A Bayesian Nonparametric Approach to Modeling Battery Health

Joshua Joseph and Finale Doshi-Velez and Nicholas Roy

***Abstract*— The batteries of many consumer products, including robots, are often both a substantial portion of the product's cost and commonly a first point of failure. Accurately predicting remaining battery life can lower costs by reducing unnecessary battery replacements. Unfortunately, battery dynamics are extremely complex, and we often lack the domain knowledge required to construct a model by hand.**

In this work, we take a data-driven approach and aim to learn a model of battery time-to-death from training data. Using a Dirichlet process prior over mixture weights, we learn an infinite mixture model for battery health. The Bayesian aspect of our model helps to avoid over-fitting while the nonparametric nature of the model allows the data to control the size of the model, preventing under-fitting. We demonstrate our model's effectiveness by making time-to-death predictions using real data from nickel-metal hydride battery packs.

I. Introduction

Batteries are often both the first point of failure and a significant fraction of a product's cost. Understanding battery failure is particularly important in robotics: at best, battery death at an inopportune time will require significant personnel intervention to rescue the robot; at worst, the robot may be lost. In robotic applications where battery death poses a large safety or financial hazard, batteries are usually replaced long before expected failure. However, this procedure introduces significant operational costs. Models to accurately predict the cycles left in a battery can help alleviate these costs.

Unfortunately, modeling battery dynamics is far from simple. Previous efforts have generally relied on a significant amount of domain knowledge [1], [2], [3]; these methods cannot be used without a detailed understanding of the internal battery structure and chemical composition. In contrast, our model can be applied to any type of battery as long as we can collect empirical measurements from a representative set of similar batteries.

We take a data-driven, non-parametric approach to predicting time to battery death. Specifically, we consider the trajectories of how voltage and temperature change as the battery cools after charging. As seen in figure 2, batteries at different points in their life-cycles exhibit different voltage and temperature trajectories while cooling. Our model first clusters together similar trajectories into *cooling behaviors*. Next, we learn a particular time-to-death for each cooling behavior. Given voltage and temperature trajectories from a new battery, we first map those trajectories to a cooling behavior learned from our training set of batteries that have already been cycled to death. We predict the new battery's remaining life based on the life that remained in the training batteries when they exhibited the same cooling behavior.

One of the key difficulties with predicting the time to battery death is that even "identical" batteries can have widely varying lifetimes. In this work, we focus on older-generation NiMH battery packs that are compatible with many robots, including the iRobot Roomba (shown in figure 1 along with its charging station). Specifically, we used a custom-built charging station to cycle 13 batteries until battery death (a process that involves months of data collection per battery). The number of charge cycles in these batteries, all from the same manufacturer, varied from 269 to 697 cycles. At the beginning of operations, all batteries behaved nearly identically and thus were difficult to differentiate. However, as each battery neared death, no matter how long it had been in operation, it tended to exhibit certain cooling behaviors after charging. A data-driven approach allows us to learn the patterns that are strongly indicative of nearing battery death.

Fig. 1. iRobot's Roomba and its charging station[1].

Finally, our nonparametric Bayesian approach allows us to make predictions given relatively little data, which is important since collecting data by cycling a fresh battery to death can take several months on a test stand. We can discover the number of cooling behaviors supported by the data automatically, without having to specify how many patterns may be present in advance. The remainder of this paper is organized as follows: we describe the battery data in section II and outline our predictive model in sections III and IV. In section V, we show how our model outpredicts several baselines on a challenging dataset of old-generation Roomba batteries.

II. Battery Data

We used data from 4683 charge cycles collected from 13 NiMH battery packs. Each pack was attached to a test stand

[1]© iRobot `store.irobot.com/product/index.jsp?productId=11305110`

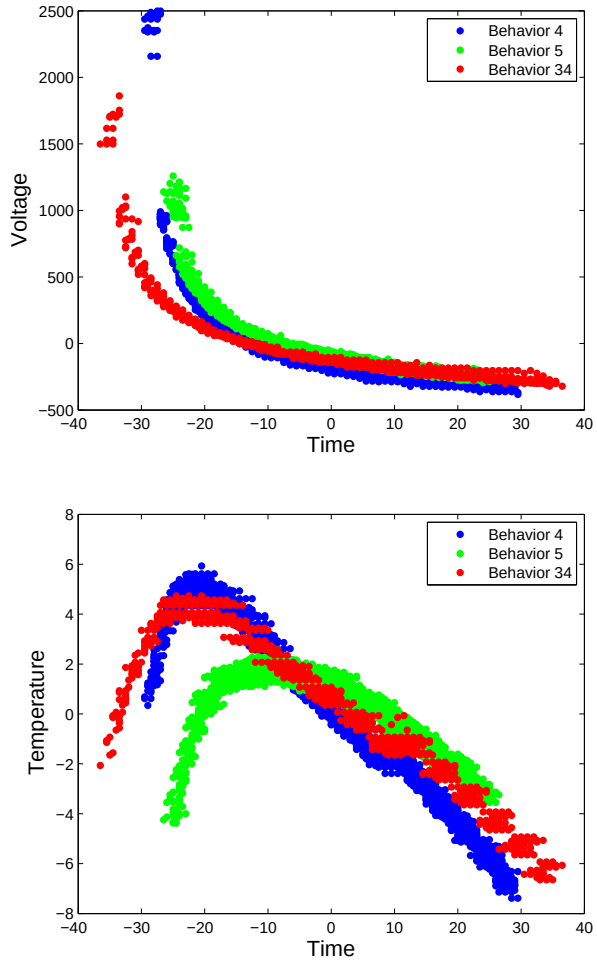

Fig. 2. Sample voltage and temperature trajectories from different parts of a battery's life-cycle showing different patterns of cooling.

which automatically cycled the battery through charging, cooling, and discharging. (Batteries heat up during charging; the charger automatically adds a cooling phase to prevent the batteries from overheating during the constant cycling.) The batteries started out new and were cycled until they could no longer hold charge, which we defined as battery death. Battery temperature and voltage were measured during each phase of each cycle.

To predict battery health, we focused on modelling the cooling phase of each cycle for two reasons. First, as seen in figure 2, batteries do exhibit a variety of distinct cooling behaviors over their lifetimes—thus, these data do contain information about battery health. Second, the cooling phase is a part of the cycle that can be most consistently measured in actual operation. Unlike on test stands, real-world discharge trajectories strongly depend on the robot's operations; charge trajectories, while somewhat more consistent, depend on the initial charge. Cooling trajectories are easily measured at the charging station and largely depend on the battery's final, fully charged state.

III. Cooling Behavior Mixture Model

To model cooling trajectories, such as those in figure 2, we assume that each voltage and temperature trajectory belongs to a particular cooling behavior. A cooling behavior is defined by a distribution over voltage and temperature trajectories. Let θ_v^j be the set of parameters that define the characteristic voltage trajectory for cooling behavior j, and let θ_e^j be the set of parameters that define the characteristic temperature trajectory for cooling behavior j.[2]

Suppose we are given a pair of voltage and temperature trajectories $v_0, v_1, ..., v_N$ and $e_0, e_1, ..., e_N$ measured at times $t_0, t_1, ..., t_N$. Our mixture model defines the probability of these data as

$$p(v_{0:N}, e_{0:N}|t_{0:N}) = \sum_{j=1}^{M} p(v_{0:N}|t_{0:N}, \theta_v^j)p(e_{0:N}|t_{0:N}, \theta_e^j)p(j) \quad (1)$$

where M is the total number of cooling behaviors. Factoring the probabilities of a voltage trajectory $p(v_{0:N}|t_{0:N}, \theta_v^j)$ and temperature trajectory $p(e_{0:N}|t_{0:N}, \theta_e^j)$ encodes our assumption that the voltage and temperature trajectories are independent given the mixture component j (which represents the battery's health state). Using a Bayesian approach, that is, placing a distribution over every parameter setting θ^j, helps avoid overfitting–a serious issue when the amount of data is limited.

Our goal is to learn the parameters θ^j for the characteristic curves and the probabilities $p(j)$ from the training data, and then learn a model of time-to-death for each behavior. (Note that only the temperature and voltage patterns for a cycle are used as input for the initial clustering, since in testing, we will not know the number of cycles remaining.) We can then use this model to predict the time-to-death by inferring the type of cooling behavior exhibited by a new battery. We describe the model we use for the cooling behaviors below; in section III-B, we describe the clustering process used to define cooling behaviors out of measured trajectories.

A. Modeling Measurement Trajectories

We model a measurement trajectory $p(v_{0:N}, e_{0:N}|t_{0:N}, \theta^j)$ as characteristic curves with additive Gaussian noise.

a) Voltage Trajectory Model: The physics of electrical charge in batteries makes an exponential curve a natural choice for the characteristic way in which we expect the voltage to change during the cooling cycle (see figure 2 for sample trajectories). We posit that the measured voltages in the trajectory will vary around this characteristic exponential curve with independent, Gaussian noise. Thus, the voltage curve associated with particular cooling behavior is characterized by the parameters of the exponential curve and the amount of measurement noise.

[2]We also considered using the number of elapsed cycles as a feature, but we found this feature actually reduced prediction accuracy by splitting clusters with different numbers of elapsed cycles but similar remaining battery life — an acute issue given the bimodal nature of battery life.

Formally, we define the probability of a voltage trajectory $p(v_{0:N}|t_{0:N}, \theta_v^j)$ given a particular characteristic curve as

$$v_i \sim \mathcal{N}\left(b_0^j + b_1^j \exp(b_2^j + b_3^j t_i), (\sigma_v^j)^2\right) \quad (2)$$

and let $\theta_v^j = \{b_0^j, b_1^j, b_2^j, b_3^j, \sigma_v^j\}$. We place an independent Gaussian prior over the values of each element of θ_v^j (posterior update equations in appendix A).

b) Temperature Trajectory Model: The temperature trajectory exhibits a more complex pattern because the battery initially continues to get warmer after the charging phase before cooling down. We evaluated first to fifth-order polynomial fits on various sample trajectories using a nested ANOVA, and a third-order polynomial was generally where the F-statistic reduced the most dramatically. Thus, we defined the probability of a temperature trajectory $p(e_{0:N}|t_{0:N}, \theta^j)$ given a particular characteristic mean curve as

$$e_i \sim \mathcal{N}\left(d_0^j + d_1^j t_i + d_2^j t_i^2 + d_3^j t_i^3, (\sigma_e^j)^2\right) \quad (3)$$

and let $\theta_e^j = \{d_0^j, d_1^j, d_2^j, d_3^j, \sigma_v^j\}$. We place an independent Gaussian prior over the values of each element of θ_e^j (posterior update equations in appendix A).

B. Clustering Measurement Trajectories into Stages

From figure 2, we see that batteries exhibit a variety of cooling behaviors—that is, a variety of characteristic voltage and temperature curves during cooling—over the battery lifetime. We use a Dirichlet process [4], [5], [6], [7] to guide the process of clustering the voltage and temperature trajectories that we measure into distinct cooling behaviors. The Dirichlet process (DP) posits that an infinite number of cooling behaviors may exist, but certain behaviors are likely to be very common and others quite rare. Of course, if only a finite number of voltage and temperature trajectories are observed, they can only belong to a finite number of cooling behaviors. We first provide background on the DP and then describe how we use it to infer the number of cooling behaviors in our battery model.

c) The Dirichlet Process Prior: Let $n_1...n_M$ be the number of trajectories assigned to behaviors $1...M$, respectively, and let z_i be the labeled behavior of trajectory i (that is, if $z_i = 5$ then the i^{th} trajectory was generated according to the 5^{th} behavior). Then, the probability that a newly observed trajectory $N+1$ belongs to behavior j, given the previous N data points, is

$$p(z_{N+1} = j, j \in 1, ..., M|n_{1:M}, \alpha) = \frac{n_j}{N+\alpha} \quad (4)$$

$$p(z_{N+1} = j, j \notin 1, ..., M|n_{1:M}, \alpha) = \frac{\alpha}{N+\alpha}. \quad (5)$$

where α is a concentration parameter that governs how often we expect to see a completely new type of cooling behavior. Having a model that can suggest that an observed trajectory is unlike any previous trajectory gives us the flexibility to automatically infer the number of behaviors in a dataset.

We place a gamma prior on $\alpha \sim \mathcal{G}(a_\alpha, b_\alpha)$ which leads us to the posterior distribution

$$p(\alpha|n_1, ..., n_M, a_\alpha, b_\alpha) \propto p(n_1, .., n_M|\alpha)p(\alpha|a_\alpha, b_\alpha). \quad (6)$$

We draw samples from this distribution using importance sampling [8].

d) Model Learning: Computing the Dirichlet Process Posterior: We now describe how to use the DP prior to infer the number of cooling behaviors in a set of trajectory data and the corresponding parameters θ of the characteristic cooling curves in each behavior. Once the number and characteristic curves for the cooling behaviors have been inferred, we can use the cooling behaviors to predict the time remaining until battery death (section IV).[3]

Exact inference over an infinite number of possible cooling behaviors is intractable. Instead, we use an iterative process to sample the relevant parameters. First, suppose that we had a set of assignments $z_1....z_N$. Given the temperature and voltage trajectories assigned to a particular cooling behavior j, we can then sample the parameters θ^j associated with the characteristic trajectories of that behavior using the equations in appendices A and A. We can also sample the concentration parameter α using equation 6. Finally, given the parameters θ^j, we can resample trajectories into more probable assignments $z_1....z_N$. Repeating this process, known as Gibbs sampling, is guaranteed to converge to a set of samples that represent the posterior $p(\{z_i\}, \{\theta^j\}, \alpha|v_0, v_1, ..., v_N, e_0, e_1, ..., e_N)$.

Algorithm 1 summarizes our approach, which follows [10], [11]. The Gibbs sampling procedure begins by resampling each z_i given the remaining parameters $\{\theta^j\}, \alpha\}$. The probability that voltage and temperature trajectories i will be assigned to an instantiated cooling pattern j is

$$p(z_i = j|\theta^j, \alpha) \propto p(v_{0:N}|t_{0:N}, \theta_v^j)p(e_{0:N}|t_{0:N}, \theta_e^j)\left(\frac{n_j}{N-1+\alpha}\right) \quad (7)$$

where $p(v_{0:N}|t_{0:N}, \theta_v^j)$ and $p(e_{0:N}|t_{0:N}, \theta_e^j)$ are defined in equations 2 and 3, respectively. The probability that the trajectories belong to a new cooling behavior is given by

$$p(z_i = M+1|\alpha) \propto \left[\int p(v_{0:N}|t_{0:N}, \theta)p(e_{0:N}|t_{0:N}, \theta)d\theta\right]\left(\frac{\alpha}{N-1+\alpha}\right), \quad (8)$$

and we use Monte Carlo integration [8] to approximate the integral as suggested in [10].

Given a set of assignments, the trajectory parameters θ^j are resampled from their posterior distributions (see appendices A and A). In practice we found that simply choosing the maximum *a posteriori* θ^j worked well due to the peakiness of the posterior. Finally, the concentration parameter α is resampled according to equation 6 from the previous section.

Figure 3 shows an example of the cooling behaviors in a single battery that are discovered by the inference process. The fact that the blocks of color (denoting cooling cycles that are inferred to be from the same behavior) are close

[3]The inference approach described here is taken from our previous work [9].

Algorithm 1 Cooling Behavior Inference

for sweep = 1 to # of sweeps **do**
  for each cooling behavior j **do**
    Draw the parameters θ^j given z_i using the equations in appendices A and A.
  end for
  Draw the DP hyperparameter α using equation 6
  for each trajectory i **do**
    Draw z_i using equations 7 and 8
  end for
end for

together in time suggests that clustering by trajectory type does indeed find groups of trajectories with similar times to battery death. We also see that at the beginning of the battery's lifetime (cycles 0 to 400), the distribution over cooling behaviors is quite spread and not strongly associated with a particular time to death. As the battery progresses through its life-cycle, each behavior marks a temporally-clustered block preceding battery death. In section V, we will see that the relative indistinguishability of cooling trajectories far from battery death means that predicting the lifetime of a fresh battery is prone to large errors; however, the more distinct characteristics of cooling trajectories closer to death allows for better predictions when the battery is nearing death.

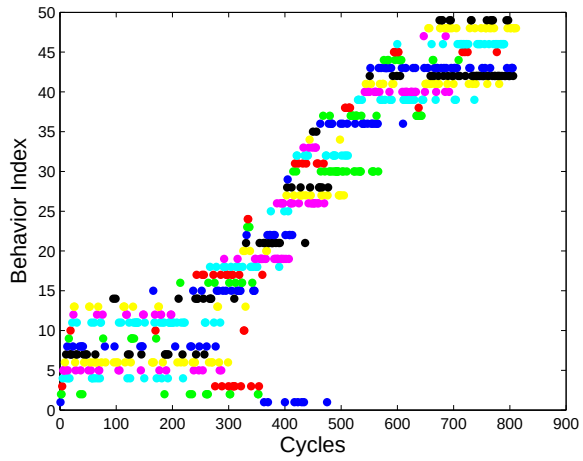

Fig. 3. Cooling behaviors (indexed both on the y-axis and by color) for a single battery as it progresses through its life-cycle. (Cycles from the same behavior are plotted with the same color.) Initially, the cooling behaviors occur at a variety of points (see the behaviors for cycles 0 to 400), but as the battery progresses through its life-cycle, each stage more distinctly marks a block of time preceding battery death.

IV. Cycles-to-Death Prediction

Once the parameters $\{\theta^j\}_{j=1}^M$ have been learned, we compute a distribution $p(l|j,\theta^j)$ of the remaining cycles to death l for each cooling behavior j. Especially for cooling behaviors that occur early in a battery's lifetime, these distributions can be highly multimodal: a fresh long-life battery and a fresh short-life battery might have very similar voltage and temperature trajectories during their initial cooling phases, but they will have very different numbers of cycles remaining until death (that is, when the battery stops holding charge).

We use an empirical distribution to model the number of remaining cycles l. This distribution places a probability mass around each observed time-to-death l_i for each cooling behavior j in the training set.

$$p(l|j,\theta^j) \propto \sum_{i=1}^{N} \mathcal{N}\left(l; l_i, \sigma_l^2\right) \mathbf{1}\{z_i = j\} \tag{9}$$

where $\mathbf{1}\{z_i = j\}$ is the indicator function which equals one when trajectory i is assigned to behavior j and zero otherwise.

We can now make predictions about the remaining cycles left in a battery given voltage and temperature trajectories. The distribution over the cycles-to-death is given by taking the expectation over the cooling behavior j

$$\begin{aligned} &p(l|t_{0:N}, v_{0:N}, e_{0:N}, \{\theta^j\}, \alpha) \\ &= \sum_{j=1}^{M+1} p(l|j,\theta^j) p(j|t_{0:N}, v_{0:N}, e_{0:N}, \theta^j, \alpha) \end{aligned} \tag{10}$$

where $p(l|j,\theta^j)$ and $p(j|t_{0:N}, v_{0:N}, e_{0:N}, \theta^j, \alpha)$ are defined in sections IV and III.[4] Throughout a test run, we maintain a distribution over l, and after we observe a cycle of data we shift the distribution by -1 to simulate the amount of life degradation that occurred during the previous cycle.

V. Results

We tested our battery health model on a set of 13 Roomba batteries from iRobot. Of these, 6 had relatively short lives (under 400 cycles), 5 had long lives (over 600 cycles), and 2 were in-between. [5]The errors in the predicted cycles-to-death were computed using leave-one-out cross-validation, where the trajectories for one entire battery's run were held out each time. We chose to hold out an entire battery—rather than a random subset of cycles—because trajectories coming from the same battery tend to be more similar to each other than trajectories coming from different batteries; having trajectories from the same battery in both test and training sets would provide an unrealistic advantage to any algorithm. We normalized the time, voltage, and temperature values of all the cycles by subtracting the mean to avoid numerical problems during inference. Additionally, we set $\sigma_l = 25$ from equation 9.

We compared our approach to four simple baselines. The naive approach estimated the remaining battery life by simply subtracting the number of elapsed cycles from the mean lifetime of the batteries in the training set. For the three remaining baselines, we first fit a curve to each temperature and each voltage trajectory using the parametric

[4]We also considered a Gaussian distribution for $p(l|j,\theta^j)$ but, due to the multimodal nature of $p(l|j,\theta^j)$, we found the empirical distribution a much better fit to the data.

[5]Note that iRobot Roombas ship with a longer-lasting battery, whereas the packs we tested were old-generation batteries provided by iRobot for research purposes.

curves described in section III-A. This fit gave us a feature vector of trajectory parameters $x = [\vec{b}\vec{d}]$ for each cycle that we could use as input to predict the remaining life l. We tested three baseline predictors: a linear regression of the form $l \sim \beta x$; k-nearest neighbor, which found the k nearest inputs x and returned their average time to death l; and k-means, which first formed k clusters based on the inputs x and then returned the mean l of the nearest cluster. For k-means and k-nearest neighbors, we tested values of k ranging from 1 to 300 (note that the entire dataset consisted of 4683 cooling trajectories). Each run of k-means had 10 restarts to avoid local optima.

e) Absolute Prediction Error: We first considered the absolute prediction error as a measure of prediction quality—after all, an ideal model would provide the user of the device an accurate picture of how many cycles are remaining in the battery's life. Figures 4 and 5 show the predictions of all the approaches on a typical short life and a typical long life battery, respectively. We chose the best-performing value of k for each plot, and the error bars show one standard error of the mean for blocks of 50 trajectories.

Our approach generally outperforms the baselines when predicting the remaining life in the short life battery. While we initially do poorly in predicting the remaining life in the long-life battery, we outperform the baselines as the long-life battery gets closer to failure. Thus, for both types of batteries, we predict the cycles to death well in regimes that matter most: when the battery is actually near death. From manual inspection, we note that the cooling behaviors near battery death tend to be more distinctive; we also have more data in this regime because every battery—long life or short life—has trajectories that are within 300 cycles of death. Only the long-life batteries have trajectories that are over 600 cycles out from death.

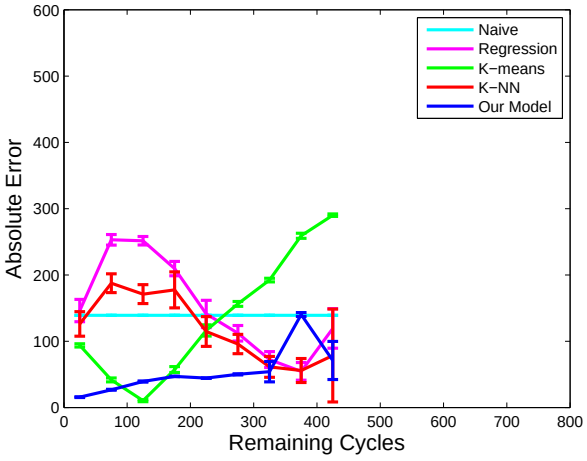

Fig. 4. Mean absolute prediction errors for a typical short-life battery.

We see this trend when we consider all of the batteries from our complete cross-validation test. As before, the best performing values for k are plotted for the k-means and k-nearest neighbor. We see in figure 6 that our approach outperforms the baselines when the actual remaining cycles

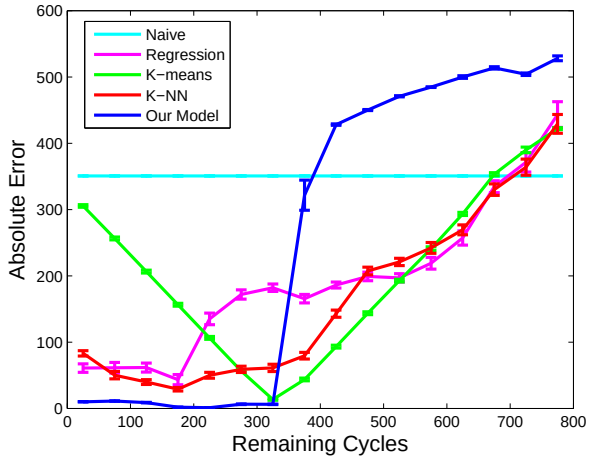

Fig. 5. Mean absolute prediction errors for a typical long-life battery.

is relatively small (roughly within the lifetime of a short-life battery). All approaches have difficulty with predictions when the battery is farther away from failure (figure 7).

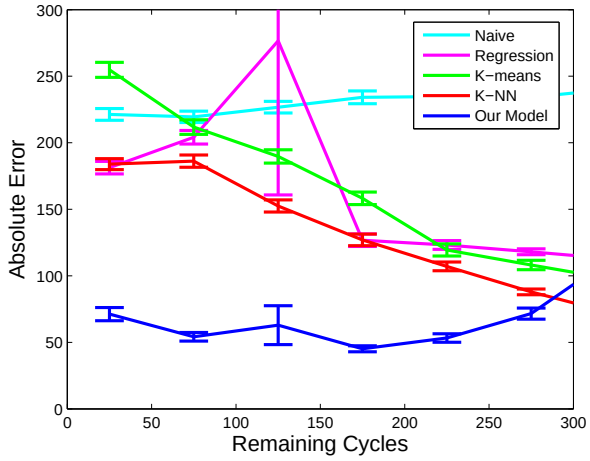

Fig. 6. Mean absolute prediction errors across all batteries, looking at batteries close to death.

f) Guiding Replacement Decisions: While knowing the number of cycles left in a battery is a useful figure, remaining life is usually an intermediate quantity needed to make the key decision of whether the battery should be replaced. When the lifetime predictions have error, the user of the device risks either waiting too long to replace the battery—and having it fail unexpectedly—or replacing the battery too early due to a false alarm. In this section, we analyze the predictions to determine how our model and the various baselines performed in this trade-off.

Figures 8 and 9 show how each of the models handle these trade-offs. Figure 8 plots how often the models predicted that a battery had less than 50 cycles remaining (an arbitrary threshold for replacement) against the actual remaining battery life; peaks far from 50 correspond to false alarms where the battery might have been replaced even when it had significant life left. Figure 9 plots how often the

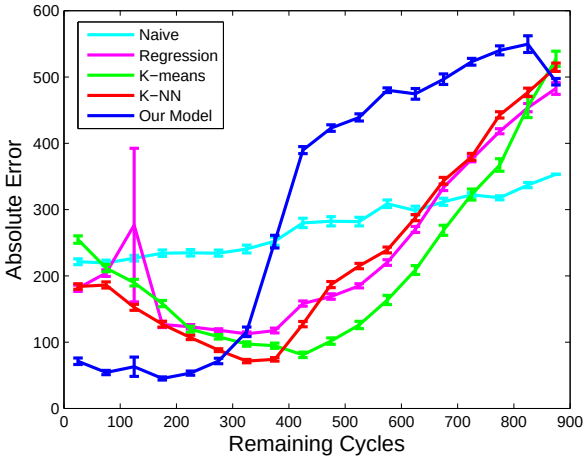

Fig. 7. Mean absolute prediction errors across all batteries. Note that the model had many fewer training points in regimes more than 500-600 cycles to failure, since many of the batteries were short-life batteries.

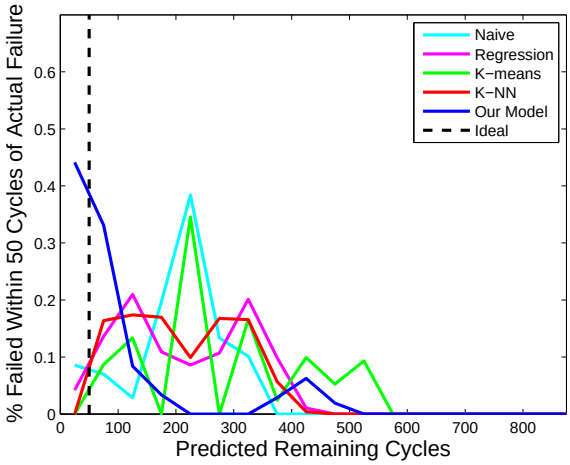

Fig. 9. Risk of late replacement: we plot how often the battery was actually within 50 cycles of failure for various predicted lifetimes. (An ideal predictor would be 0 if the predicted cycles were greater than 50, 1 otherwise.)

battery actually had less than 50 cycles remaining against the predicted battery life; peaks far from 50 correspond to situations when a battery might have failed despite a long predicted remaining life. Our model has peaks near 50 in both plots; our main source of error is confusing a short-life battery for a long-life battery or vica-versa (as seen in humps around 350-400). The other models seem to find less structure in the data; their errors are more spread out. (Note that k-nearest neighbor never predicts less than 50 cycles remaining.)

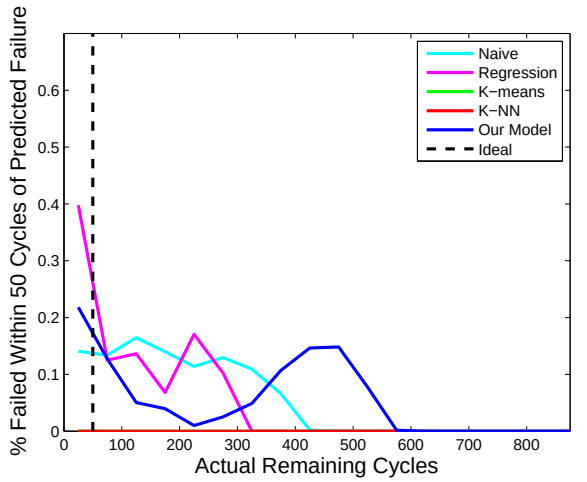

Fig. 8. Risk of early replacement: we plot how often the models predicted that the battery had less than 50 cycles remaining for various actual number of cycles remaining. (An ideal predictor would be 0 if the actual cycles were greater than 50, 1 otherwise.)

In the previous plots we created an arbitrary threshold of 50 cycles as when a battery ought to be replaced. We vary the threshold w to illustrate the trade-off more generally. Figure 10 plots false positive rates—how often did we predict that battery had more than w cycles when it had less than w cycles—against true positive rates—how often did we predict that the battery had less than w cycles when it actually had less than w cycles left. The ideal predictor would have points close to the top-left corner, corresponding to high true positive and low false positive rates; our approach achieves relatively high true positive rates without high false positive rates for a variety of thresholds w. However, we also see that our model is somewhat conservative: if avoiding false positives (replacing too early) is more important than finding true positives (replacing before death), then the baselines might offer better options. Operationally, false negatives in predicting battery death are typically dangerous for robots but the desired performance is usually domain dependent.

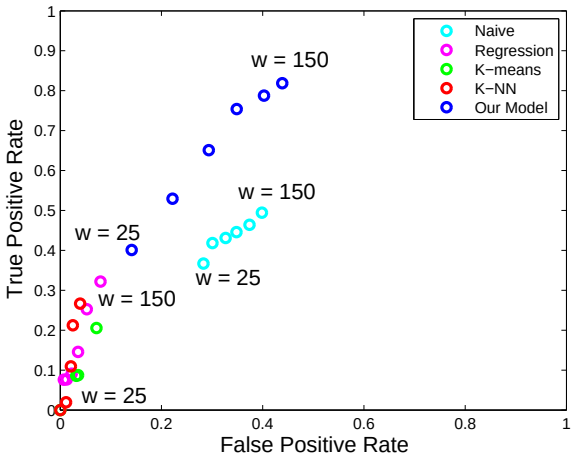

Fig. 10. False positive rate (how often the battery was replaced too early) vs. True positive rate (how often the battery was replaced in time) for various decision thresholds w.

VI. Conclusion

We developed a Bayesian nonparametric model for battery health using data from 4683 cooling cycles collected from 13 NiMH batteries. The small size of the dataset reflects the expense required to collect the data: cycling a battery to death can take months, and different types of batteries will have

different cooling patterns. Using a Bayesian nonparametric approach was valuable in this data-limited setting because it allowed us to learn a model from relatively few examples (of course, new data would be needed to train a model for a different type of battery). Specifically, by automatically clustering cycles based on their voltage and temperature trajectories during cooling, we discovered cooling behaviors that were predictive of remaining battery life, especially in the important regime where the battery within a few hundred cycles of death.

The results also showed that while we outperformed several baseline models, especially in this regime of interest, predicting battery life is, overall, a challenging problem. Interesting directions for future work include using other (less consistent) parts of the battery cycle, such as voltage and temperature trajectories during charging as additional features, as well as building stronger temporal correlations into the model. In some applications, knowing when a battery is starting to degrade (around cycle 300 for the battery in figure 3), is also a quantity of interest. We found that this change is much harder to predict using our trajectory-based clusters because trajectories tend to look similar in nominal operation; predicting battery degradation is an interesting question for future work.

ACKNOWLEDGMENTS

This work was supported by iRobot and the Army Research Office under the Nostra project, STTR W911NF-08-C-0066. The authors would additionally like to thank Michael Rosenstein at iRobot for his advice and guidance throughout this work.

REFERENCES

[1] M. Chen and G. A. Rincon-Mora, "Accurate electrical battery model capable of predicting runtime and iv performance," *IEEE Transactions on Energy Conversion*, vol. 21, no. 2, pp. 504–511, 2006.

[2] P. Gomadam, "Mathematical modeling of lithium-ion and nickel battery systems," *Journal of Power Sources*, vol. 110, no. 2, pp. 267–284, 2002.

[3] R. A. Dougal, "Dynamic lithium-ion battery model for system simulation," *IEEE Transactions on Components and Packaging Technologies*, vol. 25, no. 3, pp. 495–505, 2002.

[4] Y. W. Teh, "Dirichlet Processes," 2007, submitted to Encyclopedia of Machine Learning.

[5] M. I. Jordan, "Dirichlet Processes, Chinese Restaurant Processes and all that," 2005.

[6] J. Sethuraman, "A Constructive Definition of Dirichlet Priors," *Statistica Sinica*, vol. 4, pp. 639–650, 1994.

[7] C. E. Antoniak, "Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems." *The Annals of Statistics*, vol. 2, pp. 1152–1174, November 1974.

[8] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, August 2006.

[9] J. Joseph, F. Doshi-Velez, A. S. Huang, and N. Roy, "A bayesian nonparametric approach to modeling motion patterns," *Autonomous Robots*, 2011, search and Pursuit/Evasion with Mobile Robots.

[10] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Advances in Neural Information Processing Systems 14*, 2002.

[11] C. E. Rasmussen, "The Infinite Gaussian Mixture Model," in *Advances in information processing systems 12*, S. e. a. Solla, Ed. MIT Press, 2000, pp. 554–560.

APPENDIX

A. Voltage Parameter Posterior Distribution

We define the prior over the voltage trajectory model parameters to be

$$\begin{aligned} s_v^j|a_v, b_v &\sim \mathcal{G}(a_v, b_v) \\ b_k^j|\mu_{v,k}, s_{v,k} &\sim \mathcal{N}(\mu_{v,k}, s_{v,k}) \end{aligned}$$

where $s_v = 1/\sigma_v^2$ is the precision of the voltage trajectory model and $k = 0, 1, 2, 3$. Based on the likelihood function defined in equation 2 the posterior distributions of parameters s_v and b_0 have closed form posteriors

$$s_v^j|v_{0:N}, t_{0:N}, b_{0:3}^j, a_v, b_v \sim \mathcal{G}\left(a_v + \frac{N+1}{2}, \frac{1}{\frac{1}{2}\sum_{i=0}^{N}(v_i - b_2^j t_i^2 - b_1^j t_i - b_0^j)^2 + \frac{1}{b_v}}\right)$$

$$b_0^j|e_{0:N}, t_{0:N}, b_1^j, b_2^j, s_v^j, \mu_{v,0}, s_{v,0} \sim \mathcal{N}\left(\frac{s_{v,0}\mu_{v,0} + s_v\sum_{i=0}^{N} v_i - b_2 t_i^2 - b_1 t_i}{(N+1)s_v + s_{v,0}}, \frac{1}{(N+1)s_v + s_{v,0}}\right)$$

that are normally distributed. Unfortunately, the posterior distributions for parameters $b_{1:3}$ do not have a closed form and therefore we used importance sampling [8] to sample from them.

Temperature Parameter Posterior Distribution

We define the prior over the temperature trajectory model parameters to be

$$\begin{aligned} s_e^j|a_e, b_e &\sim \mathcal{G}(a_e, b_e) \\ b_k^j|\mu_{e,k}, s_{e,k} &\sim \mathcal{N}(\mu_{e,k}, s_{e,k}) \end{aligned}$$

where $s_e = 1/\sigma_e^2$ is the precision of the temperature trajectory model and $k = 0, 1, 2, 3$. Based on the likelihood function defined in equation 3 the posterior distributions of

the parameters have closed form posteriors

$$s_e^j | e_{0:N}, t_{0:N}, d_{0:3}^j, a_e, b_e \sim$$
$$\mathcal{G}\left(a_e + \frac{N+1}{2}, \frac{1}{\frac{1}{2}\sum_{i=0}^{N}(e_i - d_2^j t_i^2 - d_1^j t_i - d_0^j)^2 + \frac{1}{b_e}}\right)$$
$$d_0^j | e_{0:N}, t_{0:N}, d_{1:3}^j, s_e^j, \mu_{e,0}, s_{e,0} \sim$$
$$\mathcal{N}\left(\frac{s_{e,0}\mu_{e,0} + s_e\sum_{i=0}^{N} e_i - d_3 t_i^3 - d_2 t_i^2 - d_1 t_i}{(N+1)s_e + s_{e,0}}, \right.$$
$$\left. \frac{1}{(N+1)s_e + s_{e,0}}\right)$$
$$d_1^j | e_{0:N}, t_{0:N}, d_0^j, d_2^j, d_3^j, s_e^j, \mu_{e,1}, s_{e,1} \sim$$
$$\mathcal{N}\left(\frac{s_{e,1}\mu_{e,1} + s_e\sum_{i=0}^{N} e_i t_i - d_3 t_i^4 - d_2 t_i^3 - d_0 t_i}{s_e\sum_{i=0}^{N} t_i^2 + s_{e,1}}, \right.$$
$$\left. \frac{1}{s_e\sum_{i=0}^{N} t_i^2 + s_{e,1}}\right)$$
$$d_2^j | e_{0:N}, t_{0:N}, d_0^j, d_1^j, d_3^j, s_e^j, \mu_{e,2}, s_{e,2} \sim$$
$$\mathcal{N}\left(\frac{s_{e,2}\mu_{e,2} + s_e\sum_{i=0}^{N} e_i t_i^2 - d_3 t_i^5 - d_1 t_i^3 - d_0 t_i^2}{s_e\sum_{i=0}^{N} t_i^4 + s_{e,2}}, \right.$$
$$\left. \frac{1}{s_e\sum_{i=0}^{N} t_i^4 + s_{e,2}}\right)$$
$$d_3^j | e_{0:N}, t_{0:N}, d_0^j, d_1^j, d_2^j, s_e^j, \mu_{e,3}, s_{e,3} \sim$$
$$\mathcal{N}\left(\frac{s_{e,2}\mu_{e,2} + s_e\sum_{i=0}^{N} e_i t_i^3 - d_2 t_i^5 - d_1 t_i^4 - d_0 t_i^3}{s_e\sum_{i=0}^{N} t_i^6 + s_{e,3}}, \right.$$
$$\left. \frac{1}{s_e\sum_{i=0}^{N} t_i^6 + s_{e,3}}\right)$$

that are normally distributed.