Applied Energy 377 (2025) 124604

Contents lists available at ScienceDirect

Applied Energy

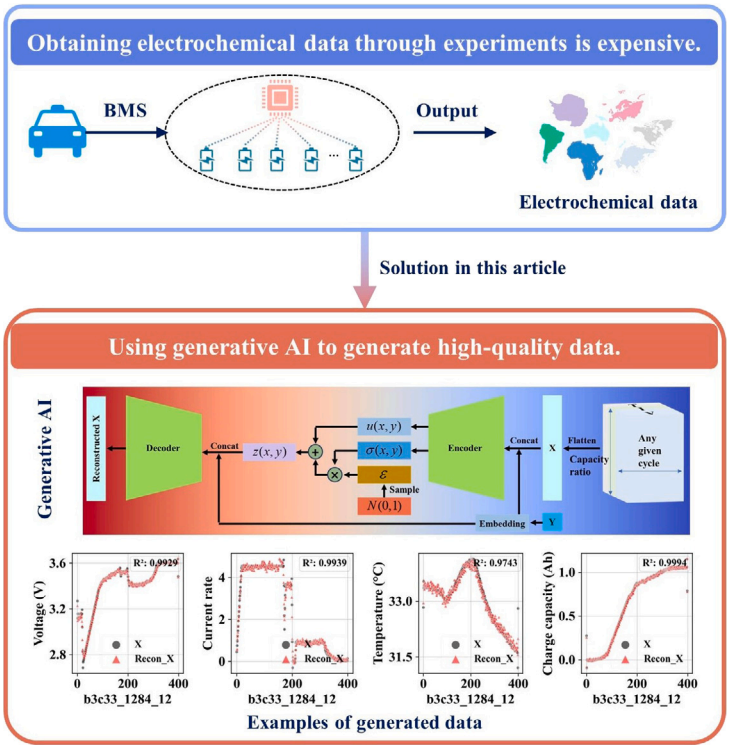
journal homepage: www.elsevier.com/locate/apenergy

Generating comprehensive lithium battery charging data with generative AI

Lidang Jiang [a], Changyan Hu [a], Sibei Ji [a], Hang Zhao [a], Junxiong Chen [b], Ge He [a,*]

[a] School of Chemical Engineering, Sichuan University, Chengdu, 610065, Sichuan, China
[b] School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu, 610023, Sichuan, China

GRAPHICAL ABSTRACT



ARTICLE INFO

Keywords:
Lithium batteries
Generative AI
Machine learning
Deep learning

ABSTRACT

In optimizing the performance and extending the lifespan of lithium batteries, accurate state prediction is crucial. Traditional regression and classification methods have achieved some success in predicting battery states. However, the effectiveness of these data-driven approaches largely depends on the availability and quality of public datasets. Additionally, generating electrochemical data primarily through battery experiments is a lengthy and costly process, making it extremely difficult to obtain high-quality data. This difficulty, combined with data incompleteness, significantly affects prediction accuracy. To address these challenges, this study introduces End of Life (EOL) and Equivalent Cycle Life (ECL) as supervised conditions for generative

* Corresponding author.
E-mail addresses: cjxzlm@my.swjtu.edu.cn (J. Chen), hege@scu.edu.cn (G. He).

https://doi.org/10.1016/j.apenergy.2024.124604
Received 16 May 2024; Received in revised form 2 September 2024; Accepted 26 September 2024

AI models. By integrating an embedding layer into the CVAE model, we developed the Refined Conditional Variational Autoencoder (RCVAE). Through preprocessing data into a quasi-video format, coupled with customized training and inference algorithms, the RCVAE generates the required charging data in real time based on the battery's ECL and EOL. This avoids storing irrelevant information, saving space and resources. RCVAE uses a lightweight architecture, enabling fast generation of necessary voltage, current, temperature, and charging capacity data. This approach provides users with a comprehensive electrochemical dataset, pioneering a new research domain for the artificial synthesis of lithium battery data. Furthermore, based on the detailed synthetic data, various battery state indicators can be calculated, offering new perspectives and possibilities for lithium battery performance prediction.

1. Introduction

Lithium-ion batteries (LIBs) have emerged as a pivotal component in energy conversion strategies due to their low cost, high energy density, and longevity, poised to replace traditional fossil-fueled automotive engines. This plays a critical role in global efforts to reduce greenhouse gas emissions and combat climate change [1]. As an environmentally friendly and efficient rechargeable energy option, lithium-ion batteries have found extensive applications in the realm of intelligent manufacturing, including computational engineering, logistics, and aerospace, among others. Their low pollution footprint, low self-discharge rates, broad operating temperature range, high energy and power density, along with long-term durability [2–4], have positioned them as a preferred energy solution. However, despite their vast potential across various application domains, the high replacement costs [5], challenges in accurately assessing battery states [6], and safety concerns [7–9] continue to be focal points of widespread user attention. Fortunately, Recurrent Neural Networks (RNN) [10], along with their derivatives [11–14], and Convolutional Neural Networks (CNN) [15], have been proven effective in precisely predicting battery states.

However, the efficacy of these data-driven methods largely hinges on the availability and quality of public datasets. With the growing interest in data-driven techniques and the pursuit of a deeper understanding of battery complex interactions, various datasets featuring different battery chemistries, quantities of tested batteries, and testing conditions have been developed. These datasets can generally be categorized into four types: cycle ageing data [16–19], drive cycle data [20], calendar ageing data [21], and specific-purpose datasets such as simulated satellite operation profile battery data [22,23]. Among these, cycle ageing datasets are the most common type in current practice, aiming to experimentally study the impact of various factors (e.g., charge current, discharge current, temperature, depth of discharge DOD) on the battery's capacity retention ability during cycling. These datasets typically include measurements of current, voltage, and temperature during the cycles, along with the capacity and internal resistance or impedance measurements for each cycle. The dataset released by NASA [24], which covers information on 34 lithium-ion 18 650 batteries with a nominal capacity of 2 Ah, marked the advent of the first publicly available battery dataset, significantly impacting the battery research field. The batteries underwent cycling tests at different environmental temperatures (4 °C, 24 °C, 43 °C) using a standard constant current–constant voltage (CC–CV) charging protocol, and various discharge regimes were implemented. Another vital resource is the collaboration between the Toyota Research Institute (TRI), MIT, and Stanford University, which launched two rich and user-friendly high-throughput cycling test datasets involving 357 (124 + 233) commercial Lithium Iron Phosphate (LFP)/Graphite batteries (APR18650M1 A) produced by A123 Systems, with a rated capacity of 1.1 Ah. The dataset of 124 batteries [25] aims to study the effects of fast charging protocols on battery ageing, while the dataset of 233 batteries [26] is similar to the former and is not repeated here. Particularly, the dataset of 124 batteries, due to its high-quality data, well-organized format, and accessibility, made a significant impact on the entire field upon its release in 2019.

Creating these battery datasets requires significant time and financial resources, yet the challenge of acquiring high-quality electrochemical data remains unresolved. Introducing synthetic data can enhance existing datasets, thereby improving model training performance. A popular method currently involves interpolating experimental data [27]. Although this strategy is straightforward, the generated data are limited in quality and diversity. On the other hand, artificial intelligence generative models offer new possibilities for data generation. Research on generative AI has mainly focused on natural language processing (NLP) and computer vision (CV) fields. In NLP, the introduction of the Transformer model [28] in 2017 further propelled advancements in this domain. Transformers process input sequences through an encoder–decoder architecture, generating hidden representations and output sequences. Its key innovation — multi-head attention mechanism — allows the model to allocate different attention weights based on the relevance between words, significantly enhancing the model's ability to handle long-term dependencies. Additionally, the architectural characteristics of Transformers enable high parallel processing capabilities, achieving exceptional performance improvements across various NLP tasks. In the CV domain, the introduction of Generative Adversarial Networks (GANs) [29] in 2014 marked a significant turning point. However, the adversarial training mechanism between the generator and discriminator makes finding the optimal balance highly challenging. Subsequent research efforts have focused on improving model generalization capabilities and the quality of generated data through structural innovations (e.g., StyleGAN [30]) and loss functions (e.g., WGAN [31]). Inspired by the Transformer model, researchers developed the Vision Transformer (ViT) [32], enabling efficient processing of image-related downstream tasks. More recently, diffusion generative models [33] have emerged as a cutting-edge technology for generating high-quality images. They generate data by gradually introducing and reversing multilevel noise perturbations, a process of progressive degradation and repair that endows diffusion models with powerful image generation capabilities.

With the rapid advancement of computational hardware such as GPUs, distributed training, and cloud computing technologies, large-scale models based on fundamental units like Transformer, ViT, and diffusion models, including BERT [34], GPT [35], and stable diffusion [36], have emerged like mushrooms after the rain. However, due to the Transformer model's demand for GPU memory being directly proportional to the square of the sequence length, generative models based on Transformer require relatively high hardware specifications. Meanwhile, although diffusion models adopt a progressive strategy during generation, their efficiency is relatively low. Variational Autoencoders (VAE) [37] attempt to generate data close to the original input by mapping data to a probability distribution and learning its reconstruction. On this basis, the introduction of Conditional Variational Autoencoders (CVAE) [38] has made it possible to generate targeted and supervised images or sequence samples. Compared to the training challenges of GANs and the high hardware requirements of Transformer-based models, CVAEs, due to their effectiveness in precisely modeling data distributions and ease of training, have been widely used across various application scenarios. Additionally, embedding layers [39] play a central role in handling categorical data within neural networks, especially indispensable in the process of encoding vocabulary in NLP tasks. Embedding layers can map various types of

Nomenclature			
EOL	End of Life	MAPE	Mean Absolute Percentage Error
ECL	Equivalent Cycle Life	MAE	Mean Absolute Error
RCVAE	Refined Conditional Variational Autoencoder	RMSE	Root Mean Square Error
LIBs	Lithium-ion batteries	BMS	Battery Management System
RNN	Recurrent Neural Networks	RUL	Remaining Useful Lives
CNN	Convolutional Neural Networks	MLP	Multi-layer Fully Connected Neural Network
NLP	Natural Language Processing	GANs	Generative Adversarial Networks
CV	Computer Vision	ViT	Vision Transformer
VAE	Variational Autoencoders	CVAE	Conditional Variational Autoencoders

categorical data (such as words, characters, or different types of labels) to dense vectors (i.e., embedding vectors) in high-dimensional space, effectively revealing complex relationships between categories, such as semantic similarity.

In this study, we use End of Life (EOL) and Equivalent Cycle Life (ECL) as supervisory conditions for a generative AI model and develop the Refined Conditional Variational Autoencoder (RCVAE) by incorporating an embedding layer into the CVAE framework. Leveraging the authoritative MIT dataset, we preprocess the data into a quasi-video format to achieve comprehensive integration of electrochemical data such as voltage, current, temperature, and charging capacity, which is then processed by the RCVAE model. Supported by customized training and inference algorithms, our model can generate detailed and high-quality electrochemical charging data under supervised conditions. This approach provides a complete electrochemical dataset, paving new ways for the artificial synthesis of lithium battery data. Moreover, the detailed synthetic data can be utilized to calculate various battery state indicators: the generated charging capacity, for instance, can be directly normalized to derive the State of Charge (SOC), while other indicators like State of Health (SOH) can be similarly determined from voltage and capacity metadata. By generating this fundamental data, we simplify the calculation of these secondary indicators, offering new perspectives and possibilities for lithium battery performance prediction.

The main contributions of this paper are summarized as follows:

1. **Introduction of RCVAE:** By integrating an embedding layer into the CVAE, we developed the RCVAE, marking a significant refinement of the CVAE model. This model is specifically optimized for generating electrochemical data, enabling it to more accurately capture and reflect key parameters and changes in battery performance.
2. **Introduction of supervised conditions:** We propose an innovative approach that utilizes the battery's expected lifespan (EOL) and Equivalent Cycle Life (ECL) as conditions for the generative artificial intelligence (AI) model. The successful implementation of this method validates its potential in battery performance prediction and health management, offering new insights for the development of future battery management systems.
3. **Generation of datasets:** The dataset is generated in real-time by RCVAE based on the battery's ECL and EOL to provide the required charging data. Compared to traditional open-source datasets, RCVAE avoids generating unnecessary data, saving disk space, and features a simple and easy-to-manage structure. The Encoder and Decoder in RCVAE use a lightweight MLP architecture, which has low hardware requirements, fast inference speed, and can quickly generate the needed voltage, current, temperature, and charging capacity data.
4. **Proposed training and inference algorithm:** This paper proposes a training and inference algorithm tailored to the generative tasks in the battery field.

The structure of the paper : Section 2 provides a detailed introduction to the MIT dataset, including its data sources, types, importance, and application methods. Section 3 thoroughly explains the application process of the RCVAE model in real-world scenarios, specifically describing the data preprocessing steps and the precise design of the RCVAE architecture, including the generation process of model input data, the components of the RCVAE model, and their interactive mechanisms. Section 4 presents our experimental results and provides an in-depth analysis. This section begins with an evaluation of the RCVAE model's performance across various tasks, followed by an analysis of the role of each component in the model and an extensive discussion and demonstration of the model's interpretability. Finally, the paper concludes with a summary of its contents.

2. Source of the data

The dataset used in this study was created by Severson et al. in 2019 [25], known as the MIT dataset, which has become one of the most authoritative data sources in this field. This battery dataset includes 124 batteries, achieving a diversity of battery lifespans by controlling different charging and discharging current sizes. As shown in Fig. 1(a), the selected batteries are lithium iron phosphate/graphite batteries produced by A123 Systems, with a nominal capacity of 1.1 Ah. The cycle number at which the battery's discharge capacity falls to 80% of its nominal capacity is defined as the EOL for the battery. This design effectively reflects the variability in battery lifespan observed in the real world.

The dataset comprises three batches of batteries. Fig. 1(b) displays the variation in charging voltage for the first battery (identified as "b3c0") in the third batch. Battery "b3c0" is chosen as an example because it exhibits fewer outliers than others, facilitating graphical presentation. As depicted in Fig. 1(b), the battery voltage changes with increasing cycle numbers, reflecting the degradation of battery performance. Even for the same battery, the voltage data across different

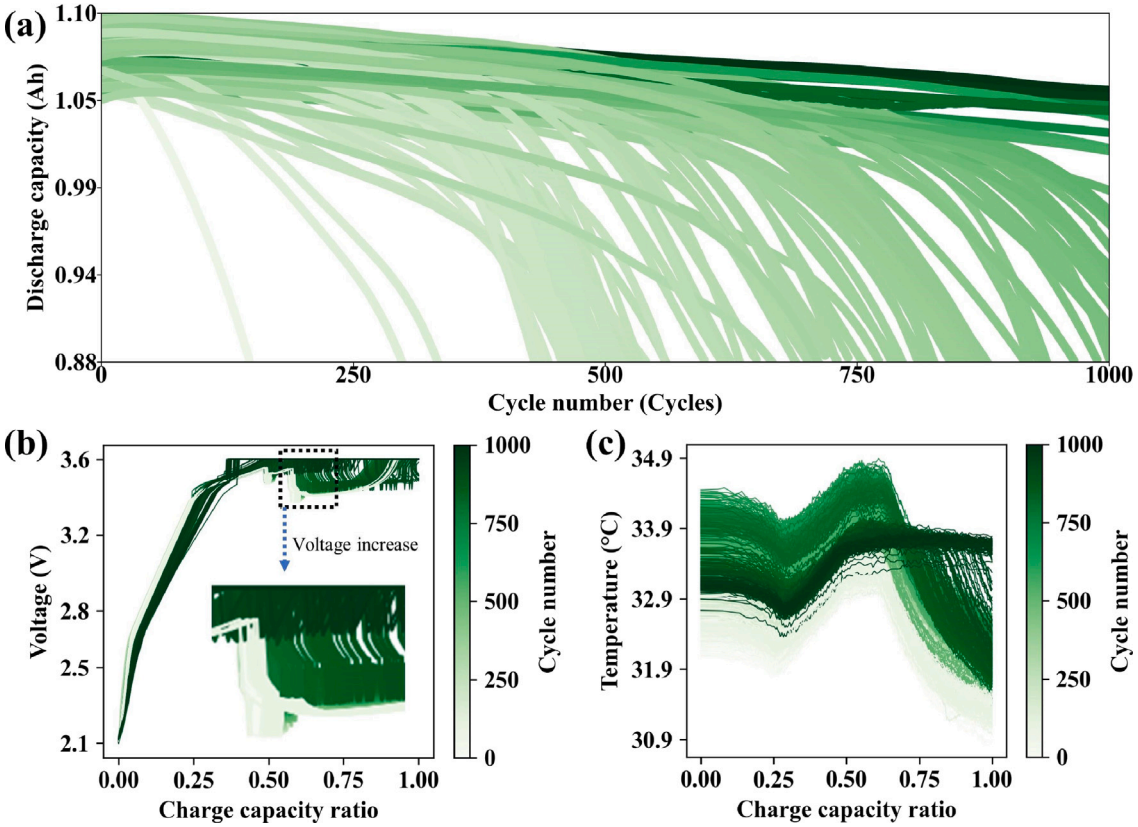


Fig. 1. Conducts a comprehensive analysis of lithium-ion battery performance: (a) based on the MIT dataset, showing the trend of lithium-ion battery discharge capacity decay over cycles; (b) displaying the variation in voltage of the "b3c0" battery across different charging cycles, with the voltage decline areas highlighted by black square markers, emphasizing the voltage decay characteristics during charging; (c) describing the temperature change trend of the "b3c0" battery during charging, reflecting the thermal management status at different charging stages.

cycles vary with the number of cycles, where the cycle count to some extent indicates the degree of battery degradation. The variance in voltage data across different cycles suggests that voltage data from any cycle can serve as unique features mapping to EOL. Similarly, in Fig. 1(c), a comparable conclusion can be drawn: temperature data across different cycles for the same battery show variability, indicating that temperature data from any cycle can also serve as distinct features mapping to EOL.

3. Methods

After a detailed introduction of the dataset, this section will focus on elucidating the complete workflow of RCVAE in generating electrochemical data. As illustrated in Fig. 2, the workflow begins with the Battery Management System (BMS), which is responsible for collecting real-time operational data from the batteries. These raw data are then transformed into a quasi-video format through a series of preprocessing steps to enhance data processability. The preprocessed data are subsequently fed into the RCVAE model for training and to perform prediction tasks.

3.1. Data preprocessing

For data preprocessing, this study draws on the methodology described in the literature [40]. After organizing the samples into a quasi-video format, each battery generated n samples, resulting in a total of $124n$ samples. To reduce randomness in experimental outcomes, this study shuffles the data at the sample level rather than at the battery level, meaning that samples from the same battery will not appear consecutively in the dataset. Following the proportion used in other studies [40–42], the first $94n$ samples were allocated to the training set, with the remaining samples assigned to the test set. Subsequently, data cleaning and normalization were performed.

Data cleaning refers to the removal of outliers to facilitate the model's ability to learn data patterns. Here, we calculate the mean and standard deviation of the training set and use them to compute the Z-score of each point in the entire dataset. If the Z-score is too large, the current point is considered an outlier and should be replaced by the mean of its neighboring points.

$$|z_i| = \left| \frac{x_i - \mu}{\sigma} \right| \geq \tau \tag{1}$$

where x_i is the value of the data point, μ is the mean of the data, σ is the standard deviation of the data, and τ is the threshold for outlier detection.

When normalizing the data, as with data cleaning, to avoid using global statistics from an unknown dataset (test set), we only use data from the training set to compute the global statistics, specifically the min and max values. Based on these values, the features of both the training and test sets are scaled to the range of -1 to 1 to facilitate learning patterns. Whether in data cleaning or normalization, the use of only the global statistics of the training set in this study indicates that no "data leakage" occurred, making the results more credible.

"Data leakage" refers to a situation in machine learning where the model unintentionally uses information during training that should not be available in real scenarios. This leads to the model performing very well during training but poorly when encountering new data because it has learned some "answers" it should not have known.

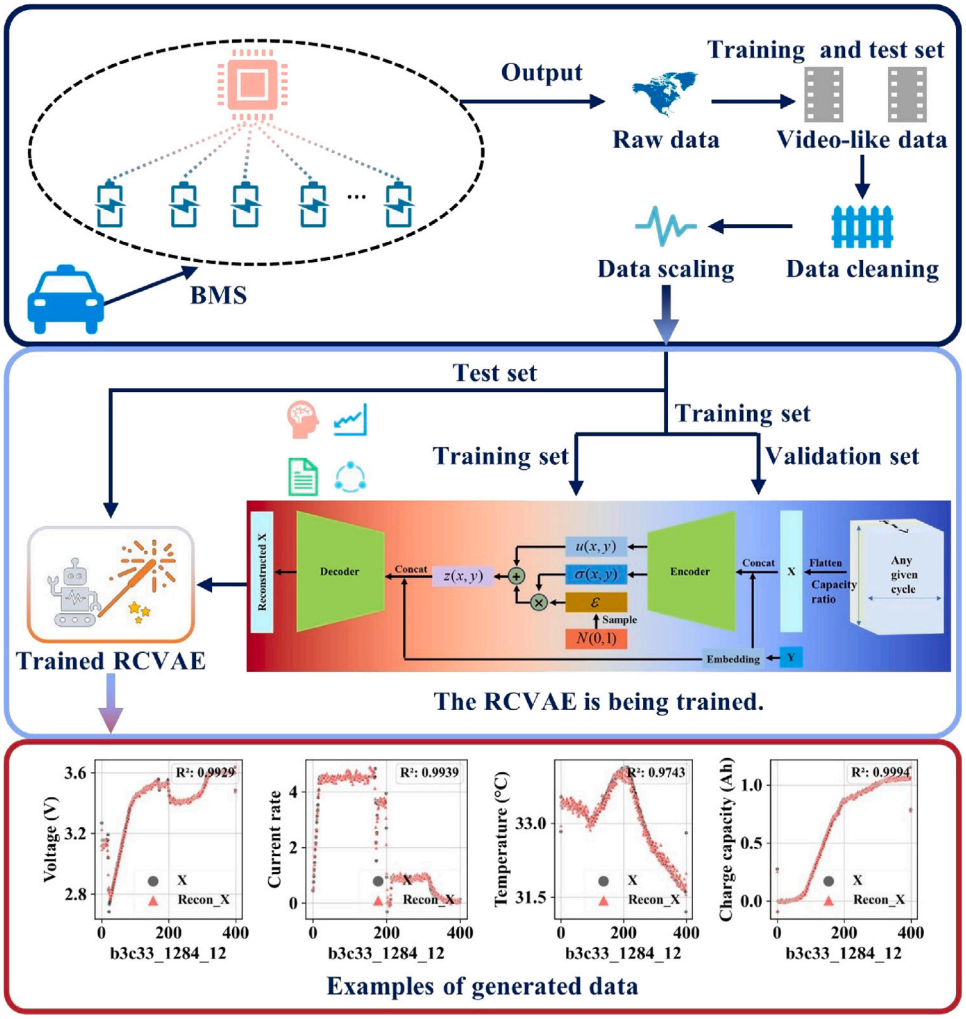


Fig. 2. Illustrates the technical roadmap for generating high-quality electrochemical data with RCVAE.

Then, the training set was further divided into a new training set and a validation set at a ratio of 4:1. The reason for creating a new validation set is to monitor the model's performance on a controlled data distribution during training, specifically by observing the validation loss. This helps determine whether the model has been sufficiently trained.

Unlike previous studies that used data from the first 10 cycles, this study uses data from any cycle as samples, as shown in Fig. 3. After data preprocessing, the data is transformed into a five-dimensional array that can be processed by 3D Convolutional Neural Networks (3D CNNs). The first dimension represents the number of samples, the second dimension represents the number of channels, which indicates three types of data (voltage, current, temperature), and the third dimension represents depth, which is added due to the inclusion of charging capacity data. The number of data points for a single type of data forms another dimension, which is reshaped into two dimensions and placed in the height and width directions, forming a five-dimensional electrochemical feature array, as shown on the left side of Fig. 3.

3.2. RCVAE (Refined Conditional Variational AutoEncoders)

3.2.1. Generating embedding vectors through labels

Before the data is input into RCVAE, all dimensions other than the number of samples are flattened into sequences. Initially, the data's labels are processed through an embedding layer. As demonstrated in Fig. 1, data with different EOLs exhibit distinct characteristics, and similarly, data with different Remaining Useful Lives (RUL) also show their unique differences. Furthermore, if two batteries have similar lifespans, their RUL and electrochemical data will also display similarities. The calculation of RUL follows the formula (2), where EOL represents the expected lifespan of the battery, and ECL refers to the number of cycles the battery has already completed. RUL is the difference between EOL and ECL:

$$\mathrm{RUL} = \mathrm{EOL} - \mathrm{ECL} \tag{2}$$

Thus, through EOL and RUL, we can delineate the characteristics of electrochemical data; in this study, "EOL_ECL" is used as a condition in RCVAE, serving as the label for this research.

Different "EOL_ECL" combinations form a condition set $C = [c_1, c_2, \ldots, c_N]$, where each c_n represents the nth unique category label y in the sequence. Before C enters the embedding layer, since "EOL_ECL" exists in string format, it needs to be converted into integer indices for numerical computation. The mapping from C to integer indices is achieved through a lookup table, which can be considered a function mapping category labels to unique integer indices. This mapping function can be represented as $f : C \to \{0, 1, 2, \ldots, N-1\}$, where N is the total number of categories. For any category $c \in C$, its integer index can be represented as:

$$i = f(c) \tag{3}$$

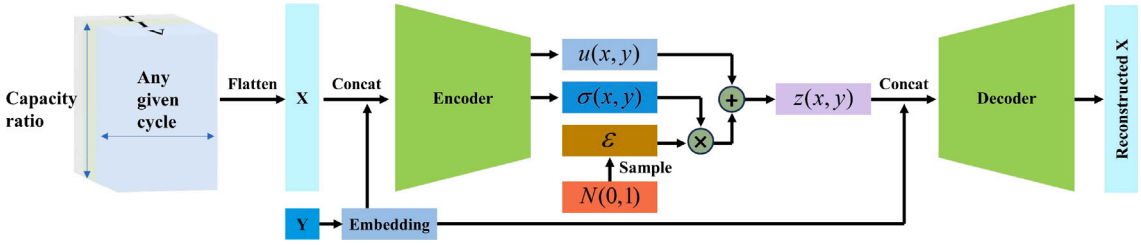


Fig. 3. Describes the quasi-video data after preprocessing and the overall architecture of RCVAE.

Here, c is the specific category label. f is the mapping function from category labels to integer indices. i is the integer index corresponding to c. Therefore, the integer index sequence converted through the mapping function f is represented as:

$$I = \left[f\left(c_1\right), f\left(c_2\right), \ldots, f\left(c_N\right)\right] \tag{4}$$

Then, the integer indices $I = \left[i_1, i_2, \ldots, i_N\right]$ are passed into the embedding layer, which is also a lookup operation, mapping discrete category labels into a continuous vector space. Suppose E is the embedding matrix, i is the integer index of the category label, v is the vector mapped through the embedding layer, then the embedding operation can be represented as:

$$v = E[i] \tag{5}$$

Here, the size of E is $N \times D$: N is the total number of categories, i.e., the number of elements in the C set. D is the dimension of the embedding vector (embedding_dim). $E[i]$ represents the row in the embedding matrix E indexed by i, i.e., the embedding vector corresponding to index i. Thus:

$$V = E\left[i_1, i_2, \ldots, i_N\right] = \left[E\left[i_1\right], E\left[i_2\right], \ldots, E\left[i_N\right]\right] \tag{6}$$

3.2.2. Forward propagation of RCVAE

The flow of preprocessed data within RCVAE is divided into several steps. Input: Initially, the embedding vectors V (representing processed labels or other categorical information) and the feature data X (the actual features of the input data) are concatenated to form a combined feature vector. This vector is then jointly input into the encoder. The dimensions of the embedding vector V and the feature data X are d_V and d_X, respectively. The concatenation process can be represented as:

$$X_{\text{combined}} = \text{concat}(V, X) \tag{7}$$

Here, concat $(\cdot,)$*denotes the concatenation operation, aligning* V and X in sequence to form a new combined feature vector with dimensions $d_V + d_X$. This combined feature vector X_{combined} subsequently serves as the input for the encoder. The purpose of this step is to merge the label or category information related to the input data (represented by the embedding vector V) with the actual feature data X, to consider both aspects of information in the ensuing encoding process. Thus, integrating the embedding vector and feature data into a unified input lays the groundwork for subsequent encoder processing.

The encoder learns the distribution parameters (namely, mean μ and variance σ^2) of the latent representation of the input data x and associated condition y. This can be expressed as the mean $\mu(x, y)$ and variance $\sigma^2(x, y)$:

$$\mu(x, y) = f_\mu\left(X_{\text{combined}}\right) \tag{8}$$

$$\log\left(\sigma^2(x, y)\right) = f_\sigma\left(X_{\text{combined}}\right) \tag{9}$$

Functions f_μ and f_σ are implemented by multi-layer fully connected neural networks, designed to extract the mean and the logarithm of the variance from the input data. Here, the number of layers in the multi-layer fully connected neural network (MLP) is set as a hyperparameter and is not fixed. In the MLP, each layer applies a linear transformation and a nonlinear activation function to the output of the previous layer. Let L be the total number of layers in the network; for each layer $l = 1, 2, \ldots, L$, linear transformations and nonlinear activations can be represented as follows:

$$h_l = W_l \cdot a_{l-1} + b_l \tag{10}$$

$$a_l = f\left(h_l\right) \tag{11}$$

W_l and b_l are the weight and bias of the l th layer, respectively, and f is the nonlinear activation function. For the first layer, $a_0 = X_{\text{combined}}$, meaning the combined feature vector serves as the input to the network. Ultimately, the mappings for the mean $\mu(x, y)$ and the logarithm of the variance $\log\left(\sigma^2(x, y)\right)$ can be represented by the output of the last layer. Assuming that the calculations for the mean and the logarithm of the variance are derived from the network's last two output layers respectively, then the calculation for the mean $\mu(x, y)$ can be represented as: $\mu(x, y) = W_\mu \cdot a_L + b_\mu$. The calculation for the logarithm of the variance $\log\left(\sigma^2(x, y)\right)$ can be represented as: $\log\left(\sigma^2(x, y)\right) = W_\sigma \cdot a_L + b_\sigma$. Here W_μ and b_μ, as well as W_σ and b_σ, are the weights and biases of the last two output layers specifically used for calculating the mean and the logarithm of the variance, with a_L being the activation output of the last hidden layer. By this means, setting an MLP as a network with any number of layers allows for flexible adaptation to different complexities in model design requirements.

Next is the sampling of the latent representation: After obtaining the mean $\mu(x, y)$ and variance $\sigma^2(x, y)$, the latent representation $z(x, y)$ can be sampled from the corresponding Gaussian distribution using the reparameterization trick:

$$z(x, y) = \mu(x, y) + \epsilon \cdot \sqrt{\exp\left(\log\left(\sigma^2(x, y)\right)\right)} = \mu(x, y) + \sigma(x, y) \cdot \epsilon \tag{12}$$

where ϵ is noise sampled from the standard normal distribution $\mathcal{N}(0, I)$. This method allows the encoder not only to output the mean and variance of the latent representation of the input data under specific conditions but also to generate diverse outputs by introducing random noise ϵ, a key characteristic of variational autoencoder generative models. In summary, the encoder operation learns to map the combination of input features and condition information to the distribution parameters of the latent space, generating the latent representation $z(x, y)$ through these parameters plus random noise. This process provides the foundation for the subsequent decoder to reconstruct the input or generate new data.

Preparation of Decoder Input: Before the latent variable $z(x, y)$ is input into the decoder, it needs to be concatenated with the corresponding embedding vector $V(y)$ to form the decoder's input vector. This can be represented by the following equation:

$$X_{\text{decoder}} = \text{concat}(z(x, y), V(y)) \tag{13}$$

By concatenating the latent variable $z(x, y)$ and the embedding vector $V(y)$ in sequence, a new vector X_{decoder} is formed, which is

then used as the input to the decoder. In this way, the decoder not only receives the latent representation mapped from the input data x and condition y but also integrates additional information about the condition y, enabling the decoder to generate target samples matching the input data under supervision while considering the condition information y. This combination of latent representation and condition information is one of the key features distinguishing CVAE from standard VAE.

Decoder Operation: The concatenated vector X_{decoder} (a fusion of latent representation and label information) is inputted, and the target samples matching the input data are generated. It can be represented as:

$$X_{\text{reconstructed}} = g\left(X_{\text{decoder}}\right) \tag{14}$$

Here: $g(\cdot)$ is the function of the decoder, implemented through an MLP, aiming to learn how to reconstruct data from the latent space. This function is similar to that of the encoder and will not be re-iterated here. $X_{\text{reconstructed}}$ is the reconstructed data output by the decoder, intended to resemble the original input data x as closely as possible, while considering the condition information y. The learning objective of the decoder is to maximize the data likelihood given the latent representation and condition information, i.e., to maximize $p(x \mid z(x, y), V(y))$. By optimizing this objective, the decoder can generate target samples that match the original input data and consider the condition information. Specifically for the application of generating electrochemical samples, this means the decoder has learned how to generate new electrochemical samples that meet these conditions based on given electrochemical conditions and corresponding feature data. The forward propagation algorithm of RCVAE is shown as Algorithm 1.

Algorithm 1 Forward Propagation Algorithm of RCVAE

Input: Feature data x and label y.
Output: Generated electrochemical sample by RCVAE.
Step 1: Embed label y into $V(y)$ by Eq. (6).
Step 2: Concat x and $V(y)$ into one vector by Eq. (7).
Step 3: Encode to latent space, get mean and variance by Eqs(8)-(9).
Step 4: Sample latent variable $z(x, y)$ using mean and variance by Eq. (12).
Step 5: Concat $z(x, y)$ and $V(y)$ for decoder by Eq. (13).
Step 6: Decoder generates sample by Eq. (14).
Return: Generated electrochemical sample by RCVAE.

3.2.3. Training and evaluation of algorithm

It should be noted that the embedding space for labels is determined based on the labels in the training set. However, the labels in the test set may not directly correspond to the embedding space of the training set. For such test samples, their labels undergo preliminary preprocessing through a similarity strategy. Specifically, the similarity of the test set labels is calculated using the difference formula (15). By iterating through each label in the training set and calculating the distance to the test label, the label with the smallest distance is selected as the new test label. Subsequent processing steps are similar to those described earlier and are not detailed here.

$$distance = weight \cdot \left|\text{EOL}_1 - \text{EOL}_2\right| + (1 - weight) \cdot \left|\text{ECL}_1 - \text{ECL}_2\right| \tag{15}$$

As shown in Algorithm 2, the training and inference process of RCVAE begins with the hyperparameter tuning stage, where the best hyperparameters are selected using the Bayesian optimization algorithm based on the performance on the validation set. The final hyperparameters are listed in Table 1.

Compared to the unstable method of setting hyperparameters based on personal subjective experience, all hyperparameters in this study were determined through automatic selection using the Bayesian optimization algorithm. This preparation ensures that the model can be well-trained. During the model training phase, to thoroughly learn the sample data, the validation set is merged with the training set to form a new training set on which the model is trained. This allows the model to learn as much data pattern as possible without causing data leakage. The total loss during the training process is obtained by combining the Mean Squared Error (MSE (16)) and the Kullback–Leibler Divergence (KLD (17)), as shown in Eq. (18).

Table 1
Hyperparameters for the model.

Parameter	Value
Batch size	64
Learning rate	3.1138e−05
Beta1	0.16012168351168404
T_max	78
Weight decay	0.023010888504871155
Patience	65
Step size	1
Gamma	0.09501282296926236
Warmup epochs	89
Leaky slope	0.15627772069234497
Gradient clip	741.0420754928822
Alpha RMSprop	0.6132744782992872
Final layer size	642
Total layers	16
Embedding dimension	473
Life weight	0.5572459533596918
Leaky slope (Encoder)	0.8714833687494885
Leaky slope (Decoder)	0.20404608524012796
Latent size	50
Scheduler type	ReduceLROnPlateau
Optimizer type	RAdam
Linear mode	fan_in
Apply weights Init	True
Use batch norm	True

$$\text{MSE} = \frac{1}{K}\sum_{i=1}^{K}\left(x_i - \hat{x}_i\right)^2 \tag{16}$$

Here, K is the total number of samples in the batch, x_i is the i th element of the original input data, and $\hat{x}_i$ is the i th element of the reconstructed data.

$$\text{KLD} = -\frac{1}{2}\sum_{j=1}^{J}\left(1 + \log\left(\sigma_j^2\right) - \mu_j^2 - \sigma_j^2\right) \tag{17}$$

where J is the dimension of the latent space, μ_j and σ_j^2 are the mean and variance of the j th latent variable, respectively.

$$\text{Loss} = \text{MSE} + \frac{\text{KLD}}{K} \tag{18}$$

In this section, normalizing by K ensures that the loss does not change with the size of the sample set, thereby making the training results comparable across different sample sizes. To ensure the model is thoroughly trained while maintaining efficiency, 1000 training epochs are set, along with an early stopping mechanism, where the early stopping counter is set to 100. This means that if the loss (Eq. (19)) on the validation set does not decrease for 100 consecutive training epochs, the training is terminated. If the model's performance does not improve over time, training is stopped promptly, and the best model is saved, which significantly speeds up the training process. During the model performance testing phase, the model's predictive performance is evaluated for the first time using the test set data. To comprehensively evaluate the model's predictive ability, this study uses MAE and RMSE as evaluation metrics, with their respective mathematical expressions defined in Eqs. (19) and (20).

$$\text{MAE} = \frac{1}{K}\sum_{i=1}^{K}\left|x_i - \hat{x}_i\right| \tag{19}$$

$$\text{RMSE} = \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(x_i - \hat{x}_i\right)^2} \tag{20}$$

where K is the total number of samples. x_i is the ith element of the original data. $\hat{x}_i$ is the ith element of the reconstructed data.

Algorithm 2 Training and Evaluation of Algorithm

Input: The training set, validation set, and test set.
Output: Trained Model, Test Loss
Step 1: Optimize hyperparameters using Bayesian optimization.
Step 2: Concatenate the training and validation sets to form a new training set.
Step 3: Set the hyperparameters found to be optimal in the previous steps.
Do
Counter = 0.
Repeat
Outer Loop for multiple epochs:
Do
Forward Propagation:
Step 4: Compute training loss by Eq. (18) and validation loss by Eq. (19).
Backward Propagation:
Step 5: Calculate gradients based on the training loss.
Step 6: Update network parameters.
Step 7: If validation loss decreases, save the model and Counter = 0.
Otherwise, Counter = Counter + 1.
Step 8: If Counter exceeds a predefined threshold, trigger early stopping.
End
Until reaching a predefined number of epochs or early stopping is triggered.
End
Step 9: Save the model, Load the model.
Step 10: Forward Propagate on the test set to compute the test loss by Eqs. (19)–(20).
return Trained Model, Test Loss

4. Results and discussion

4.1. Generative results of RCVAE

This study aims to explore the impact of dataset size on the performance of the RCVAE model in generating electrochemical data. By analyzing the model's behavior with different amounts of early-cycle data, we observed some interesting trends. Due to the values of certain data types approaching zero in some cases, the Mean Absolute Percentage Error (MAPE) exhibited significant instability, suggesting that MAPE might not be an appropriate evaluation metric. As shown in Table 2, for voltage prediction, the Mean Absolute Error (MAE) fluctuated between 0.021 and 0.023 V, while the Root Mean Square Error (RMSE) ranged from 0.043 to 0.047 V. In predicting the current rate, the MAE values ranged between 0.268 and 0.286, and the RMSE values varied from 0.66 to 0.741, demonstrating consistently high accuracy. For temperature prediction, MAE values were between 0.374 and 0.414 °C, with RMSE values ranging from 0.698 to 0.779 °C, all indicating the model's high precision. The predictions for charging capacity were equally encouraging, with MAE values between 0.019 and 0.021 Ah, and RMSE values from 0.038 to 0.044 Ah, showcasing the model's precise capture of changes in battery charging capacity. To comprehensively evaluate the model's performance in generating data for voltage, current rate, temperature, and charging capacity, the error metrics for these four data types were considered together, resulting in a total MAE value between 0.12 and 0.129, RMSE values maintained between 0.397 and 0.441, and KLD values between 0.004 and 0.011.

To further explore the generative capabilities of RCVAE, Figs. 4 and 5 display electrochemical data samples generated by the RCVAE model trained with different amounts of early-cycle data. These samples were randomly selected from the corresponding test sets. By specifying certain supervised conditions (EOL+ECL), the model is capable of generating the corresponding electrochemical data. For example, Fig. 4(a) shows the high consistency between the voltage, temperature, and charging capacity data generated by the RCVAE trained on the first 10 cycles and the original data, with R^2 values reaching at least 0.9754. For the current rate, although the quality of the generated data was slightly lower compared to other data types, the R^2 value was still 0.8546; Fig. 4(b) demonstrates that the data generated by the RCVAE trained on the first 20 cycles closely matches the original data in terms of voltage, current rate, temperature, and charging capacity, with R^2 values of at least 0.9743; In Fig. 4(c), data generated by the RCVAE trained with the first 30 cycles accurately reflects the original data, with R^2 values of at least 0.7421; As shown in Fig. 4(d), the voltage, temperature, and charging capacity data generated by the RCVAE trained with the first 40 cycles closely align with the original data, with R^2 values of at least 0.9185; Fig. 4(e) illustrates that the data generated by the RCVAE trained with the first 50 cycles almost perfectly matches the original data in terms of voltage, current rate, temperature, and charging capacity, with R^2 values of at least 0.9457. Fig. 5 shows the generative ability of RCVAE for individual samples using data from the first 60 to 100 cycles, similar to the results in Fig. 4, which will not be repeated here. Overall, RCVAE demonstrates good performance in generating various electrochemical data. Compared to traditional regression and classification algorithms, RCVAE provides more comprehensive electrochemical information. However, there is still room for improvement in the quality of the generated current rate data. This may be because current rate data is highly discrete, forming step-like patterns that are not smoothly continuous, making it challenging for machine learning to capture such patterns through mathematical rules. Conversely, the model shows stable and excellent ability to learn continuous data patterns, as seen in the figures.

To delve deeper into the variations in the generative capability of the RCVAE model with different amounts of early-cycle data, this study conducted a detailed visualization of error data, as shown in Fig. 6. In Fig. 6(a), the capability of the RCVAE to generate voltage data across varying early-cycle data amounts is displayed, where the MAE slightly exceeds 0.02 V, and the RMSE slightly exceeds 0.04 V. Fig. 6(b) presents the RCVAE's ability to generate current rate data across different early-cycle data volumes, with the MAE values consistently less than 0.3 and the RMSE slightly over 0.6. In Fig. 6(c), we observe the RCVAE's capacity to generate temperature data with varying early-cycle data amounts, where the MAE values remain around 0.4 °C, and the RMSE is below 0.8 °C. Finally, Fig. 6(d) shows the RCVAE's ability to generate charging capacity data across different early-cycle data amounts, with MAE values fluctuating around 0.02 Ah, and RMSE values around 0.04 Ah.

Fig. 6(e) displays a summary of errors for various types of electrochemical charging data generated by the RCVAE across different amounts of early-cycle data. In all cases, the MAE remains below 0.2, while the RMSE slightly exceeds 0.4. Overall, the variation in MAE across different early-cycle data amounts is minimal, while RMSE reaches its peak when the early-cycle count is at 100. In this scenario, we further demonstrate the generative capability of the trained RCVAE model. By randomly selecting a condition (i.e., label) from the test set, the corresponding electrochemical charging data was successfully generated.

The dimensions of the generated features are: $(n, \text{channels}, \text{depth}, \text{height}, \text{width})$, specifically $(n, 3, 2, 20, 20)$, where n represents the number of samples. Due to the high dimensionality, direct visualization is not feasible, so a slicing approach is applied along the depth dimension. To illustrate and present more generated samples, the charging capacity data is scaled and displayed as grayscale images. Although there are 3 channels representing red, green, and blue, the channels sliced from

Table 2
Results of electrochemical data generated by RCVAE.

Data type	V (V)		I		T (°C)		Qc (Ah)		Total		
Error type	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	KLD
10 (Cycles)	0.022	0.043	0.27	0.66	0.399	0.706	0.02	0.038	0.125	0.397	0.006
20 (Cycles)	0.023	0.044	0.285	0.683	0.391	0.737	0.021	0.04	0.127	0.413	0.005
30 (Cycles)	0.023	0.046	0.286	0.732	0.405	0.777	0.021	0.042	0.129	0.438	0.005
40 (Cycles)	0.021	0.043	0.269	0.685	0.374	0.698	0.019	0.038	0.12	0.402	0.004
50 (Cycles)	0.022	0.046	0.281	0.741	0.395	0.748	0.021	0.044	0.127	0.432	0.004
60 (Cycles)	0.022	0.046	0.281	0.734	0.413	0.773	0.02	0.042	0.129	0.437	0.006
70 (Cycles)	0.022	0.046	0.275	0.738	0.407	0.766	0.02	0.043	0.127	0.437	0.008
80 (Cycles)	0.022	0.045	0.271	0.72	0.404	0.764	0.019	0.043	0.126	0.431	0.011
90 (Cycles)	0.021	0.044	0.268	0.721	0.396	0.75	0.019	0.041	0.124	0.427	0.006
100 (Cycles)	0.022	0.047	0.279	0.739	0.414	0.779	0.02	0.043	0.129	0.441	0.01

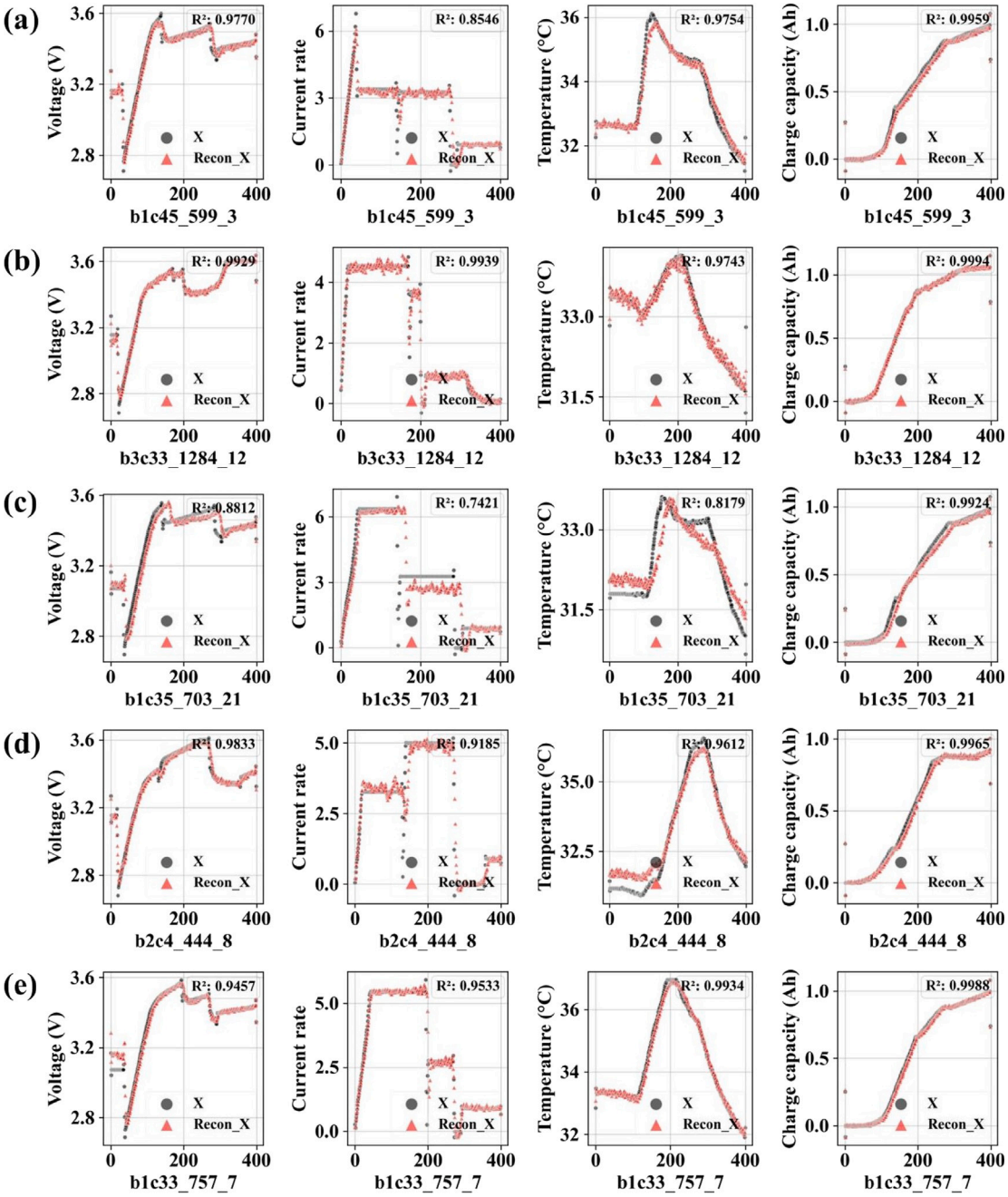


Fig. 4. Displays the voltage, current, temperature, and charging capacity data generated by RCVAE trained with data from different cycles: (a) the first 10 cycles; (b) the first 20 cycles; (c) the first 30 cycles; (d) the first 40 cycles; (e) the first 50 cycles.

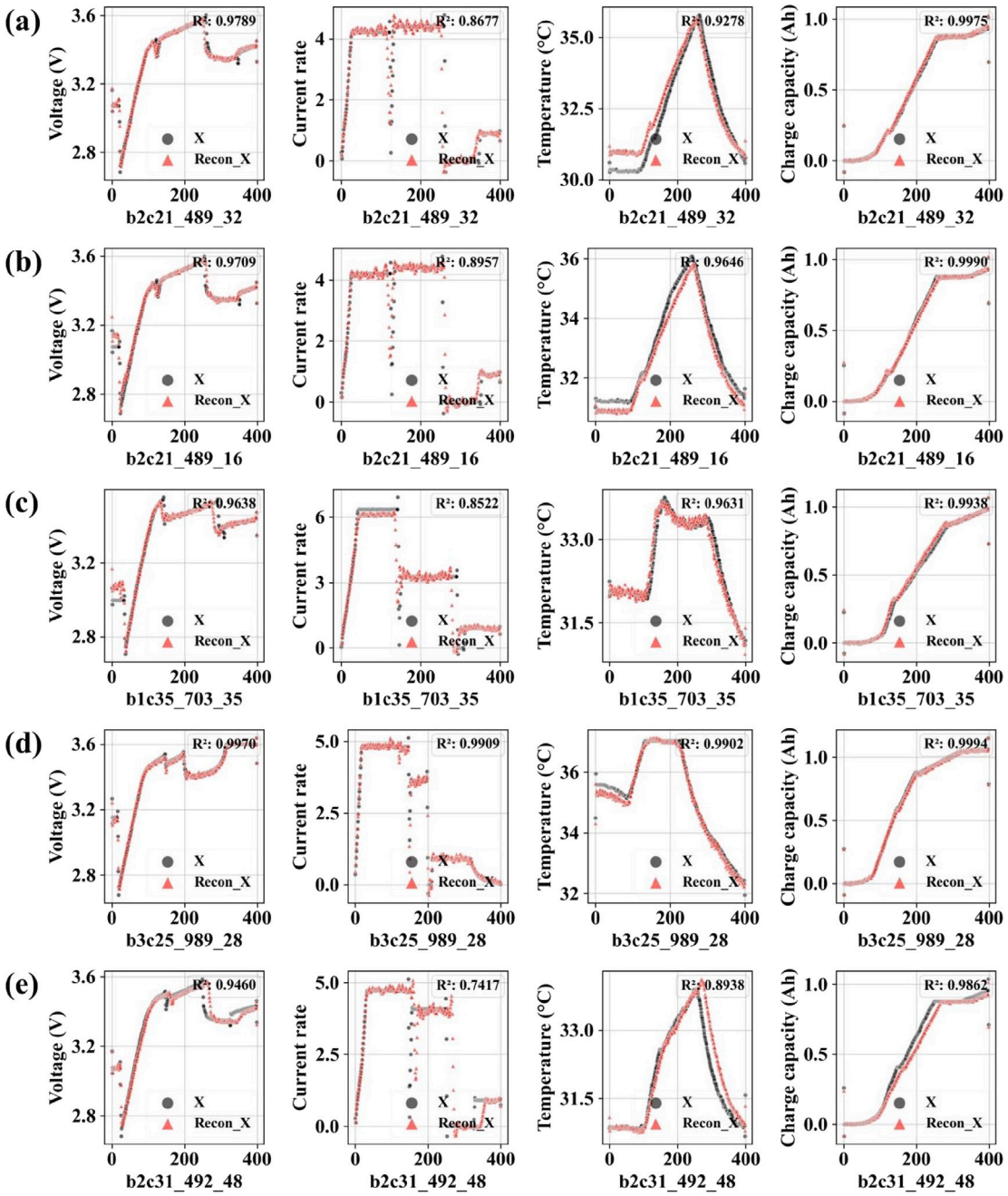


Fig. 5. Displays the voltage, current, temperature, and charging capacity data generated by RCVAE trained with data from different cycles: (a) the first 60 cycles; (b) the first 70 cycles; (c) the first 80 cycles; (d) the first 90 cycles; (e) the first 100 cycles.

this depth are identical (*charging capacity, charging capacity, charging capacity*), resulting in a grayscale image.

Another depth slice corresponds to the channels (*voltage, current rate, temperature*), where the voltage, current rate, and temperature data are combined and similarly scaled to be displayed in the form of RGB images.

Taking the grayscale images as an example, the first row displays the generated charging capacity data, while the second row shows the original charging capacity data. Each subplot is labeled at the bottom left with the battery number, EOL, and ECL. Above each pair of generated and original images, the SSIM (Structural Similarity Index) is indicated. SSIM is a measure used to assess the similarity between two images and is widely applied in the fields of image processing and computer vision. The calculation formula for the Structural Similarity Index (SSIM) is as follows:

$$\mathrm{SSIM}(x,y) = \left(\frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}\right) \cdot \left(\frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}\right) \tag{21}$$

where: x and y are local windows of the two images. μ_x and μ_y are the mean values of the two images within the window. σ_x^2 and σ_y^2 are the variances of the two images within the window. σ_{xy} is the covariance of the two images within the window. C_1 and C_2 are two constants used to prevent division by zero.

It can be observed that the SSIM values for each pair of generated and original images range between 0.987 and 0.999, indicating that the generated grayscale images, specifically the charging capacity data, are of high quality. The results show that the RCVAE can accurately generate corresponding high-quality charging capacity data for any specified EOL and ECL. The RGB images composed of voltage, current,

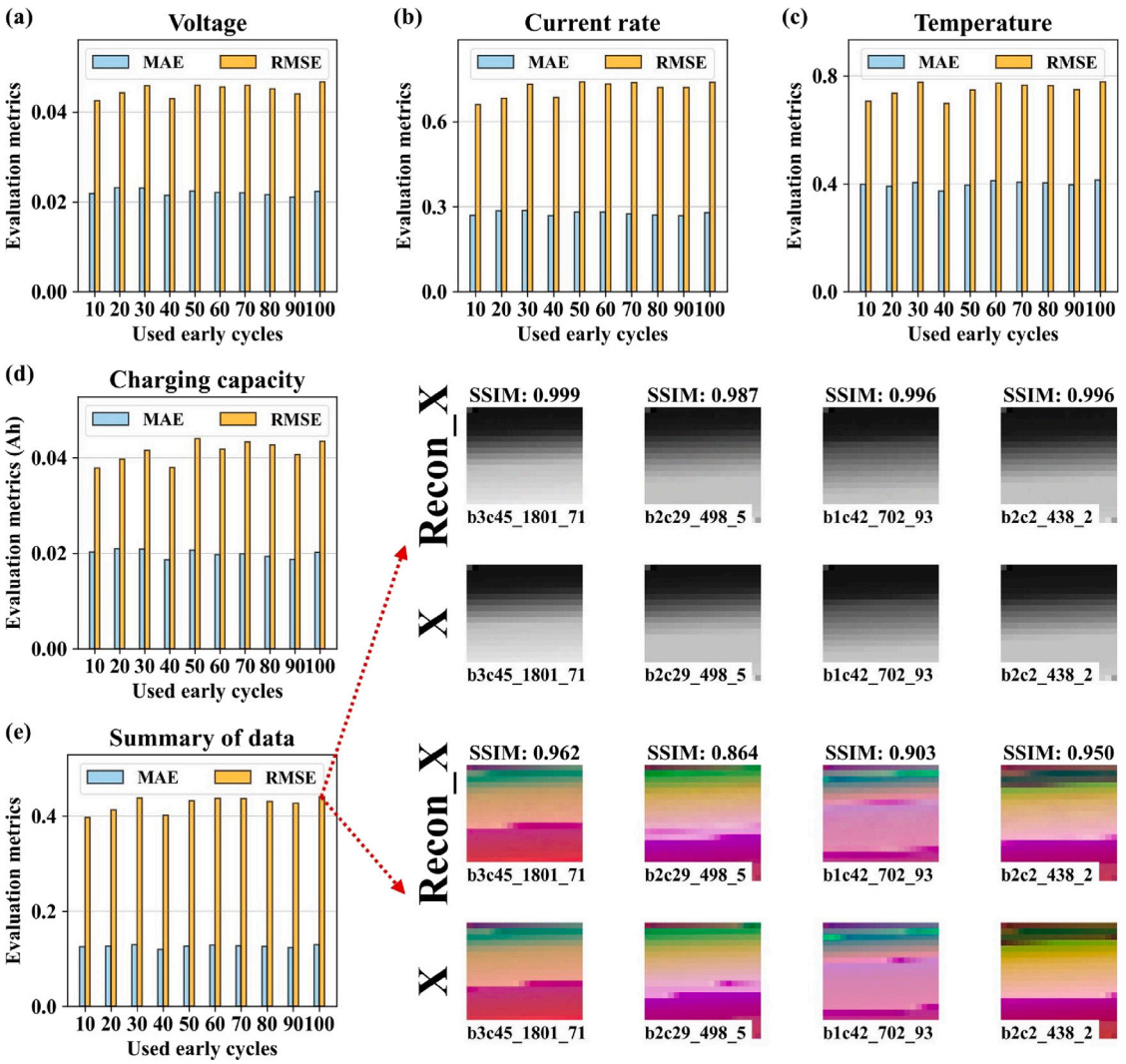


Fig. 6. Shows the statistical distribution of errors for different types of electrochemical data generated by RCVAE under various early-cycle conditions: (a) voltage error distribution, (b) current rate error distribution, (c) temperature error distribution, (d) charging capacity error distribution, and (e) a comprehensive summary of errors for all types of electrochemical data.

and temperature also demonstrate similar compatibility; due to space constraints, they are not elaborated here. The number of generated samples shown is limited. To more comprehensively demonstrate the generative capabilities of the RCVAE, Figures S1–S4 present more details of the samples generated by the RCVAE trained with different amounts of early-cycle data.

4.2. Analysis of the role of each component in the model

To gain a deeper understanding of the contribution of each component in the RCVAE, a series of ablation experiments were conducted. "Ablation experiments" in machine learning involve removing or modifying certain parts of the model to observe their impact on the model's performance, thereby identifying which components are crucial. Fig. 7(a) illustrates the changes in error when generating voltage data after removing different layers of the model. "Decn", "Encn", "Emb", and "None" represent scenarios where specific layers of the decoder, encoder, the embedding layer were removed, and the standard RCVAE model without any modifications, respectively. Additionally, the numbers following "Dec" and "Enc" indicate the specific number of layers removed. Given the critical roles of the first layer of the encoder and the last layer of the decoder, the experimental results show that even after removing any fully connected layer other than these two, the model's performance in generating voltage data remains excellent, demonstrating its high robustness due to the large number of layers. Notably, removing the embedding layer significantly increases the MAE of the generated data, with the MAE at least doubling (0.044–0.052 V) under various early data training conditions; when trained with data from the first 10–30 cycles, the MAE rose to 0.05 V or higher.

Fig. 7(b) shows the RMSE in generating voltage data, with trends similar to those of MAE. Given the model's architecture with multiple fully connected layers in both the encoder and decoder, even after removing one fully connected layer, the model's generative performance remains good, demonstrating its excellent robustness. However, the importance of the embedding layer is evident, as its removal significantly increases the RMSE of the generated data (0.061–0.078 V), especially under the training conditions of the first 10–30 cycles, where the RMSE (0.074–0.078 V) nearly doubles. Additionally, Figures S5–S7 present the heatmaps of MAE and RMSE for the ablation experiments of current rate, temperature data, and charging capacity data, respectively.

Additionally, Table 3 provides a detailed summary of the errors in the voltage and current rate data generated by the RCVAE after training with the first 100 cycles of data, as well as the weighted errors for all data types, which are consistent with previous observations. The ablation experiments for voltage and current rate (Table S1), along with those for temperature and charging capacity (Table S2), offer

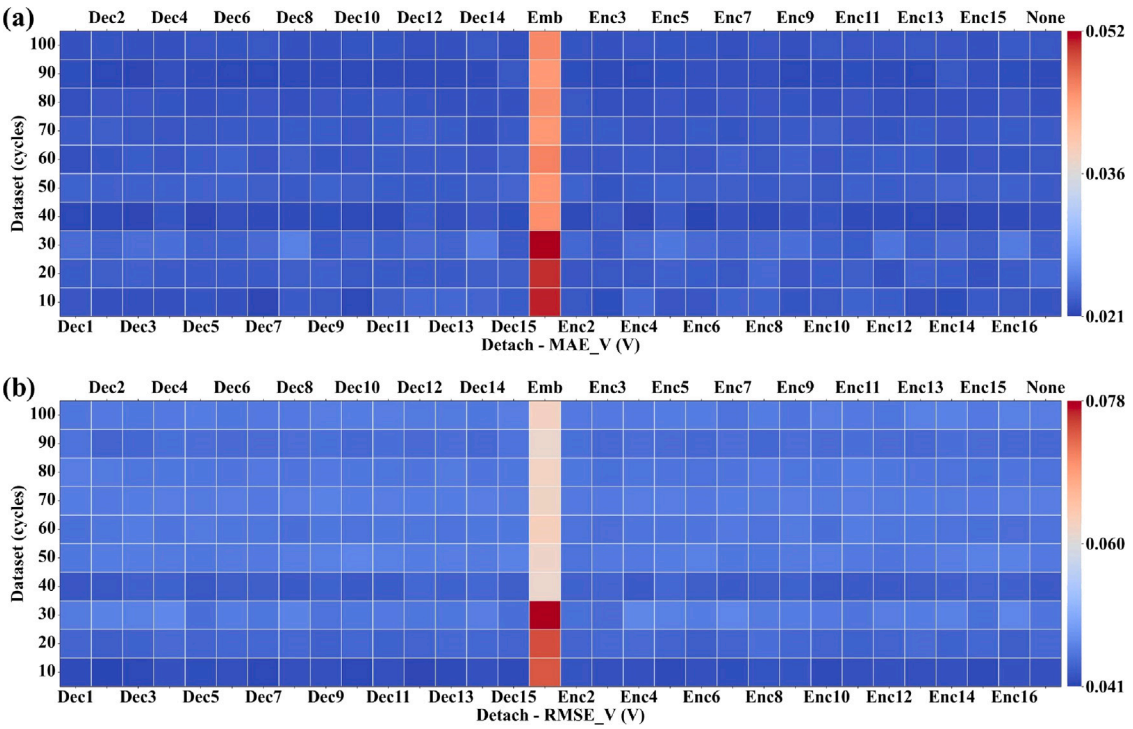


Fig. 7. Reveals the results of ablation experiments on RCVAE in generating voltage data.

comprehensive data across different volumes of early-cycle data, once again highlighting the critical role of the embedding layer. These tables fully confirm that the embedding layer is indispensable for maintaining the model's capability to generate high-quality data in the tasks of generating voltage, current rate, temperature, and charging capacity data.

4.3. Learning outcomes of the embedding layer

After discussing the importance of the embedding layer in generating electrochemical data in RCVAE in the previous section, this section further explores the working mechanism of the embedding layer and explains it through the visualization of the learned results. Fig. 8 shows the distribution of weights learned in the embedding layer. Given the numerous conditional categories resulting from the combination of EOL and ECL, the dimension of the embedding vectors is relatively high (embedding_dim = 473). To simplify the presentation, we first use t-SNE [43] to reduce the dimensionality of the embedding vectors to a two-dimensional space, followed by clustering analysis using the KNN algorithm [44].

Fig. 8(a) presents the clustering results of the embedding vectors after the model was trained with data from the first 10 cycles. Each cluster is annotated with two values: the average EOL and the average ECL within the cluster. Given that the model is trained on data from the first 10 cycles, the ECL represented in the clusters is generally around 5 or 6, which aligns with the actual variation patterns of electrochemical data in batteries: samples with different EOLs exhibit significant differences in electrochemical data. If the EOL is similar and the ECL is close, the variation in the battery's electrochemical data is smaller, showing a gradual change, as demonstrated and validated in Fig. 1(a). In other words, under the conditions formed by both EOL and ECL in the sample generation process, EOL plays a leading role, while ECL acts as an auxiliary modulator. It can be observed that different conditions, mainly EOL, with approximate values, are clustered together.

Fig. 8(b) shows the distribution of embedding vectors after the model was trained with data from the first 20 cycles. Compared to training with only the first 10 cycles, the number of labels has doubled, leading to a significant change in the distribution of the embedding vectors. The range of average EOL values across different clusters (138 = 892–754) has become more compact compared to the range during

Table 3
Ablation Experiment Results of RCVAE trained with data from the first 100 cycles.

Detach	MAE	RMSE	MAE_V (V)	RMSE_V (V)	MAE_I	RMSE_I
Decoder_1	0.125	0.432	0.022	0.046	0.274	0.732
Decoder_2	0.127	0.433	0.022	0.046	0.276	0.733
Encoder_2	0.125	0.433	0.022	0.046	0.272	0.732
Decoder_3	0.126	0.436	0.022	0.046	0.272	0.736
Encoder_3	0.125	0.436	0.022	0.046	0.272	0.737
Decoder_4	0.125	0.434	0.022	0.046	0.27	0.733
Encoder_4	0.127	0.435	0.022	0.046	0.277	0.73
Decoder_5	0.125	0.436	0.022	0.046	0.272	0.739
Encoder_5	0.126	0.437	0.022	0.046	0.274	0.737
Decoder_6	0.127	0.435	0.022	0.046	0.275	0.731
Encoder_6	0.126	0.436	0.022	0.046	0.273	0.739
Decoder_7	0.127	0.434	0.022	0.046	0.277	0.731
Encoder_7	0.124	0.427	0.022	0.045	0.269	0.724
Decoder_8	0.125	0.437	0.022	0.046	0.27	0.736
Encoder_8	0.126	0.438	0.022	0.046	0.275	0.735
Decoder_9	0.125	0.44	0.022	0.046	0.269	0.743
Encoder_9	0.125	0.432	0.022	0.046	0.272	0.731
Decoder_10	0.127	0.44	0.022	0.046	0.272	0.74
Encoder_10	0.127	0.439	0.022	0.046	0.275	0.742
Decoder_11	0.126	0.437	0.022	0.046	0.274	0.741
Encoder_11	0.128	0.435	0.022	0.046	0.279	0.737
Decoder_12	0.124	0.434	0.022	0.046	0.267	0.737
Encoder_12	0.127	0.434	0.022	0.046	0.28	0.732
Decoder_13	0.125	0.437	0.022	0.046	0.271	0.737
Encoder_13	0.128	0.44	0.022	0.047	0.278	0.741
Decoder_14	0.128	0.435	0.022	0.046	0.279	0.73
Encoder_14	0.127	0.44	0.022	0.047	0.274	0.744
Decoder_15	0.124	0.438	0.022	0.046	0.267	0.743
Encoder_15	0.125	0.435	0.022	0.046	0.269	0.735
Encoder_16	0.127	0.438	0.022	0.047	0.277	0.739
None	0.129	0.439	0.022	0.046	0.277	0.737
Embedding	0.304	0.653	0.045	0.062	0.664	0.97

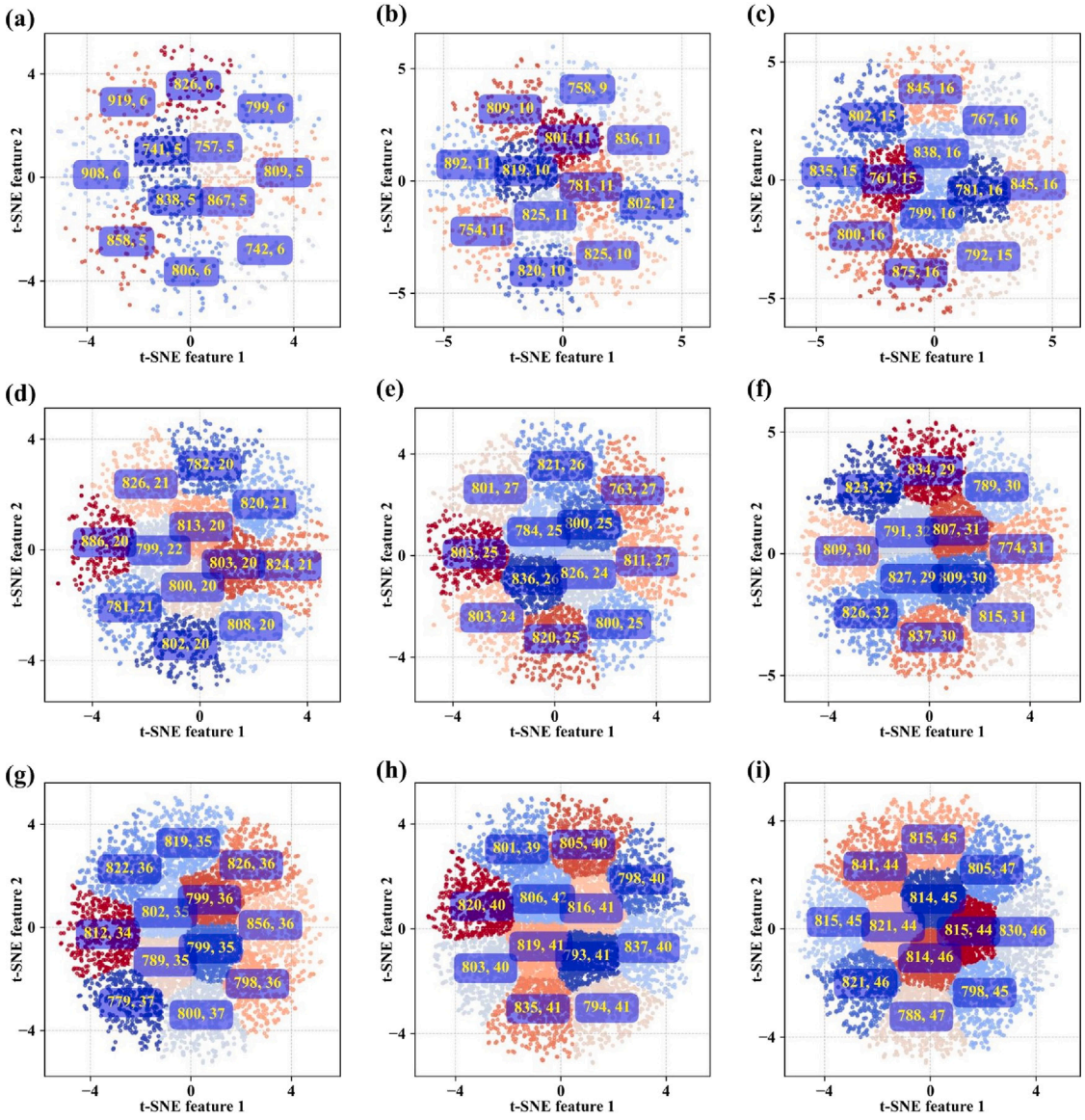


Fig. 8. Presents the learning outcomes of the RCVAE embedding layer trained with data from different cycles: (a) the first 10 cycles; (b) the first 20 cycles; (c) the first 30 cycles; (d) the first 40 cycles; (e) the first 50 cycles; (f) the first 60 cycles; (g) the first 70 cycles; (h) the first 80 cycles; (i) the first 90 cycles.

the first 10 cycles (178 = 919–741). Meanwhile, the differences in ECL within clusters (from 1 = 6–5 to 3 = 12–9) have not changed significantly. This suggests that the dominant effect of EOL has weakened, and the aggregation of similar EOL values has decreased, resulting in the enhanced auxiliary role of ECL.

Fig. 8(c) shows the distribution of embedding vectors after the model was trained with data from the first 30 cycles. Compared to training with only the first 10 cycles, the number of labels has tripled, leading to a noticeable change in the distribution of the embedding vectors. The range of average EOL values across different clusters (114 = 875–761) has become more compact compared to the range during the first 10 cycles (178 = 919–741) and the EOL range for the first 20 cycles (138 = 892–754). Although the ECL difference (1 = 16–15) did not increase compared to the ECL difference in the first 10 cycles (from 1 = 6–5), the range of EOL values (114 = 875–761) decreased significantly compared to (178 = 919–741), indicating that the effect of EOL is still weakening while the auxiliary role of ECL is further strengthened. It is noteworthy that two clusters both have labels (845, 16). Although these two clusters are relatively close in distance, they still have some separation, which may be due to some loss of information during dimensionality reduction using t-SNE, leading to a slight decrease in accuracy.

Fig. 8(d)–(i) show the distribution of RCVAE's supervised conditions when using data from the first 40–90 cycles, where the extreme differences in EOL values within different clusters have significantly decreased compared to earlier stages. However, the extreme differences in ECL have relatively increased. This pattern is consistent with the trends observed in Fig. 8(a)–(c), indicating that as the number of early cycles used increases, the aggregation of similar EOL values decreases, the influence of EOL in guiding data generation under supervision gradually weakens, while the role of ECL in the conditions gradually strengthens. Figure S6 shows the distribution of embedding layer weights for RCVAE trained with data from the first 100 cycles, maintaining consistency with the patterns observed in Fig. 8. These findings suggest that under different data distribution scenarios, the combination of EOL and ECL as strings effectively work together to adapt to different scenario needs, effectively guiding the generation of electrochemical data. This further highlights the importance of the embedding layer and the supervised condition settings in this study, as well as their central role in the RCVAE model.

5. Conclusion

By integrating the embedding layer into the CVAE model, this study successfully developed the RCVAE, specifically designed to enhance the generative capability for electrochemical data. The study employed a quasi-video data preprocessing method that effectively integrates different types of electrochemical data (such as voltage, current, temperature, and charging capacity) as input for the RCVAE. In this work, EOL and ECL are defined as supervisory conditions for the generative artificial intelligence, and in combination with customized training and inference algorithms, the RCVAE generates the required charging data in real-time based on the battery's ECL and EOL, avoiding storage of irrelevant information and saving space and resources. Additionally, the RCVAE employs a lightweight architecture, offering fast generation speeds to quickly provide the required voltage, current, temperature, and charging capacity data.

Experimental results confirm that the RCVAE can accurately generate various types of charging data. When trained under conditions with different amounts of early-cycle data, the RCVAE demonstrates outstanding capability in generating electrochemical data. Taking voltage data as an example, when using data from the first 10 to the first 100 cycles as the training set, the MAE values were 0.022, 0.023, 0.023, 0.022, 0.022, 0.022, 0.022, 0.022, 0.021, 0.022 V respectively; the RMSE values were 0.042, 0.045, 0.045, 0.043, 0.046, 0.045, 0.046, 0.045, 0.045, 0.046 V respectively. The ablation experiments on RCVAE revealed the model's high robustness, as its performance remained stable even after removing a layer. Notably, a significant drop in generative performance occurred after removing the embedding layer, highlighting the importance of this improvement to the CVAE model.

Further analysis of the learning outcomes of the embedding layer reiterated its critical role and the significance of combining EOL and ECL as supervisory conditions. Given the similarity in data characteristics observed in other tasks, this suggests that the RCVAE has the potential for broad applications across a wider range of fields.

CRediT authorship contribution statement

Lidang Jiang: Writing – original draft, Software, Methodology, Conceptualization. **Changyan Hu:** Formal analysis. **Sibei Ji:** Visualization, Data curation. **Hang Zhao:** Software, Formal analysis. **Junxiong Chen:** Writing – review & editing. **Ge He:** Writing – review & editing, Supervision, Software, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

This work was supported by the National Key Research and Development Program of China (No. 2021YFB40005).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.apenergy.2024.124604.

References

[1] Hofmann Jana, Guan Dabo, Chalvatzis Konstantinos, Huo Hong. Assessment of electrical vehicles as a successful driver for reducing CO2 emissions in China. Appl Energy 2016;184:995–1003.
[2] Jiang Jiuchun, Ruan Haijun, Sun Bingxiang, Wang Leyi, Gao Wenzhong, Zhang Weige. A low-temperature internal heating strategy without lifetime reduction for large-size automotive lithium-ion battery pack. Appl Energy 2018;230:257–66.
[3] Opitz A, Badami P, Shen L, Vignarooban K, Kannan Arunachala Mada. Can li-ion batteries be the panacea for automotive applications? Renew Sustain Energy Rev 2017;68:685–92.
[4] Xiong Rui, Zhang Yongzhi, He Hongwen, Zhou Xuan, Pecht Michael G. A double-scale, particle-filtering, energy state prediction algorithm for lithium-ion batteries. IEEE Trans Ind Electron 2017;65(2):1526–38.
[5] Kacetl Jan, Fang Jingyang, Kacetl Tomáš, Tashakor Nima, Goetz Stefan. Design and analysis of modular multilevel reconfigurable battery converters for variable bus voltage powertrains. IEEE Trans Power Electron 2022;38(1):130–42.
[6] Zhang Yan, Wang Shuting, Xia He, Guo Jing, He Kangxin, Huang Chenjie, Luo Rui, Chen Yanfei, Xu Kaijin, Gao Hainv, et al. Identification of monocytes associated with severe COVID-19 in the PBMCs of severely infected patients through single-cell transcriptome sequencing. Engineering 2022;17:161–9.
[7] Huang Peifeng, Ping Ping, Li Ke, Chen Haodong, Wang Qingsong, Wen Jennifer, Sun Jinhua. Experimental and modeling analysis of thermal runaway propagation over the large format energy storage battery module with Li4Ti5O12 anode. Appl Energy 2016;183:659–73.
[8] Wang Qingsong, Ping Ping, Zhao Xuejuan, Chu Guanquan, Sun Jinhua, Chen Chunhua. Thermal runaway caused fire and explosion of lithium ion battery. J Power Sources 2012;208:210–24.
[9] Wang Shunli, Jin Siyu, Bai Dekui, Fan Yongcun, Shi Haotian, Fernandez Carlos. A critical review of improved deep learning methods for the remaining useful life prediction of lithium-ion batteries. Energy Rep 2021;7:5562–74.
[10] Liu Jie, Saxena Abhinav, Goebel Kai, Saha Bhaskar, Wang Wilson. An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries. In: Annual conference of the PHM society. Vol. 2, 2010, Issue Number: 1.
[11] Bian Chong, He Huoliang, Yang Shunkun. Stacked bidirectional long short-term memory networks for state-of-charge estimation of lithium-ion batteries. Energy 2020;191:116538.
[12] Ma L, Hu C, Cheng F. State of charge and state of energy estimation for lithium-ion batteries based on a long short-term memory neural network. J Energy Storage 2021;37:102440.
[13] Shu Xing, Li Guang, Zhang Yuanjian, Shen Shiquan, Chen Zheng, Liu Yonggang. Stage of charge estimation of lithium-ion battery packs based on improved cubature Kalman filter with long short-term memory model. IEEE Trans Transp Electrif 2020;7(3):1271–84.
[14] Tian Jinpeng, Xiong Rui, Shen Weixiang, Lu Jiahuan. State-of-charge estimation of LiFePO4 batteries in electric vehicles: A deep-learning enabled approach. Appl Energy 2021;291:116812.
[15] Tian Jinpeng, Xiong Rui, Shen Weixiang, Lu Jiahuan, Yang Xiao-Guang. Deep neural network battery charging curve prediction using 30 points collected in 10 min. Joule 2021;5(6):1521–34.
[16] He Wei, Williard Nicholas, Osterman Michael, Pecht Michael. Prognostics of lithium-ion batteries based on Dempster–Shafer theory and the Bayesian Monte Carlo method. J Power Sources 2011;196(23):10314–21.
[17] Preger Yuliya, Barkholtz Heather M, Fresquez Armando, Campbell Daniel L, Juba Benjamin W, Romàn-Kustas Jessica, Ferreira Summer R, Chalamala Babu. Degradation of commercial lithium-ion cells as a function of chemistry and cycling conditions. J Electrochem Soc 2020;167(12):120532.
[18] Raj Trishna, Wang Andrew A, Monroe Charles W, Howey David A. Investigation of path-dependent degradation in lithium-ion batteries. Batter Supercaps 2020;3(12):1377–85.
[19] Williard Nick, He Wei, Osterman Michael, Pecht Michael. Comparative analysis of features for determining state of health in lithium-ion batteries. Int J Progn Health Manage 2013;4(1).
[20] Kollmeyer Phillip. Panasonic 18650pf li-ion battery data. Mendeley Data 2018;1(2018).
[21] CALCE Battery Research Group. CALCE battery research group homepage. 2018, https://calce.umd.edu/battery-research-overview.
[22] Cameron Zachary, Kulkarni Chetan S, Luna Ali Guarneros, Goebel Kai, Poll Scott. A battery certification testbed for small satellite missions. In: 2015 IEEE autotestcon. IEEE; 2015, p. 162–8.
[23] Kulkarni C, Guarneros A. Small satellite power simulation data set. NASA Ames Progn Res Cent 2008.
[24] Saha Bhaskar, Goebel Kai. Battery data set. NASA AMES Progn Data Repos 2007.
[25] Severson Kristen A, Attia Peter M, Jin Norman, Perkins Nicholas, Jiang Benben, Yang Zi, Chen Michael H, Aykol Muratahan, Herring Patrick K, Fraggedakis Dimitrios, et al. Data-driven prediction of battery cycle life before capacity degradation. Nature Energy 2019;4(5):383–91.

[26] Attia Peter M, Grover Aditya, Jin Norman, Severson Kristen A, Markov Todor M, Liao Yang-Hung, Chen Michael H, Cheong Bryan, Perkins Nicholas, Yang Zi, et al. Closed-loop optimization of fast-charging protocols for batteries with machine learning. Nature 2020;578(7795):397–402.
[27] Pyne Moinak, Yurkovich Benjamin J, Yurkovich Stephen. Generation of synthetic battery data with capacity variation. In: 2019 IEEE conference on control technology and applications. CCTA, IEEE; 2019, p. 476–80.
[28] Vaswani Ashish, Shazeer Noam, Parmar Niki, Uszkoreit Jakob, Jones Llion, Gomez Aidan N, Kaiser Łukasz, Polosukhin Illia. Attention is all you need. Adv Neural Inf Process Syst 2017;30.
[29] Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, Warde-Farley David, Ozair Sherjil, Courville Aaron, Bengio Yoshua. Generative adversarial nets. Adv Neural Inf Process Syst 2014;27.
[30] Karras Tero, Laine Samuli, Aila Timo. A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 4401–10.
[31] Gulrajani Ishaan, Ahmed Faruk, Arjovsky Martin, Dumoulin Vincent, Courville Aaron C. Improved training of wasserstein gans. Adv Neural Inf Process Syst 2017;30.
[32] Dosovitskiy Alexey, Beyer Lucas, Kolesnikov Alexander, Weissenborn Dirk, Zhai Xiaohua, Unterthiner Thomas, Dehghani Mostafa, Minderer Matthias, Heigold Georg, Gelly Sylvain, et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2020, arXiv preprint arXiv:2010.11929.
[33] Song Yang, Ermon Stefano. Generative modeling by estimating gradients of the data distribution. Adv Neural Inf Process Syst 2019;32.
[34] Devlin Jacob, Chang Ming-Wei, Lee Kenton, Toutanova Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018, arXiv preprint arXiv:1810.04805.
[35] Radford Alec, Narasimhan Karthik, Salimans Tim, Sutskever Ilya, et al. Improving language understanding by generative pre-training. OpenAI; 2018.
[36] Rombach Robin, Blattmann Andreas, Lorenz Dominik, Esser Patrick, Ommer Björn. High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022, p. 10684–95.
[37] Kingma Diederik P, Welling Max. Auto-encoding variational bayes. 2013, arXiv preprint arXiv:1312.6114.
[38] Sohn Kihyuk, Lee Honglak, Yan Xinchen. Learning structured output representation using deep conditional generative models. Adv Neural Inf Process Syst 2015;28.
[39] Mikolov Tomas, Chen Kai, Corrado Greg, Dean Jeffrey. Efficient estimation of word representations in vector space. 2013, arXiv preprint arXiv:1301.3781.
[40] Yang Yixin. A machine-learning prediction method of lithium-ion battery life based on charge process for different applications. Appl Energy 2021;292:116897.
[41] Zhang Qisong, Yang Lin, Guo Wenchao, Qiang Jiaxi, Peng Cheng, Li Qinyi, Deng Zhongwei. A deep learning method for lithium-ion battery remaining useful life prediction based on sparse segment data via cloud computing system. Energy 2022;241:122716.
[42] Jiang Lidang, Li Zhuoxiang, Hu Changyan, Chen Junxiong, Huang Qingsong, He Ge. A robust adapted flexible parallel neural network architecture for early prediction of lithium battery lifespan. Energy 2024;132840.
[43] Maaten LVD. Visualizing data using t-SNE. J Mach Learn Res 2008;9(Nov):2579.
[44] Cover Thomas, Hart Peter. Nearest neighbor pattern classification. IEEE Trans Inf Theory 1967;13(1):21–7.