Snapshot SoC Identification with Pulse Injection Aided Machine Learning (PIAML)

Youssef A. Fahmy[a], Weizhong Wang[a], Alan C. West[b], Matthias Preindl[a,*]

[a]*Department of Electrical Engineering, Columbia University, New York, USA*
[b]*Department of Chemical Engineering, Columbia University, New York, USA*

Abstract

Machine learning, in the form of a fully connected feedforward network, is applied to predict the state of charge (SoC) of lithium ion batteries injected with a series of pulses applied at different SoCs and states of health (SoHs). A snapshot of the normalized voltage response to these pulses is the only required input. Neither previous data nor Coulomb counting are needed. The Pulse Injection Aided Machine Learning (PIAML) algorithm is able to predict the SoC to better than 1% error on average for fresh, unaged cells and to below 2% error on average for a dataset of both fresh and aged cells. It provides SoC estimates without the need for rest periods, knowledge of capacity, or other equivalent parameters found in other methods. This algorithm can be used as a standalone estimator or as a periodic adjuster to other SoC estimation methods whose results may drift over time. PIAML is validated with constant discharge and drive cycle data.

Keywords: State-of-Charge Estimation, Machine Learning, Neural Networks, Lithium Ion Batteries

1. Introduction

Lithium ion (Li-Ion) batteries are a key technology in rapidly developing fields of interest such as electrified transportation and grid-level storage. These batteries see particularly heavy use in low emission drive trains such as hybrid and battery electric vehicles (EVs), Jackson et al. (2019); Emadi (2014). The general characteristics of these batteries are relatively well known and yet improvements in their power and energy density continue to be made, Bae et al. (2019). All Li-Ion batteries, no matter the application, need accurate and real time knowledge of internal battery parameters.

The state-of-charge (SoC) quantifies the usable charge stored in a battery in per unit, where 0 (or 0%) indicates an empty battery and 1 (or 100%) indicates a full battery. Unlike a voltage or current, the SoC cannot be directly and continuously measured from the terminals of a battery, Wang et al. (2019). This means estimators, predictors, or observers are required to provide an accurate estimate of how long a given battery can provide power, Baronti et al. (2011); Chaoui and Gualous (2017). This paper presents a method of SoC estimation that does not rely on continuous monitoring of a battery, nor does it need any information beyond the voltage response to a specific, short-lived current pulse.

An accurate SoC is a necessary parameter used in active cell balancing. When balanced, cells in a pack age more uniformly, allowing for increased pack capacity and life span, Wang and Preindl (2020); Chaoui and Gualous (2017). Battery management systems (BMS) controlling a pack of cells can use the SoC to prevent degradation and increase performance. In some cases, particularly where battery longevity is a top priority, the BMS will not allow batteries to be charged or discharged beyond a certain range Keil et al. (2016). Furthermore, if exchange of energy is of interest, such as in a hybrid vehicle or a grid storage battery, the SoC will very rarely be at its extremes. Keeping the cells at a mid-level SoC allows power to flow freely into and out of the battery. While over-discharging can lead to degradation, being too conservative with SoC estimation leaves a potentially useful portion of the battery stagnant, reducing its effective energy density.

A simple approach to obtain an SoC estimation involves mapping the open circuit voltage (OCV) characteristic onto SoC, Waag et al. (2014); Chaoui and Gualous (2017). However, the relationship is non-

*Corresponding author.
Email address: matthias.preindl@columbia.edu (Matthias Preindl)

linear. The main drawback of this method is that it requires long rest periods for the battery to equilibrate Baronti et al. (2011). Furthermore, Li-Ion batteries have a relatively flat voltage versus SoC profile in the mid-level region of interest. This means that small voltage measurement errors result in large corresponding SoC errors. Additionally, the OCV-SoC map is sensitive to temperature and cell aging, among other factors, leading to less consistent results when various conditions are applied.

A common approach for SoC estimation is through Coulomb counting, where the remaining charge is obtained by integration of the battery current

$$SoC = SoC_0 + \eta \int \frac{i}{C} dt, \tag{1}$$

where η is the Coulombic efficiency, i is the current in Amperes, and C is the capacity in Coulombs, Charkhgard and Farrokhi (2010). Hence, Coulomb counting requires accurate currents sensors, accurate knowledge of C, and accurate knowledge of the initial state SoC_0. This information must be updated periodically to prevent estimation drift over time. While this is easily achievable in a laboratory it can be challenging in some applications. For example, a capacity measurement requires a full discharge or charge cycle, which is undesirable in applications such as hybrid vehicles or grid-level batteries. Throughout this research, the variance of the capacity is defined as the state of health, $SoH = \frac{C}{C_0}$, where C_0 is the capacity of a fresh cell.

Novel SoC estimation techniques have been developed by many groups, Chemali et al. (2018a,b); Du et al. (2014); Zhao et al. (2020); Charkhgard and Farrokhi (2010); Ceraolo et al. (2020); Shen et al. (2018); Chaoui et al. (2017); Xu et al. (2020). Most approaches use elements from Coulomb counting, terminal voltage measurements to prevent SoC drifts over time, and an explicit or implicit model that links current and voltage. State estimators and Kalman filters use explicit battery models which have been recently improved by Ceraolo et al. (2020) and can be combined with machine learning Xu et al. (2020). Shen et al. (2018); Chaoui et al. (2017) co-estimate SoC and SoH using Kalman filtering and neural networks (NNs). Furthermore, standalone machine learning networks do not require explicit models such as equivalent circuit models, instead they create implicit models and can achieve SoC estimation errors on the order of 1% Chemali et al. (2018b,a).

This paper investigates and demonstrates the capability of a pulse injection aided machine learning (PIAML) algorithm to estimate the SoC directly. The algorithm does not need initial SoC, capacity, or equivalent circuit parameter information. PIAML can estimate the SoC effectively with or without rest periods; it does not require a flat OCV-SoC trajectory and it is effective across a range of SoH. The method is investigated using constant discharge data and validated using driving cycles.

2. Materials and Methods

2.1. Proposed Method

In general, high-accuracy SoC estimation requires a co-estimation of the available capacity, either because the capacity is used directly in the calculation (e.g. Coulomb counting) or because the value is used to account for changing battery properties, Malysz et al. (2016); Chemali (2018). The capacity depends on operating conditions (temperature, discharge rate, etc.) and a range of aging effects. If the capacity decreases then the SoC will become proportionally larger, according to equation 1.

The novelty of the PIAML method, as compared to many of those mentioned above, Chemali et al. (2018a,b); Du et al. (2014); Charkhgard and Farrokhi (2010) is that it is able to accurately predict the SoC at different SoHs. This is achieved by injecting current pulses and measuring the potential response. The pulses are short in duration at 3 minutes long but have been designed, Wang et al. (2019), to maximize information with minimal interruption, see section 2.3 for more information on the battery pulses.

It is hypothesized that the voltage responses during this snapshot encode critical battery information, including the SoC. The voltage responses can be processed then mapped using a neural network to accurate SoC estimates. Section 3.3 also shows that the pulses encode enough information to forgo the need for the long rest periods required in other SoC estimation techniques.

A block diagram of the setup is shown in Fig. 1. The lower portion of the figure shows the physical circuit consisting of a battery, current, voltage, and temperature sensors, and a current source. The middle level shows the control system where data are collected and the pulse generation is controlled. If deployed on a microcontroller, this portion would include a pre-trained neural network to predict the SoC based on previously collected data. The data flows both up to train the neural network and across to the estimation algorithm as the raw data from which the SoC is predicted. Finally, the upper portion shows where the NN is trained. Training data are collected and processed offline before installation, they are then fed to the machine learning algorithm. Once trained, an optimized network can be

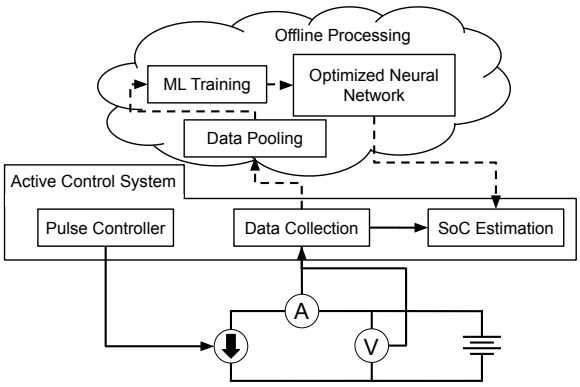


Figure 1: Block diagram of experimental setup, dashed lines indicate the data only flow this way once, during training.

downloaded to a control board and can estimate the SoC without the need for further training.

The results of the neural network are compared to other research and to a simple OCV to SoC map. The OCV was measured after a one hour rest to allow transient effects to settle. The SoC was taken at this point and a curve was interpolated between the selected points. To use the curve for estimation, a measured OCV value is binned and the closest corresponding interpolated SoC is reported. The efficacy of PIAML compared to that of the OCV method, as well as several other research techniques, is discussed in the "Results Comparison" section below.

2.2. Machine Learning using Feed Forward Networks

A fully connected feedforward network (FFN) is used as the machine learning mechanism. FFNs work by taking in data at the first layer then passing the data through a set number of hidden layers before making a prediction at the output. Each layer is composed of a number of nodes that represent values between 0 and 1. For fully connected networks, all nodes from one layer are connected to each node in the next layer. The data enter the network through the input layer which has as many nodes as data points. After this, each of the two hidden layers used can have any pre-set number of nodes, this value was varied to maximize prediction accuracy while minimizing network size. Finally, the output layer consists of a single node representing the SoC.

The neural network training consists of a series of matrix multiplications between the neurons. The activation function takes the weighted connections between each neuron along with an added bias and applies a nonlinear component to ensure it can model the broad range of potential inputs. In the following equations, $w^l_{j,k}$ represents the weight between neuron j in layer l-1 and neuron k in layer l, b^l_k is the bias function, and h^l_k is the activation function of neuron k in layer l. The network produces

$$h^l_k(p) = \max\left(0, \sum_k w^l_{j,k} h^{l-1}_k(p) + b^l_k\right), \tag{2}$$

where

$$h^l_k(p) = SoC(p) \text{ for } l = L, \tag{3}$$

and $SoC(p)$ is the estimated SoC at pulse p. The standard rectified linear unit (ReLU) is used as the nonlinear part of the activation function, it takes the maximum of zero and the input as seen in equation 2.

The root-mean-square error (RMSE) and mean-absolute error (MAE) are used as metrics for the training of the neural network:

$$MAE = \frac{1}{\tau}\sum_{t=0}^{\tau} |e(p)|, \tag{4}$$

$$RMSE = \sqrt{\frac{1}{\tau}\sum_{t=0}^{\tau} e(p)^2} \tag{5}$$

where

$$e(p) = SoC(p) - SoC^*(p), \tag{6}$$

is the error and $SoC^*(p)$ is the ground truth SoC value.

To complete one cycle (or epoch, ϵ) of training, the data are passed through the network in the forward direction, an error is calculated and then, in a process known as back-propagation, the values of the weights and internal nodes of the network are updated in reverse order. These values are adjusted to minimize a predefined error, known as a loss function, that is calculated as a function of the predicted and ground truth values. The loss function for all of the experiments shown here is the mean absolute error.

The training process is continued for a set number of epochs. As the number of epochs increases, the network is able to discern different patterns in the data and encode these patterns in the weights of the connections between nodes. As such, the error between the true and predicted values decreases and the network performs more accurately. It was found heuristically that even the largest networks settled (i.e. their errors did not change significantly with continued training) after 30000 epochs. This number was used across all experiments for training consistency. The following system of equations defines an epoch of training:

$$\begin{aligned} s_\epsilon &= \rho s_{\epsilon-1} + (1-\rho)\nabla^2 \mathcal{L}(w_{\epsilon-1}) \\ w_\epsilon &= w_{\epsilon-1} - \frac{\eta}{s_\epsilon}\nabla \mathcal{L}(w_{\epsilon-1}) \end{aligned} \tag{7}$$

where ρ is the discounting factor defaulted to 0.9, η is the learning rate defaulted to 0.001, s_ϵ is a state maintained by the algorithm that initializes to zero, and $\mathcal{L}$ is the loss function, set as the MAE:

$$\mathcal{L} = MAE. \tag{8}$$

The RMSE, defined in equation 5, is also used for analysis. Variations in the parameters of equation 7, ρ and η, were found not to have significant impacts on results or training time; the defaults are commonly used, Sebastian (2016).

To prevent weighting a range of data more heavily than others, all of the data were normalized before being input into the network. The following mapping function was used to keep all of the data in the closed interval [0,1]:

$$\tilde{x} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

where x is a data point in the original range of data to be mapped, $\tilde{x}$ is the corresponding value in the range [0,1], and x_{min} and x_{max} are the minimum and maximum values in the original range.

Additionally, random noise was introduced to the data to prevent overfitting, an issue that occurs if the algorithm simply memorizes certain values and is unable to accurately predict values that it has not seen before. The magnitude of the noise was chosen to be normally distributed around a mean of zero and with a spread of .01%, (i.e. a normalized value of 0.0001).

To quickly perform many training epochs, TensorFlow, a machine learning framework, is used with a TITAN Xp NVIDIA Graphical Processing Unit (GPU). The TensorFlow framework provides the underlying machine learning code while allowing top level adjustments of different hyper-parameters (those parameters not controlled by the network itself). The GPU can be leveraged by Tensorflow to parallelize operations and decrease the calculation time by an order of magnitude compared to a workstation CPU.

As is typical of machine learning algorithms, once trained, the computer processing required decreases dramatically. The training of a network used in these experiments (using the GPU setup described above) can take between 5 and 15 minutes. The time requirement increases with the sizes of networks and the number of epochs used in training. A single pass through the network, which would be all that is required in a real world application, is a relatively simple operation whose time scale is much less than that of the applied pulse.

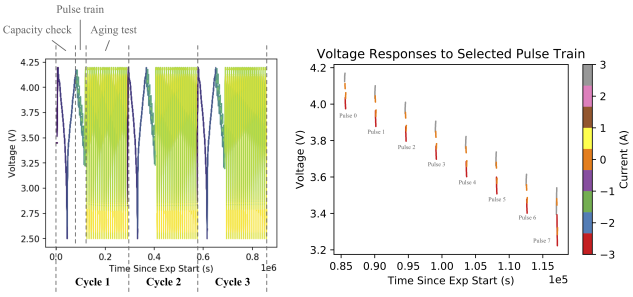
2.3. Battery Pulses

Fig. 2a shows the process that the batteries underwent during three of the twelve total cycles of the experiment. Before each pulse train, where the machine learning data are collected, the battery completes a full discharge of the cell to calculate the capacity as the integral of the current over time. It was then recharged to maximum capacity. Both the charge and the discharge were performed at a low current rate of 0.3 A to ensure minimal effects on the health of the battery. This also reduces the voltage drop across the internal resistance of the battery which prevents a premature voltage protection trigger. Such a trigger would prevent the entire capacity of the battery from being explored. After the pulse train, the capacity of each cell is reduced through rapid charging and discharging. Because of the long aging process, one aging and testing cycle takes around 3.5 days.

The batteries used were lithium ion nickel-manganese-cobalt (NMC). Table 1 lists the cell characteristics. All data were collected in a temperature chamber set to 25 degrees Celsius by a Neware BTS4000 battery cycler connected to a PC. The cycler discharges and charges the battery cells with a programmable current and records voltage, current, and temperature data at a maximum frequency of 0.1 seconds.

As mentioned, the full discharge allows for the calculation of the capacity as the integral of the current over the discharging period. Using this calculation, a ground truth SoC value is calculated using equation 1. This ground truth value is then used as a label for training the machine learning algorithm.

Once the capacity is determined and the battery is recharged, a sequence of constant current pulses are in-



(a) (b)

Figure 2: (a) Voltage over time during three cycles of the experiment. The process was repeated nine more times resulting in 12 total battery ages. (b) One voltage response pulse train showing pulses at different states of charge.

Table 1: Cell Parameters

Cell Chemistry	NMC
Nominal Capacity	3000 mAh
Cut-off Voltage/Current	2.5 V/150 mA
Nominal/Max Voltage	3.6/4.2 V
Max Charge/Discharge Current	4/15 A

jected into the battery; Fig. 3 shows one such pulse. After preliminary analysis, the pulse amplitude was chosen as 1 C rate or 3A. This rate was chosen as a compromise between the signal to noise ratio of the voltage response and the maximum discharge rate of typical lithium ion batteries. The simplicity of the pulse provides clear boundaries between the charge, rest and discharge portions. This geometry was the result of initial research on pulse frequency and amplitude. It is a starting point for future research in which pulses at a wide range of frequencies and amplitudes are investigated. The goal of this research is to develop a pulse that provides all of the same information but is able to be applied while the battery is in use without interrupting normal operation.The idea for these pulses follows from motor resolvers and sensorless motor position estimation.

These pulses perturb the battery in a controlled manner providing the basis for the PIAML algorithm. In this research, the perturbations are supplied by a pulse generator but they could instead be produced by any power electronic system, such as motor drives or battery chargers. Once the perturbing pulse is complete, a discharge is applied to lower the SoC to the next test value.

In order to reduce the effect of noise on estimation accuracy, a single SoC value is predicted for each discharge pulse. The machine learning label is calculated by averaging the SoC across each pulse. This is justified as the pulse occurs over a relatively short time (3 minutes) and the variation in the SoC over this time is small.

Figs. 4a and 4b show voltage response curves from the pulse train in Fig. 2a superimposed for comparison purposes. Fig. 4a shows voltages from different cells at a similar SoH. Data from the three tested cells are superimposed to show the similarity of the responses across the cells.

Fig. 4b shows the voltage responses from a single discharge level but across the 12 SoHs that were achieved. Here the aging effects on the batteries can be seen more clearly. In each of the pulses shown, the cells were discharged the same amount from maximum however the voltage decreased much further for older cells. For example, in the first pulse train (SoH = 1) in dark purple, the minimum normalized voltage reached after 0.6 Ah have been depleted is approximately 0.92. By the 12th SoH, shown in bright yellow in Fig. 4b, the normalized voltage was decreased to 0.86 after the same charge was removed. This difference demonstrates the critical effect aging has on the ability of any technique to estimate SoC from measured voltage.

A second set of experiments was also performed in which the constant discharge to reduce the battery SoC is replaced with one of three common drive cycles. The cycles used were the Urban Dynamometer Driving Schedule (UDDS), the US06 Supplemental Federal Test Procedure (SFTP) - Highway, and the US06 SFTP-City. For comparison, Fig. 5a shows the constant discharge case while Fig. 5b shows the drive cycle case. In both cases, the amount of net charge removed from the battery is the same at 0.3 Ah or roughly one tenth of the nameplate capacity of the cells.

For these experiments the pulses were applied both after a one hour rest, like in the constant discharge experiments, as well as before the rest, immediately af-

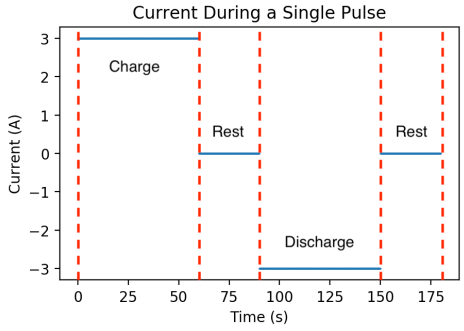
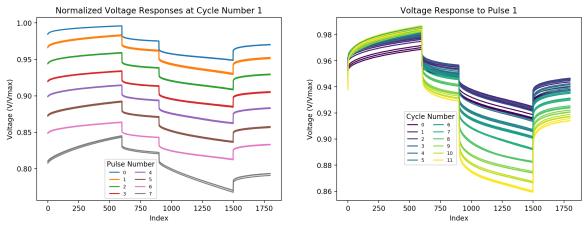
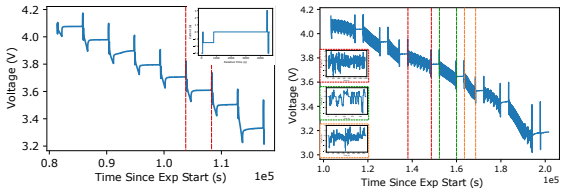


Figure 3: The current during a single pulse. All current pulses are identical, regardless of experiment type.



(a) (b)

Figure 4: (a) shows the voltage responses from a single SoH superimposed, the SoC decreases as the normalized voltage decreases. (b) shows the voltages collated to a single discharge level and that the pulse train number (a proxy for SoH) tends to decrease with voltage.



(a) (b)

Figure 5: (a) shows a pulse train with a constant discharge of 0.3Ah to lower the SoC each cycle with the inset showing a single current pulse. (b) shows drive cycles being used to lower the SoC, the red highlight contains the UDDS pulses, green shows US06City, and orange shows US06Highway. The insets show a single current pulse which are repeated until 0.3Ah are drained from the battery.

ter the drive cycle. Since the effect of a single pulse is thought to be negligible on the long term characteristics of the batteries, adding an extra pulse before the rest allows for observation of the short term effects of the rest. In other words, with the additional pulse the effects of the rest period on SoC prediction can be determined. Other methods, such as OCV mapping and more direct ML modeling, require this rest period for equilibration. Section 3.3 shows that the rest period can be removed, demonstrating a significant reduction in the time needed to predict the SoC, without compromising on the accuracy of the estimations.

3. Results and Discussion

3.1. Training and Tuning Models

Fig. 6 shows the error as a single model is trained. As expected, the error tends to decrease as the number of epochs increases. While a model is learning, the training data are further split into two subsets: regular training data and validation data. Validation data are used by the model to check as it is training. Importantly, the final testing data, which were partitioned from the training data at the start, do not provide any feedback for the model and are only fed to the model after the training is complete. Of the three groups, testing data are the most indicative of real world results as they are only fed through the model at the end of the training process.

Many hyperparameters were tuned to predict the SoC most accurately; among the most important is the number of nodes in each layer of the network. The number of nodes is directly correlated to the complexity of the network, which affects both its ability to predict accurately and the time it takes to train the network. During the course of the experiments, the time to train a network using a GPU was found to be a sublinear function of the number of network nodes. This meant each additional node did not substantially increase training time. GPU use decreased the time to train a network by a factor of 60 when compared to CPU use exclusively.

Fig. 7 shows this process as the number of nodes is swept through. At first, small numbers of nodes were used for rapid prototyping of the models but it was quickly discovered that a higher number of nodes into the hundreds produced better results. Because of the parallel computation abilities of the GPU in use, a model with a high number of nodes did not take an exceedingly long time to train. The model and corresponding parameters with the lowest RMSE were selected automatically by the code. Since prediction can be done with any one of these models, it is only this best model that was further used and examined.

In a more typical experiment, a shorter range of nodes

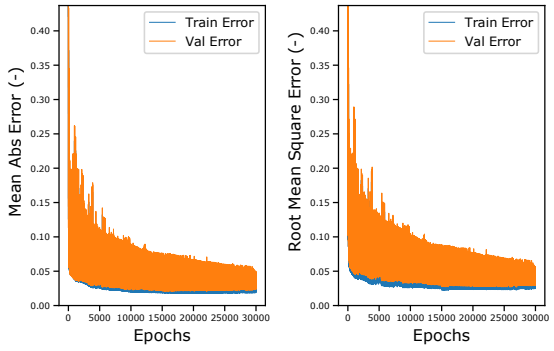


Figure 6: Example of a model training over the course of 30000 epochs. Since the training data is repeatedly fed through the network it has a slightly lower magnitude.

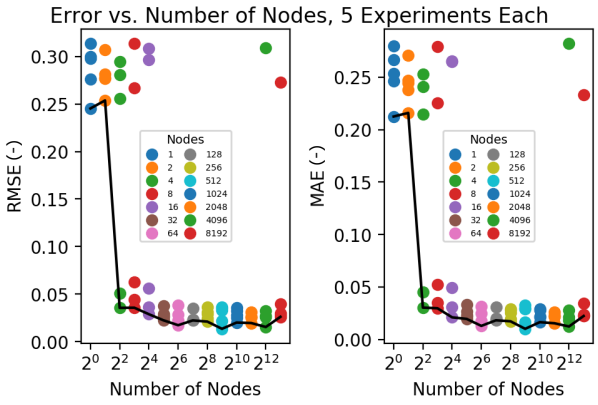


Figure 7: Each dot represents a model that was created and trained. At the end of each experiment, when all networks are trained, a single model is selected based on the lowest RMSE.

would be swept over to save run time. In this case with a wide range of nodes, Fig. 7 shows that while a network with 512 nodes performed the best, one with 64 nodes had similar results. Small changes in randomized initial values of the network can lead to different results even when no explicit parameters are changed. For this reason, each experiment is repeated five times.

A network appears to need at least four nodes to work with any reliability. Thousands of nodes appear unnecessary as diminishing returns set in and can even cause high errors comparable to models with two to four nodes. Over-fitting is one possible reason for this high error as a network attempts to memorize particular data rather than encode trends.

3.2. Selected Results: PIAML Operation and Aging

The errors of selected experiments compared to the Coulomb counting calculated values are summarized in Table 2 and three example plots are shown in Fig. 8. The experiments show that the PIAML algorithm is able to accurately predict the SoC of the cells at various SoHs, under constant discharge and drive cycle conditions, and is able to so without the need for a time consuming rest period. The models perform well across SoC values but have a slight preference for mid-to-low SoCs.

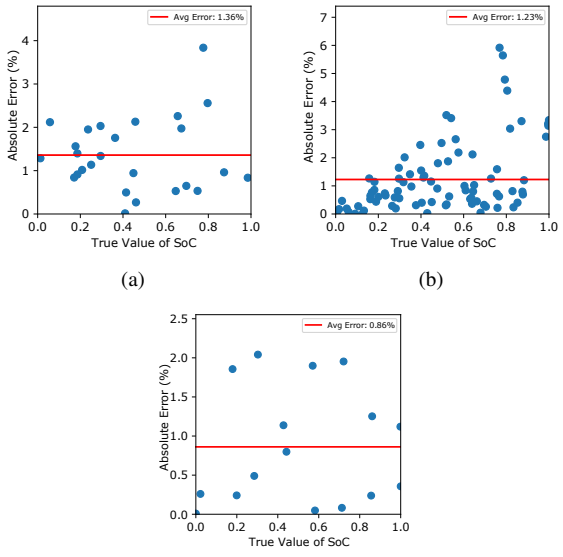


Figure 8: (a) shows the resulting error when all of the constant discharge data from 12 different SoHs are fed to the network. (b) shows results when two cells are used to train and a third is used to test the network. (c) is the error comparing rested to un-rested drive cycle data.

Fig. 8a shows the error of the network as a function of the true SoC. This network was trained with constant discharge pulses across the 12 SoHs investigated. Despite the differing ages, which can be seen in Fig. 4b to lead to differently shaped voltage response curves, the network is able to maintain an error under 2%.

3.3. Selected Results: Drive Cycle Operation and Rest Period Independence

A second experiment involves a model that is trained on drive cycle data in which the pulses were applied before the rest then tested on those after the rest. This experiment was performed at a single SoH. The drive cycles themselves were not observed to cause a significant increase in error as compared to the constant discharge case. The model predicted SoC values with well below 1% error regardless of whether constant discharge or drive cycle data were used. Furthermore, removing the rest period for a single SoC did not increase the error above 1% on average.

The error plot shown in Fig. 8b shows a model trained with two of the cells and tested on the third. This is included to emphasize the ability of the NN to predict across cells of the same type but which may have manufacturing differences. In the real world, a model will not be able to be trained on every individual cell. This experiment demonstrates the neural network is able to use information from a group of representative cells and apply it to the entire batch.

The error plot shown in Fig. 8c shows an average of 0.8% error for the pre-rest to post-rest test. The low error indicates the model does not require the rest period that is present in the other experiments. The elimination of a one hour equilibration period to accurate SoC estimation would greatly increase applicability to typical driving situations.

3.4. Results Comparison

Table 2 summarizes the most significant experiments performed. The column labeled “Split” refers to the di-

Table 2: SoC prediction error of PIAML with constant discharge data at SoH=0.7-1. Split is the percent of data used for training/testing, Max is the maximum percent error that of a predicted value.)

Type	Split	Nodes	RMSE	MAE	Max
SoH = 1	90/10	512	1.8%	0.15%	0.24%
SoH = 1	80/20	512	2.4%	0.61%	0.99%
All SoH	90/10	256	2.8%	1.36%	3.84%
All SoH	80/20	32	2.8%	1.42%	3.42%
Ind. Bat.	66/33	256	2.4%	1.47%	4.53%

vision of input data between training and testing. Two splits were performed in the constant discharge experiments: the first where 90% of the data is used for training and the second where only 80% is used for training. In Table 2, "%RMSE" is the root mean square percent error, "%Err" is the average percent error, and "%Max" is the maximum percent error.

The first four rows of Table 2 show that the PIAML algorithm is able to accurately estimate SoC at all SoHs tested. When training and testing using all of the states of health achieved, there is higher error than when trained with the fresh SoH data. This indicates that the changes seen in Fig. 4b in the structure of the SoC data with changing SoH effect prediction accuracy. However, this difference is not enough to disqualify the use of PIAML models. The training and testing node splits do not affect the error significantly, indicating the model is not simply memorizing large portions of data.

The row labeled "Ind. Bat." in Table 2 shows results from another experiment where two of the three constant discharge cells were used as training data and the third was used for testing. Error results are shown in Fig. 8b.

More information on the drive cycle experiments are summarized in Table 3. The drive cycle discharges give a more realistic look into how this algorithm might behave with data from the real world. Training and testing were done with various sets of data. When the model uses all of the collected data at varying SoHs and the drive cycle data for training it performs approximately the same as without the drive cycle data. In other words, no large errors are introduced when adding the drive cycle data.

Since the drive cycle experiments were only performed at full SoH, they were compared with the corresponding constant discharge, full SoH data. A model trained with the drive cycle data alone performs comparably to the single SoH case of the constant discharge data presented in table 2. There is a slight increase in error when separating the training and testing data by type but the lack of a major increase suggests the data are still related.

Table 3: SoC prediction error of PIAML in drive cycles with or without a rest period, where DC is drive cycle data, CD is constant discharge data, and Mix has both DC and CD.

Training Data	Testing Data	RMSE	MAE
Pre-Rest DC	CD	3.26%	1.64%
Post-Rest DC	CD	2.82%	2.38%
Mix	Mix	2.73%	1.14%
All DC	All DC	1.99%	1.08%
All DC	CD	3.62%	2.18%
CD	All DC	1.32%	3.60%
DC Pre-Rest	DC Post-Rest	2.3%	0.86%

The row labeled "Pre/Post" gives results of a model trained with pre-rest data and tested with post-rest data when drive cycles are applied for discharging. This result shows that the network does not require the long rest period found in other estimation techniques.

As a control, a simple OCV-SoC curve was used for prediction. The resulting error was 2-3x higher than that of the machine learning model as seen in the penultimate row of Table 4. OCV points were extracted after an hour long rest. This rest occurred between each discharge pulse which allowed the battery to dissipate transient effects. The same data is used in creating the OCV curve as was fed to the NN, making it comparable to the PIAML algorithm.

As shown in Table 4, the error from the method presented here is comparable to that in other research. Only Chaoui et al. (2017) has a lower error. All of these methods, which vary from Kalman filters to NNs, required past data or a continuous measurement from an initial state to estimate the SoC at a given point. Temperature and aging effects, which contribute to problem complexity, are considered differently across publications.

This research stands out as needing only the snapshot of data present during the pulse. The pulse can be applied at an arbitrary time, without the need for a rest, and still provide accurate SoC estimation. This provides flexibility for a BMS and allows for the method to be used to periodically correct other methods which may drift over time. If interrupted, methods without this flexibility will at best introduce an additional error or at worst be completely unable to make SoC estimations. PIAML performs comparably or better and does not suffer from such limitations.

At present, these results show promise however, this research focuses on pulse application in controlled conditions. PIAML cannot be used while driving because of the amplitudes and time scales currently involved. Future research intends to reduce both the size and duration of the pulses, while maintaining the low error that would be necessary for continuous estimation during general driving. Furthermore, research indicates that it is possible to co-estimate additional parameters such as the state of health.

4. Conclusions

Using the PIAML algorithm, the state of charge of fresh Li-ion cells can be accurately predicted to below

Table 4: Comparisons across literature. Various methods, conditions, and error types are presented. Unlike other common methods, PIAML does not require continuous monitoring of the battery to produce accurate results.

Source	Error Type	Error	Cells	Temperature	Aging	Type
Chen et al. (2019)	RMSE	2-4%	LiNMC	10-40^{o}C	No	Continuous
Bo et al. (2008)	Max	5%	Ni–MH	27^{o}C	No	Continuous
Chaoui et al. (2017)	RMSE	0.3%	LiFePO4	10-40^{o}C	Yes	Continuous
Zhou et al. (2019)	MAE/RMSE	3%	LiNCM/LiFePO4	25^{o}C	Yes	Continuous
Bian et al. (2020)	MAE	1-5%	LiNCA	-20-25^{o}C	No	Continuous
OCV-SoC Estimation	MAE	2-3%	LiNMC	25^{o}C	Yes	Snapshot
PIAML	**RMSE**	**1-2%**	**LiNMC**	**25^{o}C**	**Yes**	**Snapshot**

1% absolute error without the need for continuous monitoring. When SoH variation is introduced, the complex aging characteristics reduce the prediction accuracy to 2% absolute error. If drive cycles are used for SoC changes, the error is again less than one percent for unaged cells. All tests, particularly those on fresh cells, outperform the control OCV-SoC curve lookup table approach by a factor of 2-3 times. The results demonstrate that the PIAML model is on par with or outperforms similar research and techniques.

PIAML performs well with mid SoC data, where the Li-Ion battery voltage profile tends to be flat and the method does not require a rest period. This makes PIAML particularly useful in applications where batteries are rarely fully charged or discharged such as in BMSs for hybrid EVs or grid storage batteries used for frequency regulation. The model can be applied when an updated SoC value is needed and then wait without an increase in error until the a new value is needed, saving computation time and freeing up a processor to perform other tasks. Alternatively, it can be applied in parallel with a constant monitoring system and provide periodic checks to minimize drift and ensure accurate SoC estimation over long time scales.

5. Acknowledgements

This research was undertaken, in part, through funding from the Columbia University Data Science Institute (DSI) Seed Fund Program. It was facilitated by NVIDIA Corporation with the donation of a Titan Xp GPU.

References

R. Jackson, C. Le Quéré, R. Andrew, J. Canadell, J. Korsbakken, Z. Liu, G. Peters, B. Zheng, P. Friedlingstein, Global Energy Growth Is Outpacing Decarbonization. A special report for the United Nations Climate Action Summit September 2019, Technical Report, 2019.

A. Emadi (Ed.), Advanced Electric Drive Vehicles, CRC Press, 2014. doi:10.1201/9781315215570.

K. Y. Bae, S. H. Cho, B. H. Kim, B. D. Son, W. Y. Yoon, Energy-density improvement in li-ion rechargeable batteries based on LiCoO2 + LiV3o8 and graphite + li-metal hybrid electrodes, Materials 12 (2019) 2025. doi:10.3390/ma12122025.

W. Wang, N. W. Brady, C. Liao, Y. A. Fahmy, E. Chemali, A. C. West, M. Preindl, High-Fidelity State-of-Charge Estimation of Li-Ion Batteries Using Machine Learning, ArXiv Preprint 1909.02448 (2019).

F. Baronti, G. Fantechi, L. Fanucci, E. Leonardi, R. Roncella, R. Saletti, S. Saponara, State-of-charge estimation enhancing of lithium batteries through a temperature-dependent cell model, in: International Conference on Applied Electronics, 2011.

H. Chaoui, H. Gualous, Online parameter and state estimation of lithium-ion batteries under temperature effects, Electric Power Systems Research 145 (2017) 73–82. doi:10.1016/j.epsr.2016.12.029.

W. Wang, M. Preindl, Dual cell links for battery-balancing auxiliary power modules: A cost-effective increase of accessible pack capacity, IEEE Transactions on Industry Applications 56 (2020) 1752–1765. doi:10.1109/tia.2019.2959728.

P. Keil, S. F. Schuster, J. Wilhelm, J. Travi, A. Hauser, R. C. Karl, A. Jossen, Calendar aging of lithium-ion batteries, Journal of The Electrochemical Society 163 (2016) A1872–A1880. doi:10.1149/2.0411609jes.

W. Waag, C. Fleischer, D. U. Sauer, Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles, Journal of Power Sources 258 (2014) 321–339. doi:10.1016/j.jpowsour.2014.02.064.

M. Charkhgard, M. Farrokhi, State-of-charge estimation for lithium-ion batteries using neural networks and EKF, IEEE Transactions on Industrial Electronics 57 (2010) 4178–4187. doi:10.1109/tie.2010.2043035.

E. Chemali, P. J. Kollmeyer, M. Preindl, R. Ahmed, A. Emadi, Long short-term memory networks for accurate state-of-charge estimation of li-ion batteries, IEEE Transactions on Industrial Electronics 65 (2018a) 6730–6739. doi:10.1109/tie.2017.2787586.

E. Chemali, P. J. Kollmeyer, M. Preindl, A. Emadi, State-of-charge estimation of li-ion batteries using deep neural networks: A machine learning approach, Journal of Power Sources 400 (2018b) 242–255. doi:10.1016/j.jpowsour.2018.06.104.

J. Du, Z. Liu, Y. Wang, State of charge estimation for li-ion battery based on model from extreme learning machine, Control Engineering Practice 26 (2014) 11–19. doi:10.1016/j.conengprac.2013.12.014.

X. Zhao, D. Xuan, K. Zhao, Z. Li, Elman neural network using ant colony optimization algorithm for estimating of state of charge of lithium-ion battery, Journal of Energy Storage 32 (2020) 101789.

doi:10.1016/j.est.2020.101789.

M. Ceraolo, G. Lutzemberger, D. Poli, C. Scarpelli, Luenberger-based state-of-charge evaluation and experimental validation with lithium cells, Journal of Energy Storage 30 (2020) 101534. doi:10.1016/j.est.2020.101534.

P. Shen, M. Ouyang, L. Lu, J. Li, X. Feng, The co-estimation of state of charge, state of health, and state of function for lithium-ion batteries in electric vehicles, IEEE Transactions on Vehicular Technology 67 (2018) 92–103. doi:10.1109/tvt.2017.2751613.

H. Chaoui, C. C. Ibe-Ekeocha, H. Gualous, Aging prediction and state of charge estimation of a LiFePO 4 battery using input time-delayed neural networks, Electric Power Systems Research 146 (2017) 189–197. doi:10.1016/j.epsr.2017.01.032.

Z. Xu, J. Wang, Q. Fan, P. D. Lund, J. Hong, Improving the state of charge estimation of reused lithium-ion batteries by abating hysteresis using machine learning technique, Journal of Energy Storage 32 (2020) 101678. doi:10.1016/j.est.2020.101678.

P. Malysz, J. Ye, A. Emadi, R. Gu, H. Yang, State-of-charge and state-of-health estimation with state constraints and current sensor bias correction for electrified powertrain vehicle batteries, IET Electrical Systems in Transportation 6 (2016) 136–144. doi:10.1049/iet-est.2015.0030.

E. Chemali, Intelligent State-of-Charge and State-of-Health Estimation Framework for Li-ion Batteries in Electrified Vehicles using Deep Learning Techniques, Phd thesis, McMaster University, 2018.

R. Sebastian, An overview of gradient descent optimization algorithms, ArXiv Preprint 1609.04747 (2016).

X. Chen, H. Lei, R. Xiong, W. Shen, R. Yang, A novel approach to reconstruct open circuit voltage for state of charge estimation of lithium ion batteries in electric vehicles, Applied Energy 255 (2019) 113758. doi:10.1016/j.apenergy.2019.113758.

C. Bo, B. Zhifeng, C. Binggang, State of charge estimation based on evolutionary neural network, Energy Conversion and Management 49 (2008) 2788–2794. doi:10.1016/j.enconman.2008.03.013.

Z. Zhou, B. Duan, Y. Kang, N. Cui, Y. Shang, C. Zhang, A low-complexity state of charge estimation method for series-connected lithium-ion battery pack used in electric vehicles, Journal of Power Sources 441 (2019) 226972. doi:10.1016/j.jpowsour.2019.226972.

C. Bian, H. He, S. Yang, T. Huang, State-of-charge sequence estimation of lithium-ion battery based on bidirectional long short-term memory encoder-decoder architecture, Journal of Power Sources 449 (2020) 227558. doi:10.1016/j.jpowsour.2019.227558.