

## Article

# A Novel Method for State of Charge Estimation in Lithium-Ion Batteries Using Temporal Convolutional Network and Multi-Verse Optimization

Yuanmao Li, Guixiong Liu \* and Wei Deng

School of Mechanical & Automotive Engineering, South China University of Technology, Guangzhou 510640, China; me\_lym@mail.scut.edu.cn (Y.L.); 202310180425@mail.scut.edu.cn (W.D.)

\* Correspondence: megxliu@scut.edu.cn

**Abstract:** This study presents a novel data-driven method for state-of-charge estimation in lithium-ion batteries. It integrates a temporal convolutional network with multi-verse optimization to enhance the accuracy of predicting the state of charge. The temporal convolutional network possesses advantages such as an extended memory window and efficient parallel computation, exhibiting exceptional performance in time-series tasks for state of charge estimation. Its hyperparameters are optimized by adopting multi-verse optimization to obtain better model performance. The driving model utilizes various measurable data as inputs, including battery terminal voltage, current, and surface temperature. To validate the effectiveness of the proposed method, extensive datasets from diverse dynamic working conditions at different ambient temperatures are employed for model training, validation, and testing. The numerical outcomes provide evidence of the proposed method's superior performance compared to the other two methods, providing a more robust and accurate solution for the state of charge estimation in lithium-ion batteries.

**Keywords:** lithium-ion battery; state of charge; temporal convolutional network; multi-verse optimization; dynamic working condition



**Citation:** Li, Y.; Liu, G.; Deng, W. A Novel Method for State of Charge Estimation in Lithium-Ion Batteries Using Temporal Convolutional Network and Multi-Verse Optimization. *Batteries* **2024**, *10*, 12. <https://doi.org/10.3390/batteries10010012>

Received: 28 November 2023  
Revised: 24 December 2023  
Accepted: 28 December 2023  
Published: 29 December 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Given the prevailing energy crisis and the rapid advancements in alternative energy sources, the lithium-ion power battery has emerged as a pivotal determinant of the progress of electric vehicles [1,2]. The state of charge (SOC) of a lithium-ion power battery is a critical parameter reflecting the ratio between its remaining capacity and maximum available capacity [3]. This parameter plays a fundamental role in assessing the durability, reliability monitoring, and cruising range estimation within the battery management system [4,5]. Accurate SOC estimation is crucial for optimizing battery performance, including the control strategy, balancing technology, energy utilization efficiency, and cycle life. However, direct measurement of the SOC is challenging, necessitating the estimation of the SOC through the battery characteristic variables such as charge/discharge current, terminal voltage, internal resistance, temperature, etc., which poses challenges to achieving accurate real-time SOC estimation [6,7]. Furthermore, during the operation of electric vehicles, the performance of lithium-ion batteries is affected by various factors, including aging, temperature shocks, and dynamic driving conditions [8,9]. These factors pose challenges to the implementation of rapid and accurate SOC estimation techniques in the battery management system. As a result, accurate SOC estimation in real time is significant.

SOC estimation methods can be classified into four categories: lookup table, ampere-hour integration, model-based state observer, and data-driven modeling approaches [10–13]. The lookup table method establishes a correlation between the SOC and measurable data, such as open circuit voltage (OCV), to estimate the SOC [14,15]. However, achieving accurate OCV measurement necessitates a well-conditioned battery to stabilize parameters,

restricting the online application of SOC estimation in battery management systems [16]. On the other hand, the ampere-hour integration method involves assessing the total current intake throughout a specified period to obtain available charge information but relies on initial SOC values or regular measurements/estimation of battery capacity [17]. Consequently, the current emphasis on SOC estimation methods is primarily directed toward the battery model-based state observer and the data-driven modeling method.

Battery model-based state observer methods use battery models (equivalent circuit or electrochemical modeling) and integrate state observers like Kalman filters, nonlinear observers, sliding mode observers, etc., to continuously improve SOC estimation through closed-loop feedback. Tian et al. [18] proposed a gain nonlinear observer and a second-order RC model optimized with the particle swarm optimization (PSO) algorithm for SOC estimation, which exhibited excellent performance in various working conditions. Shrivastava et al. [19] investigated the adaptive extended Kalman filter (AEKF) with double forgetting factors and developed a second-order RC model to predict battery SOC and SOE together. Three battery cells with different materials were tested under various dynamic operating conditions, demonstrating the capability to rapidly obtain precise SOC estimates, even from inaccurate initial states. Sun et al. [20] developed an intelligent AEKF with a Thevenin model for accurate SOC estimation, addressing changes in residual sequence distribution caused by dynamic load current, battery model error, and Taylor approximation. The experiment results indicated the method's robustness to parameter uncertainty. Sakile et al. [21] explored an extended nonlinear state observer that utilized a second-order resistor-capacitance (RC) model for predicting SOC. It represented the OCV-SOC relationship employing a 9th-order polynomial and evaluated the stability and convergence rate of the observer through Lyapunov criterion analysis. Comparative analysis with an unscented Kalman filter (UKF) and sliding mode observer demonstrated the improved dynamic performance and estimation accuracy of this method.

Data-driven modeling methods, utilizing machine learning or deep learning technology, train on measurable data. By inputting the acquired data into the trained model, accurate SOC estimation is achieved. Yang et al. [22] explored an integrated data-driven model employing both long short-term memory (LSTM) and an UKF for accurate SOC estimation. Measurable data like voltage, current, and temperature were utilized as input variables. By effectively mitigating noise in the LSTM network, the UKF enhanced the accuracy of the SOC estimation. The findings demonstrated precise SOC estimation within a temperature range spanning from 0 to 50 °C. Hannan et al. [23] proposed a self-supervised Transformer model for SOC estimation without marked features or adaptive filtering, demonstrating accurate results even in various ambient temperatures. Yang et al. [24] explored the bidirectional LSTM network for SOC estimation, enhancing information integrity and timing dependence. Moreover, the Bayesian optimization algorithm was utilized to search for optimal network parameters, improving model performance. Experimental results demonstrated the high accuracy and generalizability of this model across various battery types and temperature environments. Yan et al. [25] introduced a hybrid data-driven method that combines LSTM and an improved particle filter (IPF), enhancing the precision of the SOC estimation by augmenting the particle filter with an adaptive algorithm. The results demonstrated low estimation errors for various battery types, temperatures, and driving cycles.

Model-based state observer estimation methods effectively reduce Gaussian white noise effects on measurement data. Accurate battery models are crucial for precise SOC estimation; however, commonly used models like the equivalent circuit model introduce modeling errors. Moreover, nonlinear prediction capabilities are essential for the state observer. Data-driven modeling methods do not necessitate the consideration of complex electrochemical and physical attributes in the battery system, enabling better battery SOC estimation. LSTM networks are commonly used for accurate SOC estimation due to their effective capture of timing and nonlinear characteristics in the battery SOC [26,27]. However, they may face overfitting issues with limited training data and encounter gradient-

vanishing challenges in longer sequences [28]. Transformers, with their self-attention mechanism, address the gradient vanishing problem and find applications in SOC prediction [29]. Nevertheless, the self-attention mechanism's association of each position with all others introduces sparsity issues due to an abundance of weight parameters. Furthermore, the Transformer exhibits increased computational complexity in longer sequences [30]. Hence, selecting a deep learning network that efficiently captures long-term dependencies, is robust against gradient vanishing, and maintains high computational efficiency is crucial for accurate SOC estimation.

S. Bai et al. [31] optimized the basic architecture of temporal convolutional networks (TCNs), applying them to sequence data modeling. This type of temporal convolutional neural network has made significant advancements in the field of time-series data processing. Its design aims to better capture long-range dependencies in temporal data. Compared to traditional recurrent neural networks and LSTM, TCNs efficiently capture information in time series through parallel convolution operations. The TCN has found widespread applications and garnered attention in various domains, including natural language processing [32], speech recognition [33], and time-series prediction [34,35]. The TCN is widely used in time-series prediction for its flexible receptive fields, resistance to gradient vanishing, and effective capture of long-term dependencies [36–38]. By employing convolutional operations, a TCN enables parallel computation across the entire input sequence, enhancing both training and inference speed. Notably, the TCN excels in capturing diverse long-term dependencies and demonstrates enhanced generalization, particularly in scenarios with limited data. Its convolutional operations simplify the handling of multivariate inputs in time-series prediction tasks. In contrast to LSTM, the TCN is less susceptible to gradient vanishing issues, making it a promising choice for SOC estimation. The primary contributions of this work include a novel data-driven method using a TCN and multi-variate optimization (MVO) that was proposed for SOC estimation; MVO was employed to optimize the hyperparameters of the TCN. Various measurable data, such as battery terminal voltage, current, and surface temperature, were used as inputs to the driving model to achieve SOC estimation. Data from nine different dynamic working conditions at different ambient temperatures was used for the training, validation, and testing of data-driven models. The sections of this paper are organized as follows: Section 2 presents the framework of the SOC estimation method based on TCN-MVO; Section 3 discusses the performance of the different data-driven methods under nine dynamic working conditions at various ambient temperatures; and Section 4 provides a concise conclusion.

## 2. SOC Estimation Method Based on TCN-MVO

In this section, the SOC estimation method based on TCN-MVO is described in detail. Firstly, the evaluation metrics used to assess SOC estimation performance are introduced. Next, the overall SOC estimation framework employed in this study is proposed. In this framework, the time-varying sequential data of battery terminal voltage, current, and surface temperature are used as input for training and prediction in the data-driven model. Subsequently, a comprehensive explanation of the TCN modeling approach for SOC estimation is presented. Additionally, the method of optimizing TCN hyperparameters using the MVO algorithm is outlined. Finally, the objective function used in optimizing deep learning networks through the MVO algorithm is introduced.

### 2.1. Evaluation Metrics

For a thorough examination of SOC estimation performance across different deep learning networks in lithium-ion batteries, this work employs multiple metrics for a comprehensive assessment.

### 2.1.1. Root-Mean-Square Error

The root-mean-square error (*RMSE*) is used as an evaluation metric to measure the deviation between estimated values and their corresponding truth values, and it is given by:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2}, \quad (1)$$

where  $n$  represents the number of estimations,  $\hat{y}_i$  denotes the estimated value, and  $y_i$  refers to the true value.

### 2.1.2. Mean Absolute Error

The mean absolute error (*MAE*) is primarily employed for assessing the accuracy and extent of deviation in prediction models. Unlike *RMSE*, *MAE* remains unaffected by outliers and solely focuses on the absolute value of errors. Consequently, it finds extensive application in scenarios necessitating outlier handling or emphasizing the magnitude of prediction errors. It is defined as follows:

$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|. \quad (2)$$

### 2.1.3. Maximum Error

The maximum error (*MAX*) is primarily used to calculate the maximum predictive error of a model for the entire dataset. It directly considers the absolute value of the error without undergoing any square operations, making it highly sensitive to outliers. It is expressed as follows:

$$MAX = \max |\hat{y}_i - y_i|, i = 1, 2, \dots, n. \quad (3)$$

### 2.1.4. Coefficient of Determination $R^2$

The coefficient of determination, denoted as  $R^2$ , represents the proportion of the regression sum of squares to the total sum of squares in a regression model. It serves as a metric for assessing the fit and excellence of the regression model. The closer  $R^2$  is to 1, the higher the proportion of the regression sum of squares to the total sum of squares, indicating a better fit. Conversely, if  $R^2$  is close to 0, it suggests a poor model fit. It is expressed as follows:

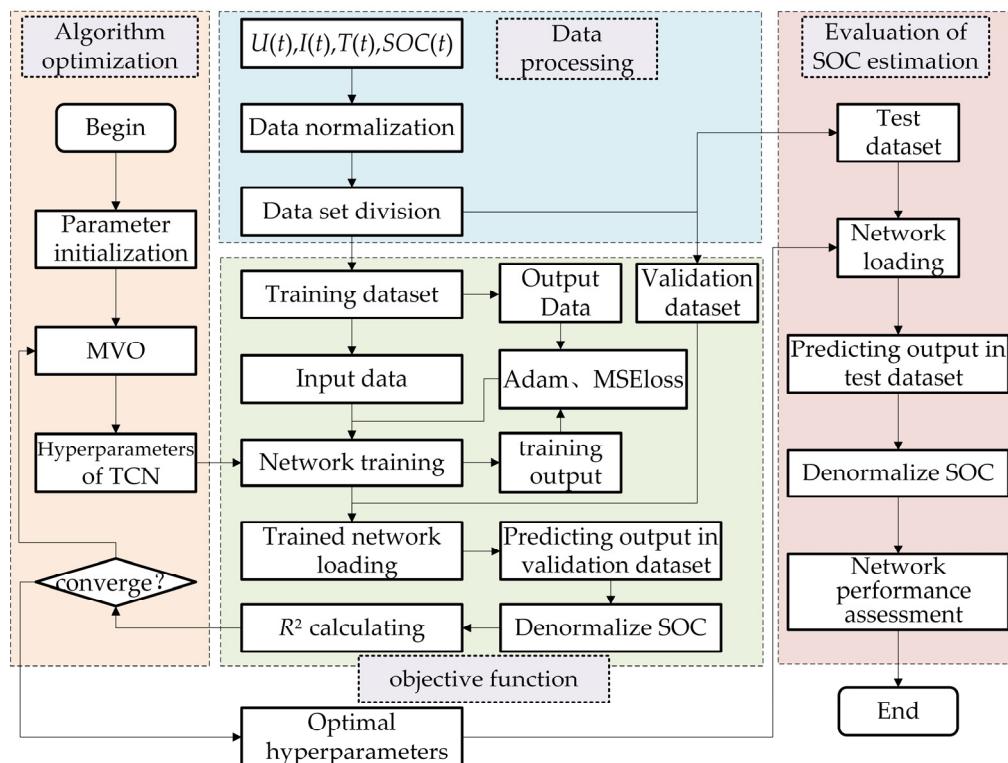
$$R^2 = 1 - \frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{\sum_{i=1}^n (\bar{y} - y_i)^2}, \quad (4)$$

where  $\bar{y}$  denotes the average value of  $y_i$ .

## 2.2. SOC Estimation Framework

Figure 1 is the flowchart depicting the SOC estimation framework based on TCN-MVO. It is structured into four modules: data processing, algorithm optimization, objective function, and evaluation of the SOC estimation.

Data processing includes the techniques of data normalization and the division of datasets. The MVO algorithm is employed for optimizing the optimal hyperparameter solution of the TCN in algorithmic optimization. The objective function mainly implements the training and verification processes of deep learning. The network training uses the Adam optimizer, the loss function uses mean square error (MSE) loss, and the network verification uses the indicator  $R^2$  of Equation (4) for evaluation. The closer  $R^2$  is to 1, the more optimal its hyperparameters are, indicating superior performance of the training network. Finally, the optimized TCN is employed for SOC estimation, and the metrics are subsequently computed and evaluated using Equations (1)–(3).



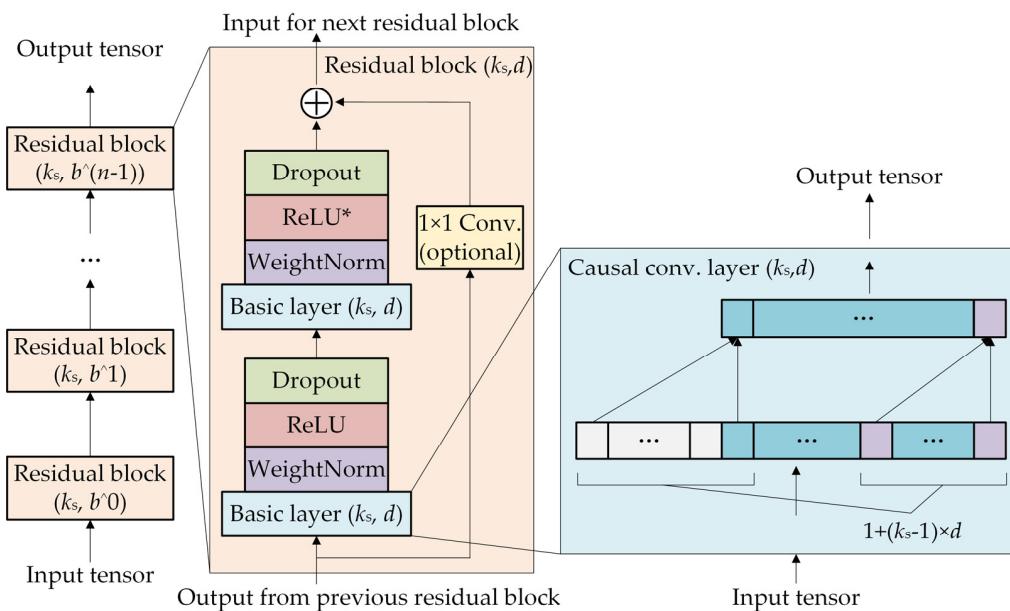
**Figure 1.** Flowchart depicting the SOC estimation framework based on TCN-MVO.

### 2.3. TCN Modeling for SOC Estimation

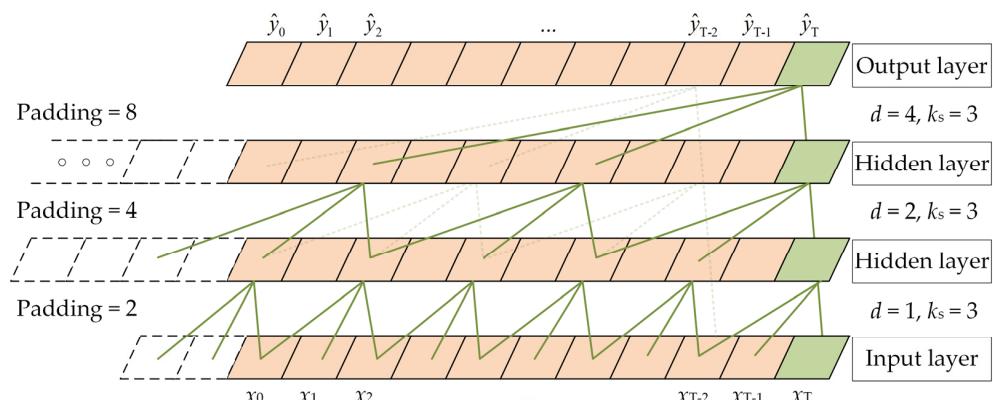
The TCN can effectively mitigate common deficiencies in recurrent neural networks, such as gradient vanishing and exploding. Through parallel computation, the network's performance can be enhanced. The SOC estimation will be performed utilizing the TCN in this work.

Figure 2 illustrates the basic architecture of the TCN. It is apparent that the TCN model is primarily constructed through the interconnection of residual blocks. These blocks comprise causal convolutions, dilated convolutions, weight normalization, rectified linear unit (ReLU) activation function, random dropout, and  $1 \times 1$  convolution. The incorporation of a  $1 \times 1$  convolution facilitates the transfer of network information across layers while maintaining consistency in input–output dimensions. The symbol “ $*$ ” following ReLU indicates its presence in all layers except the final one, ensuring the potential for negative output values.

Figure 3 depicts a schematic of dilated causal convolution. Causal convolution is unidirectional, implying that the data in layer  $i$  at time  $k$  only depend on the data at or before time  $k$  in layer  $i - 1$ . The augmentation of network layers holds the potential to enhance the capabilities of acquiring long-range data. Dilated convolution refers to the insertion of gaps or holes within the conventional convolution operation, where the sampling rate is regulated by a dilation factor denoted as  $d$ . It achieves an increased receptive field by employing spaced sampling. With a larger dilation factor  $d$ , the convolutional output can capture information from longer temporal sequences without compromising information integrity. This method allows for a larger receptive field with fewer network layers. Meanwhile,  $n$  refers to the number of residual blocks, and  $b$  denotes the dilation base. It follows that  $d$  can be calculated as  $d = b^{(n-1)}$ . Zero-padding is necessary for each layer of the network due to the incorporation of holes in dilated convolutions. The padding amount  $p$  is determined by the convolution kernel size  $k_s$  and satisfies the condition  $k_s \geq b$ , where  $p = (k_s - 1) \cdot d$ .



**Figure 2.** Basic architecture diagram of the TCN.



**Figure 3.** Schematic diagram of dilated causal convolution.

As a residual block is composed of two causal convolution layers, the total receptive field size, denoted as  $r_f$ , for the TCN is given by:

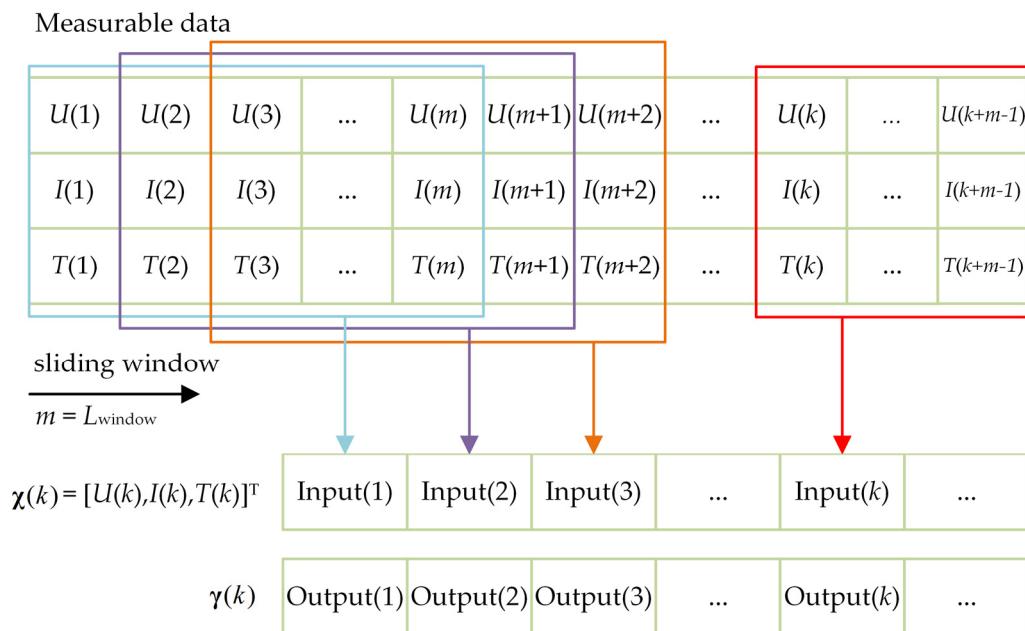
$$r_f = 1 + \sum_{i=0}^{n-1} 2 \cdot (k_s - 1) \cdot b^i = 1 + 2 \cdot (k_s - 1) \cdot \frac{b^n - 1}{b - 1}. \quad (5)$$

The minimum number of residual blocks required to fully cover the input sequence is given by:

$$n = \left\lceil \log_b \left( \frac{(l-1) \cdot (b-1)}{(k_s-1) \cdot 2} + 1 \right) \right\rceil \quad (6)$$

where  $l$  denotes the length of the input sequence, and  $[x]$  represents rounding  $x$  up to the nearest integer. Computing this value ensures that the TCN is sufficiently deep to cover all the information in the input sequence.

The network employs a sliding window method to process the input data. Figure 4 presents the input and output data format schematic for the SOC estimation method. At time  $k$ , the vector  $\chi(k) = [U(k), I(k), T(k)]^T$  denotes a vector consisting of the terminal voltage, current, and surface temperature of the battery, with  $\gamma(k)$  indicating a vector of the estimated SOC.



**Figure 4.** Input and output data format schematic for the SOC estimation method.

Additionally, the min–max normalization method is employed for data normalization, aiming to enhance the training adaptability of deep learning networks, alleviate neuron saturation, and improve algorithmic estimation accuracy. The formula for min–max normalization is:

$$\begin{aligned} \mathbf{x}'(k) &= [\mathbf{x}(k) - \mathbf{x}_{\min}] / (\mathbf{x}_{\max} - \mathbf{x}_{\min}), \\ \mathbf{y}'(k) &= [\mathbf{y}(k) - \mathbf{y}_{\min}] / (\mathbf{y}_{\max} - \mathbf{y}_{\min}). \end{aligned} \quad (7)$$

At time  $k$ , the network's overall input and output are mathematically expressed as:

$$\begin{aligned} \mathbf{x}(k) &= [\mathbf{x}'(k - L_{\text{window}} + 1), \mathbf{x}'(k - L_{\text{window}} + 2), \dots, \mathbf{x}'(k)], \\ \mathbf{y}(k) &= \mathbf{y}'(k), \end{aligned} \quad (8)$$

where  $L_{\text{window}}$  denotes the window size of the input data.

After obtaining the network's predicted output value, denoted as  $\hat{\mathbf{y}}'(k)$ , the inverse normalization is performed, and the estimated SOC is given by:

$$SOC(k) = \hat{\mathbf{y}}'(k) \cdot (\mathbf{y}_{\max} - \mathbf{y}_{\min}) + \mathbf{y}_{\min} \quad (9)$$

Moreover, the network training will employ the MSE loss function. Given a total of  $n$  training data samples, the loss function can be represented as follows:

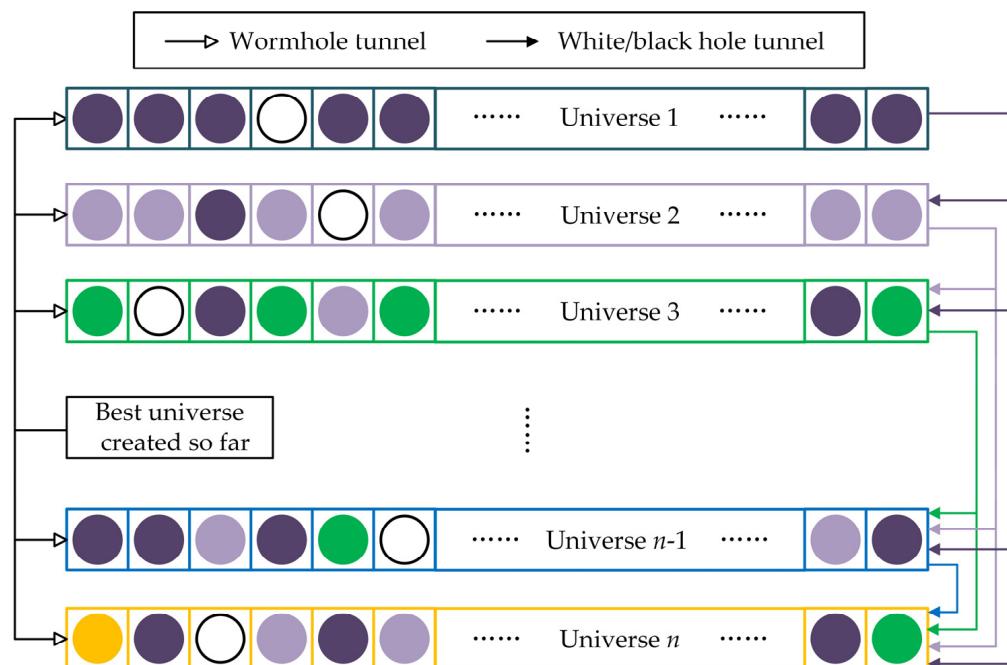
$$L_{\text{MSE}} = \frac{1}{n} \sum_{i=1}^n (\mathbf{y}'(k)_i - \hat{\mathbf{y}}'(k)_i)^2 \quad (10)$$

#### 2.4. MVO Method for Optimization of TCN Hyperparameters

Mirjalili et al. [39] proposed a multi-verse optimizer derived from the principles of the multiverse theory. This algorithm simulates the transfer of objects through white-hole/black-hole tunnels to search for the best solution. The MVO algorithm comprises exploration and exploitation phases, where white holes repel objects during higher cosmic expansion rates, while black holes attract and absorb objects during lower rates. The algorithm introduces “white hole/black hole tunnels” connecting universes for object transfer. In the exploitation phase, wormholes connect universes to the optimal one, allowing for potential object transfer. Wormhole generation probability is independent of the cosmic expansion rate. The algorithm leverages the trend of objects from high to low inflation rates

in the random generation process, facilitating object migration and progressive exploration of the search space toward optimal positions based on cosmological principles.

Figure 5 depicts the schematic diagram of the MVO algorithm. The inflation rate of the  $n^{\text{th}}$  universe is represented by  $I(U_n)$ , as shown in Figure 5, where  $I(U_1) > I(U_2) > \dots > I(U_n)$ . In the establishment of tunnels connecting two universes, a universe characterized by a higher inflation rate is more inclined to accommodate a white hole, whereas a universe with a lower inflation rate is more likely to host a black hole. The transfer of objects from a universe with a high inflation rate to one with a low inflation rate is highly likely, thereby contributing to the overall average inflation rate of the universe during successive iterations.



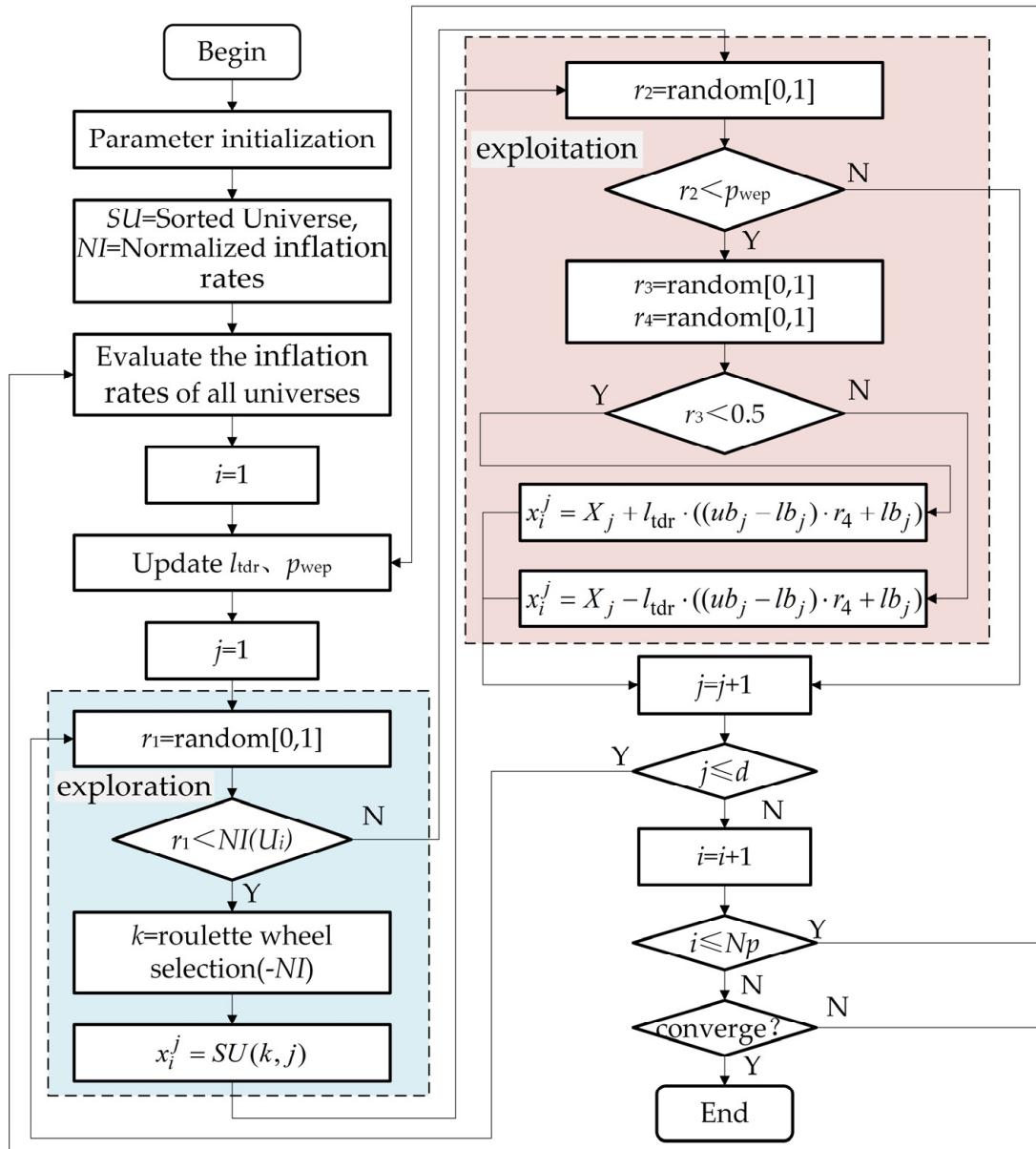
**Figure 5.** Schematic diagram of the MVO algorithm.

As depicted in Figure 5, the objects are symbolized by the white dots that are transported via wormholes. The generation of these wormholes does not take into account the inflation rate of the universe, thereby presenting optimization opportunities for each individual universe. The utilization of wormholes is exclusively limited to connecting the currently optimal individual universes in order to achieve optimization in the search space. By employing white hole/black hole tunnels and wormholes, the MVO algorithm facilitates the continuous transfer of objects between other universes and the current optimal universe, thereby iterating toward optimization. In this process, individuals in each universe symbolize feasible solutions to the optimization problem, where objects in the universe denote design variables, and the inflation rate of the universe acts as the fitness value for the objective function.

Figure 6 illustrates the flowchart of the MVO algorithm, where  $U$  represents the randomly initialized universes,  $d$  denotes the number of design variables, and  $N_p$  signifies the population size. In the exploration phase, mathematical modeling is employed to study the inter-universe exchange of objects through white-/black-hole tunnels. The algorithm incorporates a roulette wheel selection mechanism, where universes are ranked based on their inflation rates in each iteration, and a universe containing a white hole is chosen using this selection method. Let  $x_i^j$  denote the  $j^{\text{th}}$  design variable in the  $i^{\text{th}}$  universe, where  $U_i$  denotes the  $i^{\text{th}}$  universe and  $NI(i)$  indicates its normalized inflation rate. Let  $r_1$  be a

uniformly distributed random number between 0 and 1, and let  $x_k^j$  represent the  $j^{\text{th}}$  design variable in the  $k^{\text{th}}$  universe selected through a roulette wheel mechanism.  $x_i^j$  is given by:

$$x_i^j = \begin{cases} x_k^j & r_1 < NI(U_i) \\ x_i^j & r_1 \geq NI(U_i) \end{cases}. \quad (11)$$



**Figure 6.** Flowchart of the MVO algorithm.

During the exploitation phase, any universe has the potential to generate a wormhole, thereby establishing a connection with the currently optimal universe and facilitating the exchange of objects. Let  $X_j$  denote the current optimal value of the  $j^{\text{th}}$  design variable,  $l_{\text{tdr}}$  represent the rate of travel distance,  $p_{\text{wep}}$  be the probability of wormhole existence in the

universe, and  $u_{bj}$  and  $l_{bj}$  indicate the upper and lower bounds for the  $j^{\text{th}}$  design variable. Additionally,  $r_2$ ,  $r_3$ , and  $r_4$  are random numbers within the range  $[0, 1]$ .  $x_i^j$  is updated by:

$$x_i^j = \begin{cases} \begin{cases} X_j + l_{\text{tdr}} \cdot ((ub_j - lb_j) \cdot r_4 + lb_j) & r_3 < 0.5 \\ X_j - l_{\text{tdr}} \cdot ((ub_j - lb_j) \cdot r_4 + lb_j) & r_3 \geq 0.5 \end{cases} & r_2 < p_{\text{wep}} \\ x_i^j & r_2 \geq p_{\text{wep}} \end{cases}. \quad (12)$$

The current iteration number is denoted as  $N_c$ , the maximum iteration number as  $N_{\text{epoch}}$ , and  $p_{\text{wep,min}}$  and  $p_{\text{wep,max}}$  denote the minimum and maximum values of  $p_{\text{wep}}$ , respectively. Additionally,  $p$  represents the exploitation rate.  $p_{\text{wep}}$  and  $l_{\text{tdr}}$  are given by:

$$\begin{cases} p_{\text{wep}} = p_{\text{wep,min}} + N_c \cdot (p_{\text{wep,max}} - p_{\text{wep,min}}) / N_{\text{epoch}} \\ l_{\text{tdr}} = 1 - N_c^{1/p} / N_{\text{epoch}}^{1/p} \end{cases}. \quad (13)$$

## 2.5. Objective Function

The objective function is of paramount importance in optimizing deep learning networks through MVO. In this investigation, the design variables represent the hyperparameters of the deep learning network, and the output is indicated by  $-R^2$  on the validation set. The objective function subjected to optimization in the MVO process for deep learning networks cannot be directly expressed through a mathematical equation. To construct the objective function, we present the pseudocode outlined in Table 1, which delineates the process. Steps 1 to 4 involve data preprocessing; Step 5 pertains to network model construction; Steps 6 to 12 encompass network training; and Steps 13 to 17 cover the calculation of the objective function value.

**Table 1.** Pseudo-code description of the objective function.

Step	Description
1:	Allocate each element of input $\mathbf{x}$ to respective hyperparameters and convert to required types
2:	Load training and validation set data
3:	Normalize data
4:	Convert data to tensor format and create data iterators
5:	Build deep learning network model
6:	Network Training:
7:	for $i$ in [1: training epochs]
8:	Train network on training set and save weights
9:	Validate network on validation set and calculate $R^2$
10:	End
11:	Compare the $R^2$ values of all networks, and select and record the weights of the network that performs best (highest $R^2$ ) on the validation set.
12:	Remove weights of non-best-performing networks
13:	End
14:	Load weights of the currently best-performing network
15:	Use the deep learning model to predict SOC on the validation set
16:	Denormalize the estimated SOC
17:	Calculate and return $-R^2$

## 3. Results and Discussion

In this study, three distinct deep learning networks, including LSTM, Transformer, and TCN, are employed for estimating the SOC, and their respective performances are thoroughly compared. During the training process, the number of training epochs was set to 150, with a training batch size of 128. The deep learning networks in this section were implemented using the Pytorch framework and trained on an NVIDIA GeForce RTX 3090 (24 GB) GPU running Windows 10 (64-bit) with Python 3.8.

The hyperparameters subject to optimization for various deep learning networks are as follows: (1) LSTM: Sliding window length ( $L_{\text{win\_size}}^{\text{lstm}}$ ), Hidden layer output dimension ( $N_{\text{hidden\_size}}^{\text{lstm}}$ ), Fully connected layer output dimension ( $N_{\text{out}}^{\text{lstm}}$ ), Random dropout rate ( $d_r^{\text{lstm}}$ ), Learning rate ( $l_r^{\text{lstm}}$ ). (2) Transformer: Sliding window length ( $L_{\text{win\_size}}^{\text{trans}}$ ), Number of attention heads ( $N_{\text{head}}^{\text{trans}}$ ), Embedding dimension ( $N_{\text{embed}}^{\text{trans}}$ ), Feedforward network dimension ( $N_{\text{forward}}^{\text{trans}}$ ), Number of encoder layers ( $N_{\text{layer}}^{\text{trans}}$ ), Random dropout rate ( $d_r^{\text{trans}}$ ), Learning rate ( $l_r^{\text{trans}}$ ). (3) TCN: Sliding window length ( $L_{\text{win\_size}}^{\text{tcn}}$ ), Number of hidden layer channels ( $N_{\text{ch1}}^{\text{tcn}}, N_{\text{ch2}}^{\text{tcn}}, N_{\text{ch3}}^{\text{tcn}}$ ), Convolutional kernel size ( $k_s^{\text{tcn}}$ ), Random dropout rate ( $d_r^{\text{tcn}}$ ), and Learning rate ( $l_r^{\text{tcn}}$ ).

### 3.1. Dataset

A publicly accessible dataset graciously provided by the University of Wisconsin-Madison [40] was utilized to estimate the SOC for the Panasonic 18650PF battery with a nominal voltage of 3.6 V. The dataset includes nine operating conditions, covering four fundamental scenarios: aggressive driving (SFTP-US06), high-speed (HWFET), urban road cycling (UDDS), and LA92. Additionally, five mixed conditions—Cycles 1 to 4 and a neural network (NN) condition—are included. Cycles 1 to 4 are randomly formed from US06, HWFET, UDDS, LA92, and NN conditions. The NN condition intricately blends segments of the US06 and LA92 driving cycles, introducing supplementary dynamics beneficial for neural network training. This comprehensive method aims to reveal the comparative performance of LSTM, Transformer, and TCN under diverse real-world conditions.

The NN condition, intentionally designed to encompass dynamic scenarios that are beneficial for neural network training within the dataset, was consistently utilized for training in all subsequent design examples. Training, optimization, and testing were conducted at both fixed and varied ambient temperatures. To facilitate a comprehensive comparison of different deep learning networks, two distinct dataset splitting methods were employed:

(A) The training set includes basic scenarios (US06 + UDDS + LA92 + HWFET) along with the NN condition; the validation set includes mixed scenarios (Cycle 1 + Cycle 2); and the testing set includes mixed scenarios (Cycle 3 + Cycle 4).

(B) The training set includes mixed scenarios (Cycle 1~4 + NN); the validation set includes basic scenarios (LA92 + HWFET); and the testing set includes basic scenarios (US06 + UDDS).

### 3.2. Network Training and Testing at a Fixed Ambient Temperature

This section begins by describing the preliminary tests to evaluate the SOC estimation capabilities of different networks, namely, LSTM, Transformer, and TCN. The focus is on examining their mapping abilities for the nonlinear relationships between LIB voltage, current, surface temperature, and SOC at a fixed ambient temperature of 25 °C. The deep learning networks undergo training, optimization, and testing specifically at this fixed temperature.

The MVO algorithm was employed to optimize the hyperparameters for LSTM, Transformer, and TCN at a fixed temperature of 25 °C using two distinct dataset splitting methods. After optimization using the MVO algorithm, the  $R^2$  values for the validation datasets, obtained through two distinct dataset splitting methods, were very close to 1, indicating the convergence of the algorithm. Furthermore, this signifies a robust fit to the validation set, reflecting a high level of model fitting. Table 2 presents the optimization results of the hyperparameters for different networks using the MVO algorithm at a fixed temperature of 25 °C. Table 3 presents the evaluation metrics results for the test dataset at a fixed temperature of 25 °C. Observations suggest that in dataset splitting method A, the RMSE and MAE for all three networks remained below 1%, with the TCN exhibiting the smallest values for both, measuring 0.6959% and 0.4945%, respectively. The TCN's MAX, although positioned in the middle among these three methods, exhibited a mere 0.5604% deviation from the lowest MAX observed with LSTM. In dataset splitting method

B, the TCN achieved the lowest *RMSE*, *MAE*, and *MAX*, with only the TCN's *RMSE* being below 1%, measuring 0.9160%. Additionally, its *MAE* and *MAX* were recorded at 0.7937% and 2.8759%, respectively. These results validate that at a constant ambient temperature, the proposed MVO-TCN method surpasses the other two methods in achieving optimal performance in SOC estimation.

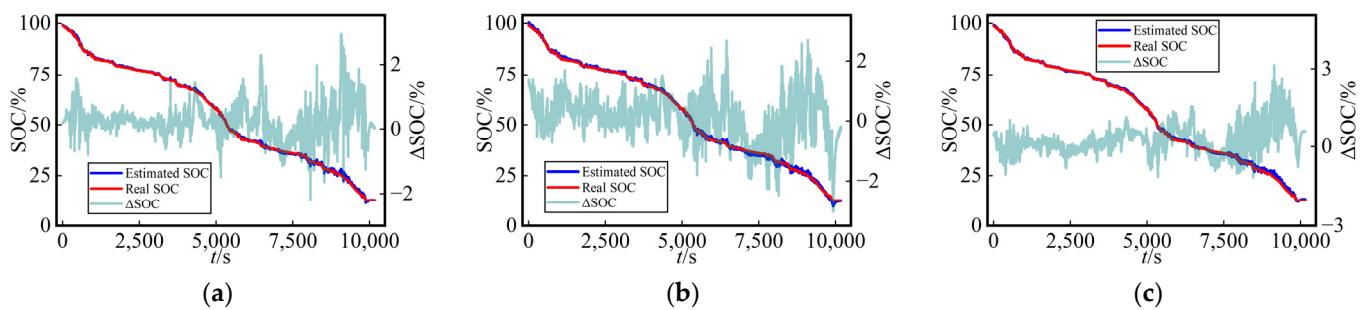
**Table 2.** Optimization results of the hyperparameters for different networks using the MVO algorithm at a fixed temperature of 25 °C.

Network	Result for dataset splitting method A					
LSTM	$L_{\text{win\_size}}^{\text{lstm}}$ 73	$N_{\text{hidden\_size}}^{\text{lstm}}$ 143	$N_{\text{out}}^{\text{lstm}}$ 121	$d_r^{\text{lstm}}$ 0	$I_r^{\text{lstm}}$ $5.3129 \times 10^{-3}$	
Transformer	$L_{\text{win\_size}}^{\text{trans}}$ 65	$N_{\text{head}}^{\text{trans}}$ 4	$N_{\text{embed}}^{\text{trans}}$ 44	$N_{\text{forward}}^{\text{trans}}$ 124	$N_{\text{layer}}^{\text{trans}}$ 1	$d_r^{\text{trans}}$ 0.0178
TCN	$L_{\text{win\_size}}^{\text{tcn}}$ 92	$N_{\text{ch1}}^{\text{tcn}}$ 96	$N_{\text{ch2}}^{\text{tcn}}$ 120	$N_{\text{ch3}}^{\text{tcn}}$ 52	$k_s^{\text{tcn}}$ 10	$d_r^{\text{tcn}}$ 0.0488
Network	Result for dataset splitting method B					
LSTM	$L_{\text{win\_size}}^{\text{lstm}}$ 96	$N_{\text{hidden\_size}}^{\text{lstm}}$ 100	$N_{\text{out}}^{\text{lstm}}$ 122	$d_r^{\text{lstm}}$ 0.0186	$I_r^{\text{lstm}}$ $3.2398 \times 10^{-3}$	
Transformer	$L_{\text{win\_size}}^{\text{trans}}$ 99	$N_{\text{head}}^{\text{trans}}$ 4	$N_{\text{embed}}^{\text{trans}}$ 44	$N_{\text{forward}}^{\text{trans}}$ 149	$N_{\text{layer}}^{\text{trans}}$ 4	$d_r^{\text{trans}}$ 0
TCN	$L_{\text{win\_size}}^{\text{tcn}}$ 96	$N_{\text{ch1}}^{\text{tcn}}$ 177	$N_{\text{ch2}}^{\text{tcn}}$ 110	$N_{\text{ch3}}^{\text{tcn}}$ 114	$k_s^{\text{tcn}}$ 8	$d_r^{\text{tcn}}$ 0
						$I_r^{\text{tcn}}$ $4.5545 \times 10^{-3}$

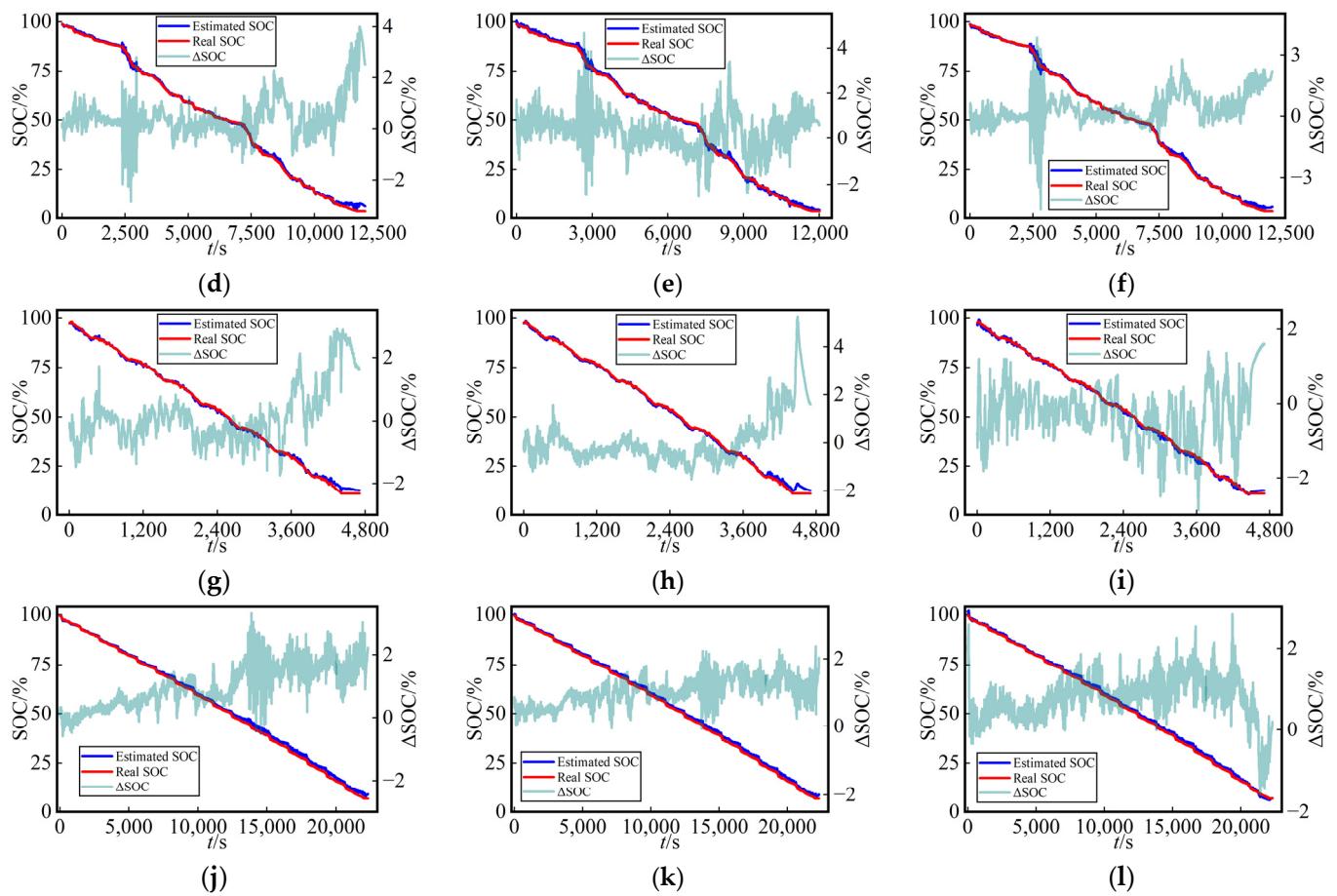
**Table 3.** Evaluation metrics results for the test dataset at the fixed temperature of 25 °C.

Network	Dataset Splitting Method A				Dataset Splitting Method B			
	RMSE (%)	MAE (%)	MAX (%)	R <sup>2</sup>	RMSE (%)	MAE (%)	MAX (%)	R <sup>2</sup>
LSTM	0.7814	0.5178	4.0052	0.9996	1.1654	0.9620	3.3061	0.9995
Transformer	0.7730	0.6046	4.6585	0.9994	1.0849	0.9693	5.2219	0.9995
TCN	0.6959	0.4945	4.5656	0.9996	0.9160	0.7937	2.8759	0.9995

For a more straightforward assessment of the SOC estimation performance among the three methods in the test dataset, Figure 7 presents the curves of SOC estimation values and their errors ( $\Delta$ SOC, calculated as the estimated value minus the true value) for the test dataset using three different networks at a fixed temperature (25 °C) with various dataset split methods. The SOC estimation curves and true SOC curves demonstrated a remarkably high level of consistency for the three methods employed, providing compelling evidence for the viability of this work.



**Figure 7. Cont.**



**Figure 7.** Curves of SOC estimation values and their errors for the test dataset using three different networks at a fixed temperature ( $25\text{ }^{\circ}\text{C}$ ) with various dataset split methods: (a) LSTM, (b) Transformer, (c) TCN at Cycle 3 with dataset split method A; (d) LSTM, (e) Transformer, (f) TCN at Cycle 4 with dataset split method A; (g) LSTM, (h) Transformer, (i) TCN at US06 with dataset split method B; (j) LSTM, (k) Transformer, (l) TCN at UDDS with dataset split method B.

### 3.3. Network Training and Testing at Various Ambient Temperatures

To further compare the mapping capabilities of the three distinct deep learning networks for characterizing the nonlinear dependencies among voltage, current, surface temperature, and SOC in lithium-ion batteries, simultaneous training, optimization, and testing were conducted at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ). Because the dataset covers various operating conditions at multiple ambient temperatures, the nonlinear relationship among the voltage, current, surface temperature, and SOC becomes more complex. This complexity necessitates a more comprehensive evaluation of the different deep learning networks' performances. The performances of these three methods were compared using both aforementioned dataset splitting methods at various ambient temperatures.

#### 3.3.1. Network Training and Testing Using the Dataset Splitting Method A

The MVO algorithm was adopted to optimize the hyperparameters for three networks, namely, LSTM, Transformer, and TCN, using dataset splitting method A. The dataset encompasses nine dynamic operating conditions across various ambient temperatures. Table 4 presents the optimization results of the hyperparameters for the different networks using the MVO algorithm at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) using dataset splitting method A. After optimization using the MVO algorithm, the  $R^2$  values for the validation dataset (Cycle1 + Cycle2) were close to 1 across all three deep learning networks,

indicating that the MVO algorithm effectively converged. Additionally, the results reveal that all three networks demonstrate exceptional fitting of the intricate relationships among voltage, current, surface temperature, and SOC for the validation dataset through the data from diverse dynamic working conditions at various temperatures.

**Table 4.** Optimization results of hyperparameters for different networks using the MVO algorithm at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset splitting method A.

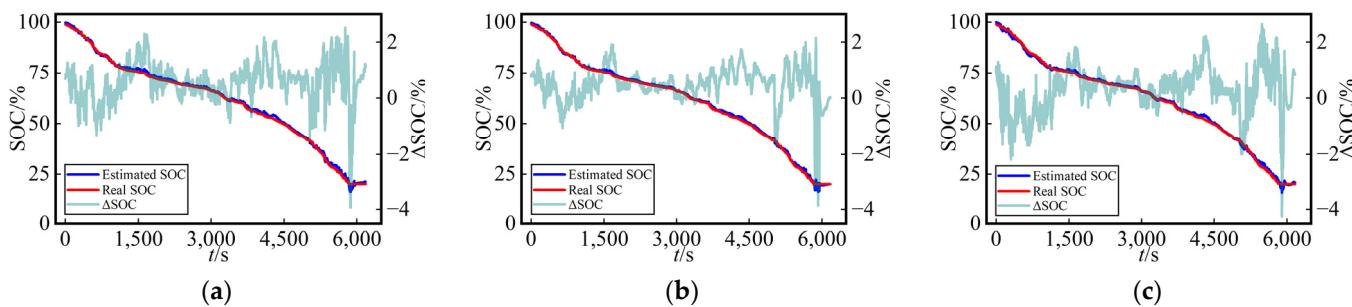
Network	Result						
LSTM	$L_{\text{win\_size}}^{\text{lstm}}$ 73	$N_{\text{hidden\_size}}^{\text{lstm}}$ 196	$N_{\text{out}}^{\text{lstm}}$ 166	$d_r^{\text{lstm}}$ 0.1037	$l_r^{\text{lstm}}$ $1.0449 \times 10^{-3}$		
Transformer	$L_{\text{win\_size}}^{\text{trans}}$ 75	$N_{\text{head}}^{\text{trans}}$ 2	$N_{\text{embed}}^{\text{trans}}$ 124	$N_{\text{forward}}^{\text{trans}}$ 198	$N_{\text{layer}}^{\text{trans}}$ 3	$d_r^{\text{trans}}$ 0.0767	$l_r^{\text{trans}}$ $1 \times 10^{-4}$
TCN	$L_{\text{win\_size}}^{\text{tcn}}$ 76	$N_{\text{ch1}}^{\text{tcn}}$ 105	$N_{\text{ch2}}^{\text{tcn}}$ 53	$N_{\text{ch3}}^{\text{tcn}}$ 50	$k_s^{\text{tcn}}$ 9	$d_r^{\text{tcn}}$ 0.1197	$l_r^{\text{tcn}}$ $1.2022 \times 10^{-3}$

Table 5 presents the evaluation metrics results for the test dataset (Cycle 3+ Cycle 4) at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) using dataset splitting method A. The *RMSE* and *MAE* values below 1% were obtained for the SOC estimation on the test dataset across all three methods. Notably, the proposed MVO-TCN achieved the smallest *RMSE* and *MAE*, measuring 0.6790% and 0.4904%, respectively. Among these methods, the TCN exhibited a moderate *MAX* value of SOC estimation at 4.2517%, but it was only 0.3944% larger than the lowest *MAX* observed with Transformer. In summary, the proposed MVO-TCN demonstrates the optimal performance in SOC estimation among the three methods considered in this study.

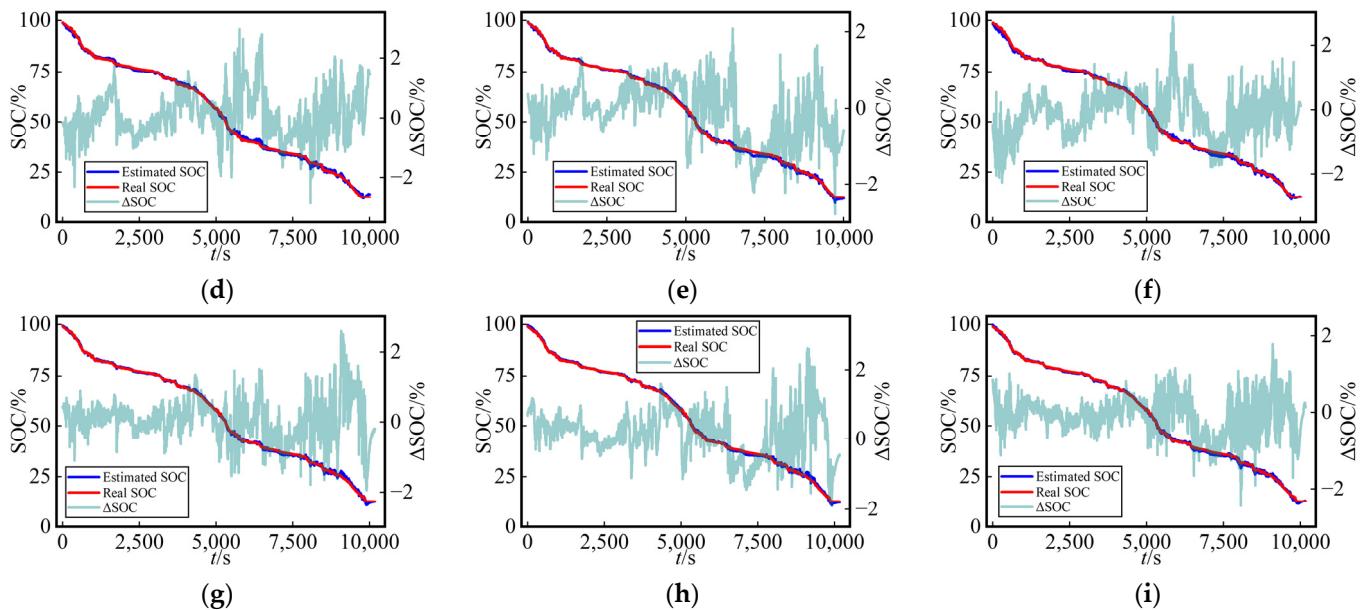
**Table 5.** Evaluation metrics results for the test dataset (Cycle 3 + Cycle 4) at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset splitting method A.

Network	RMSE (%)	MAE (%)	MAX (%)	R <sup>2</sup>
LSTM	0.7766	0.5752	5.6381	0.9989
Transformer	0.7388	0.5806	3.8573	0.9991
TCN	0.6790	0.4904	4.2517	0.9991

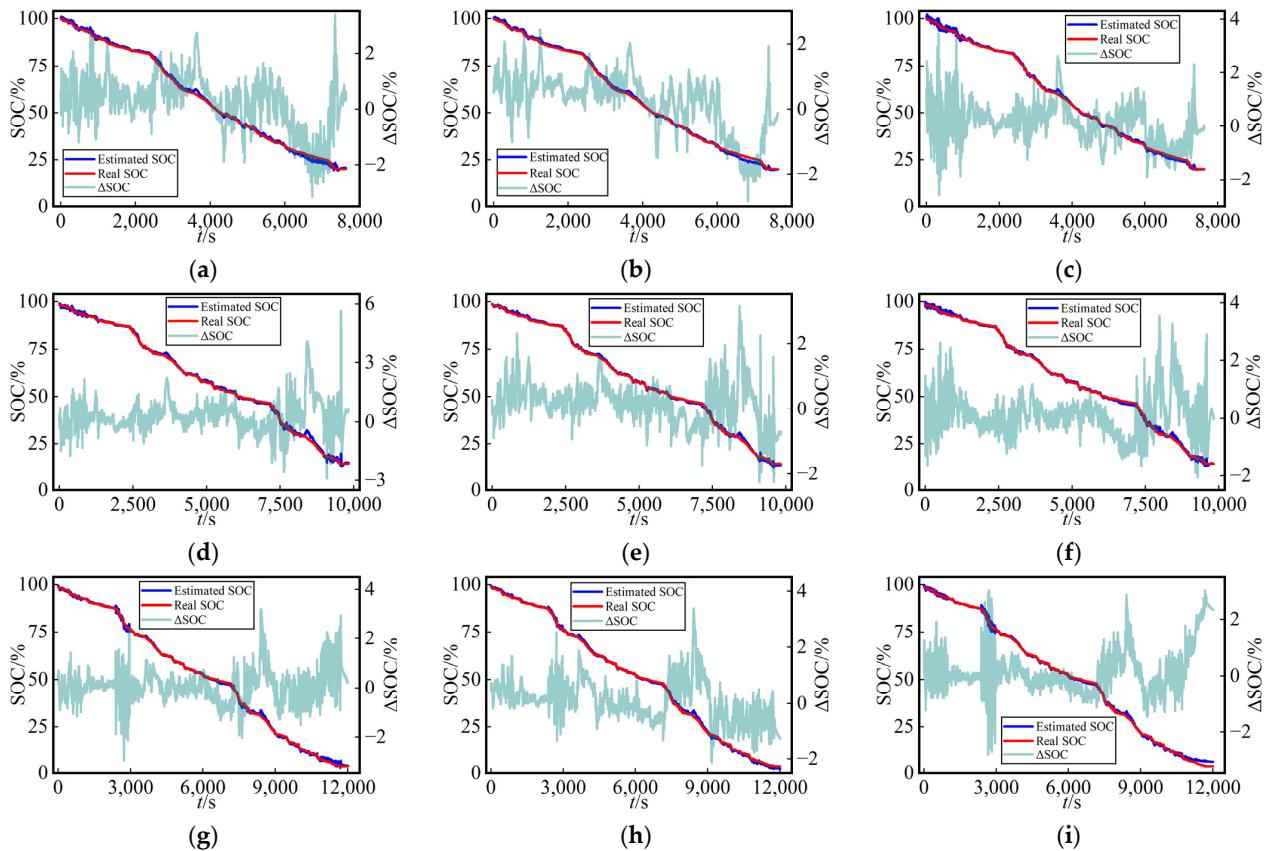
Similarly, for a comparison that is more easily interpretable in terms of the battery's SOC estimation at various ambient temperatures in the test dataset, as depicted in Figures 8 and 9, the following figure presents the curves of estimated SOC values, actual values, and their errors ( $\Delta\text{SOC}$ , calculated as the estimated value minus the true value) for the three networks at different ambient temperatures in test Cycles 3 and 4. The three methods yielded accuracy curves representing the estimated SOC. The error curves directly reflected the SOC estimation performance for different networks under various working conditions.



**Figure 8. Cont.**



**Figure 8.** Curves of SOC estimation values and their errors for Cycle 3 in the test dataset using three different networks at various temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset split method A: (a) LSTM, (b) Transformer, and (c) TCN at  $0\text{ }^{\circ}\text{C}$ ; (d) LSTM, (e) Transformer, and (f) TCN at  $10\text{ }^{\circ}\text{C}$ ; (g) LSTM, (h) Transformer, and (i) TCN at  $25\text{ }^{\circ}\text{C}$ .



**Figure 9.** Curves of SOC estimation values and their errors for Cycle 4 in the test dataset using three different networks at various temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset split method A: (a) LSTM, (b) Transformer, and (c) TCN at  $0\text{ }^{\circ}\text{C}$ ; (d) LSTM, (e) Transformer, and (f) TCN at  $10\text{ }^{\circ}\text{C}$ ; (g) LSTM, (h) Transformer, and (i) TCN at  $25\text{ }^{\circ}\text{C}$ .

### 3.3.2. Network Training and Testing Using Dataset Splitting Method B

To study the robustness of the proposed method, we conducted training, optimization, and testing under various working conditions at various temperatures using dataset splitting method B. Table 6 presents the optimization results of the hyperparameters for different networks using the MVO algorithm at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) using dataset splitting method B. After optimization with the MVO algorithm, the  $R^2$  values for all three methods on the validation dataset (LA92 + HWFET) were also very close to 1. This suggests that MVO converged, and all three methods exhibit a good fit for the SOC estimation on the validation dataset.

**Table 6.** Optimization results of the hyperparameters for different networks using the MVO algorithm at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset splitting method B.

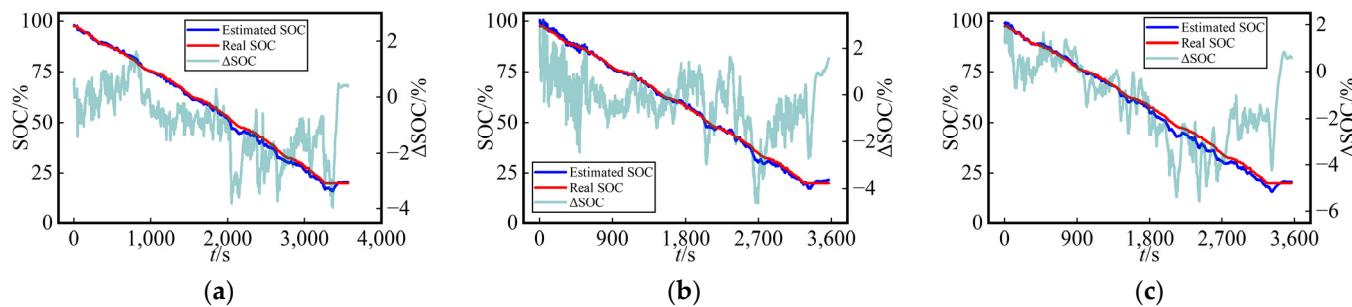
Network	Result						
LSTM	$L_{\text{win\_size}}^{\text{lstm}}$ 97	$N_{\text{hidden\_size}}^{\text{lstm}}$ 180	$N_{\text{out}}^{\text{lstm}}$ 181	$d_r^{\text{lstm}}$ 0	$l_r^{\text{lstm}}$ $4.3920 \times 10^{-3}$	-	-
Transformer	$L_{\text{win\_size}}^{\text{trans}}$ 94	$N_{\text{head}}^{\text{trans}}$ 4	$N_{\text{embed}}^{\text{trans}}$ 12	$N_{\text{forward}}^{\text{trans}}$ 43	$N_{\text{layer}}^{\text{trans}}$ 1	$d_r^{\text{trans}}$ 0.0009	$l_r^{\text{trans}}$ $5.1923 \times 10^{-3}$
TCN	$L_{\text{win\_size}}^{\text{tcn}}$ 100	$N_{\text{ch1}}^{\text{tcn}}$ 120	$N_{\text{ch2}}^{\text{tcn}}$ 151	$N_{\text{ch3}}^{\text{tcn}}$ 151	$k_s^{\text{tcn}}$ 8	$d_r^{\text{tcn}}$ 0	$l_r^{\text{tcn}}$ $1 \times 10^{-4}$

As depicted in Table 7, among the three methods using splitting method B, the proposed MVO-TCN in this study consistently exhibited the smallest RMSE and MAE for SOC estimation across various temperatures and working conditions for the test dataset (US06 + UDDS), measuring 1.2388% and 0.9909%, respectively. Despite having a moderate MAX, it was only 1.0222% larger than the smallest MAX observed with Transformer. Overall, compared with the other two methods, this method exhibits the most optimal performance in SOC estimation.

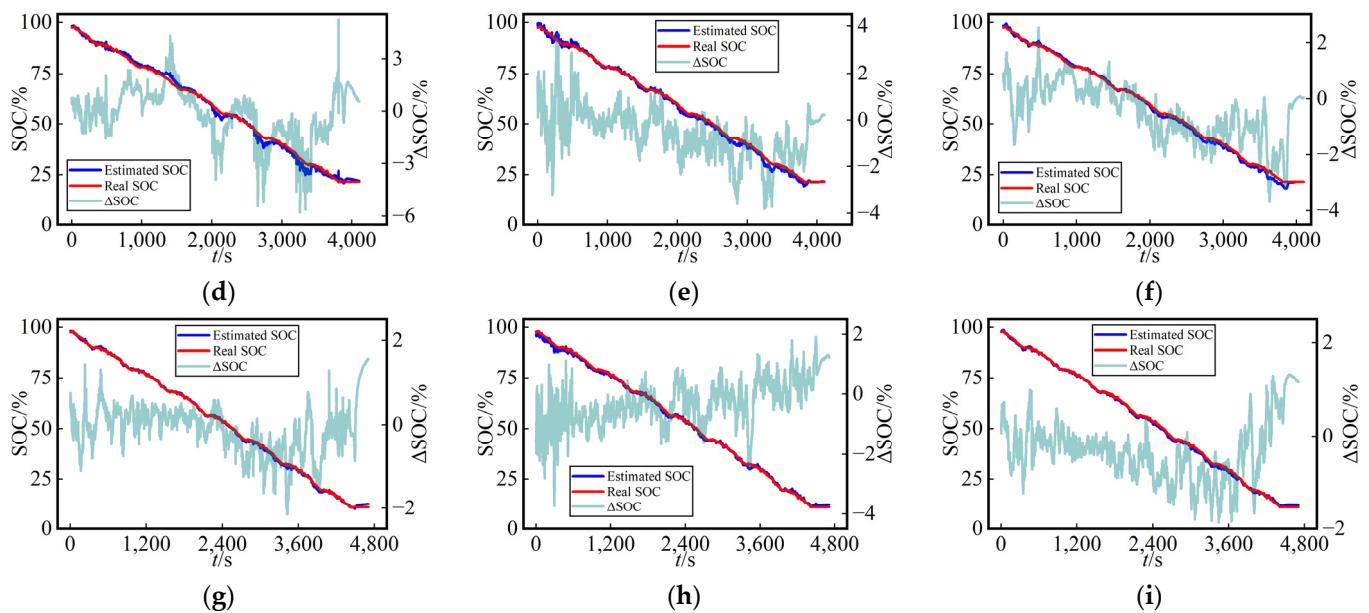
**Table 7.** Evaluation metrics results for the test dataset (US06 + UDDS) at various ambient temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset splitting method A.

Network	RMSE (%)	MAE (%)	MAX (%)	$R^2$
LSTM	1.5005	1.2642	7.1627	0.9993
Transformer	1.4205	1.1979	4.6569	0.9992
TCN	1.2388	0.9909	5.6791	0.9992

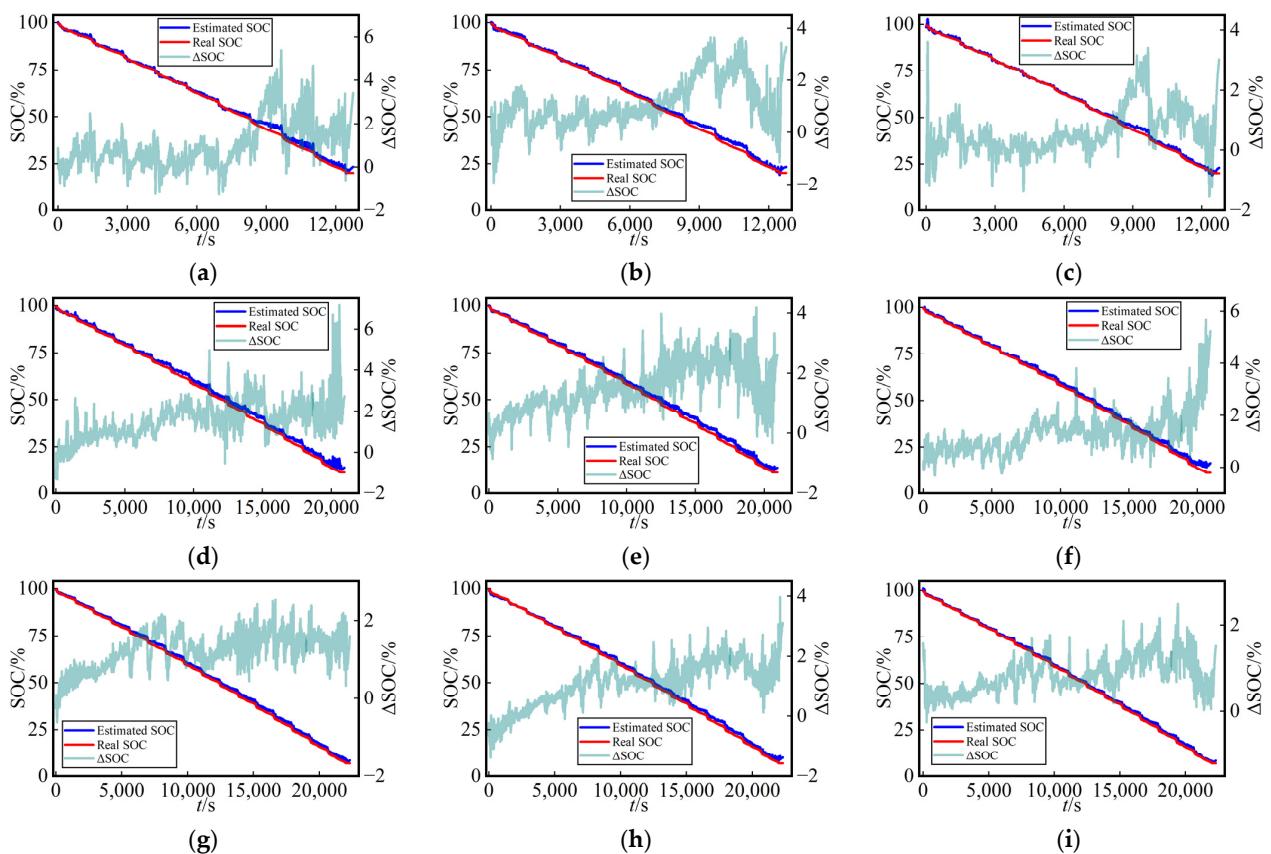
As depicted in Figures 10 and 11, we noted that the estimated SOC curves and true SOC curves were nearly identical, indicating that all three networks exhibited high precision in the SOC estimation using dataset splitting method A. Additionally, the figures provide a visual representation of the error variations for each network across different temperatures and working conditions. This suggests that the proposed MVO-TCN algorithm in this study possesses a high level of robustness.



**Figure 10. Cont.**



**Figure 10.** Curves of SOC estimation values and their errors for the US06 in the test dataset using three different networks at various temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset split method B: (a) LSTM, (b) Transformer, and (c) TCN at  $0\text{ }^{\circ}\text{C}$ ; (d) LSTM, (e) Transformer, and (f) TCN at  $10\text{ }^{\circ}\text{C}$ ; (g) LSTM, (h) Transformer, and (i) TCN at  $25\text{ }^{\circ}\text{C}$ .



**Figure 11.** Curves of SOC estimation values and their errors for the UDDS in the test dataset using three different networks at various temperatures ( $0\text{ }^{\circ}\text{C}$ ,  $10\text{ }^{\circ}\text{C}$ , and  $25\text{ }^{\circ}\text{C}$ ) with dataset split method B: (a) LSTM, (b) Transformer, and (c) TCN at  $0\text{ }^{\circ}\text{C}$ ; (d) LSTM, (e) Transformer, and (f) TCN at  $10\text{ }^{\circ}\text{C}$ ; (g) LSTM, (h) Transformer, and (i) TCN at  $25\text{ }^{\circ}\text{C}$ .

#### 4. Conclusions

In summary, this study introduces a novel SOC estimation method for lithium-ion batteries. The integration of the TCN with MVO exhibits high efficiency and effectiveness at optimizing the TCN's hyperparameters to achieve better precision in SOC prediction. The comprehensive evaluation, conducted under various conditions, demonstrates the robustness and generalization capabilities of the proposed MVO-TCN framework compared to conventional LSTM and Transformer methods. The inclusion of diverse measurable parameters, such as voltage, current, and temperature, enhances the adaptability of the method across dynamic working conditions. By considering dynamic operating conditions, the proposed method accurately estimates SOC, aligning closely with the actual performance of electric vehicles. This work eliminates the need for complex electrochemical or circuit modeling. Moreover, the SOC estimation process is simplified by relying on historical measurement data. This developed method, featuring a memory-efficient TCN, shows practical applicability for battery management systems, providing valuable support for technological advancement. This work not only improves SOC accuracy but also contributes to the development of battery management and energy storage technologies.

**Author Contributions:** Conceptualization, Y.L. and G.L.; methodology, Y.L.; software, Y.L. and W.D.; validation, Y.L. and W.D.; formal analysis, Y.L.; investigation, Y.L. and W.D.; resources, G.L.; data curation, Y.L. and G.L.; writing—original draft preparation, Y.L.; writing—review and editing, Y.L. and G.L.; visualization, Y.L. and W.D.; supervision, G.L.; project administration, G.L.; funding acquisition, G.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Guangdong Key Areas R&D Project Power Battery System Testing and Evaluation Technology Research, grant number: 2019B090908003.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

#### References

- Tian, J.P.; Chen, C.; Shen, W.X.; Sun, F.C.; Xiong, R. Deep Learning Framework for Lithium-ion Battery State of Charge Estimation: Recent Advances and Future Perspectives. *Energy Storage Mater.* **2023**, *61*, 102883. [[CrossRef](#)]
- Akhil, I.; Kumar, N.; Kumar, A.; Sharma, A.; Kaushik, M. State of charge estimation using different machine learning techniques. *J. Inf. Optim. Sci.* **2022**, *43*, 543–547. [[CrossRef](#)]
- Oyewole, I.; Chehade, A.; Kim, Y. A controllable deep transfer learning network with multiple domain adaptation for battery state-of-charge estimation. *Appl. Energy* **2022**, *312*, 118726. [[CrossRef](#)]
- Yu, K.; Wang, H.; Mao, L.; He, Q.B.; Wu, Q. IC Curve-Based Lithium-Ion Battery SOC Estimation at High Rate Charging Current. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 3160554. [[CrossRef](#)]
- Mishra, P.P.; Garg, M.; Mendoza, S.; Liu, J.; Rahn, C.D.; Fathy, H.K. How Does Model Reduction Affect Lithium-Ion Battery State of Charge Estimation Errors? Theory and Experiments. *J. Electrochem. Soc.* **2017**, *164*, A237–A251. [[CrossRef](#)]
- Yu, Q.Q.; Xiong, R.; Lin, C.; Shen, W.X.; Deng, J.J. Lithium-Ion Battery Parameters and State-of-Charge Joint Estimation Based on H-Infinity and Unscented Kalman Filters. *IEEE Trans. Veh. Technol.* **2017**, *66*, 8693–8701. [[CrossRef](#)]
- Ye, M.; Guo, H.; Xiong, R.; Yu, Q.Q. A double-scale and adaptive particle filter-based online parameter and state of charge estimation method for lithium-ion batteries. *Energy* **2018**, *144*, 789–799. [[CrossRef](#)]
- Liu, D.T.; Li, L.; Song, Y.C.; Wu, L.F.; Peng, Y. Hybrid state of charge estimation for lithium-ion battery under dynamic operating conditions. *Int. J. Electr. Power Energy Syst.* **2019**, *110*, 48–61. [[CrossRef](#)]
- Kim, W.; Lee, P.Y.; Kim, J.; Kim, K.S. A Robust State of Charge Estimation Approach Based on Nonlinear Battery Cell Model for Lithium-Ion Batteries in Electric Vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 5638–5647. [[CrossRef](#)]
- Cui, Z.H.; Wang, L.C.; Li, Q.; Wang, K. A comprehensive review on the state of charge estimation for lithium-ion battery based on neural network. *Int. J. Energ. Res.* **2022**, *46*, 5423–5440. [[CrossRef](#)]
- Zhang, R.F.; Xia, B.Z.; Li, B.H.; Cao, L.B.; Lai, Y.Z.; Zheng, W.W.; Wang, H.W.; Wang, W. State of the Art of Lithium-Ion Battery SOC Estimation for Electrical Vehicles. *Energies* **2018**, *11*, 1820. [[CrossRef](#)]
- Hannan, M.A.; Lipu, M.S.H.; Hussain, A.; Mohamed, A. A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations. *Renew. Sustain. Energy Rev.* **2017**, *78*, 834–854. [[CrossRef](#)]
- Zheng, Y.J.; Ouyang, M.G.; Han, X.B.; Lu, L.G.; Li, J.Q. Investigating the error sources of the online state of charge estimation methods for lithium-ion batteries in electric vehicles. *J. Power Sources* **2018**, *377*, 161–188. [[CrossRef](#)]

14. Yu, Q.Q.; Huang, Y.K.; Tang, A.H.; Wang, C.; Shen, W.X. OCV-SOC-Temperature Relationship Construction and State of Charge Estimation for a Series-Parallel Lithium-Ion Battery Pack. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 6362–6371. [[CrossRef](#)]
15. Xiao, R.X.; Hu, Y.W.; Zhang, W.; Chen, Z.H. A novel approach to estimate the state of charge for lithium-ion battery under different temperatures incorporating open circuit voltage online identification. *J. Energy Storage* **2023**, *67*, 107509. [[CrossRef](#)]
16. Yang, C.F.; Wang, X.Y.; Fang, Q.H.; Dai, H.F.; Cao, Y.Q.; Wei, X.Z. An online SOC and capacity estimation method for aged lithium-ion battery pack considering cell inconsistency. *J. Energy Storage* **2020**, *29*, 101250. [[CrossRef](#)]
17. Mohammadi, F. Lithium-ion battery State-of-Charge estimation based on an improved Coulomb-Counting algorithm and uncertainty evaluation. *J. Energy Storage* **2022**, *48*, 104061. [[CrossRef](#)]
18. Tian, Y.; Li, D.; Tian, J.; Xia, B. State of charge estimation of lithium-ion batteries using an optimal adaptive gain nonlinear observer. *Electrochim. Acta* **2017**, *225*, 225–234. [[CrossRef](#)]
19. Shrivastava, P.; Soon, T.K.; Idris, M.Y.I.B.; Mekhilef, S.; Adnan, S.B.R.S. Combined state of charge and state of energy estimation of lithium-ion battery using dual forgetting factor-based adaptive extended Kalman filter for electric vehicle applications. *IEEE Trans. Veh. Technol.* **2021**, *70*, 1200–1215. [[CrossRef](#)]
20. Sun, D.; Yu, X.; Wang, C.; Zhang, C.; Huang, R.; Zhou, Q.; Amietszajew, T.; Bhagat, R. State of charge estimation for lithium-ion battery based on an Intelligent Adaptive Extended Kalman Filter with improved noise estimator. *Energy* **2021**, *214*, 119025. [[CrossRef](#)]
21. Sakile, R.; Sinha, U.K. Lithium-Ion Battery State of Charge Estimation Using a New Extended Nonlinear State Observer. *Adv. Theory Simul.* **2022**, *5*, 2100552. [[CrossRef](#)]
22. Yang, F.; Zhang, S.; Li, W.; Miao, Q. State-of-charge estimation of lithium-ion batteries using LSTM and UKF. *Energy* **2020**, *201*, 117664. [[CrossRef](#)]
23. Hannan, M.A.; How, D.N.T.; Lipu, M.S.H.; Mansor, M.; Ker, P.J.; Dong, Z.Y.; Sahari, K.S.M.; Tiong, S.K.; Muttaqi, K.M.; Mahlia, T.M.I.; et al. Deep learning approach towards accurate state of charge estimation for lithium-ion batteries using self-supervised transformer model. *Sci. Rep.* **2021**, *11*, 19541. [[CrossRef](#)] [[PubMed](#)]
24. Yang, B.; Wang, Y.; Zhan, Y. Lithium Battery State-of-Charge Estimation Based on a Bayesian Optimization Bidirectional Long Short-Term Memory Neural Network. *Energies* **2022**, *15*, 4670. [[CrossRef](#)]
25. Yan, X.; Zhou, G.; Wang, W.; Zhou, P.; He, Z. A Hybrid Data-Driven Method for State-of-Charge Estimation of Lithium-Ion Batteries. *IEEE Sens. J.* **2022**, *22*, 16263–16275. [[CrossRef](#)]
26. Fan, T.-E.; Liu, S.-M.; Tang, X.; Qu, B. Simultaneously estimating two battery states by combining a long short-term memory network with an adaptive unscented Kalman filter. *J. Energy Storage* **2022**, *50*, 104553. [[CrossRef](#)]
27. Ren, X.; Liu, S.; Yu, X.; Dong, X. A method for state-of-charge estimation of lithium-ion batteries based on PSO-LSTM. *Energy* **2021**, *234*, 121236. [[CrossRef](#)]
28. Zhang, D.W.; Zhong, C.; Xu, P.J.; Tian, Y.Y. Deep Learning in the State of Charge Estimation for Li-Ion Batteries of Electric Vehicles: A Review. *Machines* **2022**, *10*, 912. [[CrossRef](#)]
29. Shen, H.; Zhou, X.; Wang, Z.; Wang, J. State of charge estimation for lithium-ion battery using Transformer with immersion and invariance adaptive observer. *J. Energy Storage* **2022**, *45*, 103768. [[CrossRef](#)]
30. Liu, Y.F.; He, Y.J.; Bian, H.D.; Guo, W.; Zhang, X.Y. A review of lithium-ion battery state of charge estimation based on deep learning: Directions for improvement and future trends. *J. Energy Storage* **2022**, *52*, 104664. [[CrossRef](#)]
31. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv* **2018**, arXiv:1803.01271.
32. Qin, Y.; Shi, Y.; Hao, X.; Liu, J. Microblog Text Emotion Classification Algorithm Based on TCN-BiGRU and Dual Attention. *Information* **2023**, *14*, 90. [[CrossRef](#)]
33. Lin, J.; van Wijngaarden, A.J.D.; Wang, K.C.; Smith, M.C. Speech Enhancement Using Multi-Stage Self-Attentive Temporal Convolutional Networks. *IEEE-Acm. Trans. Audio Speech Lang. Process* **2021**, *29*, 3440–3450. [[CrossRef](#)]
34. Zhang, W.; Zhang, Y.H.; She, Y.J.; Shao, J. Stereo Feature Enhancement and Temporal Information Extraction Network for Automatic Music Transcription. *IEEE Signal Process Lett.* **2021**, *28*, 1500–1504. [[CrossRef](#)]
35. Fan, J.; Zhang, K.; Huang, Y.P.; Zhu, Y.F.; Chen, B.P. Parallel spatio-temporal attention-based TCN for multivariate time series prediction. *Neural Comput. Appl.* **2023**, *35*, 13109–13118. [[CrossRef](#)]
36. Lin, W.H.; Wang, P.; Chao, K.M.; Lin, H.C.; Yang, Z.Y.; Lai, Y.H. Wind Power Forecasting with Deep Learning Networks: Time-Series Forecasting. *Appl. Sci.* **2021**, *11*, 335. [[CrossRef](#)]
37. Mo, J.Y.; Wang, R.; Cao, M.D.; Yang, K.; Yang, X.; Zhang, T. A hybrid temporal convolutional network and Prophet model for power load forecasting. *Complex Intell. Syst.* **2023**, *9*, 4249–4261. [[CrossRef](#)]
38. Wan, R.Z.; Mei, S.P.; Wang, J.; Liu, M.; Yang, F. Multivariate Temporal Convolutional Network: A Deep Neural Networks Approach for Multivariate Time Series Forecasting. *Electronics* **2019**, *8*, 876. [[CrossRef](#)]
39. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [[CrossRef](#)]
40. Phillip, K. Panasonic 18650PF Li-ion Battery Data. *Mendeley Data* **2018**. [[CrossRef](#)]