prediction	A123	CC-CV	Y	Y	Y	Y	Y	Y	N	N	N	Y	Per charge cycle	Toyota	25 GB	.mat
Oxford Battery Degradation Dataset 1 [24]	Battery Health	Pouch cells	CC-CV	Y	Y	Y	Y	Y	Y	N	N	N	N	1-10 seconds	Oxford	250MB	.mat
Solar Off-grid Field Data[25]	End-of-Life Detection	Lead-acid	N/A	N	Y	Y	N	N	Y	N	N	N	N	Non-uniform	Oxford	3 GB	.npy

SoC: State of Charge.

CC-CV: Constant current and constant voltage profile. The battery is first charged with constant current until reaching a certain state of charge, then switches to constant voltage charging.

EIS: Electrochemical impedance spectroscopy, a widely applied non-destructive method of characterization of Li-ion batteries.

DST: Dynamic Stress Test.

FUDS: Federal Urban Driving Schedule.

US06: US06 Highway Driving Schedule.

BJDST: Beijing Dynamic Stress Test.