Enhancing battery life by performance analysis using ML and DL

Tanishqa Somavanshi[1], Devanshu Dhanorkar[2], Aakash Shinde[3], Pranav Bhagwan Pawar[4]

[1]Student, SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, *tanishqas23@gmail.com*
[2]Student, SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, *dhanorkardevanshu@gmail.com*
[3]Student, SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, *aakashbshinde2003@gmail.com*
[4]Assistant Professor, SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, *pbpawar@pict.edu*

, ,

Abstract:

Lithium-ion battery packs find extensive use in numerous high-power applications, including electric vehicles (EVs) and smart grids, necessitating the implementation of a battery management system (BMS). The efficient management of lithium-ion batteries is crucial for optimizing the performance and lifespan of various applications, including electric vehicles and renewable energy systems. This research focuses on advancing battery state estimation techniques using DNN and LSTM models. The work aims to develop accurate models for estimating the lithium-ion batteries' state of health (SOH) and remaining usable life (RUL). The outcomes demonstrate the excellent forecast accuracy of the suggested strategy.

***Keywords*— Machine learning, Deep learning, battery management systems, state of health, remaining useful life.**

1. INTRODUCTION & LITERATURE SURVEY

In recent decades, the increasing usage of traditional power production sources and automobiles which emit carbon dioxide is one of the causes of the Earth's climatic deterioration. This prompted the development of energy storage technologies, resulting in the development of lithium-ion (Li-ion) batteries, which have become among the best options. This is why the main power source for electric vehicles (EVs), consumer gadgets, and even spaceships is lithium-ion batteries. As a result, lithium-ion battery security and reliability are an important concern when it comes to practical applications. As batteries age, their performance naturally declines, posing risks to the reliable operation of electrical devices and potentially leading to significant consequences.

Li-ion batteries' electrical properties have advanced significantly, however, problems still arise from their non-linear charging capacity and rapid aging rates. The installation of an effective battery management system is essential to resolving these problems. Such a system is critical in safeguarding the battery assembly and optimizing energy utilization by monitoring key parameters such as state-of-health (SOH) and remaining useful life (RUL). By ensuring the proper functioning of the battery, including its SOH and RUL, the performance of Li-ion batteries can be optimized.

Currently, there are two primary categories of methodologies to evaluate State-of-Health (SOH) and Remaining Useful Life (RUL): model-based and data-driven methods. Model-based strategies include models with different modeling mechanisms, such as the electrochemical model and the equivalent circuit model. The electrochemical model clarifies degradation mechanisms. It is based on the physical and chemical reactions that occur within batteries.

Equivalent circuit models (ECMs) offer a less intricate alternative to more complex Electrochemical-Chemical Models (EChMs), providing a balance between simplicity and moderate accuracy suitable for real-time applications. Johnson developed two conventional ECMs: the impedance resistance-capacitance (RC) model and the battery internal resistance equivalent (Rint) model. While ECM implementation was robust, it often overlooked the implicit relationships between internal battery state variables. The simplest Thevenin model featuring a single RC branch with constant model parameters was introduced. However, utilizing ECMs for battery aging estimation necessitates numerous internal battery and resistance aging parameters, posing challenges for parameter identification due to the complexity of lithium-ion battery capacity degradation mechanisms. Furthermore, it can be difficult to identify some of the characteristics linked to side reactions that occur in tandem with the primary reaction, which results in model-based techniques performing less well in real time.

With the burgeoning advancements in machine learning and artificial intelligence, data-driven methodologies have garnered increasing attention. Furthermore, the abundance of lithium-ion battery performance data from real-world applications has facilitated the adoption of data-driven approaches for predicting battery aging. Data-driven approaches, in contrast to model-based techniques, are nonparametric and partially ignore electro-chemical ideologies. Also, various regression and mapping methods are used to construct deprivation models for li batteries.

Data-driven approaches offer a direct means to extract degradation insights and health state evolution patterns from historical data on lithium-ion batteries, bypassing the need for explicit model formulations. These techniques include neural networks and regression methods, such as autoregressive (AR) models, support vector machines (SVM), and artificial neural networks (ANN). For instance, battery Remaining Useful Life (RUL) was predicted online using an AR model enhanced by the particle swarm optimization algorithm. While the AR model boasts simplicity in computation, its predictions lack uncertainty expression. An RUL forecast technique based on the ARIMA model was proposed, albeit requiring time series data stability and stringent battery operating conditions. Another approach involved employing a dynamically driven recurrent network (DDRN) to streamline network complexity and enhance algorithm robustness, albeit with the caveat of potential gradient disappearance issues. Similarly, a regression model leveraging the SVM algorithm was established to estimate battery RUL by training on battery terminal voltage and voltage derivative during charging. However, the SVM algorithm's susceptibility to local minima poses challenges.

Various methods have been explored for predicting the remaining useful life (RUL) of batteries, including time series analysis, ANN, GPR, SVM, and relevance vector machines (RVM), among others. An enhanced autoregressive (AR) model optimized by particle swarm optimization (PSO) for online predictions. While the AR model was straightforward to compute, it lacked uncertainty quantification in the prediction results. An employed a structured neural network approach to streamline network complexity and enhance computational efficiency. However, their method provided only point estimates in prediction results and demonstrated poor performance with limited sample sizes. A multi-kernel SVM (MSVM) utilizing polynomial and radial basis kernel functions to forecast battery RUL. Nonetheless, SVM algorithms are prone to local optima due to inherent characteristics.

Some researchers have employed empirical models to characterize the capacity degradation curve and utilized estimation techniques like the PF, KF, and their enhancements such as the AUKF, UKF, IMMP filter, or models for Approximating SOH and RUL. However, empirical models fail to capture the influence of operating conditions on battery lifespan.

Semi-empirical models, grounded in aging mechanisms, offer a more comprehensive approach by considering the impact of various service conditions on battery aging performance. These models primarily incorporate external factors such as operating environment and battery usage conditions, including temperature, charge and discharge rates, depth of discharge (DOD), and charge and discharge cutoff voltage, among others.

2. PROPOSED METHODS

The open dataset made available by the NASA Ames Prognostics Center of Excellence (PCoE) is where the lithium-ion battery dataset used in this study originated. The experimental data from four lithium-ion batteries (Nos. 5, 6, 7, and 18) that underwent charging and discharging cycles and had their impedance measured at room temperature (24°C) under varied operating conditions make up the dataset. Basically two phases to the charging procedure: first, approximately 1.5 Ampere of current was given until the battery voltage reached 4.2 V; then, a constant voltage charging was done to keep the voltage at 4.2 V while lowering the current to 20 mA. Battery 5, 6, 7, and 18 voltages decreased to 2.7 V, 2.5 V, 2.2 V, and so on. Discharging was carried out at a continuous current of 2 A. The data have been taken from a Matlab (.mat) file containing an open dataset. The procedure retrieves the designated battery dataset and, focusing on discharge events in particular, pulls critical data from its cycles. It then carefully analyzes every discharge cycle, recording important data such as the surrounding temperature, the time and date, capacity, voltage, current readings, and load-related characteristics. The metrics have been methodically organized and saved in a Pandas DataFrame named 'dataset' for additional examination. Furthermore, the function aggregates information specific to battery capacity throughout several cycles into a separate data frame called 'capacity.' Once the data processing is finished, it provides a brief overview by displaying a preview of the 'dataset' DataFrame and printing the total number of data points in the dataset. This simplified method makes it easier to analyze battery data in its early stages, which allows for more effective data loading and preparation for later research endeavors.

Deep Neural Network:

DNN, or Deep Neural Network, is an artificial neural network with more than two layers. A DNN's layers are made up of several neurons, sometimes referred to as nodes or units, which use several mathematical operations to process the incoming data. DNNs are known for their depth, which refers to their several buried layers, each of which is made up of several neurons. DNNs are very useful for jobs because of their hidden layers, which enable them to learn intricate patterns and representations from the input data.

DNN consists of several key components. The following are layers contained in DNN:

Input Layer: the network's first layer, into which input data is fed. In this layer, every neuron corresponds to a feature of the incoming data. Equation (1) shows the input layer.

Hidden Layers: The network uses the intermediate layers between the input and output layers to identify patterns and representations in the input data. These layers could have one or more buried layers with many neurons in each. After receiving input from the preceding layer and assigning a weight to each input, the neurons in each buried layer transmit the outcome through an activation function. Equation (2) shows the hidden layer.

Activation Functions: To add nonlinearity to the model and help the network discover intricate links in the data, each neuron in the hidden layers usually applies a nonlinear activation function to its input. Rectified Linear Unit (ReLU), sigmoid, and tanh are some types of activation functions. They are shown by equations (5),(4) and (3) respectively.

Dropout Layer: A regularization method called dropout keeps neural networks from overfitting. During training, a portion of the neurons are randomly removed (set to zero), forcing the network to acquire more resilient characteristics. Equations (6) and (7) represented the dropout layer during training and interference respectively.

Output Layer: the network's last layer, which generates the output predictions. The type of issue determines how many neurons are in this layer. Regression tasks usually include a single neuron, however, classification tasks may involve several neurons representing various classes. The output layer is shown by equation (8).

$$X = [\, x_1, x_2, x_3, \ldots, x_n \,] \quad (1)$$

$$h_l = \sigma(W_l\, h_{l-1} + b_l\,) \quad (2)$$

$$\sigma(x) = max(0, x) \quad (3)$$

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (4)$$

$$\sigma(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (5)$$

$$\widehat{x_i} = \begin{cases} 0, & with\ probaility\ p \\ \frac{x_i}{1-p}, & otherwise \end{cases} \quad (6)$$

$$\widehat{x_i} = (1-p).x_i \quad (7)$$

$$\hat{y} = W_l\, h_{l-1} + b_l \quad (8)$$

Where X represents the input features. The W and b are the weight matrix and bias vector respectively. The h and σ are the output of the hidden layer and activation function respectively.

Long Short Term Memory:

RNN architectures like LSTM networks were created to address the shortcomings of conventional RNNs in identifying long-term dependencies in sequential data. In contrast to conventional RNNs, which have trouble with vanishing or exploding gradients, long sequence memory cells (LSTM networks) allow them to retain information.

The LSTM cell consists of several key components, each playing a crucial role in the processing of sequential data:
Input Gate (*p*): The information entering the cell state is managed by the input gate. It selects the data that should be kept in the current cell state from the input and the previous cell state. The input gate determines how much each component of the input should be changed by taking as input its features and previously hidden states after that, it outputs a value for each cell state element ranging from 0 to 1. The input gate is represented by equation (9).
Forget Gate (*q*): The forget gate makes decisions about which data from the cell state should be discarded or forgotten. Like an input gate, it accepts input features and previously hidden states as input and outputs, for each element of the cell state, a value between 0 and 1, indicating how much of each element should be remembered or ignored. The forget gate is represented by equation (10).
Cell State (*s*): The LSTM cell's long-term memory is represented by the cell state. With the use of input gate and forget gate procedures, it updates its data storage over time. By adding fresh information (obtained from the input gate) and erasing irrelevant information (ascertained by the forget gate), the cell state is changed. Equation (11) represents the cell state.
Output Gate (*r*): This gate regulates the information transfer from the output to the cell state. It chooses which elements of the cell state should be output as the LSTM cell's final concealed state. The output gate determines how much each component of the cell state should contribute to the output by taking input features and previously hidden states as input and output a value between 0 and 1. Equation (12) shows the output gate.
Long-term dependencies in sequential data can be efficiently captured by LSTM. The network can learn and recall patterns over lengthy sequences thanks to the input gate, forget gate, and output gate, which allow it to selectively update, retain, and output information.

To enable long-term memory, the LSTM cell alternates the RNN cell in the hidden layer with an LSTM cell.

$$p_t = \sigma(\ A_{xp}x_t + A_{hp}h_{t-1} + A_{sp}s_{t-1} + d_p\ ) \quad (9)$$

$$q_t = \sigma(\ A_{xq}x_t + A_{hq}h_{t-1} + A_{sq}s_{t-1} + d_q\ ) \quad (10)$$

$$s_t = q_p s_{t-1} + p_t tanh(\ A_{xp}x_t + A_{hp}h_{t-1} + A_{sp}s_{t-1} + d_p\ ) \quad (11)$$

$$r_t = \sigma(\ A_{xr}x_t + A_{hr}h_{t-1} + A_{sr}s_{t-1} + d_r\ ) \quad (12)$$

where A and d stand for the relevant weight coefficient matrix and offset, respectively; p, q, and r represent the input gate, forgetting gate, and output gate, respectively; s represents the cell state; The hyperbolic tangent activation function (tanh) and the sigmoid function (σ) are respectively.

Optimization Algorithm and Loss Function :

Deep neural networks (DNNs) and other machine learning models are trained using optimization methods as basic building blocks. The goal of these algorithms is to minimize an objective function, often known as the loss function Adaptive moment estimation, often known as the Adam optimizer, is a well-liked optimization technique that is frequently used to train deep learning models, such as neural networks or DNNs. It effectively minimizes the loss function during training by combining the benefits of momentum approaches and adaptive learning rates by calculating the gradients' first and second moments, Adam can maintain adaptive learning rates for every parameter. It computes the first moment and second moment of exponentially decaying average estimates of historical gradients and squared gradients, respectively. Subsequently, the learning rates for every parameter are adaptively scaled using these estimations, guaranteeing quicker convergence and improved management of sparse gradients. Adam also adds momentum by adding a decay parameter to regulate the impact of previous gradients on the current update. Particularly when there are noisy gradients or excessive curvature, this momentum factor aids in accelerating convergence. All things considered, Adam provides strong performance on a variety of deep learning tasks and is renowned for its effectiveness, dependability, and user-friendliness.

$$\theta_{t+1} = \theta_t - \alpha.\frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon} \quad (13)$$

Where $\widehat{m_t}$ and $\widehat{v_t}$ are approximations of the gradient's first and second moments respectively. α is the learning rate and θ represents weights or biases.

A loss function measures the difference between expected results and actual ground truth values in a dataset and is essential to the training process of machine learning models. It is a crucial indicator that provides information about how well the model predicts things throughout the iterative optimization process. The main goal is to reduce this difference to improve the model's forecast accuracy and model parameters. Diverse learning tasks, including regression and classification, are supported by different kinds of loss functions.
In regression settings, for example, Mean Squared Error (MSE) is widely used to penalize substantial differences between predicted and actual values. Mean Absolute Error (MAE), on the other hand, assesses the average absolute difference and gives priority to robustness. In regression tasks, the average magnitude of errors between predicted and actual values is measured using the widely used Mean Absolute Error (MAE) loss function, sometimes called L1 loss.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |\widehat{y_i} - y_i| \quad (14)$$

Where the projected value for the i'th sample is denoted by $\widehat{y_i}$, the actual (ground truth) value for the i'th sample is represented by y_i, and the total number of samples in the dataset is represented by n.

Estimation of SOH :

The model used for the estimation of SOH consists of DNN. This DNN is composed of multiple densely connected layers, each of which adds non-linearity to the model by using the rectified linear unit (ReLU) activation function. The octet of neurons that make up the input layer receives input data whose dimensionality corresponds to the quantity of characteristics present in the training dataset.

Each of the next hidden layers has eight neurons, which makes it easier to extract even more intricate representations from the input data. After the hidden layers, a Dropout layer with a 0.25 dropout rate is added to avoid overfitting. It works by randomly deactivating a portion of the neurons during training. The predictions of the model are finally generated by a single neuron output layer that lacks an activation function.

A stochastic gradient descent (SGD) variant called the Adam optimizer is used to compile the model. It is set up with specific hyperparameters (beta_1=0.9, beta_2=0.999, epsilon=1e-08) and optimized to minimize the mean absolute error (MAE) loss function, which measures the average difference between the values that are predicted and those that are observed.

Estimation of RUL :

For estimation for RUL, the LSTM model is used. To handle sequence prediction problems, the LSTM neural network architecture described in the accompanying code sample makes use of its capacity to identify long-term dependencies in sequential data. The model, which consists of several 200-unit LSTM layers, is designed to efficiently learn from the temporal dynamics found in the input sequences.

By randomly deleting a portion of the input units during training, dropout layers placed at critical spots in the architecture help to prevent overfitting, which in turn promotes generalization and strengthens the model's resilience. To enable the model to transport information across numerous time steps and learn from long-range dependencies in the input data, return sequences are enabled in each LSTM layer configuration.

The implementation of dropout layers after each LSTM layer serves to regularize the learning process of the model and prevent it from being too dependent on particular characteristics or patterns found in the training data. By using this regularization method, the model performs better on sequence prediction tasks that involve noisy or unpredictable input data and can generalize to previously unseen data.

The Adam optimizer, an efficient optimization method renowned for its effectiveness and efficiency in training deep neural networks, is then used to build the model. To reduce the loss function's mean squared error (MSE), which gauges how much the model's predictions deviate from the actual values, the optimizer modifies the model's parameters throughout training.

By ensuring that the LSTM model learns to generate precise predictions while managing the intricacies of the training data, this optimization process eventually makes it capable of doing exceptionally well in sequence prediction tasks.

3. RESULT & DISCUSSION

We conducted an in-depth analysis of various machine learning algorithms for the prediction of remaining useful life and state of charge, and finally after much analyzation came the implementation of a Deep neural network for the SOH part and a Long short-term network for the RUL part.

Figure 1 shows the capacity curve of battery 5 where the threshold capacity value is 1.4. The link between battery cycles on the x-axis and battery capacity on the y-axis is seen in Figure 1. The aging process of batteries as charging cycles go on is depicted in the accompanying graph.

The threshold value is represented by the horizontal line. "Capacity" is the label on the y-axis, and its values span from 1.3 to 1.9. "Cycle" is the name of the x-axis, which runs from 0 to 175. There seems to be a slight decrease in capacity with the number of cycles, as indicated by the markers on the graph.

The picture also shows how a battery's capacity decreases throughout recharge cycles and how much is created for several production cycles.

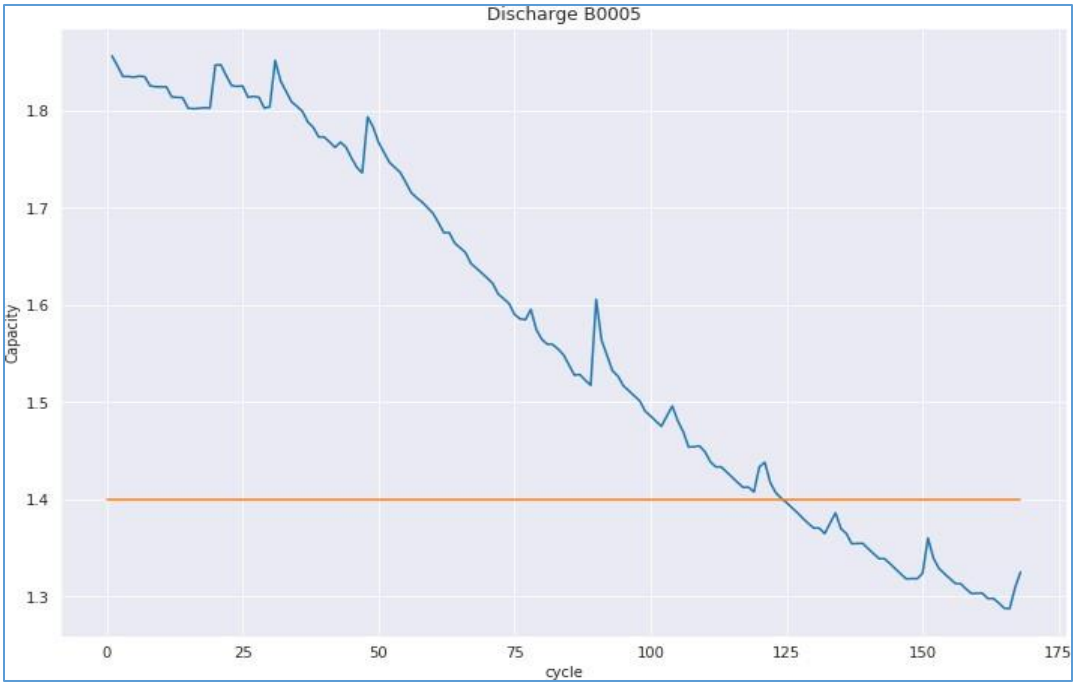

Figure 1: Battery capacity attenuation curve

Table 1 shows the calculated State of Health (SOH) that is calculated on the capacity parameter. The SOH is the ratio of the maximum releasable battery to the initial rated capacity.

$$SOH = \frac{C_{max}}{C_{rated}} \tag{15}$$

Table 1: SOH by capacity

Cycle	Capacity	SOH
0	1.856487	1.000000
1	1.846327	0.994527
2	1.835349	0.988614
3	1.835263	0.988567
4	1.834646	0.988235

Figure 2 shows the SOH curve of battery 5 where a threshold capacity value is 0.70.

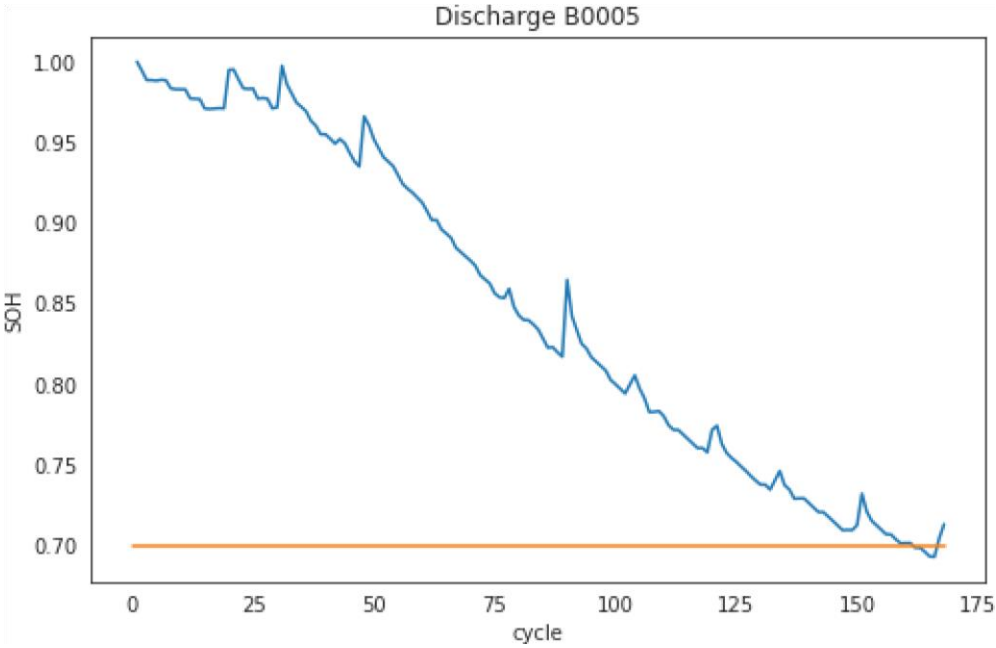

Figure 2: Battery SOH curve

The SOH on the y-axis and cycles on the x-axis are represented in the graph above. It mimics the relationship between the health of the battery and the number of cycles. After closely examining the graph, we deduce that the State of Health gradually declines as the number of cycles increases over time. For every cycle, a graph of the SOH is created like in earlier stages. The 70% mark on the horizontal line indicates that the battery has reached the limit of its useful life and needs to be replaced.

On data obtained from battery 6, using the model we predicted SOH.

Table 2 shows the comparison between SOH calculated from capacity and SOH predicted from the model. From them, the value of the root mean square error (RMSE) is 0.090887. This thus concludes that the model accuracy is precise with high correlation as the RMSE is Low.

Table 2: Comparison between SOHs

cycle	SOH by capacity	SOH by model
1	1.00000	0.959172
2	0.994990	0.956460
3	0.989185	0.953311
4	0.989165	0.953293
5	0.982898	0.949908

Figure 3 shows the visualization of SOH by capacity and by model.

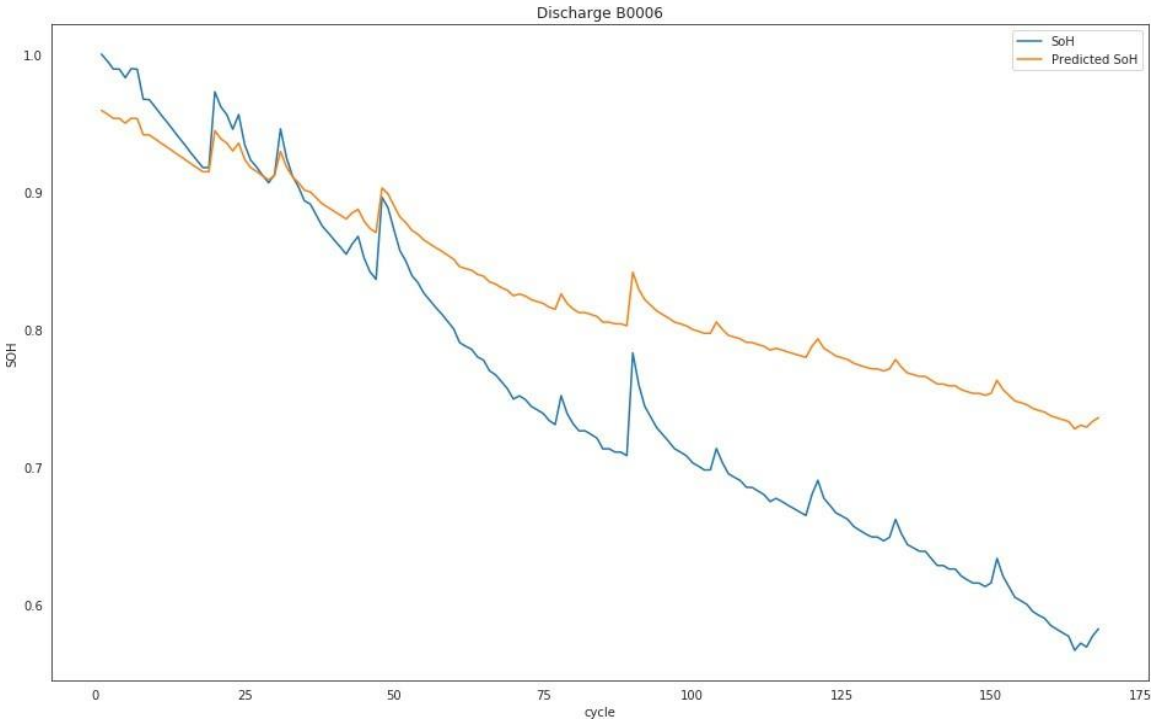

Figure 3: SOH's curve of Battery 6

Figure 3 above Lastly, a plot of the predicted and real SOH is used to illustrate the differences. This graphical representation aids in evaluating the degree to which the model's predictions agree with the actual data. Because the curve forms are almost the same, the model properly learns the data pattern for assessing the State of Health (SOH), as predicted by theory.

The State of Health (SOH) estimate and the training and testing datasets for the RUL estimate are created in the same manner. Battery capacity data is utilized in this situation. Capacity in subsequent cycles is predicted using the first data from the first 50 cycles. This enables you to calculate the battery threshold and the approximate number of cycles the battery will have left before it dies.

On data of Battery 5, The trained LSTM model runs to predict the capacity of the battery. after that comparison of predicted capacity and actual capacity, the RMSE obtained is 0.050. The obtained R^2 score is 0.8655908035539039. R-squared (R^2) score is a common metric used for evaluating linear regression models.
Figure 4 shows the plot of the battery capacity over the discharge cycles for battery 5, with a threshold of 1.4 for capacity.

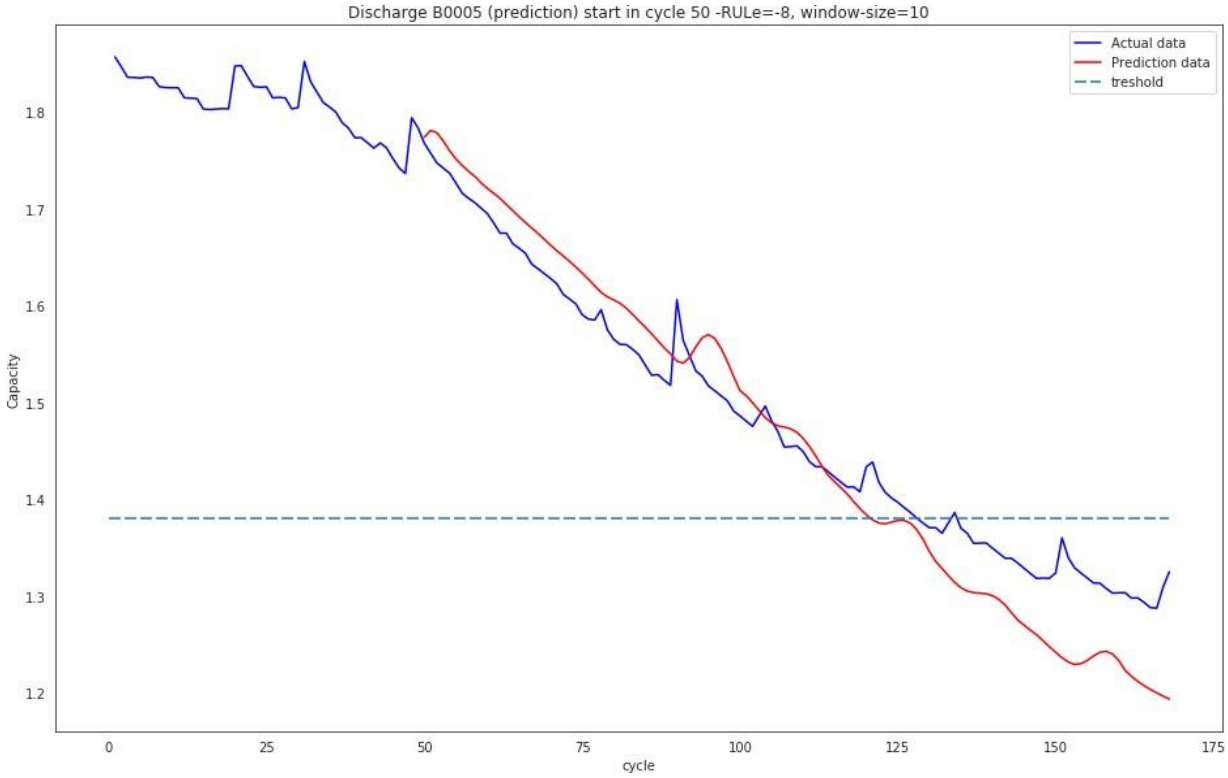

Figure 4: Capacity plot of battery 5

Finally, the graph in Figure 4. above shows that the capacity value and its trend over time roughly match the actual value. Supporting these findings, the RUL estimation error was -8, suggesting that the model anticipated the battery's life cycle end 8 cycles ahead of time.

4. CONCLUSION

In this study, we forecasted the SOH and RUL of EV batteries using deep neural networks and long short-term network techniques. With an EV battery dataset that comprised vital variables like voltage, current, temperature, and charge/discharge cycles, we aimed to build precise prediction models to anticipate battery degradation and performance over time. On the dataset, we used strict preprocessing, feature engineering, and model selection to train both DNN and LSTM models. According to our findings,

LSTM performed better than DNN in terms of RUL and SOH prediction accuracy. By putting these models into practice, EV battery management could be enhanced, lifespan could be increased, and safe and effective operation could be guaranteed. From the model for estimation for SOH and RUL, root mean square errors are 0.09087 and 0.050 respectively.

ACKNOWLEDGMENT

The authors would like to express sincere gratitude to the NASA Ames Laboratory for providing open data.

ABBREVIATIONS

SOH	State of Health of Battery
RUL	Remaining Useful Life
DNN	Deep Neural Network
LSTM	Long Short-Term Memory
ARIMA	Auto-regressive Integrated Moving Average
ANN	Artificial Neural Network
GPR	Gaussian Process Regression
SVM	Support Vector Machine
PF	Particle Filter
KF	Kalman Filter
UKF	Unscented Kalman Filter
AUKF	Adaptive Unscented Kalman Filter
IMMPF	Interactive Multi-Model Particle Filter

REFERENCES

[1] Dunn, B., Kamath, H., & Tarascon, M. (2011). Electrical Energy Storage for the Grid: A Battery of Choices. *Science*. https://doi.org/1212741

[2] Lu, L., Han, X., Li, J., Hua, J., & Ouyang, M. (2013). A review on the key issues for lithium-ion battery management in electric vehicles. *Journal of Power Sources*, *226*, 272-288. https://doi.org/10.1016/j.jpowsour.2012.10.060

[3] Takami, N., Inagaki, H., Tatebayashi, Y., Saruwatari, H., Honda, K., & Egusa, S. (2013). High-power and long-life lithium-ion batteries using lithium titanium oxide anode for automotive and stationary power applications. *Journal of Power Sources*, *244*, 469-475. https://doi.org/10.1016/j.jpowsour.2012.11.055

[4] X. Hu, C. Zou, C. Zhang and Y. Li, "Technological Developments in Batteries: A Survey of Principal Roles, Types, and Management Needs," in *IEEE Power and Energy Magazine*, vol. 15, no. 5, pp. 20-31, Sept.-Oct. 2017, doi: 10.1109/MPE.2017.2708812.

[5] Wang, Q., Mao, B., Stoliarov, S. I., & Sun, J. (2019). A review of lithium ion battery failure mechanisms and fire prevention strategies. *Progress in Energy and Combustion Science*, *73*, 95-131. https://doi.org/10.1016/j.pecs.2019.03.002

[6] Kim, T., Song, W., Son, D. Y., Ono, L. K., & Qi, Y. (2019). Lithium-ion batteries: outlook on present, future, and hybridized technologies. *Journal of materials chemistry A*, *7*(7), 2942-2964

[7] Ren, H., Zhao, Y., Chen, S., & Wang, T. (2019). Design and implementation of a battery management system with active charge balance based on the SOC and SOH online estimation. *Energy*, *166*, 908-917

[8] Tang, X., Wang, Y., Zou, C., Yao, K., Xia, Y., & Gao, F. (2019). A novel framework for Lithium-ion battery modeling considering uncertainties of temperature and aging. *Energy conversion and management*, *180*, 162-170.

[9] Su, C., & Chen, H. J. (2017, November). A review on prognostics approaches for remaining useful life of lithium-ion battery. In *IOP Conference Series: Earth and Environmental Science* (Vol. 93, No. 1, p. 012040). IOP Publishing.

[10] Wang, D.; Miao, Q.; Pecht, M. Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model. *J. Power Sources* 2013, *239*, 253–264

[11] Zhou, Y.; Huang, M. Lithium-ion batteries remaining useful life prediction based on a mixture of empirical mode decomposition and ARIMA model. *Microelectron. Reliab.* 2016, *65*, 265–273.

[12] Li, X.; Shu, X.; Shen, J.; Xiao, R.; Yan, W.; Chen, Z. An On-Board Remaining Useful Life Estimation Algorithm for Lithium-Ion Batteries of Electric Vehicles. *Energies* 2017, *10*, 691.

[13] Williard, N.; He, W.; Osterman, M.; Pecht, M. Comparative Analysis of Features for Determining State of Health in Lithium-Ion Batteries. *Int. J. Progn. Health Manag.* 2013, *4*, 1–7.

[14] Safari, M.; Morcrette, M.; Teyssot, A.; Delacourt, C. Multimodal Physics-Based Aging Model for Life Prediction of Li-Ion Batteries. *J. Electrochem. Soc.* 2009, *156*, A145.

[15] Orchard, M.E.; Tang, L.; Saha, B.; Goebel, K.; Vachtsevanos, G.J. Risk-Sensitive Particle-Filtering-based Prognosis Framework for Estimation of Remaining Useful Life in Energy Storage Devices. *Stud. Inform. Control.* 2010, *19*, 209–218.

[16] Sun, Y.H.; Jou, H.L.; Wu, J.C. Aging Estimation Method for Lead-Acid Battery. *IEEE Trans. Energy Convers.* 2011, *26*, 264–271.

[17] Ng, K.S.; Moo, C.S.; Chen, Y.P.; Hsieh, Y.C. Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries. *Appl. Energy* 2009, *86*, 1506–1511.

[18] Kim, I.S. A technique for estimating the state of health of lithium batteries through a dual-sliding-mode observer. *IEEE Trans. Power Electron.* 2009, *25*, 1013–1022.

[19] Chen, Z.; Mi, C.C.; Fu, Y.H.; Xu, J.; Gong, X.Z. Online battery state of health estimation based on genetic algorithm for electric and hybrid vehicle applications. *J. Power Sources* 2013, *240*, 184–192.

[20] Anton, J.C.A.; Nieto, P.J.G.; Viejo, C.B.; Vilán, J.A.V. Support Vector Machines Used to Estimate the Battery State of Charge. *IEEE Trans. Power Electron.* 2013, *28*, 5919–5926.