FlowBatt: Flow Matching for Probabilistic Battery Degradation Prediction

Hamidreza Eivazi[1,*] **Eslam Sharaawy**[2] **Stefan Wittek**[2]
André Hebenbrock[3] **Raphael Ginster**[4] **Steffen Blömeke**[5]

[1] Division Data-Driven Modeling of Mechanical Systems, Institute of Applied Mechanics, Technische Universität Braunschweig, Germany

[2] Institute for Software and Systems Engineering, Clausthal University of Technology, Germany

[3] Institute of Chemical and Electrochemical Process Engineering, Clausthal University of Technology, Germany

[4] Institute of Automotive Management and Industrial Production, Technische Universität Braunschweig, Germany

[5] Institute of Machine Tools and Production Technology, Technische Universität Braunschweig, Germany

Abstract

Battery degradation remains a central challenge in the development and broad application of sustainable energy technologies. Accurate degradation prediction is challenging, as battery aging emerges from complex and heterogeneous interactions among cycling behavior, operating conditions, and cell chemistry. Existing machine-learning approaches typically focus on deterministic end-of-life predictions or modeling degradation curves within fixed chemistry settings, leaving probabilistic state of health (SOH) trajectory modeling across heterogeneous chemistries underexplored. We introduce FlowBatt, a use-inspired conditional generative framework that adapts flow matching with a diffusion transformer (DiT) backbone to battery degradation trajectory modeling. FlowBatt models full SOH trajectories as probabilistic degradation processes conditioned on early-cycle capacity data. To strengthen this conditioning pathway, we use explainable AI analysis to diagnose and improve the encoder that maps early-cycle capacity matrices into conditioning vectors. We evaluate FlowBatt on five public benchmark settings spanning diverse chemistries and aging conditions, comparing it with supervised and diffusion-based trajectory models as well as established baselines for remaining useful life (RUL) prediction under shared data splits. FlowBatt achieves the best trajectory-wise SOH prediction performance on all five datasets, with errors below 2.5%, and delivers competitive RUL prediction performance. By generating multiple plausible trajectories, FlowBatt provides empirical prediction intervals for battery degradation modeling. These results indicate that flow-matching-based trajectory modeling is a promising framework for probabilistic battery health prediction.

1 Introduction

Lithium-ion batteries and degradation. Lithium-ion (Li-ion) batteries are central to modern energy-storage applications, including portable electronics, electric vehicles, and stationary storage systems [6]. Their widespread use is driven by favorable properties such as high energy density, long cycle life, and low self-discharge [14, 45]. Despite these advantages, battery degradation remains a

*Corresponding author: h.eivazi@tu-braunschweig.de (https://orcid.org/0000-0003-3650-4107)

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

major limitation for reliability, safety, and economic viability [5, 8, 15]. A key manifestation of this degradation is capacity fade, which arises from complex electrochemical and mechanical mechanisms influenced by operating conditions, usage history, and cell chemistry [7, 12, 36]. This complexity makes accurate degradation forecasting challenging.

Machine learning for battery health prediction. Battery degradation is commonly characterized through state of health (SOH), which measures the remaining capacity relative to the nominal capacity, and remaining useful life (RUL), which estimates the cycles or time remaining before end of life (EOL) [9, 25, 28, 41]. Since both quantities depend on complex and nonlinear degradation dynamics, machine learning has become an increasingly important tool for battery health prediction [40]. Prior work has explored a wide range of methods, including linear models, support vector machines, Gaussian processes, convolutional and recurrent neural networks, and transformers [21, 25, 28, 40, 41, 47]. Most of these approaches are formulated as supervised regression models and focus on endpoint prediction, such as a scalar SOH value at a specific cycle or the RUL of a cell. Standardized benchmarks such as BatteryML [51] and BatLiNet [52] have improved comparability across datasets and models. However, existing benchmarks also highlight an important limitation of the current literature: most methods are optimized for deterministic point prediction, especially RUL, while probabilistic modeling of full SOH degradation trajectories remains much less explored. A notable recent exception is the diffusion-based work of Luo et al. [32], which points to the potential of generative modeling for battery SOH prediction.

Contribution to literature. This work advances the domain of battery degradation prediction in several aspects. First, we introduce *FlowBatt*, a use-inspired conditional generative framework that adapts flow matching to probabilistic battery degradation modeling. Rather than proposing a new flow-matching objective, our contribution is to formulate battery health prediction as conditional full-trajectory SOH generation from early-cycle capacity data. Leveraging a diffusion transformer (DiT) backbone, FlowBatt captures the stochastic and nonlinear dynamics of battery aging as a probabilistic, trajectory-level process more effectively than traditional supervised approaches (section 2). Second, we use explainable AI to analyze the encoder that maps early-cycle capacity matrices to conditioning vectors, identify failure modes in a ReLU-based encoder, and adopt a more robust LeakyReLU encoder for proposed models (appendix A.3). Third, we demonstrate the benefits of flow-based generative modeling by benchmarking FlowBatt against diffusion processes and supervised learning methods. Our results show that FlowBatt achieves the best performance in SOH prediction across diverse aging conditions and chemistry settings, while also delivering competitive performance compared to the state-of-the-art in RUL prediction (section 3). Finally, we establish FlowBatt as a probabilistic model for battery degradation, capable of unifying SOH trajectory forecasting, uncertainty quantification, and degradation-curve synthesis within a single framework, enabling future extensions and adaptations across diverse energy storage applications (section 4). The source code and pre-trained models are publicly available at `https://github.com/HamidrezaEiv/FlowBatt`.

2 Methodology

Denoising diffusion probabilistic models (DDPMs) [19] are a flexible family of generative models that learn to approximate a data distribution through a gradual noising-denoising process. The forward process is defined as a parameterized Markov chain that progressively corrupts the data with Gaussian noise until the signal is indistinguishable from random noise. The reverse process, also known as the denoising or generative process, is learned by the model and gradually removes noise step by step to reconstruct samples resembling the training data. Training is performed using a variational objective that optimizes a lower bound on the data likelihood across many diffusion steps. Once trained, DDPMs can generate diverse, high-quality samples, which has led to widespread applications in image and audio synthesis [11, 20, 49], data augmentation [33], and scientific modeling [4, 13, 26, 50].

Flow matching [1, 29, 31] is an alternative to diffusion training that directly learns continuous-time generative dynamics. Instead of discretizing the forward and reverse processes into thousands of noisy steps as in DDPMs, flow matching estimates an ordinary differential equation (ODE) that smoothly transports a base distribution (e.g., Gaussian noise) into the target data distribution. This approach avoids the stochasticity and step-by-step denoising of DDPMs and enables more efficient training with fewer discretization steps. Moreover, flow matching naturally admits exact likelihood estimation, making it both expressive and computationally attractive for high-dimensional scientific data. In this work, we adopt flow matching to train our generative model of battery degradation.

Transformers have become a powerful backbone for diffusion-based generative models, giving rise to diffusion transformers (DiTs) [37]. Unlike convolutional UNet backbones, DiTs leverage global self-attention to capture long-range dependencies in the data. Diffusion transformers typically operate on tokenized patches of the input, and use an effective conditioning mechanism based on adaptive layer normalization [38] for conditioning on auxiliary information such as class labels or text encodings. In this work, the auxiliary information corresponds to the battery's performance during the early cycle life. Our approach employs DiTs directly on raw data representations of battery degradation. This design combines the scalability and expressivity of transformers with the probabilistic training of flow matching to capture the nonlinear, stochastic dynamics of battery aging.

2.1 Diffusion and flow-matching foundations

This subsection introduces the diffusion and flow-matching objectives used by DiffBatt and FlowBatt, respectively. We represent a cycle-indexed SOH trajectory at generative time t as $\boldsymbol{x}_t = (x_t^{(1)}, \ldots, x_t^{(T)}) \in \mathbb{R}^T$, where T is the number of cycles, t denotes generative time, and i denotes the sampled battery-cycle index. In diffusion, $\boldsymbol{x}_t$ is the noisy trajectory, whereas in flow matching it is the time-t interpolant.

Denoising diffusion models (DDMs) learn to transform a simple prior distribution, typically a unit Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, into an unknown data distribution $q(\boldsymbol{x})$ by gradually denoising samples corrupted with Gaussian noise [19]. A fixed *forward process* incrementally adds noise to data $\boldsymbol{x}_0 \sim q(\boldsymbol{x})$ according to a variance schedule, and a neural network is trained to approximate the *reverse process* that removes noise step by step. With the standard parameterization suggested in [19], the training objective reduces to a denoising score matching loss, where the network $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ predicts the injected Gaussian noise

$$\mathcal{L}_{\text{simple}}(\theta) := \mathbb{E}_{t, \boldsymbol{x}_0, \boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\|^2 \right]. \tag{1}$$

Here, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ denotes the sampled Gaussian noise used to construct $\boldsymbol{x}_t$ in the forward process. For conditional modeling, auxiliary information $\boldsymbol{c}$ can be incorporated into the reverse process by conditioning the network $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$, often enhanced with classifier-free guidance [18]. While diffusion models achieve strong generative performance, sampling requires simulating many reverse steps, which motivates alternative formulations.

Instead of constructing a discrete forward diffusion process, flow matching [29, 31] formulates generative modeling as learning a continuous-time transport map between a tractable prior distribution $p(\boldsymbol{x})$, typically $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and the unknown data distribution $q(\boldsymbol{x})$. The approach defines a family of interpolant distributions $\{p_t(\boldsymbol{x})\}_{t \in [0,1]}$ that smoothly connects the prior $p_0(\boldsymbol{x}) = p(\boldsymbol{x})$ to the target $p_1(\boldsymbol{x}) = q(\boldsymbol{x})$. This interpolation is governed by an ordinary differential equation (ODE) of the form

$$\frac{d\boldsymbol{x}_t}{dt} = \boldsymbol{v}_\theta(\boldsymbol{x}_t, t), \quad \boldsymbol{x}_0 \sim p(\boldsymbol{x}), \tag{2}$$

where $\boldsymbol{v}_\theta$ is a neural network parameterized vector field trained to approximate the unknown probability flow that transports samples from prior to data. To obtain a tractable training objective, one introduces a *target vector field* defined by the time derivative of a prescribed stochastic interpolant $\boldsymbol{x}_t = \alpha(t)\boldsymbol{x}_0 + \beta(t)\boldsymbol{x}_1$ with $(\boldsymbol{x}_0, \boldsymbol{x}_1) \sim p(\boldsymbol{x}) \times q(\boldsymbol{x})$. For example, choosing linear interpolation $\alpha(t) = 1 - t$, $\beta(t) = t$ yields

$$\tilde{\boldsymbol{v}}(\boldsymbol{x}_t, t | \boldsymbol{x}_0, \boldsymbol{x}_1) = \frac{d}{dt} \left[\alpha(t)\boldsymbol{x}_0 + \beta(t)\boldsymbol{x}_1 \right] = \dot{\alpha}(t)\boldsymbol{x}_0 + \dot{\beta}(t)\boldsymbol{x}_1. \tag{3}$$

The model $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ is then trained to match this target vector field in expectation:

$$\mathcal{L}_{\text{FM}}(\theta) := \mathbb{E}_{t, \boldsymbol{x}_0 \sim p(\boldsymbol{x}), \boldsymbol{x}_1 \sim q(\boldsymbol{x})} \left[\|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \tilde{\boldsymbol{v}}(\boldsymbol{x}_t, t | \boldsymbol{x}_0, \boldsymbol{x}_1)\|^2 \right]. \tag{4}$$

This training can be interpreted as regressing the dynamics of sample trajectories along linear interpolation paths from the prior to the data distribution. Once trained, the model generates samples by integrating the learned ODE

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \int_0^1 \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)\, dt, \quad \boldsymbol{x}_0 \sim p(\boldsymbol{x}), \tag{5}$$

which requires significantly fewer steps than simulating the reverse diffusion process. Moreover, flow matching provides a unifying framework: with appropriate choices of interpolants, it can recover score-based diffusion models as a special case while also supporting more efficient sampling schemes. For conditional generation, the conditioning variable $\boldsymbol{c}$ is incorporated directly into the learned vector field $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$.

2.2 Architecture

Figure 1 provides an overview of the FlowBatt architecture and its workflow. The top panel depicts the diffusion-based training stage, where noisy SOH trajectories are progressively denoised through a stack of DiT blocks conditioned on early-cycle capacity matrices. The bottom panels illustrate the auxiliary pathway and the generation stage, where the capacity matrix is first encoded by a CNN into a compact embedding, which is combined with the timestep embedding to modulate the DiT backbone and generate complete SOH trajectories.

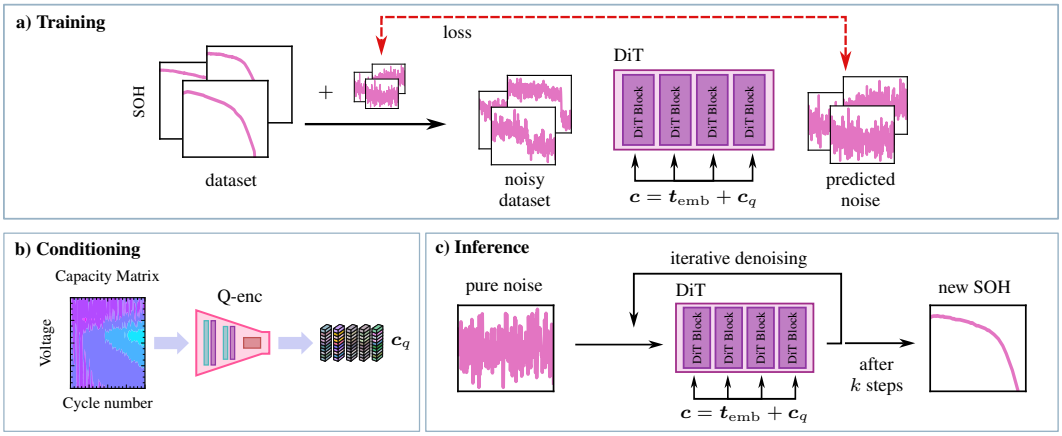


Figure 1: Schematic view of the model architecture. Adapted and modified from [13]. Modifications include context-specific changes.

DiT backbone and block design. FlowBatt instantiates a compact Diffusion Transformer (DiT) for long one-dimensional sequences representing SOH during the battery cycle life. Given the noisy, cycle-indexed SOH trajectory $\boldsymbol{x}_t \in \mathbb{R}^T$, we first lift it to a token sequence via a 1×1 convolution giving $\boldsymbol{h}_0 \in \mathbb{R}^{T\times d}$, where d is the hidden width (number of channels). After preprocessing, $T = 256$, with one SOH value sampled every ten physical cycles. We add fixed one-dimensional sinusoidal positional embeddings $\boldsymbol{p} \in \mathbb{R}^{T\times d}$ to preserve temporal order. The core of the network is a stack of N identical DiT blocks, each following the Transformer paradigm of a pre-normalized residual connection with a multi-head self-attention (MSA) sublayer, followed by a feed-forward multilayer perceptron (MLP) sublayer. The stack is followed by a final projection layer that applies an adaptive normalization conditioned on the conditioning vector and then maps the hidden states back to the data domain. This ensures that conditioning information influences not only the internal block dynamics but also the final predicted vector field used in flow matching.

Conditioning mechanism. In the adaptive layer norm (adaLN)-Zero formulation [37], each MSA and MLP sublayer is modulated by per-channel scale-offset and shift parameters (γ, β) applied to the normalized activations, together with residual scaling factors α applied directly before the residual addition. All modulation parameters are produced from a single global conditioning vector $\boldsymbol{t}_{\text{emb}} + \boldsymbol{c_q}$ that merges a timestep embedding $\boldsymbol{t}_{\text{emb}} = \phi(t)$ and an auxiliary embedding $\boldsymbol{c_q}$ obtained by encoding the capacity matrix corresponding to the early cycle life of the battery. The conditioning vector is passed through a small multilayer perceptron that produces six d-dimensional vectors as per-channel modulation parameters. The residual scalars α are initialized to zero, ensuring that every block starts as an identity mapping and that conditioning effects are introduced gradually during training. This approach has been shown to improve optimization stability and sample quality in large-scale generative modeling [29]. The detailed computational steps of one DiT block with adaLN-Zero conditioning are provided in appendix A.2.4.

CNN encoder. We employ the concept of *capacity matrix* ($\boldsymbol{Q}$) introduced by Attia et al. [3], as auxiliary information for conditional generation. The capacity matrix is a cycle-by-voltage array of baseline-subtracted discharge-capacity curves. Its construction and rationale are detailed in appendix A.2.1. We utilize the capacity matrix obtained from the early cycle life of the battery, i.e., the first 20 or 100 cycles similar to [3, 43, 51]. This choice is driven by the high costs, time, and effort associated with long-term battery testing. Our goal is to leverage early life performance data to predict battery degradation and minimize resource expenditure. To encode $\boldsymbol{Q}$ into an embedding ($\boldsymbol{c}_q$), we

utilize a CNN encoder (see figure 1b). We adapt a CNN architecture inspired by Attia et al. [3], Zhang et al. [52], which has been used for RUL prediction. This model has been reported to lack robustness with respect to different initializations, however, it provides very accurate predictions in some instances. To enhance the robustness of the model we utilize explainable AI methods, specifically Grad-CAM (gradient-weighted class activation mapping) method [42] to better understand how these deep-learning models make predictions for the RUL task. Further, we identify and eliminate potential modes of failure. Results for this analysis are reported in appendix A.3. This explainable AI analysis reveals initialization-sensitive failure modes in a ReLU-based CNN encoder, including cases with inactive gradients. We therefore replace ReLU activations with leaky rectified linear unit (LeakyReLU) activations to obtain a more robust XAI-guided CNN encoder. The final architecture consists of two convolutional layers with LeakyReLU activations, each followed by average pooling to reduce spatial dimensions. The resulting feature map is projected with a final convolution, activated, and flattened into a d-dimensional vector $\boldsymbol{c}_q$. This CNN encoder effectively captures the relevant features from the capacity matrix, enabling FlowBatt to generate accurate and context-aware SOH trajectories.

In summary, our generative models trained via flow matching and diffusion processes combine a DiT backbone with adaLN-Zero conditioning and a CNN-based encoder for auxiliary early-life data, yielding a scalable generative framework that provides empirical prediction intervals. By integrating timestep information with early-cycle capacity matrices, the model learns to generate realistic SOH trajectories that reflect both temporal dynamics and usage context. From these SOH trajectories, metrics such as RUL can be directly computed by applying practical thresholds (e.g., 80% of nominal capacity). This design positions FlowBatt as a flexible model for battery health prediction, capable of both probabilistic modeling of aging behavior and data-driven synthesis. Figure 2 illustrates the denoising trajectories of a test sample from the `MATR-1` dataset using flow matching and diffusion processes. It can be observed that the flow matching approach achieves faster denoising compared to the diffusion process. Note that diffusion requires 1000 reverse steps, while in flow matching we solve the ODE over [0,1].

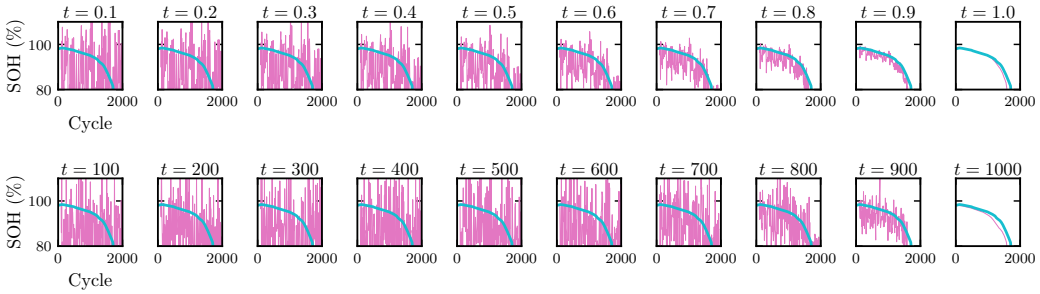


Figure 2: Comparison of denoising trajectories for a test sample from the `MATR-1` dataset using flow matching (top) and the diffusion process (bottom). The pink line shows the generated sample and the cyan line shows the reference for a given capacity matrix. The displayed FlowBatt states correspond to requested ODE output times for visualization.

2.3 Data

We conduct a comprehensive evaluation based on several publicly available datasets curated in BatteryML [51]. Specifically, we include `CALCE` [16, 48], `HUST` [34], `MATR` [2, 43], `RWTH` [27], `SNL` [39], and `UL_PUR` [22, 23], covering a range of chemistries including lithium iron phosphate (LFP), lithium cobalt oxide (LCO), nickel manganese cobalt oxide (NMC), nickel cobalt aluminum oxide (NCA), and a combination of NMC and LCO (NMC_LCO). These datasets span diverse aging conditions, including differences in capacities, voltages, ambient temperatures, state of charge (SOC), and degradation characteristics, and therefore provide a broad spectrum of battery aging behaviors.

In this study, we follow the benchmark protocol of BatLiNet [52], which derives five evaluation datasets under different aging conditions. Specifically, BatLiNet defines two `MATR` evaluation sets (`MATR-1` and `MATR-2`) to maintain comparability with existing models. In addition, the `HUST` dataset, which uses LFP cells similar to `MATR`, is included to evaluate model adaptation to different cycling protocols. Furthermore, BatLiNet constructs a `MIX` dataset by aggregating all remaining public

datasets, capturing diverse aging conditions such as different ambient temperatures, packaging structures, and cathode chemistries (NMC, LCO, and NCA). From the `MIX` dataset, two benchmark settings are defined: `MIX-100`, which evaluates the prediction of the 80% EOL point using only the first 100 cycles, and `MIX-20`, a more challenging setup requiring the prediction of the number of cycles until 90% capacity using only the first 20 cycles.

While these benchmark definitions were originally introduced for RUL prediction, in this work we use the same datasets and data splits for both SOH trajectory prediction and RUL evaluation. This allows us to assess full degradation-trajectory forecasting under the same standardized experimental conditions and to retain direct comparability with prior benchmark studies. As in [52], batteries that reached EOL prematurely during the early cycles are excluded from the analysis. Further details on datasets and preprocessing are provided in appendix A.1 and [51, 52].

3 Results and discussion

Our generative models can be utilized for trajectory-wise SOH prediction, RUL prediction, and SOH synthesis. Here, SOH synthesis refers to the generation of new SOH curves for data augmentation. Among these tasks, trajectory-wise SOH prediction is the primary focus of this work, since FlowBatt is designed to model full degradation processes rather than only scalar quantities. Cycle-wise SOH prediction estimates SOH at a predefined cycle, whereas RUL prediction estimates the first cycle at which SOH reaches or falls below the EOL threshold. Our trajectory models provide both scalar predictions without task-specific training. SOH and RUL prediction are closely related tasks in battery health management: SOH describes the temporal evolution of battery health, whereas RUL can be derived as a downstream EOL indicator from the predicted SOH trajectory.

In this section, we conduct experiments based on the BatLiNet benchmark tests [52] and compare the results across models. We first evaluate trajectory-wise SOH prediction to assess the ability of the models to forecast full degradation trajectories, which is the most relevant setting for FlowBatt. For completeness and comparability with BatteryML-style evaluation, we also evaluate cycle-wise SOH prediction. We then examine RUL prediction derived from the predicted SOH trajectories under the same benchmark setting. We report results for three models trained using supervised learning, diffusion processes, and flow matching, referred to as Transformer, DiffBatt, and FlowBatt, respectively.

In industrial applications, trajectory-wise SOH prediction supports continuous health monitoring, degradation-aware operation, warranty assessment, and long-term maintenance planning. Cycle-wise SOH prediction supports health assessment at a particular inspection or operating cycle, while RUL provides an actionable endpoint for maintenance scheduling and battery replacement. Evaluation metrics, EOL thresholds, and additional results are provided in appendix A.5.

3.1 SOH prediction

In practical applications, estimating a battery's SOH requires assessing the current discharge capacity under standardized conditions using reference performance tests and historical cycling data. However, the discrepancy between real-world battery usage and these standardized conditions poses significant challenges in obtaining precise ground-truth labels, which complicates accurate SOH prediction. Consequently, establishing robust and reproducible benchmark settings for SOH prediction remains an ongoing effort within the research community. In this work, we primarily evaluate SOH prediction in a *trajectory-wise* setting. In this setting, the models predict the full SOH trajectory, and the RMSE is computed over the degradation curve up to the EOL point, padding the remainder of the sequence with the final observed value. Results are then reported as the mean RMSE across all test samples. Since SOH is expressed as a percentage of the nominal capacity, the RMSE is also reported in percentage units. Further details on the experimental setup and error metrics are provided in appendix A.5.

Most established RUL baselines predict a scalar and require adaptation to produce full SOH trajectories. We include MLP as a standard neural baseline and RF for its best reported RUL performance on BatteryML's largest dataset, MIX [51], together with BatLiNet [52], a state-of-the-art RUL predictor. These three baselines are adapted for trajectory-wise SOH prediction. For RUL, we retain the broader set of direct scalar predictors.

The trajectory-wise SOH results are reported in table 1. This setting directly evaluates the models on their ability to forecast full degradation trajectories and is therefore the most representative evaluation

Table 1: Trajectory-wise SOH prediction results reported as mean RMSE (%) over the predicted degradation trajectories. The ± values denote standard deviations of RMSE across ten generative draws from one trained checkpoint for FlowBatt and DiffBatt, and across ten training initializations for the other models. Best results are shown in bold and second-best results are underlined. This table evaluates full-trajectory SOH prediction. RMSE is computed over the predicted degradation trajectories up to EOL and therefore measures a different task from the cycle-wise SOH setting reported in appendix A.5.3.

Method	MATR-1	MATR-2	HUST	MIX-20	MIX-100
BatLiNet	1.86 ± 0.06	3.62 ± 0.24	2.87 ± 0.23	2.04 ± 0.06	2.34 ± 0.22
RF (BatteryML)	2.01 ± 0.04	2.50 ± 0.01	3.57 ± 0.03	3.42 ± 0.02	5.45 ± 0.04
MLP (BatteryML)	1.97 ± 0.11	3.06 ± 0.17	4.11 ± 0.13	2.23 ± 0.09	3.13 ± 0.14
Transformer (ours)	1.09 ± 0.07	2.25 ± 0.22	3.18 ± 0.16	1.42 ± 0.10	1.76 ± 0.06
DiffBatt (ours)	<u>0.92 ± 0.04</u>	<u>1.85 ± 0.03</u>	<u>2.44 ± 0.14</u>	<u>1.15 ± 0.08</u>	<u>1.71 ± 0.09</u>
FlowBatt (ours)	**0.87 ± 0.02**	**1.77 ± 0.04**	**2.10 ± 0.12**	**0.88 ± 0.03**	**1.41 ± 0.02**

for FlowBatt. The supervised Transformer baseline provides a useful point of comparison, achieving reasonable performance on some datasets such as MATR-1 (1.09±0.07) and MIX-20 (1.42±0.10), but showing noticeably higher errors on more challenging datasets such as HUST (3.18±0.16). DiffBatt improves upon the Transformer on all five datasets and achieves the second-best results across the benchmark, indicating that diffusion-based generative modeling already provides a strong improvement over purely supervised trajectory prediction. FlowBatt achieves the lowest mean trajectory-wise SOH RMSE on all five datasets. In particular, it reaches an RMSE of 0.87±0.02 on MATR-1, 1.77±0.04 on MATR-2, 2.10±0.12 on HUST, and 0.88±0.03 on the heterogeneous MIX-20 dataset. These results indicate that flow matching provides a reliable and scalable approach for SOH trajectory forecasting across diverse battery chemistries and operating conditions. FlowBatt also provides a unified generative framework for full-trajectory prediction, uncertainty quantification, and synthesis. Overall, the trajectory-wise results confirm that FlowBatt is particularly strong when the task requires modeling the temporal evolution of degradation rather than only isolated SOH targets.

3.2 Uncertainty quantification

As FlowBatt is a probabilistic generative model, it can generate multiple plausible SOH trajectories for the same input capacity matrix. We therefore evaluate its uncertainty estimates by constructing empirical 90% prediction intervals from independently generated trajectories and measuring the fraction of reference SOH values contained within these intervals. The average interval width is also reported to assess the sharpness of the uncertainty bands. The interval definitions and comparisons with DiffBatt, Quantile RF, and Quantile MLP are provided in appendix A.5.4. Table 2 shows

Table 2: Empirical SOH prediction-interval coverage of FlowBatt for a nominal 90% interval. Coverage is the fraction of reference SOH values contained within the generated prediction interval. SOH width denotes the average interval width in percentage points. Best coverage values are shown in bold.

Samples	Metric	MATR-1	MATR-2	HUST	MIX-20	MIX-100
10	SOH coverage	86.0%	67.0%	65.0%	83.6%	81.7%
	SOH width	0.30%	0.42%	1.34%	0.52%	0.64%
50	SOH coverage	87.4%	70.0%	71.0%	86.4%	85.4%
	SOH width	0.36%	0.50%	1.62%	0.63%	0.77%
100	SOH coverage	88.0%	70.0%	**73.0%**	87.0%	86.0%
	SOH width	0.37%	0.51%	1.65%	0.65%	0.80%
500	SOH coverage	**88.4%**	**70.2%**	72.8%	**87.4%**	**86.3%**
	SOH width	0.38%	0.52%	1.65%	0.67%	0.83%

that FlowBatt provides informative SOH prediction intervals, and figure 3 illustrates representative examples across the evaluated datasets. With 100 generated samples, the empirical coverage reaches 88.0% on MATR-1, 87.0% on MIX-20, and 86.0% on MIX-100, while keeping the average interval width below one percentage point on these datasets. Increasing the number of generated samples generally stabilizes the estimated intervals, with the highest coverage obtained at 500 samples for most datasets. Coverage remains below the nominal 90% level, particularly on MATR-2 and HUST.

For reference, using a wider 95% interval with 500 samples increases the coverage to 90.5%, 72.5%, 76.1%, 89.6%, and 88.4% on MATR-1, MATR-2, HUST, MIX-20, and MIX-100, respectively.

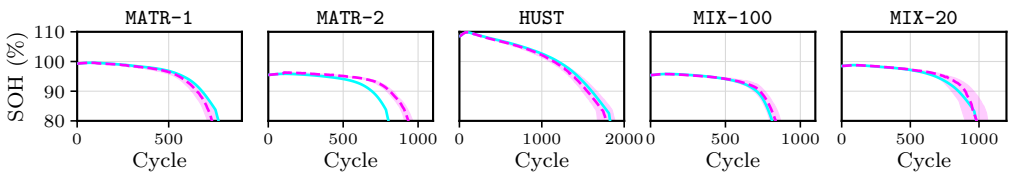


Figure 3: Example SOH prediction intervals obtained from FlowBatt across the five benchmark datasets. The cyan curve denotes the reference SOH trajectory, the magenta dashed curve denotes the median prediction, and the shaded region denotes the empirical 90% prediction interval. The displayed batteries are chosen as the test sample with median predictive uncertainty.

3.3 RUL prediction

We evaluate three different models trained with supervised learning, diffusion processes, and flow matching, referred to as Transformer, DiffBatt, and FlowBatt, respectively. All models predict SOH trajectories from the input capacity matrix, from which the RUL is computed as the cycle index at which the SOH drops below a specified threshold (e.g., 80% or 90%). Thus, in contrast to the SOH task considered in section 3.1, RUL prediction is treated here as a downstream task derived from the predicted degradation trajectories. For the probabilistic models (DiffBatt and FlowBatt), a single model is trained and evaluated by generating ten SOH trajectories from ten independent noise realizations for each input, thereby quantifying predictive variability. RUL is extracted separately from each generated trajectory before aggregating the resulting scalar predictions. For the deterministic baseline (Transformer), ten models with different random initializations are trained, and we report the mean and standard deviation of their prediction errors.

Table 3: Comparison with baseline methods for the RUL prediction task. Best results based on RMSE are shown in bold, and second-best results are underlined. For the models introduced in this study, we report the mean and standard deviation of each error metric across ten generative draws from one trained checkpoint for FlowBatt and DiffBatt, and across ten training initializations for Transformer.

Method	MATR-1		MATR-2		HUST		MIX-100		MIX-20	
	RMSE	MAPE (%)	RMSE	MAPE (%)	RMSE	MAPE (%)	RMSE	MAPE (%)	RMSE	MAPE (%)
Training Mean	399	28	511	36	420	18	573	59	593	102
"Variance" Model [43]	138	15	196	12	398	17	521	39	601	95
"Discharge" Model [43]	86	8	173	11	322	14	1743	47	>2000	>100
"Full" Model [43]	100	11	214	12	335	14	331	22	441	53
Ridge Regression [3]	125	13	188	11	1047	36	395	30	806	150
PCR [3]	100	11	176	11	435	19	384	28	701	78
PLSR [3]	97	10	193	11	431	18	371	26	543	77
SVM Regression [52]	140	15	300	18	344	16	257	18	438	46
Random Forest [3]	140	15	202	11	348	16	211	14	288	31
MLP [3]	162 ± 7	12 ± 0	207 ± 4	11 ± 0	444 ± 5	18 ± 1	455 ± 37	27 ± 1	532 ± 25	61 ± 6
LSTM	123 ± 11	12 ± 2	226 ± 36	14 ± 2	442 ± 32	20 ± 1	266 ± 11	15 ± 1	417 ± 62	37 ± 7
CNN [3]	115 ± 96	9 ± 6	237 ± 107	17 ± 8	445 ± 35	21 ± 1	261 ± 38	15 ± 1	785 ± 132	41 ± 4
BatLiNet [52]	$\mathbf{59 \pm 2}$	$\mathbf{6 \pm 0}$	$\mathbf{163 \pm 12}$	$\mathbf{11 \pm 1}$	$\underline{264 \pm 9}$	$\underline{10 \pm 1}$	$\mathbf{158 \pm 7}$	$\mathbf{10 \pm 0}$	$\mathbf{201 \pm 18}$	$\mathbf{18 \pm 1}$
Transformer (ours)	85 ± 10	8 ± 1	226 ± 39	14 ± 2	295 ± 18	14 ± 1	221 ± 34	15 ± 3	312 ± 40	27 ± 5
DiffBatt (ours)	68 ± 3	6 ± 0	202 ± 5	16 ± 0	282 ± 16	10 ± 1	218 ± 17	13 ± 1	295 ± 24	30 ± 2
FlowBatt (ours)	$\underline{67 \pm 2}$	$\underline{6 \pm 0}$	175 ± 4	11 ± 0	$\mathbf{222 \pm 12}$	$\mathbf{9 \pm 0}$	$\underline{179 \pm 6}$	$\underline{11 \pm 0}$	$\underline{229 \pm 12}$	$\underline{18 \pm 1}$

The RUL prediction results are summarized in table 3. FlowBatt shows strong and consistent performance across all datasets, achieving the best result on HUST (222 ± 12 cycles, 9% MAPE) and second-best results on MATR-1 (67 ± 2), MIX-100 (179 ± 6), and MIX-20 (229 ± 12). This demonstrates that FlowBatt remains highly competitive on the downstream RUL task, despite being primarily designed for full SOH trajectory modeling rather than direct endpoint prediction. The comparison with BatLiNet [52] is particularly relevant, as BatLiNet is specifically optimized for RUL prediction. A model designed and trained specifically for RUL may therefore perform better on this scalar task than a model trained for complete SOH trajectory prediction. On MATR-2, FlowBatt achieves an RMSE of 175 ± 4, close to BatLiNet (163 ± 12), while outperforming DiffBatt (202 ± 5) and Transformer (226 ± 39). More generally, FlowBatt is the only model besides BatLiNet

that consistently ranks among the top three across all datasets, indicating strong robustness and generalization.

Overall, these results suggest that flow-matching-based generative modeling is effective not only for SOH trajectory prediction, but also for RUL estimation derived from predicted degradation trajectories, even without being specialized exclusively for RUL.

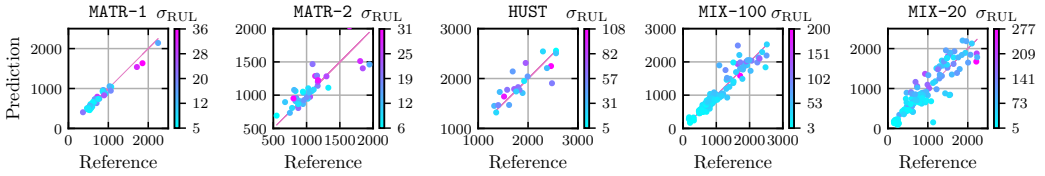


Figure 4: Results obtained from FlowBatt for RUL prediction. Colorbar shows the standard deviation σ_{RUL} of the predicted RUL from ten generated SOH trajectories per sample.

Figure 4 presents the RUL prediction results for all test samples across datasets using the FlowBatt model. FlowBatt provides uncertainty estimates by reporting the standard deviation of RUL values computed from ten generated trajectories per sample. The results suggest a general tendency for samples with larger prediction errors to also exhibit higher predictive variance.

3.4 Battery degradation synthesis

Conducting comprehensive battery degradation experiments in a real-world setting is often time-consuming and resource-intensive, requiring years to complete. Synthesizing battery degradation data offers significant advantages by reducing the time and cost of long-term experiments and ensuring consistent, high-quality data. FlowBatt can generate battery degradation curves, enabling the creation of large datasets that can enhance model training and robustness. To evaluate the effectiveness of our approach, we utilize the FlowBatt model trained on the `MATR-1` dataset to generate five synthetic SOH curves for each sample in the training set, using varying rates for classifier-free diffusion guidance. The guidance scale is denoted by w, with $w = 1$ corresponding to standard conditional generation [18]. For this synthesis setting, the early-cycle capacity conditioning vector is dropped with probability 0.2 during FlowBatt training; for the prediction experiments, the conditioning dropout probability is 0.0. We then leverage these synthesized SOH curves to train a random forest (RF) model for predicting RUL, while adding Gaussian noise with standard deviation equal to 10% of the capacity-matrix standard deviation for data augmentation. To assess the quality and diversity of our generated samples, we use FID [17], which provides insight into the similarity between real and synthetic samples, Recall [24] as the main metric for measuring diversity, and the RMSE and MAPE of the random forest model on the test dataset, serving as measures of its predictive accuracy. We also report Precision [24] for completeness. Note that FID is computed using principal component analysis (PCA) to map the data into a latent space instead of the Inception model. Results are summarized in table 4 against a random forest model trained without synthetic data following [51]. The synthetic data generated with varying guidance strengths are depicted in figure 9.

Table 4: Results for the `MATR-1` dataset obtained using random forests trained on synthetic data generated with different diffusion guidance w, against a random forest trained without synthetic data following [51]. Best values per RUL error metrics, determined before rounding, are shown in bold.

		w				
Metric	RF	0.5	1.0	2.0	3.0	4.0
FID (↓)	NA	0.129	0.001	0.001	0.002	0.003
Precision (↑)	NA	0.97	1.00	1.00	1.00	1.00
Recall (↑)	NA	0.95	0.98	0.59	0.56	0.54
RUL RMSE (↓)	136	119	**101**	102	104	107
RUL MAPE (%) (↓)	12	11	**10**	10	10	10

On the one hand, the FID scores indicate high similarity between the real and synthetic data across the guided settings. The FID decreases from 0.129 at $w = 0.5$ to 0.001 for $w = 1.0$ and $w = 2.0$. It then increases to 0.003 at $w = 4.0$. On the other hand, Recall represents the diversity by measuring

the fraction of the training data manifold covered by the generated samples. The highest Recall is obtained with $w = 1.0$, while increasing the guidance strength further reduces diversity, leading to a Recall of 0.54 for $w = 4.0$. Precision reaches 1.00 for all settings with $w \geq 1.0$. The RUL prediction metrics provide insight into the predictive performance of the random forest model. The lowest RMSE of 101 cycles and lowest MAPE of 10% are achieved with $w = 1.0$, corresponding to approximately 26% lower RMSE and 19% lower MAPE than the random forest model trained without synthetic data. All synthetic-data settings improve over the random forest model trained without synthetic data, which obtains an RMSE of 136 cycles and a MAPE of 12%.

We additionally compare FlowBatt synthetic data with a naive augmentation baseline that adds Gaussian noise to capacity matrices and SOH trajectories (Gaussian jitter). We also include MLP as a second downstream predictor. The setup and results in appendix A.5.1 show lower reported RUL RMSE with FlowBatt synthetic data for both models, with a larger reduction for RF than for MLP.

4 Conclusions and outlook

Battery degradation remains a major challenge for reliable and sustainable energy-storage systems. In this work, we introduced FlowBatt, a use-inspired adaptation of flow matching with a diffusion transformer backbone for probabilistic battery degradation trajectory modeling. Rather than predicting only an endpoint such as RUL, FlowBatt learns full SOH trajectories conditioned on early-cycle capacity data, providing a trajectory-level representation of battery aging and a unified model for forecasting, uncertainty estimation, and synthesis. Across all five benchmark datasets, FlowBatt obtains the best trajectory-wise SOH prediction results. Since the model generates multiple plausible degradation trajectories for the same early-life input, it also provides empirical prediction intervals that capture predictive uncertainty. In the downstream RUL task, RUL values derived from FlowBatt trajectories remain competitive with specialized RUL baselines, including BatLiNet [52], and rank among the top methods across all evaluated datasets. The synthesis experiments further show that FlowBatt-generated SOH trajectories can improve downstream RUL modeling, reducing random-forest RMSE and MAPE on `MATR-1` by approximately 26% and 19%, respectively.

Limitations. Prediction remains challenging on `MATR-2` and `HUST`, as reflected in higher RUL errors than on `MATR-1` and more pronounced undercoverage of the empirical prediction intervals. The model conditions only on capacity matrices from fixed 20- or 100-cycle windows, without direct temperature, current-profile, charging-protocol, environmental, or cooling inputs. The fixed benchmark splits do not establish generalization to unseen chemistries or cycling protocols. FlowBatt also does not explicitly enforce electrochemical, thermal, or conservation constraints.

Outlook. These results support flow matching as a promising framework for probabilistic battery degradation modeling. Future work should improve uncertainty calibration, especially under distribution shifts across chemistries and cycling protocols; incorporate additional operating and environmental variables; and connect generative trajectory models more closely with physics-based constraints and descriptions of battery aging. Generalization should be assessed by holding out entire chemistries or cycling protocols. Future work should include controlled evaluation of conditioning-history length and explore encoder architectures capable of processing variable-length cycle sequences. Scaling this framework to broader battery datasets may also provide a path toward more general foundation models for battery health prediction.

Acknowledgments and disclosure of funding

This research was conducted within the Research Training Group CircularLIB, supported by the Ministry of Science and Culture of Lower Saxony with funds from the program zukunft.niedersachsen of the Volkswagen Foundation (MWK | ZN3678). This work was also supported by the Lower Saxony Digital Science Support Space (DS^3) project as part of Hochschule.digital Niedersachsen, funded by zukunft.niedersachsen. We would like to thank the anonymous reviewers who contributed to the improvement of our manuscript with their valuable suggestions and comments.

References

[1] M. S. Albergo and E. Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

[2] P. M. Attia, A. Grover, N. Jin, K. A. Severson, T. M. Markov, Y.-H. Liao, M. H. Chen, B. Cheong, N. Perkins, Z. Yang, P. K. Herring, M. Aykol, S. J. Harris, R. D. Braatz, S. Ermon, and W. C. Chueh. Closed-loop optimization of fast-charging protocols for batteries with machine learning. *Nature*, 578: 397–402, 2020. doi: 10.1038/s41586-020-1994-5.

[3] P. M. Attia, K. A. Severson, and J. D. Witmer. Statistical learning for accurate and interpretable battery lifetime prediction. *Journal of The Electrochemical Society*, 168(9):090547, 2021. doi: 10.1149/1945-7111/ac2704.

[4] J.-H. Bastek, W. Sun, and D. M. Kochmann. Physics-informed diffusion models. *arXiv preprint arXiv:2403.14404*, 2024. URL https://arxiv.org/abs/2403.14404.

[5] S. Blömeke, C. Scheller, F. Cerdas, C. Thies, R. Hachenberger, M. Gonter, C. Herrmann, and T. S. Spengler. Material and energy flow analysis for environmental and economic impact assessment of industrial recycling routes for lithium-ion traction batteries. *Journal of Cleaner Production*, 377:134344, 2022. doi: 10.1016/j.jclepro.2022.134344.

[6] G. E. Blomgren. The development and future of lithium ion batteries. *Journal of The Electrochemical Society*, 164(1):A5019–A5025, 2017. doi: 10.1149/2.0251701jes.

[7] M. Broussely, P. Biensan, F. Bonhomme, P. Blanchard, S. Herreyre, K. Nechev, and R. Staniewicz. Main aging mechanisms in Li-ion batteries. *Journal of Power Sources*, 146(1-2):90–96, 2005. doi: 10.1016/j.jpowsour.2005.03.172.

[8] P. V. Chombo and Y. Laoonual. A review of safety strategies of a Li-ion battery. *Journal of Power Sources*, 478:228649, 2020. doi: 10.1016/j.jpowsour.2020.228649.

[9] Z. Cui, L. Wang, Q. Li, and K. Wang. A comprehensive review on the state of charge estimation for lithium-ion battery based on neural network. *International Journal of Energy Research*, 46(5):5423–5440, 2022. doi: 10.1002/er.7545.

[10] A. Devie, G. Baure, and M. Dubarry. Intrinsic variability in the degradation of a batch of commercial 18650 lithium-ion cells. *Energies*, 11(5), 2018. doi: 10.3390/en11051031.

[11] P. Dhariwal and A. Q. Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=AAWuCvzaVt.

[12] J. S. Edge, S. O'Kane, R. Prosser, N. D. Kirkaldy, A. N. Patel, A. Hales, A. Ghosh, W. Ai, J. Chen, J. Yang, S. Li, M.-C. Pang, L. Bravo Diaz, A. Tomaszewska, M. W. Marzook, K. N. Radhakrishnan, H. Wang, Y. Patel, B. Wu, and G. J. Offer. Lithium ion battery degradation: what you need to know. *Physical Chemistry Chemical Physics*, 23(14):8200–8221, 2021. doi: 10.1039/D1CP00359C.

[13] F. Fürrutter, G. Muñoz-Gil, and H. J. Briegel. Quantum circuit synthesis with diffusion models. *Nature Machine Intelligence*, 6(5):515–524, 2024. doi: 10.1038/s42256-024-00831-9.

[14] M. Galeotti, L. Cinà, C. Giammanco, S. Cordiner, and A. Di Carlo. Performance analysis and SOH (state of health) evaluation of lithium polymer batteries through electrochemical impedance spectroscopy. *Energy*, 89:678–686, 2015. doi: 10.1016/j.energy.2015.05.148.

[15] R. Ginster, S. Blömeke, J. Popien, C. Scheller, F. Cerdas, C. Herrmann, and T. S. Spengler. Circular battery production in the EU: Insights from integrating life cycle assessment into system dynamics modeling on recycled content and environmental impacts. *Journal of Industrial Ecology*, 28(5):1165–1182, 2024. doi: 10.1111/jiec.13527.

[16] 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. *Journal of Power Sources*, 196(23):10314–10321, 2011. doi: 10.1016/j.jpowsour.2011.08.040.

[17] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

[18] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org/abs/2207.12598.

[19] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper_files/

paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[20] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022. URL http://jmlr.org/papers/v23/21-0635.html.

[21] J. Hu, P. Fu, Z. Wei, Y. Huang, J. Early, A. Fly, and Y. Zhang. Early prediction of lithium-ion battery degradation with a generative pre-trained transformer. *Nature Communications*, 17:126, 2026. doi: 10.1038/s41467-025-66819-0.

[22] D. Juarez-Robles, J. A. Jeevarajan, and P. P. Mukherjee. Degradation-safety analytics in lithium-ion cells: Part I. aging under charge/discharge cycling. *Journal of The Electrochemical Society*, 167(16):160510, 2020. doi: 10.1149/1945-7111/abc8c0.

[23] D. Juarez-Robles, S. Azam, J. A. Jeevarajan, and P. P. Mukherjee. Degradation-safety analytics in lithium-ion cells and modules: Part III. aging and safety of pouch format cells. *Journal of The Electrochemical Society*, 168(11):110501, 2021. doi: 10.1149/1945-7111/ac30af.

[24] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/0234c510bc6d908b28c70ff313743079-Abstract.html.

[25] A. G. Li, A. C. West, and M. Preindl. Towards unified machine learning characterization of lithium-ion battery degradation across multiple levels: A critical review. *Applied Energy*, 316:119030, 2022. doi: 10.1016/j.apenergy.2022.119030.

[26] T. Li, L. Biferale, F. Bonaccorso, M. A. Scarpolini, and M. Buzzicotti. Synthetic Lagrangian turbulence by generative diffusion models. *Nature Machine Intelligence*, 6(4):393–403, 2024. doi: 10.1038/s42256-024-00810-0.

[27] W. Li, N. Sengupta, P. A. Dechent, D. Howey, A. Annaswamy, and D. U. Sauer. One-shot battery degradation trajectory prediction with deep learning. *Journal of Power Sources*, page 230024, 2021. doi: 10.1016/j.jpowsour.2021.230024.

[28] X. Li, D. Yu, V. Søren Byg, and S. Daniel Ioan. The development of machine learning-based remaining useful life prediction for lithium-ion batteries. *Journal of Energy Chemistry*, 82:103–121, 2023. doi: 10.1016/j.jechem.2023.03.026.

[29] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[30] Y. Lipman, M. Havasi, P. Holderrieth, N. Shaul, M. Le, B. Karrer, R. T. Q. Chen, D. Lopez-Paz, H. Ben-Hamu, and I. Gat. Flow matching guide and code, 2024. URL https://arxiv.org/abs/2412.06264.

[31] X. Liu, C. Gong, and Q. Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[32] C. Luo, Z. Zhang, S. Zhu, and Y. Li. State-of-health prediction of lithium-ion batteries based on diffusion model with transfer learning. *Energies*, 16(9):3815, 2023. doi: 10.3390/en16093815.

[33] L. Luzi, P. M. Mayer, J. Casco-Rodriguez, A. Siahkoohi, and R. G. Baraniuk. Boomerang: Local sampling on image manifolds using diffusion models. *arXiv preprint arXiv:2210.12100*, 2024. URL https://arxiv.org/abs/2210.12100.

[34] G. Ma, S. Xu, B. Jiang, C. Cheng, X. Yang, Y. Shen, T. Yang, Y. Huang, H. Ding, and Y. Yuan. Real-time personalized health status prediction of lithium-ion batteries using deep transfer learning. *Energy & Environmental Science*, 2022. doi: 10.1039/D2EE01676A.

[35] N. Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(35):983–999, 2006. URL https://www.jmlr.org/papers/v7/meinshausen06a.html.

[36] S. E. J. O'Kane, W. Ai, G. Madabattula, D. Alonso-Alvarez, R. Timms, V. Sulzer, J. S. Edge, B. Wu, G. J. Offer, and M. Marinescu. Lithium-ion battery degradation: how to model it. *Physical Chemistry Chemical Physics*, 24(13):7909–7922, 2022. doi: 10.1039/D2CP00417H.

[37] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4172–4182, 2023. doi: 10.1109/ICCV51070.2023.00387.

[38] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3942–3951. AAAI Press, 2018. URL https://doi.org/10.1609/aaai.v32i1.11671.

[39] Y. Preger, H. M. Barkholtz, A. Fresquez, D. L. Campbell, B. W. Juba, J. Romàn-Kustas, S. R. Ferreira, and B. Chalamala. Degradation of commercial lithium-ion cells as a function of chemistry and cycling conditions. *Journal of The Electrochemical Society*, 167(12):120532, 2020. doi: 10.1149/1945-7111/abae37.

[40] H. Rauf, M. Khalid, and N. Arshad. Machine learning in state of health and remaining useful life estimation: Theoretical and technological development in battery degradation modelling. *Renewable and Sustainable Energy Reviews*, 156:111903, 2022. doi: 10.1016/j.rser.2021.111903.

[41] Z. Ren and C. Du. A review of machine learning state-of-charge and state-of-health estimation algorithms for lithium-ion batteries. *Energy Reports*, 9:2993–3021, 2023. doi: 10.1016/j.egyr.2023.01.108.

[42] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017. doi: 10.1109/ICCV.2017.74.

[43] K. A. Severson, P. M. Attia, N. Jin, N. Perkins, B. Jiang, Z. Yang, M. H. Chen, M. Aykol, P. K. 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*, 4(5):383–391, 2019. doi: 10.1038/s41560-019-0356-8.

[44] L. Sluijterman, E. Cator, and T. Heskes. How to evaluate uncertainty estimates in machine learning for regression?, Aug. 2023. URL https://arxiv.org/abs/2106.03395v2. Version 2, 3 August 2023.

[45] J. Vetter, P. Novák, M. Wagner, C. Veit, K.-C. Möller, J. Besenhard, M. Winter, M. Wohlfahrt-Mehrens, C. Vogler, and A. Hammouche. Ageing mechanisms in lithium-ion batteries. *Journal of Power Sources*, 147(1-2):269–281, 2005. doi: 10.1016/j.jpowsour.2005.01.006.

[46] P. von Platen, S. Patil, A. Lozhkov, P. Cuenca, N. Lambert, K. Rasul, M. Davaadorj, D. Nair, S. Paul, W. Berman, Y. Xu, S. Liu, and T. Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.

[47] S. Wang, S. Jin, D. Bai, Y. Fan, H. Shi, and C. Fernandez. A critical review of improved deep learning methods for the remaining useful life prediction of lithium-ion batteries. *Energy Reports*, 7:5562–5574, 2021. doi: 10.1016/j.egyr.2021.08.182.

[48] Y. Xing, E. W. Ma, K.-L. Tsui, and M. Pecht. An ensemble model for predicting the remaining useful performance of lithium-ion batteries. *Microelectronics Reliability*, 53(6):811–820, 2013. doi: 10.1016/j.microrel.2012.12.003.

[49] Y. Yang, M. Jin, H. Wen, C. Zhang, Y. Liang, L. Ma, Y. Wang, C. Liu, B. Yang, Z. Xu, J. Bian, S. Pan, and Q. Wen. A survey on diffusion models for time series and spatio-temporal data. *arXiv preprint arXiv:2404.18886*, 2024. URL https://arxiv.org/abs/2404.18886.

[50] B. Zhang, P. Xu, X. Chen, and Q. Zhuang. Generative quantum machine learning via denoising diffusion probabilistic models. *Physical Review Letters*, 132:100602, 2024. doi: 10.1103/PhysRevLett.132.100602.

[51] H. Zhang, X. Gui, S. Zheng, Z. Lu, Y. Li, and J. Bian. BatteryML: An open-source platform for machine learning on battery degradation. In *The Twelfth International Conference on Learning Representations*, 2024. URL http://arxiv.org/abs/2310.14714.

[52] H. Zhang, Y. Li, S. Zheng, Z. Lu, X. Gui, W. Xu, and J. Bian. Battery lifetime prediction across diverse ageing conditions with inter-cell deep learning. *Nature Machine Intelligence*, 7(2):270–277, 2025. doi: 10.1038/s42256-024-00972-x.

A Appendix

Appendix contents

A.1 Datasets . . . 14
A.2 Details of FlowBatt model . . . 15
A.2.1 Capacity matrix . . . 15
A.2.2 Hyperparameters . . . 15
A.2.3 Compute resources . . . 16
A.2.4 DiT block with adaLN-Zero conditioning . . . 16
A.3 Explaining CNN performance via Grad-CAM . . . 17
A.4 Experimental setup . . . 18
A.4.1 BatteryML . . . 18
A.4.2 BatLiNet . . . 19
A.5 Additional SOH results . . . 20
A.5.1 Synthetic SOH trajectory samples . . . 20
A.5.2 Trajectory-wise SOH prediction . . . 21
A.5.3 Cycle-wise SOH prediction . . . 22
A.5.4 Uncertainty quantification . . . 24
A.6 Broader impacts . . . 24

A.1 Datasets

Battery degradation curves utilized for training and testing the models are depicted in figure 5 for each cell chemistry. The benchmark settings follow BatLiNet [52] with fixed train/test cell splits. The final train/test battery counts used in our experiments are summarized in table 5. For `MIX-20`, our training set retains 49 cells that reach 90% SOH before 100 cycles, which are excluded by BatLiNet. The resulting training set comprises 256 cells, compared with BatLiNet's 207, while the test set remains unchanged at 147 cells.

Table 5: Benchmark composition, conditioning horizon, and final train/test battery counts.

Benchmark	Source data	Chemistry	Early cycles	Train/test
`MATR-1`	`MATR-1`	LFP/graphite	100	41/42
`MATR-2`	`MATR-2`	LFP/graphite	100	41/40
`HUST`	`HUST`	LFP/graphite	100	55/22
`MIX-20`	Combined	LCO, LFP, NCA, NMC, NMC–LCO	20	256/147
`MIX-100`	Combined	LCO, LFP, NMC, NMC–LCO	100	205/137

The `CALCE` dataset includes full lifecycle data from 13 batteries with an LCO cathode. Each battery has a nominal capacity of 1100 mAh. They were all charged using a constant current/constant voltage protocol: 0.5C current until reaching $4.2\,\mathrm{V}$, maintaining $4.2\,\mathrm{V}$ until the current dropped below $0.05\,\mathrm{A}$, and a cutoff voltage of $2.7\,\mathrm{V}$ [16, 48]. The C-rate expresses the charging or discharging current relative to nominal capacity, with 1C corresponding to a current that would transfer the nominal capacity in one hour.

The `MATR` dataset, provided by Severson et al. [43] and Attia et al. [2], is one of the largest public datasets containing 180 commercial 18650 LFP batteries. These batteries, cycled at a forced convection temperature chamber of $30\,^{\circ}\mathrm{C}$, have a nominal capacity of 1.1 Ah and a nominal voltage of $3.3\,\mathrm{V}$. The dataset comprises three subsets: `MATR-1`, `MATR-2` [43], and `CLO` [2], all categorized due to distinct measurement batches.

The `HUST` dataset includes 77 LFP batteries, similar to those in the `MATR` dataset. These batteries followed an identical charging protocol with varying multi-stage discharge protocols, all conducted at a constant temperature of $30\,^{\circ}\mathrm{C}$ [34].

The `HNEI` dataset contains 14 commercial 18650 cells with a graphite anode and a blended NMC and LCO cathode. These cells were cycled at 1.5C to 100% depth of discharge for over 1000 cycles at room temperature [10].

The `SNL` dataset includes 61 commercial 18650 cells (NCA, NMC, and LFP), cycled to 80% capacity. The study evaluates the impact of temperature, depth of discharge, and discharge current on long-term degradation [39].

The UL_PUR dataset comprises 10 commercial pouch cells with a graphite negative electrode and an NCA cathode. These cells were cycled at 1C between 2.7 V and 4.2 V, equivalent to 0-100% state of charge (SOC), at room temperature until reaching 10-20% capacity fade. Additionally, modules were cycled at C/2 between 13.7 V and 21.0 V until 20% capacity fade [22, 23].

The RWTH dataset contains data from 48 lithium-ion battery cells aged under identical conditions. These cells feature a carbon anode and an NMC cathode [27]. The cells were cycled at a constant ambient temperature of 25 °C. Each cycle involved a 30-minute discharge phase down to 3.5 V and a 30-minute charge phase up to 3.9 V, with the currents capped at a maximum of 4 A. This resulted in cycles between approximately 20% and 80% state of charge.

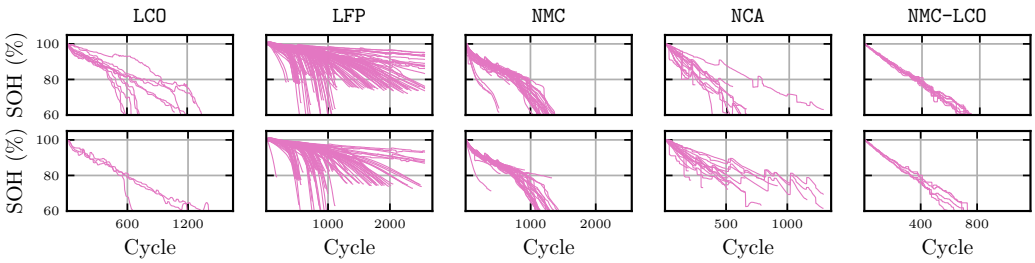


Figure 5: Train (up) and test (bottom) samples for each cell chemistry. The data is scaled using the SOH of the first cycle.

Licenses and terms of use. All benchmark datasets utilized in this study are publicly available for academic research and are curated within the open-source BatteryML framework [51], which is released under the MIT License. The PyTorch and diffusers [46] libraries used for our implementation are also open-source and their respective licenses (e.g., BSD-style, Apache 2.0) are strictly respected.

A.2 Details of FlowBatt model

A.2.1 Capacity matrix

For each battery, the discharge-capacity curve at observed cycle i is interpolated onto a common grid of 100 voltage points. The capacity matrix is a cycle-by-voltage array of baseline-subtracted discharge-capacity curves, $\boldsymbol{Q} = [M_{i,j}]$, with entries

$$M_{i,j} = Q_d^{(i)}(V_j) - Q_d^{(10)}(V_j), \tag{6}$$

where i is the observed-cycle index, j is the index on the common voltage grid, V_j is the voltage at grid point j, and $Q_d^{(i)}(V_j)$ is the discharge capacity at voltage V_j during cycle i. Cycle 10 is the reference curve subtracted from every observed cycle. The matrix therefore contains only baseline-subtracted discharge-capacity curves and records how the capacity–voltage response evolves during early cycling. We selected this representation following Severson et al. [43], Attia et al. [3], and Zhang et al. [51], which identify early-cycle differential discharge-capacity data as informative for battery degradation and lifetime. Its voltage-dependent structure reflects changes in energy dissipation and may capture open-circuit and kinetic effects, heterogeneous aging, and early voltage signatures of degradation. In particular, Attia et al. [3] reported that a model using only $\log_{10}\mathrm{Var}[\Delta Q_{100-10}(V)]$ achieved lower RMSE than their full model containing temperature and internal-resistance features on the secondary MATR test set. In the present study, FlowBatt does not receive temperature, internal resistance, current profiles, charging protocols, or environmental conditions as direct inputs. However, the conditioning pathway can be extended to incorporate these variables when they are available.

A.2.2 Hyperparameters

Table 6 summarizes the hyperparameters used for training our FlowBatt model. For implementing flow matching, we build upon the official `flow_matching` library from Facebook Research [30]. For diffusion-related components, including optimizers and schedulers, we rely on the Hugging Face `diffusers` package [46].

For model development, we preprocess the SOH trajectories; we limit each trajectory to a maximum of 2560 charge-discharge cycles and subsample the SOH values every ten cycles. Furthermore, we

Table 6: Training and model hyperparameters.

Category	Major hyperparameters
Training	Epochs: 10000 Initial learning rate: 1×10^{-3} Optimizer: AdamW Learning rate scheduler: Cosine schedule with 100 warmup steps
Model (DiT)	Sequence length: 256 Capacity matrix shape: $(1, 100, 100)$, or $(1, 20, 100)$ for `MIX-20` Number of DiT blocks: 4 Channels (embedding dim): 16 Attention heads: 1 MLP ratio: 4.0 Conditioning dropout probability: 0.0 (default), 0.2 (SOH synthesis, table 4)
Flow matching	Scheduler: Conditional optimal transport schedule

truncate the trajectory once the SOH drops below 70%, since this threshold is typically considered below the EOL criterion for many applications. If a trajectory reaches this SOH cutoff before 2560 cycles, we pad the remaining part of the sequence with a constant SOH value of 70%. As a result, all SOH trajectories are represented as fixed-length sequences of 256 steps, which defines the input dimension of our model.

A.2.3 Compute resources

All experiments, including model training, baseline evaluations, and trajectory generation, were conducted using PyTorch on a workstation equipped with a single NVIDIA RTX 3090 GPU (24 GB VRAM) and a 12th Gen Intel Core i9-12900K CPU. Training the FlowBatt model for 10,000 epochs takes approximately 30 minutes per dataset. On a single GPU, predicting one complete SOH trajectory per battery in a batch of 42 `MATR-1` test batteries takes approximately 3 ms for Transformer, 0.5 s for FlowBatt, and 2.6 s for DiffBatt, averaged over three runs following one warm-up. FlowBatt therefore achieves approximately a fivefold inference speedup over DiffBatt.

A.2.4 DiT block with adaLN-Zero conditioning

In both FlowBatt and DiffBatt models, each DiT block updates the tokenized SOH representation using self-attention and an MLP, while conditioning both sublayers on the flow or diffusion timestep and the early-cycle capacity matrix. The timestep embedding and CNN-based capacity embedding are combined into a global conditioning vector, which is mapped to adaptive layer-normalization shifts, scale offsets, and residual gates following adaLN-Zero [37]. The residual gates are initialized near zero, so the block initially behaves close to an identity map and the conditioning effect is learned gradually during training. The computational steps for one block are summarized in algorithm 1.

Algorithm 1: One DiT block with adaLN-Zero conditioning

Input: Hidden states $\boldsymbol{h} \in \mathbb{R}^{T \times d}$, timestep t, auxiliary input $\boldsymbol{Q}$ (capacity matrix).
Output: Updated hidden states $\boldsymbol{h}' \in \mathbb{R}^{T \times d}$.

`/* Step 1: Form conditioning vector */`
$\boldsymbol{t}_{\text{emb}} \leftarrow \phi(t)$ `// timestep embedding`
$\boldsymbol{c}_q \leftarrow \text{CNN}(\boldsymbol{Q})$ `// auxiliary embedding`
$\boldsymbol{c} \leftarrow \boldsymbol{t}_{\text{emb}} + \boldsymbol{c}_q$ `// global conditioning vector`
`/* Step 2: Generate modulation parameters */`
$(\gamma_{\text{attn}}, \beta_{\text{attn}}, \alpha_{\text{attn}}, \gamma_{\text{mlp}}, \beta_{\text{mlp}}, \alpha_{\text{mlp}}) \leftarrow \text{MLP}(\boldsymbol{c})$
`/* Step 3: Attention sublayer */`
$\boldsymbol{z} \leftarrow \text{LN}(\boldsymbol{h})$ `// pre-norm`
$\boldsymbol{z} \leftarrow (1 + \gamma_{\text{attn}}) \odot \boldsymbol{z} + \beta_{\text{attn}}$ `// adaLN modulation`
$\boldsymbol{h} \leftarrow \boldsymbol{h} + \alpha_{\text{attn}} \odot \text{MSA}(\boldsymbol{z})$ `// residual update`
`/* Step 4: MLP sublayer */`
$\boldsymbol{z} \leftarrow \text{LN}(\boldsymbol{h})$ `// pre-norm`
$\boldsymbol{z} \leftarrow (1 + \gamma_{\text{mlp}}) \odot \boldsymbol{z} + \beta_{\text{mlp}}$ `// adaLN modulation`
$\boldsymbol{h}' \leftarrow \boldsymbol{h} + \alpha_{\text{mlp}} \odot \text{MLP}(\boldsymbol{z})$ `// residual update`
return $\boldsymbol{h}'$

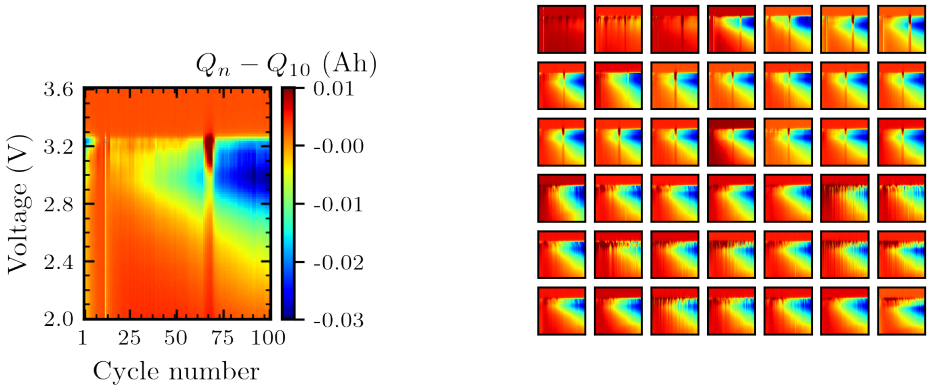
A.3 Explaining CNN performance via Grad-CAM

For the RUL prediction task, we implement a CNN model based on the architecture proposed in [3, 51, 52]. The model consists of convolutional layers with ReLU activation functions, followed by a fully connected layer. The model is trained using the Adam optimizer with a learning rate of 0.001 and a batch size of 128 (this is equal to full-batch training for `MATR-1` dataset) for 1000 epochs. The model is trained via mean-squared error (MSE) as the loss function. We train ten models with different initialization seeds and report the RMSE for each model on the test dataset. The results are summarized in table 7 for different seeds. As observed in the table, the CNN model exhibits high variability in performance across different seeds, with RMSE values ranging from 60 to 306 cycles. This variability indicates that the model's performance is sensitive to initialization and may not consistently capture the underlying degradation patterns. This lack of robustness has been also reported in [3, 51, 52]. To identify the reason behind the high variability in performance, we employ

Table 7: RUL prediction results obtained from CNN models with different initialization seeds for `MATR-1` dataset.

Seed	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Benchmark CNN [51]	76	67	64	74	60	82	65	79	367	78
CNN (Replicated)	71	77	83	306	79	60	78	62	88	64
XAI-CNN (Ours)	57	54	54	72	66	64	65	60	59	61

Grad-CAM [42] to visualize the regions of the input capacity matrix that the CNN model focuses on when making predictions. Figure 6a shows one sample of the capacity matrix from the `MATR-1` train dataset, while figure 6b displays all test samples of the `MATR-1` dataset. The heatmaps in figure 7 illustrate the Grad-CAM results for the best and worst-performing CNN models based on RMSE. The heatmaps obtained from the worst model indicate that the gradients for some test samples are zero, suggesting that the model fails to perform meaningful predictions for these samples. Same observations for the training samples indicate that the model fails to learn from those training samples. Moreover, the model tends to focus on early cycles, which may not provide sufficient information for accurate RUL prediction. In contrast, the heatmaps for the best-performing model show that the model focuses on a broader range of cycles, including later ones that are more indicative of degradation trends.

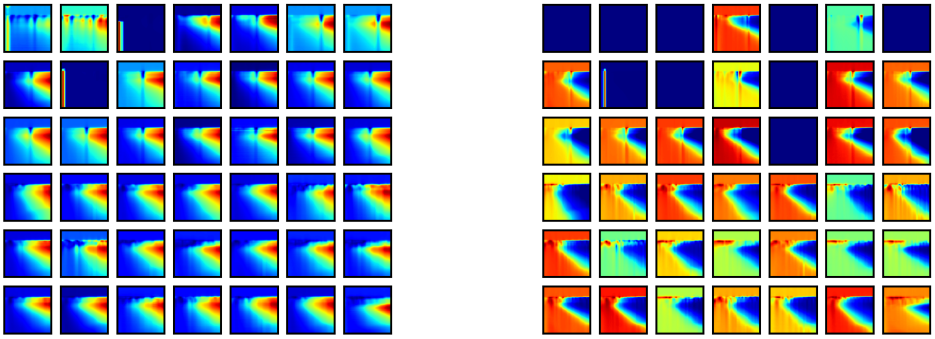


(a) Capacity matrix

(b) `MATR-1` test samples

Figure 6: (a) One sample of capacity matrix from `MATR-1` train dataset. (b) All test samples of `MATR-1` dataset.

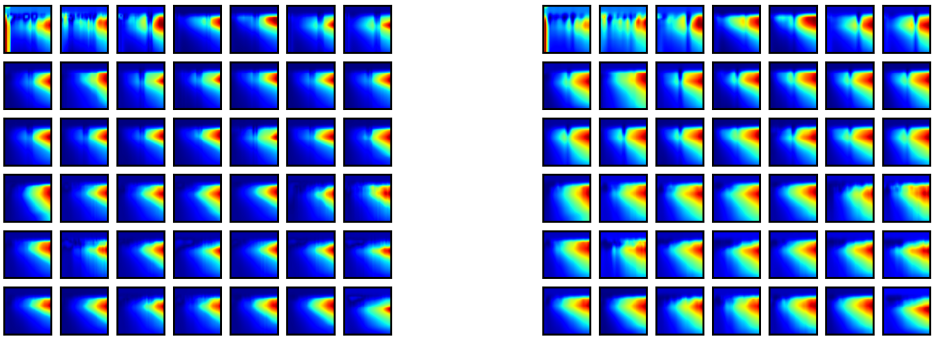
To enhance the robustness and accuracy of the CNN model, we provide a simple remedy by replacing the ReLU activation functions with Leaky ReLU activation functions with negative slope equal to 0.3. This choice is due to the realization that the ReLU activation function can lead to dead neurons, especially when the input data contains negative values or when the weights are initialized poorly. Dead neurons do not contribute to learning, which can significantly impact the model's performance and robustness. Leaky ReLU allows a small, non-zero gradient when the unit is not active, which helps to keep the neurons alive and ensures that they continue to learn during training.



(a) Grad-CAM heatmaps for best model (b) Grad-CAM heatmaps for worst model

Figure 7: Grad-CAM heatmaps for CNN model proposed in [3, 51].

This modification helps to mitigate the issue of dead neurons and improves the overall robustness of the CNN model. We refer to this modified model as XAI-CNN. The XAI-CNN model is trained using



(a) Grad-CAM heatmaps for best model (b) Grad-CAM heatmaps for worst model

Figure 8: Grad-CAM heatmaps for XAI-CNN model.

the same hyperparameters as the CNN model. The RMSE results for different seeds are summarized in table 7. This model demonstrates significantly improved robustness, with RMSE values ranging from 54 to 72 cycles across different seeds, and overall provides very accurate RUL predictions with RMSE of 61 ± 5 for `MATR-1` dataset. The Grad-CAM heatmaps for the best and worst-performing XAI-CNN models are shown in figure 8. The heatmaps indicate that the XAI-CNN model consistently focuses on relevant regions of the capacity matrix across all test samples, suggesting that it effectively captures important degradation patterns. This improvement in focus likely contributes to the enhanced performance and robustness of the XAI-CNN model compared to the original CNN architecture.

We use the XAI-CNN model as the CNN encoder in FlowBatt and DiffBatt. The final linear layer of the XAI-CNN model is removed, and the output from the last convolutional layer is flattened and passed to a linear layer to match the dimension of the condition vector in FlowBatt and DiffBatt.

A.4 Experimental setup

A.4.1 BatteryML

SOH prediction. BatteryML [51] is introduced as a general benchmark framework for battery degradation prediction, including SOH-related tasks. However, at the time of our study, the public SOH workflow was not as explicit and directly reproducible as the released RUL setting. Therefore, for the BatteryML SOH experiments, we adapted the public BatteryML framework to the SOH task

while keeping the setup as close as possible to the released RUL-oriented configurations. In particular, we preserved the public dataset definitions, train-test splits, feature extractors, and transformation pipeline whenever possible, and only modified the label definition and output setting to support SOH prediction under our cycle-wise and trajectory-wise evaluation protocols.

The RF and MLP baselines of Zhang et al. [51] use the baseline-subtracted discharge-capacity curve from the final early-life cycle, rather than the complete capacity matrix defined in appendix A.2.1.

Table 8: Training and model settings for the BatteryML baselines used in the SOH experiments.

Category	Major hyperparameters
Evaluation	Cycle-wise SOH Trajectory-wise SOH
MLP	Input representation: capacity matrix Input settings: final early-life curve minus baseline curve Target: cycle-wise SOH or SOH trajectory Trajectory settings: max cycles $= 2560$, stride $= 10$, SOH cutoff $= 0.7$ Label transformation: Z-score Optimizer: Adam, learning rate $= 10^{-3}$ Epochs: 5000, batch size: 128 Activation: `tanh`, dropout: 0.05
Random Forest	Input representation: capacity matrix Input settings: final early-life curve minus baseline curve Target: cycle-wise SOH or SOH trajectory Trajectory settings: max cycles $= 2560$, stride $= 10$, SOH cutoff $= 0.7$ Label transformation: Z-score Estimator: default Random Forest training settings

We evaluated the RF and MLP baselines of Zhang et al. [51] under the same dataset definitions and train-test splits used throughout this study. For cycle-wise SOH prediction, evaluation was performed at cycle indices 100, 200, and 300. For trajectory-wise SOH prediction, the model output was evaluated over the full SOH trajectory. We additionally evaluated full-capacity-matrix variants for trajectory prediction. RF uses the flattened matrix, whereas MLP processes the voltage and cycle dimensions sequentially. For these variants, hyperparameters were selected using only training data, through three-fold cross-validation for RF and an internal 80/20 split for MLP, followed by refitting on the complete training split. For all RF and MLP models, we report the mean and standard deviation over ten random seeds. FlowBatt achieves lower trajectory-wise RMSE than both full-matrix baselines on all five datasets (tables 1 and 9).

Table 9: Trajectory-wise SOH RMSE (%) for RF and MLP using the final early-life curve or the complete capacity matrix. Values are mean $\pm$ standard deviation over ten training runs. Final-cycle results are reproduced from table 1.

Method	`MATR-1`	`MATR-2`	`HUST`	`MIX-20`	`MIX-100`
RF (final-cycle curve)	2.01 ± 0.04	2.50 ± 0.01	3.57 ± 0.03	3.42 ± 0.02	5.45 ± 0.04
MLP (final-cycle curve)	1.97 ± 0.11	3.06 ± 0.17	4.11 ± 0.13	2.23 ± 0.09	3.13 ± 0.14
RF (full matrix)	2.26 ± 0.02	3.05 ± 0.06	4.20 ± 0.03	1.50 ± 0.01	2.51 ± 0.02
MLP (full matrix)	2.75 ± 0.08	4.04 ± 0.35	4.16 ± 0.23	2.87 ± 0.20	2.91 ± 0.14

RUL prediction. For RUL prediction, we follow the implementation and benchmark settings of Zhang et al. [51].

A.4.2 BatLiNet

SOH prediction. BatLiNet [52] was originally designed for battery lifetime prediction, i.e., the RUL task, across diverse aging conditions. To obtain SOH prediction results in our study, we adapted the original BatLiNet framework with minimal task-specific modifications. In particular, we replaced the original RUL target with an SOH target while preserving the same benchmark datasets, train-test splits, feature extraction pipeline, and model hyperparameters as closely as possible. This allowed us to evaluate BatLiNet under both cycle-wise and trajectory-wise SOH settings while keeping the experimental protocol aligned with the original benchmark design.

Table 10: Training and model settings for the BatLiNet baseline used in the SOH experiments.

Category	Major hyperparameters
Evaluation	Cycle-wise SOH; trajectory-wise SOH
Model	Input channels: 6; hidden channels: 32; input size: 100×1000; feature extractor: BatLiNetFeatureExtractor, `smooth_features=False`
Training	Epochs: 1000; train batch size: 128; test batch size: 1; train support size: 2; test support size: 32; gradient accumulation steps: 1
Target setup	Original BatLiNet uses RUL labels with log-scale and Z-score transformation. For SOH experiments, only the target label was adapted while keeping the remaining setup unchanged.

For cycle-wise SOH prediction, evaluation was performed at cycle indices 100, 200, and 300. For trajectory-wise SOH prediction, the full predicted SOH trajectory was evaluated. In all BatLiNet [52] SOH experiments, we report the mean and standard deviation over 10 random seeds.

RUL prediction. For the RUL task, we follow the original BatLiNet setting [52].

A.5 Additional SOH results

A.5.1 Synthetic SOH trajectory samples

To assess whether naive augmentation can provide similar gains, we compare Gaussian jitter with FlowBatt synthetic data for downstream RUL prediction on the `MATR-1` dataset. In addition to RF, we evaluate an MLP that takes the complete capacity matrix as input and processes its voltage and cycle dimensions sequentially. Gaussian jitter applies independent pointwise Gaussian noise to the capacity matrix and SOH trajectory, with standard deviation equal to 10% of the corresponding training-data standard deviation. RUL is then extracted from the perturbed trajectory at the first 80% SOH crossing.

Table 11: RUL RMSE (cycles) for real-only training and data augmentation on `MATR-1`. Published RF and MLP benchmark results from Zhang et al. [52] are included for reference. Values include the reported standard deviations where available. The lowest reported RMSE for each downstream model is shown in bold.

Model	Published benchmark	Real only	Gaussian jitter	FlowBatt synthetic data
RF	140	136 ± 3	110 ± 1	$\mathbf{101 \pm 2}$
MLP	162 ± 7	88 ± 7	85 ± 4	$\mathbf{79 \pm 4}$

As shown in table 11, Gaussian jitter improves the reported RUL RMSE for both downstream models, while FlowBatt synthetic data yields the lowest errors. The additional reduction is larger for RF and more modest for MLP. These results support the utility of synthetic data for the evaluated downstream models and augmentation settings. The RF RMSE of 140 cycles in table 3 is the benchmark result reported by Zhang et al. [52], whereas the 136-cycle baseline in tables 4 and 11 is our independent reproduction. This difference reflects separate implementations and runs, not a gain from augmentation.

Figure 9 shows synthetic SOH trajectories generated by FlowBatt on the `MATR-1` dataset for different classifier-free guidance strengths. Across all settings, the generated curves preserve the expected monotonic degradation behavior while covering a broad range of lifetimes and fading rates. Increasing the guidance strength changes the spread of the generated trajectories, with higher values producing a more concentrated set of samples, consistent with the reduction in Recall reported in table 4.

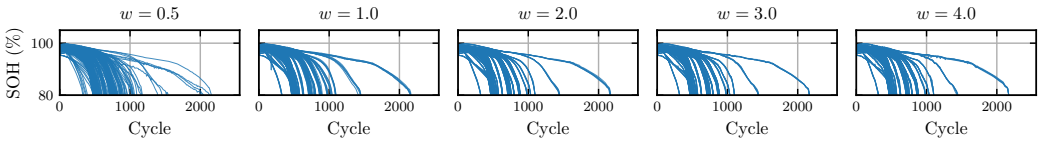


Figure 9: `MATR-1` synthetic data generated by FlowBatt with different guidance strengths.

A.5.2 Trajectory-wise SOH prediction

For prediction tasks we generate ten samples for each input capacity matrix. The capacity matrix is constructed from the first 100 or 20 cycles. The RMSE for an SOH sample j is computed as

$$\text{RMSE}_j = \sqrt{\frac{1}{n_j}\sum_{i=1}^{n_j}(\tilde{y}_i - y_i)^2} \tag{7}$$

where $\tilde{y}$ and y represent the predicted and the reference SOH in percentage, respectively, i denotes the cycle number and n_j is the cycle number at which the predicted SOH reaches the EOL. Further, we report the mean RMSE across all the test samples as the RMSE for the dataset. We pad both the reference and predicted sequences with the last value to ensure they have the same length. The mean and standard deviation of the RMSE are computed over ten generated samples per each input capacity matrix.

For downstream RUL evaluation, the scalar target is the first cycle at which SOH reaches or falls below 80% (90% for `MIX-20`). We report RMSE in cycles and MAPE (%).

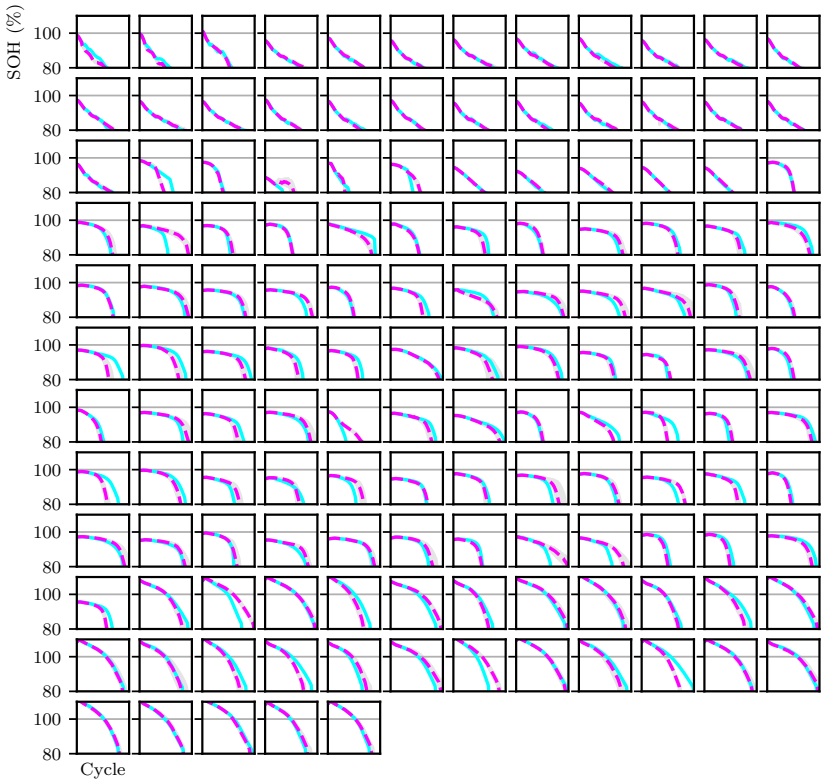


Figure 10: SOH predictions obtained from FlowBatt against reference for all the test samples of `MIX-100` dataset. The pink dashed line shows the prediction and the cyan solid line shows the reference. The gray area shows the prediction uncertainty computed from ten generated samples.

Figure 10 illustrates the predicted SOH versus the reference SOH for all test samples in the `MIX-100` dataset. The results demonstrate that FlowBatt effectively captures various degradation dynamics and accurately predicts SOH for the majority of test samples and highlights FlowBatt's ability to generalize across different battery chemistries and operational conditions present in the `MIX-100` dataset. This capability is essential for developing reliable battery health monitoring systems that can adapt to diverse usage patterns and environmental factors.

Figure 11 and figure 12 show the predicted SOH against the reference SOH for all test samples in the `MIX-100` dataset obtained from DiffBatt and Transformer, respectively. Compared to FlowBatt, DiffBatt shows a higher deviation from the reference for several test samples. The Transformer model shows a higher deviation from the reference for most of the test samples compared to both

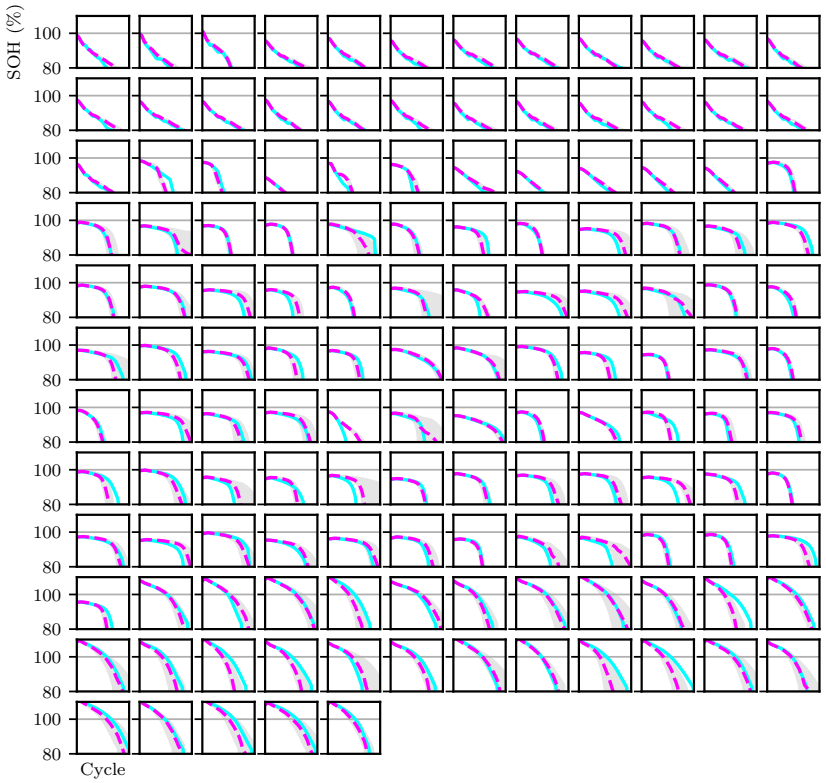


Figure 11: SOH predictions obtained from DiffBatt against reference for all the test samples of `MIX-100` dataset. The pink dashed line shows the prediction and the cyan solid line shows the reference. The gray area shows the prediction uncertainty computed from ten generated samples.

FlowBatt and DiffBatt. These results further highlight the advantages of flow matching over both diffusion-based training and supervised learning.

A.5.3 Cycle-wise SOH prediction

For completeness and comparability with prior BatteryML-style evaluation [51], we also consider a *cycle-wise* SOH prediction setting. In this setting, the model predicts the SOH at a predefined cycle index, and the error is reported as the RMSE across all cells in the dataset. Since BatteryML does not natively evaluate full degradation trajectories for SOH, we adapt both FlowBatt and BatLiNet [52] to the same cycle-wise setting in order to ensure a fair comparison with the BatteryML baselines.

In the main comparison, cycle-wise SOH is evaluated at cycle 100, which follows the standard BatteryML-style early-life setting [51]. To further assess whether the relative behavior of the models remains consistent at later stages of degradation, we additionally extend the cycle-wise analysis to cycle indices 200 and 300. This extended evaluation provides a complementary pointwise view of SOH prediction beyond the early-cycle regime.

Table 12: Cycle-wise SOH prediction results at cycle 100, reported as RMSE (%) across cells. Best results are shown in bold and second-best results are underlined. This table follows the BatteryML-style cycle-wise SOH protocol, where SOH is predicted at a predefined cycle index and RMSE is computed across cells. These results are therefore not directly comparable to the trajectory-wise SOH prediction results in table 1.

Method	`MATR-1`	`MATR-2`	`HUST`	`MIX-20`	`MIX-100`
BatLiNet	<u>0.33 ± 0.01</u>	**0.18 ± 0.01**	**0.05 ± 0.03**	**0.33 ± 0.01**	<u>0.27 ± 0.02</u>
RF (BatteryML)	0.88 ± 0.01	1.52 ± 0.03	1.18 ± 0.03	2.19 ± 0.05	2.04 ± 0.07
MLP (BatteryML)	1.01 ± 0.13	1.45 ± 0.30	1.58 ± 0.13	3.59 ± 1.31	3.72 ± 1.49
FlowBatt (ours)	**0.07 ± 0.01**	<u>0.23 ± 0.01</u>	<u>0.06 ± 0.01</u>	<u>0.89 ± 0.06</u>	**0.15 ± 0.01**

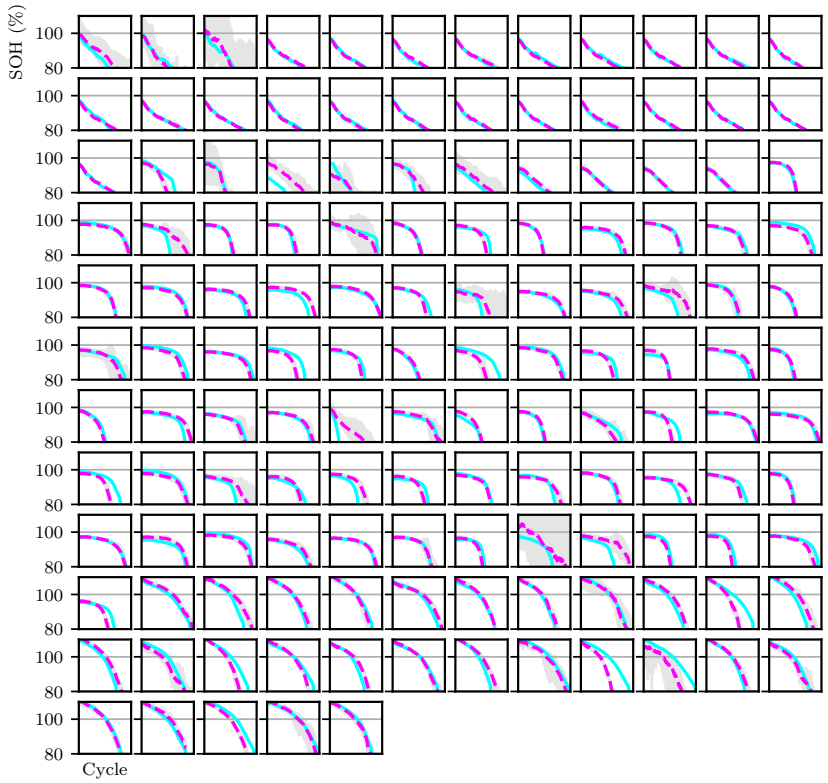


Figure 12: SOH predictions obtained from Transformer model against reference for all the test samples of MIX-100 dataset. The pink dashed line shows the prediction and the cyan solid line shows the reference. The gray area shows the variability across predictions from ten independently trained Transformer models.

The cycle-wise SOH results at cycle 100 are summarized in table 12. Under this pointwise evaluation setting, both FlowBatt and BatLiNet [52] substantially outperform the BatteryML baselines [51] on all datasets, indicating that trajectory-aware representations can still provide strong performance even when evaluated only at a single cycle target. FlowBatt achieves the best performance on MATR-1 with an RMSE of $0.07 \pm 0.01\%$ and on MIX-100 with $0.15 \pm 0.01\%$, while BatLiNet performs best on MATR-2, HUST, and MIX-20. These results show that FlowBatt remains highly competitive even under a protocol that does not explicitly exploit its full-trajectory modeling capability.

Table 13: Extended cycle-wise SOH prediction results at cycles 200 and 300, reported as RMSE (%) across cells.

Method	Cycle	MATR-1	MATR-2	HUST	MIX-20	MIX-100
FlowBatt (ours)	200	0.36 ± 0.03	0.73 ± 0.03	0.18 ± 0.03	0.93 ± 0.02	1.13 ± 0.08
	300	0.69 ± 0.03	0.94 ± 0.04	0.21 ± 0.03	1.12 ± 0.04	0.82 ± 0.03
RF (BatteryML)	200	1.25 ± 0.35	1.57 ± 0.05	1.17 ± 0.02	2.30 ± 0.00	3.54 ± 0.87
	300	1.48 ± 0.04	1.65 ± 0.03	1.18 ± 0.03	2.25 ± 0.02	1.64 ± 0.04
MLP (BatteryML)	200	1.31 ± 0.11	1.25 ± 0.35	1.52 ± 0.14	3.98 ± 0.88	1.64 ± 0.02
	300	1.97 ± 0.28	1.19 ± 0.18	1.51 ± 0.14	3.66 ± 0.55	3.47 ± 0.57
BatLiNet	200	0.16 ± 0.03	0.17 ± 0.08	0.85 ± 0.10	0.24 ± 0.01	0.34 ± 0.02
	300	0.45 ± 0.02	0.34 ± 0.07	1.14 ± 0.17	0.42 ± 0.03	0.71 ± 0.06

The extended cycle-wise results in table 13 show that the relative performance trends remain broadly consistent beyond the standard evaluation point at cycle 100. FlowBatt continues to clearly outperform the BatteryML baselines [51] at cycles 200 and 300 across most datasets, while BatLiNet [52] remains very strong at cycle 200 on several datasets. This extended analysis further supports the conclusion that

the gains of FlowBatt are not limited to a single predefined evaluation point, but remain competitive at later stages of battery degradation as well.

A.5.4 Uncertainty quantification

We evaluate empirical prediction intervals using coverage and mean width, as discussed by Sluijterman et al. [44]. For the FlowBatt sample-count evaluation in table 2, reference and generated trajectories are clipped from below at 80% SOH (90% for `MIX-20`). Let $y_{b,i}$ and $\hat{y}_{b,i}^{(n)}$ denote the reference and generated SOH, respectively, in percent, for test battery b at cycle i. The index $n = 1, \ldots, N$ identifies the N independent generative draws from a fixed checkpoint. The nominal 90% empirical prediction interval is

$$[L_{b,i}, U_{b,i}] = \left[q_{0.05}\left(\hat{y}_{b,i}^{(1)}, \ldots, \hat{y}_{b,i}^{(N)}\right), q_{0.95}\left(\hat{y}_{b,i}^{(1)}, \ldots, \hat{y}_{b,i}^{(N)}\right) \right], \tag{8}$$

where q_p denotes the empirical quantile at probability p, and $L_{b,i}$ and $U_{b,i}$ are the lower and upper bounds. Coverage is the percentage of reference values satisfying $L_{b,i} \leq y_{b,i} \leq U_{b,i}$. Mean width is the average of $U_{b,i} - L_{b,i}$, reported in SOH percentage points. Both metrics are averaged over all test battery-cycle pairs across the full evaluation horizon.

Comparison with probabilistic baselines. Table 14 compares FlowBatt and DiffBatt with Quantile RF and Quantile MLP under the fixed benchmark splits. Quantile RF estimates conditional quantiles following Meinshausen [35], while Quantile MLP is trained with the pinball loss. Both baselines estimate the 5th and 95th percentiles directly, without generative sampling. Quantile RF uses the baseline-subtracted discharge-capacity curve from the final early-life cycle, whereas Quantile MLP uses the complete capacity matrix.

Table 14: Comparison of nominal 90% SOH prediction intervals. Coverage is reported in percent and mean width in SOH percentage points. FlowBatt and DiffBatt use ten generated trajectories per test battery. Quantile RF and Quantile MLP results are averaged over ten training seeds.

Method	Metric	`MATR-1`	`MATR-2`	`HUST`	`MIX-20`	`MIX-100`
FlowBatt	SOH coverage	86.0%	67.0%	65.0%	83.6%	81.7%
	SOH width	0.30%	0.42%	1.34%	0.52%	0.64%
DiffBatt	SOH coverage	76.6%	64.8%	56.5%	79.8%	79.1%
	SOH width	0.25%	0.36%	2.52%	0.97%	1.37%
Quantile RF	SOH coverage	93.0%	96.0%	67.3%	91.9%	89.4%
	SOH width	3.06%	5.79%	6.10%	3.82%	4.07%
Quantile MLP	SOH coverage	91.4%	77.2%	69.3%	95.4%	94.4%
	SOH width	2.10%	2.64%	6.05%	4.64%	4.89%

FlowBatt produces substantially narrower prediction intervals than both quantile baselines across all evaluated datasets. It also achieves higher empirical coverage than DiffBatt throughout the benchmark. On `HUST`, `MIX-20`, and `MIX-100`, this improvement is accompanied by narrower intervals, indicating a favorable coverage–width trade-off relative to DiffBatt. The quantile baselines attain higher coverage than FlowBatt, but their intervals are substantially wider. Nevertheless, both generative models remain undercovered, particularly on challenging datasets `MATR-2` and `HUST`. Despite their wider intervals, undercoverage also persists for both quantile baselines on `HUST` and for Quantile MLP on `MATR-2`. Figures 13 and 14 show the quantile baselines' median predictions and prediction intervals for all `MIX-100` test batteries.

A.6 Broader impacts

Our work on probabilistic battery degradation prediction has positive societal implications by supporting the transition to sustainable energy technologies. Accurate SOH forecasting and uncertainty quantification can optimize battery usage, extend lifespans, and improve the reliability of electric vehicles and stationary storage systems, thereby reducing electronic waste and resource consumption. However, there are potential negative impacts to consider. The development and training of scaled up versions of generative models, such as diffusion transformers, require significant computational resources, contributing to energy consumption and carbon emissions. Furthermore, over-reliance on data-driven predictive models without proper physical validation could lead to misestimations of

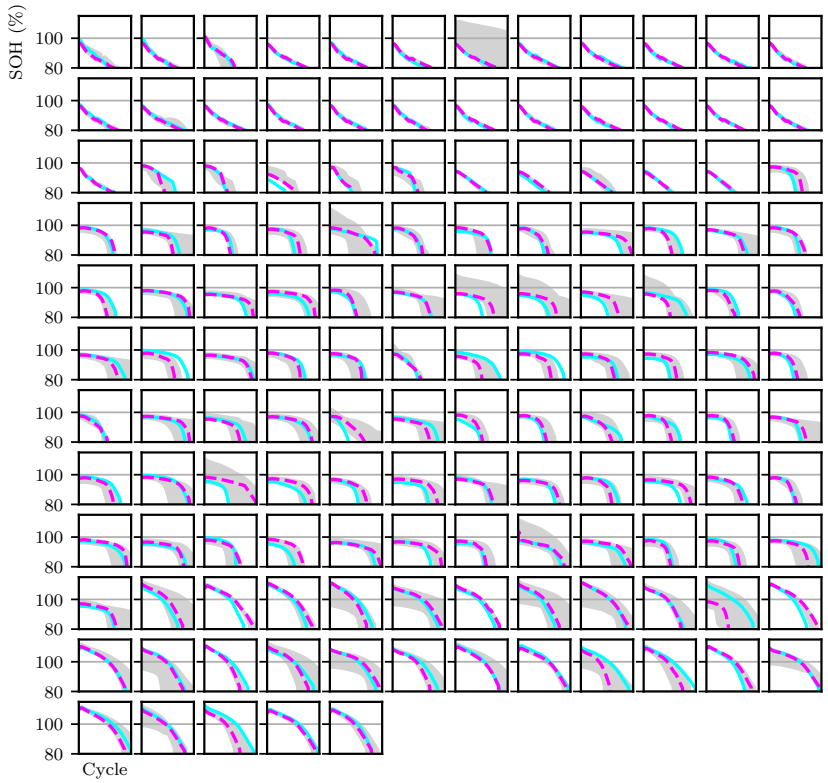


Figure 13: SOH predictions obtained from Quantile RF against the reference for all test samples of the `MIX-100` dataset. The pink dashed line shows the median prediction and the cyan solid line shows the reference. The gray area shows the nominal 90% prediction interval between the 5th and 95th percentiles.

battery health, potentially posing safety risks if applied blindly in critical infrastructure. We encourage future work to balance computational efficiency and tightly integrate physics-based safeguards.

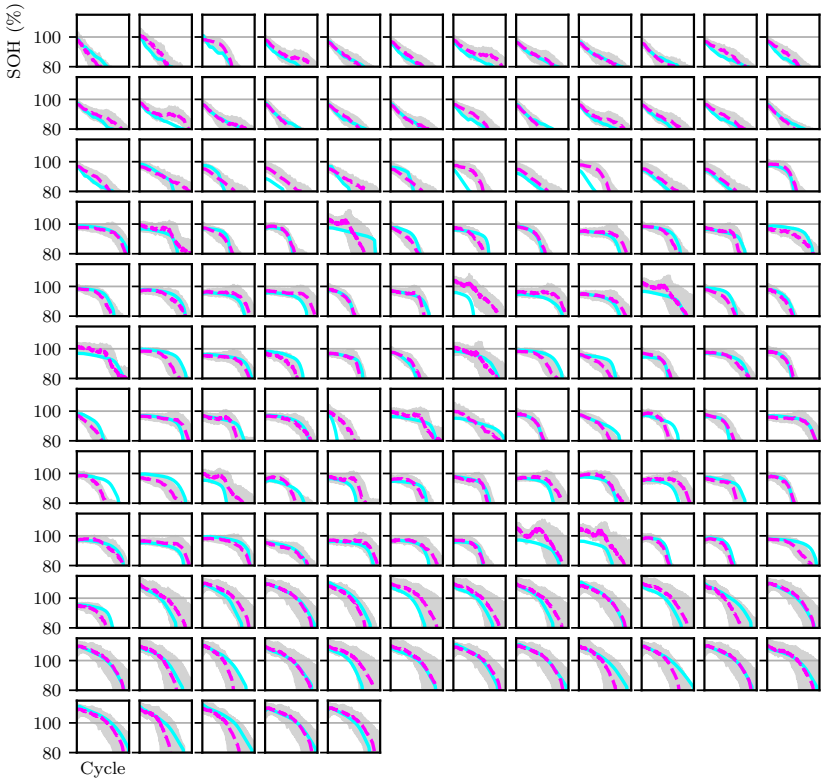


Figure 14: SOH predictions obtained from Quantile MLP against the reference for all test samples of the `MIX-100` dataset. The pink dashed line shows the median prediction and the cyan solid line shows the reference. The gray area shows the nominal 90% prediction interval between the 5th and 95th percentiles.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions of FlowBatt, namely trajectory-level SOH prediction, comparison against supervised and diffusion-based baselines, and downstream RUL evaluation. These claims are supported by the methodology and experimental results in the main paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses its limitations in section 3.2, noting that uncertainty calibration remains challenging for specific datasets (`MATR-2` and `HUST`). Additionally, section 4 highlights current constraints, such as the absence of additional operational variables (e.g., temperature) and the need for stronger integration with physics-based battery aging models, framing these as areas for future work.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present new theoretical results with theorem-proof structure. It summarizes established diffusion and flow-matching formulations as background and focuses on model design and empirical evaluation.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides comprehensive details required to reproduce the experimental results. The model architecture and preprocessing steps are described in section 2.2. Furthermore, extensive details including datasets, hyperparameter settings, algorithmic steps, and baseline configurations are explicitly documented in the appendix (see appendices A.1, A.2.2, A.2.4, A.4.1 and A.4.2).

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All benchmark datasets used in this study are publicly available. The source code, pre-trained models, and instructions for reproducibility are publicly available at `https://github.com/HamidrezaEiv/FlowBatt`.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides comprehensive details on the experimental setup. The benchmark datasets and data splits are described in the main text and appendix A.1. Model architecture, preprocessing steps, and specific hyperparameters are detailed in section 2.2, appendix A.2.2, and table 6. Furthermore, detailed configurations for the baselines are provided in table 8 and table 10.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main result tables (e.g., tables 1, 3 and 12) report the mean and standard deviation. As detailed in the main text (section 3) and the appendix (appendices A.4.1 and A.5.2), probabilistic models are evaluated using multiple generated trajectories to quantify predictive variability, while deterministic baselines are evaluated across multiple random initializations.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Information regarding the computational resources, including the hardware setup, software environment, and approximate training times, is detailed in appendix A.2.3.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The work uses publicly available battery datasets, does not involve human subjects, and is presented as a scientific modeling study intended to improve battery-health prediction and reproducibility.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: A dedicated discussion regarding both the positive societal impacts (e.g., sustainability, resource efficiency) and potential negative societal impacts (e.g., computational cost, over-reliance on ML models) is provided in appendix A.6.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk generative assets such as language models, image generators, or scraped datasets, and the proposed model is a domain-specific battery-degradation model.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All original creators of the utilized datasets and benchmark frameworks are properly cited in the main text and appendix A.1. A statement confirming adherence to the open-source licenses and terms of use of these assets has been explicitly added to appendix A.1.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The new assets developed in this work (the FlowBatt source code and pre-trained models) are available at https://github.com/HamidrezaEiv/FlowBatt. Comprehensive documentation, including requirements and usage instructions for training and evaluation, is provided within the included README file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research relies exclusively on technical, publicly available machine learning datasets for battery degradation. It does not involve any crowdsourcing or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: As the research does not involve human subjects, IRB approval or equivalent ethical reviews are not applicable to this work.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Large Language Models (LLMs) were not utilized as a component of the core methodology in this research. Any usage of LLMs was strictly limited to general language editing and formatting, which does not require a declaration according to the conference guidelines.