sensors

MDPI

Article

Remaining Useful Life Prediction of Lithium-Ion Batteries Using Neural Networks with Adaptive Bayesian Learning

Karkulali Pugalenthi [1,*], Hyunseok Park [2], Shaista Hussain [3] and Nagarajan Raghavan [1]

[1] Engineering Product Development Pillar, Singapore University of Technology & Design, 8 Somapah Road, Singapore 487372, Singapore; nagarajan@sutd.edu.sg
[2] Department of Information Systems, Hanyang University, Seoul 133791, Korea; hp@hanyang.ac.kr
[3] Computational Intelligence Group, A*STAR Institute of High-Performance Computing (IHPC), Singapore 138632, Singapore; hussains@ihpc.a-star.edu.sg
* Correspondence: karkulali@mymail.sutd.edu.sg

Abstract: With smart electronic devices delving deeper into our everyday lives, predictive maintenance solutions are gaining more traction in the electronic manufacturing industry. It is imperative for the manufacturers to identify potential failures and predict the system/device's remaining useful life (RUL). Although data-driven models are commonly used for prognostic applications, they are limited by the necessity of large training datasets and also the optimization algorithms used in such methods run into local minima problems. In order to overcome these drawbacks, we train a Neural Network with Bayesian inference. In this work, we use Neural Networks (NN) as the prediction model and an adaptive Bayesian learning approach to estimate the RUL of electronic devices. The proposed prognostic approach functions in two stages—weight regularization using adaptive Bayesian learning and prognosis using NN. A Bayesian framework (particle filter algorithm) is adopted in the first stage to estimate the network parameters (weights and bias) using the NN prediction model as the state transition function. However, using a higher number of hidden neurons in the NN prediction model leads to particle weight decay in the Bayesian framework. To overcome the weight decay issues, we propose particle roughening as a weight regularization method in the Bayesian framework wherein a small Gaussian jitter is added to the decaying particles. Additionally, weight regularization was also performed by adopting conventional resampling strategies to evaluate the efficiency and robustness of the proposed approach and to reduce optimization problems commonly encountered in NN models. In the second stage, the estimated distributions of network parameters were fed into the NN prediction model to predict the RUL of the device. The lithium-ion battery capacity degradation data (CALCE/NASA) were used to test the proposed method, and RMSE values and execution time were used as metrics to evaluate the performance.

Keywords: lithium-ion batteries; particle filters; neural networks; remaining useful life; prognostics

Citation: Pugalenthi, K.; Park, H.; Hussain, S.; Raghavan, N. Remaining Useful Life Prediction of Lithium-Ion Batteries Using Neural Networks with Adaptive Bayesian Learning. *Sensors* **2022**, *22*, 3803. https://doi.org/10.3390/s22103803

Academic Editor: Hossam A. Gabbar

Received: 20 March 2022
Accepted: 10 May 2022
Published: 17 May 2022

1. Introduction

Maintenance of electronic devices, physical equipment and systems is imperative for ensuring successful functioning, minimal downtime, reduced unprecedented maintenance costs, and prolonging the life of the device/system. Maintenance strategies are broadly categorized as preventive and predictive maintenance strategies. Although preventive maintenance adopts a conventional approach of regular scheduled maintenance protocols, predictive maintenance strategies are preemptive methods. With the availability of affordable sensor systems and advances in machine learning algorithms, predictive maintenance approaches promise economic benefits both for the manufacturers and end-users.

The goal of predictive maintenance strategies is to predict the remaining useful life (RUL) of the device/system. RUL prediction of devices can be realized through two methods, namely, model-based methods and data-driven methods. Model-based methods

employ physical or statistical models which best capture the degradation of the desired system. However, obtaining such models requires extensive experimental work and an in-depth understanding of the underlying failure mechanisms. These models are effective until the system is upgraded or changed. Commonly used model-based prognostic approaches are Kalman filters (KF) [1,2], Extended Kalman filters (EKF) [3,4], and Particle filters (PF) [5,6]. These methods either use empirical models or incremental state-space representations of the underlying governing partial differential equations (PDE). On the other hand, data-driven methods use machine learning based approaches to characterize the failure of the desired device/system. Machine learning methods use pattern recognition to learn features from raw sensor data and identify failure patterns. Machine learning based data-driven methods successfully used for developing prognostic algorithms include regression methods [7,8], support vector machine (SVM) [9,10], relevance vector machine (RVM) [11,12], Bayesian networks (BN) [13,14], hidden Markov model (HMM) [15], and artificial neural networks (ANN) [16,17]. Among the different machine learning methods, neural network (NN) based methods are preferred by researchers due to their versatility and adaptive parameter optimization capabilities. To elaborate, NN based methods can address uncertainties in the system, such as measurement uncertainties introduced by noisy sensor data and prediction model uncertainties caused by variations in operating conditions. However, shallow neural network architectures, such as feedforward neural networks and radial bias function neural networks, face challenges in predicting the RUL with good confidence due to its dependency on large training datasets. Additionally, the complexity of the network architecture increases for highly nonlinear systems, which severely affects parameter optimization and further leads to overfitting issues.

Several deep learning models are proposed in the literature to address optimization and overfitting issues, such as convoluted neural networks (CNN) [18,19], recurrent neural networks (RNN) [20,21], deep belief networks (DBN) [22], and long short-term memory networks (LSTM) [23,24]. Although deep learning models are efficient methods for RUL prediction due to their ability to learn features on-the-fly, they are computationally expensive. For instance, combining convoluted features across all time steps in CNN is time-consuming and inhibits its application in processing large-scale time-series data. Similarly, RNN based methods are sequential wherein long-term information has to sequentially fed through all cells before being processed, subsequently causing vanishing gradient issues. LSTM based methods, on the other hand, require a large amount of memory bandwidth for processing large-scale time-series data.

Additionally, there have been several attempts made to develop hybrid prognostic algorithms combining different model-based and data-driven methods to overcome the limitations arising due to dependency on accurate physical models and large amount of failure data. To name a few, Ma et al. [25] developed a hybrid prognostic model combining CNN and LSTM. The authors used CALCE and NASA datasets to evaluate their performance and used nearly 50% of data from each battery for the purpose of training. The training dataset size was optimized by using false nearest neighbor method. Even though the authors obtained good prediction accuracy, the RUL prediction can be performed only at mid/late degradation cycles which limits their applicability to safety-critical devices wherein the failures even in the early degradation cycles can lead to catastrophic results. Similarly, Wu et al. [26] developed a hybrid prognostic approach combining NN and particle filters wherein they had used the NN model equation as the system degradation model in the PF algorithm. In this case, the authors had assumed that they have the run-to-failure data for all the batteries available and used curve fitting results of each battery as the initial parameter guess for the PF algorithm. This limits the generalization capability of their proposed approach. This calls for the need to optimize shallow neural network models by regularizing network parameters and complexity to provide an efficient and computationally inexpensive framework for prognostic studies appropriate for real-time applications.

2. Training Algorithms for Neural Networks

Artificial neural network models have been successfully applied in prognostic studies [16,17] due to their ability to model non-linearities in degradation data and generalization capabilities. However, choosing an appropriate NN model is an arduous task as its optimization capabilities rely heavily on network architecture, training algorithms, and initial values of connection weights and bias. The NN model learns patterns from training data during the training phase and uses that information to produce the desired output. The NN model parameters (weights and bias) are modified to minimize the error between the predicted value and the desired output. A suitable backpropagation algorithm does the process of adjusting the network parameters for minimizing the error value. Thus, choosing an efficient training algorithm is imperative as it directly affects the performance of the network model. The commonly used NN training algorithms are gradient descent and Levenberg–Marquardt algorithms. However, the major drawback of using backpropagation-based training algorithms is that if the error values are multimodal, the algorithm becomes trapped at local minima. Additionally, improper/random initial parameter configuration aids the algorithm to settle at incorrect local minima.

To overcome the disadvantages of backpropagation algorithms, several evolutionary algorithms have been proposed in the literature. Gudise et al. [27] proposed the particle swarm optimization (PSO) algorithm for training a simple feedforward NN model. The authors compared the performance of PSO with gradient descent (GD) algorithm. The results clearly showed that the convergence rate of PSO was much better than GD. Similarly, Karaboga et al. [28] proposed an artificial bee colony (ABC) algorithm to train a feedforward NN model. The performance of the algorithm was compared with genetic algorithm (GA) and GD. The results proved that ABC algorithm does not become trapped at local minima. However, the computational cost of the proposed method was considerably high as it took a very high number of epochs for convergence. Additionally, in all of the algorithms mentioned above, the initial parameter values were deterministic, so when applied to time-series prediction models, they would only yield a single-point estimate for the future values.

This work proposes using a Bayesian inference-based training algorithm to identify the NN model parameters and provide probabilistic RUL estimates over a single point estimate. Here, we resort to using particle filters for training a feedforward MLP model wherein the NN model equation is used in the PF algorithm as the incremental state transition function for predicting the system's future state. The PF algorithm performs state estimation to deduce the model parameters best representing the training dataset. Further, the PF estimated model parameters are used to configure the initial connection weight and bias values of the MLP network for other devices being monitored to predict the system's future state and subsequently for RUL estimation.

Usage of PF for NN training is promising due to its ability to easily capture the highly nonlinear and non-Gaussian statistics by using different weights for different samples. It is to be noted that the particle filter weights are different from the NN weight values. However, the weighted sampling approach of PF leads to a particle collapse wherein the particle weights accumulate over a small fraction of particles known as particle degeneracy. If left unaddressed, particle degeneracy eventually leads to particle impoverishment wherein the particle weights are accumulated on a single particle, and the rest of the particles have zero weights. Particle impoverishment affects the performance of the NN model in the prognosis stage as the network model becomes configured with incorrect initial parameter weight and bias values. Although particle degeneracy can be controlled by adopting suitable resampling strategies, it is challenging to overcome particle impoverishment. In order to reduce particle impoverishment, we proposed *particle roughening* as a weight regularization method.

The rest of the paper is organized as follows. Section 3 describes the two different lithium-ion battery capacity degradation datasets used in this work. Section 4 describes the methodology used in this work—description of multilayer perceptron, particle filters, and

the proposed RUL estimation framework. Section 5 summarizes the results and discussion, and conclusions and future work are discussed in Section 6.

3. Degradation Datasets

Two different lithium-ion battery capacity degradation datasets from CALCE and NASA repository are used in this work for testing the performance of the proposed method [29,30].

3.1. CALCE Dataset

Accelerated aging tests were performed for a set of prismatic cells (CS) with $LiCoO_2$ cathode of 1.1 Ah capacity rating. The CS cells were charged and discharged repeatedly till the cells reached their End-of-Life (EoL). The cells were subjected to a charging profile using constant current/constant voltage protocol. The current rate was maintained at 1C till the voltage reached 4.2 V.

The charge was sustained at 4.2 V till the charging current dropped to 0.05A. The failure threshold for the CS cells was set to be at 0.88 Ah. The capacity degradation curves for three CS cells from CALCE repository are shown in Figure 1a.

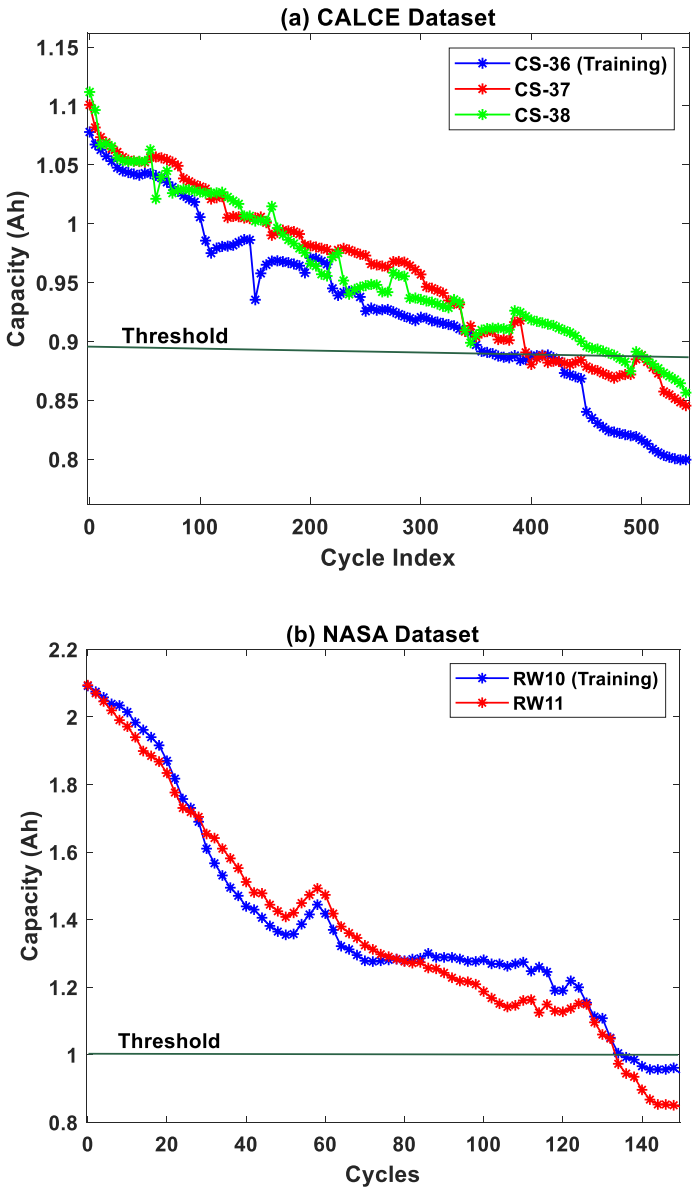

Figure 1. The battery capacity degradation datasets from the (**a**) CALCE and (**b**) NASA repositories.

3.2. NASA Dataset

The second dataset used in this work is from NASA Ames Prognostic Center of Excellence [26]. In this dataset, the $LiCoO_2$ cathode cells with a rated capacity of 2.1 Ah were used for generating the battery capacity degradation data. Unlike the CALCE dataset, the NASA batteries were tested under random discharge currents rather than constant discharge currents. The charge/discharge cycles termed Random Walk (RW) cycling was performed wherein the current profile for both charging and discharging were changed every 5 min with a current value randomly selected from 0–4 A. The randomized loading conditions were applied to generate degradation data to simulate more realistic operating conditions. The failure threshold was set to be at 1 Ah, and the battery capacity degradation curves are shown in Figure 1b.

4. Methodology

4.1. Multilayer Perceptron

The most popular NN architecture used is the feedforward multilayer perceptron (MLP). We resort to using two-layer MLP in this work. The NN architecture represents the degradation state of the device as a function of time. The input node and output node represent the input and output variables, respectively. The input node is fed with time in cycles (battery) as the input to obtain the corresponding degradation state—capacity (battery). The hidden layer connects the input and output node and is represented by nonlinear nodes called hidden neurons. Each layer transmits information forward through an activation function. A sigmoid activation function is used between the input and hidden layer, followed by a linear activation function between the hidden and output layer. The sigmoid activation function in terms of NN parameters (weights and bias) is expressed as:

$$h_i = \frac{1}{1 + e^{-(w_i^{(1)} \times k + b_i^{(1)})}} \tag{1}$$

where $w_i^{(1)}$ and $b_i^{(1)}$ is the input weight and bias corresponding to the ith hidden neuron, k is the time index, $i = 1, \ldots, M$ is the number of hidden neurons, and $h_i(.)$ is the tan-sigmoid activation function corresponding to the input layer. The predicted capacity/light output at the output node can be represented as

$$g((w,b),k) = f(\sum_{i=1}^{M} \left(h\left(\left(k \times w_i^{(1)} + b_i^{(1)} \right) w_i^{(2)} \right) + b_i^{(2)} \right) \tag{2}$$

where $g((w,b),k)$ is the output of the MLP network, $w_i^{(2)}$ and $b_i^{(2)}$ represent the weight and bias values at the hidden layer, and $h(.)$ is the tan-sigmoid activation function between the input and hidden layer as shown in Equation (1).

Choice of Network Architecture

One of the significant challenges with NN is the choice of network architecture. Choosing an appropriate number of hidden layers and neurons dramatically affects the performance of the NN model, especially while handling prognostic applications. During the training phase, the NN model parameters (w_i and b_i) are optimized to minimize the prediction error on the training patterns. Minimal error values denote a stable network, whereas high error values reflect overfitting. Using more hidden neurons might cause the network to overfit and lead to significant deviations in the predicted values. As there are no standard approaches available in the literature to deduce the optimum number of hidden neurons, we chose to use the Bayesian Information Criterion (*BIC*) to fix the suitable number of hidden neurons. *BIC* is a widely used metric for statistical model selection owing to its computational efficiency and simplicity [31,32]. *BIC* of a model can be evaluated as

$$BIC = -2 \times LL + \ln(N) \times q \tag{3}$$

where *LL* refers to the log-likelihood function of the model, *N* is the size of the training dataset, and *q* is the number of parameters to be estimated by the model. Additionally, the *BIC* penalizes a model based on the number of estimated parameters; hence, a complex model with higher neurons would be penalized and yield a poor (higher) *BIC* value. The model with the minimum *BIC* values is chosen as the best model for the purpose. The *BIC* analysis is shown in Figure 2. From Figure 2, it can be inferred that the NN model with three hidden neurons is suitable for the battery degradation dataset.

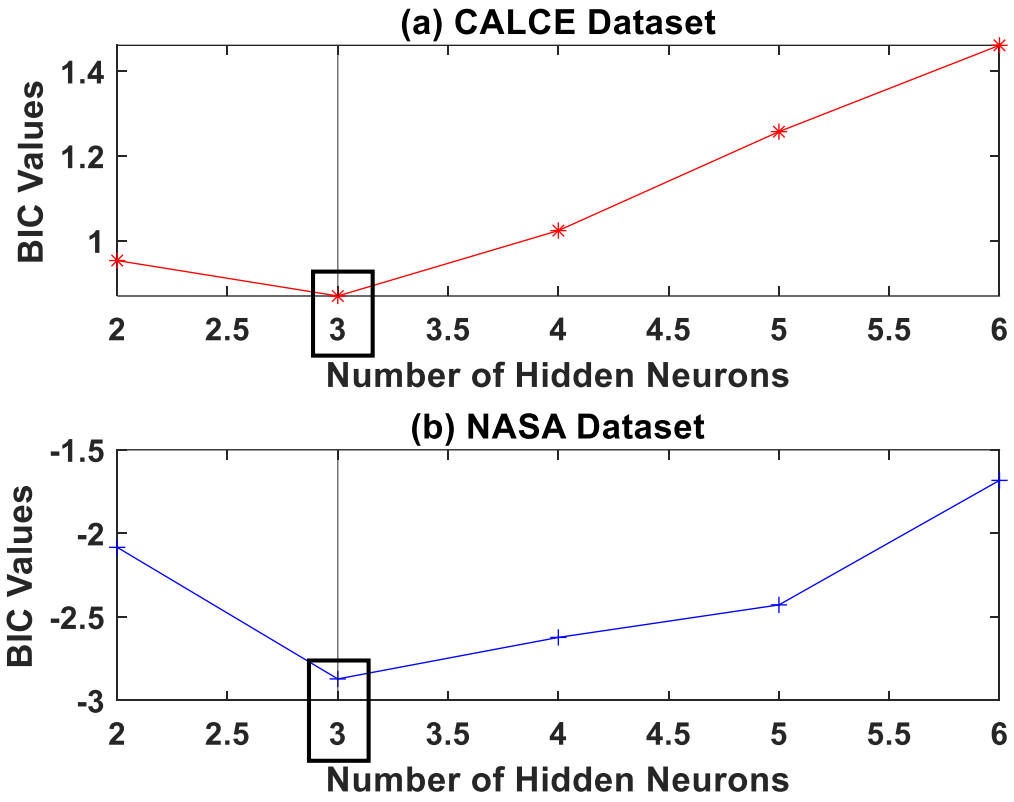

Figure 2. The *BIC* analysis for model selection for the two lithium-ion battery capacity degradation datasets—(**a**) CALCE and (**b**) NASA datasets.

4.2. Adaptive Bayesian Learning

4.2.1. Particle Filters

In this work, the PF algorithm is used as a state estimation method to deduce the optimum NN model parameter values (weights and bias). PF is a recursive Bayesian algorithm in which the system is represented by a state-space model comprising of a state transition model and measurement model as shown below:

$$x_k = f(x_{k-1}, \theta) + \omega_{k-1} \tag{4}$$

$$z_k = q(x_k, \theta) + \varepsilon_k \tag{5}$$

where x_k and x_{k-1} represent the current and previous degradation state of the system, z_k represents the available test data at kth time instant, k is the time index(cycles/hours), θ represents the vector with MLP model parameters (w_i and b_i), ω_{k-1} is the process noise, and ε_k is the measurement noise present in the system. $f(.)$ represents an incremental model of the state transition function, and $q(.)$ represents the measurement function. In this work, the measurement function used is $g((w,b),k)$ represented by Equation (2). The implementation of the PF algorithm is explained below:

- a. Particle Initialization—At $k = 1$ time instant, the initial prior distribution is populated based on prior knowledge of the system model parameters as shown in Figure 3a. In

this case, the curve fitting coefficients corresponding to one device from each device datasets are used. The initial prior probability density function $p(x_0)$ is then sampled into weighted particles.

- b. Particle Update—Whenever new measurement data are available for prediction, the weights of the particles are recursively updated based on the likelihood function, as shown below

$$p(z_k|x_k^i, \theta_k^i) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2}\frac{(z_k - q(x_k^i)^2}{\sigma^2}\right] \quad (6)$$

$$w_k^i = \frac{p(z_k|x_k^i, \theta_k^i)}{\sum_{i=1}^{j} p(z_k|x_k^i, \theta_k^i)} \quad (7)$$

where $p(z_k|x_k^i, \theta_k^i)$ is the likelihood function with θ_k^i representing the MLP network parameter at kth time instant, z_k is the available test data, x_k is the current system degradation state, and ${w_i}^k$ represents the particle weights. It is to be noted that the particle weights of the PF algorithm are different from the MLP network parameter weights.

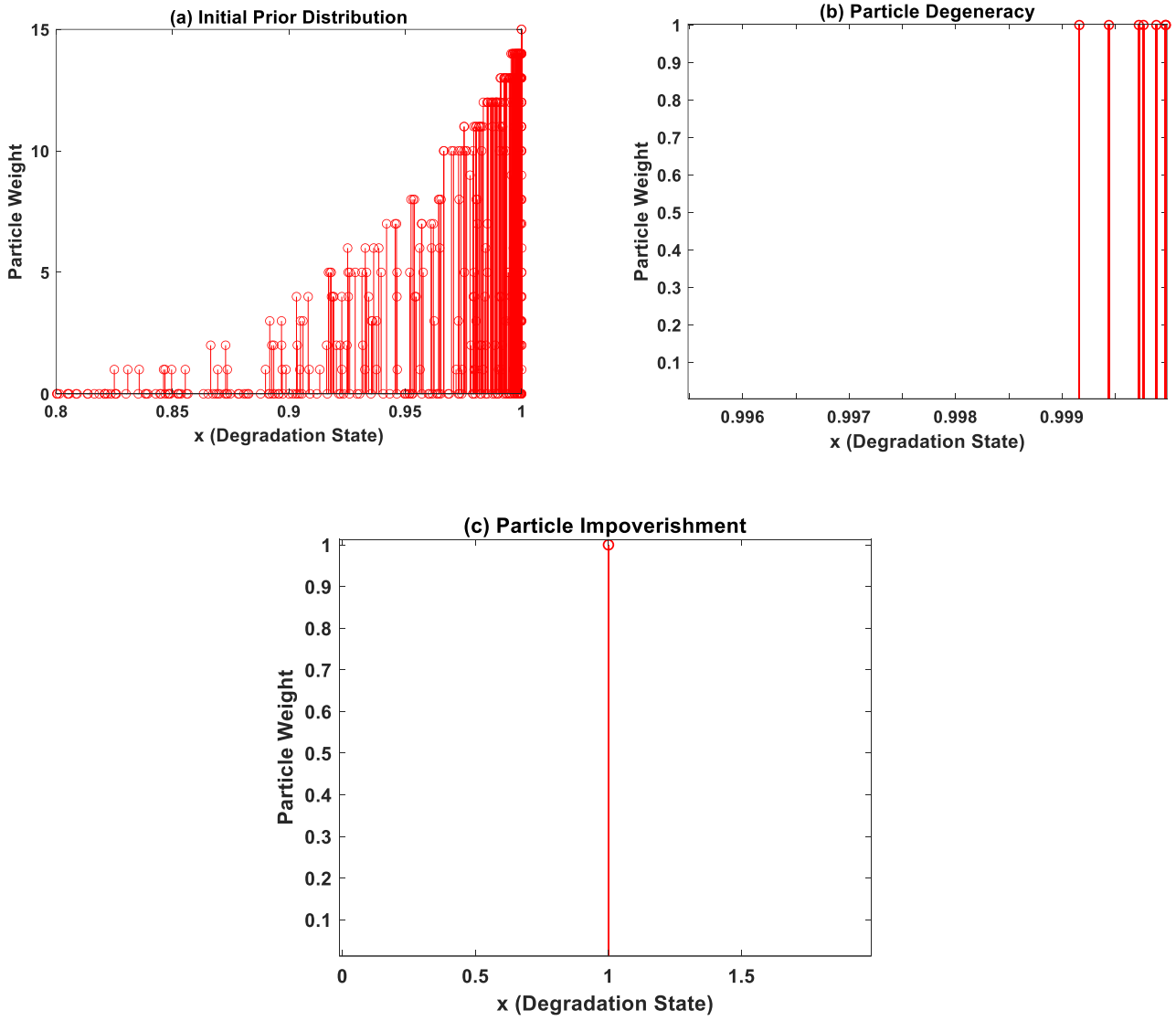

Figure 3. The particle weight distributions obtained while executing the PF algorithm: (**b**) weight distribution depicting particle degeneracy; and (**c**) weight distribution depicting particle impoverishment.

- c. Particle Resampling—During particle update, the weights accumulate over a few particles, and the rest of the particles carry negligible weights after a few iterations as shown in Figure 3b. In order to enhance diversity amongst the particles, the smaller weight particles are replaced with large weight particles by a process called particle resampling. Thus, the basic idea of resampling is to maintain all the samples/particles at the same weight.
- d. State Estimation—Finally, the degradation state of the system at $(k+1)$th time instant is evaluated based on the new set of weighted particles. The process is repeated for all available measurement data, and the posterior distribution at the current step becomes the prior distribution for the next step.

4.2.2. Weight Regularization Methods

Although a very good candidate for non-linear degradation prognosis in general with non-Gaussian noise components, the significant drawbacks of the PF algorithm are the particle degeneracy phenomenon followed by particle impoverishment. During particle updates, the particles with negligible weights are replaced by large weight particles. After few iterations, small weights will cease to exist and only large weight particles are present in the distribution. This phenomenon is termed particle degeneracy, and particle impoverishment is a severe case of particle degeneracy wherein all but one particle is eliminated during resampling, i.e., a single particle carries all the weights, as shown in Figure 3c. This dramatically affects the diversity of particles, thus constraining the evolution of model parameters and subsequently affects the accuracy of predictions.

Choosing an appropriate resampling strategy is one of the simplest methods to address particle degeneracy. In this work, we have applied three different resampling strategies based on our previous work in Ref. [33] to improve the adaptive Bayesian learning framework used in this study. The three resampling strategies considered are Multinomial Resampling (MR), Stratified Resampling (StR), and Systematic Resampling (SyR). The schematic representation for all three resampling strategies is depicted in Figure 4a. The weights of five particles after normalization are shown for illustration in Figure 4a. The length of the rectangles depicts the weights of the particles.

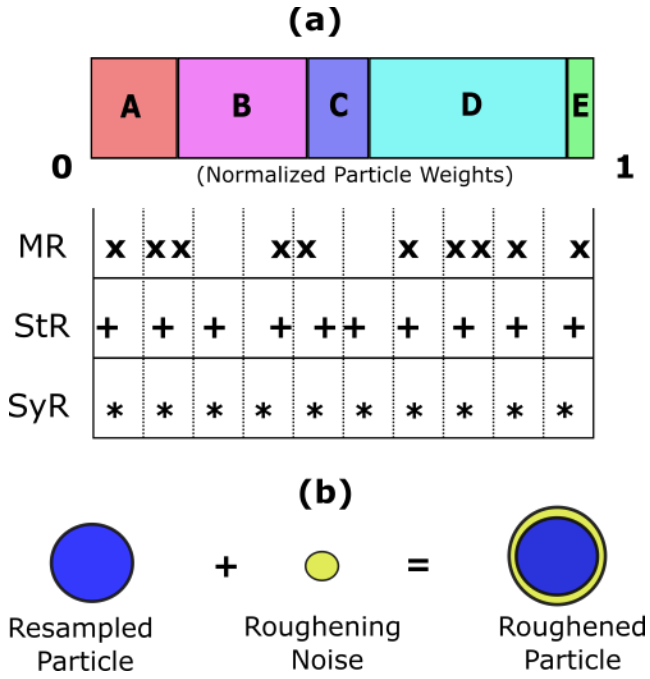

Figure 4. (**a**) The schematic representation of different resampling strategies used in this work—Multinomial Resampling (MR), Stratified Resampling (StR), and Systematic Resampling (SyR); and (**b**) the schematic representation of particle roughening weight regularization method.

MR is a random search approach where N independent particles are randomly selected from the particle distribution. StR divides the population into equal segments called strata, and particles are randomly selected from each stratum. SyR, on the other hand, is an extension of stratified resampling wherein one particle from each stratum from a fixed location is selected during resampling. Thus, SyR is a more deterministic approach compared to the other two resampling strategies.

Similarly, to address particle impoverishment, we use particle roughening as a weight regularization technique [34]. Although resampling strategies try to diversify the particle weight distribution, particle roughening, on the other hand, is a compensation technique. If particle impoverishment has occurred despite choosing an appropriate resampling strategy, one way to reduce it would be to redistribute the weights centered around one particle by jittering their values. For jittering, a small random noise (roughening noise) is added to the resampled particles. The roughening noise is small Gaussian jitter with zero mean and constant covariance. The covariance matrix is obtained from the standard deviation of the system degradation state as shown in Equation (8):

$$\sigma_K^i = (\sigma_{r1}, \sigma_{r2}) \tag{8}$$

$$\sigma_{r1} = \sqrt{-2\sigma_1^2 \times \ln\left(L_K^i\right)} \tag{9}$$

$$\sigma_{r2} = \sqrt{-2\sigma_2^2 \times \ln\left(L_K^i\right)} \tag{10}$$

where σ_K^i is the standard deviation of the Gaussian jitter with σ_{r1} being the standard deviation corresponding to the lower limit (σ_1^2) and σ_{r2} corresponding to the upper limit (σ_2^2) and L_K^i represents the likelihood function corresponding to Equation (6). The standard deviation limits used in this work are summarized in Table 1. When particle diversity improves, the distribution of model parameters is decentralized, and hence the prediction performance of the NN model improves.

Table 1. Limits of the standard deviation values chosen for particle roughening.

Sigma Label	Lower Limit (σ_{r1})	Upper Limit (σ_{r2})
Sigma—1	0.001	0.01
Sigma—2	0.01	0.1
Sigma—3	0.1	0.1

4.3. Remaining Useful Life Estimation

The proposed approach, as shown in Figure 5, is split into two stages: Stage A—Adaptive Bayesian learning framework with weight regularization and Stage B—Prognosis using NN. Stage A is further split into three steps—Data Preprocessing, Particle Filters, and Weight Regularization, the descriptions of which are explained below.

1. Data Preprocessing—An appropriate NN model is chosen based on the *BIC* analysis described in previous sections.
2. Particle Filters—The chosen model is used as the measurement function in the particle filter algorithm to recursively update the model parameters using the available degradation data from the training dataset.
3. Weight Regularization—To overcome the particle degeneracy and impoverishment issues, suitable resampling strategies, and roughening methods are adopted for weight regularization. Additionally, resampling/roughening at every time step is unnecessary as it introduces additional variance in the posterior distribution. Hence, Effective Sample Size (ESS) is introduced to regulate the resampling/roughening process. If the ESS of the distribution is lower than a predefined threshold N_T, then the particles are resampled/roughened. This reduces unnecessary additional computational load. The

parameters obtained at the final time step for the training dataset is the PF estimated MLP network parameters.

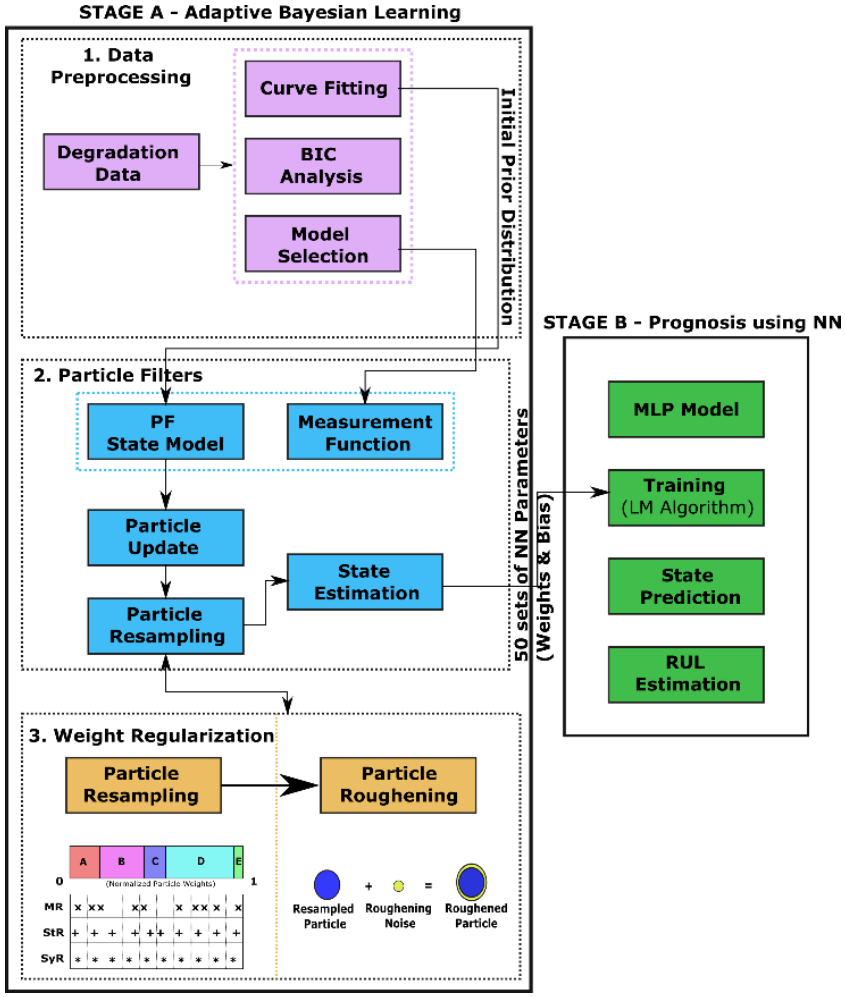

Figure 5. The schematic of the proposed prognostic framework using an adaptive Bayesian learning framework with weight regularization for training the MLP network model.

In Stage B, a new MLP network architecture is configured with PF estimated network parameters, which are the initial parameter values for corresponding weight and bias at the input and hidden layers. The available degradation data of other devices in the lot (test dataset—device 2 and 3) is fed to the MLP model as input. The MLP is trained using the LM algorithm for the test dataset. The informed PF estimated initial parameter values help to prevent local minima encountered in the backpropagation learning algorithm of neural networks. The trained NN model is used to predict the future degradation state of the device till end-of-life.

During backpropagation, there is a possibility that the network parameters can go astray and distort the prediction traces. A suitable success criterion is essential to filter out outliers from the degradation traces. Two success criteria are incorporated into the framework. One is to eliminate traces that lie beyond the 2σ limits of the majority of the prediction traces. The second criterion is to eliminate traces which has a monotonically increasing trend utterly different from the actual degradation data used in this work. The predicted EoL is evaluated based on the successful traces, and eventually, the RUL of the device is calculated. The performance of the proposed framework is evaluated using the percentage of successful iterations, Root Mean Squared Error (RMSE), Relative Accuracy (RA), and computational time as metrics.

5. Results and Discussion

5.1. RUL Estimation Using Different Resampling Strategies

The prediction results for CALCE battery degradation dataset are discussed in this section. The battery CS-36 was used for training the model, and batteries CS-37 and CS-38 were chosen as the test datasets. As per the *BIC* analysis discussed in Section 3, the network architecture which best represents the CALCE dataset consists of an MLP network with three hidden neurons. The corresponding model equation was used for generating the curve fitting coefficients which was, in turn, used to populate the initial prior distribution in the PF algorithm. The three neuron NN model equation (as per Equation (2)) was used as the measurement function in the PF algorithm, which recursively updates the model parameters for the training dataset (CS-36). The PF algorithm encounters weight decay issues during particle updates, as illustrated in Figure 3b. Hence, suitable resampling strategies were adopted to sort the weight decay issues. 50 sets of model parameter values from the posterior distribution of the PF algorithm were fed into the MLP model as initial parameter configuration value. Available data points from the test dataset were fed as input to the MLP model for prognosis.

The prediction results for CS-37 using multinomial resampling (MR) are shown in Figure 6b. Based on the assumed success criteria, the green traces in the Figure 6b are considered unsuccessful and eliminated for RUL prediction. The magenta traces correspond to prediction traces with RMSE values below 1% and thus is considered the best traces amongst the 50 repetitions. It can be observed from Figure 6b that only about 78% of the prediction traces were successful. The corresponding posterior model parameter distributions obtained after state estimation is shown in Figure 6a. A significant variance in the posterior distributions indicates that the weight decay issues of the PF algorithm remain unresolved for MR. The prediction results for CS-37 using stratified resampling (StR) are shown in Figure 7b. In this case, we opted to stratify the weight distribution into $N_s/2$ times, where N_s is the total number of particles used in the PF algorithm. During resampling, at least one particle from each stratum was picked, thereby improving particle diversity. Although StR yields a success rate of about 90%, the posterior distribution of model parameters shown in Figure 7a still shows negatively skewed distributions with no diversity. This indicates that if the PF algorithm fails to converge close to the true parameter values, then the prediction performance can be affected adversely.

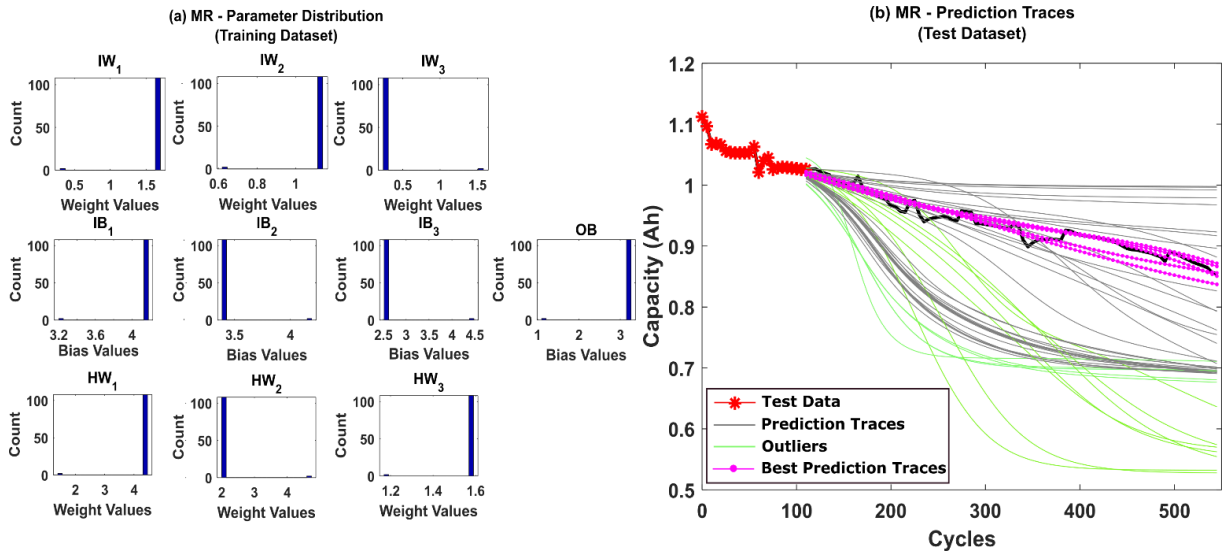

Figure 6. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using MR and (**b**) the degradation prediction traces for CALCE battery (CS-37) using MR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

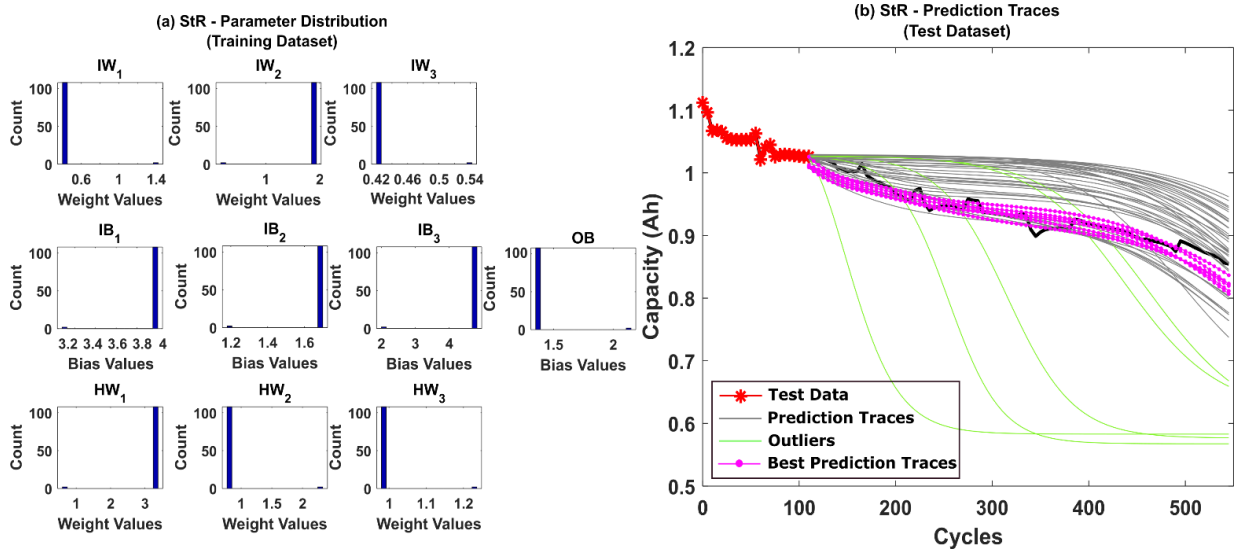

Figure 7. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using StR and (**b**) the degradation prediction traces for CALCE battery (CS-37) using StR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

Further, systematic resampling (SyR) was adopted to improve the accuracy of predictions. The number of successful iterations jumped to 96%, and the posterior parameter distributions were found to be more spread out, as shown in Figure 8a,b, respectively. The results clearly depict that resolving the weight decay issues in the PF algorithm regularizes the NN weights and bias values and, in turn, helps to improve the prediction accuracy.

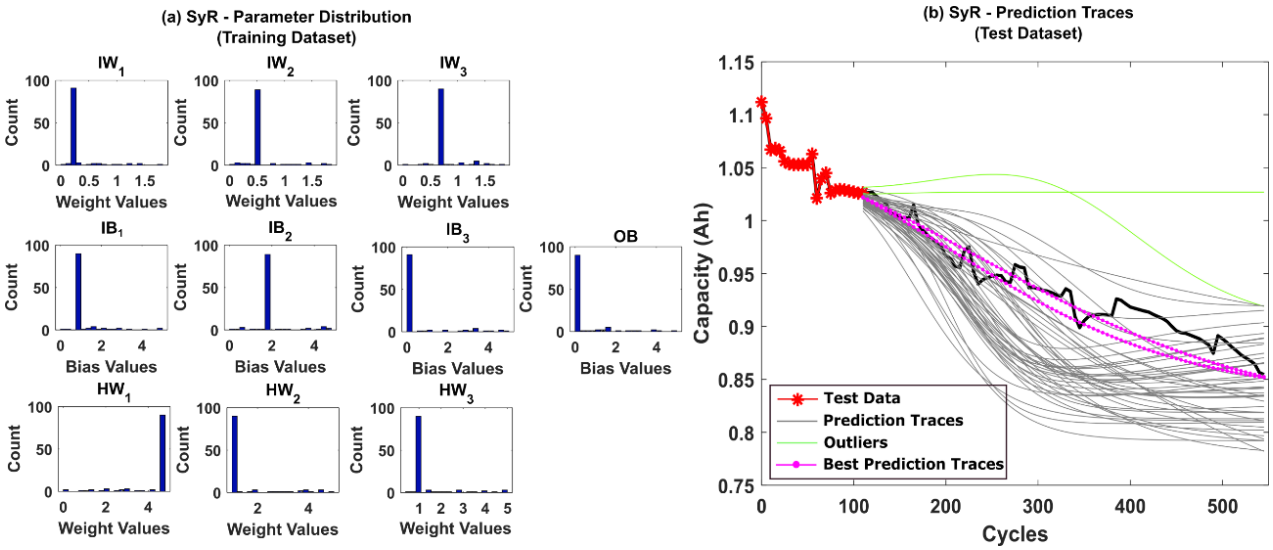

Figure 8. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using SyR and (**b**) the degradation prediction traces for CALCE battery (CS-37) using SyR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

The performance metrics comprising of RMSE, computational time and percentage of successful iterations are summarized in Table 2. It can be inferred from Table 2 that SyR performs the best amongst the different resampling strategies explored in this work and also eliminates the particle degeneracy issues. Since SyR is deterministic in nature, it proves to be better than the other two resampling strategies. However, systematic resampling

does not overcome particle impoverishment. To elucidate, we have shown the particle weight distribution plots in Figure 9a–d, while using SyR. The total number of time steps till the actual EoL for CALCE dataset is 112. The particle weight distributions at 76th, 80th, 96th, and 112th time step (last time step) is shown in Figure 9. The particle weights start to lose their diversity around 80th time step and subsequently suffers severe particle impoverishment around the last iteration. In order to over this issue, particle roughening strategies were adopted.

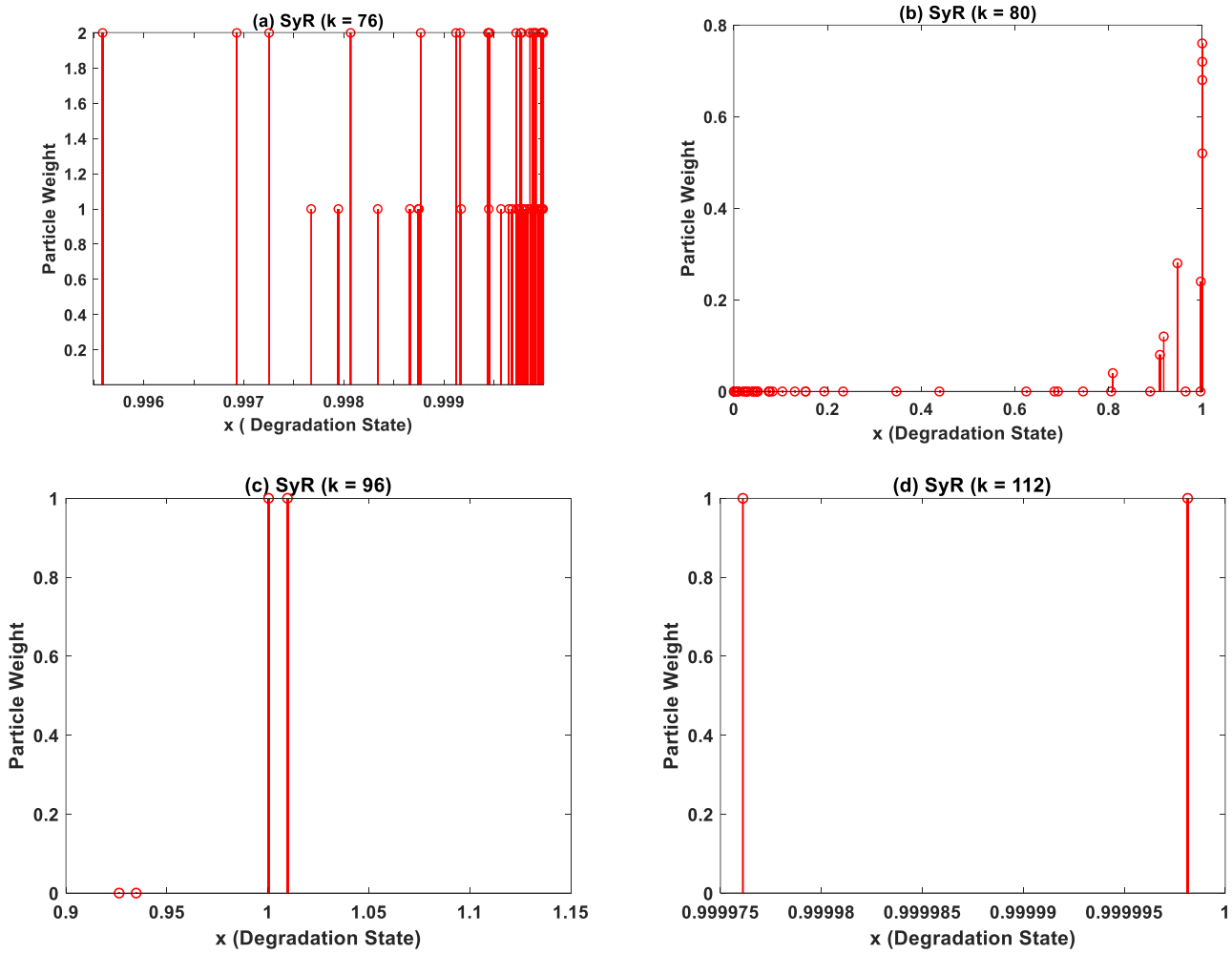

Figure 9. The particle weight distribution obtained during execution of particle filter algorithm using systematic resampling (SyR) at the (**a**) 76th (**b**) 80th (**c**) 96th, and (**d**) 112th time step. The particle weight distribution in (**d**) depict particle impoverishment despite using adopting robust resampling strategies.

Table 2. Comparison of Performance Metrics for Different Resampling Strategies and Roughening for CALCE and NASA degradation datasets.

	MR				StR				SyR				Particle Roughening			
Dataset	**RMSE**	**RA**	**Time (s)**	**Successful Iterations (%)**	**RMSE**	**RA**	**Time (s)**	**Successful Iterations (%)**	**RMSE**	**RA**	**Time (s)**	**Successful Iterations (%)**	**RMSE**	**RA**	**Time (s)**	**Successful Iterations (%)**
CALCE (CS-37)	0.14	0.85	47.0	78	0.06	0.56	41.9	90	0.05	0.92	42.1	96	0.04	0.96	343	98
NASA (RW11)	0.37	0.58	66.3	84	0.35	0.72	31.3	90	0.27	0.94	83.8	98	0.24	0.95	255	94

5.2. RUL Estimation Using Particle Roughening Method

For particle roughening, the standard deviation of the Gaussian jitters is the key influencing factor for improving particle diversity. Based on Ref. [34], three different sigma values were used in literature to simulate the jittering effect as shown in Table 1, and Sigma-2 values were found to be the best-performing ones. The sigma limits were chosen based on the admissible values of measurement noise to be present in the system under consideration. Hence, we adopted Sigma-2 for generating the Gaussian jitter to be added to the resampled particles. The particles were resampled using SyR and the resampled particles were added with a small Gaussian jitter with zero mean and standard deviation corresponding to Sigma-2.

However, roughening comes with an additional computational cost as shown in Table 2. Particle roughening takes about 8 to 10 times more time than SyR. The big advantage though is that the proposed method eliminates particle impoverishment. The posterior parameter distributions and the prediction traces for CS-37 using particle roughening strategy are shown in Figure 10a,b, respectively. The number of successful iterations improves to 98% with just one unsuccessful iteration. Additionally, the prediction traces are more intact and closer to the true values shown by the black line in Figure 9a. Moreover, the posterior distributions of parameters are more spread out clearly indicating a better particle diversity.

The proposed weight regularization method has advantages over other evolutionary algorithms, such as genetic algorithm and particle swarm optimization algorithm, which are highly computationally expensive approaches. To the best of our knowledge, the proposed adaptive Bayesian learning with weight regularization is the first of its kind to be used to optimize the MLP parameters for prognostic applications.

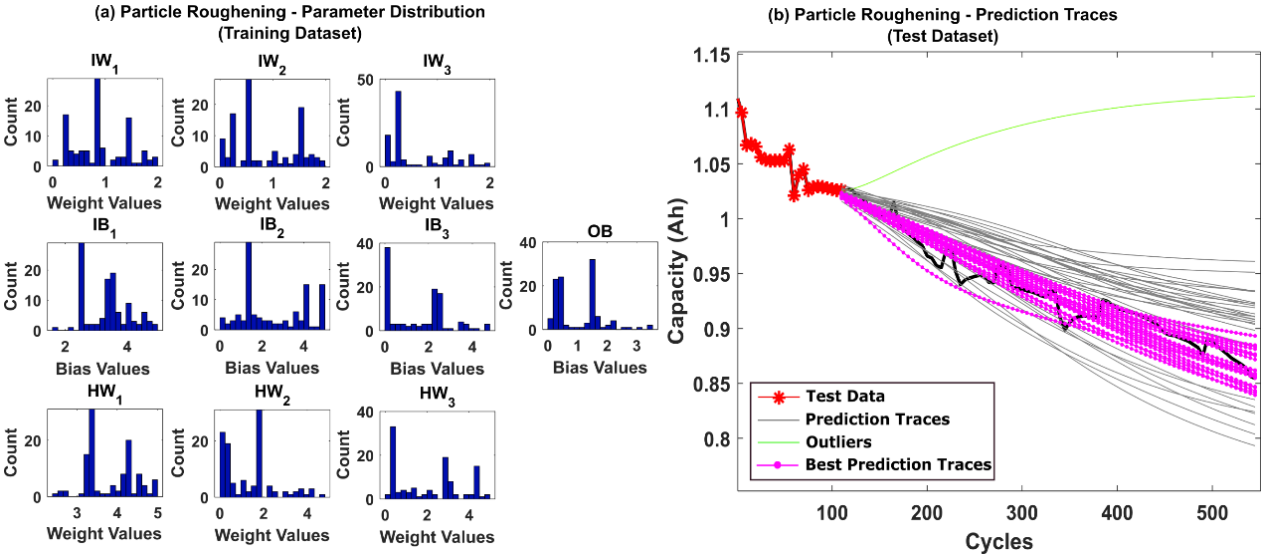

Figure 10. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using particle roughening and (**b**) the degradation prediction traces for CALCE battery (CS-37) using particle roughening. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

To check the robustness of the proposed method, we also applied the proposed prognostic approach to the NASA battery degradation dataset. RW10 was used for training the model, and battery RW11 was the test dataset. The performance results are again summarized in Table 2. The prognostic metrics used in this work are root mean squared error (RMSE), relative accuracy (RA) and computational time. The RMSE and RA values deduced in this work were evaluated using the following standard expressions in

Equations (11) and (12). In Equation (11), n is the size of the training dataset, T is the prediction starting point index value and k is the number of cycles.

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=T}^{n}\left(x_{predicted} - x_{true}\right)_k^2} \tag{11}$$

$$Relative\ Accuracy\ (RA) = 1 - \frac{|Predicted\ RUL - True\ RUL|}{True\ RUL} \tag{12}$$

The results indicate that the proposed prognostic framework with weight regularization outperforms the standard resampling strategies in the literature. The parameter distributions from the adaptive Bayesian learning framework and the corresponding degradation traces for NASA battery dataset are shown in Figures 11–14. Additionally, the comparison between predicted RUL and true RUL for both the datasets using the different resampling and roughening methods at three different prediction starting points are shown in Figures 15 and 16, respectively, in terms of the RA metric.

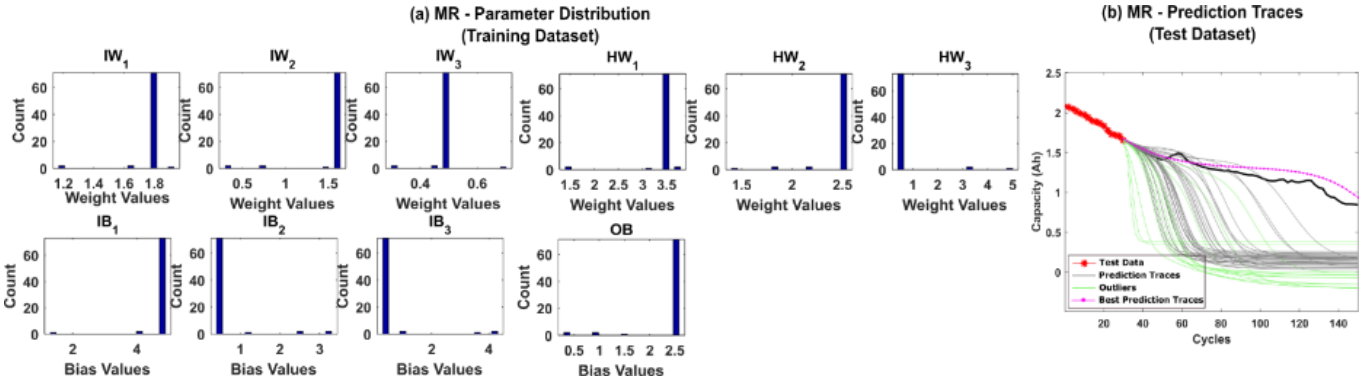

Figure 11. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using MR and (**b**) the degradation prediction traces for NASA battery (RW-11) using MR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

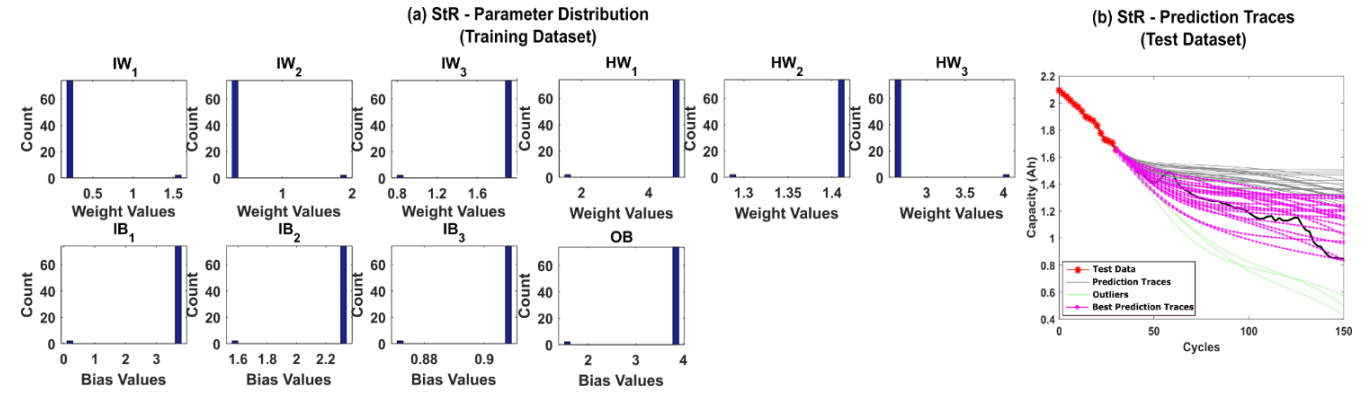

Figure 12. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using StR and (**b**) the degradation prediction traces for NASA battery (RW-11) using StR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

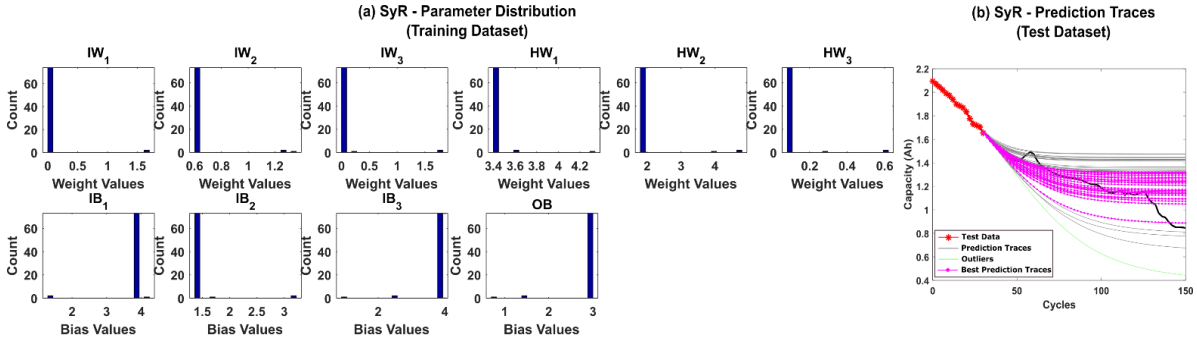

Figure 13. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using SyR and (**b**) the degradation prediction traces for NASA battery (RW-11) using SyR. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

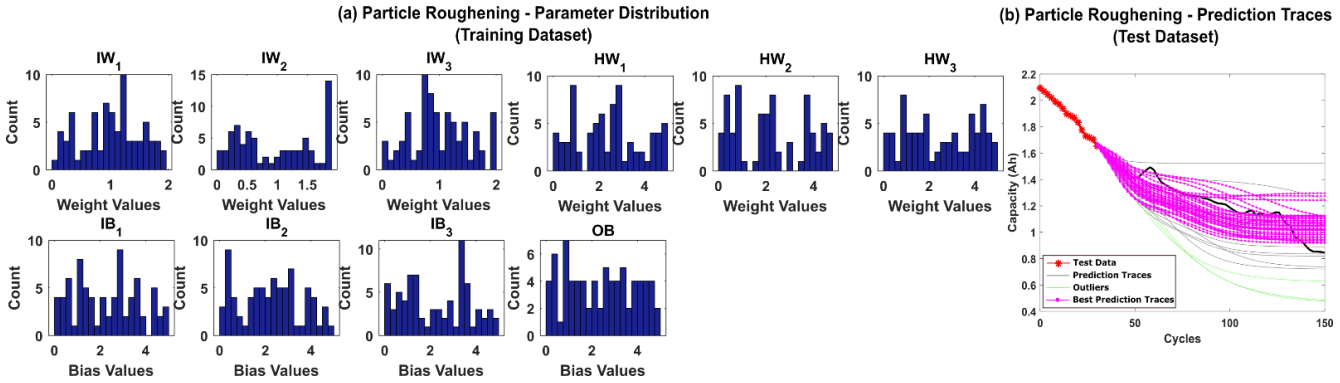

Figure 14. (**a**) The posterior distribution of MLP network parameters estimated by the adaptive Bayesian learning framework using particle roughening and (**b**) the degradation prediction traces for NASA battery (RW-11) using particle roughening. The prediction traces for 50 repetitions are shown using the gray lines, the green lines represent outliers, and the magenta lines represent the traces with minimum RMSE values.

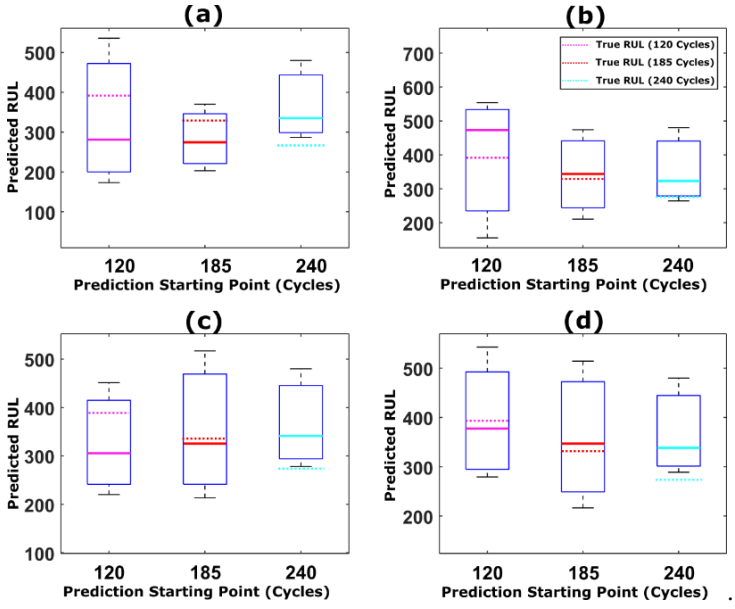

Figure 15. Box plot showing the comparison between predicted RUL and true RUL for CALCE dataset (CS-37) for three different prediction starting points using (**a**) MR, (**b**) StR, (**c**) SyR, and (**d**) particle roughening methods.

From Figures 15 and 16, it can be inferred that the accuracy of the resampling strategy adopted is reflected in the closeness of the predicted RUL to the true RUL value of the device under consideration The true RUL for both the datasets at different prediction starting points are represented by the magenta, red and cyan dotted lines. For both the datasets, particle roughening method performs the best with minimum RUL error value. Additionally, the variance in the RUL distribution can be inferred from the height of the box-plot shown in Figures 15 and 16. Thus, the results clearly show that the predictions results obtained using particle roughening method are both accurate as well as precise (relatively low variance in predicted RUL compared to most other common resampling methods).

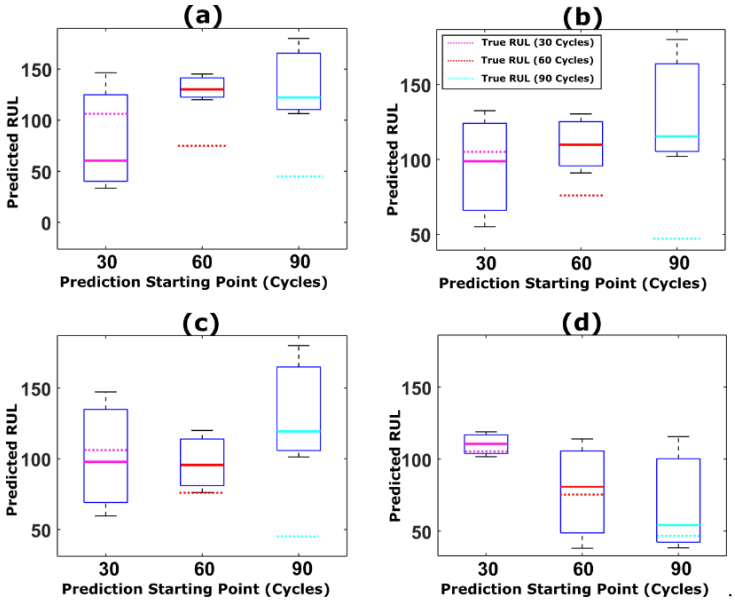

Figure 16. Box plot showing the comparison between predicted RUL and true RUL for NASA dataset (RW-11) for three different prediction starting points using (**a**) MR, (**b**) StR, (**c**) SyR, and (**d**) particle roughening methods.

5.3. Comparison of Prediction Results with Previous Works in the Literature

In order to evaluate the performance of our proposed method, we have compared our prediction results with two other relevant methods available in the literature. The prediction results obtained from Refs. [26,35] are used for comparison. The authors in Ref. [26] had developed a hybrid prognostic algorithm wherein the NN degradation model was used in the particle filter algorithm as the degradation model and further extrapolated in future for prognosis. In this work, the run-to-failure data of each battery were needed to determine the initial parameter guess which were fed to the PF algorithm. Similarly, Ref. [35] is one of our earliest works wherein we had developed a particle filter trained neural network model for prognosis. However, the efficacy of the proposed method was limited due to particle degeneracy and impoverishment issues. Hence, we adopted suitable weight regularization techniques in this work to overcome those disadvantages. The prognostic metric used in all the three work is the prediction RMSE value which are summarized in Table 3 below. It is evident from Table 3 that our proposed method here with weight regularization techniques incorporated into the hybrid framework performs better than the previous works from the literature.

Table 3. Comparison of RMSE values of the proposed method with Refs. [26,35].

Dataset	Ref. [26] (Wu et al.)	Ref. [35] (Our Previous Work)	This Work
CALCE (CS-37)	0.2698	0.050	0.040
NASA (RW11)	0.3686	0.367	0.24

6. Conclusions

In this work, we proposed an adaptive Bayesian learning framework to train MLP models for prognostic application on multiple electronic devices. The proposed adaptive Bayesian learning framework uses a particle filter algorithm for state estimation, wherein the weight decay issues commonly encountered in PF algorithms adversely affect the convergence of MLP weight and bias values and lead to poor prognostic performance. Hence, we adopted three different resampling strategies and particle roughening approaches into the PF framework. These strategies enable weight regularization in the MLP model used for prognosis. The proposed method was tested out on CALCE and NASA battery degradation datasets with high non-linearity and non-monotonicity. The prediction results clearly showed that systematic resampling helped improve particle diversity in the PF algorithm and subsequently helped eliminate particle degeneracy. Additionally, including the suitable particle roughening strategies in systematic resampling helps to eliminate particle impoverishment. The results imply that the proposed adaptive Bayesian learning framework with weight regularization helps the model parameters converge closer to the true values, prevent local minima problems, and helps to improve the generalization capabilities of NN models.

In the future, we intend to modify the proposed framework and incorporate physics informed loss functions into the MLP architecture. The purpose of including a physics-based loss function would be to apply the framework for devices with highly noisy data (and sparse good data scenarios as well) and underlying failure mechanisms with hidden physics. Although physics informed machine learning approaches are the state-of-the-art, the computational framework proposed thus far still have discrepancies in their convergence rates and suffer from vanishing backpropagation gradients. Thus, introducing a Bayesian inference-based training process can help to overcome the optimization and convergence issues.

Author Contributions: Conceptualization, K.P.; Data curation, K.P.; Formal analysis, K.P. and S.H.; Funding acquisition, N.R.; Investigation, H.P. and S.H.; Methodology, K.P.; Project administration, N.R.; Software, K.P.; Supervision, H.P. and N.R.; Validation, S.H.; Visualization, N.R.; Writing—original draft, K.P.; Writing—review & editing, N.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by A*STAR (Agency for Science, Technology and Research) Explainable Physics based AI Program (ePAI) under Programmatic Proposal Grant No. A20H5b0142.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Acknowledgments: The first author would also like to thank the Ministry of Education (MOE), Singapore for providing the research student scholarship (RSS) for 2018–2022.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kulkarni, C.; Biswas, G.; Saha, S.; Goebel, K. A model-based prognostics methodology for electrolytic capacitors based on electrical overstress accelerated aging. In Proceedings of the Annual Conference of the PHM Society, Montreal, QC, Canada, 29 November–2 December 2021.
2. Baptista, M.; Henriques, E.M.; de Medeiros, I.P.; Malere, J.P.; Nascimento, C.L., Jr.; Prendinger, H. Remaining useful life estimation in aeronautics: Combining data-driven and Kalman filtering. *Reliab. Eng. Syst.* **2019**, *184*, 228–239. [CrossRef]
3. Singleton, R.K.; Strangas, E.G.; Aviyente, S. Extended Kalman filtering for remaining-useful-life estimation of bearings. *IEEE Trans. Ind. Electron.* **2014**, *62*, 1781–1790. [CrossRef]
4. Celaya, J.R.; Saxena, A.; Kulkarni, C.S.; Saha, S.; Goebel, K. Prognostics approach for power MOSFET under thermal-stress aging. In Proceedings of the 2012 Annual Reliability and Maintainability Symposium, Reno, NV, USA, 23–26 January 2012.
5. An, D.; Choi, J.H.; Kim, N.H. Prognostics 101: A tutorial for particle filter-based prognostics algorithm using Matlab. *Reliab. Eng. Syst. Saf.* **2013**, *115*, 161–169. [CrossRef]
6. Zhang, H.; Miao, Q.; Zhang, X.; Liu, Z. An improved unscented particle filter approach for lithium-ion battery remaining useful life prediction. *Microelectron. Reliab.* **2018**, *81*, 288–298. [CrossRef]

7. Ahmad, W.; Khan, S.A.; Islam, M.M.; Kim, J.M. A reliable technique for remaining useful life estimation of rolling element bearings using dynamic regression models. *Reliab. Eng. Syst. Saf.* **2019**, *184*, 67–76. [CrossRef]
8. Si, X.S.; Wang, W.; Hu, C.H.; Zhou, D.H. Remaining useful life estimation–a review on the statistical data driven approaches. *Eur. J. Oper. Res.* **2011**, *213*, 1–14. [CrossRef]
9. Khelif, R.; Chebel-Morello, B.; Malinowski, S.; Laajili, E.; Fnaiech, F.; Zerhouni, N. Direct remaining useful life estimation based on support vector regression. *IEEE Trans. Ind. Electron.* **2016**, *64*, 2276–2285. [CrossRef]
10. Benkedjouh, T.; Medjaher, K.; Zerhouni, N.; Rechak, S. Remaining useful life estimation based on nonlinear feature reduction and support vector regression. *Eng. Appl. Artif. Intell.* **2013**, *26*, 1751–1760. [CrossRef]
11. Liu, D.; Zhou, J.; Pan, D.; Peng, Y.; Peng, X. Lithium-ion battery remaining useful life estimation with an optimized relevance vector machine algorithm with incremental learning. *Measurement* **2015**, *63*, 143–151. [CrossRef]
12. Yuchen, S.; Datong, L.; Yandong, H.; Jinxiang, Y.; Yu, P. Satellite lithium-ion battery remaining useful life estimation with an iterative updated RVM fused with the KF algorithm. *Chin. J. Aeronaut* **2018**, *31*, 31–40.
13. Benker, M.; Furtner, L.; Semm, T.; Zaeh, M.F. Utilizing uncertainty information in remaining useful life estimation via Bayesian neural networks and Hamiltonian Monte Carlo. *J. Manuf. Syst.* **2020**, *61*, 799–807. [CrossRef]
14. Nielsen, J.S.; Sørensen, J.D. Bayesian estimation of remaining useful life for wind turbine blades. *Energies* **2017**, *10*, 664. [CrossRef]
15. Le, T.T.; Chatelain, F.; Bérenguer, C. Multi-branch hidden Markov models for remaining useful life estimation of systems under multiple deterioration modes. *Proc. Inst. Mech. Eng. Part O J. Risk Reliab.* **2016**, *230*, 473–484. [CrossRef]
16. Saon, S.; Hiyama, T. Predicting remaining useful life of rotating machinery based artificial neural network. *Comput. Math. Appl.* **2010**, *60*, 1078–1087.
17. Farsi, M.A.; Hosseini, S.M. Statistical distributions comparison for remaining useful life prediction of components via ANN. *Int. J. Syst. Assur. Eng. Manag.* **2019**, *10*, 429–436. [CrossRef]
18. Li, X.; Ding, Q.; Sun, J.Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab. Eng. Syst. Saf.* **2018**, *172*, 1–11. [CrossRef]
19. Xia, M.; Li, T.; Shu, T.; Wan, J.; De Silva, C.W.; Wang, Z. A two-stage approach for the remaining useful life prediction of bearings using deep neural networks. *IEEE Trans. Industr. Inform.* **2018**, *15*, 3703–3711. [CrossRef]
20. Zhang, X.; Dong, Y.; Wen, L.; Lu, F.; Li, W. Remaining useful life estimation based on a new convolutional and recurrent neural network. In Proceedings of the 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, 22–26 August 2019.
21. Song, Y.; Li, L.; Peng, Y.; Liu, D. Lithium-ion battery remaining useful life prediction based on GRU-RNN. In Proceedings of the 2018 12th International Conference on Reliability, Maintainability, and Safety (ICRMS), Shanghai, China, 17–19 October 2018.
22. Pan, Y.; Hong, R.; Chen, J.; Wu, W. A hybrid DBN-SOM-PF-based prognostic approach of remaining useful life for wind turbine gearbox. *Renew. Energy* **2020**, *152*, 138–154. [CrossRef]
23. Wu, Y.; Yuan, M.; Dong, S.; Lin, L.; Liu, Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing* **2018**, *275*, 167–179. [CrossRef]
24. Karasu, S.; Altan, A. Crude oil time series prediction model based on LSTM network with chaotic Henry gas solubility optimization. *Energy* **2022**, *242*, 122964. [CrossRef]
25. Ma, G.; Zhang, Y.; Cheng, C.; Zhou, B.; Hu, P.; Yuan, Y. Remaining useful life prediction of lithium-ion batteries based on false nearest neighbors and a hybrid neural network. *Appl. Energy* **2019**, *253*, 113626. [CrossRef]
26. Wu, Y.; Li, W.; Wang, Y.; Zhang, K. Remaining useful life prediction of lithium-ion batteries using neural network and bat-based particle filter. *IEEE access* **2019**, *7*, 54843–54854. [CrossRef]
27. Gudise, V.G.; Venayagamoorthy, G.K. Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706), Indianapolis, IN, USA, 26–26 April 2003.
28. Karaboga, D.; Akay, B.; Ozturk, C. Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In Proceedings of the International Conference on Modeling Decisions for Artificial Intelligence, Umeå, Sweden, Kitakyushu, Japan, 16–18 August 2007.
29. CALCE Battery Dataset Repository. Available online: https://web.calce.umd.edu/batteries/data.htm (accessed on 13 December 2021).
30. Randomized Battery Usage Data Set. Available online: http://ti.arc.nasa.gov/project/prognostic-data-repository (accessed on 13 December 2021).
31. Vrieze, S.I. Model selection and psychological theory: A discussion of the differences between the Akaike information criterion (AIC) and the Bayesian information criterion (*BIC*). *Psychol. Methods* **2012**, *17*, 228. [CrossRef]
32. Chakrabarti, A.; Ghosh, J.K. AIC, *BIC* and recent advances in model selection. *Philos. Stat.* **2011**, *7*, 583–605.
33. Pugalenthi, K.; Raghavan, N. A holistic comparison of the different resampling algorithms for particle filter based prognosis using lithium ion batteries as a case study. *Microelectron. Reliab.* **2018**, *91*, 160–169. [CrossRef]
34. Pugalenthi, K.; Raghavan, N. Roughening Particle Filter Based Prognosis on Power MOSFETs Using ON-Resistance Variation. In Proceedings of the 2018 Prognostics and System Health Management Conference (PHM-Chongqing), Chongqing, China, 26–28 October 2018.
35. Pugalenthi, K.; Park, H.; Hussain, S.; Raghavan, N. Hybrid Particle Filter Trained Neural Network for Prognosis of Lithium-Ion Batteries. *IEEE Access* **2021**, *9*, 135132–135143. [CrossRef]