Distributed Active Learning with Application to Battery Health Management

Huimin Chen, University of New Orleans
X. Rong Li, University of New Orleans

Available at: https://works.bepress.com/huimin_chen/16/
Distributed Active Learning
with Application to Battery Health Management

Huimin Chen
Department of Electrical Engineering
University of New Orleans
New Orleans, LA 70148, U.S.A.
Email: hchen2@uno.edu

X. Rong Li
Department of Electrical Engineering
University of New Orleans
New Orleans, LA 70148, U.S.A.
Email: xli@uno.edu

Abstract—This paper focuses on distributed implementation of active learning with a limited number of queries. In the prognostics and health management domain, the cost to obtain a training sample can be fairly high, especially when studying the aging process for remaining useful life prediction of a mission critical component. Active learning with limited resource is formulated as a reinforcement learning problem, where the sampling strategy has to minimize the expected generalization error within a finite horizon. An importance sampling based method is adopted for active learning and extended to distributed implementation with multiple active learners. The fusion of importance weights from multiple learners is interpreted as a special boosting strategy. Empirical study shows that the fusion of distributed active learners achieves better classification and regression problems as well as semi-supervised learning. The remaining useful life prediction for battery health management is used to compare the proposed method with conventional passive learning methods. Empirical study shows that the fusion of distributed active learners achieves better classification and prediction accuracy with a reduced number of training samples needed to have a complete run-to-failure profile.

Keywords: Active learning, reinforcement learning, remaining useful life prediction, battery health management.

I. INTRODUCTION

Active learning, as a special form of supervised learning, aims at accurately approximating a target concept with carefully chosen examples. For classification, an active learner judiciously chooses the training samples to ask for their labels in the hope that it can achieve, with a significantly smaller number of queries, the same classification accuracy as that of a passive learner. Most of the existing active learning methods rely on properties of the hypothesis space, e.g., VC dimension or covering numbers, and apply various criteria to estimate the additional information provided by a new example [7], [8]. The active learner iteratively selects a good example to learn in the sense of the criterion being considered.

A. Background of Active Learning

Let \( S_t = \{(x_i, y_i), i = 1, \ldots, t\} \) be a collection of \( t \) training samples, where \( x_i \in X \) is the \( i \)-th input in the instance space \( X \) and \( y_i \in Y \) is the corresponding output in the label space \( Y \). To ease the discussion, we first consider binary classification case, where \( Y = \{-1, 1\} \). From \( S_t \), a learning algorithm provides some hypothesis \( h \) in a hypothesis space \( \mathcal{H} \) that specifies the mapping \( X \to Y \). The expected loss of \( h \) is compared with the target concept \( h^* \) and computed over the joint distribution \( P_{XY} \) of \((x_i, y_i)\) on the problem domain. In supervised learning, one often assumes that training samples \((x_i, y_i)\) are independently drawn from \( P_{XY} \). A passive learner usually tries to find the best hypothesis using all training samples. Alternatively, an active learner selects some instance \( x_i \) at time step \( i \) from a pool of training samples drawn from \( X \) and inquires its label \( y_i \). An active learning strategy, also referred to as a policy, is a mapping from \((X \times Y)^t\) to \( X \) to select a new instance \( x_{t+1} \in X \) based on \( S_t \). To generalize, an active learning algorithm maps \((X \times Y)^t\) to \( \mathcal{H} \) associating a hypothesis \( h \) to a given training set \( S_t \). The version space \( \mathcal{H}(S_t) \) is the set of hypotheses consistent with \( S_t \), i.e., \( h(x_i) = y_i \) for \( i = 1, \ldots, t \). Let \( T \) be the total number of queries that an active learner will make. The state space \( \mathcal{S} \) may contain all possible training samples. The set of actions \( \mathcal{A} \) depends on \( S_t \) through \( h \). Let \( a_t(S_{t-1}, h) \) be the action at time \( t \) and \( a_{1:T} \) the action sequence up to horizon \( T \). The penalty incurred by \( a_{1:T} \) is denoted by \( e(a_{1:T}, S_T(h), h) \), which can be the generalization error of the hypothesis \( h \) learned from \( S_T \). It is natural to define the optimal policy as the one minimizing the expected generalization error at horizon \( T \)

\[
a_{1:T}^* = \arg \min_{a_{1:T}} E_{h \in \mathcal{H}} e(a_{1:T}, S_T(h), h). \tag{1}
\]

However, the penalty is only known at the terminal time \( T \), which limits the efficiency of search in the hypothesis space. A reasonable modification is to limit the target hypothesis \( h \) within the version space \( \mathcal{H}(S_T) \), leading to the solution

\[
a_{1:T}^* = \arg \min_{h \in \mathcal{H}(S_T)} e(a_{1:T}, S_T(h), h). \tag{2}
\]

If we further assume that the queries taken sequentially are based on minimizing the expected generalization error conditioned on \( S_t \), then the problem becomes a Markov decision process where any policy needs to specify the probability transition matrix \( P(a_t|S_{t-1}) \) [16]. It means that for any sample \((x, y)\), one has to compute \( P(y = h(x)|h \in \mathcal{H}(S_t)) \) for \( t = 1, \ldots, T \). Since the expected generalization error is hard to optimize, one often adopts the empirical error from surrogate hypotheses sampled uniformly in the version space to guide the search for the optimal policy.
B. Importance Weighted Estimator

Since finding the optimal policy by solving (2) directly can be computationally intractable for a large hypothesis space, we relax this optimality requirement and instead we identify a good active learning strategy that achieves the performance comparable to that of a passive learner with high probability as the sample size grows. With this setting, an active learner can examine unlabeled data $x_1, x_2, ..., x_T$ sequentially and assign a query probability $P_i$ to $x_i$ to determine whether one wants to inquire its label $y_i$ or not. In experimental design, one faces the variable selection problem in a similar situation where the selected variables to be queried should result in low variance of certain regression function [13]. Selective sampling in active learning was first exploited by [1] using boosting and bagging. A Bayesian strategy for adaptive sampling was proposed in [18] to reduce the prediction variance. However, these methods may introduce sampling bias that leads to convergence problem even with infinite many queries. We consider an importance sampling strategy proposed in [4] with guaranteed unbiasedness. Let $Q_i \in \{0, 1\}$ be the indicator on whether $y_i$ is queried. The probability $P_i$ can depend on $Q_{1, ..., Q_{i-1}}$, but not the actual label of the queried sample. Assume that samples $S_t$ are independently drawn based on $P_{XY}$. Then for some function $f : X \times Y \rightarrow \mathcal{R}$, we have the empirical estimate of $E[f(x, y)]$ given by

$$\hat{f}(S_t) = \frac{1}{T} \sum_{i=1}^{T} \frac{Q_i}{P_i} f(x_i, y_i).$$

Note that the estimate depends on $y_i$ only when $Q_i = 1$. One possible estimate of the classification error rate is given by

$$\hat{e}(S_T(h), h) = \frac{1}{T} \sum_{i=1}^{T} \frac{1}{P_i} 1(h(x_i) \neq y_i).$$

This importance weighted estimate is unbiased for fairly general choices of $\{P_i\}$ [4].

C. Prognostics and Battery Health Management

Prognostics concerns the prediction of the health condition of a system or system component using the knowledge of historical usage and current state. Existing prognostic methods can be largely categorized as physics-based and data-driven and they all fall into the passive learning regime in the sense that learning examples are chosen without any feedback from the prognostic output. We hope to transform those passive learning methods into the corresponding active learning ones with a significant saving in the number of queries to the output label. The benefit can often be quantified when comparing data-driven prognostic methods using real data sets. However, real world prognostic problems do not have adequate training and testing samples either due to proprietary restriction from public usage or simply the high cost to collect them. Battery health monitoring is an important prognostic application where Li-ion battery data sets have been used for prognostic algorithm development and evaluation [10]. It is less costly to perform run-to-failure test on batteries than on many other industrial system components such as milling machines and turbofan engines. Thus we investigate the applicability of active learning in battery health management in the hope of extending the approach to other prognostic applications where the data collection is more expensive.

In battery health management, it is crucial to ensure that a battery will provide the desired power during the current discharge cycle. Useful battery health indicators include state-of-charge, state-of-life and state-of-health. Battery health monitoring methods include offline analysis and online prediction with a physics-based aging model or regression from the online observed parameters. Offline analysis aims at developing the failure mechanism model characterized by diagnostic features of the battery. For Li-ion cells, it is noted that charge transfer resistance and electrolyte resistance vary depending on the battery capacity indicating an aging process [15]. Online prediction provides the remaining useful life (RUL) estimate based on the model obtained from offline analysis, the current state and future usage of the battery. The offline analysis can be expensive and time consuming since a battery has to be repeatedly charged and discharged with a possibly expedited aging profile in order to create a complete run-to-failure training sample. On the other hand, the online prediction of RUL for another battery may have a different aging curve from the one obtained in offline analysis due to the difference in the usage profile or internal structure. Active learning method takes the cost of generating a training sample from offline analysis into account. It does not require all batteries to get through run-to-failure tests in order to learn the target concept for RUL prediction. We focus on the active learning algorithm implemented on each battery while taking the health status learned from other batteries into account for predicting the discharge time and number of discharge cycles before the battery capacity reduces to the critical limit.

II. DISTRIBUTED ACTIVE LEARNING

Consider $K$ active learners, each applying an importance sampling based algorithm on the unlabeled data. If a sample $x_i$ has been queried by one of the learners, then other learners can inquire its label without incurring additional cost. On the other hand, $x_i$ will have an importance weight (associated with the sampling threshold) to be updated by any active learner who decides not to inquire its label. The importance weight on each unqueried sample will affect the sampling threshold of the learner who first encounters this sample. The goal is to achieve a convergence rate of the expected generalization error no worse than that of a centralized active learner using the same version space. Nevertheless, different active learners can have their own version spaces and thus making different decisions on the testing samples. It is expected that the overall performance made by $K$ active learners is better than any single active learner if a fusion center can combine the learned hypotheses by properly exploiting the difference among these learners. In addition, the number of samples being queried should be less than the total number of training samples.
A. Importance Sampling Based Active Learning

We assume that \( \mathcal{H} \) is finite for an active learner and the sampling probability sequence \( \{P_i\} \) satisfies \( P_{\min} = \min_i \{P_i\} > 0 \). Define a general loss function \( l(h(x), y) \) as the metric to assess a learning algorithm that uses \( x \) to predict \( y \) under hypothesis \( h \). Denote the expected loss under hypothesis \( h \) as

\[
L(h) = E_{XY}[l(h(x), y)]
\]

and the importance weighted estimate of \( L(h) \) with \( T \) samples parsed through the active learner as

\[
L_T(h) = \frac{1}{T} \sum_{i=1}^{T} \frac{Q_i}{P_i} l(h(x_i), y_i).
\]

In the worst case, the probability that the estimated loss deviates from the expected loss greater than the sample complexity bound is within

\[
P\left( \max_{h \in \mathcal{H}} |L_T(h) - L(h)| > \frac{1}{p_{\min}} \sqrt{\frac{2 \log |\mathcal{H}|/\delta}{T}} \right) < \delta \tag{7}
\]

for any \( \delta > 0 \) [7]. Note that we used \( \log |\mathcal{H}| \) rather than the VC dimension of \( \mathcal{H} \) for simplicity. Clearly, the sample complexity increases on the order of \( 1/p_{\min}^2 \) compared with the same setting for a passive learning algorithm [11]. Thus there is a design tradeoff between the expected number of samples to be queried and the sharpness of the generalization error bound. Let \( L_T^* = \min_h L_T(h) \). Intuitively, we only need to focus on the version space within some range \( \Delta_T \) related to the sample complexity bound, e.g., scaling in \( 1/\sqrt{T} \). This suggests that we can shrink the version space to

\[
\mathcal{H}(S_T) = \{h|h \in \mathcal{H}(S_{T-1}), L_T(h) < L_T^* + \Delta_T\} \tag{8}
\]

with \( \mathcal{H}(S_0) = \mathcal{H} \). In addition, the sampling probability \( P_T \) can be related to the maximum disagreement among the hypotheses in the version space [11]. For example, one can set

\[
P_T = \max\{P_{\min}, P^*\} \tag{9}
\]

where

\[
P^* = \max_{h_1, h_2 \in \mathcal{H}(S_{T-1})} \max_y l(h_1(x_T), y) - l(h_2(x_T), y) \tag{10}
\]

assuming that the loss function \( l(x, y) \) has been normalized to the range \([0, 1]\).

B. Distributed Implementation

Assume that \( K \) active learners works on hypothesis spaces \( \{\mathcal{H}^j\}_{j=1}^K \). Each active learner takes its own sequence of training samples and marks an input \( x_j \) as either queried on its label \( y_j \), or not queried. A training sample that has not been queried by any active learner will have its sampling probability \( P_i \) initially set to \( P_{\min} \). If a sample has been marked as queried by an active learner, then other active learners will also get its label to update their learning algorithms. When an active learner encounters an unqueried sample, it will change the query probability associated with this sample based on the estimated maximum excess from the expected loss. The algorithm for each active learner runs as follows.

1) Initialization: Active learner \( j \) works on weighted sample sequence \( \{x_t, y_t, P_t\}_{t=1}^T \) and hypothesis space \( \mathcal{H}^j(S_0) = \mathcal{H}^j \).

2) Update: If \( P_t < 1 \), then calculate sampling probability \( P_t^j \) using (9) and update \( P_t^j \) with

\[
P_t^{\text{new}} = \sqrt{P_t^j P_t} \tag{11}
\]

Flip a biased coin with \( P_t^{\text{new}} \) to obtain \( Q_t^j \). If \( Q_t^j = 1 \), then inquire the label \( y_t \) and set \( P_t = 1 \). Otherwise, set \( P_t = P_t^{\text{new}} \). Shrink the version space using (8).

3) Output: The best hypothesis \( h^{\text{test}} \) that yields \( L_T^* \).

When \( K = 1 \), the algorithm reduces to importance sampling based active learning. When \( \mathcal{H}^1 = \cdots = \mathcal{H}^K \), the sampling probability \( P_t \) will be the same for all learners when they apply the same hypothesis reduction (8). However, the sampling probability of \( x_t \) by one of the active learners in a fully distributed update is \( 1 - (1 - P_t) K \), which is larger than the sampling probability \( P_t \) by an individual learner. It is noted that the overall learning result is consistent in the sense that it converges to the optimal hypothesis as the number of samples goes to infinity. The real advantage of distributed active learning is that it allows different learners to work on different hypothesis spaces. Active learners interact with each other only through the update of sampling probability and in turn the importance weight in the generalization error estimate by the individual learner.

Next, we show that the distributed active learning algorithm preserves the consistency with a high probability. Let \( h^{\text{test}} \in \mathcal{H} \) be the optimal hypothesis that minimizes the expected loss with respect to \( P_{XY} \). For any given \( \delta > 0 \), let \( \Delta_{t-1} = \sqrt{(8/\delta) \log(2t(1 + 1)|\mathcal{H}|/\delta)} \). Then with probability at least \( 1 - \delta \), we have \( h^{\text{test}} \in \mathcal{H}(S_t) \) and

\[
|L(f) - L(g)| < 2\Delta_{t-1}, \forall f, g \in \mathcal{H}(S_t). \tag{12}
\]

Proof: Let \( Q_t \) be the indicator variable of the event that the active learner queries \( (x_t, y_t) \) with probability \( p_t \). For any \( t \) and \( f, g \in \mathcal{H}(S_t) \), define the difference between the empirical loss and the true loss by

\[
W_t = (L(f) - L(g)) - (L(f) - L(g)). \tag{13}
\]

Since the importance weighted loss is an unbiased estimate of the true loss, we have

\[
E[W_t|W_1, ..., W_{t-1}] = 0. \tag{14}
\]

Note that \( f, g \in \mathcal{H}(S_t) \) implies that \( f \) and \( g \) are in \( \mathcal{H}(S_{t-1}) \), ..., \( \mathcal{H}(S_1) \). For a bounded loss satisfying \( p_t \geq |l(f(x_t), y_t) - l(g(x_t), y_t)| \), we have

\[
|W_t| \leq \frac{1}{p_t}|l(f(x_t), y_t) - l(g(x_t), y_t)| + |L(f) - L(g)| \leq 2. \tag{15}
\]
Applying Azuma’s inequality [3] to the martingale sequence $W_t$, we have

$$P(|(L_t(f) - L_t(g)) - (L(f) - L(g))| > \Delta_{t-1})$$

$$= P\left(\sum_{i=1}^{t} W_i > t\Delta_{t-1}\right) < 2e^{-\frac{t^2\Delta_{t-1}^2}{\delta^2}} = \frac{\delta}{t(t+1)|H|^2}.$$ 

Since $\mathcal{H}(S_t)$ is a subset of $\mathcal{H}$, taking a union bound over all $f, g \in \mathcal{H}$ suffices the proof for all $t$. Thus we have

$$|L(f) - L(g)| < |L_{t-1}(f) - L_{t-1}(g)| + \Delta_{t-1} < 2\Delta_{t-1}$$

(16) with probability at least $1 - \delta$.

Note that $\mathcal{H}(S_t)$ contains hypotheses having expected losses within $[L^*, L^* + 2\Delta_{t-1}]$. We would hope the expected sampling probability $P_t$ to be close to $L^* + \Delta_{t-1}$ so that the expected number of queries up to time $t$ is around $L^*t + \sqrt{t}\log|\mathcal{H}|$. Note that an oracle still needs $L^*t$ queries to identify $h^*$ while the additional term in general depends on the VC dimension and disagreement coefficient [11] rather than the cardinality of $\mathcal{H}$ shown here.

C. Combining Active Learners

Assume that $S_T$ is the sample set parsed by $K$ active learners. Let $h_j$ be the hypothesis from the $j$-th classifier. For a binary classification problem, we can effectively combine all classifiers in the following form

$$h(x) = \text{sign}\left(\sum_{j=1}^{K} \alpha_j h_j(x)\right)$$

(17)

where $\alpha_j$ depends on the weighted error rate

$$\hat{e}_j = \frac{1}{T} \sum_{i=1}^{T} \frac{Q_i^j}{P_i} 1(h_j(x_i) \neq y_i).$$

(18)

Since every active learner provides its own best hypothesis as well as the empirical loss of each queried training sample, the combined hypothesis can be interpreted as a boosting strategy [9] with the weights depending on not only the queried samples but also those not being queried by all active learners. Note that there exist other hypothesis combining strategies that boost weak learners’ performance significantly [12]. Here we only consider a generic form of combining active learners that allows optimization via boosting method. For example, AdaBoost chooses [9]

$$\alpha_j = \frac{1}{2} \log \frac{1 - \hat{e}_j}{\hat{e}_j}$$

(19)

which was quite effective in passive learning with weakly coupled classifiers.

D. Coping with Regression, Prediction, and Semi-Supervised Learning

For an active learning algorithm to save the number of queries in the course of identifying the best hypothesis, we need to choose an appropriate loss function that exhibits certain properties. For any $f, g \in \mathcal{H}$, define the maximum disagreement between $f$ and $g$ by

$$\theta(f, g) = \max_{y} E_x [||l(f(x), y) - l(g(x), y)||]$$

(20)

When the version space is reduced using (8), we have shown that, with a high probability, any hypothesis within the set $\mathcal{H}(S_t)$ yields the loss at most $L^* + 2\Delta_{t-1}$ for any time $t$. We would hope that any hypothesis in $\mathcal{H}(S_t)$ is close to $h^*$ in terms of $\theta(h, h^*)$. Specifically, we say that a loss function $l$ has bounded disagreement in the hypothesis space $\mathcal{H}$ if there exists some constant $c_l$ such that

$$\theta(h, h^*) \leq c_l(L(h) + L^*)$$

(21)

Note that for $\mathcal{H} = \{h : X \rightarrow Y\}$, we can choose

$$c_l = \max_{f, g \in \mathcal{H}} \max_{y} \frac{\max_{y}[l(f(x), y) - l(g(x), y)]}{\min_{y}[l(f(x), y) - l(g(x), y)]}$$

(22)

if the loss function $l$ has bounded slope asymmetry [11].

For regression problems, we prefer to use logistic loss instead of the popularly used squared loss owing to its bounded slope asymmetry. For prediction with asymmetrical loss between undershoot and overshoot, we can use the sigmoidal loss with different weights, e.g.,

$$l(y', y) = \begin{cases} \frac{1}{1 + w_u e^{y/(y-y')}} & \text{if } y' < y \\ \frac{1}{1 + w_o e^{y-y'}} & \text{if } y' \geq y \end{cases}$$

(23)

In typical prognostic applications, we have $w_u > w_o > 0$.

For semi-supervised learning problems, we have a few labeled training samples and many unlabeled samples. We will apply $K$ base learning algorithms on the training samples to build the hypothesis spaces and then use the active learning procedure to shrink the version space with the unlabeled samples. The distributed learning algorithm will stop when all unlabeled samples have been updated by each active learner. The fusion center combines the output hypotheses using AdaBoost on the labeled training samples again. The above extension does not rely on any generative model, which is often required in co-training for semi-supervised learning [19].

However, there is also no guarantee that the unlabeled samples will always help without any generative model assumption.

III. BATTERY HEALTH MANAGEMENT

Battery health management is an important application domain where prognostic methods have been applied to predict the remaining useful life (RUL) for a given discharge cycle as well as for its cycle-life. Realistic Li-ion battery data sets for a complete aging process have been collected and provided by Prognostics Center of Excellence at NASA Ames Research Center\(^1\). The currently available data sets contain 34 battery aging trajectories under different operational profiles. Each battery has repeated charge and discharge cycles under different temperatures with its voltage and current measured over time until it is considered no longer usable.

\(^1\)http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/
In addition, impedance cycles were inserted between certain charge and discharge cycles using electrochemical impedance spectroscopy with frequency sweep from 0.1Hz to 5kHz. The whole process can take months running on the battery prognostics testbed and often requires human intervention due to unexpected events. In other prognostic applications, collecting a complete run-to-failure trajectory of the system component can be much more costly and time consuming.

A. Passive Learning vs. Active Learning

Modeling the aging mechanism of a Li-ion battery from the principle of internal electrochemical reactions seems computationally intractable and very sensitive to the operation profile as well as the environmental condition. In order to predict RUL for a given discharge cycle, one often derives the discharge curve with a simplified model depending on a few key factors that can be conveniently monitored, such as the initial capacity and load impedance. Unfortunately, the past usage and rest time of the battery, although crucial for prediction accuracy, is usually hard to be included as part of a practical battery prognostic model. Nevertheless, the prediction of cycle-life depends critically on the knowledge of the future usage profile, which has a lot of uncertainty. In a realistic run-to-failure experiment, battery charge was carried out at a constant current of 1.5A until the voltage reached 4.2V. The charge continued in a constant voltage mode until the charge current dropped below 20mA. Battery discharge was carried out at a constant current of 2A until the battery voltage fell to 2.7V. The battery’s end-of-life (EoL) was declared when its capacity dropped by 30% (from 2Ah to 1.4Ah).

A passive learning method intends to learn the battery aging model using the full trajectory of the training sample. Then the learned model is applied to the RUL prediction of a new Li-ion battery. An active learner decides when to stop the aging process of the repeated charging and discharging cycles so that a significant saving can be made in collecting the training samples. For predicting the end-of-discharge (EoD) time at any particular discharge cycle of each battery, we consider linear regression under a logistic loss. Typically, a battery’s EoD time decreases as the repeated charge/discharge cycles progress. However, a few batteries exhibit significant fluctuations in EoD times among different discharge cycles possibly due to variation in initial capacity, usage profile, depth-of-discharge difference and duration of the rest time. In most of these cases, the actual EoD time is much shorter than expected without any diagnostic indication from the impedance cycle. We ran the regression model on the complete run-to-failure discharge cycles of each battery. Figure 1 shows the EoD time vs. discharge cycle number for all test batteries, where dotted trajectories were well predicted by the regression model, while circles indicate a significant fluctuation of the EoD time for some batteries the regression model failed to predict with the desired accuracy.

For an arbitrarily selected discharge cycle of a Li-ion battery, we can roughly classify its EoD time as predictable (denoted by a dot) or unpredictable (denoted by a circle). We only consider those discharge cycles with the EoD time longer than 1000 seconds. The classification criterion (between dot and circle shown in Figure 1) is whether the prediction made at 500 seconds using the regression model can have precision of 90% accuracy as suggested in [14]. Note that there are several discharge cycles where the capacity was very low in the original battery data set. Nevertheless, there are 21 batteries that contain all discharge cycles with the prediction accuracy of EoD time above 90%. We assume that those dots and circles obtained by the passive learner are true labels indicating the predictability of EoD time for a battery’s discharge cycle. For active learning using the same regression model, we adjust $\Delta t$ to decrease linearly in $1/\sqrt{t}$ and stop the battery aging process when the algorithm does not inquire the label in three consecutive discharge cycles. Interestingly, those 21 batteries with a highly predictable EoD time at 500 seconds were all correctly labeled while the average number of discharge cycles inquired is only 34.3 per battery, indicating a significant saving (around 1/3 of the average RUL for the Li-ion battery data set) in the run-to-failure test.

B. Centralized vs. Distributed Active Learning

We study the EoD time prediction based on the regression model for an individual battery and based on a combination of discharge cycles from multiple batteries in the training set, respectively. A passive learning strategy builds the regression model using the complete run-to-failure discharge cycles for each battery. A centralized active learner starts with early cycles of all batteries and decides whether to inquire the label (predictability) of a particular discharge cycle with a global sampling threshold determined by the empirical loss. Alternatively, we engage an active learner on each battery and the distributed active learners exchange their learned models only through the modification of the sampling probability of an unqueried example. The final prediction of the EoD
time uses the combined regression models from all distributed active learners via AdaBoost. The number of discharge cycles being queried for each battery is shown in Figure 2. Clearly, distributed active learning uses more queries than the single active learner with centralized decision of the sampling threshold. However, we still have about a 1/2 saving in terms of unqueried cycles.

Next, we compare the prediction accuracy of the regression models learned from passive, centralized and distributed active learning methods. For each discharge cycle, we identify the earliest time that the prediction of the EoD time onwards has at least 90% accuracy. This corresponds to the $\alpha$-$\lambda$ metric used in the prognostic area [14] with $\alpha = 0.1$ and $\lambda = 0$. Figure 3 shows the prediction performance sorted by discharge cycles from the best to the worst. The results are truncated when the prediction time is later than 550 seconds, where the prognostic effort is no longer meaningful. We can see that active learning yields only a slight performance degradation compared with passive learning. Interestingly, the combined regression model from distributed active learning outperforms the passive learning in prediction accuracy. This indicates that building a mixture of regression models on all tested batteries rather than on individual ones is beneficial in the EoD time prediction.

C. Battery Remaining Useful Life Prediction

The degradation of Li-ion battery capacity with aging can be encapsulated by Coulombic efficiency, $\eta_C$, defined as the fraction of the prior charge capacity available during the subsequent discharge cycle. During the discharge, a battery will have reaction products built up around the electrodes that slow down the reaction. By letting a battery rest, the reaction products may dissipate, thus increasing the capacity for the next cycle. An empirical aging model given by

$$C_{k+1} = \eta_C C_k + \beta_1 e^{-\beta_2 \Delta t_k}$$

(24)

was proposed in [14] where $C_k$ is the charge capacity at the $k$-th cycle; $\Delta t_k$ is the rest period between the $k$-th and $(k+1)$-th cycle; $\eta_C$, $\beta_1$ and $\beta_2$ are parameters to be estimated from a battery’s aging data. We applied the nonlinear least squares using battery capacity data from all cycles before the end of life (EoL) of each battery to estimate $\eta_C$, $\beta_1$ and $\beta_2$, and we used the learned model to predict the number of cycles a battery can repeatedly charge and discharge before its EoL, assuming the rest period is known for the planned future usage. With the complete run-to-failure profiles of 34 batteries, we have at least 90% EoL prediction accuracy from 40-th cycle onward for 16 batteries. The remaining 18 batteries can not achieve the desired prediction accuracy. We assume that the result represents the true label of the EoL predictability of each testing battery.

Next, we compare the centralized and the distributed active learning algorithms using the same empirical aging model. Due to the small sample size, we only ran 3 active learners in the distributed implementation. We assume that the EoL prediction has to be made at the 40-th cycle for each battery in terms of the remaining number of cycles that the battery can operate with above 70% of its rated capacity. The predictor needs to assess its prediction accuracy as either above 90% or below 90%. A false positive is made when the actual prediction accuracy is below 90% but the predictor made a declaration that it is above. A false negative is made when the actual prediction accuracy is above 90% but the predictor made a declaration that it is below. Table I compares the precision in predictability assessment and the number of queries made on the predictability label using the centralized and distributed active learning methods. Both methods achieved high classification accuracy with only partial queries on the labels. The distributed method made more queries but has zero false positive instance. In addition, it yields higher prediction accuracy in terms of one more false negative instance owing to the fusion of active learning results with different models.

It is worth noting that accurate prediction of a battery’s remaining useful life requires physics-based electro-chemical model [5] as opposed to equivalent circuit model [6], which is
A general distributed active learning method is proposed where each active learner uses importance sampling strategy and updates the sampling probability of each unqueried sample. The fusion of individually best hypotheses from multiple learners is interpreted as a special boosting strategy. The prediction of battery end-of-discharge time and end-of-life is studied and the proposed method is compared with the conventional passive learning method. The empirical study shows that the fusion of distributed active learners achieves better classification and prediction accuracy with a reduced number of training samples. The proposed method has potential to save the cost in collecting realistic prognostic data, which traditionally have to go through run-to-failure tests.

**IV. CONCLUSIONS**

A general distributed active learning method is proposed and updates the sampling probability of each unqueried sample. The fusion of individually best hypotheses from multiple learners is interpreted as a special boosting strategy. The prediction of battery end-of-discharge time and end-of-life is studied and the proposed method is compared with the conventional passive learning method. The empirical study shows that the fusion of distributed active learners achieves better classification and prediction accuracy with a reduced number of training samples. The proposed method has potential to save the cost in collecting realistic prognostic data, which traditionally have to go through run-to-failure tests.

**ACKNOWLEDGEMENT**

This work was supported in part by ARO through grant W911NF-08-1-0409, ONR-DEPSCORe through grant N00014-09-1-1169 and Office of Research & Sponsored Programs at the University of New Orleans. The authors are grateful to Dr. Kai Goebel at NASA Ames Research Center for providing battery data sets and constructive comments to the experimental study of battery remaining useful life prediction.

**REFERENCES**