IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 20 December 2024, accepted 27 December 2024, date of publication 30 December 2024, date of current version 6 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3524321

APPLIED RESEARCH

SambaMixer: State of Health Prediction of Li-Ion Batteries Using Mamba State Space Models

JOSÉ IGNACIO OLALDE-VERANO, SASCHA KIRCH, (Graduate Student Member, IEEE), CLARA PÉREZ-MOLINA, (Senior Member, IEEE), AND SERGIO MARTÍN, (Senior Member, IEEE)

Department of Electric and Computer Engineering, UNED-Universidad Nacional de Educación a Distancia, 28040 Madrid, Spain

Corresponding author: Sascha Kirch (skirch1@alumno.uned.es)

This work was supported by the In4Labs Project funded by MICIU/AEI/10.13039/501100011033 and "European Union Next Generation EU/PRTR" under Grant TED2021-131535BI00.

ABSTRACT The state of health (SOH) of a Li-ion battery is determined by complex interactions among its internal components and external factors. Approaches leveraging deep learning architectures have been proposed to predict the SOH using convolutional networks, recurrent networks, and transformers. Recently, Mamba selective state space models have emerged as a new sequence model that combines fast parallel training with data efficiency and fast sampling. In this paper, we propose SambaMixer, a Mamba-based model for predicting the SOH of Li-ion batteries using multivariate time signals measured during the battery's discharge cycle. Our model is designed to handle analog signals with irregular sampling rates and recuperation effects of Li-ion batteries. We introduce a novel anchor-based resampling method as an augmentation technique. Additionally, we improve performance and learn recuperation effects by conditioning the prediction on the sample time and cycle time difference using positional encodings. We evaluate our model on the NASA battery discharge dataset, reporting MAE, RMSE, and MAPE. Our model outperforms previous methods based on CNNs and recurrent networks, reducing MAE by 52%, RMSE by 43%, and MAPE by 7%.

INDEX TERMS Li-ion battery, mamba, state space model, state of health prediction, multivariate time series, deep learning.

I. INTRODUCTION

Lithium-ion (Li-ion) batteries are among the most widely used energy storage solutions today, powering everything from consumer electronics to electric vehicles. They even resulted in the 2019 Nobel Prize in Chemistry [1]. Their popularity stems from their high energy density, long lifespan, and low self-discharge rate, making them both efficient and durable [2].

However, ensuring safety, reliability, and efficiency of Li-ion batteries over time requires sophisticated battery management systems (BMS) that monitor, control, and optimize battery performance. Accurate prediction of the state of health (SOH), state of charge (SOC), and remaining useful life (RUL) is essential to prevent unexpected failures and extend battery life.

Traditional BMS often rely on equivalent circuit models [3] as well as electrochemical models [4]. However, these models are limited by their complexity and sensitivity to varying operational conditions. In recent years, many data-driven approaches to predict SOH (see Section II), SOC [5], [6], or RUL [7], [8] for Li-ion batteries have been proposed. These approaches learn complex, non-linear relationships directly from data, providing more accurate, adaptive, and scalable solutions for real-time health monitoring [9].

We noticed that most recent works do not consider recent advances of deep learning [10], [11]. We acknowledge that some works [12] focus on deploying models on embedded devices to show that small deep learning-based models can be used for real-time health monitoring of Li-ion batteries.

The associate editor coordinating the review of this manuscript and approving it for publication was Sajid Ali.

At the same time, the problem of SOH prediction is a multi-disciplinary problem that requires expertise in many different disciplines, such as battery technology, signal processing, and deep learning. Some works use modern transformer architectures [13], [14], [15], which have shown great success in many deep learning disciplines, such as natural language processing and computer vision. However, these architectures are not well-suited for analog time signals with many measurement samples because of their quadratic work complexity [16] and their substantial need for resources and large datasets to train successfully [17].

Recently, Mamba [18], a selective state space model, has been proposed as a new deep learning architecture that combines the best of both worlds: fast parallel training, being able to handle long-range dependencies, fast inference and sub-quadratic work complexity. In this paper, we propose SambaMixer, a Mamba-based model to predict the SOH of Li-ion batteries from multivariate time signals measured during the battery's discharge cycle. Our model addresses the challenges of modeling long-range dependencies in time series data, handling multivariate time signals, and addressing the problem of varying signal lengths, irregular sampling rates, and recovery effects of Li-ion batteries. We evaluate our model on NASA's real-world dataset of Li-ion battery discharge cycles [19] and demonstrate its superior performance compared to state of the art deep learning models.

In this sense, we summarize our main contributions in this paper as follows:

1) Introducing Mamba selective state space models to the task of Li-ion battery SOH prediction from multivariate time signals.
2) Developing an anchor-based resampling scheme to resample time signals to a fixed length while serving as a data augmentation method.
3) Applying a time-based positional encoding scheme to address sample jitter, time signals of varying length, and recovery effects of Li-ion batteries.

We have released our code on GitHub.[1]

II. RELATED WORK

This section provides a comprehensive review of the most relevant works on the prediction of state of health for Li-ion batteries and structured state space models.

A. STATE OF HEALTH PREDICTION OF LI-ION BATTERIES

Ren and Du [20] categorizes state of health prediction methods into two classes: model-driven and data-driven methods. In this work we focus on data-driven methods.

Many studies combine recurrent networks and convolution networks to predict the SOH of Li-ion batteries. Mazzi et al. [10] use a 1D-CNN followed by BiGRU layers, utilizing measured voltage, current, and temperature signals from the NASA PCoE dataset [19]. Yao et al. [11] develop a CNN-WNN-WLSTM network with wavelet activation functions, also using the same dataset. Shen et al. [21] employ an extreme learning machine algorithm on voltage signals measured during the charging mode. Wu et al. [22] combine convolutional and recurrent autoencoders with GRU networks. Zhu et al. [23] use a CNN-BiLSTM with attention for both SOH and RUL estimation. Ren et al. [24] utilize an autoencoder feeding parallel CNN and LSTM blocks. Tong et al. [25] develop an ADLSTM network with Bayesian optimization. Tan et al. [26] propose a feature score rule for LSTM-FC networks. Crocioni et al. [12] compare CNN-LSTM and CNN-GRU networks. Li et al. [27] introduce an AST-LSTM network. Yang et al. [28] merge CNN with random forest in a CNN-RF network. Garse et al. [29] use a random forest regression and FC network in the RFR-ANN model. Chen et al. [30] tackle SOH with a self-attention knowledge domain adaptation network.

Other studies focus on transformer-based models. Feng et al. [13] introduce GPT4Battery, a large language model (LLM) fine-tuned to estimate SOH on the GOTION dataset [31]. It employs a pre-trained GPT-2 backbone, followed by a feature extractor and two heads for charging curve reconstruction and SOH estimation. Gomez et al. [14] use a temporal fusion transformer on a Toyota dataset [32], integrating Bi-LSTM layers for time series forecasting. Zhu et al. [15] develop a transformer with sparse attention and dilated convolution layers on the CALCE [33] and NASA PCoE datasets. Huang et al. [34] use singular value decomposition before inputting data into a transformer model. Nakano et al. [35] combine a CNN with a transformer model in an experimental EV, feeding voltage, current, and speed signals along with the SOC.

B. STRUCTURED STATE SPACE MODELS

Recently, state space models (SSMs) have emerged as a new sequence model architecture in the field of deep learning, challenging the performance of transformers [36] and the inference speed of RNNs. While transformers are successfully used in most fields of deep learning, their quadratic scaling law makes them challenging and expensive to use for certain tasks with long sequences. At the same time, RNNs are limited by their sequential nature and compressed state representation, which hinders efficient training through parallelization and the ability to capture long-range dependencies.

The LSSL model by Gu et al. [37] incorporated the HiPPO framework [38] into SSMs and demonstrated for the first time that a SSM's parameters can be trained with gradient based optimization. They further highlighted the duality of SSMs' recurrent and convolution representations, which allows for $O(N)$ complexity during inference in the recurrent view and parallel training using the convolution representation on modern hardware accelerators. The S4 model by Gu et al. [39] constrained its state matrix $\mathbf{A}$ to have a certain structure, enabling a more efficient construction

[1] GitHub Repo: https://github.com/sascha-kirch/samba-mixer

of the convolution kernel required for training. Subsequent work by Smith et al. [40], Gupta et al. [41], Fu et al. [42], and others [43], [44] further improved existing SSMs, ultimately leading to the development of the Mamba model by Gu et al. [18]. Mamba introduced selectivity into the SSM, enhancing its performance while maintaining sub-quadratic complexity during inference. This transformer-like performance, while being fast like RNNs during inference, makes it particularly suitable for sequential data tasks with long sequences, such as audio [45], [46], images [47], [48], [49], video [50], [51], NLP [52], segmentation [53], motion generation [54], and stock prediction [55].

Recent works have focused on the connection between attention and SSMs [56], [57] to simplify their formulation and leverage the extensive research on attention mechanisms in transformers, including hardware-aware and efficient implementations. Behrouz et al. [58] extended Mamba-like models to apply selectivity not only along tokens but also along channels, making them particularly well-suited for multivariate time signals, such as those found in the state of health prediction of Li-ion batteries.

III. PRELIMINARIES

This section provides a brief introduction to the concepts of state of health for Li-ion batteries and structured state space models.

A. STATE OF HEALTH OF LI-ION BATTERIES

Li-ion batteries are widely used in portable electronics, electric vehicles, and renewable energy storage systems due to their high energy density, long cycle life, and low self-discharge rate. The degradation of the battery's performance is often indicated by its state of health, which decreases over time due to various internal and external factors that will be detailed later in this section. The state of health of a battery is a measure of its ability to deliver the rated capacity and power compared to its initial state.

The state of health of a Li-ion battery, denoted as SOH_k, represents the battery's health as a percentage and is defined as follows:

$$SOH_k = \frac{Q_k}{Q_r} \cdot 100, \tag{1}$$

where Q_k represents the battery's current capacity at cycle k, and Q_r denotes its rated capacity.

As the battery is used and repeatedly charged and discharged, its SOH decreases with each cycle, which can be observed in the measured voltage, current, and temperature profiles. An example of this behavior is depicted in Fig. 1.

The end-of-life (EOL) of a battery is defined as the point at which the battery can no longer deliver the rated capacity and power, marking the end of its useful life. Typically, the EOL of a battery is reached when the SOH drops below a certain threshold, such as 70% of the rated capacity. It is important to note that due to recuperation effects, the SOH of a battery can increase again, surpassing the EOL threshold multiple times. In this work, we set the EOL indicator to the first cycle after the SOH drops below the threshold for the last time.

As mentioned earlier, the aging of Li-ion batteries is influenced by both internal and external factors [59]. Internal factors are related to the battery's chemical properties, while external factors include manufacturing processes, environmental conditions, and battery usage, among others.

1) INTERNAL FACTORS

Zeng and Liu [60] identified 21 potential internal factors that contribute to the degradation of a Li-ion battery's state of health. These factors can be categorized into three fundamental concepts: loss of lithium inventory (LLI), loss of active material (LAM), and increase in internal resistance. Among these three groups, the loss of lithium inventory has a significant impact on the aging process [61].

LLI factors include lithium precipitation and solid electrolyte interphase (SEI) formation. Lithium precipitation occurs at the anode during charging, where lithium ions form dendrites that can puncture the separator, causing short circuits [62]. SEI formation occurs during the first charge, reducing the available lithium ions and affecting their dynamics [63].

LAM factors primarily involve the degradation of lithium oxide at the cathode, leading to gas generation and increased internal resistance [64].

Increased internal resistance is also caused by electrode corrosion [65], electrolyte decomposition [66], and diaphragm degradation [67].

2) EXTERNAL FACTORS

External factors are categorized based on the battery's temperature, charge rate, overcharge/overdischarge level, and mechanical stresses [68], [69].

Using a battery outside its specified temperature range can affect the battery's performance in different ways. Both excessively high and low temperatures have adverse effects. High temperatures can lead to the formation of a solid electrolyte interface, degradation of the cathode, and ultimately thermal runaway [70], [71]. Low temperatures slow down the transport of lithium ions, increase internal resistance, and impact the battery's capacity [72].

Charging a battery at a high rate, meaning with a high charging current, can lead to the precipitation of ions on the anode, which is favored by the increase in temperature due to the Joule effect [73], [74]. Similarly, overcharging a battery can cause irreversible structural changes in the cathode and an increase in internal resistance [75], [76]. Overdischarging a battery can result in the dissolution of the anode material into copper ions, which can generate dendrites during the charging process [65].

In conclusion, a wide range of internal and external factors can contribute to the degradation of a Li-ion battery's state of health, making it a complex and challenging problem.

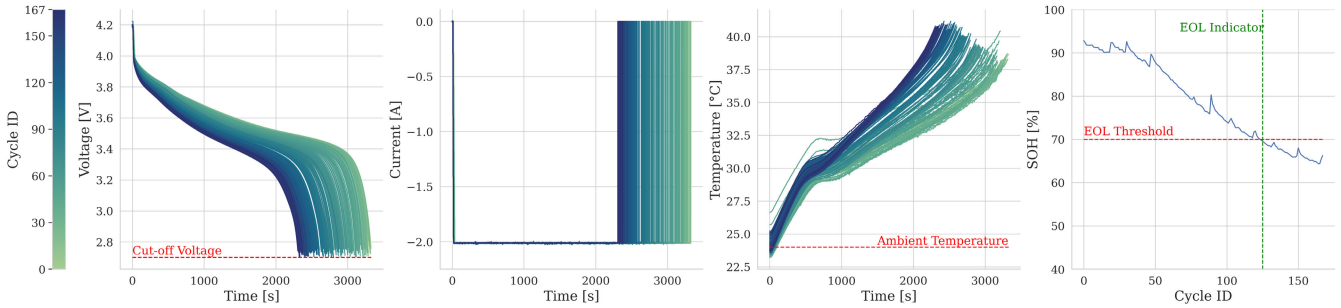

FIGURE 1. Effect of aging on the voltage, current and temperature signals of various discharge cycles of a Li-ion battery measured at the battery's terminal. We show battery #05 of NASA's battery dataset, which was cycled for 168 cycles using a constant discharge current of 2 A at an ambient temperature of 24°C.

B. STRUCTURED STATE SPACE MODELS

A state space model (SSM) describes the relationship between an input signal $x(t)$ and an output signal $y(t)$ via a hidden state $h(t)$, which evolves over time according to a linear dynamical system. The SSM is defined by the following equations:

$$\begin{aligned} h\prime(t) &= \mathbf{A}h(t) + \mathbf{B}x(t), \\ y(t) &= \mathbf{C}h(t) + \mathbf{D}x(t). \end{aligned} \tag{2}$$

Matrix $\mathbf{D}$ transforms the input $x(t)$ directly to the output $y(t)$ and is usually pulled from the SSM's equation and modeled as a skip connection. Since most applications deal with discrete signals, such as discretized analog time signals or text tokens, and the fact that the above differential equation cannot be directly solved, the SSM is discretized. This results in the following discrete-time SSM:

$$\begin{aligned} h_t &= \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t, \\ y_t &= \mathbf{C}h_t, \end{aligned} \tag{3}$$

where $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ represent the discretized state matrix and input matrix, respectively. Various discretization techniques have been employed, with the Zero Order Hold (ZOH) technique being the most prominent one in recent studies.

$$\begin{aligned} \bar{\mathbf{A}} &= e^{\triangle\mathbf{A}}, \\ \bar{\mathbf{B}} &= (\triangle\mathbf{A})^{-1}\left(\bar{\mathbf{A}} - I\right) \triangle\mathbf{B}. \end{aligned} \tag{4}$$

In other words, the discrete SSM maps an input sequence $x \in \mathbb{R}^{L\times D} = \{x_t | t \in \mathbb{N}_L\}$ to an output sequence $y \in \mathbb{R}^{L\times D} = \{y_t | t \in \mathbb{N}_L\}$ with $\mathbb{N}_L$ being the indices of the sequence with L samples and D the dimensionality of individual data points. Since matrices $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$ and $\mathbf{C}$ are constant over time, the SSM is said to be a linear time-invariant (LTI) system. In an LTI system, the recurrent representation of the SSM can be written in form of a convolution:

$$\begin{aligned} \bar{\mathbf{K}} &= \left(\mathbf{C}\bar{\mathbf{B}}, \mathbf{C}\bar{\mathbf{A}}\bar{\mathbf{B}}, \ldots, \mathbf{C}\bar{\mathbf{A}}^{L-1}\bar{\mathbf{B}}\right), \\ y &= x * \bar{\mathbf{K}}. \end{aligned} \tag{5}$$

Note that the convolution kernel $\bar{\mathbf{K}}$ is a function of the SSM matrices and contains L elements, which is quite expensive to compute for large L and dense matrices $\bar{\mathbf{A}} \in \mathbb{R}^{N\times N}$. Gu et al. [39] restricted matrix $\mathbf{A}$ to be a diagonal plus low rank (DPLR) matrix with $\mathbf{A} = \Lambda - PP^*$, which allows for a more efficient computation of the convolution kernel $\bar{\mathbf{K}}$.

To further enhance the performance of the SSM, Gu et al. [18] introduced Mamba, which adds selectivity to the SSM by making the matrices $\mathbf{B}_t$, $\mathbf{C}_t$, and $\mathbf{\Delta}_t$ time-variant. This means that each token is processed by its own matrix. It is important to note that in Mamba, the matrix $\mathbf{A}_t$ is directly optimized, while $\mathbf{B}_t$, $\mathbf{C}_t$, and $\mathbf{\Delta}_t$ are constructed from learned linear projections of the SSM's input x_t.

Behrouz et al. [58] highlighted that Mamba's selectivity applies only at the token level, but not at the channel level. This means that information cannot be passed between channels, reducing its capability to model complex relationships in multi-channel data. To address this issue, they introduced the MambaMixer, which incorporates channel-wise selectivity into the SSM. This makes it particularly suitable for handling multi-channel data, such as images or multivariate time series.

A little simplified, the MambaMixer consists of two mixing operations, the token mixer $M_{\texttt{token}}$ and the channel mixer $M_{\texttt{channel}}$, which are defined as follows:

$$\begin{aligned} M_{\texttt{token}} &: \mathbb{R}^{L\times D} \mapsto \mathbb{R}^{L\times D}, \\ M_{\texttt{channel}} &: \mathbb{R}^{D\times L} \mapsto \mathbb{R}^{D\times L}. \end{aligned} \tag{6}$$

Those mixers are built from one or more Mamba-like blocks. To obtain the output y of a single MambaMixer block, the input x is first processed by the token mixer $M_{\texttt{token}}$ and then by the channel mixer $M_{\texttt{channel}}$:

$$\begin{aligned} y_{\texttt{token}} &= M_{\texttt{token}}(x_{\texttt{token}}), \\ y_{\texttt{channel}} &= M_{\texttt{channel}}(x^T_{\texttt{channel}}), \\ y &= y^T_{\texttt{channel}}. \end{aligned} \tag{7}$$

Note that the transpose operation is necessary to apply the channel mixer along the channel dimension.

Inspired by DenseNet [77], MambaMixer further implements a learned weighted averaging of earlier blocks' outputs

to the current block's input, which is defined as follows:

$$x^{(m)}_{\texttt{token}} = \sum_{i=0}^{m-1} \alpha^{(i)}_m \, y^{(i)}_{\texttt{token}} + \sum_{i=0}^{m-1} \beta^{(i)}_m \, y^{(i)}_{\texttt{channel}},$$
$$x^{(m)}_{\texttt{channel}} = \sum_{i=0}^{m} \theta^{(i)}_m \, y^{(i)}_{\texttt{token}} + \sum_{i=0}^{m-1} \gamma^{(i)}_m \, y^{(i)}_{\texttt{channel}}, \quad (8)$$

where m is the current index of the M stacked MambaMixer blocks, $\alpha^{(i)}_m$, $\beta^{(i)}_m$, $\theta^{(i)}_m$, and $\gamma^{(i)}_m$ are learnable parameters and $y^{(0)}_{\texttt{token}} = y^{(0)}_{\texttt{channel}} = x_{\texttt{embedd}}$, where $x_{\texttt{embedd}}$ is the input to the encoder model.

IV. PROPOSED METHOD

In this section, we detail our proposed SambaMixer model to predict the state of health of Li-ion batteries. We first formulate the problem in Section IV-A. We then describe the model architecture in Section IV-B. We further detail the training procedure in Section IV-C and the sampling procedure in Section IV-D.

A. PROBLEM FORMULATION

Let $\mathbb{N}_B = \{0, 1, \ldots, \Psi - 1\}$ be the indices of Ψ different Li-ion batteries $B = \{b_\psi | \psi \in \mathbb{N}_B\}$ and $\mathbb{N}^\psi_K = \{0, 1, \ldots, K^\psi - 1\}$ be the indices of K^ψ different discharge cycles $C^\psi = \{k | k \in \mathbb{N}^\psi_K\}$ for each of the Ψ different Li-ion batteries in B. Each discharge cycle k consists of a sequence of measured samples of the current signal I_k, voltage signal V_k, temperature signal T_k and sample time S_k. All signals are measured at the battery's terminal.

$$I_k = \{i^{(k)}_t\}, V_k = \{v^{(k)}_t\}, T_k = \{\tau^{(k)}_t\}, S_k = \{s^{(k)}_t\}, \quad (9)$$

where $t \in [0, L^\psi_k) \subset \mathbb{N}$ is the index of individual samples, with L^ψ_k being the total number of samples in cycle k of battery b_ψ. Note that S_k is the sample time in seconds, where $s^{(k)}_{t=0}$ always starts at 0 s.

Through our anchor-based resampling introduced in Section IV-B1 we ensure that for all cycles in C^ψ the total number of samples are equal $L^\psi_k = L$.

By concatenating the input signals, we get the input tensor $P_k \in \mathbb{R}^{L\times 4}$ for cycle k of battery b_ψ:

$$P_k = I_k \parallel V_k \parallel T_k \parallel S_k, \quad (10)$$

where $\parallel$ denotes the concatenation operation. The objective of SambaMixer is to learn a parameterized function f_Θ that maps the input tensor P_k to the state of health SOH_k for a given cycle k of a given battery b_ψ:

$$f_\Theta : P_k \mapsto SOH_k. \quad (11)$$

B. MODEL ARCHITECTURE

Fig. 2 illustrates the top-level architecture of SambaMixer, which comprises five main components: resampling, input projection, position encoding, encoder backbone, and the prediction head.

We input a multivariate time series of current, voltage, temperature, and sample time for a single discharge cycle k of a given battery b_ψ. Our SambaMixer model then predicts the state of health SOH_k for that cycle.

1) ANCHOR-BASED RESAMPLING OF TIME SIGNALS

As the battery ages, the discharge cycles become shorter. Further, different sample rates might be chosen to sample the analog signals at the battery. Consequently, the number of samples from different discharge cycles and batteries varies significantly. Additionally, a larger number of samples leads to a wider model, which requires more resources to train. The required number of samples varies depending on the discharge mode. For instance, in a constant current discharge mode, the current remains nearly constant while the voltage continuously drops. In such cases, a smaller number of samples might be sufficient. On the other hand, high-frequency discharge profiles may require more samples to avoid anti-aliasing effects and accurately model the system dynamics.

Our anchor-based resampling technique addresses two main challenges: the need for a fixed number of samples for all cycles as expected by the model architecture and the need for data augmentation to improve the model's generalization capabilities. This augmentation of the training data with physically plausible samples is crucial and well suited in combination with our sample time position embeddings introduced in Section IV-B3.

Generally speaking, we define a resampling function f_R that resamples the sample time sequence S_k of length L^ψ_k. L^ψ_k varies for each cycle k and battery b_ψ. The result is the resampled sample-time sequence S^*_k which has the the same length L for all cycles and batteries.

$$f_R : S_k \in \mathbb{R}^{L^\psi_k} \mapsto S^*_k \in \mathbb{R}^L. \quad (12)$$

After obtaining S^*_k, we perform linear interpolation on the current, voltage, and temperature signals.

We conducted experiments using three different approaches for the resampling function f_R: linear resampling, random resampling, and our anchor-based resampling. The results are presented in Section V-D3.

In the case of linear resampling f^l_R, we take L equidistant samples between the minimum and maximum values of S_k.

$$f^l_R(S_k) := linspace(\min(S_k), \max(S_k), L). \quad (13)$$

For the random resampling f^r_R, we draw L samples from a uniform distribution $\mathcal{U}$.

$$f^r_R(S_k) := \{s^k_t\}^L_{t=0}, \; with \; s^k_t \sim \mathcal{U}_{[\min(S_k),\max(S_k)]}. \quad (14)$$

For our proposed anchor-based resampling f^a_R, we first define the anchors by using linear resampling f^l_R and then add some noise z to each anchor.

$$f^a_R(S_k) := f^l_R(S_k) + \{z_t\}^L_{t=0}, \; with \; z_t \sim \mathcal{U}_{[-\frac{w}{2},\frac{w}{2}]}, \quad (15)$$

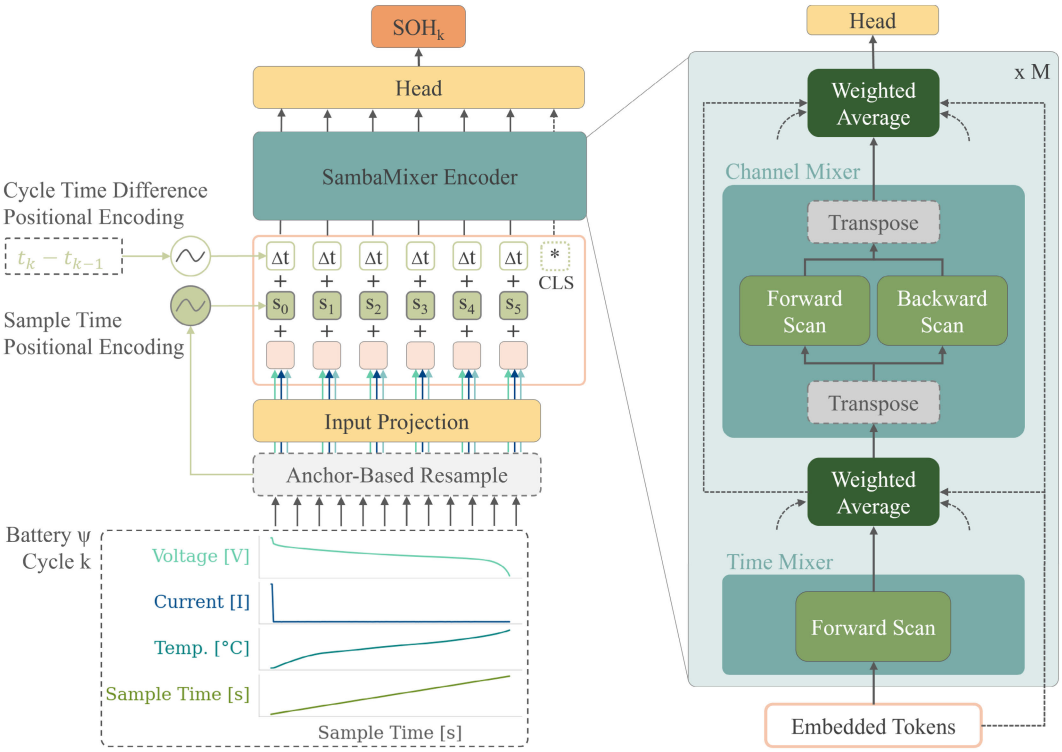

FIGURE 2. SambaMixer architecture. We input a multivariate time series of current, voltage, temperature, and sample time. First, we resample the time signals using our anchor-based resampling technique. Then, we feed the resampled sample time into the sample time positional encoding layer. Additionally, we feed the time difference between two discharge cycles in hours into the cycle time difference positional encoding layer. The other signals, namely current, voltage, and temperature, are fed into the input projection. The projected signals are added to the sample time embeddings and the cycle time difference embeddings. Optionally, a class (CLS) token can be inserted at any position. The embedded tokens are then fed into the SambaMixer Encoder, which consists of M stacked SambaMixer Encoder blocks. Finally, the output of the encoder is fed into the head, which predicts the state of health of the current cycle k for battery b_ψ.

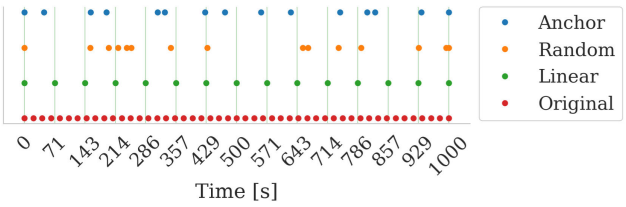

FIGURE 3. Resample techniques. (Original): The original sample time sequence with L_k^ψ samples. (Linear): linear resampling with L equidistant samples. (Random): random resampling with L samples drawn from a uniform distribution. (Anchor): anchor-based resampling with random uniform noise z added to L equidistant samples.

where w is the interval width between two linearly resampled samples. In Fig. 3 we illustrate the resulting sample time for those three resample techniques.

2) INPUT PROJECTION

The resampled voltage, current, and temperature signals are fed into the input projection. A simple linear projection layer is used to project the multivariate time signal from $\mathbb{R}^{L\times 3}$ to $\mathbb{R}^{L\times d_{\text{model}}}$.

3) SAMPLE TIME POSITION EMBEDDINGS

As shown in the top-level architecture in Fig. 2, we utilize time information in our positional encoding layer to generate position embeddings $PE^{(k)} \in \mathbb{R}^{L\times d_{\text{model}}}$ for cycle k, which are then added to the projected tokens.

In the original transformer by Vaswani et al. [36], position embeddings were added because the transformer lacks knowledge of the order of its inputs due to the absence of recurrence or convolutions. Among the various techniques available to encode absolute or relative position, the sinusoidal position embedding introduced by the transformer is still commonly used. It encodes the samples based on their absolute position p in the sequence.

$$PE_{orig}\,[p, 2i] = \sin\left(p/10.000^{2i/d_{\text{model}}}\right),$$
$$PE_{orig}\,[p, 2i+1] = \cos\left(p/10.000^{2i/d_{\text{model}}}\right). \quad (16)$$

In contrast, an SSM is a recurrent model, and inside the Mamba block, we also have a convolution. However, in VisionMamba by Zhu et al. [49], position embeddings were still added to capture the spatial position of image patches. In this work, despite applying an SSM to causal time signals,

we still include position embeddings, but instead of encoding the position of each sample, as shown in Eq. 16, we encode the sample time $s_t^{(k)}$ of cycle k at position p. This results in the positional embeddings $PE_{st}^{(k)}$.

$$PE_{st}^{(k)}[p, 2i] = \sin\left(s_{t=p}^{(k)}/10.000^{2i/d_{\text{model}}}\right),$$
$$PE_{st}^{(k)}[p, 2i+1] = \cos\left(s_{t=p}^{(k)}/10.000^{2i/d_{\text{model}}}\right). \quad (17)$$

The decision to add position embeddings is motivated by the resampling of the time signals to have equal length L. As a result, even though two samples from different cycles k are located at the same absolute position p in the sequence, they are likely sampled at different times. By encoding the sample time, the model can learn from temporal information, such as the duration of battery discharge, and become robust against variations in sample rates and number of samples.

Furthermore, Li-ion batteries can recover their capacity over time if not used. This implies that the state of health in a certain cycle k depends not only on the start time $t^{(k)}$ of the current cycle k, but also on the time difference $\Delta t^{(k)}$ in hours from the start time $t^{(k-1)}$ of the previous cycle $(k-1)$.

$$\Delta t^{(k)} := t^{(k)} - t^{(k-1)}. \quad (18)$$

Therefore, we include a second positional encoding to represent the time difference $\Delta t^{(k)}$ in hours between the start time $t^{(k)}$ of the current discharge cycle k and the start time $t^{(k-1)}$ of the previous cycle $(k-1)$. This allows the model to learn the recovery of the battery's capacity over time. We calculate the positional embeddings $PE_{\Delta}^{(k)}$ for cycle k at position p using the following formula:

$$PE_{\Delta}^{(k)}[p, 2i] = \sin\left(\Delta t^{(k)}/10.000^{2i/d_{\text{model}}}\right),$$
$$PE_{\Delta}^{(k)}[p, 2i+1] = \cos\left(\Delta t^{(k)}/10.000^{2i/d_{\text{model}}}\right). \quad (19)$$

The final positional embedding $PE^{(k)}$ for cycle k is obtained by summing the sample time positional embedding $PE_{st}^{(k)}$ and the cycle time difference positional embedding $PE_{\Delta}^{(k)}$:

$$PE^{(k)} = PE_{st}^{(k)} + PE_{\Delta}^{(k)}. \quad (20)$$

Note that the cycle time difference positional embedding $PE_{\Delta}^{(k)}$ is constant within a single cycle k while the sample time positional embedding $PE_{st}^{(k)}$ is different for each sample t in the cycle k.

We ablate different positional encoding methods in Section V-D4.

4) ENCODER BACKBONE

We stack M SambaMixer blocks to obtain our SambaMixer Encoder. The SambaMixer Encoder consists of a Time Mixer module and a Channel Mixer module, each of which consists of one or more Mamba SSM layers with different scan directions. The Time Mixer module applies the SSM along the token axis and consists of a single forward scanning SSM due to the causal nature of sequence data. On the other hand, the Channel Mixer module applies its SSMs on the channel/feature axis, which does not have this causal nature. Therefore, we apply forward and backward scanning SSMs in the Channel Mixer module.

In addition to the Time Mixer and Channel Mixer, learnable weighted average layers incorporate the results from previous layers, as described in Eq. 8.

The SambaMixer Encoder is a sequence-to-sequence model, which means that the input and output dimensions are equal. Optionally, a single learnable CLS token can be inserted before passing the sequence through the encoder, resulting in an input and output sequence of tokens of $\mathbb{R}^{d_{\text{model}} \times (L+1)}$.

Our SambaMixer Encoder is inspired by the TSM2 model proposed by Behrouz et al. [58]. Although no code was provided, we implemented the encoder backbone from scratch, following the description in the paper. We made several enhancements, including the incorporation of drop-path regularization and the clarification of normalization and activation layers, to align with current best practices and address any ambiguities in the original description.

5) REGRESSION HEAD

The regression head takes the encoded sequence of tokens from the SambaMixer Encoder as input. If a CLS token is used, the regression head selects the token that represents the encoded CLS token and projects it from $\mathbb{R}^{d_{\text{model}}}$ into $\mathbb{R}$ using an MLP. This projection is used to obtain the final prediction of the state of health for a given cycle k. It is important to note that the CLS token can be located at any position.

If no CLS token is used, we apply a mean operation to average the encoded sequence of tokens. This averaging operation produces a single token that represents the entire sequence. The token is then projected from $\mathbb{R}^{d_{\text{model}}}$ into $\mathbb{R}$ using an MLP. This projection is used to obtain the final prediction of the state of health for a given cycle k.

C. TRAINING

Similar to Mamba [18], we initialize the learned diagonal matrix $\mathbf{A}$ of the SSM using the S4D-Real [43] initialization, where $A_n = -(n+1)$. Additionally, for the bias of the linear projection that constructs the learned step size $\mathbf{\Delta}$, we follow the approach described in [44] and log-uniformly sample from the range $[\Delta_{min} = 0.001, \Delta_{max} = 0.1]$. All remaining layers, such as convolutions, linear projections, or normalization layers, are initialized with PyTorch's default initializer.

To train our SambaMixer model, we use the AdamW optimizer [78] with a learning rate of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of $5 \cdot 10^{-2}$. We employ the mean squared error (MSE) loss function and train the model for 60 epochs. A step learning rate scheduler is used, halving the learning rate every 20 epochs. During training, we randomly sample a batch of 32 discharge cycles from random batteries to predict the state of health of these cycles.

TABLE 1. Hyperparameters for our SambaMixer models of varying model size for a sequence length $L = 128$.

Model	d_{model}	d_{state}	# layer	# Param
SambaMixer-S	256	16	8	4.7 M
SambaMixer-M	512	16	8	15.2 M
SambaMixer-L	768	24	12	48.7 M
SambaMixer-XL	1024	24	12	85.6 M

We apply drop-path regularization [79] with a drop-path rate of 0.2, occasionally dropping entire mixer blocks. Additionally, we utilize mixed precision training [80] to accelerate the training process.

To ensure that all cycles have the same number of samples and to augment the data, we employ our proposed anchor-based resampling technique during training.

D. SAMPLING

To recap, our SambaMixer model takes a multivariate time series of current, voltage, temperature, and sample time from a single discharge cycle k of a battery b_ψ, along with the time difference to the previous cycle $k-1$, and predicts the state of health SOH_k for that cycle. We use the trained model to predict the SOH of a given cycle k for a specific battery b_ψ. To predict the complete capacity degradation of a battery, we iteratively predict the SOH for all cycles of the battery. In contrast to training, we use linear resampling to obtain time signals of the same length.

It is important to note that in our sampling schema, the prediction of the SOH for a cycle k is independent of the prediction of the SOH for the previous cycle $k-1$. This means that the quality of the predictions is not affected by the battery's history, such as the number of cycles it has undergone or the profile of the discharge cycle. This design choice ensures that the model performs well in realistic scenarios where the battery's history is unknown.

V. EXPERIMENTS AND ABLATIONS

In this section, we present our results. We introduce the dataset in Section V-A, followed by the metrics used for evaluation in Section V-B. We present the results of our experiments in Section V-C and the results of our ablations in Section V-D. Table 1 shows the hyperparameters for our SambaMixer models of varying model size. We consider SambaMixer-L our default model.

A. DATASET

We use the discharge cycles from the Li-ion Battery dataset provided by the NASA Ames Prognostics Center of Excellence (PCoE) [19].

As shown in Table 2, this dataset includes multiple Li-ion batteries that were tested under various discharge profiles, ambient temperatures (T_{amb}), cut-off voltages (V_{CO}), and initial capacities. All of these batteries are 18650 NCA cells with a nominal capacity of 2000 mAh and an upper voltage threshold of 4.2 V.

TABLE 2. Discharge specifications for various NASA Li-ion batteries. For the profile we report the discharge current signal form and the discharge amplitude. T_{amb} is the ambient temperature, V_{CO} is the cut-off voltage and Initial Capacity is the initial capacity of the battery at the beginning of the measurement campaign.

ID	Profile	T_{amb}	V_{CO}	Initial Capacity
#05	(const.) 2.0A	24 °C	2.7 V	1.8565 Ah
#06	(const.) 2.0A	24 °C	2.5 V	2.0353 Ah
#07	(const.) 2.0A	24 °C	2.2 V	1.8911 Ah
#18	(const.) 2.0A	24 °C	2.5 V	1.8550 Ah
#25	(PWM 0.05Hz) 4.0A	24 °C	2.0 V	1.8470 Ah
#26	(PWM 0.05Hz) 4.0A	24 °C	2.2 V	1.8133 Ah
#27	(PWM 0.05Hz) 4.0A	24 °C	2.5 V	1.8233 Ah
#28	(PWM 0.05Hz) 4.0A	24 °C	2.7 V	1.8047 Ah
#29	(const.) 4.0A	43 °C	2.0 V	1.8447 Ah
#31	(const.) 1.5A	43 °C	2.5 V	1.8329 Ah
#34	(const.) 4.0A	24 °C	2.2 V	1.6623 Ah
#36	(const.) 2.0A	24 °C	2.7 V	1.8011 Ah
#45	(const.) 1.0A	4 °C	2.0 V	0.9280 Ah
#46	(const.) 1.0A	4 °C	2.2 V	1.5161 Ah
#47	(const.) 1.0A	4 °C	2.5 V	1.5244 Ah
#48	(const.) 1.0A	4 °C	2.7 V	1.5077 Ah
#54	(const.) 2.0A	4 °C	2.2 V	1.1665 Ah
#55	(const.) 2.0A	4 °C	2.5 V	1.3199 Ah
#56	(const.) 2.0A	4 °C	2.7 V	1.3444 Ah

TABLE 3. Different training and evaluation splits for the NASA Li-ion batteries used throughout our experiments and ablations.

ID	NASA-S	NASA-M	NASA-L
#05	train	train	train
#06	eval	eval	eval
#07	eval	eval	eval
#18	-	train	train
#25	train	-	-
#26	-	-	-
#27	-	-	-
#28	-	-	-
#29	train	-	-
#31	-	-	train
#34	-	-	train
#36	-	-	train
#45	-	train	train
#46	-	train	train
#47	eval	eval	eval
#48	train	train	train
#54	-	-	train
#55	-	-	train
#56	-	-	train

Table 3 presents the various training and evaluation splits that we compiled from these batteries. NASA-S follows the same configuration as used by Mazzi et al. [10].

In our pre-processing, we remove cycles that have obvious issues with the measurement setup, such as those where the measured capacity occasionally drops to 0.0 mAh. Specifically, we filter out cycles where the state of health drops more than 10 % from one cycle to the next. Additionally, for each cycle, we remove individual samples that were recorded after the load has been disconnected. We also calculate the time between two cycles, which is needed for our positional encoding, and we resample the time signals to have a constant number of samples. During training, we use our anchor-based resampling technique introduced in Section IV-B. During inference, we use linear resampling.

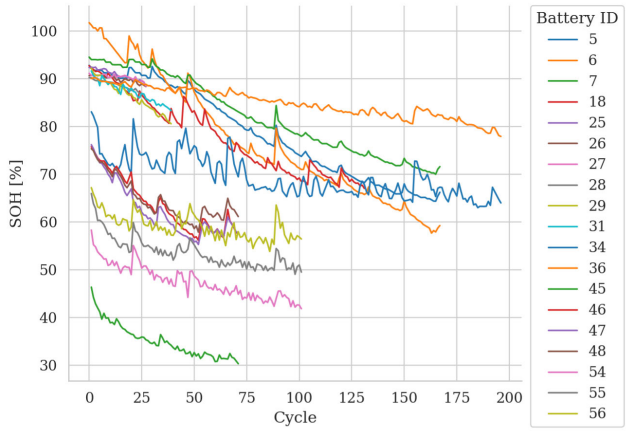

FIGURE 4. Capacity degradation as a result of repeated charge and discharge cycles reported in terms of SOH for all batteries listed in Table 3.

Fig. 4 shows the capacity degradation for all selected and pre-processed batteries, illustrating the state of health in percent over the discharge cycle ID.

B. METRICS

We evaluate our results using the commonly used metrics for state of health prediction: mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE), and absolute end of life error (AEOLE). These metrics are defined as follows:

$$\text{MAE} = \frac{1}{K}\sum_{k=1}^{K}\left|\text{soh}_k^{\text{gt}} - \text{soh}_k^{\text{pred}}\right|, \tag{21}$$

$$\text{RMSE} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(\text{soh}_k^{\text{gt}} - \text{soh}_k^{\text{pred}}\right)^2}, \tag{22}$$

$$\text{MAPE} = \frac{1}{K}\sum_{k=1}^{K}\frac{\left|\text{soh}_k^{\text{gt}} - \text{soh}_k^{\text{pred}}\right|}{\left|\text{soh}_k^{\text{gt}}\right|}, \tag{23}$$

$$\text{AEOLE} = \left|\text{eol}^{\text{gt}} - \text{eol}^{\text{pred}}\right|, \tag{24}$$

where soh_k^{gt} represents the ground truth for cycle k, $\text{soh}_k^{\text{pred}}$ denotes the predicted value for cycle k, K represents the total number of cycles, eol^{gt} represents the ground truth of the end of life indicator, and eol^{pred} represents the prediction for the end of life indicator.

C. EXPERIMENTS

In this section, we present the results of our experiments using the SambaMixer-L model trained on the NASA-L dataset. We demonstrate the estimation of state of health for the entire battery lifetime in Section V-C1. Additionally, we evaluate the performance of our model when trained on datasets of different sizes in Section V-C2. We also investigate the impact of scaling the model size and dataset size on the model's performance in Section V-C3. Finally, we analyze the performance of our model when starting the prediction at different cycle IDs to simulate pre-aged batteries in Section V-C4.

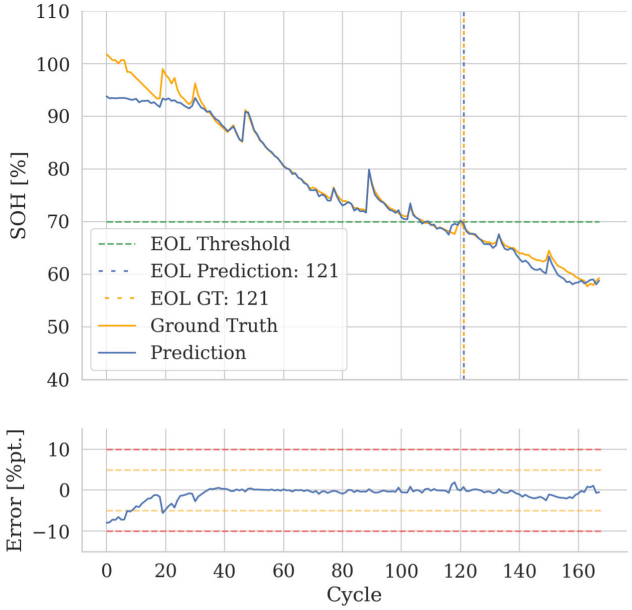

FIGURE 5. SOH prediction for battery #06. (Top) Predicted SOH vs. ground truth SOH for each cycle. (Bottom) Prediction error for each cycle.

1) SOH ESTIMATION FOR ENTIRE BATTERY LIFETIME

As described in Section IV, we input the resampled time signal from a single discharge cycle and predict the battery's state of health for that specific cycle. By sampling the model, as described in Section IV-D, we can obtain the capacity degradation over the cycle ID for each battery in the evaluation set. Figs. 5, 6, and 7 depict the comparison between the predicted SOH values and the ground truth SOH values. Additionally, we present the error for each cycle and the resulting end-of-life (EOL) indicator.

We have observed that our SambaMixer model accurately predicts the dynamics of the SOH curves and the EOL indicator without any errors for the evaluation batteries #06, #07, and #47. However, we have noticed that the model exhibits a relatively large error when predicting SOH values above 92 % for battery #06. We hypothesize that this is due to the model's limited generalization ability caused by the relatively small dataset, which does not include samples with SOH values above 92 % (refer to Fig. 8). Other Mamba-like models, such as [51] and [48], have also encountered similar issues with models overfitting easily.

Table 4 presents a comparison of our SambaMixer model with [10], [24], and [25] for each battery in the evaluation set. We observe that our SambaMixer model surpasses the works compared. Only on battery #07 MAPE is slightly worse.

2) DATASET SPLIT

In this experiment, we evaluate the performance of our SambaMixer model when trained on different training sets and compare the results with the models discussed in [10].

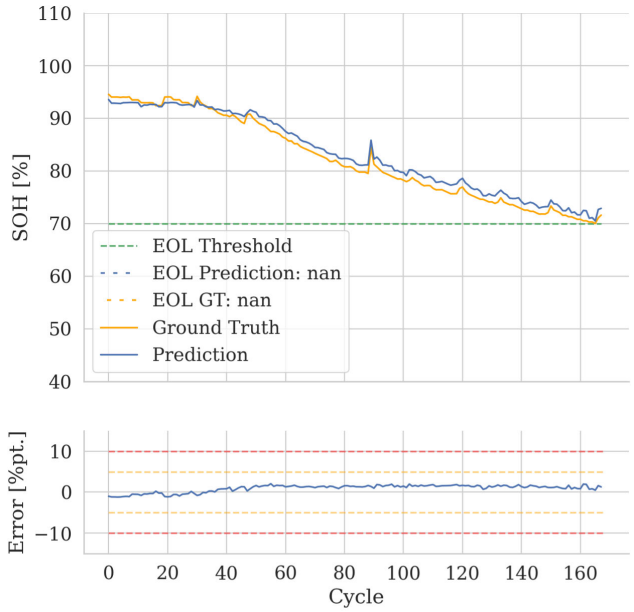

FIGURE 6. SOH prediction for battery #07. (Top) Predicted SOH vs. ground truth SOH for each cycle. (Bottom) Prediction error for each cycle.

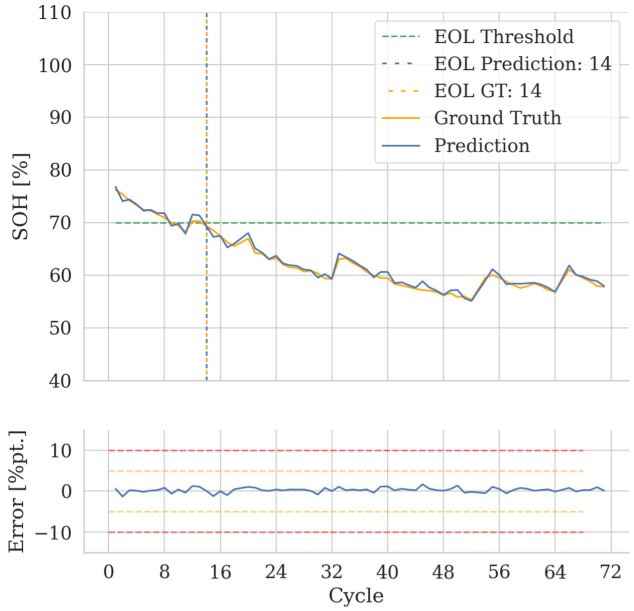

FIGURE 7. SOH prediction for battery #47. (Top) Predicted SOH vs. ground truth SOH for each cycle. (Bottom) Prediction error for each cycle.

Specifically, we train our SambaMixer-L model on NASA-S, NASA-M, and NASA-L. The results are presented in Table 5.

We observe that our SambaMixer model achieves the best performance in terms of MAE, RMSE, and MAPE for all datasets, except for MAPE on NASA-S.

3) MODEL SCALING

In this experiment, we evaluate the performance of our SambaMixer model when trained with models of different sizes.

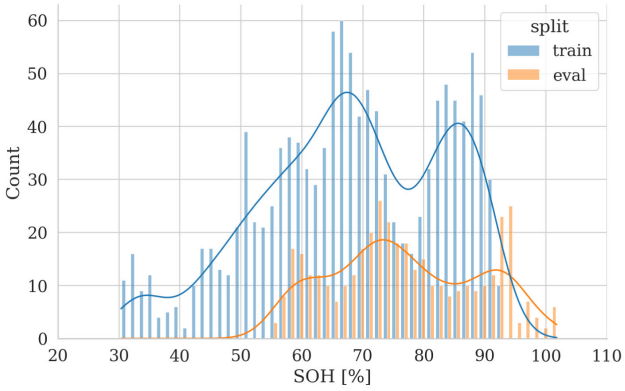

FIGURE 8. Histogram of SOH value counts. Comparison of train and eval split of the NASA-L dataset. Number of bins: 50.

TABLE 4. Comparing our SambaMixer models with the state of the art on the NASA Li-ion batteries. We report the MAE, RMSE and MAPE for each battery. Best results per evaluation battery are highlighted in bold, second best are underlined, and - are not reported.

Model	MAE↓	RMSE↓	MAPE↓
Battery #06			
ADLSTM-MC [25]	2.7	3.3	–
ANN-2 [10]	4.333	5.360	2.731
ANN-1 [10]	4.538	5.638	2.774
1D CNN [10]	3.969	4.489	2.644
BiGRU [10]	5.467	6.550	3.377
CNN-LSTM [10]	3.446	5.050	2.246
CNN-BiGRU [10]	2.448	3.177	1.579
SambaMixer-L (ours)	**1.173**	**2.068**	**1.406**
Battery #07			
Auto-CNN-LSTM [24]	–	5.03	–
ANN-2 [10]	5.413	5.682	2.573
ANN-1 [10]	5.467	7.150	3.283
1D CNN [10]	3.673	4.445	2.240
BiGRU [10]	2.233	4.218	2.022
CNN-LSTM [10]	3.188	5.123	1.910
CNN-BiGRU [10]	1.861	2.252	**1.114**
SambaMixer-L (ours)	**1.197**	**1.285**	1.498
Battery #47			
ANN-2 [10]	3.045	4.014	2.467
ANN-1 [10]	4.323	5.128	3.608
1D CNN [10]	3.379	4.319	2.644
BiGRU [10]	3.066	3.866	2.514
CNN-LSTM [10]	2.984	3.611	2.330
CNN-BiGRU [10]	2.549	3.094	1.969
SambaMixer-L (ours)	**0.512**	**0.645**	**0.822**

We train our SambaMixer-S, SambaMixer-M, SambaMixer-L, and SambaMixer-XL models on NASA-S, NASA-M, and NASA-L datasets. The results are reported in Table 6.

We observe that the performance of our model increases with the model size and the dataset size. This is expected since larger models have a greater capacity to learn complex patterns in the data, and larger datasets provide more data for the model to learn from.

Fig. 9 plots the MAE for the SOH estimation task for different model sizes and datasets. We can see that for SambaMixer-S, increasing the dataset size from NASA-M to NASA-L has almost no impact on the performance, indicating that the model is too small to benefit from the additional data.

TABLE 5. Performance of our SambaMixer model on the evaluation metrics MAE, RMSE and MAPE when trained on different training sets. Evaluation sets are the same for all datasets. Overall best results are highlighted in bold, second best are underlined.

Dataset	Model	MAE↓	RMSE↓	MAPE↓
NASA-S	ANN-2 [10]	4.565	5.267	2.621
	ANN-1 [10]	4.888	6.181	3.126
	1D-CNN [10]	3.747	4.442	2.476
	BiGRU [10]	3.718	5.128	2.668
	CNN-LSTM [10]	3.261	4.838	2.116
	CNN-BiGRU [10]	2.220	2.778	1.451
	SambaMixer-L (ours)	1.764	2.404	2.320
NASA-M	SambaMixer-L (ours)	1.334	1.902	1.641
NASA-L	SambaMixer-L (ours)	**1.072**	**1.592**	**1.346**

TABLE 6. Model scaling experiment. We report the metrics MAE, RMSE and MAPE for the SOH estimation task for different model sizes and datasets. Overall best results are highlighted in bold, second best are underlined.

Model	Dataset	MAE↓	RMSE↓	MAPE↓
SambaMixer-S	NASA-S	2.478	3.974	3.325
	NASA-M	1.920	2.829	2.461
	NASA-L	1.895	2.929	2.315
SambaMixer-M	NASA-S	1.987	2.879	2.609
	NASA-M	1.736	2.414	2.170
	NASA-L	1.230	2.027	1.493
SambaMixer-L	NASA-S	1.764	2.404	2.320
	NASA-M	1.334	1.902	1.641
	NASA-L	**1.072**	**1.592**	**1.346**
SambaMixer-XL	NASA-S	1.693	2.431	2.218
	NASA-M	1.349	1.966	1.642
	NASA-L	1.133	1.800	1.396

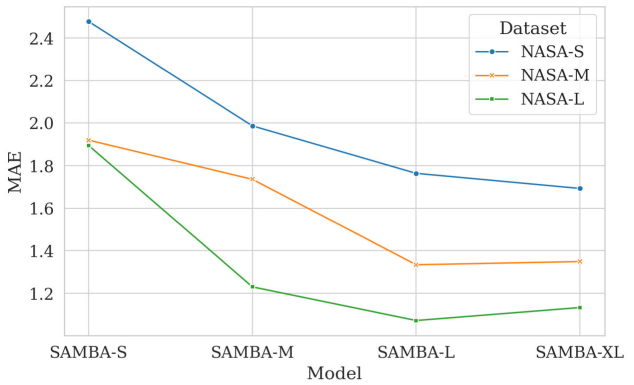

FIGURE 9. Model scaling experiment. MAE metric for the SOH estimation task for different model sizes and datasets. Values are reported in Table 6.

Furthermore, increasing the model size from SambaMixer-L to SambaMixer-XL slightly decreases the performance, suggesting that the model is too large for the dataset and likely overfits to the training data.

4) SOH ESTIMATION FOR USED BATTERIES

In a real-world scenario, it is not only necessary to predict the state of health for new batteries, but also for batteries that have been used for an unknown number of cycles or where not all discharge cycles have been recorded. A robust model is expected to reliably predict the SOH values in such scenarios.

TABLE 7. SOH estimation performance on the evaluation batteries starting at different cycle IDs. We report the metrics MAE, RMSE and MAPE for the SOH estimation task and the AEOLE for EOL indication. Capital letters in brackets for the start column represent Mazzi et al.'s [10] notation for those scenarios. Best results per evaluation battery and per start cycle are highlighted in bold, second best are underlined, and - are not reported.

Start	Model	MAE↓	RMSE↓	MAPE↓	AEOLE↓
Bat. #06					
0	CNN-LSTM [10]	3.446	5.050	2.246	–
	CNN-BiGRU [10]	2.448	3.177	1.579	–
	SambaMixer-L	**1.173**	**2.068**	**1.406**	**0**
30 (A)	CNN-LSTM [10]	4.226	4.609	2.974	8
	CNN-BiGRU [10]	2.445	3.090	1.726	**0**
	SambaMixer-L	**0.575**	**0.824**	**0.845**	**0**
70 (C)	CNN-LSTM [10]	3.850	5.056	2.975	1
	CNN-BiGRU [10]	2.080	2.516	1.650	3
	SambaMixer-L	**0.680**	**0.905**	**1.045**	**0**
100 (E)	CNN-LSTM [10]	3.771	4.379	2.945	4
	CNN-BiGRU [10]	2.440	2.859	1.901	**0**
	SambaMixer-L	**0.808**	**1.045**	**1.275**	**0**
Bat. #07					
0	CNN-LSTM [10]	3.188	5.123	1.910	–
	CNN-BiGRU [10]	1.861	2.252	**1.114**	–
	SambaMixer-L	**1.197**	**1.285**	1.498	**0**
30 (B)	CNN-LSTM [10]	2.388	3.254	1.675	–
	CNN-BiGRU [10]	1.748	2.285	**1.092**	–
	SambaMixer-L	**1.309**	**1.371**	1.665	**0**
70 (D)	CNN-LSTM [10]	1.608	2.175	1.159	–
	CNN-BiGRU [10]	1.794	2.101	**1.180**	–
	SambaMixer-L	**1.400**	**1.433**	1.839	**0**
100 (F)	CNN-LSTM [10]	1.788	2.162	1.221	–
	CNN-BiGRU [10]	1.608	1.868	**1.011**	–
	SambaMixer-L	**1.395**	**1.434**	1.878	**0**
Bat. #47					
0	CNN-LSTM [10]	2.984	3.611	2.330	–
	CNN-BiGRU [10]	2.549	3.094	1.969	–
	SambaMixer-L	**0.512**	**0.645**	**0.822**	**0**
15 (G)	CNN-LSTM [10]	3.506	4.209	2.849	–
	CNN-BiGRU [10]	2.774	3.491	2.345	–
	SambaMixer-L	**0.507**	**0.638**	**0.843**	**0**
35 (H)	CNN-LSTM [10]	2.510	3.292	2.19	–
	CNN-BiGRU [10]	2.110	2.540	1.841	–
	SambaMixer-L	**0.508**	**0.638**	**0.871**	**0**
50 (I)	CNN-LSTM [10]	2.643	3.480	2.322	–
	CNN-BiGRU [10]	1.806	2.416	1.570	–
	SambaMixer-L	**0.480**	**0.592**	**0.825**	**0**

To simulate the prediction task for used batteries, we select batteries from the evaluation set, remove the initial discharge cycles, and update their cycle ID. Specifically, for batteries #06 and #07, we conduct experiments starting the prediction at cycles 0, 30, 70, and 100. For battery #47, we experiment with starting points at cycles 0, 15, 35, and 50. In Table 7, we present our results and compare them against the two best models in [10].

We observe that SambaMixer outperforms the other models in all reported metrics for all batteries and starting points, except for the MAPE for battery #07. Our SambaMixer model performs the prediction task independently for each cycle, making it robust against missing cycles and batteries of different ages. The SOH prediction curve remains consistent, with the metrics only varying for different starting points due to normalization by the total number of cycles K for each battery.

D. ABLATION STUDY

In this section, we analyze the impact of our contributions and design choices. Unless otherwise specified, we utilize our SambaMixer-L model, which was trained on NASA-L. In Section V-D1, we examine the usage and position of the CLS tokens that can be optionally inserted into the

TABLE 8. Ablation study on usage and position of a learnable CLS token in the embedded input sequence. Best results are highlighted in bold, second best are underlined.

CLS Token Type	MAE↓	RMSE↓	MAPE↓
Tail	5.515	8.141	6.612
Middle	1.977	4.131	2.260
Head	<u>1.746</u>	<u>3.384</u>	<u>2.029</u>
None (Avg.)	**1.072**	**1.592**	**1.346**

TABLE 9. Ablation study on different backbone architectures. Best results are highlighted in bold.

Backbone	MAE↓	RMSE↓	MAPE↓
Vanilla Mamba	1.709	2.386	2.161
SambaMixer-L (ours)	**1.072**	**1.592**	**1.346**

input token sequence. In Section V-D2, we evaluate the performance of our SambaMixer backbone and compare it with a vanilla Mamba backbone from [18]. We further investigate the performance of various resampling techniques in Section V-D3. Finally, we assess the performance of different input projections and position encodings in Section V-D4.

1) USAGE AND POSITION OF CLASS TOKEN

We ablate the usage and the potential position of CLS tokens inserted into the token sequence. We train our SambaMixer-L model on NASA-L inserting a CLS token either at the tail, middle or head and compare it with a model that inserts no CLS token. If we use a CLS token, the head is attached to the position at the output that corresponds to the position where the CLS token was placed. If no CLS token is used, we average the output of all output tokens and feed it to the regression head. The results are reported in Table 8.

2) BACKBONE

In this ablation, we compare the performance of our SambaMixer backbone with the Vanilla Mamba backbone proposed by Gu et al. [18]. Both models are trained on the NASA-L dataset, and the results are presented in Table 9. The main objective of this ablation study is to demonstrate the effectiveness of our SambaMixer backbone in handling multivariate time signals.

We observe that our SambaMixer backbone outperforms the vanilla Mamba backbone. This is because the SambaMixer backbone is specifically designed to handle multivariate time signals and effectively capture the complex relationships among the variables in the dataset.

3) RESAMPLING

In this ablation study, we compare the performance of various resampling methods. We train our SambaMixer-L model on NASA-L using linear, random, and our proposed anchor-based resampling. The results are presented in Table 10. The objective of this ablation study is to demonstrate the effectiveness of our anchor-based resampling method introduced in Section IV-B1.

TABLE 10. Ablation study on various resampling methods. Best results are highlighted in bold, second best are underlined.

Resample Type	MAE↓	RMSE↓	MAPE↓
Linear	<u>1.272</u>	<u>1.862</u>	<u>1.631</u>
Random	3.315	4.368	4.302
Anchors (ours)	**1.072**	**1.592**	**1.346**

TABLE 11. Ablation study on various positional encoding methods. Best results are highlighted in bold, second best are underlined.

Encoding Type	MAE↓	RMSE↓	MAPE↓
No Encoding	3.097	3.966	4.257
Sample Time	<u>1.160</u>	<u>1.721</u>	<u>1.450</u>
Sample Time + Cycle Diff	**1.072**	**1.592**	**1.346**

Our anchor-based resampling method outperforms the linear and random resampling methods. We hypothesize that this is due to the fact that the anchor-based resampling acts as a form of data augmentation, allowing the model to learn more robust features from the data.

4) POSITIONAL ENCODING

In this ablation, we compare the performance of different positional encoding methods to justify our choice of the sample time positional encoding introduced in Section IV-B3. We train our SambaMixer-L model on NASA-L using no encoding, sample time encoding, and our proposed combined sample time and cycle time difference encoding. The results are shown in Table 11.

The addition of our proposed positional encoding to the model significantly enhances its performance. Furthermore, incorporating the time difference between discharge cycles as an additional feature to the positional encoding further boosts the performance. The rationale behind this is that capturing the difference between discharge cycles is crucial for capturing the recuperation effects of the battery and adjusting the prediction accordingly.

VI. CONCLUSION

We present SambaMixer, a novel approach for predicting the state of health of Li-ion batteries using a Mamba-based selective state space model. Our model demonstrates superior performance compared to the state of the art on the NASA battery discharge dataset, achieving a 52% reduction in MAE, a 43% reduction in RMSE, and a 7% reduction in MAPE. To enhance the performance of our model, we introduce a novel anchor-based resampling method and incorporate sample time and cycle time difference positional encoding. Our results indicate that our model achieves high accuracy and robustness in predicting the state of health of Li-ion batteries. It is capable of extracting information from multivariate time series data and modeling recuperation effects.

A. LIMITATIONS

While our model outperforms the state of the art on the NASA battery discharge dataset, we acknowledge the

limitations of our evaluation, which was confined to a single dataset - the NASA battery discharge dataset. This dataset exclusively comprises batteries of the same chemistry, and our experiments have primarily focused on constant discharge cycles.

B. FUTURE WORK

Future work should focus on evaluating our model on various datasets and different battery chemistries to further validate its generalization capabilities. Additionally, different discharge profiles' impact on the model's performance should be investigated. Different model architectures and state space models should be explored to further enhance the performance of our model.

REFERENCES

[1] A. Fernholm, "The Nobel prize in chemistry 2019," Roy. Swedish Acad. Sci., Stockholm, Sweden, Tech. Rep., Oct. 2019. [Online]. Available: https://www.nobelprize.org/prizes/chemistry/2019/press-release/

[2] M. Li, J. Lü, Z. Chen, and K. Amine, "30 years of lithium-ion batteries," *Adv. Mater.*, vol. 30, no. 33, Jun. 2018, Art. no. 1800561.

[3] G. Liu, L. Lu, H. Fu, J. Hua, J. Li, M. Ouyang, Y. Wang, S. Xue, and P. Chen, "A comparative study of equivalent circuit models and enhanced equivalent circuit models of lithium-ion batteries with different model structures," in *Proc. IEEE Conf. Expo Transp. Electrific. Asia–Pacific*, Aug. 2014, pp. 1–6.

[4] M. Elmahallawy, T. Elfouly, A. Alouani, and A. M. Massoud, "A comprehensive review of lithium-ion batteries modeling, and state of health and remaining useful lifetime prediction," *IEEE Access*, vol. 10, pp. 119040–119070, 2022.

[5] L. Ma and T. Zhang, "Deep learning-based battery state of charge estimation: Enhancing estimation performance with unlabelled training samples," *J. Energy Chem.*, vol. 80, pp. 48–57, Feb. 2023.

[6] L. Ma, C. Hu, and F. Cheng, "State of charge and state of energy estimation for lithium-ion batteries based on a long short-term memory neural network," *J. Energy Storage*, vol. 37, Mar. 2021, Art. no. 102440.

[7] L. Ma, J. Tian, T. Zhang, Q. Guo, and C. Y. Chung, "Enhanced battery life prediction with reduced data demand via semi-supervised representation learning," *J. Energy Chem.*, vol. 101, pp. 524–534, Oct. 2024.

[8] Y.-M. Tang, S. Zhong, L. Wang, Y. Zhang, and Y. Wang, "Remaining useful life prediction of high-capacity lithium-ion batteries based on incremental capacity analysis and Gaussian kernel function optimization," *Sci. Rep.*, vol. 14, no. 1, p. 23524, Oct. 2024.

[9] S. Gupta and P. Kumar Mishra, "Estimation of SoC, SoH and RUL of Li-ion battery: A review," in *Proc. 5th Int. Conf. Energy, Power Environ., Towards Flexible Green Energy Technol. (ICEPE)*, Jun. 2023, pp. 1–6.

[10] Y. Mazzi, H. B. Sassi, and F. Errahimi, "Lithium-ion battery state of health estimation using a hybrid model based on a convolutional neural network and bidirectional gated recurrent unit," *Eng. Appl. Artif. Intell.*, vol. 127, Oct. 2023, Art. no. 107199.

[11] Q. Yao, X. Song, and W. Xie, "State of health estimation of lithium-ion battery based on CNN–WNN–WLSTM," *Complex Intell. Syst.*, vol. 10, no. 2, pp. 2919–2936, Jan. 2024.

[12] G. Crocioni, D. Pau, J.-M. Delorme, and G. Gruosso, "Li-ion batteries parameter estimation with tiny neural networks embedded on intelligent IoT microcontrollers," *IEEE Access*, vol. 8, pp. 122135–122146, 2020.

[13] Y. Feng, G. Hu, and Z. Zhang, "GPT4Battery: An LLM-driven framework for adaptive state of health estimation of raw Li-ion batteries," Jan. 2024, *arXiv:2402.00068*.

[14] W. Gomez, F. Wang, and J. Chou, "Li-ion battery capacity prediction using improved temporal fusion transformer model," *Energy*, vol. 296, Mar. 2024, Art. no. 131114.

[15] X. Zhu, C. Xu, T. Song, Z. Huang, and Y. Zhang, "Sparse self-attentive transformer with multiscale feature fusion on long-term SOH forecasting," *IEEE Trans. Power Electron.*, vol. 39, no. 8, pp. 10399–10408, Aug. 2024.

[16] F. Keles, P. Wijewardena, and C. Hegde, "On the computational complexity of self-attention," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2022, pp. 597–619.

[17] M. Popel and O. Bojar, "Training tips for the transformer model," *Prague Bull. Math. Linguistics*, vol. 110, no. 1, pp. 43–70, Apr. 2018.

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

[19] B. Saha and K. Goebel, "Battery data set," NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, uSA, Tech. Rep., 2007. [Online]. Available: https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/

[20] Z. Ren and C. Du, "A review of machine learning state-of-charge and state-of-health estimation algorithms for lithium-ion batteries," *Energy Rep.*, vol. 9, pp. 2993–3021, Feb. 2023.

[21] J. Shen, W. Ma, X. Shu, S. Shen, Z. Chen, and Y. Liu, "Accurate state of health estimation for lithium-ion batteries under random charging scenarios," *Energy*, vol. 279, Jun. 2023, Art. no. 128092.

[22] J. Wu, J. Chen, F. Xiong, H. Xiang, and Q. Zhu, "State of health estimation of lithium-ion batteries using autoencoders and ensemble learning," *J. Energy Storage*, vol. 55, Sep. 2022, Art. no. 105708.

[23] Z. Zhu, Q. Yang, X. Liu, and D. Gao, "Attention-based CNN-BiLSTM for SOH and RUL estimation of lithium-ion batteries," *J. Algorithms Comput. Technol.*, vol. 16, Jan. 2022, Art. no. 17483026221130598.

[24] L. Ren, J. Dong, X. Wang, Z. Meng, L. Zhao, and M. J. Deen, "A data-driven Auto-CNN-LSTM prediction model for lithium-ion battery remaining useful life," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3478–3487, May 2021.

[25] Z. Tong, J. Miao, S. Tong, and Y. Lü, "Early prediction of remaining useful life for lithium-ion batteries based on a hybrid machine learning method," *J. Cleaner Prod.*, vol. 317, Jul. 2021, Art. no. 128265.

[26] Y. Tan and G. Zhao, "Transfer learning with long short-term memory network for state-of-health prediction of lithium-ion batteries," *IEEE Trans. Ind. Electron.*, vol. 67, no. 10, pp. 8723–8731, Oct. 2020.

[27] X. Li, C. Yuan, and Z. Wang, "State of health estimation for Li-ion battery via partial incremental capacity analysis based on support vector regression," *Energy*, vol. 203, May 2020, Art. no. 117852.

[28] N. Yang, Z. Song, H. Hofmann, and J. Sun, "Robust state of health estimation of lithium-ion batteries using convolutional neural network and random forest," *J. Energy Storage*, vol. 48, Oct. 2020, Art. no. 103857.

[29] A. Roy, "Hybrid random forest regression and artificial neural networks for modelling and monitoring the state of health of Li-ion battery," *J. Electr. Syst.*, vol. 20, no. 2, pp. 2231–2243, Apr. 2024.

[30] Y. Cheng, Y. Qin, W. Zhao, Q. Yang, N. Cai, and K. Wu, "A self-attention knowledge domain adaptation network for commercial lithium-ion batteries state-of-health estimation under shallow cycles," *J. Energy Storage*, vol. 86, Mar. 2024, Art. no. 111197.

[31] J. Lu, R. Xiong, J. Tian, C. Wang, and F. Sun, "Deep learning to estimate lithium-ion battery state of health without additional degradation experiments," *Nature Commun.*, vol. 14, no. 1, p. 2760, May 2023.

[32] K. Severson, P. M. Attia, N. Jin, N. Perkins, B. Jiang, Z. J. Yang, M. H. Chen, M. Aykol, P. Herring, D. Fraggedakis, M. Z. Bazant, S. J. Harris, W. C. Chueh, and R. D. Braatz, "Data-driven prediction of battery cycle life before capacity degradation," *Nature Energy*, vol. 4, no. 5, pp. 383–391, Mar. 2019.

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

[34] C. Huang, N. Li, J. Zhu, and S. H. Shi, "Battery health state prediction based on singular spectrum analysis and transformer network," *Electronics*, vol. 13, no. 13, p. 2434, Jun. 2024.

[35] K. Nakano and K. Tanaka, "Transformer-based online battery state of health estimation from electric vehicle driving data," in *Proc. 15th Int. Conf. Appl. Energy (ICAE)*, vol. 43. Doha, Qatar: Energy Proceedings, Dec. 2024.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.

[37] 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," 2021, *arXiv:2110.13985*.

[38] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Re, "HiPPO: Recurrent memory with optimal polynomial projections," 2020, *arXiv:2008.07669*.
[39] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," 2021, *arXiv:2111.00396*.
[40] J. T. H. Smith, A. Warrington, and S. W. Linderman, "Simplified state space layers for sequence modeling," 2022, *arXiv:2208.04933*.
[41] A. Gupta, A. Gu, and J. Berant, "Diagonal state spaces are as effective as structured state spaces," 2022, *arXiv:2203.14343*.
[42] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Ré, "Hungry hungry hippos: Towards language modeling with state space models," 2022, *arXiv:2212.14052*.
[43] A. Gu, A. Gupta, K. Goel, and C. Ré, "On the parameterization and initialization of diagonal state space models," 2022, *arXiv:2206.11893*.
[44] A. Gu, I. Johnson, A. Timalsina, A. Rudra, and C. Ré, "How to train your HiPPO: State space models with generalized orthogonal basis projections," 2022, *arXiv:2206.12037*.
[45] J. Lin and H. Hu, "Audio mamba: Pretrained audio state space model for audio tagging," 2024, *arXiv:2405.13636*.
[46] M. Hamza Erol, A. Senocak, J. Feng, and J. Son Chung, "Audio mamba: Bidirectional state space model for audio representation learning," 2024, *arXiv:2406.03344*.
[47] E. Nguyen, K. Goel, A. Gu, G. W. Downs, P. Shah, T. Dao, S. A. Baccus, and C. Ré, "S4ND: Modeling images and videos as multidimensional signals using state spaces," 2022, *arXiv:2210.06583*.
[48] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, and Y. Liu, "VMamba: Visual state space model," 2024, *arXiv:2401.10166*.
[49] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," 2024, *arXiv:2401.09417*.
[50] G. Chen, Y. Huang, J. Xu, B. Pei, Z. Chen, Z. Li, J. Wang, K. Li, T. Lu, and L. Wang, "Video mamba suite: State space model as a versatile alternative for video understanding," 2024, *arXiv:2403.09626*.
[51] K. Li, X. Li, Y. Wang, Y. He, Y. Wang, L. Wang, and Y. Qiao, "VideoMamba: State space model for efficient video understanding," 2024, *arXiv:2403.06977*.
[52] O. Lieber, B. Lenz, H. Bata, G. Cohen, J. Osin, I. Dalmedigos, E. Safahi, S. Meirom, Y. Belinkov, S. Shalev-Shwartz, and O. Abend, "Jamba: A hybrid transformer-mamba language model," 2024, *arXiv:2403.19887*.
[53] Z. Wan, P. Zhang, Y. Wang, S. Yong, S. Stepputtis, K. Sycara, and Y. Xie, "Sigma: Siamese mamba network for multi-modal semantic segmentation," 2024, *arXiv:2404.04256*.
[54] Z. Zhang, A. Liu, I. Reid, R. Hartley, B. Zhuang, and H. Tang, "Motion mamba: Efficient and long sequence motion generation," 2024, *arXiv:2403.07487*.
[55] Z. Shi, "MambaStock: Selective state space model for stock prediction," 2024, *arXiv:2402.18959*.
[56] A. Ali, I. Zimerman, and L. Wolf, "The hidden attention of mamba models," 2024, *arXiv:2403.01590*.
[57] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality," 2024, *arXiv:2405.21060*.
[58] A. Behrouz, M. Santacatterina, and R. Zabih, "MambaMixer: Efficient selective state space models with dual token and channel selection," 2024, *arXiv:2403.19888*.
[59] Y. Liu, C. Liu, Y. Liu, F. Sun, J. Qiao, and T. Xu, "Review on degradation mechanism and health state estimation methods of lithium-ion batteries," *J. Traffic Transp. Eng.*, vol. 10, no. 4, pp. 578–610, Aug. 2023.
[60] J. Zeng and S. Liu, "Research on aging mechanism and state of health prediction in lithium batteries," *J. Energy Storage*, vol. 72, Nov. 2023, Art. no. 108274.
[61] Y. Li, K. Liu, A. Foley, A. Zülke, M. Berecibar, E. Nanini-Maury, J. V. Mierlo, and H. E. Hoster, "Data-driven health estimation and lifetime prediction of lithium-ion batteries: A review," *Renew. Sustain. Energy Rev.*, vol. 113, Jul. 2019, Art. no. 109254.
[62] X. Yang, Y. Leng, G. Zhang, S. Ge, and C. Wang, "Modeling of lithium plating induced aging of lithium-ion batteries: Transition from linear to nonlinear aging," *J. Power Sources*, vol. 360, pp. 28–40, Jun. 2017.
[63] P. M. Kekenes-Huskey, C. E. Scott, and S. Atalay, "Quantifying the influence of the crowded cytoplasm on small molecule diffusion," *J. Phys. Chem. B*, vol. 120, no. 33, pp. 8696–8706, Jun. 2016.
[64] L. Wang, Y. Jia, and J. Xu, "Mechanistic understanding of the electrochemo-dependent mechanical behaviors of battery anodes," *J. Power Sources*, vol. 510, Aug. 2021, Art. no. 230428.
[65] M. Yamada, T. Watanabe, T. Gunji, J. Wu, and F. Matsumoto, "Review of the design of current collectors for improving the battery performance in lithium-ion and post-lithium-ion batteries," *Electrochem*, vol. 1, no. 2, pp. 124–159, May 2020.
[66] Q. Wang, P. Ping, X. Zhao, G. Chu, J. Sun, and C. Chen, "Thermal runaway caused fire and explosion of lithium ion battery," *J. Power Sources*, vol. 208, pp. 210–224, Mar. 2012.
[67] N. Yang, X. Zhang, B. Shang, and G. Li, "Unbalanced discharging and aging due to temperature differences among the cells in a lithium-ion battery pack with parallel combination," *J. Power Sources*, vol. 306, pp. 733–741, Dec. 2015.
[68] H. Tian, P. Qin, K. Li, and Z. Zhao, "A review of the state of health for lithium-ion batteries: Research status and suggestions," *J. Cleaner Prod.*, vol. 261, Mar. 2020, Art. no. 120813.
[69] J. Vetter, P. Novák, M. R. Wagner, C. Veit, K. Möller, J. Besenhard, M. Winter, M. Wohlfahrt-Mehrens, C. Vogler, and A. Hammouche, "Ageing mechanisms in lithium-ion batteries," *J. Power Sources*, vol. 147, nos. 1–2, pp. 269–281, Mar. 2005.
[70] T. Waldmann, M. Wilka, M. Kasper, M. Fleischhammer, and M. Wohlfahrt-Mehrens, "Temperature dependent ageing mechanisms in lithium-ion batteries—A post-mortem study," *J. Power Sources*, vol. 262, pp. 129–135, Mar. 2014.
[71] D. P. Finegan, M. Scheel, J. B. Robinson, B. Tjaden, I. Hunt, T. J. Mason, J. Millichamp, M. D. Michiel, G. J. Offer, G. Hinds, D. J. L. Brett, and P. R. Shearing, "In-operando high-speed tomography of lithium-ion batteries during thermal runaway," *Nature Commun.*, vol. 6, no. 1, p. 6924, Apr. 2015.
[72] Z. Wang and C. Du, "A comprehensive review on thermal management systems for power lithium-ion batteries," *Renew. Sustain. Energy Rev.*, vol. 139, Jan. 2021, Art. no. 110685.
[73] Y. Gao, J. Jiang, C. Zhang, W. Zhang, Z. Ma, and Y. Jiang, "Lithium-ion battery aging mechanisms and life model under different charging stresses," *J. Power Sources*, vol. 356, pp. 103–114, Apr. 2017.
[74] J. Jaguemont, L. Boulon, and Y. Dubé, "A comprehensive review of lithium-ion batteries used in hybrid and electric vehicles at cold temperatures," *Appl. Energy*, vol. 164, pp. 99–114, Dec. 2015.
[75] Y. B. He, F. Ning, Q. H. Yang, Q. S. Song, B. Li, F. Su, H. Du, Z. Y. Tang, and F. Kang, "Structural and thermal stabilities of layered Li$(Ni_{1/3}Co_{1/3}Mn_{1/3})O_2$ materials in 18650 high power batteries," *J. Power Sources*, vol. 196, pp. 10322–10327, Dec. 2011.
[76] M. Ouyang, D. Ren, L. Lu, J. Li, X. Feng, X. Han, and G. Liu, "Overcharge-induced capacity fading analysis for large format lithium-ion batteries with $Li_yNi_{1/3}Co_{1/3}Mn_1/_3O_2$+$LiyMn_2O_4$ composite cathode," *J. Power Sources*, vol. 279, pp. 626–635, Apr. 2015.
[77] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4700–4708.
[78] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.
[79] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," 2016, *arXiv:1605.07648*.
[80] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, "Mixed precision training," 2018, *arXiv:1710.03740*.

JOSÉ IGNACIO OLALDE-VERANO received the master's degree in research in industrial technologies from Universidad Nacional de Educación a Distancia (UNED), Spain, the degree in technical engineering from the University of Zaragoza, and the degree from the University of León. He is currently pursuing the Ph.D. degree with UNED. Since 2009, he has been in the automotive industry. His research interests include machine learning techniques applied to industry 4.0.

SASCHA KIRCH (Graduate Student Member, IEEE) received the B.Eng. degree in electrical engineering from the Cooperative State University Baden-Wuerttemberg (DHBW), Germany, and the M.Sc. degree in electronic systems for communication and information from Universidad Nacional de Educación a Distancia (UNED), Spain, where he is currently pursuing the Ph.D. degree. He works as an Expert in deep learning for automotive sensor perception and has more than 11 years of experience in software and hardware projects in various industries, in which he has taken multiple roles, including an Engineer, a Project Lead, and a Technical Expert. His research interests include self-supervised multi-modal generative deep learning. He is a member of IEEE's honor society Eta Kappa Nu and a President of the chapter Nu Alpha.

CLARA PÉREZ-MOLINA (Senior Member, IEEE) received the M.Sc. degree in physics from the Complutense University of Madrid and the Ph.D. degree in industrial engineering from Universidad Nacional de Educación a Distancia (UNED), Spain. She is currently an Associate Professor with tenure of the Electrical and Computer Engineering Department, UNED. She has worked as a Researcher in several national and European projects and has published different technical reports and research articles for international journals and conferences, and several teaching books. Her research activities are centered on educational competences and technology enhanced learning applied to higher education in addition to renewable energy management and artificial intelligence techniques.

SERGIO MARTÍN (Senior Member, IEEE) is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department, Industrial Engineering School, Universidad Nacional de Educación a Distancia (UNED), Spain. He is an Associate Professor with UNED. He is a Computer Engineer in distributed applications and systems by the Carlos III University of Madrid. He teaches subjects related to microelectronics and digital electronics in the Industrial Engineering School, UNED, since 2007. He has participated in national and international research projects related to mobile devices, ambient intelligence, and location-based technologies and in projects related to ''e-learning'', virtual and remote labs, and new technologies applied to distance education, since 2002. He has published more than 200 papers both in international journals and conferences.