EAI Endorsed Transactions
on Energy Web
Research Article EAI.EU

A deep learning-based method for estimating the state of charge and predicting the remaining life of energy storage batteries

Yuanjing Qian[1], Ning Zhao[2*], Zhenrong Yang[3], Chang Liu[4]

[1]School of Energy and Environment, Southeast University, Nanjing, Jiangsu, 211189, China
[2]Shaanxi Water Development Huaze Zhihui Equipment Group Co., Ltd., Shaanxi, 710000, China,
[3]North China Municipal Engineering Design & Research Institute Co., Ltd., Xi'an, Shaanxi, 710000, China,
[4]Capital Construction Office, Shaanxi Provincial Hospital of Traditional Chinese Medicine, Shaanxi, 710000, China,

Abstract

INTRODUCTION: In order to improve the precision of state estimation and forecast of long term deterioration tendency in complicated charging and discharging conditions.
OBJECTIVES: This paper adopted a collaborative modeling approach for state of charge (SOC), state of health (SOH), and Remaining Useful Life (RUL) for SOC estimation.
METHODS: By integrating the Huber loss and state continuity constraints, this method can capture both the local and long-term temporal dependencies; The SOH estimation uses multiscale convolutions, gated temporal coding, and residual regression to represent cross-cycle degradation characteristics, and generates future health trajectories using an encoder-decoder architecture, which enables RUL threshold search prediction.
RESULTS: The results show that the average absolute error of SOC is 0.63%, and the root mean square error is 0.86%, and the SOH estimate reaches a mean absolute error of 0.58%, and the RUL estimate reaches 13.8 cycles.
CONCLUSION: Under joint learning, the SOC root mean square error, SOH mean absolute error, and RUL mean absolute error are further reduced to 0.79%, 0.55%, and 12.9 cycles, respectively, confirming that shared degradation representation and task-coordinated optimization improve state estimation and lifetime prediction.

Keywords: Energy storage batteries, deep learning, state of charge, state of health, remaining useful life

Received on 02 September 2026, accepted on 06 November 2026, published on 19 November 2026

Copyright © 2026 Yuanjing Qian *et al.*, licensed to EAI. This is an open access article distributed under the terms of the CC BY-NC-SA 4.0, which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/ew.14866

1. Introduction

The Smart State Estimation for Battery Management Systems (BMS) requires the reconstruction of the SOC, the health status (SOH), and the residual life — which is not directly measurable — from continuous voltage, current, temperature, and cycle degradation data. In recent years, the focus of the research has been shifted from static nonlinear mapping to multi-time feature extraction, long sequence dependency modeling, and generalization. Kim et al. [1] Using Long Short-Term Memory (LSTM) networks, it is shown that RNN can model long-term dependencies in SOC. However, a single sequence structure is still limited in describing local voltage spikes and multiscale dynamic

*Corresponding author. Email: zhaoningxp@163.com

responses. Schmitz et al. [2] Combined one dimensional convolutions with LSTM to perform on-line SOH estimation on a subset of constant current charging curves, achieving a MAE of 0.418% and a RMSE of 0.531%. This shows that the local feature coding can reduce the dependency on the whole charging process, but the input is still concentrated in a particular charging phase. Ezugwu et al. [3] further developed an SOC mapping using EIS and ensemble machine learning, where Extra Trees achieved a RMSE of 1.33% and a determination coefficient of 0.9977, showing that high dimensional electrochemical characteristics have strong state recognition capabilities; however, the need for additional impedance measurements limits their direct use in conventional BMS systems. Lanubile et al. [4] Using domain knowledge to extract power auto-correlation and energy health metrics from real world operations, the maximum absolute percent error of capacity estimation is kept within 1.5% to 2.5%, demonstrating that physical information is useful in improving the generalizability of data-driven models. Sonthalia et al. [5] Integrated Voltage, Charge/Discharge Time and Incremental Capacity Characteristics. The comparison of CNN, TCN, LSTM and BiLSTM shows that the SOH prediction performance of CNN and multi-layer LSTM is better. However, most of the methods are focused on SOC or SOH, and there is still no continuous information transmission mechanism between sample level operating state and cycle level degradation characterization.

For lifespan prediction, Suh et al. [6] proposed a spatiotemporal multimodal attention network, which includes temperature and internal resistance, which is often neglected in the RUL modeling. Their MAE and MSE values were 0.0275 and 0.0014, showing that the attention mechanism could enhance the fusion of complex degradation information; however, there was a lack of explicit consistency of SOC, SOH, and lifetime output. Reza et al. [7] Optimization of LSTM hyperparameters by using an algorithm to improve future capacity and RUL prediction precision, but the performance gains mainly depended on network parameters optimization and did not address the physical trend shift in long-term recursion. Onyenagubo et al. [8], which introduced multiple classes of machine learning into a web-based RUL prediction workflow, enhanced the applicability of the algorithm, but still focused on end-of-life regression. Feng et al. [9] developed a multi-strategy attention regression network (MARN) for joint SOH and RUL prediction using charging data, where differential data structuring and multiple attention strategies were used to strengthen degradation feature learning. In comparison, the proposed framework links the current SOH estimate with a recursively generated future SOH trajectory and further coordinates the optimization of the shared degradation representation. Rastegarpanah et al. [10] utilizes a hybrid neural network and Bayesian hyperparameter optimization to deal with capacity, temperature, internal resistance, and discharge time, which further reduces the RUL prediction error; however, there is still no unified constraint on the present health value, the future health path, and the failure threshold. This leads to particular research gaps: SOC and SOH/RUL are usually trained separately, making it difficult to propagate second-level operating characteristics toward cycle-level degradation states; future SOH recursion lacks an overall degradation constraint, which may produce non-physical recovery; and multi-task error scales and gradient directions require coordinated optimization. The 1D convolution, gated recurrent units, temporal attention, encoder-decoder prediction, and multi-task learning used in the framework are established learning components. The original contribution lies in their cross-timescale organization for battery state modeling: continuous SOC estimation is connected with cycle-level degradation characterization, the current SOH output constrains the starting point of recursively generated future health trajectories, and trajectory trend and SOH–RUL consistency constraints are jointly combined with dynamic task weighting and established gradient-conflict correction. This formulation provides coordinated state transfer from instantaneous SOC estimation to current SOH assessment and long-term RUL prediction under a unified degradation representation.

2. Modeling the State of Charge Estimation Problem

It is not possible to directly measure the SOC (SOC) of a battery using a sensor; its value is jointly affected by the charge/discharge current, the terminal voltage, the operating temperature, the degradation of the capacity, and the historical operating conditions. For this reason, the SOC estimation can be formulated as a hidden state regression problem in the case of multiple time series observations [11]. For a battery with a rated capacity of C_n , and neglecting self-discharge, the physical evolution relationship of SOC between adjacent sampling times can be expressed as

$$s_t = s_{t-1} - \frac{\eta_t I_t \Delta t}{C_n} \tag{1}$$

where s_t and s_{t-1} represent the SOC at the current and previous time steps, respectively; I_t is the charge/discharge current; Δt is the sampling interval; and η_t is the coulombic efficiency, which varies with temperature and operating conditions. Because the current integration error, capacity decline, and sensor noise may result in a state shift, the model input should not be restricted to single time step observations, but should instead organize the terminal voltage, current, and temperature sequences into a continuous window of length L, maintaining their dynamic relations [12]. Consequently, the SOC estimation task is defined as

$$\hat{s}_t = f_\theta(X_{t-L+1:t}, s_{t-1}), \quad X_t = [V_t, I_t, T_t]^T \tag{2}$$

where f_θ represents the nonlinear mapping to be trained, $X_{t-L+1:t}$ is the continuous operating feature sequence, and $\hat{s}_t$ is the estimated value. This method not only preserves the temporal continuity of the battery state but also provides a

uniform input for the extraction of local fluctuations and long term dependencies.

3. Deep Learning-Based State-of-Charge Estimation Methods

3.1. Estimation Model Based on a Hybrid Deep Temporal Network

In order to solve the problems of local abrupt changes, long term dependencies, and uneven contributions in voltage, current, and temperature variations, a hybrid deep temporal sequence network, consisting of one dimensional convolutional neural network, bidirectional gated recurrent units, and temporal attention. Figure 1 explains the interconnection of sliding window inputs, local feature coding, bidirectional state propagation, temporal weight assignment, and regression output. Firstly, the input sequence is subjected to a 1-dimensional convolutional layer with a kernel size of 3, which is used to maintain the continuity between adjacent sampling points. Its output is represented as

$$c_t = \sigma(W_c * X_{t-L+1:t} + b_c) \quad (3)$$

where $*$ denotes a one-dimensional convolution, andσ is a nonlinear activation function. The convolved features are fed into two bidirectional gated recurrent units (BGRUs) with a hidden dimension of 64, which are used to encode the history state evolution and the two-way correlation in the window, so that one-way recursion does not dilute the effective information at an early stage. Since different operating segments affect the present SOC to varying degrees, temporal attention assigns weights according to the correlation between the hidden state and the learnable context vector:

$$\alpha_i = \frac{exp(u^T\, tanh(W_a h_i + b_a))}{\sum_{j=1}^{L} exp(u^T\, tanh(W_a h_j + b_a))}, \quad \hat{s}_t = \varphi(\sum_{i=1}^{L} \alpha_i h_i) \quad (4)$$

where α_i represents the contribution weight at the i th time step, and φ is the regression mapping. This architecture balances the short term operation interference with the long term state memory, and provides a stable feature for the continuous SOC estimation under complicated charging and discharging conditions.

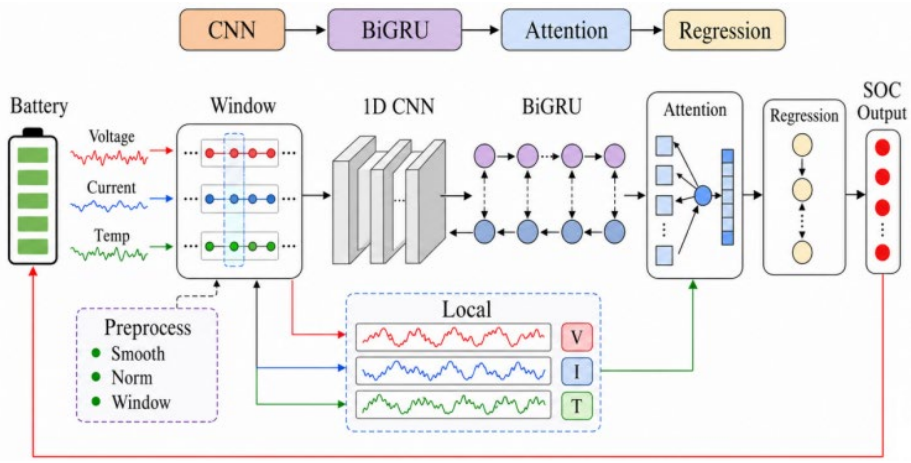


Figure 1. SOC estimation model based on a hybrid deep temporal network

3.2. Loss Function and Training Strategy

Voltage spikes, temperature disturbances, and sampling noise in the running data of the energy storage battery may result in a local anomaly. If the mean squared error is used directly, large residuals may dominate gradient updates and weaken the model's ability to fit steady-state discharge intervals. Thus, a Huber loss is used to constrain the SOC estimate bias:

$$L_H = \frac{1}{N}\sum_{i=1}^{N} \begin{cases} \frac{1}{2}e_i^2, & |e_i| \le \delta, \\ \delta\left(|e_i| - \frac{1}{2}\delta\right), & |e_i| > \delta, \end{cases} \quad e_i = \hat{s}_i - s_i \quad (5)$$

where N is the number of samples in a batch, s_i and $\hat{s}_i$ represent the true value and the estimated value, respectively, and the threshold δ is used to balance fine-tuning for small errors with suppression of anomalous residuals. Taking into account the physical properties of SOC's continuous evolution over time, the training objective simultaneously incorporates a constraint on changes between adjacent time steps and a weight regularization term:

$$L = L_H + \lambda_t \frac{1}{N-1}\sum_{i=2}^{N}[(\hat{s}_i - \hat{s}_{i-1}) - (s_i - s_{i-1})]^2 + \lambda_r \|\theta\|_2^2 \quad (6)$$

where λ_t controls the consistency of state changes, and λ_r suppresses excessive growth of network parameters. The training sample is divided according to the time sequence so as to avoid information leakage due to the overlapping of adjacent windows; the AdamW optimizer is used to update the parameters, and the learning rate is adjusted by coagulation-annealing [13]. Moderate dropout is applied to each layer, and gradient norm constraints are used to reduce gradient oscillations. Training is terminated when the validation error does not decrease for a number of consecutive epochs, thus achieving a good balance between convergence stability and generalization.

3.3. Overall Process for State of Charge Estimation

The State of Charge (SOC) is composed of two interrelated paths: the offline training and the on-line state reasoning. Figure 2 shows the relationship between operation data collection, anomaly detection, feature normalization, sliding window update, time series feature extraction, SOC regression, and state feedback [14]. During the offline phase, voltage, current, temperature, and SOC tags are arranged in time sequence; normalization parameters are calculated only from the training set, and the validation set is used to select the model so as to prevent the entry of future operational condition information into the training process. During the online phase, sampled data from the battery management system is continuously received; missing values, out-of-range values, and sudden noise spikes are flagged for quality control, and the most recent L sample points are constructed into an input window consistent with that of the training phase. The hybrid deep temporal network outputs an initial estimate $\hat{s}_t$, which is then combined with the state from the previous time step and the current integral result to perform continuity correction:

$$\tilde{s}_t = clip\left[\omega_t \hat{s}_t + (1-\omega_t)\left(\tilde{s}_{t-1} - \frac{\eta_t I_t \Delta t}{C_n}\right), 0,1\right] \quad (7)$$

where ω_t is a fusion weight determined based on the completeness of the input data and the model's confidence level, and$clip$ constrains the result to a valid range. The SOC after correction is used as the current output and is written into the status buffer for the next time step so that the estimation process can suppress the transient noise, the integration drift, and the discontinuous jump, thus providing a consistent operating state input for the estimation of the health condition and the life expectancy.

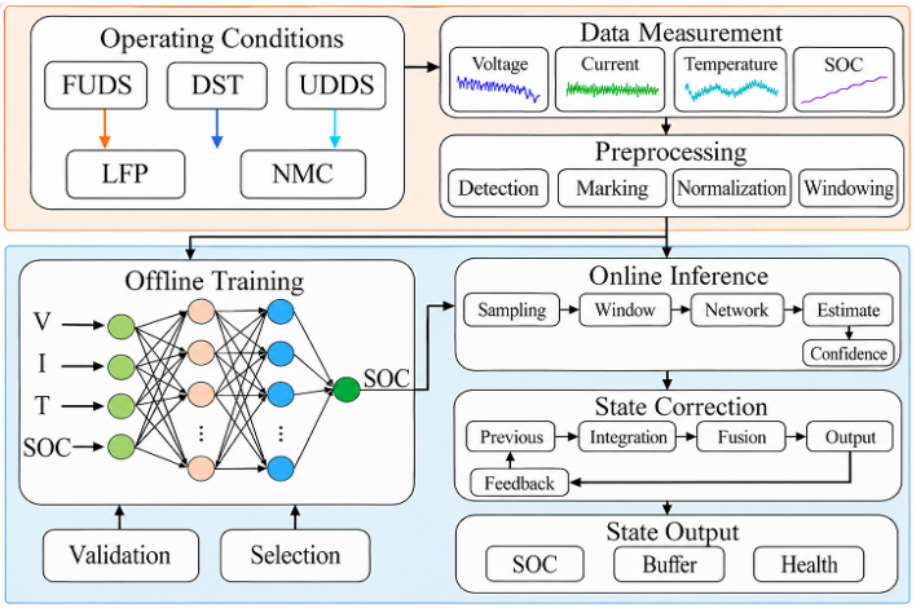


Figure 2. Overall Workflow for deep learning-based state of charge estimation

4. Deep Learning-Based Health Status Estimation and Remaining Useful Life Prediction

4.1. Problem Definition for SOH Estimation and RUL Prediction

The SOC estimates described above reflect the state within an individual charge–discharge process, whereas SOH and RUL require cycle-level information describing capacity loss, internal-resistance growth, and thermal variation [15]. For the kth prediction cycle, the current-cycle usable capacity is reserved exclusively for constructing the supervised SOH label in Eq. (9) and is not supplied to the

prediction network. In practical BMS operation, the preceding-cycle usable capacity is obtained by accumulating calibrated current over a completed discharge interval between validated charge and discharge endpoints. Equivalent internal resistance is estimated from the voltage response to an available current step using the synchronized voltage–current samples, while constant-current charging time, constant-voltage charging time, average discharge voltage, peak temperature, temperature-rise rate, and SOC variation range are accumulated directly from the voltage, current, temperature, and SOC records of the completed preceding cycle. The resulting cycle-level input is expressed as:

$$z_k = \left[Q_{k-1}, R_k, \tau_k^{CC}, \tau_k^{CV}, \bar{V}_k, T_k^{max^T_{k\,k}}[]^T\right] \quad (8)$$

Accordingly, only features available before cycle k are used for prediction. SOC can therefore be updated continuously at the sampling-window scale, whereas SOH and RUL are refreshed after a valid cycle-level descriptor becomes available. If a complete qualified cycle is unavailable, the present implementation retains the latest valid cycle-level descriptor and does not reconstruct an unobserved capacity value from future data. Partial-cycle SOH/RUL updating is outside the present validation scope. These characteristics correspond to the degradation of the active lithium, the increase in the ohmic polarization, the lower the charge acceptance, and the heat accumulation. To reduce the interference of single-cycle fluctuations on health assessment, the model uses a sequence of W consecutive cycles as the input sequence $Z_{k-W+1:k}$ and estimates the current SOH based on historical degradation rates and local changes. A capacity based SOH label shall be constructed on the basis of available capacity, measured at the same temperature, speed and cut-off voltage conditions:

$$SOH_k = \frac{Q_k}{Q_{ref}} \times 100\% \quad (9)$$

where Q_{ref} is the battery's initial reference capacity. Standardized test conditions prevent rate variations and temperature differences from being misinterpreted as permanent capacity degradation. RUL is defined as the number of cycles remaining from the current cycle to the end-of-life cycle:

$$K_{EOL} = min\{j \geq k | SOH_j \leq \gamma\}, \quad RUL_k = K_{EOL} - k \quad (10)$$

In particular, γ is determined using a capacity-based SOH threshold of 80% as defined in Eq. (9); thus, end of life is reached when the available capacity decreases to 80% of the initial reference capacity under the same standardized test conditions. Because capacity recovery and measurement disruption can lead to a temporary rebound in the SOH, the prediction of RUL has to include continuous deterioration trends instead of a single point of health. The model outputs a health track that extends from the present period to the failure threshold, together with the remaining number of cycles, to provide uniform samples, tags, and lifespan boundaries for the future SOH model based on regression and the lifetime prediction model [16].

4.2. SOH Estimation Model Based on Deep Regression Networks

To address the issues of capacity decay rates varying with the cycle stage and the susceptibility of single health indicators to temperature and rate disturbances, the SOH estimation model takes the degradation feature matrix $Z_{k-W+1:k}$ from a continuous sequence of W cycles as input and constructs a deep neural network that combines multi-scale convolutions, gated temporal encoding, and residual regression.The input layer normalizes the training set statistics for capacity, inner resistance, charge time, average voltage, and temperature rise, while retaining the relative variation of each characteristic throughout the cycle [17]. The current-cycle capacity Q_k, from which the SOH target is derived, is excluded from the input. The multiscale convolutional branches can be used to detect short term capacity fluctuations, intermediate degradation acceleration, and trend inflection points. Their fused features are represented as

$$F_k = Concat[\sigma(W_3 * Z_{k-W+1:k} + b_3), \\ \sigma(W_5 * Z_{k-W+1:k} + b_5), \sigma(W_7 * Z_{k-W+1:k} + b_7)] \quad (11)$$

where W_3 ,W_5 , and W_7 are the convolution parameters for different receptive fields, and$Concat$ denotes channel concatenation. The convolution features are fed into the Gated Recurrent Unit (GRUs), whose hidden dimension is set to 64, in order to maintain the equilibrium of the model capacity and the size of the cycle sample; the gated structure is used to store information on the continuous degradation and to reduce the interference of local anomalies — e.g., capacity renewal and environmental fluctuations — on the health representation [18] .Hidden states from each cycle are pooled through attention to form a degradation context vector, and the output of SOH is calculated by a residual regression head:

$$\hat{h}_k = clip\left[h_{k-1} + w_o^T \sigma\left(W_o \sum_{i=k-W+1}^{k} \alpha_i h_i + b_o\right), 0, 1\right] \quad (12)$$

where α_i represents the contribution weight of the i th cycle to the current health state, the residual term captures the health increment between adjacent cycles, and $clip$ ensures that the estimated values fall within the physically valid range. This architecture integrates multiple time-scale degradation patterns, cross-cycle dependencies, and states continuity into SOH regression, providing a stable health state sequence for the RUL model, which includes the degradation trend information.

4.3. RUL Prediction Model Based on an Encoder-Decoder Architecture

The RUL forecast is based on continuous SOH estimation and cycle level degradation characteristics, which can be used to generate future health paths and find failure thresholds. Figure 3 shows the full work flow, including the coding of the historical degradation sequence, the propagation of the latent state, the future SOH recursion, and the end-of life determination, and the combination of the SOH, the capacity variation rate, the inner resistance increment, the charging time, and the temperature rise characteristics from the last W cycle are used by the encoder. Its state update is expressed as

$$h_i^{enc} = GRU_{enc}(z_i, h_{i-1}^{enc}), \quad c_k = \sum_{i=k-W+1}^{k} \alpha_i h_i^{enc} \quad (13)$$

where c_k is a context vector characterizing the current degradation phase, decay rate, and local fluctuations, and the attention weightsα_i are used to enhance the contribution of recent periods of accelerated degradation and trend inflection points. The decoder takesc_k , the current SOH, and the health value from the previous prediction step as inputs to generate future health trajectories through cycle-based recursion:

$$\hat{h}_{k+\tau} = clip\left[\hat{h}_{k+\tau-1} - Softplus(w_d^T s_\tau + b_d), 0, 1\right],$$
$$s_\tau = GRU_{dec}\left(\left[\hat{h}_{k+\tau-1}, c_k\right], s_{\tau-1}\right) \quad (14)$$

Here, $Softplus$ constrains the predicted health decline to non-negative in the complete model, and the clipping operation keeps the recursively generated SOH within the physically valid range. The discrete first-crossing operation is used only during inference and evaluation, where the predicted RUL is determined by the first step satisfying $\hat{h}_{k+\tau} \le \gamma$ with $\gamma = 0.800$. Because this operation is nondifferentiable, it is not included directly in back-propagation. During training, the crossing time is approximated by a differentiable soft-survival formulation,

$$\tilde{y}_i^{RUL} = \sum_{\tau=1}^{H} \sigma\left(\frac{\hat{h}_{i,k+\tau} - \gamma}{T_s}\right), \quad L_{RUL} = \frac{1}{N}\sum_{i=1}^{N} SmoothL1(\tilde{y}_i^{RUL} - y_i^{RUL})$$

where H is the recursive prediction horizon, $T_s = 0.02$ controls the smoothness of threshold approximation, and y_i^{RUL} is the labeled remaining cycle number. The trend constraint in the complete model consists of both the trajectory-level trend penalty defined in Section 4.4 and the non-negative decrement projection in Eq. (14). Accordingly, the trend-constraint ablation removes both components and allows the decoder to output signed increments; under this ablation setting, local upward SOH changes and non-physical rebounds can therefore occur [20].

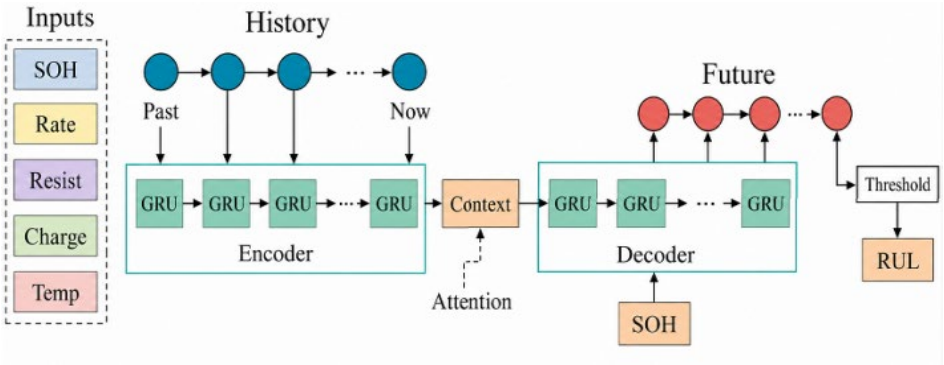


Figure 3. RUL prediction model based on an encoder-decoder architecture

4.4. Multi-Task Joint Learning Framework

SOC estimation operates at the sampling-window scale, whereas SOH and RUL prediction are defined at the cycle scale. To connect these time scales, the SOC temporal states belonging to cycle k are aggregated according to their cycle indices to form a cycle-aligned SOC descriptor. This descriptor is concatenated with the leakage-free degradation vector in Eq. (8) and then passed to the shared degradation encoder. Figure 4 shows the complete paths from sample-level SOC encoding to cycle-level SOH and RUL prediction, together with the corresponding gradient propagation. The shared encoder feeds the degradation feature matrix from W successive cycles into a multiscale convolutional layer. The convolutional outputs are propagated back along the cycle direction through a gated cycle unit, which allows for the integration of historical degradation information into the current health representation. The shared feature representation is given by

$$z_k^{SOC} = \sum_{t \in C_k} \alpha_{k,t} h_t^{SOC}, \quad h_k^D = F_{sh}([x_k || z_k^{SOC}]) \quad (15)$$

where C_k denotes the sampling windows assigned to cycle k, h_t^{SOC} is the temporal representation generated by the SOC encoder, and $\alpha_{k,t}$ is the normalized alignment weight. x_k is the cycle-level degradation vector defined in Eq. (8), and F_{sh} denotes the shared degradation encoder. Thus, sample-level SOC dynamics are aligned with cycle-level degradation information without introducing the target-defining current-cycle capacity. The SOH branch uses the present shared state to perform continuous regression, and uses the residual link to characterize the slowly varying health increment between neighboring cycles [22]; the RUL branch receives the shared status, SOH estimation, and degradation rate, and inputs them into an encoder decoder to produce the future health trajectory. The two tasks are connected by agreement at the predicted starting point: the first health value produced by the RUL branch should be near the output of the SOH branch, and the point at which the predicted path reaches the failure threshold should correspond to the RUL tag [23]. The integrated training objective is defined as

$$L_{\text{joint}} = \omega_{SOC}L_{SOC} + \omega_{SOH}L_{SOH} + \omega_{RUL}L_{RUL} + \lambda_c L_{con} + \lambda_t L_{trend} \quad (16)$$

where L_{SOC} is the SOC objective defined in Eq. (6), L_{SOH} and L_{RUL} are the corresponding cycle-level prediction losses, L_{con} constrains SOH–RUL starting-point consistency, and L_{trend} constrains the overall degradation direction. The task weights ω_{SOC}, ω_{SOH}, and ω_{RUL} are dynamically adjusted according to their loss variation rates. The coefficients are fixed at λ_c =0.20 and λ_t =0.10 throughout training. The SOC, SOH, and RUL task weights are initialized to 1.0 and updated once per epoch according to their relative loss variation rates. For task m∈{SOC,SOH,RUL}, the ratio between the losses of two consecutive epochs is calculated and normalized by a temperature-scaled Softmax with T_w=2.0, while the sum of the three task weights is maintained at 3. This strategy prevents a task with a larger numerical loss scale from dominating the shared representation. For the shared parameters, gradients of the weighted task losses are calculated separately. If two task gradients g_m and g_n satisfy $g_m^T g_n$ <0, the conflicting component is removed according to

$$g_m \leftarrow g_m - \frac{g_m^T g_n}{\|g_n\|_2^2 + \varepsilon} g_n$$

following the projected-gradient procedure in [24]. The corrected gradients are then accumulated to update the shared encoder, whereas task-specific parameters receive only the gradient of their corresponding task.

To facilitate reproducibility, detailed pseudocode for SOC preprocessing and estimation, SOH regression, recursive RUL prediction, and multi-task joint optimization is provided in Supplementary Algorithms S1–S3, with all major hyperparameters consistent with Table 1. To provide an explicit and reproducible description of the complete computational procedure, Algorithm 1 summarizes the data preprocessing, SOC estimation, cycle-level degradation modeling, SOH–RUL joint optimization, and threshold-based RUL inference.

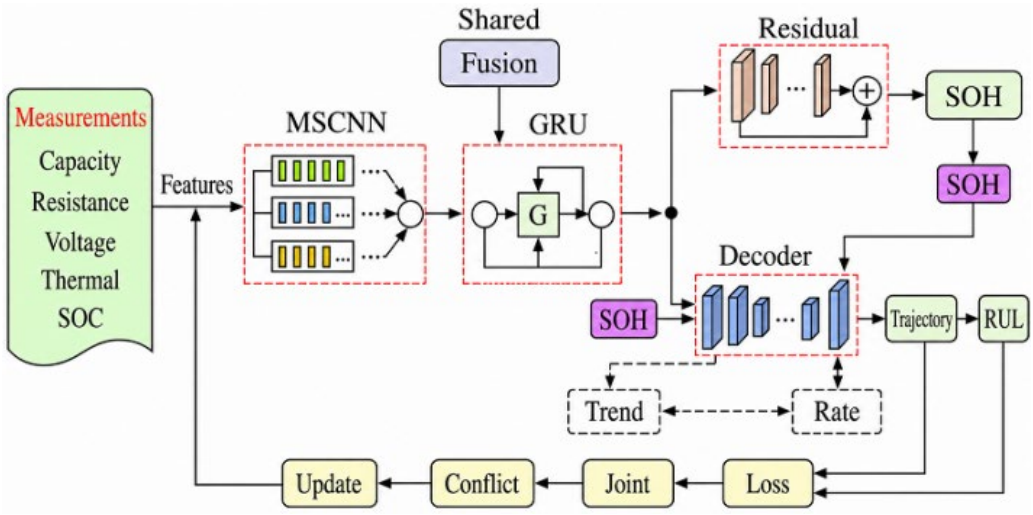


Figure 4. Multi-task joint learning framework for SOH estimation and RUL prediction

Algorithm 1. Unified procedure for SOC estimation, SOH estimation, and RUL prediction

Input:
Sampling sequences of voltage V, current I, and temperature T; SOC labels y^{SOC}; cycle-level degradation variables X^{deg}; SOH labels y^{SOC}; end-of-life threshold SOH_{EOL}=80%; SOC window length L =120; sliding step S =10; cycle window length W =20.
Output:
Estimated SOC $\hat{y}^{SOC}$, estimated SOH y^{SOC}, predicted future SOH trajectory $\hat{y}^{SOH}_{future}$, and RUL $\hat{y}^{RUL}$.

1. Calculate normalization parameters using the training set only, and normalize V, I, T, and cycle-level degradation variables.
2. Arrange the operating data chronologically and construct SOC sliding-window samples of length L with step S.
3. For each SOC window, extract local dynamic features using the one-dimensional convolution layer.
4. Feed the convolution features into the two-layer BiGRU and obtain bidirectional hidden states describing the temporal evolution within the window.
5. Calculate temporal attention weights over the hidden states, aggregate the weighted temporal representation, and obtain the initial SOC estimate through the regression layer.
6. Calculate the SOC training objective using the Huber loss, adjacent-state continuity constraint, and parameter regularization; update the SOC network using AdamW until the validation stopping criterion is reached.
7. During inference, combine the initial SOC estimate with the previous state and the current coulomb-integral result according to Eq. (7), constrain the corrected value to the valid SOC range, and store the corrected SOC for the next sampling step.
8. For each battery cycle, calculate usable capacity, equivalent internal resistance, constant-current charging time, constant-voltage charging time, average discharge voltage, peak temperature, temperature-rise rate, and SOC variation range to form the cycle-level degradation vector defined in Eq. (8).
9. Construct a degradation sequence using W consecutive cycles and extract short-, medium-, and long-range degradation characteristics through the multiscale convolutional branches.
10. Encode cross-cycle temporal dependencies using the gated recurrent unit, calculate attention weights over the cycle states, and obtain the shared degradation representation.
11. Feed the shared representation into the SOH residual regression branch and obtain the current SOH estimate $\hat{y}^{SOH}$.
12. Feed the historical degradation representation, current SOH estimate, and degradation-rate information into the encoder to obtain the degradation context state.
13. Initialize the decoder with the current SOH state and recursively generate the future SOH sequence while imposing the overall non-increasing degradation constraint.
14. During training, calculate the differentiable soft-crossing RUL estimate and L_{RUL} according to Section 4.3; during inference, terminate recursion at the first step satisfying $\hat{h}_{k+\tau}$ ≤0.80 and assign this first-passage step as the predicted RUL.
15. Calculate L_{SOH}, L_{RUL}, L_{con}, and L_{trend}, and combine them with L_{SOH} according to Eq. (16), using λ_c=0.20 and λ_t=0.10.
16. Update the SOC, SOH, and RUL task weights from their relative loss-variation rates using the temperature-normalized rule described in Section 4.4, while keeping the sum of the three task weights equal to 3.
17. Calculate task gradients on the shared parameters separately. For each pair satisfying $g_m^T g_n$<0, remove the conflicting projection component according to [24], and then update the shared parameters using the corrected gradients.
18. Update the shared degradation encoder and the task-specific SOH and RUL branches until the validation loss satisfies the early-stopping criterion.
19. Return $\hat{y}^{SOC}$, $\hat{y}^{SOH}$, $\hat{y}^{SOH}_{future}$, and $\hat{y}^{RUL}$.

End Algorithm

5. Experimental Validation and Results Analysis

5.1. Experimental setup and evaluation metrics

The experiments used a laboratory-generated full-life dataset collected from 12 commercial LiFePO4 (LFP) cells with a nominal capacity of 2.0 Ah using a programmable battery aging platform. Each cell was assigned a fixed temperature–rate condition: C01–C03 were operated at 0 °C with 0.5C, 1C, and 1.5C, respectively; C04–C06 at 10 °C, C07–C09 at 25 °C, and C10–C12 at 40 °C with the same three rates. Each

cycle consisted of constant-current/constant-voltage charging to 3.65 V, a 0.05C current cutoff, a 30-min rest, constant-current discharge to 2.50 V, and another 30-min rest. Terminal voltage, current, and surface temperature were sampled every 1 s, and the effective cycle life ranged from 936 to 1,184 cycles. Reference SOC labels were generated offline by calibrated current integration, anchored at 100% after complete CC–CV charging and corrected to 0% at the discharge cutoff; the measured cycle capacity was used only for reference-label normalization and was not supplied as a target-cycle input to the SOH/RUL network. To avoid sample leakage, the cells rather than overlapping windows were used as the partition unit [25]. The primary split contained 8 training, 2 validation, and 2 test cells, and three additional cell-disjoint 8/2/2 splits were introduced by rotating the held-out cells. The SOC samples retained a window length of 120 and a step of 10, yielding 168,420 sequences in the primary split, whereas SOH/RUL used 20 consecutive cycles and 10,436 cycle samples. Normalization statistics were estimated from training cells only. AdamW, a learning rate of 0.001, batch size of 64, maximum 200 epochs, and 15-epoch early stopping were retained. Five independent runs with different random initializations were performed for each trainable model, and results are reported as mean ± standard deviation. SOC and SOH were evaluated using MAE, RMSE, maximum absolute error, and R2, while RUL used MAE, RMSE, and relative lifetime error. Detailed settings are shown in Table 1.

The inference time is measured based on the forward calculation of single sample SOC on RTX 4090, using FP32 accuracy and batch size=1. The input tensor has been passed into the computing device before timing, so it does not include data acquisition, sliding window construction, and data transfer time from the host to the GPU. After completing 100 warm-up cycles, time 1000 forward inferences before and after CUDA synchronization operation, and report the average delay. Considering that desktop GPU latency cannot directly represent BMS side execution performance, further embedded testing was conducted on the NXP i.MX RT1176 evaluation platform, which uses a 1 GHz Cortex-M7 processor. After completing the training of the model, INT8 quantization was performed, followed by 100 warm-up cycles and 1000 timed runs.

Table 1. Experimental environment, data split, and model parameter settings

Category	Setting Item	Parameters or Configuration
Battery Data	Battery Type	LFP Energy Storage Battery
Battery Data	Number of Battery Cells	12 sets
Battery Specifications	Rated Capacity	2.0 Ah
Operating Conditions	Ambient Temperature	0°C, 10°C, 25°C, 40°C
Operating Conditions	Charge/Discharge Rate	0.5C, 1C, 1.5C
Operating Conditions	Sampling Interval	1 s
Data Segmentation	Training Set/Validation Set/Test Set	8 sets / 2 sets / 2 sets of batteries
SOC samples	Input window length	120 sampling points
SOC samples	Sliding step size	10 sample points
SOH/RUL samples	Cycle window length	20 cycles
Model Training	Optimizer	AdamW
Model Training	Initial learning rate	0.001
Model Training	Batch Size	64
Model Training	Dropout Rate	0.20
Model Training	Maximum number of training iterations	200
Model Training	Early Stopping Threshold	15 rounds
Experiment Platform	Processor	Intel Core i9-14900K
Test Platform	Graphics Processor	NVIDIA RTX 4090 24 GB
Test Platform	Software Environment	Python 3.11, PyTorch 2.5, CUDA 12.4
Statistical Settings	Number of Independent Replications	5

5.2. State of charge estimation experiment

To verify the adaptability of the hybrid deep temporal network under complicated charging and discharging conditions, the proposed model was compared with Coulomb counting, the extended Kalman filter (EKF), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), CNN-LSTM, and Transformer. Coulomb counting used the same calibrated current and SOC initialization as the reference procedure; EKF adopted a second-order Thevenin model whose parameters were identified from the training cells. LSTM and GRU used two 64-unit recurrent layers, CNN-LSTM used a 64-channel one-dimensional convolution with kernel size 3 followed by a 64-unit LSTM, and Transformer used two encoder blocks with d_{model}=64, four attention heads, and a 128-dimensional feed-forward layer. All trainable baselines used the same cell split, 120-point input window, AdamW optimizer, batch size, dropout, and early-stopping rule as the proposed model. Table 2 reports mean ± standard deviation over five independent runs. The proposed model achieved an MAE of 0.63±0.04% and an RMSE of 0.86±0.05%. Compared with Transformer, the corresponding mean errors were reduced by 29.21% and 27.73%, with R2=0.996±0.001. At 0 °C, the RMSEs of Transformer and the proposed model were 1.64±0.08% and 1.18±0.07%, respectively, while the proposed model obtained 1.29±0.09% RMSE for SOC ≤20%. For significance testing, one held-out battery cell, rather than an individual sliding window, was treated as the independent statistical unit. RMSE was first averaged across the five runs within each held-out cell, and a two-sided paired t-test was then performed on the eight cell-level paired observations collected from the four cell-disjoint splits. The difference from Transformer remained significant ($p=0.014<0.05$).

Table 2. Comparison of SOC estimation performance among different models

Model	MAE / %	RMSE / %	RMSE / % at 0 °C	RMSE / % in SOC ≤ 20% Range	R2	Number of Parameters / M	Inference Time / ms
Coulomb Counting	2.48	3.21	4.37	4.62	0.963	—	0.02
Extended Kalman Filter	1.91	2.54	3.48	3.79	0.975	—	0.18
LSTM	1.26±0.07	1.68±0.08	2.31±0.10	2.56±0.11	0.986±0.002	0.18	0.31
GRU	1.18±0.06	1.57±0.07	2.18±0.09	2.42±0.10	0.988±0.002	0.15	0.27
CNN-LSTM	0.92±0.05	1.24±0.06	1.73±0.08	1.89±0.08	0.992±0.001	0.31	0.42
Transformer	0.89±0.05	1.19±0.06	1.64±0.08	1.81±0.09	0.993±0.001	0.46	0.58
Proposed Model	0.63±0.04	0.86±0.05	1.18±0.07	1.29±0.09	0.996±0.001	0.38	0.47

Note: The difference in RMSE between the proposed model and Transformer, as determined by a paired t-test, satisfies p = 0.014 < 0.05.

5.3. SOH estimation and RUL prediction experiments

The primary SOH and RUL evaluation used two cells, C03 and C11, that were completely excluded from model fitting and validation. This subsection evaluates the task-specific SOH regression network and RUL encoder-decoder independently, with each model optimized for its own prediction objective before multi-task coupling. Figure 5 shows their actual SOH curves, model estimates, cycle-level error profiles, and future health trajectories from different prediction starting points. Across five independent runs, the standalone SOH model achieved an MAE of 0.58±0.04%, an RMSE of 0.76±0.05%, and R^2=0.997±0.001. For RUL prediction, the end-of-life threshold remained SOH = 80%, corresponding to 80% retention of the initial reference capacity according to Eq. (9). For the test cell with an actual end of life at 1,042 cycles, the predicted terminal cycles from starting cycles 400, 600, and 800 were 1,018, 1,031, and 1,047, yielding absolute endpoint errors of 24, 11, and 5 cycles, respectively. The standalone RUL model yielded an MAE of 13.8±1.8 cycles, an RMSE of 18.6±2.2 cycles, and a relative lifetime error of 4.7±0.6% on the primary test set. These standalone results characterize the task-specific models and are therefore distinct from the controlled multi-task strategy comparisons reported in Table 3. To supplement the original two-cell evaluation, three additional cell-disjoint 8/2/2 splits were evaluated with test pairs C01/C07, C04/C08, and C05/C12. No held-out cell contributed windows to model fitting or normalization. The additional split-level RUL MAEs were 12.7±1.6, 11.6±1.5, and 13.9±1.7 cycles, respectively, and the corresponding per-cell SOC, SOH, and RUL results are reported in Supplementary Table 3. For the RUL baselines in Figure 5(d), LSTM and BiGRU used two 64-unit recurrent layers, TCN used four 64-channel dilated convolution blocks, and Seq2Seq used a 64-unit GRU encoder–decoder; all methods used the same 20-cycle input and training protocol. Their

RUL MAEs were 24.5±2.7, 20.8±2.4, 18.9±2.1, 16.7±1.9, and 13.8±1.8 cycles for LSTM, BiGRU, TCN, Seq2Seq, and the proposed model, respectively.

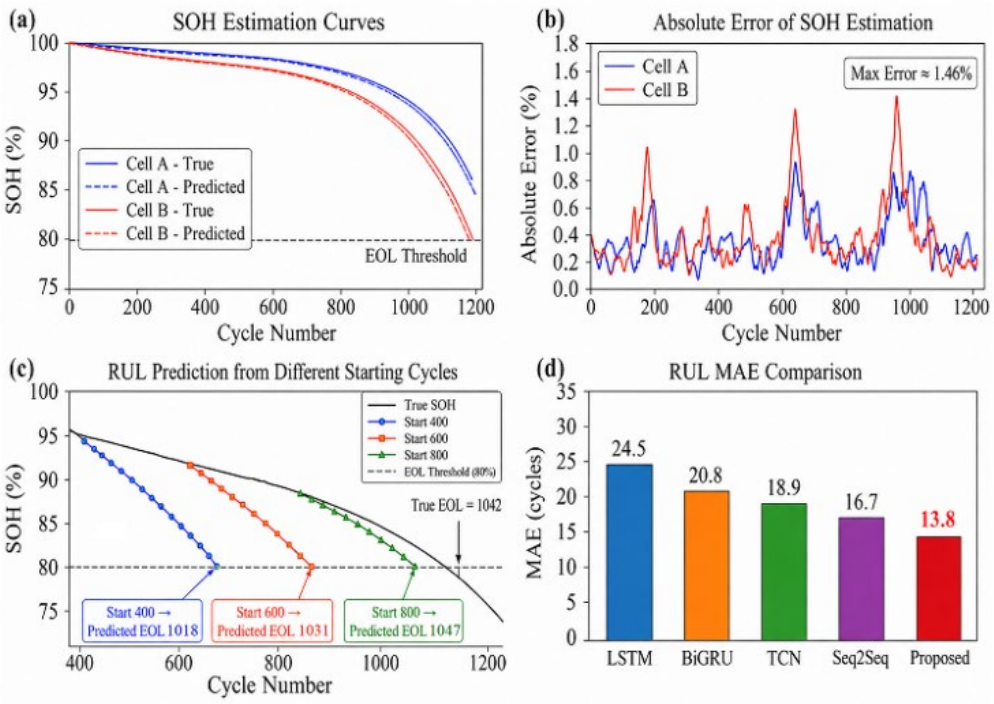


Figure 5. SOH Estimation and RUL Prediction Results

Table 3. Cell-level performance across cell-disjoint validation splits

Split	Test Cell	Temperature	Rate	SOC RMSE/%	SOH MAE/%	RUL MAE/cycles
A	C03	0 °C	1.5C	1.18±0.07	0.68±0.05	16.2±2.11
A	C11	40 °C	1.0C	0.54±0.04	0.48±0.03	11.4±1.51
B	C01	0 °C	0.5C	0.95±0.06	0.64±0.04	15.0±1.9
B	C07	25 °C	0.5C	0.68±0.04	0.46±0.03	10.4±1.3
C	C04	10 °C	0.5C	0.82±0.05	0.56±0.04	12.7±1.6
C	C08	25 °C	1.0C	0.70±0.04	0.47±0.03	10.5±1.4
D	C05	10 °C	1.0C	0.86±0.05	0.58±0.04	13.2±1.5
D	C12	40 °C	1.5C	0.93±0.06	0.62±0.05	14.6±1.8

5.4. Comprehensive evaluation of the multi-task joint estimation model

To isolate the contribution of multi-task coupling, five controlled training strategies were compared under an identical joint-framework architecture and common hyperparameter schedule: three-task separate training, fixed-weight shared encoder, dynamic-weight shared encoder, SOH–RUL consistency constraints, and the complete joint model. Unlike the independently tuned task-specific models in Section 5.3, the first row of Table 4 uses the same branch dimensions, training schedule, and optimization settings as the joint framework but disables parameter sharing and cross-task constraints. Consequently, its SOH MAE and RUL MAE are 0.66% and 15.7 cycles, rather than the independently optimized standalone values of 0.58% and 13.8 cycles reported in Section 5.3. All three models require separate parameter storage, with a cumulative parameter count of 0.91 M. The shared encoder reduces redundant feature extraction for capacity, internal resistance, temperature, and SOC changes; however, a fixed loss weight results in a significant percentage of the gradient in the initial training period, which leads to limited improvements in SOH

and SOC performance. All three error measures are reduced when dynamic task weights are used, which suggests that adjusting gradient contributions based on the uncertainty of the task can reduce the difference in the scale of loss. The average deviation of the beginning point of RUL decoding track and SOH branch output was reduced from 1. 32% to 0. 53%. In the whole model, SOC root mean square error, SOH mean absolute error, and RUL average absolute error decreased to 0.79%, 0.55%, 12.9%, respectively, which represented a reduction of 8.14%, 16.67%, 17.83% in comparison with the independent training. Compared with the MARN joint SOH–RUL framework in [9], which emphasizes differential charging-data organization and multi-strategy attention regression, the present joint model explicitly uses the current SOH estimate as the starting constraint of future health decoding and further coordinates shared optimization through SOH–RUL consistency and gradient correction. Since the datasets and evaluation protocols differ, the comparison here concerns modeling mechanisms rather than direct numerical performance.

Table 4. Comparison of overall performance across different multitask learning strategies

Learning Strategy	SOC RMSE /%	SOH MAE /%	RUL MAE /iteration	RUL Relative Error /%	SOH–RUL Consistency Error /%	Number of Arguments /M	Joint Inference Time /ms
Three-Task Independent Training	0.86	0.66	15.7	5.42	1.84	0.91	1.12
Shared Encoder + Fixed Weights	0.84	0.62	14.9	5.16	1.32	0.63	0.82
Shared Encoder + Dynamic Weights	0.82	0.59	14.2	4.91	0.96	0.64	0.84
Dynamic Weights + Consistency Constraints	0.81	0.57	13.7	4.73	0.53	0.65	0.86
Complete joint model	0.79	0.55	12.9	4.41	0.41	0.66	0.88

5.5. Ablation experiments

In the ablative experiment, the key modules were removed one after another with the same data partition, training iteration, and hyperparameter conditions. In table 5, the SOC root mean square error is increased from 0. 79% to 0. 92%, and the average SOH is 0. 64%. The removal of the gated temporal coding led to the greatest decrease in RUL performance, with an average absolute error of 17.4 cycles, suggesting that cross-cycle temporal dependency modeling is critical for determining the intersection point between future decay rates and lifetime thresholds. The SOC and SOH errors were increased by 0.08 and 0.06 percent after the removal of temporal attention, reflecting the contribution of the stable operating segment and the degradation transitional segment to different information. When the SOH residual regression was removed, the SOH error rose to 0.72%, and the output of neighboring cycles showed significant fluctuation; when the trend constraint of RUL was removed, RUL error rose to 16.8 cycles, consistency error was 0.93%, and some predicted trajectories showed non-physical rebound. Fixing the task weights led to an increase in both SOH and RUL errors, suggesting that the convergence rate and the error magnitude of different tasks need to be dynamically balanced. Correcting the consistency constraint resulted in a rise of 1.21% in the SOH – RUL consistency error; the removal of the gradient correction led to a slight decrease in the performance of the three tasks, demonstrating that there is indeed competition in the shared parameter updates. The results show that the performance of this model is due to the combination of local feature extraction, long term dependency modeling, physical trend constraint, and cross-task collaboration.

Table 5. Ablation experiment results for the multi-task joint model

Model Configuration	SOC RMSE /%	SOH MAE /%	RUL MAE /event	SOH–RUL Consistency Error /%
Complete Joint Model	0.79	0.55	12.9	0.41
Remove multiscale convolution	0.92	0.64	15.1	0.58
Remove gated temporal encoding	1.01	0.69	17.4	0.74
Removing temporal attention	0.87	0.61	14.3	0.52
SOH Residual Regression Removal	0.80	0.72	15.0	0.66
Remove RUL trend constraint	0.81	0.57	16.8	0.93

Fix task loss weights	0.84	0.62	14.9	1.32
Removal of SOH–RUL consistency constraint	0.82	0.60	14.6	1.21
Remove shared gradient correction	0.83	0.59	14.1	0.87

6. Conclusions

In order to meet the requirements of SOC, SOH and RUL, the proposed deep temporal computing framework provides a continuous mapping from instantaneous state estimation to long term degradation prediction. The hybrid convolutional-gated temporal-attention architecture improves the capability of co-representation of local fluctuations and long-term dependencies, while the multiscale degradation coding and encoder architecture transforms the RUL prediction into the problem of the generation of future SOH trajectories and the search of failure thresholds. The root mean square error of SOC, average absolute error of SOH, and average absolute error of RUL decreased to 0.79%, 0.55% and 12.9 cycles respectively, suggesting that dynamic task weights, SOH – RUL consistency constraints, and gradient conflict correction can alleviate the competition in a shared feature space. This framework is characterized by its unified modeling of the degradation information sharing across time scales, physical trend constraints, and multi-task computation. Currently, validation remains limited to 12 LFP cells, a restricted range of temperature and rate conditions, and an embedded-board benchmark rather than a production battery pack. Although the measured embedded latency indicates that the computational budget is compatible with the present SOC update interval, field deployment still requires validation of sensor interfaces, communication delay, incomplete-cycle operation, pack-level cell inconsistency, and long-term model drift. Future work could therefore include cross-domain transfer, uncertainty quantification, continuous learning, and hardware-in-the-loop and pack-level BMS validation before deployment in real energy-storage stations.

References

[1] Kim S B, Lee S H. Application of deep learning techniques for the state of charge prediction of lithium-ion batteries[J]. Applied Sciences, 2024, 14(17): 8077.

[2] Schmitz M, Kowal J. A deep learning approach for online state of health estimation of lithium-ion batteries using partial constant current charging curves[J]. Batteries, 2024, 10(6): 206.

[3] Ezugwu E O, Bhattacharya I, Ayomide A I, et al. State of charge (soc) estimation with electrochemical impedance spectroscopy (eis) data using different ensemble machine learning algorithms[J]. Electronics, 2025, 14(22): 4423.

[4] Lanubile A, Bosoni P, Pozzato G, et al. Domain knowledge-guided machine learning framework for state of health estimation in Lithium-ion batteries[J]. Communications engineering, 2024, 3(1): 168.

[5] Sonthalia A, Josephin J S F, Varuvel E G, et al. A deep learning multi-feature based fusion model for predicting the state of health of lithium-ion batteries[J]. Energy, 2025, 317: 134569.

[6] Suh S, Mittal D A, Bello H, et al. Remaining useful life prediction of lithium-ion batteries using spatio-temporal multimodal attention networks[J]. Heliyon, 2024, 10(16): e36236.

[7] Reza M S, Hannan M A, Mansor M, et al. Towards enhanced remaining useful life prediction of lithium-ion batteries with uncertainty using optimized deep learning algorithm[J]. Journal of Energy Storage, 2024, 98: 113056.

[8] Onyenagubo C, Ismail Y, Belu R, et al. Forecasting the Remaining Useful Life of Lithium-Ion Batteries Using Machine Learning Models−A Web-Based Application[J]. Algorithms, 2025, 18(6): 303.

[9] Feng W, Sun Z, Han Y, et al. A multi-strategy attention regression network for joint prediction of state of health and remaining useful life of lithium-ion batteries using only charging data[J]. Journal of Power Sources, 2025, 636: 236507.

[10] Rastegarpanah A, Asif M E, Stolkin R. Hybrid neural networks for enhanced predictions of remaining useful life in lithium-ion batteries[J]. Batteries, 2024, 10(3): 106.

[11] Guirguis J, Ahmed R. Transformer-based deep learning models for state of charge and state of health estimation of li-ion batteries: A survey study[J]. Energies, 2024, 17(14): 3502.

[12] Babu P S, V I, B A, et al. Enhanced SOC estimation of lithium ion batteries with RealTime data using machine learning algorithms[J]. Scientific Reports, 2024, 14(1): 16036.

[13] Iftikhar M, Shoaib M, Altaf A, et al. A deep learning approach to optimize remaining useful life prediction for Li-ion batteries[J]. Scientific Reports, 2024, 14(1): 25838.

[14] Hasib S A, Islam S, Ali M F, et al. Enhancing prediction accuracy of Remaining Useful Life in lithium-ion batteries: A deep learning approach with Bat optimizer[J]. Future Batteries, 2024, 2: 100003.

[15] Oka K, Tanibata N, Takeda H, et al. Deep learning based emulator for predicting voltage behaviour in lithium ion batteries[J]. Scientific Reports, 2024, 14(1): 28905.

[16] Jang J, Jo J, Kim J, et al. State of health estimation of lithium-ion battery cell based on optical thermometry with physics-informed machine learning[J]. Engineering Applications of Artificial Intelligence, 2025, 140: 109704.

[17] Mu B Q, Doukhi O, Wang L, et al. State-of-charge estimation of commercial electric vehicles by using battery-aware transformer network algorithms[J]. Journal of Computational Design and Engineering, 2025, 12(6): 73-95.

[18] Bairwa B, Pareek K, Jadoun V K. Cycle based state of health estimation of lithium ion cells using deep learning architectures[J]. Scientific Reports, 2025, 15(1): 37078.

[19] Rout S, Samal S K, Gelmecha D J, et al. Estimation of state of health for lithium-ion batteries using advanced data-driven techniques[J]. Scientific Reports, 2025, 15(1): 30438.

[20] Ibrahim M, Ansari S, Ayob A, et al. Hybrid optimized remaining useful life prediction framework for lithium-ion batteries with limited data samples[J]. Scientific Reports, 2025, 15(1): 39126.

[21] Singh A, Singh S, Kamboj R K, et al. Explainable OptiCNN-SLSTM hybrid model for enhanced lithium-ion battery state of health prediction[J]. Scientific Reports, 2025, 15(1): 44364.

[22] Carrera R, Quiroz L, Guevara C, et al. State-of-charge

estimation of medium-and high-voltage batteries using LSTM neural networks optimized with genetic algorithms[J]. Sensors, 2025, 25(15): 4632.
[23] Saneep K, Sundareswaran K, Nayak P S R, et al. State of charge estimation of lithium-ion batteries using PSO optimized random forest algorithm and performance analysis[J]. Journal of Energy Storage, 2025, 114: 115879.
[24] Yu T, Kumar S, Gupta A, et al. Gradient surgery for multi-task learning[J]. Advances in neural information processing systems, 2020, 33: 5824-5836.
[25] Al-Dulaimi A A, Guneser M T, Al-Shabandar R, et al. AE-BPNN: autoencoder and backpropagation neural network-based model for lithium-ion battery state of health estimation[J]. Scientific Reports, 2025, 15(1): 29193.