Accelerating battery development via early prediction of cell lifetime

Peter Attia, Marc Deetjen, Jeremy Witmer
Stanford University

***Abstract*—Early prediction of battery lifetime would aid in their development, manufacture, and optimization. Furthermore, for high-rate applications such as fast charging, first-principles electrochemical models fail to capture the dynamics. In this work, we develop machine learning models to predict the final cycle life using the first 20-100 cycles for a dataset consisting of commercial lithium-ion batteries cycled to failure during fast charging. We develop a novel visualization of voltage data, identifying a linear trend that is useful for early prediction. Our best models achieve around 12% mean percent error for cycle numbers from 40 to 100. This work demonstrates high prediction accuracy at low cycle number, accelerating battery high-throughput screening applications.**

***Index Terms*—Batteries, materials science, remaining useful life prediction**

I. Introduction

Lithium-ion batteries are deployed in a wide range of applications due to their low costs, high energy densities, and long cycle lives. However, long battery cycle life entails delayed feedback of performance, often many months to years. The high manufacturing variability of commercial lithium-ion batteries makes this task challenging. Furthermore, electrochemical models fail to capture the dynamics of degradation during operating modes of interest, such as fast charging. Accurate prediction of cycle life using early-cycle data would unlock new opportunities in battery development, manufacturing, and optimization.

The goal of this work is to predict the final cycle life (1000s of cycles) using data from the first 100 cycles or fewer. Here, cycle life is defined as the number of cycles before the capacity falls below 80% of the rated capacity. We build predictive models using the elastic net, random forest regression, and AdaBoost regression. Our inputs are data including voltage, capacity, temperature, and internal resistance as a function of cycle number.

II. Related Work

Early prediction of battery lifetime has been widely studied, but has proven difficult. Harris *et al.* [1] found a weak correlation (ρ=0.1) between capacity values at cycle 80 and capacity values at cycle 500 for 24 cells exhibiting nonlinear degradation profiles, illustrating the difficulty of this task. Others have used more intensive data-driven approaches including neural networks [2], [3], [4], [5], [6], [7], [8] and support vector machines (SVM) [9], [10], [11], [12]. Typically, these authors are limited by extremely small data set sizes (<10 cells) and make predictions only after the cell has nearly failed.

In our previous work [13], we generated a dataset of 124 commercial lithium-ion batteries (see Figure 1). This work was performed in the Chueh lab at Stanford Materials Science and Engineering. Our objective was to predict the final cycle life using data from only the first 100 cycles. Using an elastic net fit (i.e. regularized linear regression), we achieved a test error of 9.1%. However, further reducing the number of cycles required for prediction could accelerate high-throughput optimization applications.

In this work, our objective is to predict battery cycle life using only the first 20-50 cycles while achieving comparable or improved accuracy to our previous results, which used the first 100 cycles. We focus on both improving feature engineering and applying more advanced machine learning methods.

III. Dataset

Our dataset consists of 124 nominally identical commercial lithium-ion batteries cycled under various fast charging conditions. Some properties of this dataset are displayed in Figure 1. The cycle lives range from 150 to 2,300; this variation in lifetime comes from using 72 different fast-charging conditions. Figure 1b illustrates the weak relationship between initial capacity and cycle life (color). Figure 1c illustrates that the capacity initially rises, suggesting the difficulty of early prediction.

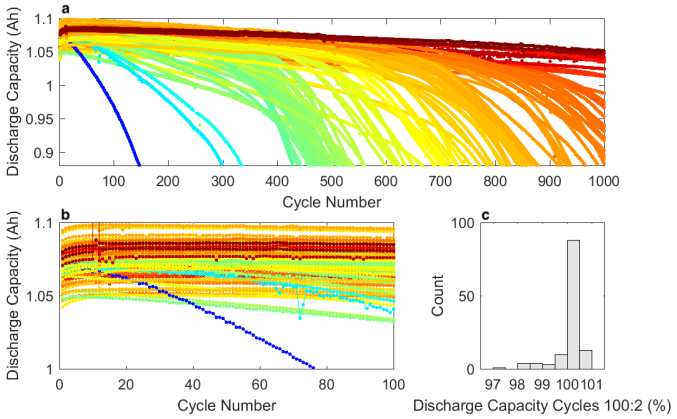

Fig. 1. Dataset depiction. (a) Discharge vs cycle number for the first 1,000 cycles of our 124 cells. The variation in lifetime (150 to 2300 cycles) comes from different charging policies. Red and blue indicate high and low cycle life, respectively. (b) A detailed view of (a), showing only the first 100 cycles. A clear ranking of cycle life has not emerged by cycle 100. (c) Histogram of the capacity ratio between cycle 100 and cycle 2. For most cells in this dataset, the capacity initially rises in the first 100 cycles, suggesting prediction will be difficult.

Our training set consists of 84 cells, while our test set consists of 40 cells (consistent with previous work[13]). This dataset is not yet publicly available but will be soon.

IV. Feature Generation

Because we have a relatively small dataset (124 cells), careful feature selection is key to creating a successful predictive model of cell lifetime.

The most complex data source consists of high dimensional voltage data, with 1000 features per cycle. However, features from this dataset are expected to be among the most predictive. Figure 2 illustrates our novel visualization of voltage data. Figure 2a displays voltage vs capacity for the first 100 cycles for a typical cell in our dataset. The voltage shifts slightly to the left with increasing cycle number. Figure 2b displays this same data visualized in a three-dimensional heat map. The contrast in cycle number is low, since the shifts in voltage are low. To enhance the contrast, we subtract the second-cycle capacity from this heat map to visualize the *change* in voltage, as shown in Figure 2c. The contrast in cycle life is largest at 2.9 V. Lastly, we plot the baseline-subtracted discharge capacity at 2.9 V (Figure 2d). Surprisingly, this plot reveals a linear relationship, which is highly encouraging from a predictive standpoint.

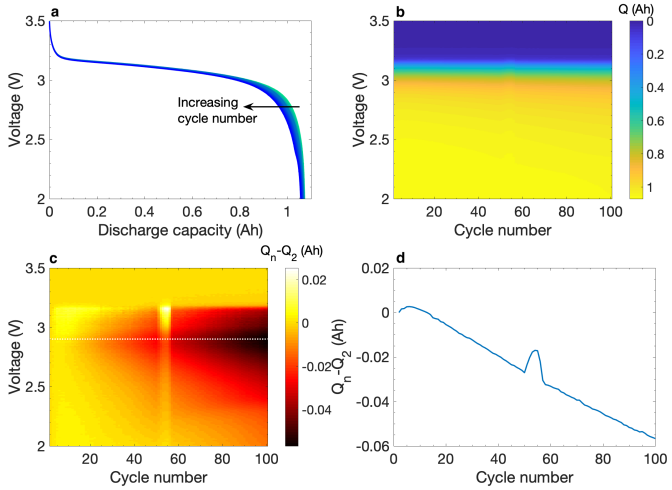

Fig. 2. Voltage visualizations for feature generation. (a) Voltage vs discharge capacity for cycles 1 through 100 for a typical lithium-ion battery. (b) Discharge capacity as a function of voltage (y-axis) and cycle number (x-axis). This plot is another way to visualize the data in (a). (c) Discharge capacity as a function of voltage (y-axis) and cycle number (x-axis) with the second-cycle capacity subtracted. (d) Baseline-subtracted capacity vs cycle number at 2.9 V. The peak around cycle 50 can be attributed to a temperature rise in the environmental testing chamber.

Before extracting features from the median-subtracted capacity heatmap, we filter out noise due to temperature variations which appear around cycles 50-55 for many of our cells. We fit a 2nd order surface polynomial using RANSAC [14] to the median-subtracted data in the region from 2.0-3.155 volts (the region from 3.155-3.5 volts was constant) and from cycles 8 through the number of cycles used in the prediction. For the polynomial fit, we normalized the axis prior to fitting. We chose the best of 30 random initializations of RANSAC and defined outliers as points which were not fit to within 0.007 Ah. Any outliers were replaced with the polynomial fit as shown in Figure 2.

In addition to eliminating outliers, a RANSAC surface fit is useful for generating features themselves. We fit a 2nd, 3rd, and 4th order surface polynomial to find which parameters were most useful for fitting the surface. We selected a simpler model as detailed in Eqn. 1:

$$z = p_1 + p_2 * y + p_3 * x * y, \tag{1}$$

where p_1, p_2, p_3 are fitting parameters (or features), x is the cycle number, y is the voltage, and z is the baseline-subtracted discharge rate. Outliers here were more strictly defined as anything not fit to within 0.003 Ah, and the features generated with this method had relative success.

If we compare these voltage visualizations for a typical good cell and a typical bad cell (Figure 2c), the most striking difference is that the bad cell has a large asymmetry between the left and right side of the plot. The difference between the charging curves at the beginning and end of the first 100 cycles was found to be an important feature in the previous paper using this data. To examine this further, we can look at an individual horizontal slice of the data (Figure 2d). Using only the first 40 cycles, we found that the slope in the high-contrast region of the curve (at 2.9 V) correlates strongly with final cell lifetime (r=0.79).

In addition to the slope of the discharge curve vs. cycle number, we also explored a number of other features extracted from the capacity heatmap. Although some of these features are highly correlated with the cycle life, they also have high correlation with one another. While some features may be redundant representations of the same information, adding regularization and pruning to our regression algorithm should mitigate issues caused by redundant features.

V. Methods

We build models with three regression methods in this work: elastic net, random forest regression, and AdaBoost regression.

A. Elastic net

The first regression model we employed was elastic net regression, in which the parameter vector θ is defined by the following equation:

$$\theta = \text{argmin}_\theta\{\|y - X\theta\|_2^2 + \alpha(\lambda\|\theta\|_1 + (1-\lambda)\|\theta\|_2^2)\} \tag{2}$$

Elastic net regression combines the benefits of both 1-norm and 2-norm regularization in linear regression (also called lasso and ridge regularization, respectively). The main difference between these two regularization methods is in the way they treat correlated features. While 2-norm regularization will constrain the weights of correlated features in a more or less uniform way, 1-norm regularization implements a "winner-take-all" behavior in which one of the correlated features will be selected and the others will be ignored. The advantage of lasso regularization is that it performs feature selection, which can improve both model accuracy and interpretability when the dataset is feature-rich. Adding the 2-norm penalty term

to lasso regression has been shown to improve performance, especially when the number of features is large compared to the number of data points.

We started by applying elastic net regression to our full dataset of 64 engineered features. By investigating the relative weights assigned to these features by the elastic net we were able to select a smaller subset of 13 important features which we used for the remainder of our analysis. We found that all three of our models performed better using this smaller set of features rather than the orginal set, likely because having a reduced number of features helped to prevent overfitting.

The hyperparameters involved in elastic net regression are α, the overall regularization factor, and λ, the l-1 ratio which sets the relative importance of the 1-norm and 2-norm regularization. In the `scikit-learn` implementation of elastic net regression, the algorithm performs stochastic gradient descent with cross-validation to optimize over these hyperparameters automatically. We optimized over the hyperparameters separately for each number of cycles. We found that the optimum combination varied, but the optimal value for λ tended to be around 0.01, indicating that the model was relying more on the 2-norm for our downselected feature set. The optimal value of α ranged between 0.001 and 0.01.

B. Random forest regression

Since our dataset is feature rich but contains a relatively small number of training examples (only 84), we found that our models tended to be prone to overfitting. We decided to try random forest regression because of its ability to reduce overfitting by aggregating results.

Random forest regression averages the results from a number of decision trees. Decision tree regression works by dividing the data recursively and assigning the mean value of the leaf training examples to all points contained in that leaf. At each split, it selects the optimal feature and split value to minimize the loss.

For a random forest with N trees, the first step is to generate N different training sets of the same size as the original by sampling the original with replacement (this process is known as *bootstrapping*). The individual decision trees are then trained separately on their respective data sets. To further decorrelate the trees, only a randomly selected subset of the features are made available at each split decision. Finally, after the trees are trained, the output of the random forest is just taken to be the average of the predictions of the constituent trees. Because the predictions are averaged in this way, the variance of the model can be greatly reduced by simply increasing the number of decision trees.

Applying random forest regression to our data, we swept over the number of trees and the maximum depth of trees. We found that the results generally improved with increasing number of trees and increasing tree depth. For our final testing we chose a model with 1000 trees and no limit to the maximum tree depth.

C. AdaBoost regression

AdaBoost (short for "adaptive boosting") regression is a popular regression method that relies on the idea of *boosting*, in which the results of many "weak learners" are combined in a weighted average to achieve an accurate prediction. The algorithm is adaptive because each training example is assigned a weight which varies over time. If a weak learner performs poorly on a particular training example, the weight of that example is increased so that the next weak learner will be more likely to fit it correctly. Decision stumps (decision trees with small depths) are commonly used weak learners.

In our application of AdaBoost we used decision trees of depth 3 as our weak learners. As our hyperparameters we swept over the number of trees and the learning rate (the factor by which the predictions of subsequent learners are discounted). The optimal learning rate was found to be 0.1 and the model performance was found to plateau with approximately 1000 trees.

D. Cross-validation and scoring

For each of these three learning methods we employed 5-fold cross-validation to tune the hyperparameters. This method allowed us to make the best use of our small training set. We scored the models based on the achieved mean percent error, which is a more important figure of merit for battery lifetime prediction than mean-squared error.

VI. Results and Discussion

A. Prediction results

Figure 3 displays mean percent error of our three methods - elastic net (3a), random forest regression (3b), and AdaBoost regression (3c) - as a function of cycle number. Figure 4 displays observed vs predicted cycle life for the case of 100 cycles used for prediction. Not surprisingly, we find that the mean percent error generally increases with decreasing cycle number, ranging from 5-10% train error and 10-15% test error around cycles 80-100 and from 10-15% train error and $\sim$35% train error around cycle 20.

We find that random forest regression most consistently achieves the lowest test error of our three methods. Our hypothesis is that random forest regression best captures the bias in our dataset without overfitting to outliers. Ultimately, linear regression is limited in its ability to reduce bias, while AdaBoost regression is highly sensitive to outliers. In Figure 54c, we find that the five high-lifetime cells in the training set have low error with the Adaboost model, while the high-lifetime cells in the training set have high error. This observation implies overfitting, particularly for high-lifetime cells. We believe we underestimated the effect of these outliers during model training. In contrast, random forest regression is well suited for low bias and variance.

For all three methods, the test error is consistently higher than the training error. We attribute this result to three sources. First, our models may be overfit. We attribute this both to suboptimal hyperparameter optimization, as well as the specific attributes of the dataset (i.e. high-lifetime cells)

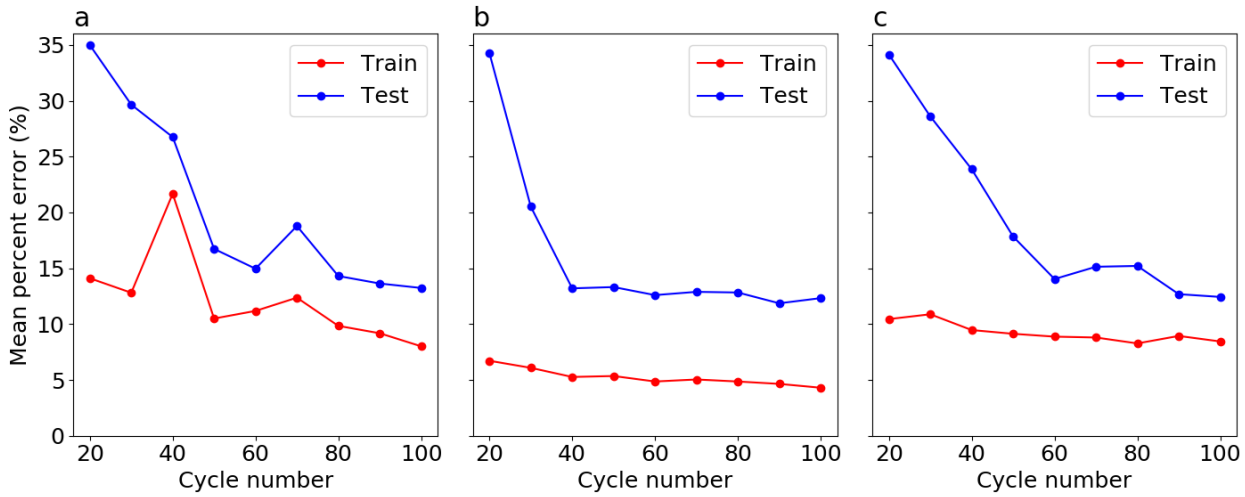

Fig. 3. Mean percent error vs number of cycles used in prediction for (a) elastic net regression, (b) random forest regression, and (c) AdaBoost regression.

for AdaBoost regression. Second, the test set was selected to maintain consistency with previous work. However, this data was generated from a separate testing "batch" of cells. Each batch is subject to small experimental differences such as temperature fluctuations, etc. We expect our models to be sensitive to these differences. Finally, our sensitive error metrics are sensitive to our small sample sizes.

B. Feature selection

Figure 5 displays the selected features and their coefficients for elastic net regression, while Figures 6 and 7 display feature importance for random forest regression and AdaBoost regression. These plots reveal which features our models most heavily rely on as a function of number of cycles used in prediction and help diagnose issues in prediction.

All three figures reveal the same general trend. At low cycle number (20-30), the `chargetime` feature dominates the elastic net weights and importances, with other features like the `IR` (internal resistance) features also contributing. The `chargetime` feature is a feature related to the time required for the cell to charge; from previous work, we know that that cycle life is highly sensitive to charge time. At low cycle number (20-30 cycles), the changes in capacity as presented in Figure 2c have yet to develop. As a result, the model relies heavily on `chargetime`, which is independent of the degradation. These models perform poorly on the test set, since charge time is not the only factor that determines the lifetime; for example, two charging policies (i.e. combinations of current steps) may have the same charging time but yield dramatically different lifetimes.

At high cycle number (∼40-100), features such as `log_slope_2pt9V_corr` and `log_DeltaQ_var` are selected. These features are derived from the capacity curve heatmaps presented in Figure 2. In fact, `log_slope_2pt9V_corr` is the slope of the curve presented in 2d. At this point, the degradation has manifested in the capacity heatmaps (see Figure 2c). The performance of all models is generally good at these cycle numbers. Thus, features based on the measured changes in capacity are most predictive of cycle life, and these features are best used when the degradation is discernible (i.e. by cycle 40).

Specific attributes of each feature importance map are also revealing. For elastic net regression, we note that for most cycle numbers, nearly all features are selected. We attribute this result to the initial feature screening that yielded a subset of predictive features. Another possibility is that all features are selected due to overfitting, given the high variance of the results (i.e. discrepency between train and test). We also note that only four features were selected for the $n = 40$ case, which also corresponded to high errors of both test and train.

For random forest regression, we find that while `chargetime` is most predictive at low cycle number, the five features consistently selected are all derived from the capacity heatmaps. These models consistently perform well at cycle numbers from 40 to 100.

For AdaBoost regression, we find that for a given number of cycles used in prediction, feature importance is generally dominated by a single feature. The overreliance on single features leads to models that do not outperform random forest regression.

VII. Conclusions and Future Work

Machine learning techniques are a promising approach for early prediction of battery cycle life. In this work, we develop novel features that capture voltage data and apply state-of-the-art learning techniques to reduce the number of cycles (∼40) required for prediction while achieving similar performance (∼12% mean percent error). Overall, we find that tree-based methods, specifically random forest regression, achieve high accuracy at low cycle number, outperforming regularized linear regression. We attribute this result to the

We have a few proposals for future work. Primarily, we will try to reduce overfitting by performing a test/train split over all batches and by using packages for automated hyperparameter

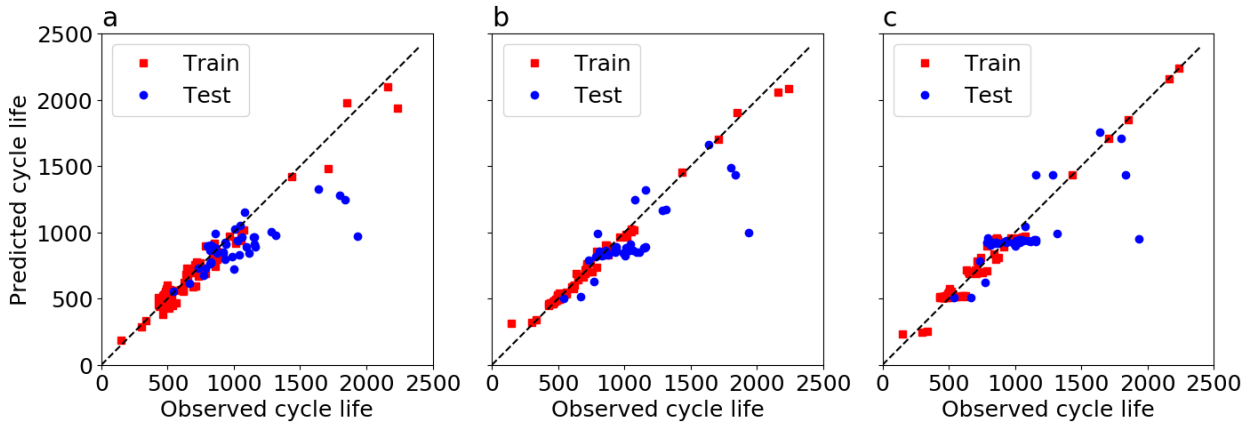

Fig. 4. Mean percent error vs number of cycles used in prediction for (a) elastic net regression, (b) random forest regression, and (c) AdaBoost regression.

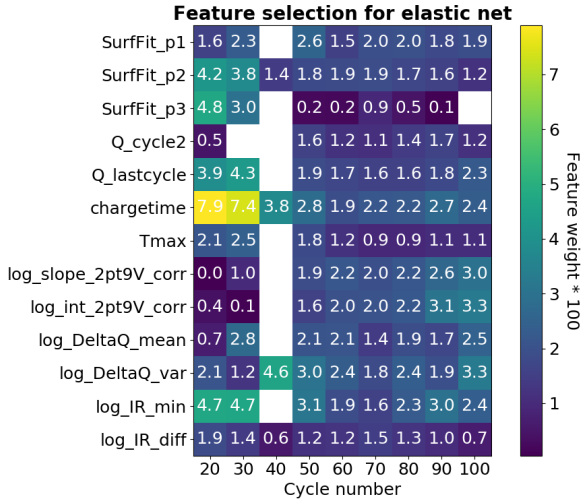

Fig. 5. Feature weights for elastic net regression as a function of number of cycles used in prediction. The weights are normalized by the standard deviation of the features to enable direct comparison.

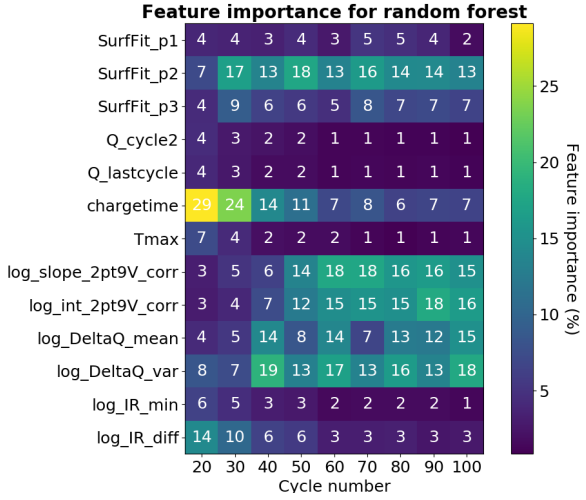

Fig. 6. Feature importance for random forest regression as a function of number of cycles used in prediction.

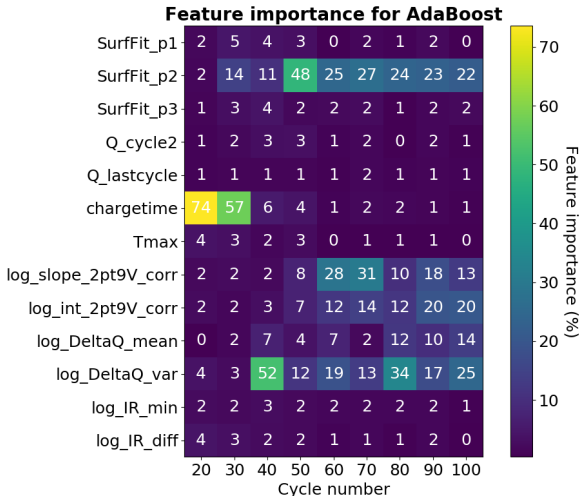

Fig. 7. Feature importance for AdaBoost regression as a function of number of cycles used in prediction.

optimization. Additionally, we will expand our feature set by incorporating features from other sources of data. components of the data set, such as rest periods, charging, constant-voltage holds, etc. Applying principle components analysis on the voltage heatmaps and using the component weights as features is a promising path forward. Lastly, developing a convolutional neural network to X-ray tomography images taken before cycling could be used to detect manufacturing defects, which we expect are strong predictors of cycle life.

We will also consider other use cases of early prediction. Classification of cells into low-lifetime and high-lifetime classes would be useful in preliminary screening applications in manufacturing and charging policy optimization; we expect significantly fewer cycles (on the order of 5-10) would be required for this application. Lastly, we will employ this early prediction model in a reinforcement learning context to efficiently screen for fast charging policies with high lifetime.

CONTRIBUTIONS

All authors contributed to data exploration, feature generation, model development, and report writing.

REFERENCES

1 Harris, S. J., Harris, D. J., and Li, C., "Failure statistics for commercial lithium ion batteries: A study of 24 pouch cells," *Journal of Power Sources*, vol. 342, pp. 589–597, 2017.

2 Eddahech, A., Briat, O., Bertrand, N., Deltage, J.-Y., and Vinassa, J.-M., "Behavior and state-of-health monitoring of li-ion batteries using impedance spectroscopy and recurrent neural networks," *International Journal of Electrical Power & Energy Systems*, vol. 42, no. 1, pp. 487 – 494, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0142061512001779

3 Eddahech, A., Briat, O., Bertrand, N., Delétage, J.-Y., and Vinassa, J.-M., "Behavior and state-of-health monitoring of li-ion batteries using impedance spectroscopy and recurrent neural networks," *International Journal of Electrical Power & Energy Systems*, vol. 42, no. 1, pp. 487–494, 2012.

4 Kim, J., Lee, S., and Cho, B., "Complementary cooperation algorithm based on dekf combined with pattern recognition for soc/capacity estimation and soh prediction," *IEEE Transactions on Power Electronics*, vol. 27, no. 1, pp. 436–451, 2012.

5 Bai, G., Wang, P., Hu, C., and Pecht, M., "A generic model-free approach for lithium-ion battery health management," *Applied Energy*, vol. 135, pp. 247–260, 2014.

6 Wu, Z., Cao, L., Chao, T., Li, T., Zeng, L., and Hu, B., "Research of modified elman neural network in the lithium-ion battery capacity prediction method," *J. Southwest Univ. Sci. Technol*, vol. 27, pp. 65–69, 2012.

7 Liu, J., Saxena, A., Goebel, K., Saha, B., and Wang, W., "An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries," National aeronautics and space administration Moffett Field CA AMES Research, Tech. Rep., 2010.

8 Zhang, Y., Xiong, R., He, H., and Pecht, M., "Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries," *IEEE Trans. Veh. Technol.*, vol. 1-1, 2018.

9 Nuhic, A., Terzimehic, T., Soczka-Guth, T., Buchholz, M., and Dietmayer, K., "Health diagnosis and remaining useful life prognostics of lithium-ion batteries using data-driven methods," *Journal of power sources*, vol. 239, pp. 680–688, 2013.

10 Wang, S., Zhao, L., Su, X., and Ma, P., "Prognostics of lithium-ion batteries based on battery performance analysis and flexible support vector regression," *Energies*, vol. 7, no. 10, pp. 6492–6508, 2014.

11 Dong, H., Jin, X., Lou, Y., and Wang, C., "Lithium-ion battery state of health monitoring and remaining useful life prediction based on support vector regression-particle filter," *Journal of power sources*, vol. 271, pp. 114–123, 2014.

12 Klass, V., Behm, M., and Lindbergh, G., "A support vector machine-based state-of-health estimation method for lithium-ion batteries under electric vehicle operation," *Journal of Power Sources*, vol. 270, pp. 262–272, 2014.

13 Severson, K. A., Attia, P. M., Jin, N., Perkins, N., Jiang, B., Yang, Z., Chen, M. H., Aykol, M., Herring, P. K., Fraggedakis, D., Bazant, M. Z., Harris, S. J., Chueh, W. C., and Braatz, R. D., "Data-driven prediction of battery cycle life before capacity degradation," 2018, manuscript in review. [Online]. Available: https://www.dropbox.com/s/emzk9y2aimzto14/2018-10-01_main.pdf?dl=0

14 Fischler, M. A. and Bolles, R. C., "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

CODE AVAILABILITY

All code is available at https://github.com/petermattia/predicting-battery-lifetime. Features were generated in MATLAB, and machine learning was performed in python.