# Long Short-Term Memory Recurrent Neural Network for Remaining Useful Life Prediction of Lithium-Ion Batteries

Yongzhi Zhang, *Student Member, IEEE*, Rui Xiong, *Senior Member, IEEE*, Hongwen He, *Senior Member, IEEE*, and Michael G. Pecht, *Fellow, IEEE*

***Abstract*—Remaining useful life (RUL) prediction of lithium-ion batteries can assess the battery reliability to determine the advent of failure and mitigate battery risk. The existing RUL prediction techniques for lithium-ion batteries are inefficient for learning the long-term dependencies among the capacity degradations. This paper investigates deep-learning-enabled battery RUL prediction. The long short-term memory (LSTM) recurrent neural network (RNN) is employed to learn the long-term dependencies among the degraded capacities of lithium-ion batteries. The LSTM RNN is adaptively optimized using the resilient mean square backpropagation method, and a dropout technique is used to address the overfitting problem. The developed LSTM RNN is able to capture the underlying long-term dependencies among the degraded capacities and construct an explicitly capacity-oriented RUL predictor, whose long-term learning performance is contrasted to the support vector machine model, the particle filter model, and the simple RNN model. Monte Carlo simulation is combined to generate a probabilistic RUL prediction. Experimental data from multiple lithium-ion cells at two different temperatures is deployed for model construction, verification, and comparison. The developed method is able to predict the battery's RUL independent of offline training data, and when some offline data is available, the RUL can be predicted earlier than in the traditional methods.**

***Index Terms*—Lithium-ion battery, remaining useful life, deep learning, long short-term memory, Monte Carlo simulation.**

## I. Introduction

LITHIUM-ION batteries are widely used as power sources for many types of systems, including consumer electronics, electric vehicles (EVs), airplanes, and spacecrafts, owing to their high energy density, high galvanic potential, and long lifetime [1], [2]. However, their performance degrades over repeated charge/discharge cycles. For many applications, failure is considered to occur when the battery's capacity decreases below 80% of its initial capacity [3]. Because the battery's capacity and power both tend to drop much faster, its performance is thus unreliable after this point, and it should be replaced. Battery failure could result in degraded capability, unavailable operation, downtime, and even catastrophic occurrence.

Prognostics and health management (PHM) is an enabling discipline consisting of methods and technologies to evaluate system reliability under real cycle life conditions to detect incipient faults and predict fault progression [4]. PHM of lithium-ion batteries enables users to make maintenance decisions in advance to prevent loss caused by unexpected failures. Remaining useful life (RUL) prediction is a key approach of PHM [5].

To predict the RUL of lithium-ion batteries, an aging model is needed to map input sequences to output sequences of degradation characteristics. The prediction requires a system that will store and update degradation information (i.e., information computed from the past inputs and useful to produce desired outputs). This information is then used to capture the long-term dependencies of degradation evolution of lithium-ion batteries [6].

Lithium-ion battery degradation can cover hundreds of cycles, and the degradation evolution among these cycles is highly related. The degradation characteristics data consists of a trajectory that can be predicted based on historical data. However, some data contributes more to constructing the trajectory because the historical data is noisy and can even possess outliers limited to the present measurement technology. It is thus reasonable and efficient to predict the degradation trajectory by extracting key information from the inputs [7]. The difficulty for RUL prediction of lithium-ion batteries lies in how to store and update the key information from the degradation data via effective learning of long-term dependencies. RUL prediction methods for batteries can be categorized into two main categories: model-based methods and data-driven methods.

Manuscript received June 13, 2017; revised November 20, 2017, December 24, 2017, and January 24, 2018; accepted February 7, 2018. Date of publication February 12, 2018; date of current version July 16, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 51507012 and in part by the Beijing Nova Program under Grant Z171100001117063. The systemic experiments of the lithium-ion batteries were performed at the Advanced Energy Storage and Application (AESA) Group, Beijing Institute of Technology. The review of this paper was coordinated by Dr. M. Kazerani. *(Corresponding author: Rui Xiong.)*

Y. Zhang is with the National Engineering Laboratory for Electric Vehicles, Department of Vehicle Engineering, School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China, and also with the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: yzzhangbit@gmail.com).

R. Xiong and H. He are with the National Engineering Laboratory for Electric Vehicles, Department of Vehicle Engineering, School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China (e-mail: rxiong@bit.edu.cn; hwhebit@bit.edu.cn).

M. G. Pecht is with the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: pecht@umd.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2805189

Model-based methods use mathematical models to capture the long-term dependencies of battery degradations, which are combined with advanced filtering techniques, such as the particle filter (PF) algorithm, to predict the battery RUL [8]–[13]. The filtering process can be used to update the model parameters by evaluating the importance of the data points adaptively based on the measured signals. He *et al.* [9] used the Dempster–Shafer theory (DST) and PF method to predict the RUL of batteries. DST was used to initialize model parameters, and PF was then used to update the model parameters and predict the RUL based on the available data through battery capacity monitoring. To further improve modeling accuracy, Xing *et al.* [10] developed an ensemble model to characterize the capacity degradation and predict the RUL of batteries. Due to the lack of knowledge of the degradation evolution, the mathematical model is usually constructed by fitting the degraded capacities of lithium-ion batteries, but this step could lead to overfitting. The model is able to fit the training data accurately but its prediction accuracy is low. To solve this problem, the Akaike information criterion (AIC) was introduced to impose a heavier penalty on model complexity and thus avoid overfitting [10]. To add more battery mechanism information to PF-based RUL prediction methods, Lyu *et al.* [11] introduced an electrochemical model-based PF framework to describe the capacity degradation of a battery and thus predict battery RUL. In the PF framework, model parameters that reflected battery degradation were seen as state variables. Liu *et al.* [12] proposed an improved particle learning (PL) framework to prevent particle degeneracy and impoverishment. They resampled the state particles first based on the newly obtained measurement and then propagated them to generate state posterior particles at the current time. Su *et al.* [13] developed an improved method for determining battery RUL based on the interacting multiple model particle filter (IMMPF). This method was able to fuse various capacity models of lithium-ion batteries. Although significant progress has been made in model-based RUL prediction methods in recent years, two drawbacks still exist. First, there is no accurate aging model that can be used as a basis for RUL prediction [5]. Second, RUL prediction accuracy based on the most widely used filter method, PF, is limited by the particle degeneracy problem [12].

Compared to model-based methods, data-driven methods do not need an explicit mathematical model to describe the degradation evolution of batteries and are only dependent on historical degradation data. Data-driven methods basically extract key degradation information from the data points by a specific learning algorithm. Support vector machine (SVM) provides support vectors as the dominant data points to construct the aging model of lithium-ion batteries. Nuhic *et al.* [14] used SVM to embed diagnosis and prognostics of system health with an aim to estimate the state of health (SOH) and RUL of lithium-ion batteries. Patil *et al.* [15] proposed a multistage SVM approach for RUL prediction of lithium-ion batteries. This approach integrated classification and regression attributes of SVM, in which the classification model provided a gross estimation and the regression model was used to predict the RUL. Wang *et al.* [7] used relevance vector machine (RVM) to derive relevance vectors that can represent the capacity degradation trajectories of lithium-ion batteries. These relevance vectors were then used to fit a capacity degradation model to extrapolate the RUL. Zhou *et al.* [16] employed the Box–Cox transformation to build a relationship between the extracted features and capacities of lithium-ion batteries, and RVM was then used to predict RUL based on the extracted features. Kernel techniques applied for the RUL prediction are able to select the dominant data points to represent the degradation evolution of lithium-ion batteries. This reduction of degradation data improves the calculation efficiency but decreases the prediction accuracy since all degradation data points should contribute more or less to constructing an accurate aging model.

Besides the kernel techniques, neural network (NN) techniques were also employed to predict the RUL of lithium-ion batteries. Much earlier, Liu *et al.* [17] proposed an adaptive recurrent neural network (RNN) for system dynamic state forecasting for the purpose of predicting the RUL of lithium-ion batteries. RNNs are well suited for this task because they have an internal state that can represent aging information. The cycles in the graph of a recurrent network allow it to keep information about past inputs for an amount of time that is not fixed a priori, but rather depends on its weights and on the input data. Based on the degradation data of lithium-ion batteries from NASA, Liu *et al.* [17] validated that the adaptive RNN showed better learning capability than classical training algorithms, including RVM and PF methods. Since then, there have been no published studies on employing NN techniques for battery RUL predictions except for one paper published by Rezvani *et al.* [18]. Rezvani et al. used both RNN and linear prediction error method (L-PEM) for battery capacity estimation and RUL prediction. They observed that RNN provided a more accurate capacity prediction of one-step ahead while L-PEM showed a more accurate RUL prediction.

This lack of research is mainly due to the "Achilles heel" of RNN/long-term dependencies learning. In other words, if RNN allows the storage of information over some time, the network gradients tend to vanish. In this case, RNN will be unable to learn anymore [6]. The long short-term memory (LSTM) RNN, which is a deep-learning NN, is explicitly designed to learn the long-term dependencies. It is able to remember information for long periods of time via the introduced gates [19]–[21]. The forget gate is able to discard redundant information; the input gate is able to select key information to be stored in the internal state; and the output gate is used to determine output information. Readers can refer to Part A in Section III for more detailed descriptions and definitions of these gates. In this case, LSTM RNN is able to store and update the key information efficiently over a long time without gradients vanishing. LSTM RNN has been widely and successfully applied in language modeling [22], machine translation [23], image captioning [24], and handwriting generation [25]. Guo *et al.* [26] presented LSTM RNN to forecast streaming time-series data in the presence of anomalies and change points. Qu *et al.* [27] proposed a wind power prediction model based on the LSTM RNN. Simulation results showed that LSTM RNN had higher prediction accuracy than SVM. Duan *et al.* [28] used LSTM RNN for travel time prediction. The capacity degradation data, which covers hundreds

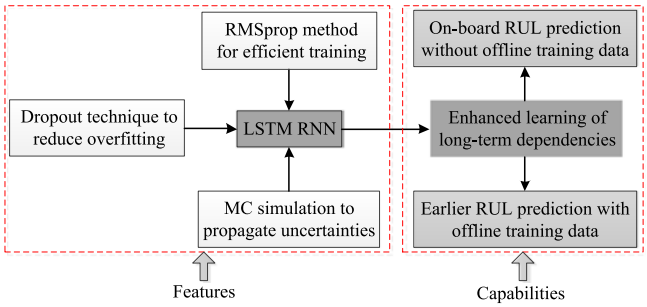


Fig. 1. Features and capabilities of the constructed LSTM RNN.

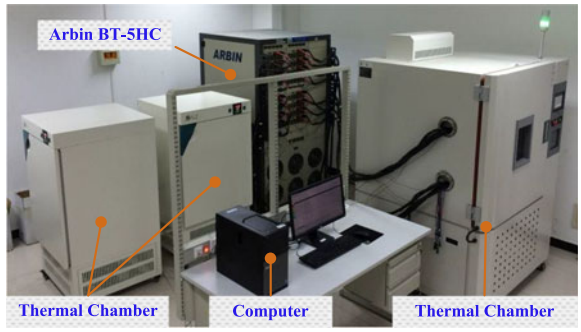


Fig. 2. Battery test equipment.

of charge/discharge cycles and represents the degradation evolution of batteries, can be regarded as long-term time series data. Therefore, LSTM RNN is used to learn the long-term dependencies and predict the RUL of lithium-ion batteries.

Fig. 1 shows the features and capabilities of the constructed LSTM RNN for battery RUL prediction. The network is trained with a mini-batch-based optimization algorithm—the resilient mean square back-propagation (RMSprop) method [29]; the overfitting of LSTM RNN is addressed by introducing the dropout technique [30]; and Monte Carlo (MC) simulation is used to generate the prediction uncertainties. The constructed LSTM RNN is able to learn long-term dependencies and can predict the battery RUL independent of offline training data. When some offline data is available, only a small amount of online data is required by LSTM RNN for accurate RUL prediction, indicating that the RUL prediction can be made earlier than in the traditional methods. In the literature [9]–[13], when some offline data is available, the required online data for accurate RUL prediction generally amounts to 40%–70% of the complete data of the online cell, whereas the required online data by LSTM RNN amounts to 20%–25% of the complete data. In this paper, the data collected in real applications is defined as online data, whereas the data collected in the experiments is defined as offline data. Therefore, an online cell refers to the cell whose degradation data is considered to be collected on board, and its RUL needs to be predicted.

The rest of this paper is organized as follows: Section II describes the battery testing and degradation data. Section III describes the LSTM-RNN-oriented RUL prediction method, followed by the experimental results of RUL predictions in Section IV, and Section V presents conclusions.

## II. Battery Testing and Degradation Data

Fig. 2 shows the equipment used to conduct the acceleration aging tests of lithium-ion batteries. It includes an Arbin BT-5HC test system to charge/discharge batteries, three thermal chambers to control battery temperatures, and a computer for data monitoring and storage.

To investigate the aging characteristics of lithium-ion batteries, high-energy 18650 lithium-ion batteries manufactured by Panasonic, labeled NCR18650PF, were used. The cells had a rated capacity of 2.7 Ah. The nominal voltage was 3.6 V, with upper and lower cut-off voltages of 4.2 V and 2.5 V, respectively. The materials consisted of graphite on the anode and

TABLE I
Cell Specifications of the NCR18650PF Battery

| | |
|---|---|
| Nominal Capacity | 2.7 Ah |
| Material | $Li(NiCoAl)O_2$/Carbon |
| Maximum continuous discharge current | 10 A |
| Allowed voltage range | 2.5–4.2 V |
| Temperature range | |
| Charge | 10–45 °C |
| Discharge | −20–60 °C |
| Storage | −20–50 °C |

$Li(NiCoAl)O_2$ on the cathode. The cell specifications are summarized in Table I.

Cells were cycled in the same state of charge (SOC) range (0%–100%) under different temperatures and current rates, and in each case, one or two cells were cycled. Temperature was measured at the cell level using temperature sensors on the cell surface. Note that the temperature used in the paper refers to the ambient temperature rather than the cell surface temperature. At 25 °C, the cells were, respectively, discharged under 1C and 2 C current rate, while at 40 °C, the cells were, respectively, discharged under 1 C and 3.5 C current rate. The charging profile of all cells is a constant current (CC)-constant voltage (CV) process with a CC rate of 0.5 C followed by a CV of 4.2 V, and the cut-off current rate is 0.05 C. When the cell was discharged to the low cut-off voltage, a 0.5 C current rate was then loaded on the cell until the low cut-off voltage was reached again. As the battery capacity can be estimated in real-time based on the measured current and voltage, the capacity is considered to be available and thus is used to indicate the battery health state [5]. The capacity used to indicate battery degradation was obtained by accumulating all discharged capacities. The aging test ended at 600 cycles. The capacity degradation trajectories of batteries under different temperatures and current rates are presented in Fig. 3. At 25 °C, cells 1 and 2 were cycled under 1 C and 2 C, respectively, and are considered online cells, whereas cell A tested under 1 C is considered an offline cell. At 40 °C, cells 3 and 4 were cycled under 1 C and 3.5 C, respectively, and are considered online cells, whereas cell B tested under 1 C is considered an offline cell.

Table II lists the capacity retention of each cell at the end cycle (600th cycle). The initial capacity is the maximum discharge capacity of each cell, which is usually obtained during the first several cycles. Cells 1 and A were cycled under a lower current

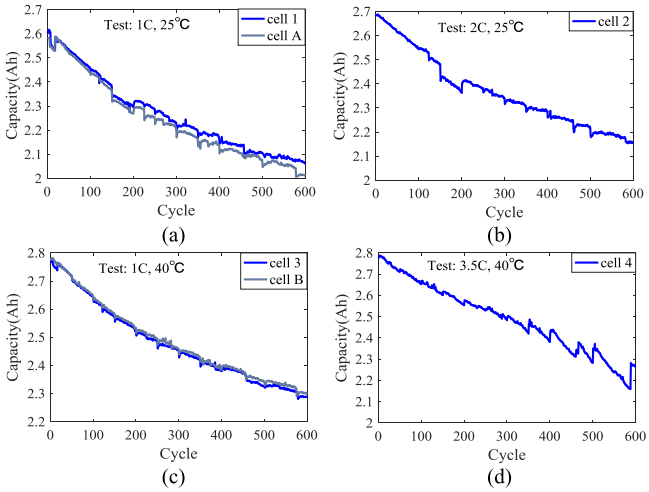


Fig. 3. Capacity degradation trajectories of batteries under different temperatures and currents.

TABLE II
CAPACITY RETENTION AT 600 CYCLES

| Capacity | Cell 1 | Cell 2 | Cell 3 | Cell 4 | Cell A | Cell B |
|---|---|---|---|---|---|---|
| Initial capacity (Ah) | 2.62 | 2.69 | 2.78 | 2.79 | 2.58 | 2.78 |
| Capacity retention | 79% | 81% | 83% | 77% | 78% | 82% |

than that loaded on cell 2, thus it is counterintuitive that their capacity decreased even more quickly than cell 2. Although cells 1 and A were cycled at a lower temperature than that loaded on cells 3 and B, their capacity still decreased more quickly than cells 3 and B. These anomalies can be caused by the specific degradation mechanisms or manufacturing variability [31]. Note that the failure threshold value of cells 1 and 4 is 80% capacity retention, while for cells 2 and 3, the failure threshold values are 81% and 83% capacity retention, respectively.

The experimental data from the battery aging tests was used to evaluate the performance of the developed battery RUL prediction method in real applications. Although lithium-ion batteries in practice work under changeable currents, temperatures, and SOCs, the capacity degradation trends are expected to be similar under different working conditions [32], [33]. This degradation similarity can be observed in Fig. 3, where the capacity follows similar nonlinear degradations under different conditions. Therefore, the verification results of the developed method based on capacity data in Fig. 3 can be used to evaluate its prediction performance in real applications.

## III. LSTM-RNN-ORIENTED RUL PREDICTION

When applying LSTM RNN to predict the RUL of lithium-ion batteries, there are three concerns. First, traditional training algorithms based on batch gradient descent or stochastic gradient descent for deep-learning networks can be significantly slow to converge to the correct network weights, which are thus not applicable for the purpose of online failure prediction of lithium-ion batteries. Second, overfitting is a serious problem in deep neural networks like the LSTM RNN. Third, the maintenance of lithium-ion batteries is based on an RUL probability distribution function (PDF), while LSTM RNN is unable to obtain uncertainties of the RUL predictions.

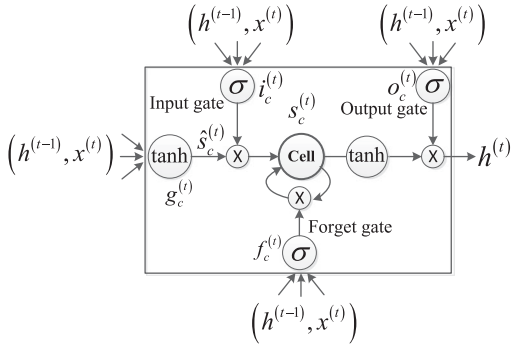


Fig. 4. Network architecture of the LSTM RNN predictor [21].

To address these challenges, the developed LSTM RNN predictor includes four main components: a long short-term memory and recurrent neural network architecture, network parameter optimization using the RMSprop method, a dropout technique to prevent the neural network from overfitting, and MC simulation to generate prediction uncertainties.

### A. LSTM RNN Architecture

The network architecture of the LSTM RNN predictor is schematically shown in Fig. 4 [21]. All elements of the LSTM RNN are enumerated and described below. Note that the subscript $c$ of each vector represents the neuron, and the superscript $(t)$ represents the moment $t$. $x^{(t)}$ is the input layer at the current moment $t$, and $h^{(t-1)}$ is the hidden layer at the previous moment $t-1$. "Moment" is used here for the generality of LSTM RNN applications, and for the RUL prediction of lithium-ion batteries, "moment" amounts to "cycle".

The key to the LSTM RNN is the internal state $s_c$. It is at the heart of each neuron that has been linearly activated. The internal state $s_c$ has a self-connected recurrent edge with fixed unit weight. The internal state can be regarded as a carrier to which information has been added or from which has been removed. This flow of information can be carefully regulated by structures called "gates". Gates are a distinctive feature of the LSTM approach and are able to optionally decide what information should pass though. A gate is a sigmoidal unit that is activated from the current input layer $x^{(t)}$ as well as from the hidden layer at the previous time step.

Forget gate: The first step in the LSTM RNN is to decide what information is going to be discarded. This decision is made by a sigmoid layer called the "forget gate", $f_c$. It takes inputs of $x^{(t)}$ and $h^{(t-1)}$ and outputs a number between 0 and 1 for each state value in the internal state $s_c$. A 1 represents retaining the state value completely, whereas a 0 represents discarding the value completely. For example, for RUL prediction of lithium-ion batteries, the discarded information can be the outliers, noises, and redundant information of degradation data among adjacent

cycles. The forget gate is calculated as follows:

$$f^{(t)} = \sigma\left(W^{fX}x^{(t)} + W^{fh}h^{(t-1)} + b_f\right) \tag{1}$$

Input gate and input node: The next step is to decide what new information is going to be stored in the internal state. This step has two parts. First, a sigmoid layer called the "input gate", $i_c$, decides which values to update. Then, a tanh layer called the "input node", $g_c$, creates a vector of the new candidate state $\hat{s}_c^{(t)}$, which could be added to the state. The equations to calculate the two outputs are

$$i^{(t)} = \sigma\left(W^{iX}x^{(t)} + W^{ih}h^{(t-1)} + b_i\right) \tag{2}$$

$$g^{(t)} = \tanh\left(W^{gX}x^{(t)} + W^{gh}h^{(t-1)} + b_g\right) \tag{3}$$

Combining (1)–(3) to update the previous internal state $s_c^{(t-1)}$ into the current state $s_c^{(t)}$:

$$s^{(t)} = g^{(t)*}i^{(t)} + s^{(t-1)*}f^{(t)} \tag{4}$$

Output gate: Finally, there is a sigmoid layer called the "output gate", $o_c$, that determines what information to output. After putting the internal state $s_c$ through a tanh layer (to push the values to be between $-1$ and 1), it is multiplied by the output of the sigmoid gate to obtain the remaining state value. This can be implemented as

$$o^{(t)} = \sigma\left(W^{oX}x^{(t)} + W^{oh}h^{(t-1)} + b_o\right) \tag{5}$$

$$h^{(t)} = \tanh\left(s^{(t)}\right)^* o^{(t)} \tag{6}$$

where $W$ and $b$ values are the layer weights and biases, respectively.

### *B. LSTM RNN Training*

The mini-batch gradient descent-based optimization algorithm is introduced to train LSTM RNN in this section. Mini-batch gradient descent performs a gradient update for a mini-batch of *n* training examples and thus takes the best of the traditional methods including the batch gradient descent and the stochastic gradient descent (SGD). It computes the gradient for the *n* samples simultaneously by using highly optimized matrix optimizations, which are highly efficient, especially for deep learning. This computation reduces the variance of the gradient updates, which can lead to more stable convergence [34].

The resilient back-propagation (Rprop) algorithm proposed by Riedmiller and Braun [35] is a widely used optimization algorithm for learning with neural networks. Because it is substantially different than other gradient descent-based adaptive techniques, the effect of the Rprop adaptation process is not blurred by the unforeseeable influence of the size of the gradient but is only dependent on the temporal behavior of its sign. This leads to an efficient and transparent adaptation process. Rprop is successful for batch learning, but not for mini-batch learning. Rprop is equivalent to using the gradient but also dividing by the size of the gradient, therefore the gradient is divided by a different number for each mini-batch and this leads to slow convergence. To combine the robustness of Rprop and the efficiency of mini-batches, Tieleman and Hinton [29] developed the Rprop and proposed the RMSprop. RMSprop keeps a moving average of the squared gradient for each weight and uses the moving average to divide the gradient. This can force the dividend value to be similar for adjacent mini-batches [29]. An update of the network parameters $\theta$ (weights and biases) for RMSprop is as follows:

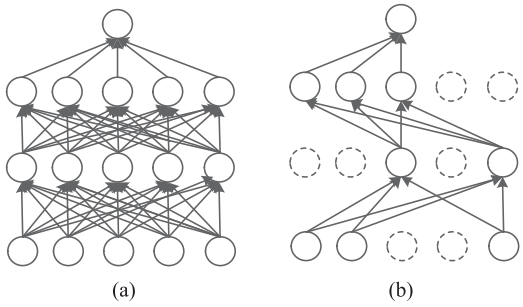


Fig. 5. Dropout neural network model: (a) standard neural network with 2 hidden layers; (b) neural network after applying dropout.

$$g_t = \nabla_\theta J\left(\theta_t; x^{(i:i+n)}; y^{(i:i+n)}\right) \tag{7}$$

$$E\left[g^2\right]_t = \gamma E\left[g^2\right]_{t-1} + (1-\gamma)\, g_t^2 \tag{8}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \varepsilon}} g_t \tag{9}$$

where $(x^{(i:i+n)}, y^{(i:i+n)})$ represents $n$ samples of each mini-batch; $x, y$ represents the network inputs and targets, respectively; $J(\theta; x^{(i:i+n)}; y^{(i:i+n)})$ represents the objective function of each mini-batch; $g_t$ denotes the gradient of the objective function with regard to the parameter $\theta$ at time step $t$; $\gamma$ is a coefficient that determines the average level of the squared gradient; $\eta$ denotes the learning rate; and $\varepsilon$ is a smoothing term that avoids division by zero. Tieleman and Hinton [29] suggested $\gamma$ to be set to 0.9, while a good default value for $\eta$ was 0.001. Note that these two default values were both used in this paper.

### *C. Dropout to Prevent LSTM RNN From Overfitting*

L1 and L2 regularization methods [36], [37] are the two most widely used methods to prevent neural networks from overfitting. Both of these methods regularize the weights learning by adding an extra term to the cost function. For L1 regularization, the sum of all the weights in the network is added, while for L2 regularization, the sum of the squares of all the weights is added. Srivastava *et al.* [30] proposed a dropout technique to address this problem more efficiently. This technique randomly drops neurons (along with their connections) from the neural network during training. The neuron is temporarily removed from the network, along with all its incoming and outgoing connections, as shown in Fig. 5. Each neuron is retained with a fixed probability *p* independent of other neurons. Applying dropout to a neural network amounts to sampling a "thinned" network from it. The thinned network consists of all the neurons that survived dropout [see Fig. 5(b)]. So training a neural network with dropout can

be regarded as training many thinned networks with extensive weight sharing, where each thinned network is trained rarely. The network thus becomes less sensitive to the specific weights of neurons. This in turn results in a network that is capable of better generalization. Srivastava *et al.* [30] suggested a preferred $p$ value of 0.5 for a wide range of networks and tasks.

### D. Monte Carlo Simulation

Although LSTM RNN is capable of learning long-term dependencies on time-series data and predicting the multi-step-ahead data points, it is unable to provide prediction confidence. The predicted value and its confidence are both vital parameters to evaluate the performance of a predictor. To address this problem, MC simulation is applied. This technique, which was specifically proposed to make predictions where there are significant uncertainties [38], is able to propagate the input uncertainties into prediction uncertainties. To generate a sufficiently high number of process realizations, an MC simulation requires a numerical algorithm, which is used to describe the system. In this paper, all information that can be gathered about the performance degradation process of a battery is contained in the capacity data. The LSTM RNN, which is trained based on the capacity data, is thus defined as the numerical algorithm for the process realizations of battery degradations. A variety of degradation process realizations of battery characteristics can be produced by starting the generation with different initial conditions. For initializing the LSTM RNN, the starting input vector needs to be defined. These are generated randomly by being drawn from a Gaussian distribution of the observed capacity degradation processes of batteries. This initialized vector is then propagated through the LSTM RNN, with a certain realization of the stochastic processes in the algorithm.

## IV. Results and Discussion

In this section, an analysis of six degradation datasets under different temperatures and current rates was conducted to validate the developed approach. Python 3.5 was used to program for simulation. The simulation was implemented on a laptop equipped with an Intel Core i7-6700 HQ processor (6 MB cache, up to 3.50 GHz), and a graphic card of NVIDIA Quadro M1000M at 4 GB. When there was no offline training data, the SimRNN and the SVM were employed to predict the RUL of lithium-ion batteries as comparisons. As a state-of-the-art kernel technique for nonlinear regression, the SVM model was considered for benchmarking purposes. When some offline training data was available, the SimRNN and the PF algorithms were applied for RUL prediction as comparisons. Note that both the LSTM RNN and SimRNN algorithms were computed based on the graphic card with a parallel calculation distributed over its multi-graphics processor unit (GPU) to accelerate the simulation, whereas the SVM and PF algorithms were computed based on the Intel Core i7-6700 HQ processor. For fair comparisons, the SimRNN algorithm had the same structure as the LSTM RNN except that the LSTM layers were all replaced with the simple RNN layers.

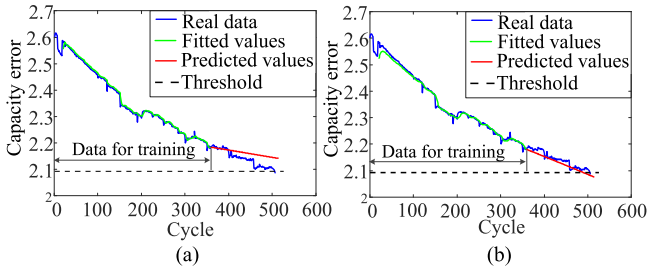


Fig. 6. LSTM RNN-based RUL prediction results for cell 1: (a) prediction results with no methods to prevent overfitting; (b) prediction results with dropout.

TABLE III
RUL Prediction Results for Cell 1

| Overfitting Prevention Method | Capacity Fitting Error (Ah) | Capacity Prediction Error (Ah) | RUL Prediction Error (Cycle) |
|---|---|---|---|
| No method | $5.7 \times 10^{-3}$ | $2.4 \times 10^{-2}$ | NAN |
| Dropout | $6.6 \times 10^{-3}$ | $1.2 \times 10^{-2}$ | 16 |
| L1 regularization | $5.7 \times 10^{-3}$ | $8.2 \times 10^{-2}$ | 93 |
| L2 regularization | $1.4 \times 10^{-2}$ | $1.2 \times 10^{-2}$ | –170 |

The applied LSTM RNN was constructed with one input layer that fed into an LSTM layer with 50 neurons. This layer in turn fed into another LSTM layer with 100 neurons, which then fed into a fully connected normal layer of 1 neuron with a linear activation function. In the experiment, it was observed that a retained probability of 0.2 of each neuron rather than the suggested value of 0.5 [30] usually predicted more accurate RULs of lithium-ion batteries. Therefore, the retained probability of hidden layers of both LSTM RNN and SimRNN was set to 0.2. This meant that for each neuron in the hidden layer, there was a 20% probability it would be retained at each epoch of training.

### A. Dropout to Prevent LSTM RNN From Overfitting

To show the effectiveness of the dropout technique to prevent LSTM RNN from overfitting, different overfitting prevention techniques were used as comparisons, and the RUL prediction results are presented in Fig. 6 and Table III. Fig. 6 plots the RUL prediction results of cell 1, where Fig. 6(a) shows the LSTM RNN-based RUL prediction results without any technique to prevent overfitting, whereas in Fig. 6(b), an LSTM RNN-based predictor combined with the dropout was used to predict the battery RUL. Comparing Fig. 6(a) to (b) shows that the LSTM RNN, without any method to prevent overfitting, has a higher fitting accuracy but lower prediction accuracy and even fails to report an advanced failure. This low prediction accuracy indicates a serious overfitting problem. In Table III, the capacity fitting or prediction error is the mean absolute error between real capacity and fitting or prediction capacity. The RUL prediction error is the error between the real failure cycle and the predicted failure cycle. Table III also lists the RUL prediction results based on the LSTM RNN combined with another two standard regularizers including L1 regularization [36] and L2 regularization [37]. The L1 regularization-based LSTM RNN

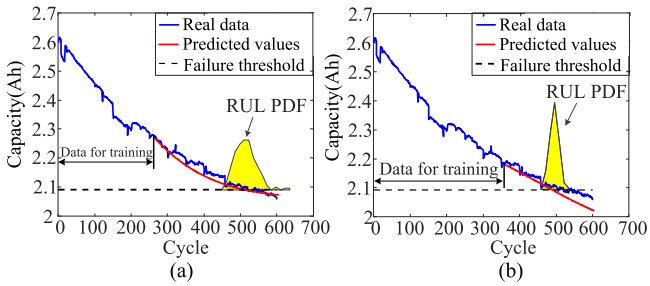


Fig. 7. LSTM RNN-based prediction results of RUL for cell 1: (a) prediction results at 253 cycles; (b) prediction results at 354 cycles.

TABLE IV
RUL PREDICTION RESULTS FOR CELL 1

| Method | Starting Cycle | Error | STD | 95% Confidence Bound | Training Time (s) |
|---|---|---|---|---|---|
| LSTM RNN | 253 | –3 | 28 | [470, 564] | 20.74 |
| | 354 | 15 | 11 | [473, 511] | 23.04 |
| SVM | 253 | –21 | 9 | [511, 546] | 0.003 |
| | 354 | 30 | 9 | [458, 496] | 0.008 |
| SimRNN | 253 | 135 | 7 | [358, 388] | 44.15 |
| | 354 | 78 | 19 | [395, 470] | 55.25 |

predicts the RUL with an error being 93 cycles, which is 77 cycles longer than that of the dropout-based LSTM RNN. The L2 regularization-based LSTM RNN predicts accurate future capacities, but with a large prediction error of failure, which is 170 cycles longer than the real failure and the absolute value of which is 154 cycles longer than that of the dropout-based LSTM RNN. Therefore, when using the LSTM RNN to predict the RUL of lithium-ion batteries, the dropout reduces overfitting more significantly than the other two standard regularizers with higher prediction accuracies of both the capacities and RULs. Therefore, this paper uses the dropout to prevent LSTM RNN from overfitting.

### *B. RUL Prediction Without Offline Training Data*

The degradation data of cells 1 to 4 was utilized to realize the RUL predictions by using the developed method independent of the offline training data. For each cell, two training strategies were conducted on the network, respectively. One strategy of LSTM RNN training covered 50% of the complete degradation data of the online cell, and the other one covered 70% of the complete degradation data. When the training was stopped, the LSTM RNN was used to predict the multi-step-ahead capacity data in the following cycles. When the predicted capacity was lower than the threshold value, a failure was reported.

The capacity of cell 1 dropped below the threshold at 506 cycles. The RUL prediction results for cell 1 by using the LSTM RNN are presented in Fig. 7. Fig. 7(a) shows the prediction results at 253 cycles, when only data from the first 253 cycles was used to train the model. Fig. 7(b) shows the RUL prediction results at 354 cycles. Table IV lists the statistics of the RUL predictions for cell 1. The prediction uncertainties were obtained via MC simulation. The negative RUL prediction error indicates a delayed predicted failure, whereas the positive error indicates an advanced predicted failure. In practice, an advanced predicted failure with a small standard deviation (STD) value is expected. The 95% confidence interval was obtained based on the RUL PDF. The training time is the time taken for training the algorithm until a low training error was obtained. Fig. 7 shows that the PDF of RUL prediction based on the LSTM RNN is narrower when more data is used for training, meaning that the confidence level of the prediction increases. Table IV shows that, when the RUL is predicted at 253 cycles, the RUL prediction error based on the LSTM RNN is $-3$ cycles, which is an accurate failure prediction, whereas the failure prediction error predicted by the SVM is up to $-21$ cycles. Although the STD of the RUL prediction based on SVM is smaller, the 95% confidence bound, which is 511–546 cycles, covers a range after the failure. When more data is used for training, the RUL prediction by using both the LSTM RNN and the SVM change into an advanced prediction of failure. The LSTM RNN predicts a more accurate RUL than the SVM, and the prediction error is 15 cycles, which is 15 cycles shorter than that based on the SVM. The STD of the RUL prediction based on the SVM is only 2 cycles shorter than that based on the LSTM RNN. Still, the 95% confidence bound of the predicted RUL based on the SVM, which is 458–496 cycles, fails to cover the real failure cycle. The accurate and precise RUL predictions based on the developed method indicate that the LSTM RNN captures the long-term dependencies of capacities more accurately and efficiently, whereas the degradation model constructed by the SVM with only support vectors to capture the long-term dependencies predicts a more advanced (at 253 cycles) or delayed (at 354 cycles) failure than the LSTM RNN. The prediction error of the failure cycle by using the SimRNN at 253 cycles is up to 135 cycles, and the STD is 7 cycles. The 95% confidence bound of the predicted failure cycle is 358–388 cycles, which is significantly distant from the real failure cycle. When the prognostic is implemented at 354 cycles, the 95% confidence bound of the RUL prediction is 395–470 cycles, which moves closer to the real failure cycle but still is unable to cover it. This large prediction error indicates that, without any gates introduced, the SimRNN is unable to store important information of battery degradation for a long time and thus fails to learn long-term dependencies efficiently.

The training time of the SVM is less than 0.01 s, which is much less than the training time of the SimRNN and LSTM RNN, which are more than 40 s and 20 s, respectively. When the SimRNN is trained, hundreds of epochs are needed to obtain a high estimation accuracy, whereas the number of training epochs needed to train the LSTM RNN is usually less than 50. Thus, more time is taken by training the SimRNN. In practice, the training time to construct an LSTM RNN with a high estimation accuracy should be longer and can be tens of minutes. However, the RUL of lithium-ion batteries is usually predicted every dozens of cycles, and the interval time is much longer than the training time. Therefore, on-board regular training of the LSTM RNN for an accurate RUL prediction is practical.

The capacity of cell 2 dropped below the threshold at 571 cycles. Fig. 8 shows the RUL prediction results of cell 2 based on the LSTM RNN. The RUL PDFs presented in Fig. 8(a) and

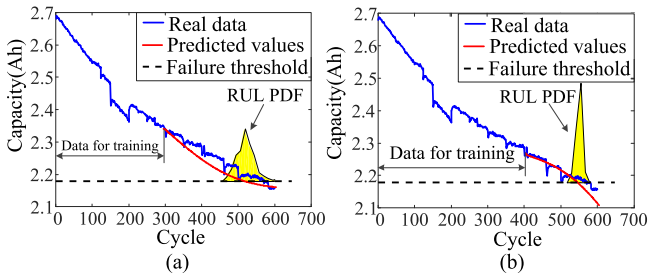


Fig. 8. LSTM RNN-based prediction results of RUL for cell 2: (a) prediction results at 285 cycles; (b) prediction results at 399 cycles.

TABLE V
RUL PREDICTION RESULTS FOR CELL 2

| Method | Starting Cycle | Error | STD | 95% Confidence Bound | Training Time (s) |
|---|---|---|---|---|---|
| LSTM RNN | 285 | 48 | 24 | [487, 585] | 20.49 |
| | 399 | 26 | 7 | [532, 561] | 28.50 |
| SVM | 285 | –34 | 9 | [589, 623] | 0.005 |
| | 399 | 42 | 8 | [517, 546] | 0.009 |
| SimRNN | 285 | 195 | 6 | [368, 392] | 41.73 |
| | 399 | 95 | 18 | [442, 514] | 56.86 |

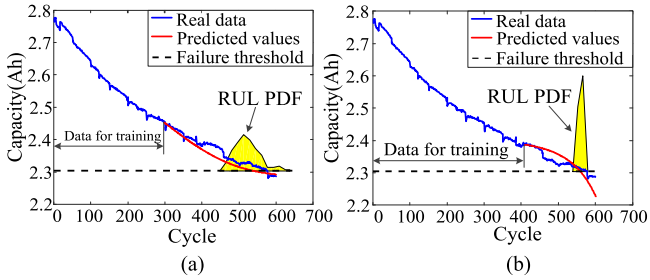


Fig. 9. LSTM RNN-based prediction results of RUL for cell 3: (a) prediction results at 289 cycles; (b) prediction results at 404 cycles.

(b) indicate that, when more data is used for training, more key degradation information is obtained, leading to a more accurate and precise RUL prediction. Table V lists the statistics of the RUL prediction. When the RUL prediction is implemented at 285 cycles, the failure prediction error based on the LSTM RNN is 48 cycles and the STD is 24 cycles. The SVM predicted a delayed failure at 285 cycles with the error and STD being, respectively, 30 cycles and 9 cycles. The 95% confidence bound is 589–623 cycles, which is still after the real failure. When the RUL is predicted at 399 cycles, the RUL prediction error and STD based on the LSTM RNN are, respectively, 16 cycles and 1 cycle shorter than that based on the SVM. For the SimRNN, the 95% confidence bound of the failure prediction are both significantly distant from the real value.

The capacity of cell 3 dropped below the threshold at 578 cycles. Fig. 9 shows the failure prediction results of cell 3 based on the LSTM RNN. It is observed based on the RUL PDFs that, when more data is used for training, both the accuracy and precision of the RUL prediction are improved, indicating a more accurate and precise learning of long-term dependencies.

TABLE VI
RUL PREDICTION RESULTS FOR CELL 3

| Method | Starting Cycle | Error | STD | 95% Confidence Bound | Training Time (s) |
|---|---|---|---|---|---|
| LSTM RNN | 289 | 35 | 32 | [499, 628] | 21.53 |
| | 404 | 19 | 5 | [550, 570] | 28.98 |
| SVM | 289 | 40 | 8 | [522, 554] | 0.002 |
| | 404 | 23 | 11 | [533, 577] | 0.002 |
| SimRNN | 289 | 181 | 15 | [370, 433] | 39.27 |
| | 404 | 72 | 22 | [465, 558] | 55.36 |

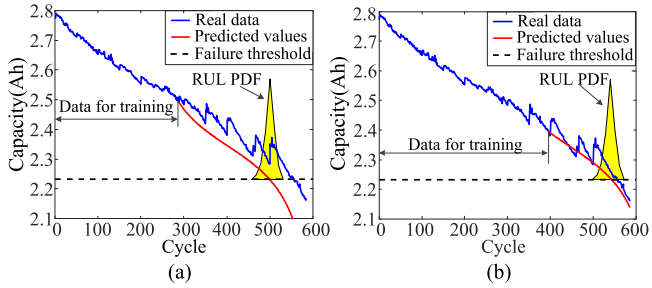


Fig. 10. Prediction results of RUL for cell 4: (a) prediction results at 278 cycles based on LSTM RNN; (b) prediction results at 389 cycles based on LSTM RNN.

Table VI shows that the RUL prediction error based on the LSTM RNN is reduced to 19 cycles from 35 cycles, and the STD is reduced to 5 cycles from 32 cycles. When the SVM is used to predict the cell's RUL at 289 cycles, the prediction error is 40 cycles, which is 5 cycles longer than that based on the LSTM RNN. But the STD is only 8 cycles, which is 24 cycles shorter than that based on the LSTM RNN. These comparison results indicate that the degradation model constructed based on the SVM is more precise than that of the LSTM RNN, but the modeling accuracy of the SVM is lower. When the failure prognostic is conducted at 404 cycles, the LSTM RNN predicts the failure with both higher accuracy and precision than the SVM. In this case, the RUL prediction error and STD using the LSTM RNN are, respectively, 4 cycles and 6 cycles shorter than those using the SVM. The SimRNN still learns the long-term dependencies least accurately, and the RUL prediction errors are 185 cycles and 72 cycles when the prediction starts at 289 cycles and 404 cycles, respectively.

The capacity of cell 4 dropped below the threshold at 557 cycles. Fig. 10 shows the RUL prediction results of cell 4. From Fig. 10(a) and (b), it is observed that when more data is used for training, the predicted trajectories track the capacity degradation with much higher accuracy. Table VII lists the statistics of the failure prediction. When the RUL prediction starts at 278 cycles, the errors of the predicted failure based on the LSTM RNN and the SVM are 58 cycles and −47 cycles, respectively, which indicates that the SVM fails at an advanced failure report. When the failure is predicted at 389 cycles, the prediction error based on the LSTM RNN is 14 cycles, which is 1 cycle shorter than that using the SVM, and the STD is 4 cycles shorter than that using the SVM. The SimRNN fails at capturing the long-term dependencies when the prediction starts at 278 cycles, and the

TABLE VII
RUL PREDICTION RESULTS FOR CELL 4

| Method | Starting Cycle | Error | STD | 95% Confidence Bound | Training Time (s) |
|---|---|---|---|---|---|
| LSTM RNN | 278 | 58 | 5 | [490, 509] | 22.67 |
| | 389 | 14 | 5 | [534, 553] | 28.55 |
| SVM | 278 | −47 | 16 | [575, 638] | 0.008 |
| | 389 | 15 | 9 | [526, 562] | 0.013 |
| SimRNN | 278 | – | – | – | – |
| | 389 | 88 | 11 | [448, 491] | 63.32 |

TABLE VIII
RUL PREDICTION RESULTS BASED ON OFFLINE DATA FOR CELL 1 AND CELL 3

| Label | Method | Error | STD | 95% Confidence Bound | Training Time (s) |
|---|---|---|---|---|---|
| Cell 1 | LSTM RNN | –19 | 24 | [477, 573] | 3.23 |
| | PF | 52 | 25 | [402, 506] | 0.58 |
| | SimRNN | 110 | 16 | [364, 428] | 9.62 |
| Cell 3 | LSTM RNN | 40 | 22 | [494, 582] | 4.49 |
| | PF | 58 | 27 | [474, 566] | 0.61 |
| | SimRNN | 93 | 14 | [457, 513] | 10.11 |

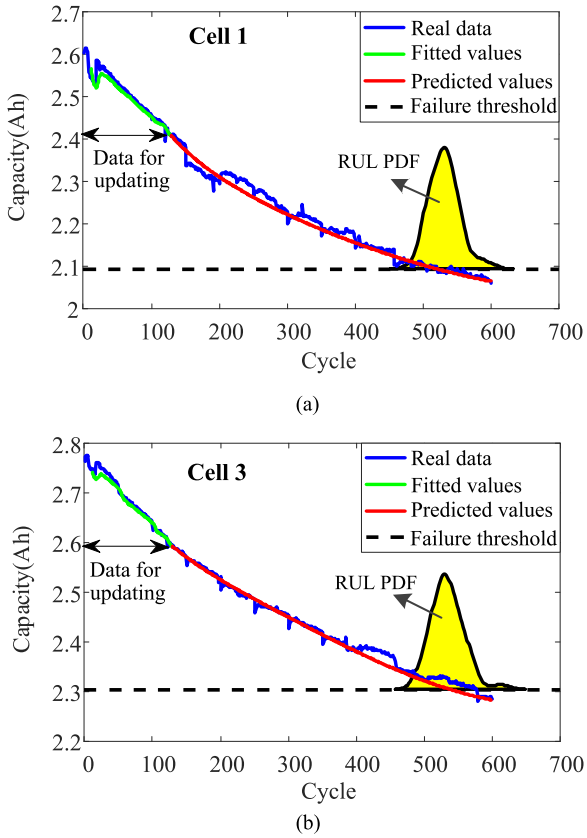


Fig. 11. LSTM RNN-based RUL prediction results based on some offline training data of: (a) cell 1; (b) cell 3.

RUL prediction error is 74 cycles longer than that of the LSTM RNN at 389 cycles.

When 50% of the complete data is available, the LSTM RNN can predict accurate RULs for batteries, and more accurate RULs can be predicted as more online data is collected. Generally, the LSTM RNN predicts more accurate and precise RULs of lithium-ion batteries under different temperatures and current rates than the SVM and the SimRNN, indicating that the LSTM RNN learns the long-term dependencies of the battery degradations most efficiently.

### C. RUL Prediction With Some Offline Training Data

When some offline training data is available, the required online updating data for an accurate and precise RUL prediction can be reduced. Fig. 11 shows the RUL prediction results of cells 1 and 3, which are, respectively, dependent on the offline training data of cells A and cell B. For cells 1 or 3, the training data of cells A or B under the same experimental condition is used to train the LSTM RNN, and then the trained network, whose parameters are further updated based on the available online capacity data of cells 1 or 3, is used to predict the cell's future capacities and RUL. The RUL predictions of cells 1 and 3 are both started at 120 cycles, which amounts to 20%–25% of the complete data of cells 1 or 3. The trained LSTM RNN based on offline data is only further trained by 3–10 epochs based on online data for accurate RUL predictions. Table VIII lists the RUL prediction results of cells 1 and 3 dependent on offline training data.

Fig. 11(a) shows the RUL prediction results of cell 1, where the fitted values and predicted values are both close to the real data. The predicted failure is 525 cycles, which is 19 cycles longer than the real value, and the STD of the RUL predictions is 24 cycles. The 95% confidence bound is [477, 573] cycles, which contains the predicted failure value. The online training time of the LSTM RNN listed in Table VIII is only 3.23 s owing to the small number of training epochs. The PF-based method was also applied to predict the cell's RUL as in Ref. [9]. The two-term exponential model was used to fit the offline training data, which was then used to predict the online cell's RUL combined with the PF algorithm. The RUL prediction error is 52 cycles, the absolute value of which is 33 cycles longer than that using the LSTM RNN. The STD based on the PF is 25 cycles, which is only 1 cycle longer than that using the LSTM RNN. The online training time of the PF method is 0.58 s, which is about 2 s less than that based on the LSTM RNN. The RUL prediction error based on the SimRNN is 110 cycles, the absolute value of which is 91 cycles longer than that of the LSTM RNN method, and the online training time of the SimRNN is about 6 s longer than that of the LSTM RNN owing to the more epochs required by the SimRNN for an accurate RUL prediction. Therefore, the LSTM RNN method predicts the most accurate and precise RUL based on only 120 cycles' online data when some offline training data is available.

Fig. 11(b) shows the LSTM RNN-based RUL prediction results of cell 3, and the prediction statistics are listed in Table VIII. In Fig. 11(b), the fitted values and predicted values are both close to the real data, and the RUL prediction error is 40 cycles with an STD being 22 cycles. The online training time is 4.49 s. The PF method predicts an RUL with the error and STD being, respectively, 58 and 27 cycles, which are, respectively, 18 and 5 cycles longer than those based on the LSTM RNN method.

The RUL prediction error of the SimRNN method is 93 cycles, which is 50 cycles longer than that of the LSTM RNN.

When some offline data is available, the required online data for an accurate RUL prediction using the LSTM RNN method is reduced to 20%–25% of the complete data of the online cell, which is 20%–50% smaller than that required by the developed methods in Refs. [9]–[13] for an accurate RUL prediction dependent on offline training data. It is noted that, in this case, the offline data should be collected under similar working conditions to those of online cells. For example, the urban dynamometer driving schedule (UDDS) test [2] can be conducted to collect effective offline training data, and based on which, the developed LSTM-RNN can predict the battery RUL for EVs at an early life stage.

## V. Conclusions

As a key approach of prognostics and health management (PHM), remaining useful life (RUL) prediction can provide the probable failure time of lithium-ion batteries in advance. The challenge of RUL prediction for lithium-ion batteries lies in how to accurately learn the long-term dependencies over hundreds of cycles based on limited degradation data.

This paper used the long short-term memory recurrent neural network (LSTM RNN) to synthesize a data-driven battery RUL predictor. The resilient mean square back-propagation (RMSprop) technique, which is specifically designed for mini-batch training, was used to train the constructed neural network. A dropout technique was applied to address overfitting and improve the prediction ability of the LSTM RNN, and Monte Carlo (MC) simulation was used to generate the RUL prediction uncertainties.

The constructed LSTM RNN can be used to predict the battery RUL independent of offline training data. This prediction capability can be used when the offline training data is unavailable or inapplicable. Experimental results based on cells 1 to 4 under different temperatures, including 25 °C and 40 °C, and different current rates, including 1 C, 2 C, and 3.5 C, indicate that the LSTM RNN generally predicts more accurate and precise RULs of lithium-ion batteries than the support vector machine (SVM) and simple recurrent neural network (SimRNN). When some limited offline training data is available, the required online data for accurate RUL prediction amounts to 20%–25% of the online cell, which is 20%–50% less than that required by the developed methods in the literature [9]–[13]. The LSTM RNN also generally predicts more accurate and precise RULs than the PF and SimRNN.

## References


[1] Y. Z. Zhang, R. Xiong, H. W He, and W. X. Shen, "Lithium-ion battery pack state of charge and state of energy estimation algorithms using a hardware-in-the-loop validation," *IEEE Trans. Power Electron.*, vol. 32, no. 6, pp. 4421–4431, Jun. 2017.

[2] R. Xiong, Y. Zhang, H. He, X. Zhou, and M. Pecht, "A double-scale, particle-filtering, energy state prediction algorithm for lithium-ion batteries," *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1526–1538, Feb. 2018.

[3] J. L. Zhang and J. Lee, "A review on prognostics and health monitoring of Li-ion battery," *J. Power Sources*, vol. 196, no. 15, pp. 6007–6014, Aug. 2011.

[4] H. M. Elattar, H. K. Elminir, and A. M. Riad, "Prognostics: A literature review," *Complex Intell. Syst.*, vol. 2, no. 2, pp. 125–154, Jun. 2016.

[5] W. Waag, C. Fleischer, and D. U. Sauer, "Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles," *J. Power Sources*, vol. 258, pp. 321–339, Jul. 2014.

[6] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[7] D. Wang, Q. Miao, and M. Pecht, "Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model," *J. Power Sources*, vol. 239, pp. 253–264, Oct. 2013.

[8] J. Li, C. Lyu, L. Wang, L. Zhang, and C. Li, "Remaining capacity estimation of Li-ion batteries based on temperature sample entropy and particle filter," *J. Power Sources*, vol. 268, pp. 895–903, Dec. 2014.

[9] W. He, N. Williard, M. Osterman, and M. Pecht, "Prognostics of lithium-ion batteries based on Dempster–Shafer theory and the Bayesian Monte Carlo method," *J. Power Sources*, vol. 196, no. 23, pp. 10314–10321, Dec. 2011.

[10] Y. Xing, E. W. Ma, K. L. Tsui, and M. Pecht, "An ensemble model for predicting the remaining useful performance of lithium-ion batteries," *Microelectron. Reliab.*, vol. 53, no. 6, pp. 811–820, Jun. 2013.

[11] C. Lyu, Q. Lai, T. Ge, H. Yu, L. Wang, and N. Ma, "A lead-acid battery's remaining useful life prediction by using electrochemical model in the particle filtering framework," *Energy*, vol. 120, pp. 975–984, Feb. 2017.

[12] Z. Liu, G. Sun, S. Bu, J. Han, X. Tang, and M. Pecht, "Particle learning framework for estimating the remaining useful life of lithium-ion batteries," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 2, pp. 280–293, Feb. 2017.

[13] X. Su, S. Wang, M. Pecht, L. Zhao, and Z. Ye, "Interacting multiple model particle filter for prognostics of lithium-ion batteries," *Microelectron. Reliab.*, vol. 70, pp. 59–69, Mar. 2017.

[14] A. Nuhic, T. Terzimehic, T. Soczka-Guth, M. Buchholz, and K. Dietmayer, "Health diagnosis and remaining useful life prognostics of lithium-ion batteries using data-driven methods," *J. Power Sources*, vol. 239, pp. 680–688, Oct. 2013.

[15] M. A. Patil *et al.*, "A novel multistage Support Vector Machine based approach for Li ion battery remaining useful life estimation," *Appl. Energy*, vol. 159, pp. 285–297, Dec. 2015.

[16] Y. Zhou, M. Huang, Y. Chen, and Y. Tao, "A novel health indicator for on-line lithium-ion batteries remaining useful life prediction," *J. Power Sources*, vol. 321, pp. 1–10, Jul. 2016.

[17] J. Liu, A. Saxena, K. Goebel, B. Saha, and W. Wang, "An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries," in *Proc. Annu. Conf. Prognostics Health Manage. Soc.*, 2010, pp. 1–9.

[18] M. Rezvani, M. AbuAli, S. Lee, J. Lee, and J. Ni, "A comparative analysis of techniques for electric vehicle battery prognostics and health management (PHM)," SAE Tech. Paper 2011-01-2247, 2011, doi: 10.4271/2011-01-2247.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[20] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.

[21] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," Unpublished paper, 2015. [Online]. Available: https://arxiv.org/abs/1506.00019

[22] S. Yoon, H. Yun, Y. Kim, G. T. Park, and K. Jung, "Efficient transfer learning schemes for personalized language modeling using recurrent neural network," Unpublished paper, 2017. [Online]. Available: https://arxiv.org/abs/1701.03578

[23] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[24] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3156–3164.

[25] A. Graves, "Generating sequences with recurrent neural networks," Unpublished paper, 2013. [Online]. Available: http://arxiv.org/abs/1308.0850

[26] T. Guo, Z. Xu, X. Yao, H. Chen, K. Aberer, and K. Funaya, "Robust online time series prediction with recurrent neural networks," in *Proc. 2016 IEEE Int. Conf. Data Sci. Adv. Anal.*, Oct. 2016, pp. 816–825.

[27] X. Y. Qu, X. N. Kang, C. Zhang, S. Jiang, and X. D. Ma, "Short-term prediction of wind power based on deep long short-term memory," in *Proc. 2016 IEEE PES Asia-Pac. Power Energy Eng. Conf.*, Oct. 2016, pp. 1148–1152.

[28] Y. Duan, Y. Lv, and F. Y. Wang, "Travel time prediction with LSTM neural network," in *Proc. 2016 IEEE 19th Int. Conf. Intell. Transp. Syst.*, Nov. 2016, pp. 1053–1058.

[29] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA, Neural Netw. Mach. Learn.*, vol. 4, no. 2, pp. 26–31, 2012.
[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
[31] J. Wang *et al.*, "Degradation of lithium ion batteries employing graphite negatives and nickel–cobalt–manganese oxide+ spinel manganese oxide positives: Part 1, Aging mechanisms and life estimation," *J. Power Sources*, vol. 269, pp. 937–948, Dec. 2014.
[32] X. Jin *et al.*, "Physically-based reduced-order capacity loss model for graphite anodes in Li-ion battery cells," *J. Power Sources*, vol. 342, pp. 750–761, Feb. 2017.
[33] M. Schimpe, M. E. von Kuepach, M. Naumann, H. C. Hesse, K. Smith, and A. Jossen, "Comprehensive modeling of temperature-dependent degradation mechanisms in lithium iron phosphate batteries," *ECS Trans.*, vol. 80, no. 10, pp. 147–170, 2017.
[34] S. Ruder, "An overview of gradient descent optimization algorithms," Unpublished paper, 2017. [Online]. Available: https://arxiv.org/abs/1609.04747
[35] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. IEEE Int. Conf., Neural Netw.*, 1993, pp. 586–591.
[36] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. Ser. B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
[37] A. N. Tikhonov, "On the stability of inverse problems," *Doklady Akademii Nauk SSSR*, vol. 39, no. 5, pp. 195–198, 1943.
[38] M. Beer and P. D. Spanos, "Neural network based Monte Carlo simulation of random processes," in *Proc. 9th Int. Conf. Struct. Saf. Reliab.*, 2005, pp. 2179–2186.

**Yongzhi Zhang** (S'15) received the B.S. degree in vehicle engineering from Chongqing University, Chongqing, China, in 2013. He is currently working toward the Ph.D. degree in mechanical engineering at the National Engineering Laboratory for Electric Vehicles, Beijing Institute of Technology, Beijing, China.

He is currently a Research Scholar with the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD, USA. His research interests include prognostics and health management of lithium-ion batteries.

Mr. Zhang was the recipient of the Best Paper Awards in the International Symposium on Electric Vehicles, Sweden, 2017.

**Rui Xiong** (S'12–M'14–SM'16) received the M.Sc. degree in vehicle engineering and the Ph.D. degree in mechanical engineering from Beijing Institute of Technology, Beijing, China, in 2010 and 2014, respectively. He conducted scientific research as a joint Ph.D. student in the DOE GATE Center for Electric Drive Transportation at the University of Michigan, Dearborn, MI, USA, between 2012 and 2014.

Since 2014, he has been an Associate Professor in the Department of Vehicle Engineering, School of Mechanical Engineering, Beijing Institute of Technology, Beijing, China. Since 2017, he has been an Adjunct Professor in the Faculty of Science, Engineering and Technology, Swinburne University of Technology, Melbourne, Vic., Australia. He has conducted extensive research and authored more than 100 peer-reviewed articles. He holds eight patents. His research interests mainly include electrical/hybrid vehicles, energy storage, and battery management system.

Dr. Xiong received the Excellent Doctoral Dissertation from Beijing Institute of Technology in 2014, the first prize of Chinese Automobile Industry Science and Technology Progress Award in October 2015 and the second prize of National Defense Technology Invention Award in December 2016. He received Best Paper Awards from the journal *Energies* and several International conferences. He is serving as the Associate Editor of IEEE ACCESS, Editorial Board of the *Applied Energy, Energies, Sustainability* and *Batteries*, and Guest editor of the *Journal of Cleaner Production*. He was the conference chair of the 2017 International Symposium on Electric Vehicles held in Stockholm, Sweden.

**Hongwen He** (M'03–SM'12) received the Ms.E. degree from the Jilin University of Technology, Changchun, China, in 2000 and the Ph.D. degree from the Beijing Institute of Technology, Beijing, China, in 2003, both in vehicle engineering.

He is currently a Professor with the National Engineering Laboratory for Electric Vehicles, School of Mechanical Engineering, Beijing Institute of Technology. He has published more than 100 papers and holds 20 patents. His research interests include power battery modeling and simulation on electric vehicles, design, and control theory of the hybrid power trains.

Dr. He was the recipient of the second prize of Chinese National Science and Technology Award for the work on the development of new-energy electric bus powertrains in 2015, the first prize of Henan Science and Technology Award for the work on the development of hybrid bus powertrain in 2013, the first prize of Henan Science and Technology Award for the work on the development of battery electric bus powertrain in 2014 and the second prize of National Defense Technology Innovation Award for the work on the development of hybrid powertrains in off road vehicles in 2016. He received Best Paper Awards from the journal *Energies* and International conference. He is serving as the Editorial Board of the *Energies*. He was the conference chair of the 2017 International conference on Energy, Ecology and Environment held in Stockholm, Sweden.

**Michael G. Pecht** (S'78–M'83–SM'90–F'92) received the B.S. degree in physics in 1976, the M.S. degree in electrical engineering in 1978, and the M.S. and Ph.D. degrees in engineering mechanics from the University of Wisconsin-Madison, Madison, WI, USA, in 1979 and 1982 respectively.

He is the Founder and the Director of the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD, USA, which is funded by more than 150 of the world's leading electronics companies at more than US$6M/year. He is also a Chair Professor of mechanical engineering and a Professor of applied mathematics, statistics, and scientific computation with the University of Maryland. He has written more than 20 books, 400 technical articles, and has eight patents.

Dr. Pecht is a Professional Engineer, a Fellow of the American Society of Mechanical Engineers, a Fellow of the Society of Automotive Engineers, and a Fellow of the International Microelectronics Assembly and Packaging Society. He is the Editor-in-Chief for the IEEE ACCESS, and served as the Chief Editor for the IEEE TRANSACTIONS ON RELIABILITY for nine years and the Chief Editor for the *Microelectronics Reliability* for 16 years. He has also served on three U.S. National Academy of Science studies, two U.S. Congressional investigations in automotive safety, and as an expert to the U.S. Food and Drug Administration.