BATTERY [LIFE] NOT INCLUDED

USING SEQUENCE MODELS TO PREDICT THE USEFUL LIFE OF BATTERIES

MICHAEL WHITMEYER, ADEL SETOODEHNIA, SAMUEL PARADIS

Abstract. In this paper, we use data on 124 batteries released by Stanford to first try to solve the binary classification problem of determining if a battery is "good" or "bad" given only the first 5 cycles of data (i.e., will it last longer than a certain threshold of cycles), as well as the prediction problem of determining the exact number of cycles a battery will last given the first 100 cycles of data. We approach the problem from a purely data-driven standpoint, hoping to use deep learning to learn the patterns in the sequences of data that the Stanford team engineered by hand. For both problems, we used a similar deep network design, that included an optional 1-D convolution, LSTMs, an optional Attention layer, followed by fully connected layers to produce our output. For the classification task, we were able to achieve very competitive results, with validation accuracies above 90%, and a test accuracy of 95%, compared to the 97.5% test accuracy of the current leading model [Sev19]. For the prediction task, we were also able to achieve competitive results, with a test MAPE error of 12.5% as compared with a 9.1% error achieved by the current leading model [Sev19].

1. Problem Statement and Background

In this paper we aim to tackle the problem of predicting how long a lithium-ion battery will last before it "dies", meaning it reaches approximately 80% of its original capacity. There are many factors which go into a battery's state of health (SOH), and a lot of complex chemical processes [C.B99] [J.V05]. Dendrites can build up in the battery, and a phenomenon known as SEI layer growth can both lead to degradation in battery capacity, but have been difficult to accurately and quantitatively model with known equations [J.V05]. This has led some to try a semi-empirical or fully empirical approach to battery aging. Recently, Stanford and MIT released a joint effort [Sev19] attempting to predict how many *cycles* a battery would last, where a cycle is simply a full charge and discharge of the battery. They tried to predict two things: 1) given data on the first 5 cycles, predict whether the battery will last past a certain cycle threshold, which is a binary classification problem, and 2) given data on the first 100 cycles, predict exactly how many cycles the battery will last. They attempted to solve these problems by engineering features and performing regularized linear regression in the engineered feature space. We will attempt to solve the problem by using deep learning architectures that can process sequences of data in order to allow relevant patterns and features to be learned rather than hand-selected.

Solving the problem of battery degradation uncertainty is important because a solution has the potential to get new battery technologies to the market faster, save materials/energy, and allow consumers to have access to more consistently reliable batteries. To get an accurate understanding of the cycle-life of a battery, currently you must drain thousands of these batteries to failure. Further, every batch of batteries has outliers: a small percentage of batteries produced will have significantly shorter cycle-life than expected. Is there a way we could classify and predict the cycle-life of a Lithium-ion battery with deep learning to both analyze battery life

Date: May 13, 2019.

and identify outliers, without actually ever draining the batteries themselves, which wastes materials and energy?

In order to answer this question, we will split the dataset of 124 batteries into a training set, validation set, and test set. For classification, the accuracy will simply be the percentage of correct classifications. For predicting the exact cycle-life, the accuracy will be in terms of the MAPE (mean absolute percentage error) accuracy, which as far as we can tell is the same metric Stanford used to measure their success. We hoped to achieve accuracies comparatively competitive to Stanford's, which represents the current leading model for battery degradation.

2. Data

The data we used is associated with the paper 'Data driven prediciton of battery cycle life before capacity degradation' by K.A. Severson, P.M. Attia, et al. [Sev19]. The researchers have made the dataset–the largest of its kind–publicly available. The data consists of 124 batteries, with each battery lasting a certain number of cycles, and each cycle containing data on various measurements, such as temperature, voltage, current, and charge/discharge capacity for each time step of the cycle. Processing the data involved extracting information from a set of nested dictionaries, and then interpolating or zero padding where necessary to ensure the data matrix would be a tensor. Cleaning the data involved developing a simple outlier removal tool to remove noisy readings. An outline of the variables observed over the course of each cycle is in Table 1:

Variable	**Description**
T	Temperature
V	Voltage
Qd	Discharge Capacity
Qd-lin	Linearly interpolated discharge capacity (over cycle)
Td-lin	Linearly interpolated temperature (over cycle)
dQdV	Change in discharge capacity over change in voltage

Table 1. An attribute is the mean, variance, minimum, or maximum of a variable over an entire cycle.

A key problem we faced is that we wanted our model to learn complex sequential patterns, but the dataset is quite small, with only 124 (X, Y) data-label pairs. Each of the batteries had hundreds of cycles, and each cycle had thousands of data points, but we found that much of the data was 1) very similar looking 2) not useful for prediction. With the model initially failing to learn when passing in all of the possible data within each cycle, we determined that we needed to reduce the data (via mean, variance, max, min), and then identify the attributes that provide meaningful insights for the model. In order to do so, we visualized each variable. These plots are relevant because at each timestep, a battery contains a significant amount of information, and in order to optimize the ability of the LSTM-based models to learn, we needed to pass in the most relevant data.

Each point on the plot represents the cycle-life of a battery as function of the change between cycle 100 and cycle 10 of some reduction function over all the data for some variable collected during the cycle. These plots identify the variables that provide meaningful information. Looking at the plots, any plot where the data is spread uniformly vertically conveys little information for predicting cycle-life; the cycle life is independent from the attribute in this case.

Very Useful Attributes

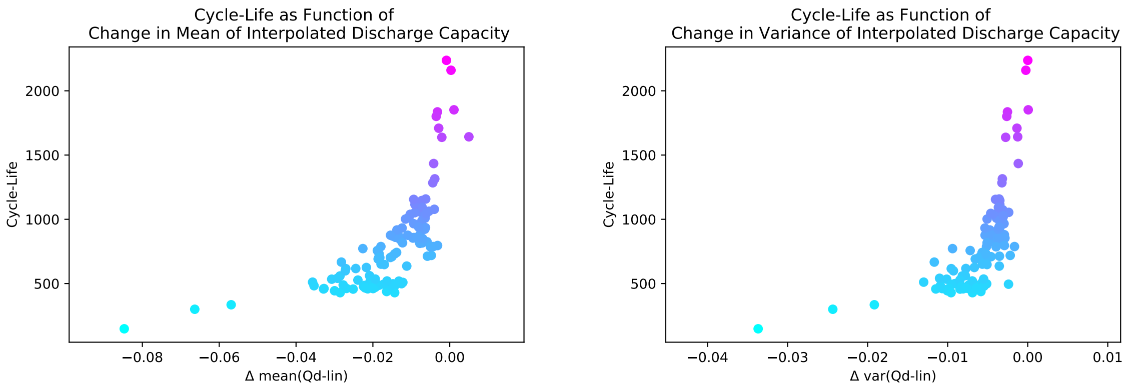

FIGURE 1. Top ranked attributes: mean(Qd-lin), var(Qd-lin), var(dQ/dV). For these attributes, we see the value varied between cycles, and this change directly impacted the cycle-life.

Somewhat Useful Attributes

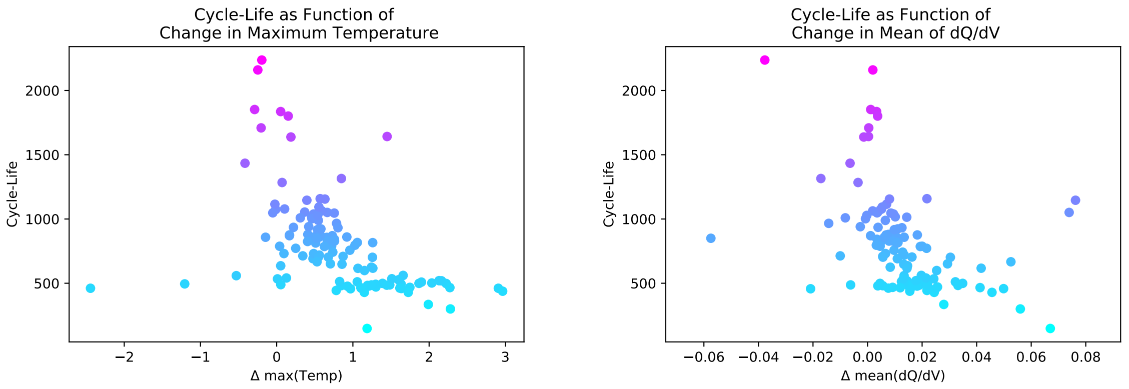

FIGURE 2. Middle ranked attributes: max(T), mean(dQ/dV), mean(Qd), max(Qd-lin), min(T), var(T), mean(Td-lin), max(Qd), mean(T), max(Td-lin), var(Td-lin), min(Td-lin). For these attributes, we see the value varied between cycles, and this variation impacted the cycle-life.

Not Useful Attributes

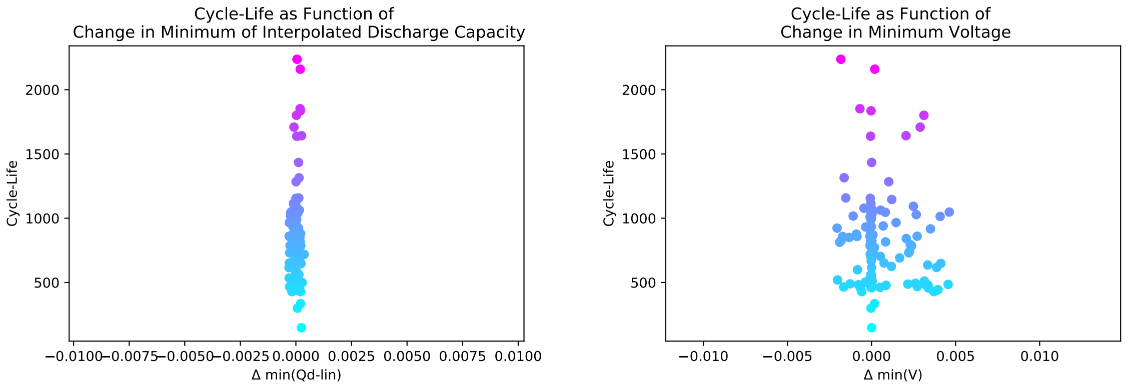

FIGURE 3. Lowest ranked attributes: min(Qd-lin), min(V), min(Qd), max(V), mean(V), var(V), min(dQ/dV), max(dQ/dV), var(Qd). For these attributes, we see the value fails to vary between cycles, and when it does, this variation does not seem to impact the cycle-life.

The input to our model was the top and middle ranked attributes of the first 5 cycles for classification and first 100 cycles for prediction, passed in sequentially by cycle.

3. Models

While Stanford relied on the physical properties of batteries, as well as human trial and error, to identify the relevant sequential relations between cycles, our sequence model allowed for these sequential relations to be learned. The state of the art model processes the data by passing in the differences between certain values in the 5th and 1st cycle for classification, and 100th and 10th cycle for prediction. Instead of using differences, we wanted to leverage LSTMs and/or Attention to process the data of the cycles sequentially to allow relevant patterns be learned rather than hand-selected. The output of this sequence section is passed into dense layers to both increase the flexibility of the model to fit the data, as well as to reduce the output to a single number.

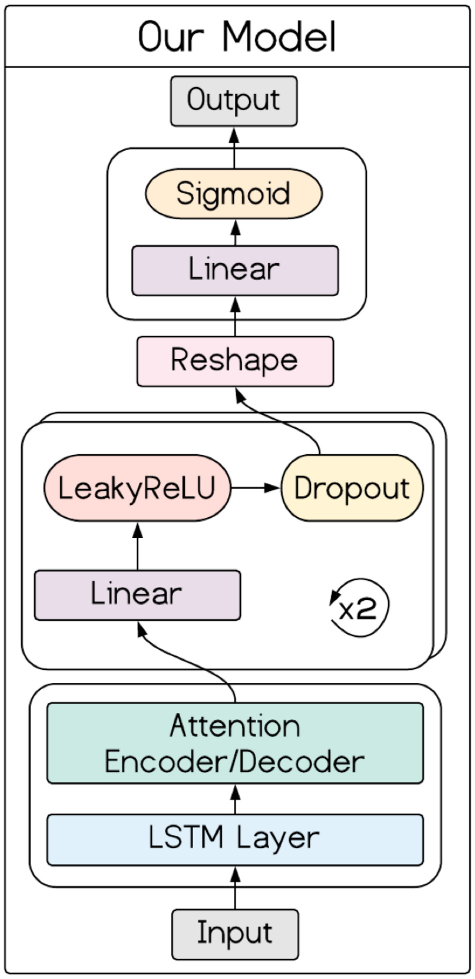

FIGURE 4. Baseline Model. Note the Attention Encoder/Decoder is optional, and the sigmoid activation was only used for the classification task.

3.1. **Classification.** For the classification problem, we tuned the baseline model to maximize performance for the classification task. This involved iterating over different models and evaluating them based on their training and validation performance. We investigated models with LSTM/Attention hidden vector sizes ranging from 16 to 256, dense layers with sizes ranging from 16 to 256, with and without dropout, and also with and without a second and third hidden layer. We found that the more complex models struggled with overfitting the training data, which makes sense considering the small size of the dataset. Validation accuracy was low for many of the complex models, especially when the LSTM size was at or above 128. In order to allow the model to generalize well, we used a simple form of our base model:

LSTM/Attention hidden size of 16, followed by two dense layers with 16 outputs and one dense layer with one output. Used dropout with .3 probability and LeakyReLU between the dense layers, ending with a sigmoid activation. Figure 4 shows the training and validation accuracy over 500 epochs for this model, which generalized well to new data.

3.2. **Prediction.** In order to predict the exact number of cycles a battery would last, we followed Stanford's example and used the first 100 cycles of data as our input. We experimented with using summary data (for example, the mean of the temperature over the cycle) vs. actually using all the data available within a cycle, but we once again found that summarizing the data in a smart way was more effective than trying to throw all of our data (the vast majority of which was not useful for prediction) at our model, and hoping it learned what was useful and what was not. However, we also found that it was very useful to add a 1-D convolutional layer before feeding our sequence of 100 cycles into our LSTM. The reasoning behind this is that LSTMs (despite their name), can be quite forgetful, and learning a sequence of 100 inputs can be difficult to train, especially with our limited amount of data. We also considered using some sort of average or max pooling to downsample the data, but then we decided to go for the 1-D convolutional layer instead, as the 1-D convolution is learnable and therefore we wouldn't be making any assumptions about which parts of the data were useful, letting the network figure that out for itself. We also experimented with using an attention layer, but ultimately found that our best model did not use the attention layer (however, the results were comparable with and without the attention). We used an augmented form of our base model:

1-D Convolution (15 filters, stride 4, height 4), LSTM hidden size of 32, followed by two dense layers with 64 outputs, one dense layer with one output, using dropout with 0.2 probability and LeakyReLU between the dense layers, ending with no activation function. Figure 9 shows the training, validation, and test MAPE error for this model.

4. Results

4.1. **Classification.** The entire 124 battery dataset was used to evaluate the model. The data was partitioned as followed: 79 train, 25 validation, 20 test. The dataset was partitioned randomly. Each model was trained on 79 batteries, and the first 5 cycles of the battery were inputted as a sequence into the model. At the end of our model search, we tested the best performing model on the 20 test batteries. Figure 5 shows the training and validation accuracy over 500 epochs for this model.

	Train	Validation	Test
Accuracy (%)	90.5	90.4	95.0

FIGURE 5. Classification accuracy results

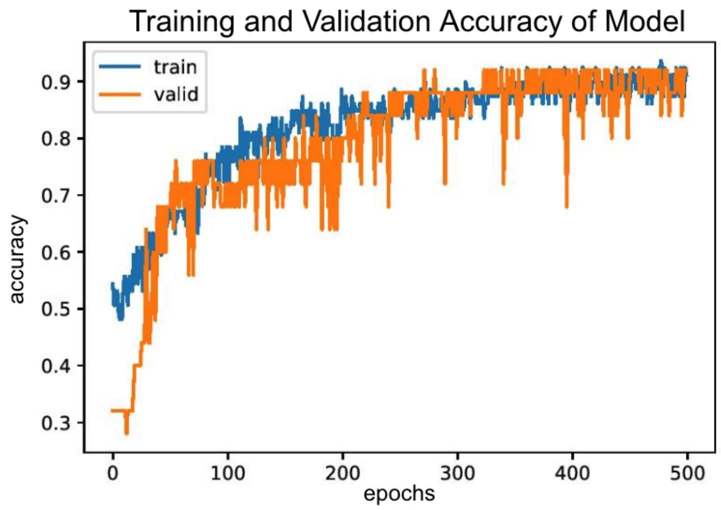

FIGURE 6. Training and validation accuracy over 500 epochs with our simple, yet best preforming model. The noise is likely due to our entire dataset only being 124 batteries: 79 train, 25 valid, 20 test.

After developing this simplified model, we hoped it would translate well to the test set, and results could be competitive with that of Stanford. Performance on the test set is visualized in Figure 5.

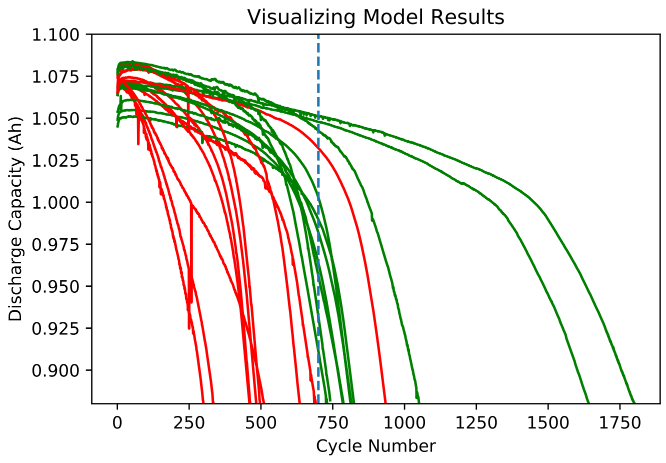

FIGURE 7. Discharge capacity (Qd) as a function of cycle number on our test set (95% accuracy). When the Qd lowers to the bottom of the figure, the battery is declared dead, and the cycle life is determined. The blue vertical line is at cycle 700. Each curve is a battery; looking at just the first 5 cycles, the green curves were predicted to last longer than 700 cycles, the red shorter. We see that 19 out of 20 batteries were correctly classified; one battery was classified as "bad", but lasted longer than the cutoff.

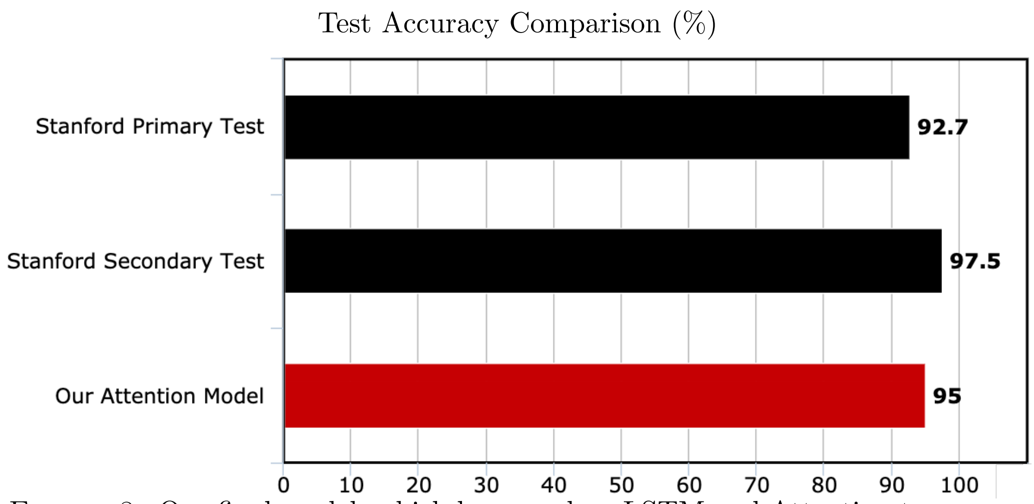

FIGURE 8. Our final model, which leveraged an LSTM and Attention to process the sequential data, compares to Stanfords state of the art model on a held out test set of 20 batteries. While we acknowledge the limited size of the dataset, the model is certainly competitive with the current leading model for battery cycle degradation.

4.2. **Prediction.** The entire 124 battery dataset was used to evaluate the model. The data was partitioned as followed: 79 train, 25 validation, 20 test. The dataset was partitioned randomly. Each model was trained on 79 batteries, and the first 100 cycles per battery were inputted as a sequence into the model. At the end of our model search, we tested the best performing model on the 20 test batteries.

Our final model achieved a 12.5% test error, compared with Stanford's 9.1%, using the mean absolute percentage error (MAPE) for the comparison.

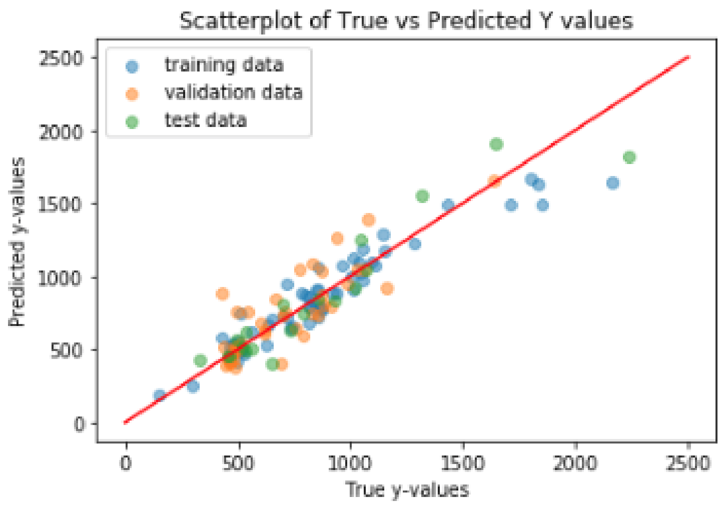

FIGURE 9. Our final model for predicting the exact cycle-life of a battery finished with a training MAPE error of 10%, a validation MAPE error of 16%, and a test MAPE error of 12.5%.

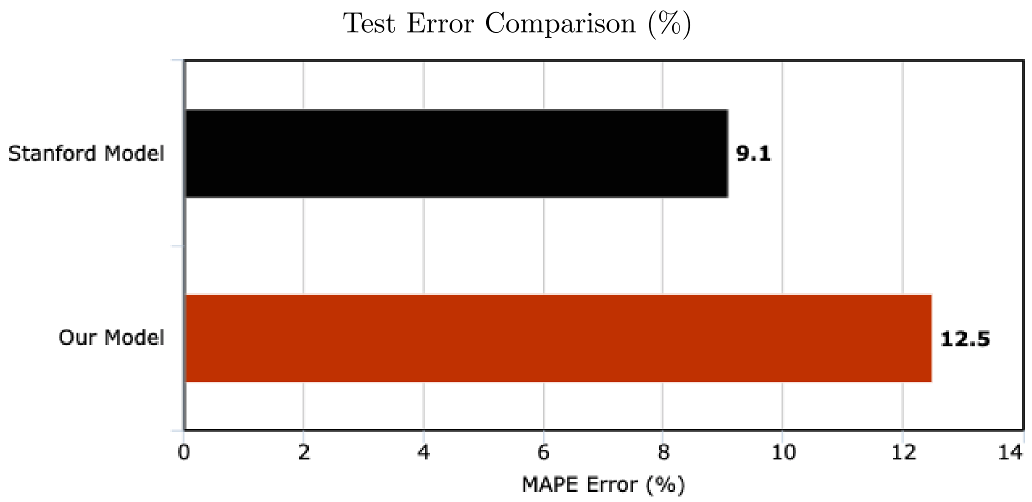

Figure 10. Our final model, which leveraged a 1-D convolutional layer plus an LSTM to process the sequential data compares to Stanfords state of the art model on a held out test set of 20 batteries. While we acknowledge the limited size of the dataset, our model is comparable to the current leading model for battery cycle-life prediction.

5. Lessons Learned

By the end of the evolutionary process, we had tried various model architectures with different combinations of hyperparameters, all in an effort to try to tackle the problem of predicting how a lithium-ion battery will last before it reaches approximately 80% of its original capacity, rendering it "dead." On this jourmey, many lessons were learned. For one, we learned that the availability of data and the complexity of our models have a vast effect on the outcomes of our task. We learned that simply passing in the most amount of data possible and hoping the model would learn was not a viable strategy; performance was greatly improved by filtering and reducing the data. Further, we learned that the most complex model doesn't neccessary preform the best—we tried many different, more complex iterations of our baseline model, and ultimately found that the simpler models actually turned out to be our best ones. Additionally, we observed the effectiveness of the concept of Attention being used outside the scope of NLP. It applied to our task, although we found that it did not help all that much compared to just using pure LSTMs as it does in NLP tasks. We postulate this to be due to the fact that the relationship between each of the cycles of the battery are more temporally linear than words in a sentence, which can refer to things far in the past, as well as be related to multiple other words.

Our biggest goal of this paper was to create a model competitive with the current state of the art accuracies, approaching the problem from a purely data-driven standpoint, hoping to use deep learning to learn the patterns in the sequences of data that the Stanford team engineered by hand. For the classification task, were able to achieve very competitive results, with validation accuracies above 90%, and a test accuracy of 95%, compared to the 97.5% test accuracy of the current leading model [Sev19]. For the prediction task, we were also able to achieve competitive results, with a test MAPE error of 12.5% as compared with a 9.1% error achieved by the current leading model [Sev19]. With that said, we are extremely pleased with our results and believe that with the lessons and ideas taken from this project, given a larger dataset and further experimentation with different techniques, we would certainly be able to improve upon our current results.

6. Tools Utilized

In the end, we opted to use the Keras Deep Learning Library to help implement our model. Keras has a very easy to work with API which provides a high-level framework for implementing neural networks and perfectly suited our needs for this task. We also used Google Colab to help us train our models in lieu of using an AWS EC2 instance because Colab provides a free GPU and has a much easier and cleaner setup process as opposed to AWS. Most visuals were created using Matplotlib.

7. Team Contributions

Michael (35%): I had meetings with Saehong Park, a grad student with extensive knowledge in batteries and battery aging to discuss the feasibility, the chemistry, the data, and the general idea of the project several times. I worked to develop an initial Pytorch implementation of the classification model (which struggled to train properly and we had trouble debugging, hence Sam switching over to Keras). I also coded, trained, and performed grid search for the entire prediction model in Keras, and attended office hours when I was stuck on how to effectively downsample our sequence of data. I also figured out how to best visualize the prediction model's accuracy, and how to compute the error in order to compare our model effectively with Stanford's. For the poster, I filled in the problem statement and edited many of the other sections, and generated one of the bar charts. For the paper, I found the template, wrote the problem statement, wrote the abstract, wrote the sections pertaining to my prediction model, and wrote the acknowledgements and reference sections.

Adel (30%): I worked on retrieving and reformatting the data used by Stanford for our purposes, which I then passed onto Sam who then performed analysis on the selection of attributes which we were to use in training our models. I worked on developing our Attention Encoder/Decoder using Keras and incorporated this into our model structure which ended up becoming our final model. I trained the Attention-based model and ran a Grid Search to tune the hyperparameters in order to give us our best result. For the poster, I found the template and designed the color scheme as well as the graphic of our final model structure and helped to proofread and edit various other sections. For the paper, I summarized our results and gave the reflection on what we learned and what could be improved upon in addition to helping proofread and edit various other sections.

Samuel (35%): In terms of data processing, I cleaned the data in terms of outlier removal and adding interpolation/zero-padding where necessary to ensure the data was a tensor. I built the module behind each dataset used to train and test the models; it allowed for the user to select which attributes they needed from each cycle, and then extracted the data from the dataset, made it a tensor, and generated a training, validation, and test set. In order to identify the attributes that provide meaningful insights for the model, I decided to build a module that computes and then visualizes the cycle-life of a battery as function of the change between cycle 100 and cycle 10 of some reduction function over all the data for some variable collected during the cycle, which was used to identify the attributes in our final datasets, as seen on page 3. In terms of the model, I wrote the baseline model in Keras, and then also coded and trained the entire classification model, using a grid search based on the original baseline model to tune it. I generated the visuals for the results of the classification model as well, as seen on the poster and in this paper. Moreover for the poster, I wrote the Data & Prepossessing section, as well the captions for the figures along with Michael. For this paper, I provided part of the background, as well as wrote and generated the visuals for Section 2 (Data), 3.1 (Classification Model), and 4.1 (Classification Results).

8. Acknowledgements

Keras Implementation of Attention: https://github.com/datalogue/keras-attention

We would like to thank the CS182 Staff for their support and guidance throughout the project. We would like to thank Saehong Park for his advice, guidance, and support throughout this project as well; his experience with batteries and battery aging were invaluable.

References

[C.B99] J.-N.Chazalviela S.Lascaud C.Brissota M.Rossoa. "Dendritic growth mechanisms in lithium/polymer cells". In: *Journal of Power Sources* 81-82 (1999), pp. 925–929.

[J.V05] M.R.Wagner-C.Veit et. al. J.Vettera P.Novka. "Ageing mechanisms in lithium-ion batteries". In: *Journal of Power Sources* 147 (2005), pp. 269–281.

[Sev19] Jin-Perkins et. al. Severson Attia. "Data-driven prediction of battery cycle life before capacity degradation". In: *Nature Energy* (2019).