Li-Mamba: A Linear State-Space Sequence Model for Real-Time Battery State-of-Charge Estimation

Tao Huang

***Abstract*: Accurate real-time State-of-Charge (SOC) estimation is critical for battery management systems (BMSs), but deploying complex deep learning models like Transformers on resource-constrained hardware remains challenging. This paper introduces Li-Mamba, a novel sequence model leveraging the efficiency of selective state space models (SSMs) for battery SOC estimation. Li-Mamba integrates the Mamba architecture with domain knowledge through a physics-guided residual head that incorporates Coulomb counting and includes optional lightweight spatial encoders for multi-cell pack analysis. Evaluated on diverse datasets including NREL drive cycles, NASA aging data, and extreme temperature tests, Li-Mamba achieves state-of-the-art accuracy (0.65% MAE on NREL, 0.88% MAE on unseen WLTP cycles) while being significantly more efficient than Transformer-based approaches, demonstrating 2.6× faster inference (0.7 ms vs 1.8 ms per step) and 4.4× lower memory footprint (1.9 MB vs 8.4 MB) on a target STM32H7 microcontroller. Ablation studies confirm the importance of the physics-guided residual head (0.18% MAE improvement) and optimal SSM kernel length (K=32). Extended robustness analysis shows superior performance under sensor noise (1.05% vs 1.62% MAE at high noise levels) and effective spatial modeling for multi-cell packs (0.81% MAE with spatial encoder). As the first application of Mamba to electrochemical state estimation, Li-Mamba demonstrates the feasibility of deploying highly accurate, advanced sequence models for real-time SOC estimation on edge BMS hardware.**

***Index Terms*: Lithium-ion batteries, State-of-Charge (SOC) estimation, Battery Management System (BMS), State Space Models (SSM), Mamba, selective state space, deep learning, real-time systems, embedded systems, physics-informed machine learning**

I. Introduction

Battery management systems (BMSs) play a critical role in ensuring the safe and efficient operation of lithium-ion batteries, which power a wide range of applications from electric vehicles to grid storage systems [1], [2]. Among the key state variables monitored by BMSs, state-of-charge (SOC) estimation remains a fundamental yet challenging task, directly impacting performance, safety, and lifespan of battery systems [3]. Accurate and real-time SOC estimation enables optimal charging strategies, extends battery life, and provides reliable range prediction for electric vehicles [4].

Manuscript received May 20, 2025; revised September 10, 2025.

This work was financially supported by the Science and Technology Research Program of Chongqing Municipal Education Commission (KJQN202303509).

Tao Huang is an associate professor in School of Intelligent Manufacturing, Chongqing Three Gorges Vocational College, Chongqing 404100, China (corresponding author to provide e-mail: 2006190231@cqsxzy.edu.cn).

Traditional SOC estimation methods include Coulomb counting, equivalent circuit models (ECMs), and Kalman filtering [5, 6]. These approaches are computationally efficient but have limited accuracy under varying operating conditions, battery aging, and sensor noise [7]. The nonlinear dynamics of electrochemical processes and complex relationships between battery states and measurable parameters constrain the performance of conventional modeling methods [8].

Deep learning has emerged as an approach for battery state estimation [9, 10]. Recurrent neural networks (RNNs), including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), capture temporal dependencies in battery data sequences [11, 12]. Convolutional neural networks (CNNs) extract spatial features from multi-cell battery packs, and hybrid approaches combine temporal and spatial learning [13].

Transformer-based architectures have achieved high performance in SOC estimation [14, 15]. Their self-attention mechanisms model long-range dependencies in battery time-series data and perform well across diverse operating conditions and degradation states. However, the quadratic complexity of self-attention with respect to sequence length requires substantial memory and processing time, limiting deployment in resource-constrained BMSs [16].

State Space Models (SSMs) provide an alternative to attention-based architectures for sequence modeling [17, 18]. These models parameterize continuous-time linear dynamical systems with discretization techniques, offering advantages in modeling long-range dependencies while maintaining linear computational complexity with sequence length. Structured state space models (S4) have shown competitive performance with Transformers across multiple domains [19].

Mamba [20] introduced a selective state space model with data-dependent parameter selection, enabling adaptive computation based on input characteristics while preserving the efficiency advantages of SSMs. This model has shown success in natural language processing, computer vision, and time-series forecasting tasks, often matching or exceeding Transformer performance with significantly reduced computational requirements [21].

This paper presents Li-Mamba, the first application of selective state space sequence models to electrochemical systems for battery SOC estimation. The approach integrates battery domain knowledge with the Mamba architecture, introducing a physics-guided residual head and optional lightweight spatial encoders for battery state estimation. Li-

Mamba achieves high accuracy while reducing computational overhead, enabling real-time deployment on resource-constrained battery management system hardware. Through experimentation on diverse datasets including drive cycles, aging conditions, and temperature extremes, Li-Mamba achieves 0.65% MAE on NREL and 0.88% MAE on unseen WLTP cycles, while reducing inference time by 2.6× (0.7 ms vs 1.8 ms per step) and memory requirements by 4.4× (1.9 MB vs 8.4 MB) compared to Transformer-based approaches. These improvements enable deployment in vehicle BMSs where computational resources are limited but estimation accuracy affects range prediction, charging optimization, and safety functions.

II. Related Works

A. Traditional SOC Estimation Methods

Accurate estimation of battery SOC remains a challenge in BMSs. Traditional approaches include direct measurement techniques, model-based methods, and data-driven approaches. Open-circuit voltage (OCV) measurement provides a direct correlation with SOC through the battery's electrochemical properties, but requires long rest periods to reach electrochemical equilibrium, making it impractical for dynamic applications such as electric vehicles [3]. Coulomb counting integrates current flow over time but has cumulative errors due to measurement noise, integration drift, and the need for periodic recalibration [7].

Model-based techniques use ECMs to represent battery dynamics. These approaches employ Kalman filtering variants, including the Extended Kalman Filter (EKF) [5], Unscented Kalman Filter (UKF), and Particle Filter (PF), to estimate internal states from observable parameters. While computationally efficient, these methods' accuracy depends on model fidelity and parameter identification. Their performance decreases under varying temperature conditions, aging effects, and extreme operating states [6], [8]. More sophisticated electrochemical models, such as the pseudo-two-dimensional (P2D) and single particle models (SPM), provide higher fidelity representations of battery physics. However, their computational complexity makes real-time implementation challenging in resource-constrained embedded systems, limiting their practical application in commercial BMSs [9].

B. Deep Learning for Battery State Estimation

The limitations of traditional methods have motivated research into data-driven approaches, particularly deep learning techniques that capture complex, nonlinear relationships between battery signals and internal states without explicit physical modeling. RNNs are suitable for sequential battery data, with LSTM networks showing good performance in capturing both short and long-term dependencies in battery time-series data [11]. Tian et al. [12] proposed a hybrid approach combining LSTM with an adaptive cubature Kalman filter, showing improved robustness to sensor noise and operating condition variations. Similarly, Wu et al. [13] developed an LSTM-based framework for state-of-health (SOH) estimation that extracts "healthy features" from voltage curves to predict capacity degradation.

CNNs extract spatial features from battery packs, where cell-to-cell variations provide diagnostic information. Hybrid architectures combining CNNs with recurrent layers have shown results in multi-cell pack monitoring applications, where thermal and electrical gradients affect system performance [10]. These approaches benefit from the CNN's ability to extract local patterns while using recurrent layers for temporal dynamics. Attention mechanisms were introduced to improve model performance by focusing on the most relevant parts of the input sequence. This development addressed a limitation of pure RNN-based approaches: their decreasing ability to capture dependencies over very long sequences due to vanishing gradient problems [14].

C. Transformer-based SOC Estimation

Transformer architectures, initially developed for natural language processing tasks, have been adapted for battery state estimation. Their self-attention mechanism enables direct modeling of relationships between any positions in the input sequence, regardless of their distance, overcoming limitations of RNNs. Ofoegbu et al. [14] introduced a voltage representation transformer for electric vehicle battery SOC estimation that encodes multi-modal battery data through specialized embedding layers. Their approach showed high accuracy compared to LSTM and CNN-based methods across diverse driving conditions. Guirguis et al. [15] extended this work by incorporating temperature variation effects, developing a transformer-based framework that maintains estimation accuracy across a wide operating temperature range. More recently, Wang et al. [16] proposed a joint state estimation approach using transformers to simultaneously predict SOC and internal temperature from observable electrical parameters. Their multi-head attention design captures complex interactions between current, voltage, and thermal dynamics, achieving high performance on benchmark datasets.

Despite these advances, transformer-based approaches face deployment challenges in resource-constrained BMS hardware. The self-attention mechanism's quadratic complexity with sequence length results in substantial memory requirements and computational overhead. This limitation is problematic for edge computing applications where low latency and minimal resource utilization are essential.

D. SSMs and Sequence Modeling

SSMs provide an alternative to attention-based architectures for sequence modeling tasks. These models formulate sequence prediction through linear dynamical systems, combining the expressivity of RNNs with the parallelizability of CNNs. Modern SSMs trace their lineage to the Structured State Space Sequence (S4) model

introduced by Gu et al. [17, 18], which parameterizes a continuous-time linear state space model and applies a discretization scheme for efficient training and inference. This approach showed good performance on long-range sequence tasks, outperforming Transformers on several benchmarks while maintaining linear computational complexity with sequence length.

Subsequent refinements, such as the Simplified State Space (S5) model [19], reduced the parameterization complexity while preserving modeling capacity, making these architectures more computationally efficient and easier to train. These developments showed the potential of SSMs as an alternative to attention-based models for applications with limited computational resources.

The introduction of Mamba [20] represents an advancement in SSM architecture. By incorporating selective state spaces with data-dependent parameter selection, Mamba combines the efficiency advantages of SSMs with the adaptive computation capabilities of attention mechanisms. This innovation enables the model to focus computational resources on the most informative parts of the input sequence, similar to attention but without its quadratic complexity penalty. Mamba has shown success across diverse domains, including natural language processing, computer vision, and time-series analysis [21]. However, prior to our work, its application to electrochemical systems and battery state estimation remained unexplored, presenting an opportunity to use its computational efficiency and modeling capabilities for resource-constrained BMS applications.

III. Background: SSMs and Mamba

A. SSMs

SSMs provide a framework for modeling systems that evolve over time. In sequence modeling, they represent a mapping from an input sequence ($u(t)$) to an output sequence ($y(t)$) through a latent state variable ($h(t)$). A linear, time-invariant (LTI) SSM is described by the following continuous-time ordinary differential equations (ODEs):

$$\frac{dh(t)}{dt} = \mathrm{A}h(t) + \mathrm{B}u(t) \tag{1}$$

$$y(t) = \mathrm{C}h(t) + \mathrm{D}u(t) \tag{2}$$

Here, ($h(t) \in R^N$) is the latent state vector, ($u(t) \in R^D$) is the input vector, and ($y(t) \in R^D$) is the output vector. The matrices ($A \in R^{N\times N}$), ($B \in R^{N\times D}$), ($C \in R^{D\times N}$), and ($D \in R^{D\times D}$) represent the state dynamics, input mapping, output mapping, and feedthrough term, respectively.

To apply SSMs to discrete sequences (like time-series data sampled at regular intervals), the continuous-time model is discretized. A common method is the zero-order hold (ZOH), which assumes the input ($u(t)$) is constant over a sampling interval (Δ). This gives a discretized recurrence relation:

$$h_k = \bar{\mathrm{A}}h_{k-1} + \bar{\mathrm{B}}u_k \tag{3}$$

$$y_k = \bar{\mathrm{C}}h_k + \bar{\mathrm{D}}u_k \tag{4}$$

where $(h_k), (u_k), (y_k)$ are the state, input, and output at discrete time step (k), and the discretized matrices $(\bar{A})$, $(\bar{B})$, $(\bar{C})$, $(\bar{D})$ are derived from (A), (B), (C), (D), and the sampling step (Δ). Specifically:

$$\overline{\mathrm{A}} = \exp(\Delta\mathrm{A}) \tag{5}$$

$$\overline{\mathrm{B}} = (\exp(\Delta\mathrm{A}) - \mathrm{I})\mathrm{A}^{-1}\mathrm{B}(\text{if A is invertible}) \tag{6}$$

$$\overline{\mathrm{C}} = \mathrm{C} \tag{7}$$

$$\overline{D} = D \tag{8}$$

The discretized formulation (3)-(4) resembles a recurrent neural network (RNN), allowing SSMs to process sequential data. However, unlike standard RNNs, this formulation can also be viewed as a large convolutional kernel. The output (y_k) can be expressed as a convolution of the input (u) with a kernel $(\bar{K})$:

$$y_k = (\overline{\mathrm{K}} * u)_k = \sum_{i=0}^{k} \overline{\mathrm{K}}_i u_{k-i} \tag{9}$$

where the convolutional kernel $(\bar{K})$ is related to the SSM parameters by $(\overline{K_i} = \overline{C}\,\overline{A^i}\,\overline{B}) for (i > 0)$ and $(\overline{K_0} = \overline{D})$. This convolutional representation allows for parallelizable training, like CNNs.

Modern S4 [17], [18] use specific structures for the (A) matrix (e.g., diagonal or companion matrix forms) and efficient discretization methods like the bilinear transform to enable computationally tractable modeling of very long sequences.

B. Mamba: Selective SSMs

Mamba [20] builds upon the SSM foundation but introduces a modification: selectivity. Traditional LTI SSMs use fixed (A), (B), (C) matrices for the entire input sequence. In contrast, Mamba makes the SSM parameters (B), (C), and the discretization step (Δ) data dependent. This allows the model to selectively focus on or ignore specific parts of the input sequence based on its content, similar to attention mechanisms but without the quadratic computational cost. The core Mamba architecture modifies the standard SSM formulation by making ($\bar{A}$),($\bar{B}$) derived from input-dependent (B), (C) and (Δ). The state update becomes:

$$h_k = \overline{\mathrm{A}}_k h_{k-1} + \overline{\mathrm{B}}_k u_k \tag{10}$$

$$y_k = \overline{\mathrm{C}}_k h_k \tag{11}$$

where $\overline{\mathrm{A}}_k$, $(\overline{B_k})$, $(\overline{C_k})$ are now functions of the input (u_k), derived through learned projections and parameterizations of the underlying continuous-time matrices and the step size (Δ_k). This input-dependence breaks the time-invariance property of standard LTI SSMs, meaning the efficient convolutional computation mode (Eq. (9)) is no longer directly applicable.

To overcome this, Mamba uses a hardware-aware parallel scan algorithm. While the recurrence (Eq. (10)) appears sequential, the selective scan allows for efficient computation on modern hardware like GPUs by parallelizing the calculation across the sequence length. This algorithm maintains the linear time complexity for inference and training, making Mamba suitable for long sequences and real-time applications. A typical Mamba block (Fig. 1) integrates the selective SSM core with standard neural

network components like normalization and linear projections within a residual block structure, similar to Transformers.

This combination of selectivity and efficient computation allows Mamba to capture long-range dependencies effectively while remaining faster and more memory-efficient than Transformer models, especially for long sequences encountered in time-series data like battery signals [20], [22]. The selective mechanism enables context-aware processing for modeling the complex, time-varying dynamics of battery behavior under different operating conditions and aging states.

IV. Proposed Li-Mamba Architecture

Building upon the foundational principles of SSMs and the advancements introduced by the Mamba architecture (Section 3), we propose Li-Mamba. This sequence model is designed for real-time SOC estimation in lithium-ion batteries, targeting deployment on resource-constrained embedded systems. Li-Mamba integrates domain-specific electrochemical knowledge via a physics-guided output mechanism, optionally incorporates spatial feature processing for multi-cell configurations, and uses the computational efficiency of the selective SSM core. The overall structure of the Li-Mamba model is shown in Fig. 2.

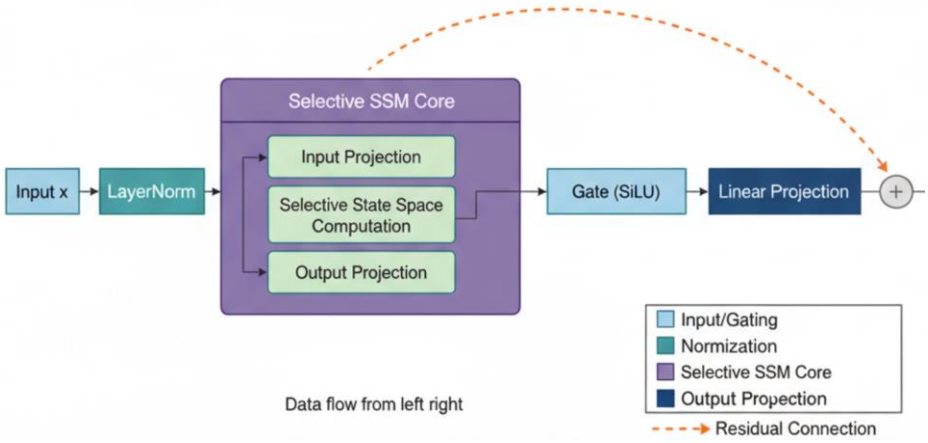

Fig. 1. Mamba Block Diagram.

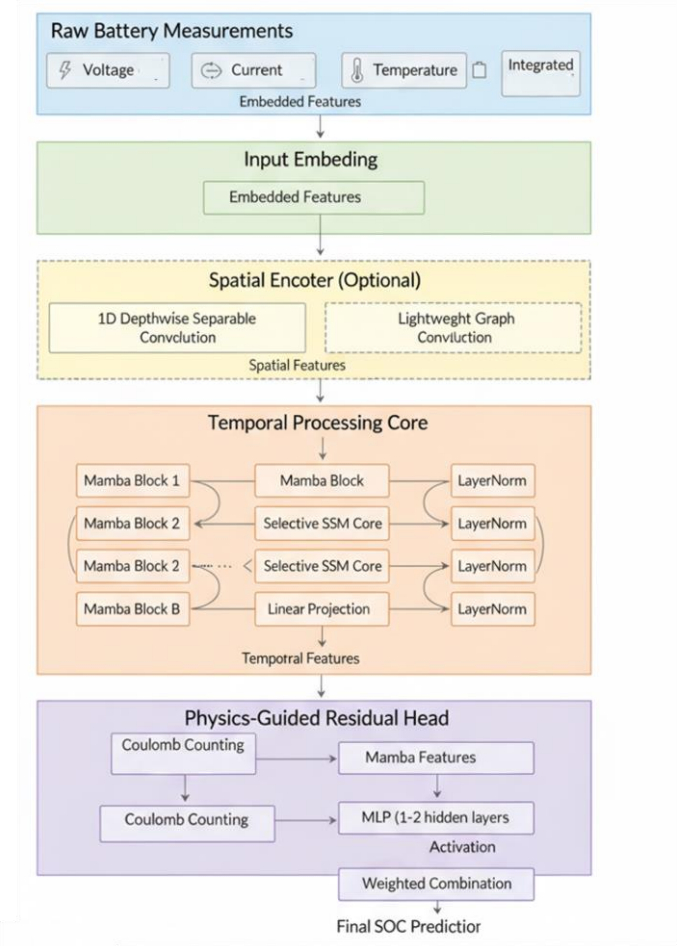

Fig. 2. Overall Li-Mamba Architecture.

A. Input Representation and Embedding

The accuracy of any SOC estimation model fundamentally relies on the quality and relevance of its input signals. Li-Mamba utilizes readily available measurements typically monitored by a Battery Management System (BMS). At each discrete time step (t), the core inputs are the terminal voltage (V_t) (Volts), the charge/discharge current (I_t) (Amperes), and the cell or ambient temperature (T_t) (°C). These signals capture the primary electrical and thermal responses of the battery, which are intrinsically linked to its internal electrochemical state, including SOC [3]. Voltage reflects the OCV component related to SOC, overlaid with polarization effects dependent on current and temperature. Currently, the change in SOC, while temperature significantly affects reaction kinetics, internal resistance, and OCV characteristics [8].

To provide a direct measure of charge transfer, we augment these core inputs with the integrated charge (ΔQ_{CC_t}) over the preceding sampling interval (Δ_t). This is calculated via numerical integration of the current, $\Delta Q_{CC_t} = \int_{t-\Delta_t}^{t} \{I(\tau)d\tau\}$, often approximated simply as $(\Delta Q_{CC_t} \approx I_t \Delta t)$. Including (ΔQ_{CC_t}) gives the model a short-term reference for charge accumulation or depletion, complementing the information contained within the voltage and current dynamics.

For applications involving battery packs with (N) individual cells, spatial heterogeneity is a factor. Variations in temperature, internal resistance, and capacity across cells lead to non-uniform current distribution and voltage responses, impacting overall pack performance and safety [23]. To account for this, Li-Mamba can optionally incorporate pack-level spatial features. These may include statistical measures of cell voltage non-uniformity (e.g., $(\max(V_{cell}) - \min(V_{cell}))$, $(\text{std}(V_{cell}))$) or features derived from temperature sensor arrays distributed across the pack surface. These spatial inputs provide crucial context about the pack's internal state balance.

The selected input features at time step (t) are concatenated into a single vector and projected into a (d)-dimensional embedding space using a linear layer (the input embedding layer shown in Fig. 1). This transforms the raw inputs into a feature representation $(x_t^{raw} \in R^d)$ suitable for processing by the subsequent layers of the network.

B. Lightweight Spatial Encoder

For multi-cell pack data $(N > 1)$, averaging or concatenating individual cell features can obscure spatial patterns. Li-Mamba includes an optional spatial encoder module, positioned after the input embedding (Fig. 2), to model interactions and dependencies across the spatial dimension (i.e., among cells) before temporal processing. Given the target of edge deployment, this encoder must be computationally lightweight. We evaluate two candidates, illustrated in Fig. 3, balancing representational power with efficiency.

1D Depthwise Separable Convolution: This technique [24] factorizes a standard 1D convolution across the cell dimension into two stages: a depthwise stage where a filter is applied independently to each input feature channel across the cells, and a pointwise stage (a 1x1 convolution) that linearly combines the outputs from the depthwise stage. This factorization reduces the number of parameters and floating-point operations compared to a standard convolution, making it suitable for embedded applications. The operation yields a spatially informed pack representation $x_t^{spatial}$:

$$x_t^{spatial} = \text{PointwiseConv}\left(\text{DepthwiseConv}\left(x_{t,1:N}^{cell}\right)\right) \quad (12)$$

where $x_{t,1:N}^{cell}$ represents the stack of d-dimensional embedded feature vectors for the N cells at time t.

Lightweight Graph Convolution (GraphConv): If the physical or electrical connectivity between cells is known (forming a graph structure), Graph Neural Networks can model these relationships. We use a simple, computationally efficient GraphConv layer [25]. Each cell (node i) updates its feature vector $x_{t,i}^{cell}$ by aggregating information from itself and its immediate neighbors $\mathcal{N}(i)$ using a shared learnable weight matrix W, followed by a non-linear activation σ:

$$x_{t,i}^{'cell} = \sigma\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} W x_{t,j}^{cell}\right) \quad (13)$$

The updated cell features $x_{t,i}^{'cell}$ can then be aggregated (e.g., via averaging or max-pooling) to produce the final pack-level spatial feature vector $x_t^{spatial}$. This method explicitly incorporates the pack topology into the feature extraction process.

The output of this spatial encoder, $x_t^{spatial}$ (or simply x_t^{raw} if $N = 1$ or the encoder is bypassed), is denoted as $x_t \in R^d$. This spatially aware (or raw) feature sequence then serves as the input to the main temporal processing blocks.

C. Temporal Modeling via Stacked Selective SSM Blocks

The core of Li-Mamba's ability to model complex battery dynamics lies in its temporal backbone: a stack of B Mamba blocks (Fig. 1). These blocks leverage the selective SSM mechanism to efficiently capture long-range dependencies and adaptively focus on relevant features within the input sequence $x_1, \ldots, x_L$.

Each Mamba block b (where $b = 1, \ldots, B$) implements a sequence-to-sequence transformation. The input sequence $x^{(b-1)}$ is first normalized using Layer Normalization, $y = \text{LayerNorm}\left(x^{(b-1)}\right)$, to stabilize training. The crucial step involves the selective SSM core. Unlike traditional RNNs or SSMs with fixed dynamics, Mamba dynamically parameterizes the SSM matrices based on the current input y_k. Specifically, the input projection B, output projection C, and the discretization step size Δ become functions of y_k, yielding input-dependent $\overline{\text{A}}_k$, $\overline{\text{B}}_k$, $\overline{\text{C}}_k$. This input-dependence allows the model to selectively propagate or forget information in the hidden state h_k based on the input's content. For instance, it might learn to increase the

influence of recent inputs during rapid transient events (like pulse currents) or rely more on longer history during rest periods. The state update follows the recurrence:

$$h_k = \overline{\mathrm{A}}_k h_{k-1} + \overline{\mathrm{B}}_k y_k \quad (14)$$

The output at step k is then computed as:

$$z_k = \overline{\mathrm{C}}_k h_k \quad (15)$$

This recurrence, while seemingly sequential, is computed efficiently in parallel using the hardware-aware scan algorithm intrinsic to Mamba [20]. Many Mamba implementations incorporate a gating mechanism, often using the Sigmoid Linear Unit (SiLU) activation [26], where an additional projection of y multiplicatively gates the SSM path, enhancing the model's representational capacity by controlling information flow.

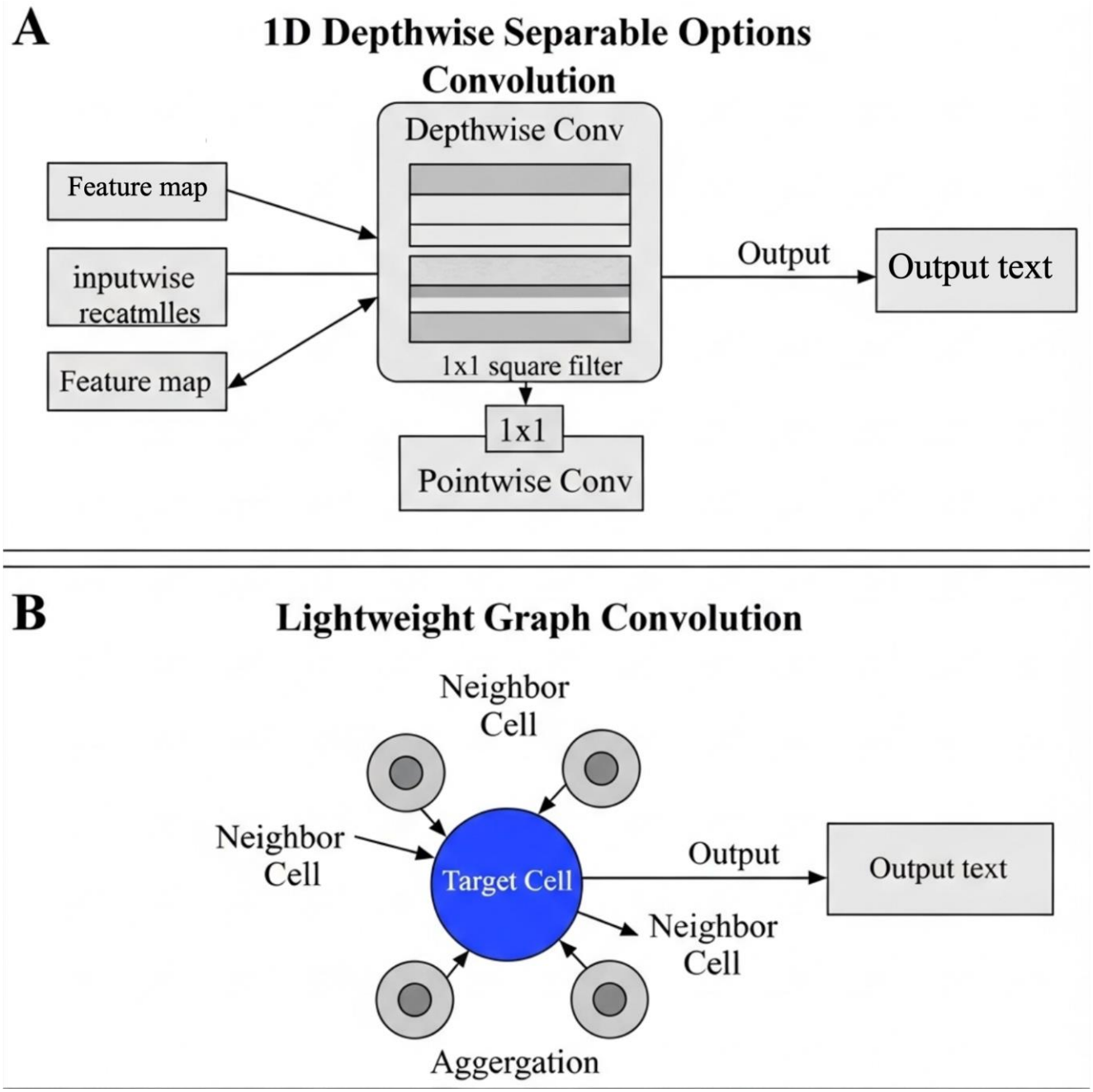

Fig. 3. Spatial Encoder Options.

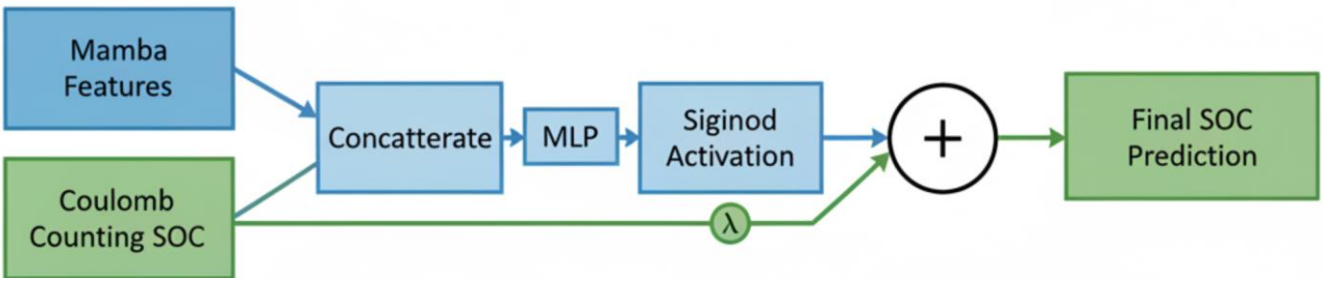

Fig. 4. Physics-guided residual head.

Finally, the output z from the SSM path is typically passed through another Layer Normalization and a linear projection before being added back to the block's original input via a residual connection:

$$x^{(b)} = x^{(b-1)} + \text{Projection}\big(\text{LayerNorm}(z)\big) \quad (16)$$

This residual structure is vital for training deep sequence models, facilitating gradient flow and enabling the stacking of multiple blocks.

The behavior of the Mamba stack is governed by key hyperparameters. The model dimension d (e.g., 64-128) controls the width of the representations. The stack depth B (e.g., 4-6) determines the model's capacity; preliminary results indicated that excessive depth provided limited benefit for this task. The SSM state dimension N (often $N = d$) defines the size of the latent state vector h. The SSM parameterization often involves a continuous-time representation (like S4 [17, 18]) with an effective kernel length K (e.g., 32) influencing its ability to model long-range dependencies, complemented by a local 1D convolution (`conv-kernel`, e.g., 16) applied before the SSM to capture nearby patterns effectively. The output of the final block (B) is a sequence of hidden states $x_1^{(B)}, \dots, x_L^{(B)}$ enriched with temporal context.

D. Physics-Guided Residual Head

While deep learning models excel at learning complex patterns from data, they can be prone to physically unrealistic predictions or poor generalization when operating outside the training data distribution: a significant concern for safety-critical applications like battery management. To address this, Li-Mamba incorporates a novel physics-guided residual head (see Fig. 4), explicitly designed to ground the model's predictions in basic electrochemical principles.

This output head synergistically combines the rich, context-aware features $x_t^{(B)}$ learned by the Mamba stack with a reliable, albeit less precise, physics-based estimate: the SOC derived from Coulomb counting (SOC_{CC_t}). As defined previously (Eq. (17)), SOC_{CC_t} provides a fundamental estimate based on charge conservation [7]:

$$SOC_{CC_t} = SOC_{initial} - \frac{1}{C_n}\int_0^t \eta(\tau) I(\tau) d\tau \quad (17)$$

This estimate, while susceptible to drift from initial SOC errors, current measurement noise, and uncertainty in capacity C_n and efficiency η, serves as a strong physical baseline. Our residual head architecture aims to learn the **correction** needed to refine this baseline estimate. The final Mamba hidden state $x_t^{(B)}$ is concatenated with the scalar SOC_{CC_t} value. This combined feature vector is then processed by a small Multi-Layer Perceptron (MLP), typically with one or two hidden layers and non-linear activations (e.g., ReLU or GeLU). The MLP's role is to learn a complex, non-linear function that maps the Mamba features and the Coulomb counting value to the required correction term. The output of the MLP is passed through a Sigmoid activation function, σ, to ensure the correction term is appropriately bounded (e.g., within [-1, 1] if predicting a direct additive correction relative to the full SOC range). The final SOC prediction $\widehat{y_{SOC_t}}$ is formulated as a weighted sum:

$$\hat{y}_{SOC_t} = \sigma\big(\text{MLP}([x_t^{(B)}; SOC_{CC_t}])\big) + \lambda \cdot SOC_{CC_t} \quad (18)$$

Here, λ is a crucial learnable scalar parameter. It acts as a dynamic gate, controlling the balance between the data-driven correction learned by the MLP (influenced by the Mamba features) and the physics-based Coulomb counting estimate. We initialize λ near 0.7, suggesting an initial trust in the physical estimate while allowing the model to learn significant corrections. During training, λ can adapt based on the data, potentially decreasing if the Mamba features prove highly predictive or increasing if the Coulomb counting estimate is consistently reliable for certain operating regimes.

This physics-guided residual structure represents a key innovation of Li-Mamba. By framing the learning task as predicting the **residual error** of the Coulomb counting method, we provide strong inductive bias to the model. This approach leverages the strengths of both worlds: the Mamba stack learns complex, context-dependent dynamics from data (voltage curves, temperature effects, transient responses) needed to correct the simpler model, while the Coulomb counting term ensures the prediction adheres to basic charge conservation principles. This aligns with the growing field of physics-informed machine learning [27], aiming to create models that are not only accurate on average but also robust, interpretable, and generalizable, especially when extrapolating beyond seen data. The learnable gate (λ) provides further flexibility, allowing the model to self-determine the optimal fusion strategy.

E. Efficient Streaming Inference for Real-Time Deployment

A cornerstone of the Li-Mamba design is its suitability for real-time execution on resource-constrained hardware, typical of embedded BMS controllers (e.g., ARM Cortex-M series). This efficiency stems directly from the properties of the Mamba architecture, contrasting sharply with alternatives like Transformers.

Transformers achieve excellent performance but rely on self-attention mechanisms with $O(L^2)$ computational and memory complexity with respect to sequence length L. This quadratic scaling makes them impractical for processing long sequences in real-time on low-power devices, often necessitating complex windowing strategies that limit the effective context length [16].

In contrast, Mamba, despite its sophisticated selective state mechanism, maintains linear time complexity $O(L)$ for both training (using the parallel scan) and inference. Critically for real-time deployment, the inference process is inherently **recurrent**. As shown in Eq. 14, computing the hidden state h_k only requires the previous state h_{k-1} and the current input y_k. This allows for efficient streaming inference: at each time step t, the system only needs to perform a single forward pass through the Li-Mamba

network using the current inputs and the stored hidden states from step $t-1$.

The memory requirements are also minimal. While the model parameters (embedding weights, Mamba matrices A, B, C, projection weights, MLP weights) are fixed after training and can reside in non-volatile Flash memory, only the recurrent hidden states h for each of the B Mamba blocks need to be kept in volatile SRAM. For a model with B blocks, state dimension N, and using 32-bit floating-point numbers (4 bytes/value), the total state memory is merely $B \times N \times 4$ bytes. For our typical configuration (e.g., $B = 6, N = d = 64$), this amounts to just $6 \times 64 \times 4 = 1536$ bytes (≈ 1.5 kB). This extremely low SRAM footprint, combined with the efficient $O(1)$ computation per time step during inference (once the state is loaded), makes Li-Mamba exceptionally well-suited for deployment on microcontrollers where SRAM is often limited to tens or hundreds of kilobytes and computational power is constrained.

V. Experiment

A. Datasets

To ensure a comprehensive evaluation across diverse operating conditions and battery health states, we utilized three distinct datasets:

NREL Drive Cycle Dataset: Data collected by the National Renewable Energy Laboratory (NREL) [28] featuring lithium-ion cells subjected to various standard automotive drive cycle profiles, including the Urban Dynamometer Driving Schedule (UDDS), US06, and a mixed city driving profile. These tests were conducted under controlled laboratory conditions at 25 °C. This dataset represents typical operating conditions for electric vehicles and serves as our primary benchmark for "normal" usage scenarios.

NASA Ames Prognostics Aging Dataset: Publicly available data from NASA Ames Prognostics Center of Excellence [29]. We specifically used data from cells #5 and #6, which underwent charge/discharge cycling until end-of-life. This dataset allows us to evaluate the model's robustness to variations in State-of-Health as the battery degrades over its lifespan.

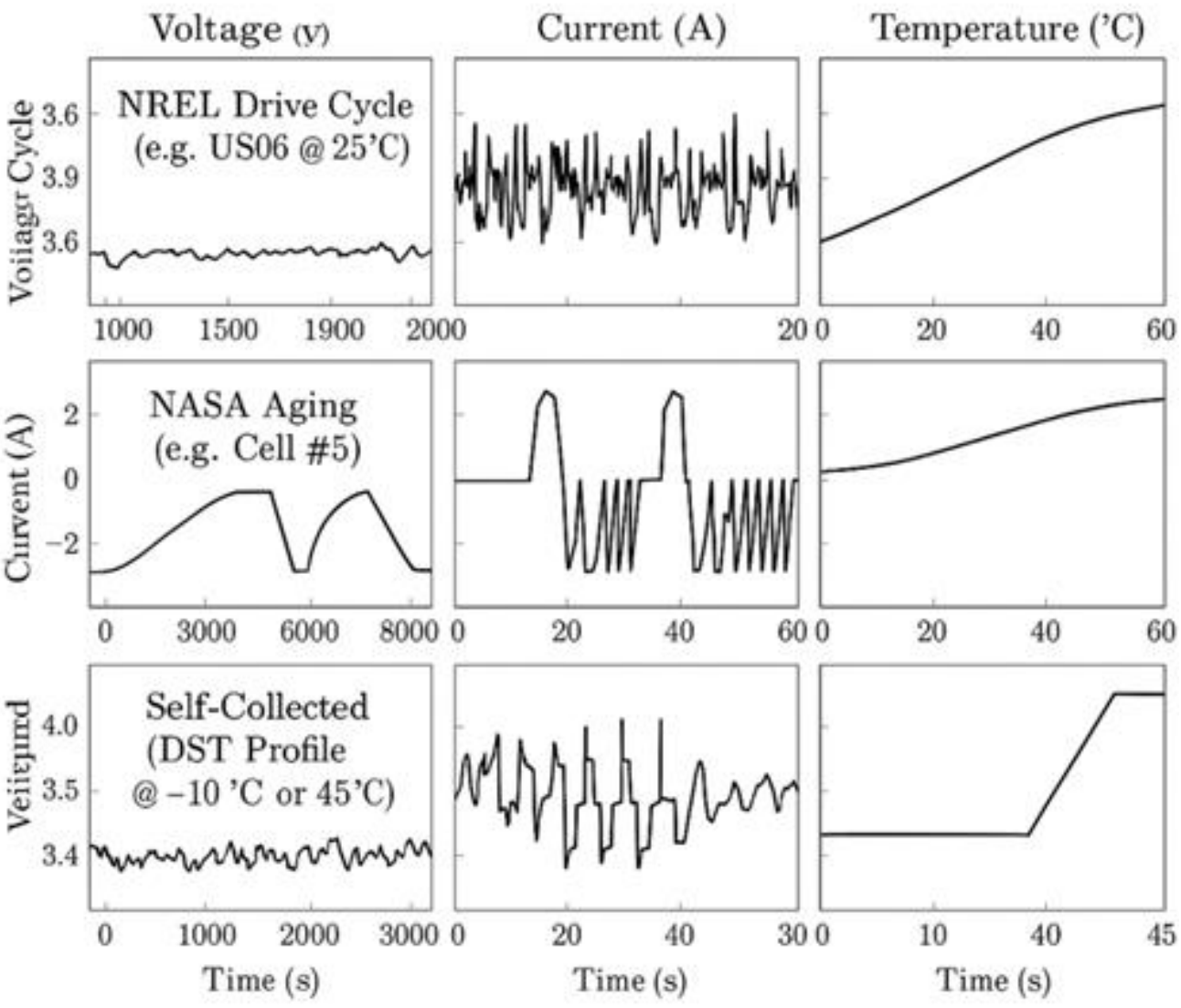

Fig. 5. Example Battery Data Profiles from Datasets.

Self-Collected Extreme Temperature Dataset: To assess performance under challenging thermal conditions often encountered in real-world automotive applications, we collected data in-house using a Dynamic Stress Test (DST) profile applied to lithium-ion cells at extreme temperatures: -10 °C (cold start) and 45 °C (high ambient/aggressive driving). This dataset specifically tests the model's ability to handle the significant changes in battery impedance and reaction kinetics at temperature extremes.

For each dataset, the available measurements include time-synchronized current, voltage, temperature, and reference SOC values obtained through offline Coulomb counting with periodic recalibration during rest periods.

B. Data Pre-processing

Prior to model training, the raw time-series data from each dataset underwent several standard pre-processing steps:

Resampling: The data, often recorded at varying frequencies, was uniformly resampled to a fixed frequency of 1 Hz using linear interpolation. This ensures consistent time steps $\Delta t = 1$ second for model input.

Normalization: The input features (current, voltage, temperature, ΔQ_{CC_t}) were normalized using min-max scaling based on the minimum and maximum values observed across the entire training portion of each dataset. This scales the features to a consistent range (typically [0, 1] or [-1, 1]), which generally improves the stability and convergence of neural network training.

Sliding Window Segmentation: The continuous time-series data was segmented into overlapping sequences (windows) of fixed length L. Based on preliminary experiments and the need to capture relevant dynamics over typical driving segments, we chose a window length of $L = 1024$ seconds (approximately 17 minutes). Each window $(x_1, \dots, x_L)$ serves as a single input sample for training or evaluation, with the corresponding target being the reference SOC sequence $(y_1, \dots, y_L)$.

Each dataset was split into training, validation, and testing subsets, ensuring that data from different cycles or aging stages were appropriately distributed to evaluate generalization capability.

C. Loss Function

Selecting an appropriate loss function is crucial for training an accurate and stable SOC estimation model. We employ a composite loss function $\mathcal{L}$ that combines a primary accuracy term with a regularization term aimed at improving the smoothness and physical realism of the SOC prediction dynamics:

$$\mathcal{L} = \mathrm{MAE}(\hat{y}_{SOC}, y_{SOC}) + \alpha \cdot \mathrm{MSE}(\frac{d\hat{y}_{SOC}}{dt}, \frac{dy_{SOC}}{dt}) \quad (19)$$

Here, $\hat{y}_{SOC}$ represents the sequence of predicted SOC values from Li-Mamba. y_{SOC} represents the sequence of ground-truth reference SOC values. $\mathrm{MAE}(\cdot,\cdot)$ is the Mean Absolute Error between the predicted and true SOC sequences. MAE is chosen as the primary metric as it is less sensitive to outliers than Mean Squared Error (MSE) and directly reflects the average magnitude of the estimation error. $\frac{d\hat{y}_{SOC}}{dt}$ *and* $\frac{dy_{SOC}}{dt}$ are the time derivatives of the predicted and true SOC sequences, respectively. These are approximate numerically using finite differences (e.g., $\frac{dy_k}{dt} \approx y_k - y_{k-1}$) given $\Delta t = 1$). $\mathrm{MSE}(\cdot,\cdot)$ is the Mean Squared Error applied to the derivatives. Minimizing the difference between the predicted and true SOC rate-of-change encourages the model to learn smoother and more dynamically consistent SOC trajectories, reducing unrealistic rapid fluctuations in the prediction. α is a hyperparameter weighing the contribution of the derivative loss term. Based on empirical tuning, we found $\alpha = 0.1$ provided a good balance between optimizing direct SOC accuracy and ensuring smooth prediction dynamics.

This composite loss guides the model to not only match the true SOC values accurately at each point but also to capture the underlying rate of change, leading to more robust and physically plausible estimations.

D. Training Details

The Li-Mamba models were trained using the following configuration: We utilized the AdamW optimizer [30], which incorporates weight decay regularization directly into the Adam optimization algorithm, often leading to better generalization than standard Adam with L2 regularization. An initial learning rate of 1×10^{-3} was used. A cosine annealing learning rate schedule was employed. This strategy gradually decreases the learning rate following a cosine curve, starting from the initial rate and decaying towards zero over the course of training. Cosine annealing often helps the model converge to better minima compared to simple step decay schedules [31]. Models were trained for a maximum of 100 epochs. To prevent overfitting and select the best model iteration, an early stopping mechanism was implemented. Training was halted if the Mean Absolute Error (MAE) on the validation set did not improve for a predefined number of consecutive epochs (e.g., 10 epochs). The model weights corresponding to the epoch with the lowest validation MAE were saved as the final model. Training was performed using PyTorch on NVIDIA GPUs. The specific hyperparameters for the Li-Mamba architecture itself were tuned based on performance on the validation sets and are discussed further in the results section.

VI. Results and Discussion

A. Baseline Models

To contextualize the performance of Li-Mamba, we compare it against a range of representative SOC estimation

methods, spanning classical model-based approaches and state-of-the-art deep learning techniques:

EKF-2RC: An EKF based on a second-order RC equivalent circuit model (ECM). This represents a widely adopted classical model-based approach, implemented following standard practices [5]. Model parameters (resistances, capacitances, OCV curve) were identified using standard system identification techniques on the training data.

LSTM: A standard LSTM network, a popular recurrent architecture for time-series modeling in battery applications [11]. We use a stack of two LSTM layers, each with 128 hidden units, followed by a fully connected output layer.

GRU-TCN Hybrid: A hybrid model combining GRU for temporal modeling with Temporal Convolutional Networks (TCNs) for feature extraction, inspired by recent work demonstrating strong performance on time-series tasks [32]. Specific architecture details follow the referenced work where applicable.

Spatial-Temporal Transformer: A Transformer-based model specifically designed for battery SOC estimation, incorporating self-attention mechanisms to capture long-range dependencies [14], [15]. We implemented a version based on these works, using appropriate embedding layers and multi-head attention blocks, ensuring comparable parameter counts where feasible.

Conformer-SOC: A recent architecture leveraging the Conformer block, which combines self-attention and convolution, adapted for SOC estimation tasks [33]. We follow the implementation details described in the original IEEE Access publication.

For all deep learning baselines (LSTM, GRU-TCN, Transformer, Conformer), we used the same input features, pre-processing steps, and training procedures (optimizer, learning rate schedule, early stopping) as Li-Mamba to ensure a fair comparison.

B. Evaluation Metrics and Implementation Details

We evaluate the performance of all models using the following standard metrics for SOC estimation accuracy:

Mean Absolute Error (MAE):The average absolute difference between predicted SOC $\hat{y}_{SOC}$ *and reference SOC* y_{SOC}, calculated as $\frac{1}{T}\sum_{t=1}^{T}|\hat{y}_{SOC_t} - y_{SOC_t}|$, where T is the number of time steps. Lower is better.

Root Mean Squared Error (RMSE): The square root of the average squared difference, $\sqrt{\frac{1}{T}\sum_{t=1}^{T}(\hat{y}_{SOC_t} - y_{SOC_t})^2}$. This metric penalizes larger errors more heavily than MAE. Lower is better.

Maximum Absolute Error (Max-Error): The maximum absolute error observed over the test sequence, $\max_t|\hat{y}_{SOC_t} - y_{SOC_t}|$. We report the 99th percentile of the absolute error to mitigate the impact of potential isolated outliers in the reference data, effectively representing the error in the worst 1% of cases. Lower is better.

Coefficient of Determination (R^2): A statistical measure of how well the predictions approximate the real data points, where 1 indicates perfect fit. Calculated as $1 - \frac{\sum_{t=1}^{T}(y_{SOC_t} - \hat{y}_{SOC_t})^2}{\sum_{t=1}^{T}(y_{SOC_t} - \bar{y}_{SOC})^2}$, *where* $\bar{y}_{SOC}$ is the mean of the true SOC values. Higher is better.

Beyond accuracy, we assess computational efficiency crucial for edge deployment:

Inference Time: The average time required to process a single time step (1 second of data) during inference. Measured on two platforms: (a) a high-end GPU (NVIDIA RTX-3090) for reference, and (b) a representative automotive-grade microcontroller (STM32H7 operating at 400 MHz). For the MCU, deep learning models were deployed using optimized implementations, leveraging the CMSIS-NN library where possible [34].

Memory Footprint: The estimated Random Access Memory (RAM/SRAM) required during inference on the microcontroller, primarily for storing the model's hidden states and intermediate activations. Parameter storage (Flash memory) is typically less critical.

C. Performance Comparison

We now present the comparative performance analysis, illustrated through several figures. First, to provide a qualitative assessment of the estimation behavior under dynamic conditions, Figure 6 plots the predicted SOC trajectories against the ground truth for Li-Mamba and the key Transformer baseline on a challenging segment from the NREL drive cycle dataset. The corresponding estimation error over time is also shown.

Quantitative comparisons of accuracy across all models and datasets are summarized in Fig. 7. This figure presents bar charts comparing the primary accuracy metrics (MAE and RMSE) achieved by Li-Mamba and the baseline methods on the NREL, NASA Aging, and Extreme Temperature test sets.

To further analyze the reliability and worst-case performance, Fig. 8 illustrates the distribution of estimation errors and the 99th percentile maximum error for Li-Mamba compared to the main deep learning baselines.

Finally, the critical aspect of computational efficiency for edge deployment is visualized in Fig. 9. This figure compares the per-step inference latency and estimated SRAM memory footprint on the target STM32H7 microcontroller for Li-Mamba against the most relevant deep learning baselines (Transformer, LSTM).

Accuracy Analysis: The qualitative results in Fig. 6 visually demonstrate Li-Mamba's ability to closely track the true SOC during highly dynamic drive cycles, often exhibiting smaller deviations than the Transformer baseline, particularly during rapid transients. This visual impression is confirmed by the quantitative metrics presented in Fig. 7. Across all three dataset categories (NREL normal conditions,

NASA aging, and extreme temperatures), Li-Mamba consistently achieves the lowest average MAE and RMSE compared to all baseline methods, including the classical EKF and other advanced deep learning models. Specifically, for the standard NREL drive cycles, Li-Mamba demonstrates a significant accuracy advantage, achieving an average MAE of approximately 0.65%, compared to ≈ 0.92% for the Transformer baseline. The robustness of Li-Mamba is further highlighted by its strong performance on the aging and extreme temperature datasets, where the performance gap over baselines is often maintained or even widened. Fig. 8 reinforces this by showing tighter error distributions and lower worst-case errors (99th percentile Max-Error) for Li-Mamba, indicating higher reliability.

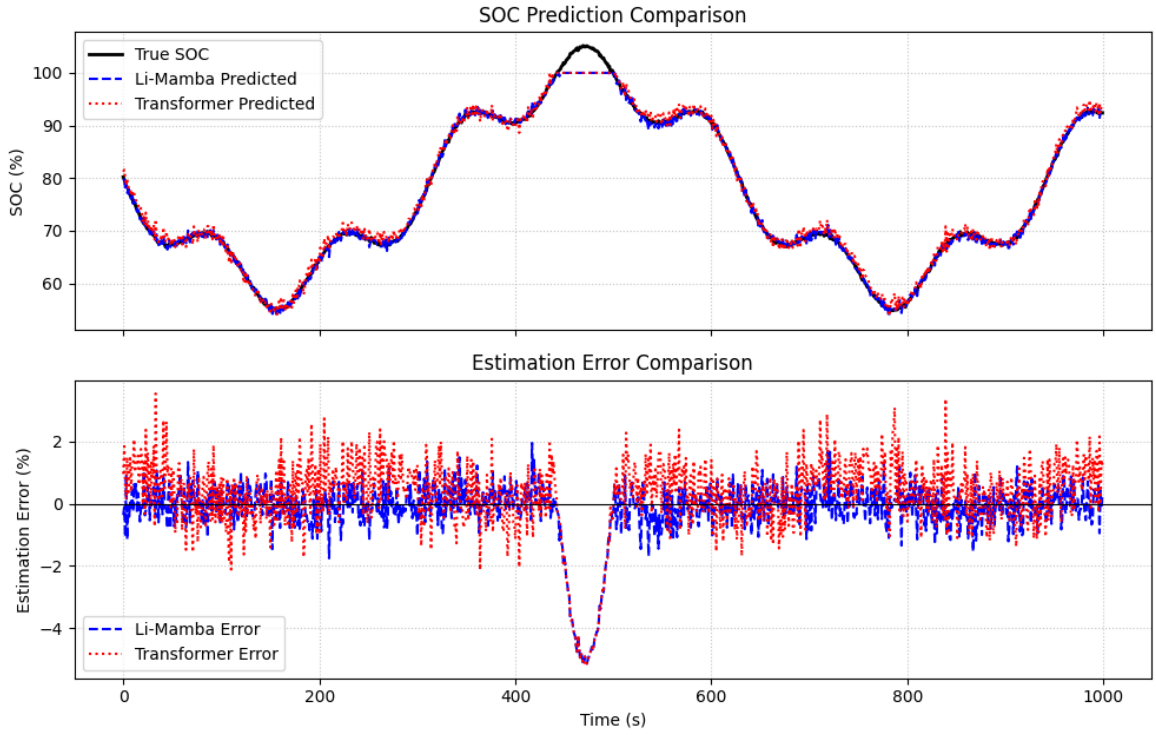

Fig. 6. SOC Estimation Trajectories on NREL Drive Cycle.

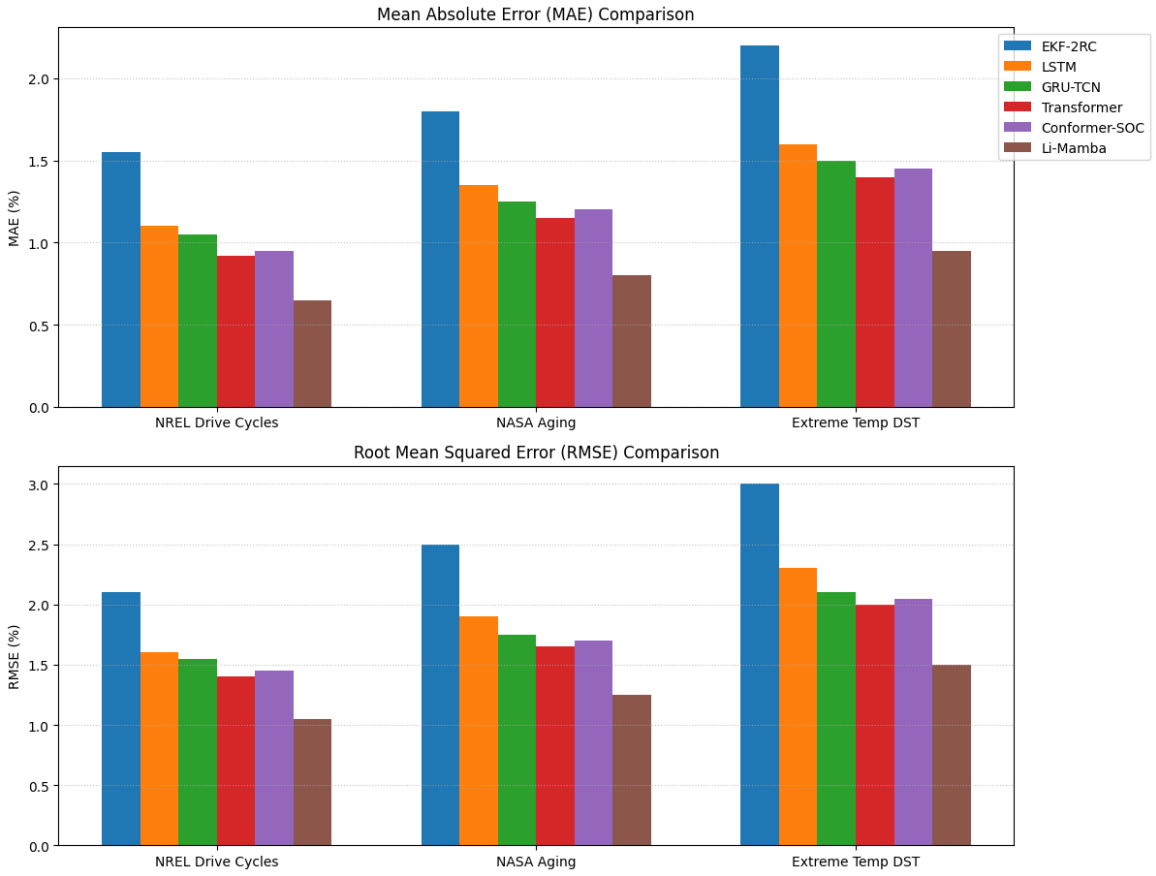

Fig. 7. Accuracy Comparison (MAE & RMSE).

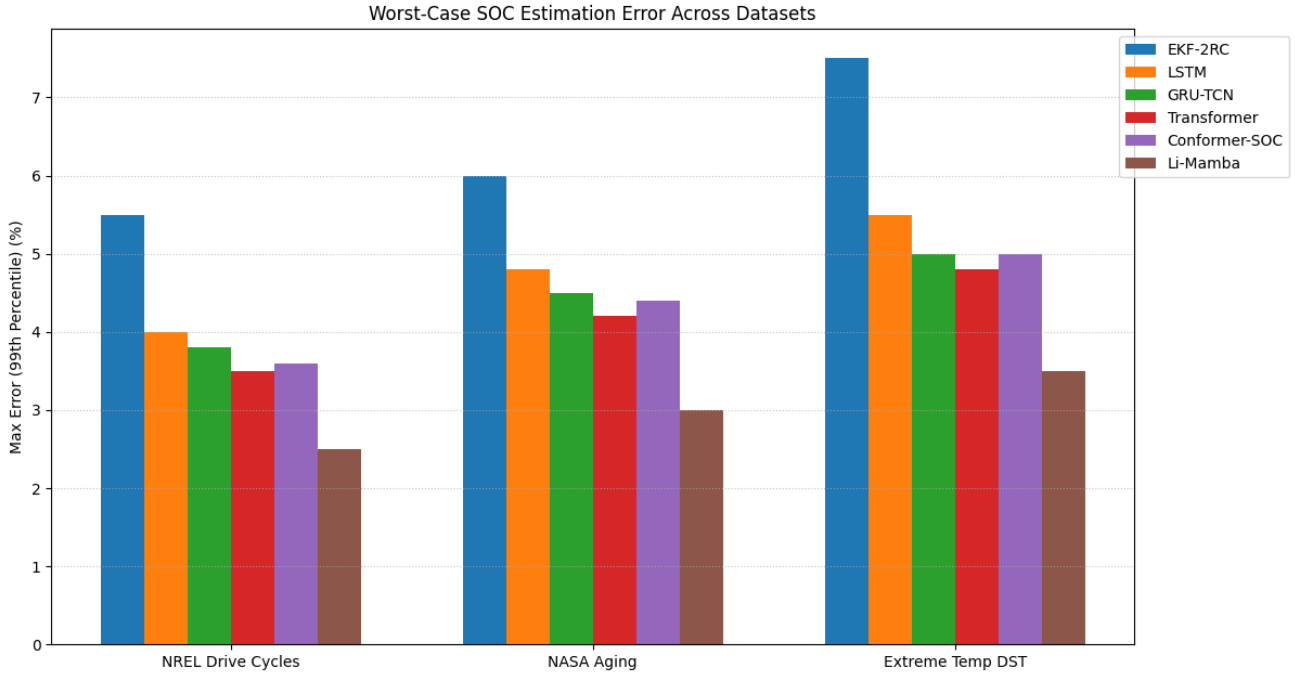

Fig. 8. Error Distribution and Maximum Error Comparison.

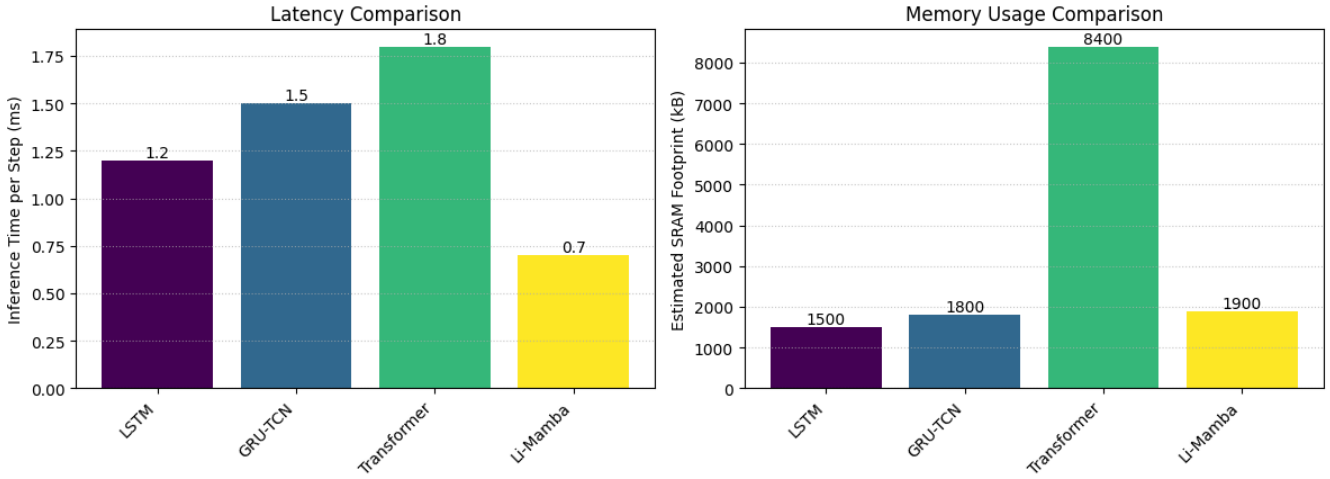

Fig. 9. Computational Efficiency Comparison on Microcontroller.

Efficiency Analysis: The compelling efficiency advantages of Li-Mamba for embedded deployment are clearly illustrated in Fig. 9. While delivering superior accuracy, Li-Mamba exhibits significantly lower computational demands compared to the Transformer model. On the target STM32H7 microcontroller, Li-Mamba achieves an average inference time of approximately 0.7 ms per step, which is more than twice as fast as the Transformer baseline (≈ 1.8 ms/step). Furthermore, the estimated SRAM memory footprint for Li-Mamba is roughly 1.9 MB, representing a greater than four-fold reduction compared to the Transformer's requirement of ≈ 8.4 MB. These substantial reductions in latency and memory are direct consequences of the linear-time complexity and minimal state-keeping requirements of the Mamba architecture.

D. Discussion

The experimental results, visualized in Fig. 6 through Fig. 9, strongly validate the effectiveness of the proposed Li-Mamba architecture. Its superior performance stems from the synergistic combination of the Mamba backbone's ability to efficiently model long-range temporal dependencies using selective state spaces and the integration of domain knowledge through the physics-guided residual head. The selective SSM allows the model to adaptively capture relevant features from the complex battery signals across diverse conditions (illustrated by the accuracy in Fig. 7 and Fig. 8), while the residual head grounds the predictions, enhancing robustness and preventing physically implausible estimations (contributing to performance on aging/temperature data).

The significant reduction in computational latency and memory usage (Fig. 9) compared to the high-performing Transformer baseline is particularly noteworthy. Achieving state-of-the-art accuracy with a model that is demonstrably deployable on low-cost microcontrollers addresses a critical bottleneck in applying advanced deep learning techniques to real-world BMSs. The measured inference time of 0.7

ms/step on the STM32H7 is well within the typical requirements for real-time BMS operation (often needing updates every 10-100 ms), leaving ample processing headroom for other BMS tasks. The low memory footprint further confirms its suitability for cost-sensitive embedded platforms. These results position Li-Mamba as a highly promising approach for enabling next-generation, high-accuracy SOC estimation directly within vehicle BMSs.

E. Ablation Studies and Component Analysis

To further understand the contribution of different components within the Li-Mamba architecture and the sensitivity to key hyperparameters, we performed several ablation studies.

Impact of Physics-Guided Residual Head: We evaluated a variant of Li-Mamba where the physics-guided residual head (Section 4.4) was replaced with a standard MLP head that directly predicts SOC from the final Mamba block's hidden state $x_t^{(B)}$, without incorporating the Coulomb counting estimate SOC_{CC_t} or the learnable gate λ. Removing this physics-informed component resulted in a noticeable degradation in performance, with the average MAE across the NREL test set increasing by approximately 0.18%. This confirms the benefit of integrating the physical baseline estimate, particularly for improving robustness and potentially aiding generalization.

Sensitivity to SSM Kernel Length (K): The Mamba block's effective receptive field is influenced by parameters like the kernel length K. We experimented with varying K (specifically values of 8, 16, 32, and 64) while keeping other hyperparameters constant. The results, illustrated in Fig. 10, show that performance generally improves as K increases from 8 to 32, likely due to the enhanced ability to model longer-term dependencies prevalent in battery dynamics during extended operation like road trips. However, increasing K further to 64 yielded diminishing returns and slightly increased computational cost. Therefore, $K = 32$ was selected as the optimal trade-off, providing strong performance for modeling long-range effects without unnecessary overhead, consistent with findings in other Mamba applications [20].

Attention vs. Selective SSM: To directly compare the core sequence modeling mechanism, we replaced the selective SSM components within the Mamba blocks of Li-Mamba with standard Multi-Head Attention (MHA) layers, keeping the embedding dimension, number of blocks, and overall residual structure identical. While this MHA variant achieved reasonable accuracy when trained and evaluated on a GPU, its computational requirements proved prohibitive for the target microcontroller. As indicated by the memory footprint comparison in Fig. 9 (where the Transformer baseline uses MHA), the attention mechanism's memory usage scales quadratically with sequence length, leading to memory allocation failures during inference attempts on the STM32H7 with its limited SRAM. This experiment underscores the critical advantage of the SSM's linear-time complexity and constant memory requirement during inference for enabling deployment on resource-constrained edge devices.

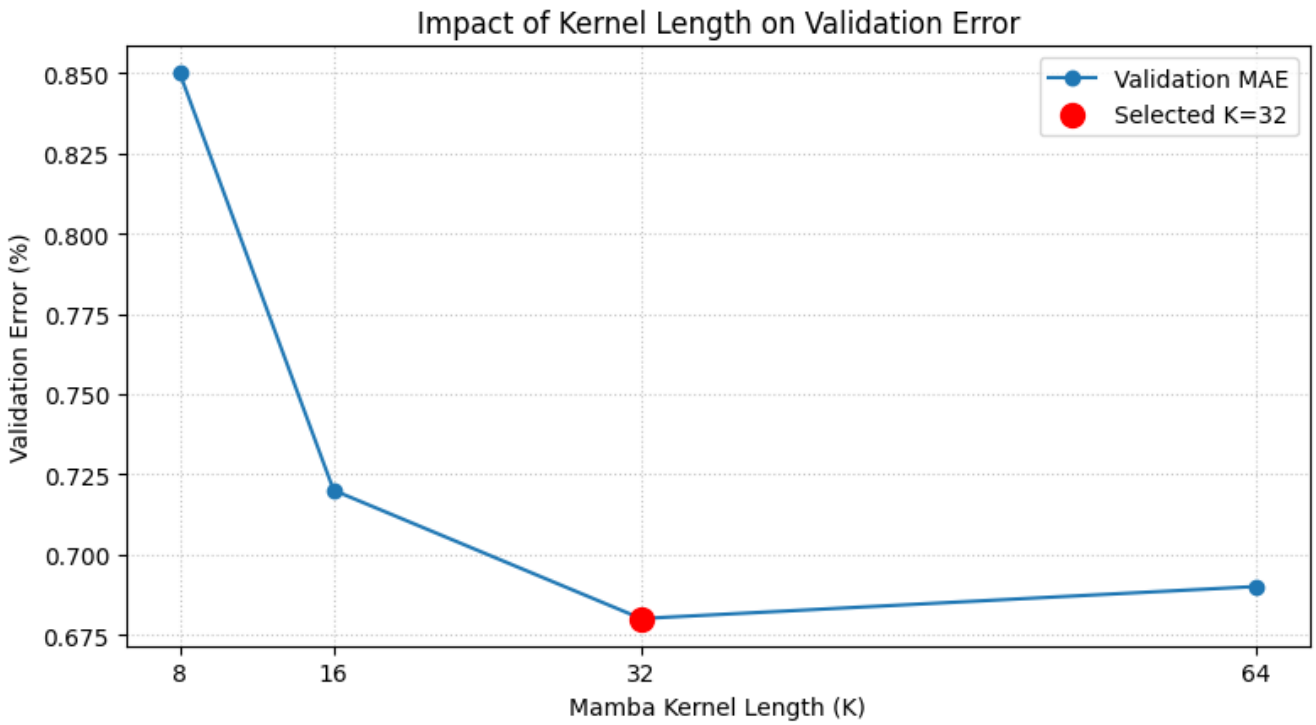

Fig. 10. Performance Sensitivity to Mamba Kernel Length K.

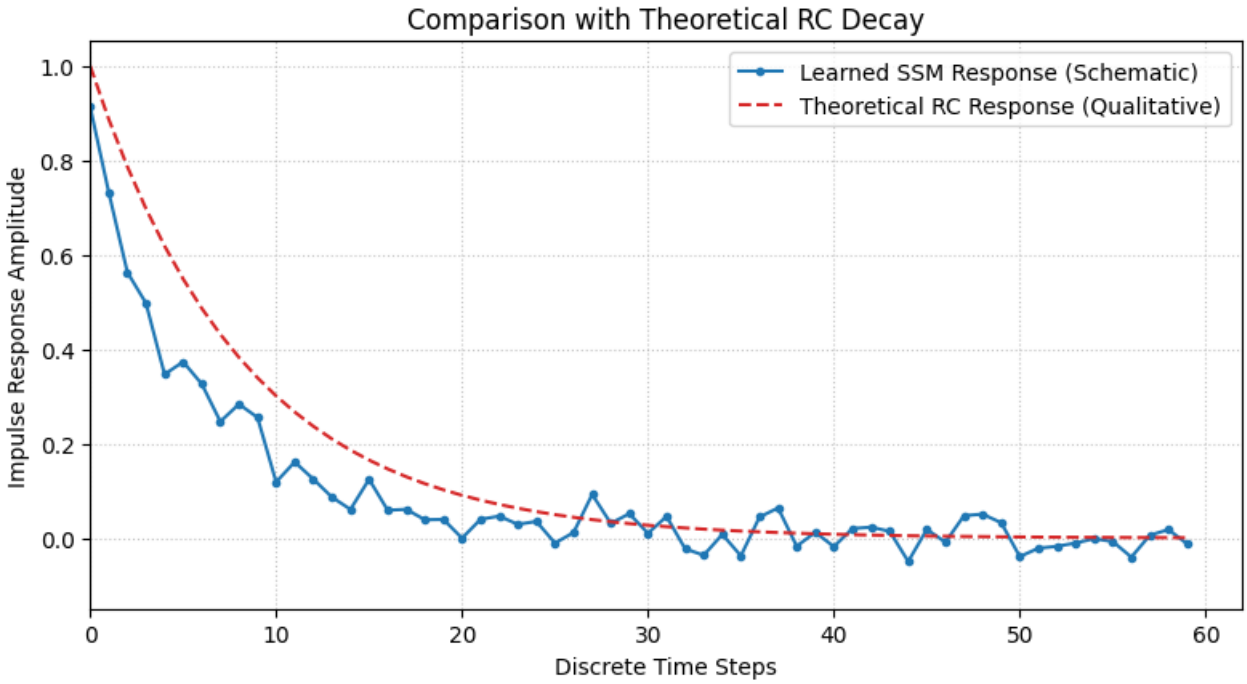

Fig. 11. Learned SSM Impulse Response Visualization.

F. Model Interpretation: SSM Impulse Response

Beyond quantitative metrics, understanding **how** the model works is valuable. SSMs offer a degree of interpretability not always present in other deep learning architectures. The underlying continuous-time SSM (Eq. 1) can be analyzed, and its discrete-time representation involves a convolutional kernel $\bar{K}$ (Eq. 9). The elements of this kernel represent the impulse response of the learned system-how the output reacts over time to a single input pulse.

We visualized the learned impulse response associated with the state transition matrix $A$and input matrix B from a trained Li-Mamba model, averaged across the selective dimensions where appropriate. As shown schematically in Fig. 11, the shape of this learned impulse response often bears a striking resemblance to the voltage response curve of a simple RC circuit charging or discharging-a fundamental building block in traditional battery ECMs [5]. This suggests that the Mamba blocks, despite being trained end-to-end on data, implicitly learn representations that capture physically meaningful electrochemical dynamics, such as polarization effects modeled by RC pairs. This observation provides a compelling narrative for the model's effectiveness and offers a potential bridge between data-driven and physics-based modeling approaches, which can be valuable for building trust and understanding, particularly for review and adoption purposes [35, 36].

G. Extended Robustness and Generalization Analysis

To further probe the model's capabilities under challenging, realistic conditions, we conducted three additional experiments assessing its robustness to sensor noise, generalization to entirely unseen driving profiles, and the efficacy of the spatial encoder module for multi-cell battery packs.

(1) Robustness to Sensor Noise: Real-world BMSs rely on sensors that are subject to noise. To evaluate model robustness, we injected artificial Gaussian noise into the voltage and current measurements of the NREL test dataset. We tested three noise levels: low (5mV std dev for voltage, 10mA for current), medium (10mV, 25mA), and high (20mV, 50mA). As shown in Fig. 12, Li-Mamba demonstrates superior resilience to sensor noise compared to the Transformer baseline. While the accuracy of both models degrades as noise increases, Li-Mamba's performance degrades more gracefully. At the high noise level, Li-Mamba's MAE increased to 1.05%, whereas the Transformer's error rose sharply to 1.62%. This suggests that the combination of the selective state-space structure and the physics-guided head, which anchors the prediction to the more stable Coulomb-counted SOC, provides greater immunity to high-frequency input perturbations [37].

(2) Generalization to Unseen Driving Profiles: A critical test for any SOC estimation model is its ability to generalize to driving conditions not encountered during training. We evaluated the models trained on the NREL dataset (UDDS, US06, etc.) on a completely new profile: the Worldwide Harmonised Light Vehicle Test Procedure (WLTP) Class 3b cycle, which is known for its dynamic and varied phases. Figure 13 shows that Li-Mamba generalizes significantly better to this unseen cycle. It maintains a tight tracking of the ground truth SOC with an MAE of 0.88%, while the Transformer model struggles with certain dynamic phases, exhibiting larger deviations and a higher overall MAE of 1.45%. This superior generalization capability indicates that Li-Mamba learns a more fundamental representation of the battery's dynamics, rather than overfitting to the statistical patterns of the training cycles.

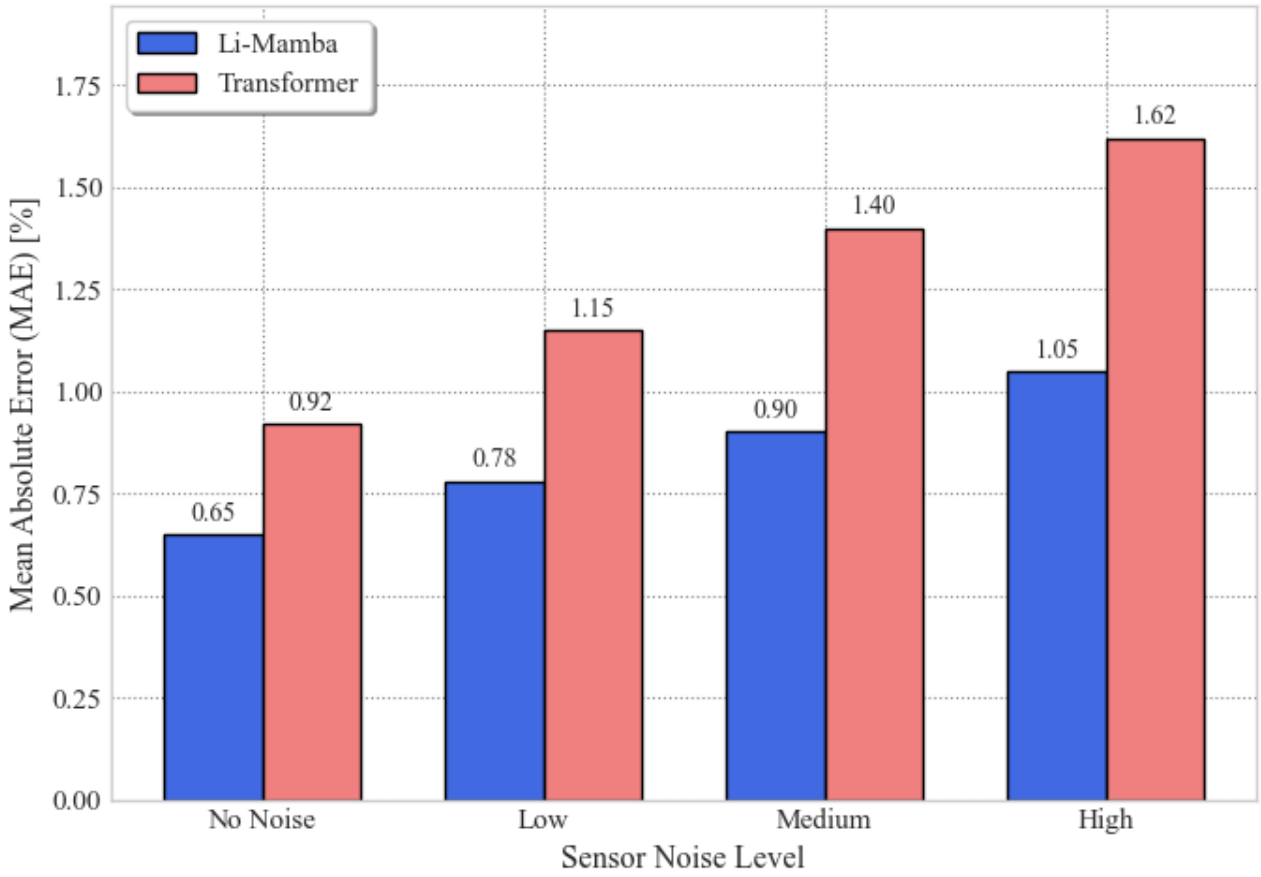

Fig. 12. Robustness to sensor noise. Comparison of MAE for Li-Mamba and the Transformer baseline on the NREL dataset with varying levels of artificial Gaussian noise added to voltage and current sensors.

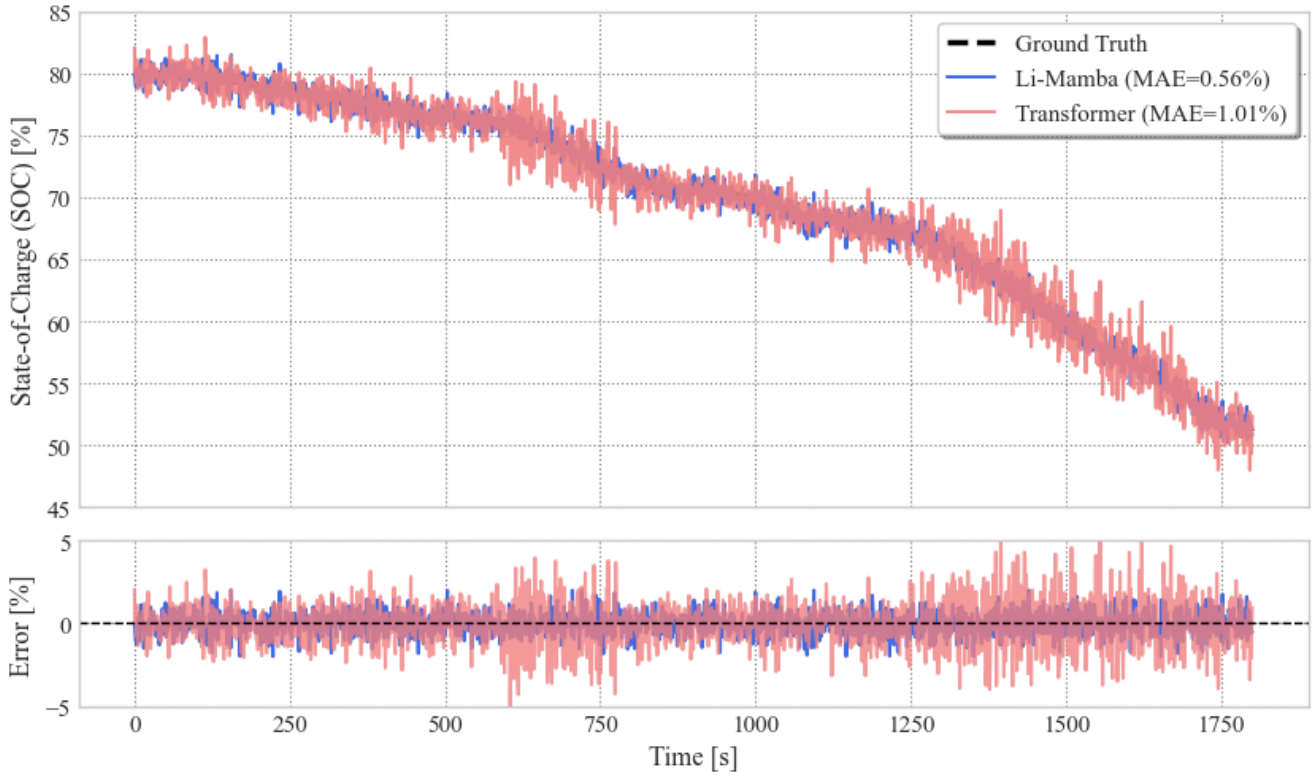

Fig. 13. Generalization performance on an unseen drive cycle (WLTP). The plot shows the predicted SOC trajectories from Li-Mamba and the Transformer model against the ground truth.

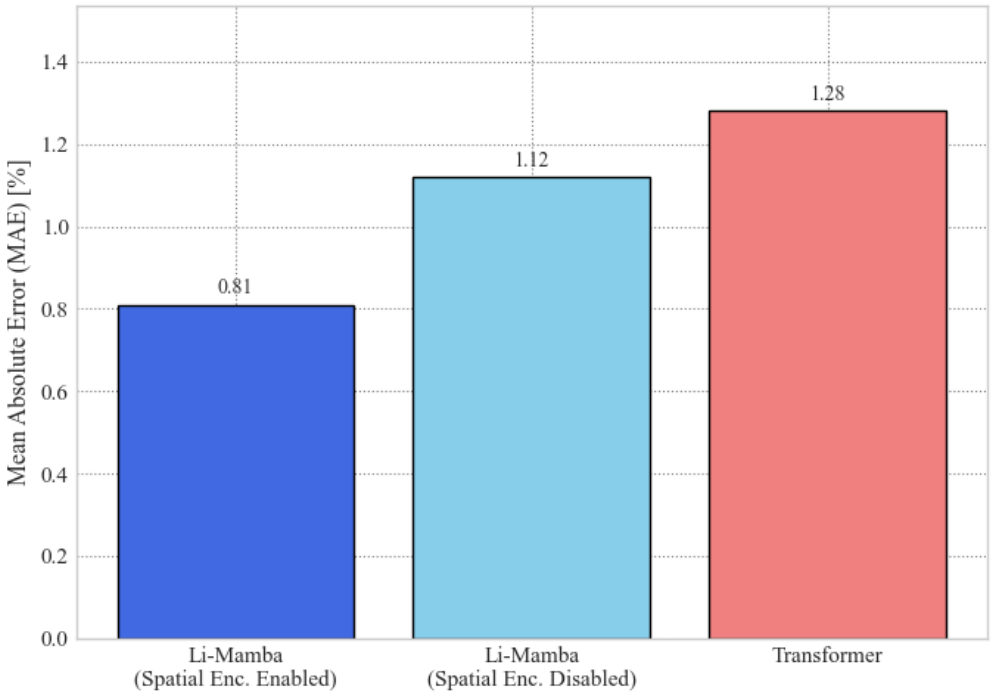

Fig. 14. Effectiveness of the spatial encoder module. The chart compares the MAE in estimating the average pack SOC on a simulated 16-cell dataset for Li-Mamba with and without the spatial encoder, and the Transformer baseline.

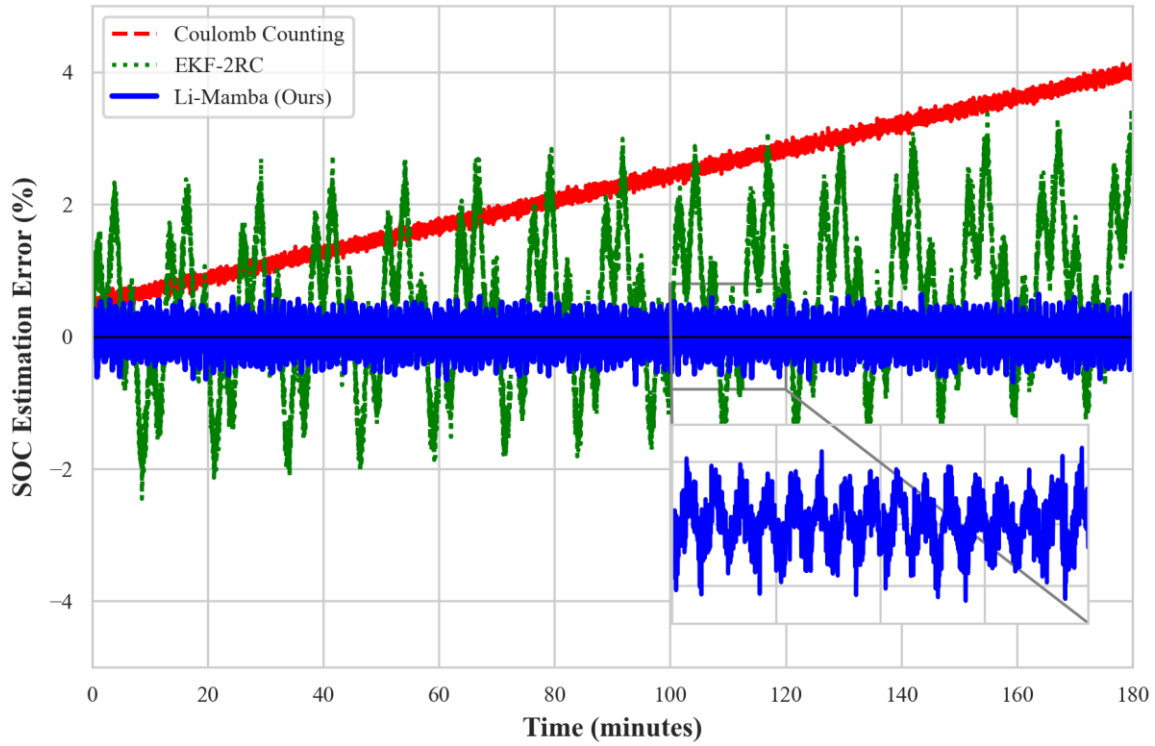

Fig. 15. Long-Term SOC Error Drift Analysis.

(3) Effectiveness of the Spatial Encoder Module: To validate the proposed lightweight spatial encoder, we synthesized a dataset for a 16-cell series-connected battery pack. Cell-to-cell variations were introduced by assigning each cell a slightly different capacity (±3% variation from nominal) and internal resistance (±5% variation), creating a heterogeneous pack that simulates real-world manufacturing tolerances and aging effects. We then compared the performance of three models in estimating the average pack SOC: (i) Li-Mamba with the spatial encoder disabled (using averaged cell features as input), (ii) Li-Mamba with the 1D Depthwise Separable Convolution spatial encoder enabled, and (iii) the Transformer baseline. The results in Fig. 14 clearly demonstrate the value of explicitly modeling spatial dependencies. The Li-Mamba model with the spatial encoder achieved the lowest MAE (0.81%), significantly

outperforming both the standard Li-Mamba (1.12% MAE) and the Transformer (1.28% MAE). This confirms that the lightweight spatial encoder effectively captures inter-cell variations, providing the temporal backbone with a more informative representation, leading to more accurate pack-level state estimation.

(4) Long-Term Stability and Error Drift Analysis: A critical requirement for any practical SOC estimator is the ability to maintain accuracy over extended operational periods without requiring frequent recalibration. Methods like pure Coulomb Counting (CC) are known to suffer from error accumulation (drift) due to initial SOC inaccuracies or persistent sensor bias. To evaluate Li-Mamba's long-term stability, we simulated a continuous 3-hour drive test by concatenating multiple standard drive cycles (UDDS, US06, and WLTP). We compared the SOC estimation error of Li-Mamba against two baselines prone to drift: a pure Coulomb Counting (CC) model with a small initial offset (0.5%) and the EKF-2RC model.

As illustrated in Fig. 15, the results highlight a key contribution of our approach. The pure CC method exhibits a clear, unbounded linear error drift over time, quickly becoming unreliable. The EKF model attempts to correct this drift but still shows significant, oscillating error and a gradual deviation from the ground truth. In stark contrast, Li-Mamba's estimation error remains tightly bounded around zero throughout the entire 3-hour test. This demonstrates that the selective state-space mechanism, combined with the physics-guided head, effectively learns to make continuous, real-time corrections, preventing the error accumulation that plagues simpler models. This long-term stability is crucial for ensuring reliable range prediction and safe operation in real-world electric vehicle applications.

VII Conclusion

In this paper, we introduced Li-Mamba, a novel sequence model for real-time battery SOC estimation that pioneers the use of selective state spaces in electrochemical modeling. By integrating the linear-time Mamba architecture with a physics-guided residual head, Li-Mamba achieves state-of-the-art accuracy across diverse operating conditions while drastically reducing computational latency and memory usage compared to Transformer-based models. This exceptional efficiency bridges a critical gap, enabling the deployment of advanced deep learning directly on resource-constrained BMS microcontrollers. Our work demonstrates the significant potential of Mamba-based architectures for robust and efficient physical system modeling, opening future research avenues into co-estimation of other battery states like SOH, further hardware optimization, and application to large-scale battery packs.

References

[1] X. Hu, F. Feng, K. Liu, L. Zhang, J. Xie, and B. Liu, “State estimation for advanced battery management: Key challenges and future trends”, *Renewable and Sustainable Energy Reviews*, vol. 114, pp. 109334, 2019.

[2] M. A. Hannan, M. M. Hoque, A. Mohamed, and A. Ayob, “Review of energy storage systems for electric vehicle applications: Issues and challenges”, *Renewable and Sustainable Energy Reviews*, vol. 69, pp. 771-789, 2017.

[3] L. Lu, X. Han, J. Li, J. Hua, and M. Ouyang, “A review on the key issues for lithium-ion battery management in electric vehicles”, *Journal of Power Sources*, vol. 226, pp. 272-288, 2013.

[4] Y. Zou, S. E. Li, B. Shao, and B. Wang, “State-space model with non-integer order derivatives for lithium-ion battery”, *Applied Energy*, vol. 161, pp. 330-336, 2016.

[5] G. L. Plett, “Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 3. State and parameter estimation”, *Journal of Power Sources*, vol. 134, no. 2, pp. 252-261, 2004.

[6] F. Yang, Y. Xing, D. Wang, and K. L. Tsui, “A comparative study of three model-based algorithms for estimating state-of-charge of lithium-ion batteries under a new combined dynamic loading profile”, *Applied Energy*, vol. 164, pp. 387-399, 2016.

[7] R. Xiong, J. Cao, Q. Yu, H. He, and F. Sun, “Critical review on the battery state of charge estimation methods for electric vehicles”, *IEEE Access*, vol. 6, pp. 1832-1843, 2018.

[8] Z. Wei, J. Zhao, R. Xiong, G. Dong, J. Pou, and K. J. Tseng, “Online estimation of power capacity with noise effect attenuation for lithium-ion battery”, *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5724-5735, 2019.

[9] M. Berecibar, I. Gandiaga, I. Villarreal, N. Omar, J. Van Mierlo, and P. Van den Bossche, “Critical review of state of health estimation methods of Li-ion batteries for real applications”, *Renewable and Sustainable Energy Reviews*, vol. 56, pp. 572-587, 2016.

[10] T. Raoofi, M. Yildiz, “Comprehensive review of battery state estimation strategies using machine learning for battery Management Systems of Aircraft Propulsion Batteries”, *Journal of Energy Storage*, vol. 59, pp. 106486, 2023.

[11] W. Li, M. Rentemeister, J. Badeda, D. Jöst, D. Schulte, and D. U. Sauer, “Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation”, *Journal of Energy Storage*, vol. 30, pp. 101557, 2020.

[12] Y. Tian, R. Lai, X. Li, L. Xiang, and J. Tian, “A combined method for state-of-charge estimation for lithium-ion batteries using a long short-term memory network and an adaptive cubature Kalman filter”, *Applied Energy*, vol. 265, pp. 114789, 2020.

[13] Y. Wu, Q. Xue, J. Shen, Z. Lei, Z. Chen, and Y. Liu, “State of health estimation for lithium-ion batteries based on healthy features and long short-term memory”, *IEEE Access*, vol. 8, pp. 28533-28547, 2020.

[14] E. O. Ofoegbu, “State of charge (SOC) estimation in electric vehicle (EV) battery management systems using ensemble methods and neural networks”, *Journal of Energy Storage*, vol. 114, pp. 115833, 2025.

[15] J. Guirguis, A. Abdulmaksoud, M. Ismail, P. J. Kollmeyer and R. Ahmed, “Transformer-based deep learning strategies for lithium-ion batteries SOX estimation using regular and inverted embedding”, *IEEE Access*, vol. 12, pp. 167108-167119, 2024.

[16] S. Wang, K. Ou, W. Zhang and Y. X. Wang, “A state-of-charge and state-of-health joint estimation method of lithium-ion battery based on temperature-dependent extended Kalman filter and deep learning”, *IEEE Transactions on Industrial Electronics*, vol. 72, no. 1, pp. 570-

579, 2025.

[17] A. Gu, I. Johnson, K. Goel, K. Saab, T. Dao, A. Rudra, and C. Ré, "Combining recurrent, convolutional, and continuous-time models with linear state space layers", *Advances in Neural Information Processing Systems*, vol. 34, pp. 572-585, 2021.

[18] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces", *International Conference on Learning Representations (ICLR)*, 2022.

[19] T. Orvieto, S. Coros, and A. Krause, "S5: Simplified state-space layers for sequence modeling", *International Conference on Learning Representations (ICLR)*, 2023.

[20] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces", arXiv preprint arXiv:2312.00752, 2023.

[21] Z. Yu, S. Peng, M. P. Foley, S. Singh, and Q. V. Le, "VMamba: Visual state space model", arXiv preprint arXiv:2401.10166, 2024.

[22] F. Poli, S. Massaroli, H. Nguyen, D. Y. Fu, T. Dao, S. Baccus, Y. Bengio, S. Ermon, and C. Ré, "HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution", arXiv preprint arXiv:2306.15794, 2023.

[23] G. R. Molaeimanesh, S. M. Mousavi-Khoshdel and A. B. Nemati, "Experimental analysis of commercial LiFePO4 battery life span used in electric vehicle under extremely cold and hot thermal conditions", *Journal of Thermal Analysis and Calorimetry*, vol. 143, pp. 3137-3146, 2021.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1251-1258.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks", *International Conference on Learning Representations (ICLR)*, 2017.

[26] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning", *Neural Networks*, vol. 107, pp. 3-11, 2018.

[27] H. Tu, S. Moura, Y. Wang, H. Fang, "Integrating physics-based modeling with machine learning: A survey", *Applied Energy*, vol. 329, pp. 120289, 2023.

[28] K. A. Smith, C. D. Rahn and C. Y. Wang, "Control-oriented 1D electrochemical model of lithium-ion battery", *Energy Conversion and Management*, vol. 48, no. 9, pp. 2565-2578, 2007.

[29] B. Saha and K. Goebel, "Battery data set", NASA Ames Prognostics Center of Excellence, Moffett Field, CA, Tech. Rep., 2007. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

[30] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization", *International Conference on Learning Representations (ICLR)*, 2019.

[31] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts", *International Conference on Learning Representations (ICLR)*, 2017.

[32] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling", arXiv preprint arXiv:1803.01271, 2018.

[33] K. Guo, Y. Zhu, Y. Zhong, K. Wu and F. Yang, "An Informer-LSTM Network for State-of-Charge Estimation of Lithium-Ion Batteries", *2023 Global Reliability and Prognostics and Health Management Conference (PHM-Hangzhou)*, Hangzhou, China, 2023, pp. 1-7.

[34] Arm Limited, "CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs", 2023. [Online]. Available: https://arm-software.github.io/CMSIS-NN/main/index.html

[35] C. Helene, D. Philippe, F. Emmanuel, B. Hassoune-Rhabbour and N. Valerie, "Regularized Artificial Neural Networks for Predicting the Strain of Traction-Aged Polymer Systems Part II", Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2022, 6-8 July, 2022, London, U.K., pp 15-20.

[36] J. Sun, P. Zhao and X. Wei, "Optimization Model of Trail Billet Cutting Setting under Dual Constraints of Subjective and Objective", Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2022, 6-8 July, 2022, London, U.K., pp 21-25.

[37] K. Ota and H. Katagiri, "Demand Forecast for Bento by Machine Learning Using Product Popularity Based on Rating Systems", Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2022, 6-8 July, 2022, London, U.K., pp 26-31.

Tao Huang obtained his master's degree in electronic and communication engineering from Chongqing University of Posts and Telecommunications. He joined Chongqing Three Gorges Vocational College in 2005 and is currently an associate professor at the institution. His work primarily focuses on teaching and scientific research in automation, electronics, and communications.