IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 12, 2020, accepted May 13, 2020, date of publication May 21, 2020, date of current version June 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2996225

Lithium-Ion Batteries State of Charge Prediction of Electric Vehicles Using RNNs-CNNs Neural Networks

FEN ZHAO[1], YINGUO LI[1,2], XINHENG WANG[3], LING BAI[1,4], AND TAILIN LIU[5]

[1]School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[2]School of Automation, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[3]School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[4]Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4L8, Canada
[5]The 32nd Institute of China Electronics Technology Corporation, Shanghai 201808, China

Corresponding author: Xinheng Wang (xhwangcqupt@gmail.com)

This work was supported by the Ph.D. Training Program of the Chongqing University of Posts and Telecommunications under Grant BYJS201916.

ABSTRACT In order to achieve the safe and efficient energy use in the electric vehicle, the continuous and accurate monitoring of lithium-ion batteries (LIBs) has become a long-standing research hot spot. However, existing researches of LIBs state of charge (SOC) prediction are at the cost of unrefined vector representation and inadequate feature extraction, which have been unable to meet prediction requirements of LIBs SOC. Complementarily, in this study, a deep learning-based SOC prediction model is proposed to ensure reliable vector representation and sufficient feature extraction. In order to improve battery data representation, a recursive neural networks (RNNs)-based method is proposed. Then, aiming to fully extract feature information, a multi-channel extended convolutional neural networks (CNNs)-based method, which is fed with the well-trained vector representation, is proposed to accurately predict LIBs SOC. Based on the reliable vector representation and sufficient feature extraction, the proposed method can provide improved SOC prediction performance. Merits of the proposed method are verified using simulation test, which shows that the proposed method gives improved prediction performance of 4.3% and 11.3% compared with recurrent neural networks and Ah counting method, respectively.

INDEX TERMS Lithium-ion batteries, state of charge, recursive neural networks, convolutional neural networks.

I. INTRODUCTION

The past years have witnessed the significant development of electric vehicle industries, which have played a significant role in improving the natural environment [1]. Lithium-ion batteries (LIBs) are widely preferred in electric vehicles [2], [3]. Battery management system (BMS) [4] can improve energy efficiency and prolong the remaining service life of batteries by supervising and controlling LIBs' operation appropriately [5], which ensures the normal work of the battery within a safe range. The working mechanism of LIBs in electric vehicles is shown in FIGURE 1. It illustrates the simplified structure of LIBs, which are composed of the positive material of the battery, the cathode material of the battery, electrolyte, diaphragm and battery case. It also describes the charging and discharging process of LIBs, which clearly marks out the movement direction of lithium ions and electrons.

The key function of BMS is to monitor the state of batteries and ensure the safe operation of batteries. It is worth mentioning that state of charge (SOC) [6], [7] is considered as battery's energy gauge which is one of the uppermost states of BMS [4]. SOC reflects the ratio of the battery's utilizable capacity to its nominal capacity and can be written as follows: $SOC = \frac{C_r}{C_I}$, where C_r denotes the remaining capacity of the battery, and C_I denotes the capacity of the battery when it is discharged at a constant current I. Accurate SOC estimation is essential, and techniques for battery SOC estimation could be achieved by various methods such as Ampere-hour (Ah) counting method, open-circuit voltage (OCV) method, extended kalman filter (EKF) method and deep neural networks (DNNs) method.

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

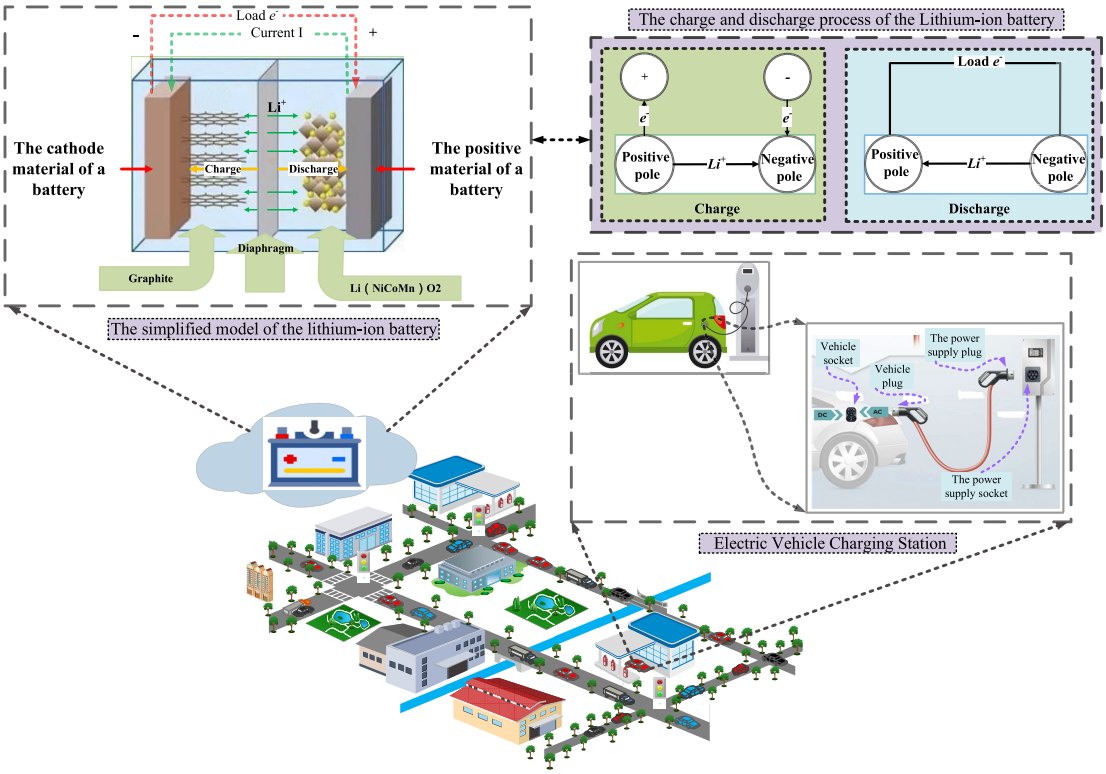


FIGURE 1. Lithium-ion batteries model applied to electric vehicles.

Ah counting, also called Coulomb counting method [8], is one of the most straightforward approaches of SOC prediction, in which SOC is determined by integrating charging and discharging currents during operating period. However, Ah method suffers from poor precision owing to accumulation of the current sensor error [9]. Another rational method of SOC estimation is OCV method [10] where voltage of the battery is directly correlated with charging status of the battery. However, this method needs to disconnect loads, and the open-circuit voltage can only be measured after the battery is stationary for a long time. Another technique for battery SOC estimation is EKF method, which is usually used in non-linear systems. EKF method uses partial derivatives and first order Taylor series expansion to linearize the battery model. This method can often produce good battery SOC estimation, but it causes problem such as bad application in real-time, as well. Thus, it raises urgencies of exploring other alternatives regarding SOC prediction of LIBs.

In order to address these above-mentioned impediments, a simple and effective optimization technique based on deep neural networks (DNNs) is proposed. DNNs have been successfully applied to various perceptual domains, including computer vision and natural language processing. Moreover, DNNs-based techniques are showing impressive performance in new domains such as medicine and finance. Success in non-perceptual domains suggests that DNNs-based techniques could be successfully used to develop BMS so as to monitor SOC of LIBs [11], [12]. Therefore, the DNNs-based SOC prediction method has been a growing research field in recent years. Among the various fields of DNNs, recursive neural networks (RNNs) perform well in natural language processing, and convolutional neural networks (CNNs) are better than most existing models in image recognition. RNNs store sequential information in hidden memory to properly process, represent, and store information. CNNs extract abstract feature information like points, lines, and faces of the input data, and preserve the relationship between the input data.

In recent years, many researchers have conducted to extract features by combining RNNs and CNNs. LIBs SOC prediction problem also deals with the correlation between multivariate variables. Most classical methods only use some features and only model single feature information for LIBs SOC prediction. Since RNNs model battery data and map battery information into separable space and CNNs are used to remove noise and to extract feature information between multivariate variables, RNNs-CNNs model is chosen to predict LIBs SOC.

Motivated by the aforementioned observations, in this paper, a RNNs-CNNs neural network combining RNNs and CNNs is proposed to predict LIBs SOC. Due to the inadequate vector representation of battery data, an adaptive vector representation model based on RNNs is proposed. Then, in order to fully extract feature information, the well-trained

vectors are integrated into multi-channel extended CNNs to realize accurate SOC prediction.

Main contributions of this paper are stated below:

- To improve battery data representation, LIBs data is fully represented by establishing the RNNs-based data representation model. The model provides the refined vector representation of battery data for subsequent SOC prediction, which ensures validity and stability of SOC prediction.
- To fully extract feature information, a multi-channel extended CNNs-based model, which is fed with the well-trained battery vector representation, is proposed to extract the unknown feature information hidden in battery vector.
- Extensive simulation using LIBs subject to dynamic current profiles, dynamic voltage profiles and change in temperature, is conducted. Simulation results demonstrate that, in terms of mean absolute error, our scheme improves 4.3% and 11.3% compared with recurrent neural networks and Ah counting method, respectively.

The rest of this paper is organized as follows: Section II gives an overview of related work. Section III presents the problem and challenges. Section IV analyzes the method of SOC estimation of LIBs. In Section V, the experimental results are presented. Finally, Section VI concludes the paper.

II. RELATED WORK

In this section, the most relevant works with regard to LIBs SOC estimation are briefly presented, and differences between our research and existing researches are identified.

Many researches of battery SOC estimation, each with its own advantages and limitations, were proposed [13]. The most commonly used estimation models include electrochemical mechanism model [14], Ah counting model [15], equivalent circuit model (ECM) [16], neural network model [17] and other SOC estimation models.

A detailed review of electrochemical mechanism models is presented in [18]. Since electrochemical model can analyse the battery mechanism, the model is increasingly popular. However, since battery parameters have relationship with the structure, dimensions and materials of batteries, the model is too complicated to estimate LIBs SOC efficiently. Ah counting method is trivial to be implemented, which is often used in the battery SOC prediction [19]. In practice, due to uncertain disturbances, Ah counting method may induce accumulated estimation errors. In addition, the method requires a known initial value to estimate LIBs SOC adaptively, but it is quite challenging to determine the initial SOC value. ECM has advantages of intuitive structure and low computational requirement [20]; hence, the model has been widely used in the SOC estimation. However, since ECM mimics the input-output behavior by electric components such as resistors and capacitors, it cannot accurately characterize electrochemical dynamics.

Different from the above systems, neural network model can show the nonlinear relationship between SOC and its influencing factors. Many intelligent approaches, such as artificial neural networks (ANNs) model [12], fuzzy logic model [21], RNNs model [22] and support vector regression (SVR) model [23], were proposed to establish SOC estimation models. This type of methods have the powerful function of approximating nonlinear function, which can often provide good SOC estimation [24].

Recently, the idea of battery uncertainties online estimation has been introduced to achieve higher precision in the LIBs SOC estimation. With the development of neural network technologies, a radial basis function (RBF) neural network is provided to solve the estimation uncertainty in the online training [25]. To be more specific, a RBF neural network-based nonlinear observer is designed to estimate LIBs SOC. Then, following Lyapunov stability analysis, it is proved that battery's SOC estimation error is ultimately bounded and the error bound can be arbitrarily small. In [26], a more accurate method using ANNs is investigated for SOC estimation. More specifically, a state observer based on electrochemical principles is designed, and this electrochemical observer is simulated using ANNs. The result from undertaking this research provided more accurate SOC estimation of real-time applications. In summary, compared with other kinds of models, neural network-based models are the most suitable models of LIBs SOC estimation.

III. PROBLEM DESCRIPTION AND CHALLENGES

In this section, the problem of existing models is presented. And then, the challenges faced by the system are summarized.

A. PROBLEM DESCRIPTION

In the battery control system, SOC is one of the most critical parameters. The estimation of battery remaining capacity is mainly achieved by SOC prediction. SOC plays an important role in managements of battery energy use, which prevents the batteries from overcharging and overdischarging [27], [28]. In addition, the estimation performance of the battery remaining capacity has a significant influence on driving safety of electric vehicles. Therefore, a well-developed model plays a crucial role in accurate SOC prediction. However, existing methods of LIBs SOC prediction are at the cost of unrefined vector representation and inadequate feature extraction, which have been unable to meet prediction requirements of LIBs SOC.

B. CHALLENGES

Aiming to make prediction performance of LIBs SOC achieve a certain level of accuracy, a number of problems need to be solved in the deep learning-based SOC prediction model. That is, how the model implements a refined vector representation of battery data, and how the model realizes sufficient feature extraction for accurate SOC prediction of LIBs. The challenges can be summarized as follows:

- It is difficult to obtain a sufficient vector representation of LIBs information owing to the incomplete expression of the relationship between multiple statuses.

- The CNNs-based model is a promising way to solve the low accuracy of SOC prediction. However, a huge challenge remains on the transmission of feature information within a network-such as only using single channel to extract feature information.

We note that a refined vector representation of battery data is the prerequisite for accurate SOC prediction. Before improving predictive performance, the refinement vector representation of battery data is first realized.

IV. THE METHOD OF STATE OF CHARGE PREDICTION

In this section, a DNNs-based SOC observer is proposed to accurately predict LIBs SOC. More specifically, a data representation model based on RNNs is presented to obtain the fine vector representation. Then, a multi-channel extended CNNs-based model is proposed to extract feature information hidden in vectors.

A. A RECURSIVE NEURAL NETWORKS-BASED VECTOR REPRESENTATION MODEL

The implementation details of data representation model include two parts. Firstly, a data representation model is constructed. Secondly, the vector space model is trained to realize the sufficient vector representation.

1) THE FUNDAMENTALS OF RECURSIVE NEURAL NETWORKS

RNNs have the rich expressive power and deterministic internal dynamics, which provide a computationally attractive alternative. Now the example of RNNs training is considered. Assume that there is a parse tree $(h_2(h_1xy)z)$ where x, y and z show inner vector representations. The representation of a parent node is computed in a bottom up manner. In order to compute the parent node, a neural network containing the weight matrix $W_1 \in O^{n\times n}$ for left children and the weight matrix $W_2 \in O^{n\times n}$ for right children, is used. For every parent node h, a vector representation h_i is assumed. h_1 is computed as follows:

$$h_1 = f(W_1x + W_2y + b) \tag{1}$$

where f is an activation function (e.g., sigmoid or tanh), W_1 is the weight matrix for left children, W_2 is the weight matrix for right children, x and y are child nodes, and b is a bias vector. Next, h_2 can be computed, moving a level up in the hierarchy. The representation of parent node is given as follows:

$$h_2 = f(W_1h_1 + W_2z + b) \tag{2}$$

where z is the child node. The process is continued until the root node is reached.

RNNs are used to parse text into a structured tree. The nodes in the tree represent interdependence of words in the text. That is, larger feature is divided into a number of small features, each of which exists on the basis of their internal correlations.

2) THE VECTOR REPRESENTATION MODEL

In order to obtain the appropriate vector representation of battery data, building the data representation model is an important yet challenging task. The data representation model is expected to remove redundant information to meet the demands of subsequent SOC prediction. At dealing with the structured representations of LIBs data, the compact vector representations are expected to provide, but the compact vector representations are difficult to obtain directly. To address the above-mentioned impediment, a novel RNNs-based data representation model is proposed.

In the RNNs-based vector representation model, battery state values such as temperature, voltage and current are vectorized. The model obtains LIBs state data and then maps $n \times 1$ dimension data into the vector space of $n \times m$ dimension. Suppose that a input $X_{3\times1} = \begin{bmatrix} T \\ V \\ C \end{bmatrix}$ is given. (Note that, T represents the temperature value, V represents the voltage value, and C represents the current value.) Here, the dimension n of the data is equal to 3. Analogously, when the input is temperature and voltage, the dimension n of the data is equal to 2. Hence, a input is $X_{2\times1} = \begin{bmatrix} T \\ V \end{bmatrix}$. RNNs requires the vector representation of the input original data as output. Moreover, the correlation between battery data is saved to the vector space due to the powerful logic analysis advantage of neural networks. When the network inputs are temperature, voltage and current, the network output is $Y_{3\times m} = \begin{bmatrix} t_1, t_2, \ldots, t_i, \ldots, t_m \\ v_1, v_2, \ldots, v_i, \ldots, v_m \\ c_1, c_2, \ldots, c_i, \ldots, c_m \end{bmatrix}$. Here, the range of i is $1 < i \leq m$, t_i represents the ith data vector of temperature, v_i represents the ith data vector of voltage, and c_i represents the ith data vector of current. The process provides the refined vector representation of battery data for later SOC prediction, which ensures correctness and stability of SOC prediction.

Construction steps of the vector representation model are as following:

Step 1: Build the RNNs-based vector representation model in Matlab 2016a.

Step 2: Train the vector representation model by adjusting model parameters to realize the refined vector representation.

3) TRAINING VECTOR SPACE MODEL

The experiment platform is installed on high performance server and operated in Matlab environment. At first, environment and GCC are updated. Next, Java running environment is built. Meanwhile, mathematical processing tools: Numpyscipytheano and Pylearn2, are installed. Then, DNNs kit: Scikit-neural network, Java test tool and some dependence, is installed. Afterwards, the display environment is initialized. Finally, the vector space model is trained.

Training steps of the model are as follows:

Step 1: Input 80% of the collected data into RNNs model.

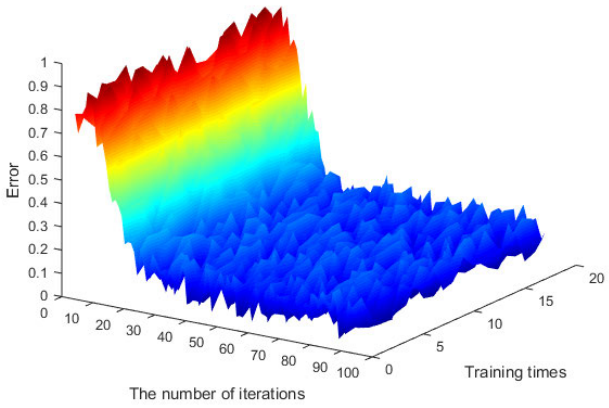


FIGURE 2. The training process of RNNs.

Step 2: Training the RNNs-based vector space model in Java environment.

Step 3: Observe training errors, and adjust model parameters.

Step 4: Terminate training within the error range.

Step 5: Input the remaining 20% data to validate model performance.

Step 6: If the error is not less than 0.02, keep the vector space dimension unchanged, adjust model parameters, and repeat Step 1-Step 5 to continue training.

FIGURE 2 shows the training process of the RNNs-based vector representation model where X-axis represents the number of iterations, Y-axis represents the number of training, and Z-axis represents the training error (namely mean absolute error). Mean absolute error is to calculate the absolute value of the error between the output value and the true value. The training process shows that the initial learning rate is large resulting in faster convergence of the model. When the training error changes more gently and the error is less than 0.02, the training process is completed. It will take 1536 seconds for one-time training process.

LIBs data is mapped by RNNs, and one-dimensional data of voltage and temperature is mapped to the high-dimensional space of 40 dimensions. LIBs data is represented refinedly, and the non-linear evolution of LIBs data is described in detail. FIGURE 3 shows that voltage and temperature of the first batch are respectively intercepted in the 3d vector space. Note that colors in the FIGURE 3, such as green, blue and cyan, represent the state information of LIBs under a SOC value. As shown in FIGURE 3, coordinates of voltage and temperature data are distributed in $-1.0 \sim 1.0$. In the same SOC, voltage and temperature map to different positions in the vector space; in the different SOC, the distribution of voltage and temperature in vector space is fused with each other; in the same coordinate space region, there are voltage and temperature under different SOC. This indicates that SOC of LIBs is affected by state information such as temperature and voltage.

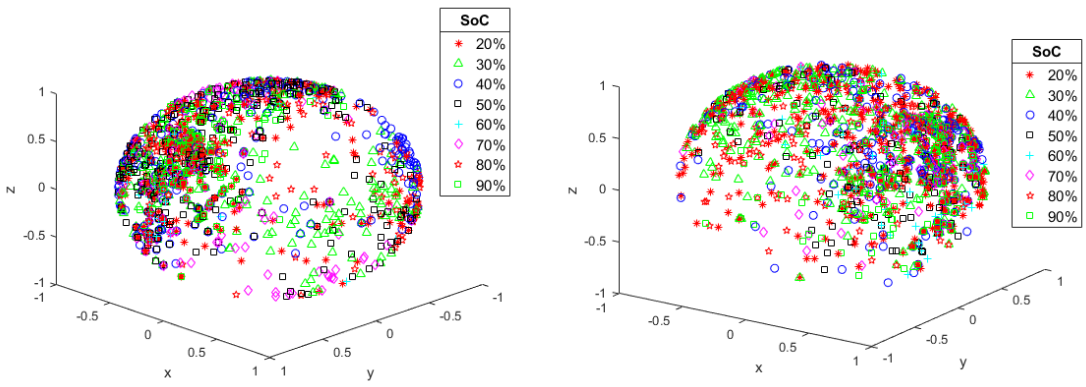
B. THE MULTI-CHANNEL EXTENDED CONVOLUTIONAL NEURAL NETWORKS-BASED FEATURE EXTRACTION MODEL

In this section, the operation steps of the CNNs-based feature extraction model are shown below. Firstly, a feature extraction model is constructed. Secondly, benefiting from the performance characteristic of CNNs, the training operation of the model is performed to accurately predict LIBs SOC.

1) THE INTRODUCTION OF CONVOLUTIONAL NEURAL NETWORKS

CNNs are a kind of feed-forward neural networks, which are equipped with convolution layers interleaved with pooling layers. Convolution layers include a number of computation units, and every unit takes small region vectors as input. The nonlinear activation function be applied to every vector component. This study makes use of Rectified Linear Unit (namely, ReLU). CNNs with ReLU train several times faster than its equivalents with sigmoid.

Pooling layers take convolution layers' output as input. In order to make top layer obtain more abstract and more global feature vectors, pooling layers substantially reduce



(a) Vector space represented by voltage data.

(b) Vector space represented by temperature data.

FIGURE 3. The first batch of LIBs data space mapping.

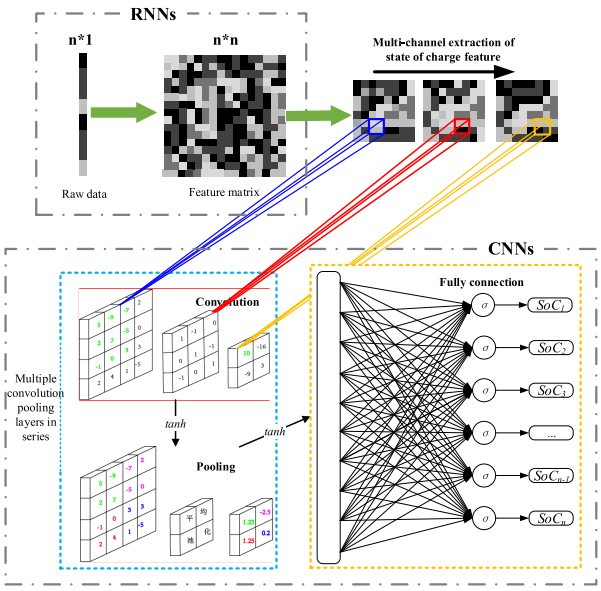


FIGURE 4. Multi-channel extended CNNs-based feature extraction model.

dimensions of input by incorporating neighboring region vectors. Pooling layers are composed of a number of pooling units, and every pooling unit has a connection with small region vectors. Top layer makes use of these feature vectors to make prediction [29], [30], and then the output layer of networks exports results.

Different from most neural networks, CNNs are marked by local connection [31], weight sharing [32] and pooling [33]. Firstly, local connection decreases parameters of hidden layers and simplifies the complexity of the model. Secondly, weight sharing is another striking feature of CNNs, which makes the model obtain translation invariance and enhances the generalization ability. In addition, weight sharing also reduces parameters of hidden layers. Thirdly, pooling is another important concept of CNNs. The goal of the pooling operation is to condense a certain dimension of feature vectors into a feature vector. Existing pooling methods are divided into average pooling method and max pooling method, which respectively calculates mean value and maximum value of each region feature vector [34], [35].

2) THE FEATURE EXTRACTION MODEL

The feature extraction model based on CNNs is composed of three parts: training module, validation module and test module, as shown in Alg. 1. In order to fully extract feature information, the model transmits the well-trained vectors by multiple channels, which realizes the accurate prediction of LIBs SOC, as shown in FIGURE 4.

In the prediction model, the convolution operation uses single-step convolution, and the pooling operation uses average pooling. The implementation details of the prediction model are shown in FIGURE 4. The inner representation of the convolutional operation is given by

$$temp = I \otimes KA \tag{3}$$

Algorithm 1 The Training Algorithm Based on CNNs

Input:
training set TS, validation set VS and test set ES
Output:
mean absolute error MAE
1. set a threshold $\hbar$;
2. input training set TS;
3. **while** mean absolute error $MAE \geq \hbar$ **do**
4. input training set TS;
5. **for** convolution layers in CNNs **do**
6. filters perform convolution operations on the input matrix;
7. obtain feature in different dimensions;
8. **end for**
9. **for** pooling layers in CNNs **do**
10. perform max pooling operations;
11. generate $n \times 1$ dimension of feature vectors $U_{n\times 1}$;
12. **end for**
13. **for** fully connection layer in CNNs **do**
14. $U_{n\times 1}$ is used to make prediction;
15. **end for**
16. **return** training results MAE, as shown in FIGURE 5;
17. **end while**
18. input validation set VS, and then adjust parameters, as shown in FIGURE 6;
19. input test set ES, and then test the prediction capacity of the model, as shown in FIGURE 8.

where $temp$ represents the convolutional operation, I is input of the convolutional layer, and KA is the convolution kernel of the convolutional layer. C represents output of the convolutional layer, with C defined as follows

$$C = f(\eta \times temp + b1) \tag{4}$$

where f is an activation function (eg., tanh, sigmoid or ReLU), η is the learning rate, and $b1$ is a bias vector. $Pool$ represents the pooling operation, and $Pool$ is computed as follows

$$Pool = Pooling(C) \tag{5}$$

Having computed $Pool$, then S can be computed as follows

$$S = f(\eta \times Pool + b2) \tag{6}$$

where S is output of the pooling layer, and $b2$ is a bias vector.

SOC is the output label, the dimension of which is $n \times 1$. $SOC_{n\times 1}$ is defined as follows

$$SOC_{n\times 1} = f(\eta \times WS + b3) \tag{7}$$

where W is a weight vector, S is output of the pooling layer, and $b3$ is a bias vector.

3) TRAINING FEATURE EXTRACTION MODEL

The rich vector representation learned during RNNs training facilitates further training of CNNs. In the training process of CNNs, firstly, the residual is calculated layer by layer, and then the layer error is inversely derived. Next, combined with the learning rate, the weight and the bias of each layer are updated by the derivative of the activation function. Finally, model training is realized.

The specific training process is as follows: firstly, the residual of the output layer is calculated as follows

$$\delta_{n\times 1} = tarSOC_{n\times 1} - SOC_{n\times 1} \tag{8}$$

where, $\delta_{n\times 1}$ is the residual of the output layer, $tarSOC_{n\times 1}$ is the target output (namely, the true measured SOC value), and $SOC_{n\times 1}$ is the actual output of the prediction model. Let the j_{th} submatrix of the l_{th} layer be

$$\delta_j^l = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \tag{9}$$

where, a, b, c and d are elements inside the matrix. Set the size of the pooling area to 2×2, upsample the above matrix, and expand each row and column of the matrix by 1 where the matrix is filled with 0. Obtain the following new matrix

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & a & b & 0 \\ 0 & c & d & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{10}$$

In this paper, average pooling is adopted to average the pooled area (namely 2×2), and the new matrix is expressed as

$$\begin{pmatrix} a/4 & a/4 & b/4 & b/4 \\ a/4 & a/4 & b/4 & b/4 \\ c/4 & c/4 & d/4 & d/4 \\ c/4 & c/4 & d/4 & d/4 \end{pmatrix} \tag{11}$$

These above formulas (9) (10) (11) are upsample processes. After the pooling operation, the formula for calculating the feedback error of the network is described by

$$J^l = \left(\left(\omega^{l+1}\right)^T J^{l+1}\right) \otimes \sigma\left(Input^l\right) \tag{12}$$

where J is the error, ω is a weight vector, σ is an activation function, and $Input^l$ is the l_{th} input. Hence, the network can obtain the first reverse input of convolution. Then, in the backpropagation process, the formula solving the convolutional error is expressed as

$$J^{l-1} = J^l \times rot180(\omega^l) \otimes \sigma'(Input^{l-1}) \tag{13}$$

where, $rot180$ represents rotating the shared matrix ω 180 degrees, σ' represents the derivative of the activation function, and $Input^{l-1}$ represents the $(l-1)_{th}$ input. Finally, the gradient descent algorithm is used to continuously update parameters to obtain the optimal solution of the error function. Gradients of the shared weight ω and the bias term b are calculated by the error function, and then the weight and the bias term are updated. In this way, the error function $J(\omega, b)$ updates parameters as follows

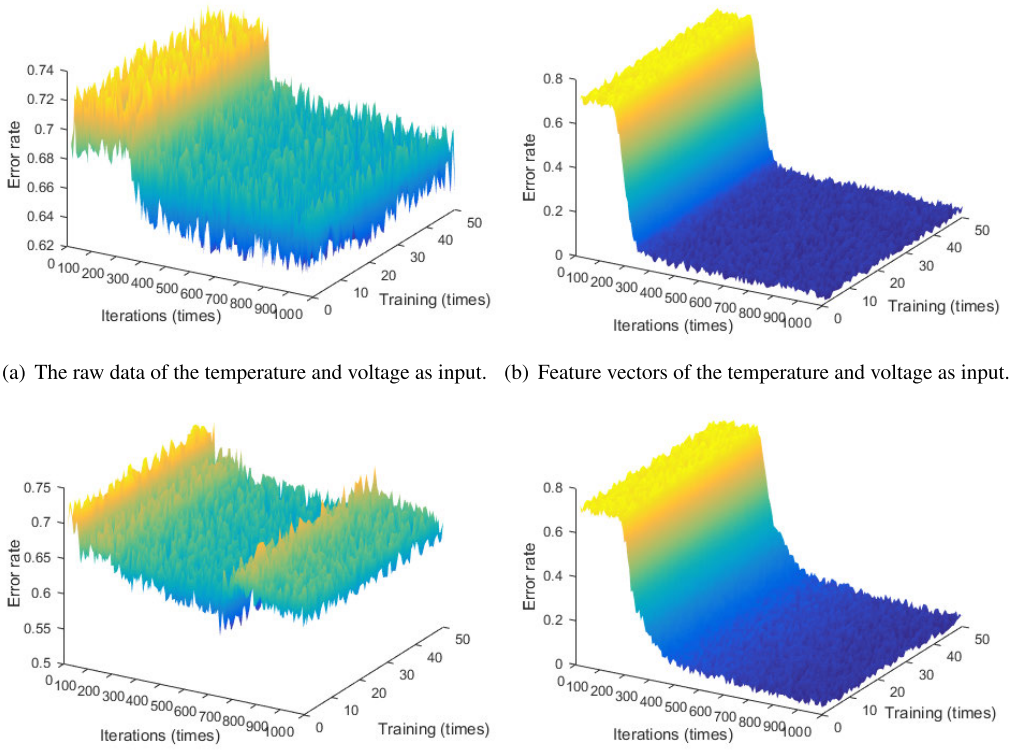
$$\omega_k' = \omega_k - \eta \frac{\partial J(\omega, b)}{\partial \omega_k} \tag{14}$$

$$b_l' = b_l - \eta \frac{\partial J(\omega, b)}{\partial b_l} \tag{15}$$

where ω_k is a weight of the k_{th} neuron, ω_k' is the updated weight of the k_{th} neuron, η is the learning rate, $\frac{\partial J(\omega,b)}{\partial \omega_k}$ means that the error function takes the partial derivative of the weight. Similarly, b_l is a bias of the l_{th} neuron, b_l' is the updated bias of the l_{th} neuron, $\frac{\partial J(\omega,b)}{\partial b_l}$ means that the error function takes the partial derivative of the bias. Equation (14) and Equation (15) represent the process of one-time updated parameters, which require loop iterations until the objective function converges on the optimal value.

In this section, TV indicates that the model takes the raw data of the temperature and the voltage (namely, $X_{2\times 1} = \begin{bmatrix} T \\ V \end{bmatrix}$) as input, TVC indicates that the model takes the raw data of the temperature, the voltage and the current (namely, $X_{3\times 1} = \begin{bmatrix} T \\ V \\ C \end{bmatrix}$) as input, $TV - Vec$ indicates that the model uses vectors of temperature and voltage (namely, $Y_{2\times m} = \begin{bmatrix} t_1, t_2, \ldots, t_i, \ldots, t_m \\ v_1, v_2, \ldots, v_i, \ldots, v_m \end{bmatrix}$) as input, and $TVC - Vec$ indicates that the model uses vectors of temperature, voltage and current (namely, $Y_{3\times m} = \begin{bmatrix} t_1, t_2, \ldots, t_i, \ldots, t_m \\ v_1, v_2, \ldots, v_i, \ldots, v_m \\ c_1, c_2, \ldots, c_i, \ldots, c_m \end{bmatrix}$) as input. According to different types of input data, the multi-channel extended CNN-based model completed 900 iterations, and the same training process was repeated 50 times. The three-dimensional spatial relation diagram of error rate, the number of iterations and training times (namely, epoches) was obtained, as shown in FIGURE 5.

FIGURE 5 (a) and FIGURE 5 (c) are relation diagrams among the error rate, the number of iterations and training times in the training process. Compared with FIGURE 5 (c), FIGURE 5 (a) has smaller error fluctuation, in which the error is lower in most cases. Clearly, increasing current parameters does not improve model performance when raw data is used as input. Moreover, neither of them is strictly monotonous, and the error is relatively high which is all above 0.6. Thus, the model which is trained directly with the raw data cannot meet the effective SOC prediction. Besides, FIGURE 5 (b) and FIGURE 5 (d) are relation diagrams of the error rate, the number of iterations and training times in the training process. In order to obtain feature vectors (namely, $TV - Vec$ and $TVC - Vec$), the raw battery data is trained by RNNs. Then, the learned vectors are integrated into the CNN-based prediction model. From FIGURE 5 (b) and FIGURE 5 (d), the error is less than 0.02. In the case of adding a current parameter, the model needs to be iterated 500 times approximately to converge the optimal value, as shown in FIGURE 5 (d). Why does the excess current parameter



(a) The raw data of the temperature and voltage as input. (b) Feature vectors of the temperature and voltage as input.

(c) The raw data of the temperature, voltage and current as input. (d) Feature vectors of the temperature, voltage and current as input.

FIGURE 5. **The training process of SOC prediction model based on CNNs.**

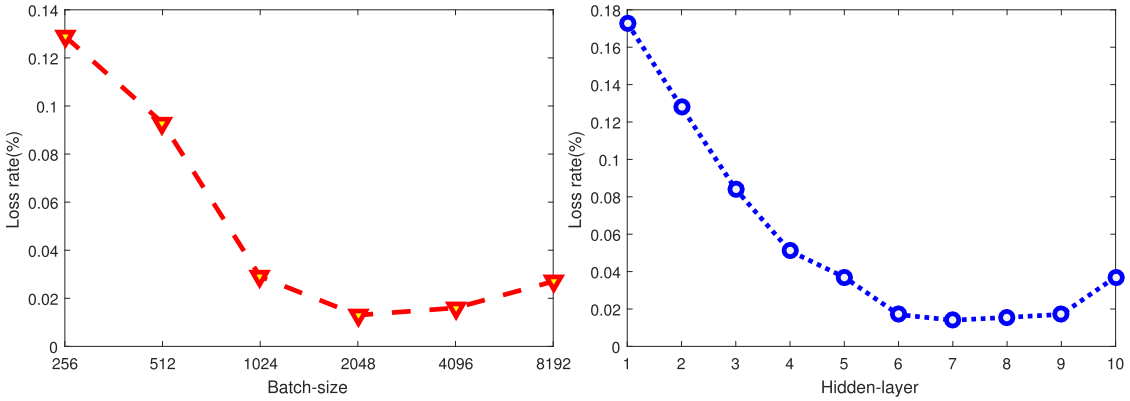
TABLE 1. **Comparison experiment.**

Input data type	TV	TV-Vec	TVC	TVC-Vec
Iterations	900	900	900	900
Training time/s	47071.444	74965.086	55475.659	77594.487
Mean absolute error	0.109	0.019	0.139	0.037

increase the number of iterations? The reason is that adding the current parameter increases computational complexity, which in turn increases computational resource consumption. In addition, the consumption of computing resources affects the logical reasoning ability of DNNs model, which in turn affects the prediction performance of the model to some extent.

In TABLE 1, we note that in the vector space (namely $TV - Vec$ or $TVC - Vec$), mean absolute error is 0.019 or 0.037; nevertheless, in the non-vector space (namely TV or TVC), mean absolute error is 0.109 or 0.139. Based on these observations, we find that the prediction ability in the vector space is obviously stronger than that in the non-vector space. Mean absolute error in the non-vector space is greater than 0.13 (such as 0.139), which is difficult to meet prediction requirements of LIBs SOC. Compared with non-vector space, simulation results demonstrate that vector space improves 12% in prediction performance. In addition, training time of TVC is more than that of TV. Analogously, training time of $TVC - Vec$ is more than that of $TV - Vec$. The reason is that the additional current data adds computational burden of the computer, wastes computational resources and leads to impaired model performance ultimately. Moreover, training time in the vector space is obviously greater than that in the non-vector space. The reason is that the data dimension in vector space is larger, which means the data representation is more complex, so the model needs more time to train. It is worth mentioning that in both vector space and non-vector space, the number of iterations is 900.

The whole data set can be randomly divided into training set (70%), validation set (10%) and testing set(20%). Training set is used to update weights and biases; validation set is able to adjust hyperparameters; testing set is employed to evaluate model performance. In terms of hyper-parameter setting, batch-size and hidden layers of CNNs are optimized manually through 50-fold cross-validation. Keeping other parameters unchanged, CNNs runs 50 epoches setting batch-size to 256, 512, 1024, 2048, 4096 and 8192. The average loss function values of each batch-size during 50 epoches are regarded as the evaluating indicators. FIGURE 6 (a) shows that the loss function value of batch-size=2048 is lower than that of other batch-size; hence batch-size of CNNs to be optimized is set as 2048. Then, keeping batch-size=2048 and remaining parameters unchanged, the setting of hidden layer number is given as 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. FIGURE 6 (b) indicates the best setting of hidden layers is 7 under the condition of batch-



(a) The change of training loss with the increase of batch-size. (b) The change of training loss with the increase of the number of hidden layers.

FIGURE 6. Loss rate (%) of CNNs during 50-fold cross-validation.

size=2048. We find that the depth of hidden layers improves model performance. The reason is that the more hidden layers is, the more feature information the model extracts. However, as the number of hidden layers increases, the model becomes more and more complex, which adds computational burden of the computer, wastes computational resources and leads to impaired model performance ultimately. Note that since the loss function was convergent during 60 epoches, epoches are set as 50. In addition, dropout remains to be 0.1 because there is no overfitting phenomenon in the training process.

V. SIMULATION RESULTS

In this section, simulation is conducted to illustrate validity of the proposed SOC prediction algorithm.

A. DATASET

Dataset collected by relevant scholars is derived from Red Star X1 electric vehicles, which are produced by Do - Fluoride Chemicals Co., Ltd. The electric vehicle uses LIBs as the power battery. Battery specification is 111V, 192Ah, battery capacity is 21.3Kwh, energy density is greater than 120 wh/kg, and maximum mileage is 190Km. 20 cell monomers, which are the same capacity (namely C_I), are connected in series. The temperature and voltage of 20 cell monomers, the total current and the true measured SOC value (that is, twenty temperature values, twenty voltage values, one total current value and one true measured SOC value) are collected in one batch.

In order to collect LIBs SOC in the dataset, firstly, the standard charging method is adopted to charge the lithium battery with constant current, so that SOC of the lithium battery is 100%. Then, the lithium battery is discharged, and the current is taken once every 5 seconds with a high-precision current meter. Finally, SOC is calculated by Ah counting method. The formula for calculating SOC by Ah counting method can be expressed as

$$SOC_{k+1} = SOC_k - \frac{\int_k^{k+5} idt}{C_I} \tag{16}$$

$$= \frac{C_{r_k}}{C_I} - \frac{\int_k^{k+5} idt}{C_I} \tag{17}$$

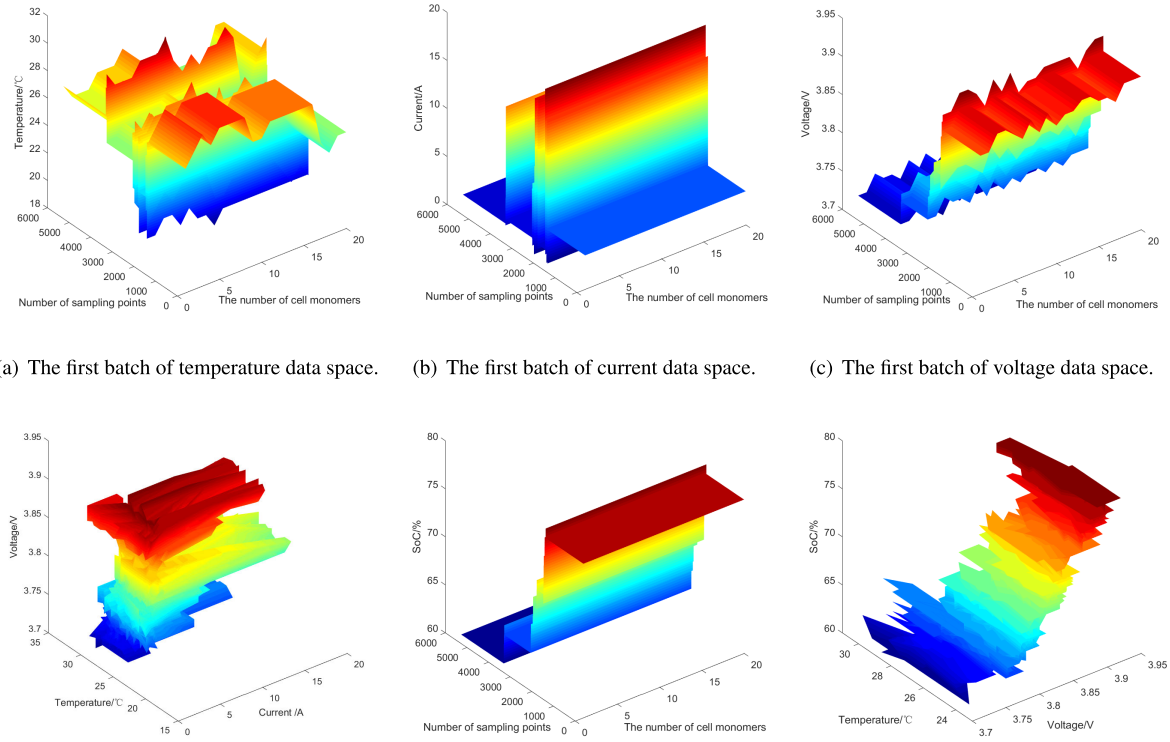
$$= \frac{C_{r_k} - \int_k^{k+5} idt}{C_I} \tag{18}$$

$$= \frac{C_{r_{k+1}}}{C_I} \tag{19}$$

where SOC_{k+1} represents the SOC value at time $k+1$, which is also the SOC value to be calculated; SOC_k represents the SOC value at time k; C_I represents the capacity of the battery when it is discharged at a constant current I; i represents current; $\int_k^{k+5} idt$ is the integral of current i over time t (Note that owing to small fluctuations in the current, it is collected by a high-precision current meter every 5 seconds.); C_{r_k} denotes the remaining capacity of the battery at time k; $C_{r_{k+1}}$ denotes the remaining capacity of the battery at time $k+1$.

The true measured SOC value in the dataset is compared with the predictive SOC value of the proposed model to verify the predictive ability of the proposed model. Note that when cell monomers with the same capacity are connected in series, the voltage increases, but the capacity remains the same (namely C_I). In addition, since the remaining capacity of the battery (namely C_r) can be obtained by Ah counting method, SOC for a LIBs pack can be obtained as well. Sampling period is 15 seconds, sampling time is 24 hours, and total acquisition batches are 60. Thus, $24 \times 3600 \div 15 \times 42 \times 60 = 14515200$ pieces of data were collected, which is regarded as a corpus to make simulation.

Taking the first batch of data as an example, the data space composed of voltage, temperature, current and SOC of 20 cell monomers is shown in FIGURE 7. In FIGURE 7 (a), the temperature of LIBs varies complicated during the collection period, and the temperature of each cell monomer is different. This indicates that the working environment of the electric vehicle is complex, and the temperature change of LIBs is affected by the running state of the electric vehicle. In FIGURE 7 (b), the current appears to break out in a steady state. Due to the small number of sudden changes in the whole



(a) The first batch of temperature data space.

(b) The first batch of current data space.

(c) The first batch of voltage data space.

(d) The first batch of temperature, current and voltage data space.

(e) The first batch of SOC data space.

(f) The first batch of temperature, voltage and SOC data space.

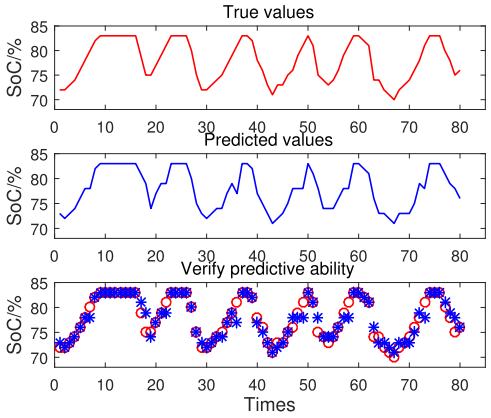
FIGURE 7. State information data space of LIBs.

collection cycle, it could be the high current generated by electric cars when they start. In FIGURE 7 (c), the voltage of LIBs shows a trend of decreasing step by step during the collection period. In addition, the voltage of each cell monomer is inconsistent at the same time. In FIGURE 7 (d), given the spatial relationship between current, temperature and voltage, it can be seen that the relationship between them is complicated. In FIGURE 7 (e), SOC decreases with the increase of collecting time. In FIGURE 7 (f), the lower the voltage is, the less SOC of LIBs is, but the overall is neither in direct proportion nor a smooth surface, thus the temperature has some influence on SOC.

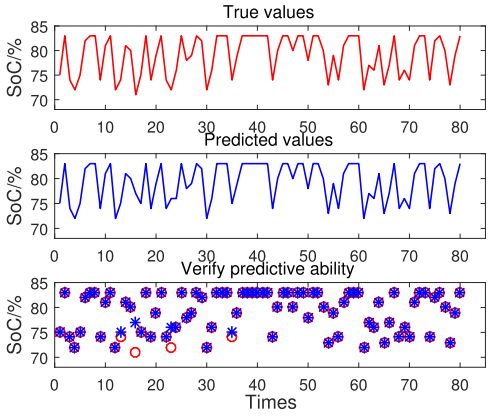
B. PERFORMANCE EVALUATION

The prediction model based on CNNs is applied to LIBs SOC prediction. From above training models, the model with the least error is selected for SOC prediction. In FIGURE 8, the true value is represented by the red curve, and the predicted output value is represented by the blue curve. During the test process, the true value is represented by the red "o", and the predicted output value is represented by the blue "*". Observing the distance between the red "o" and the blue "*" during the test process, the predictive performance of the model can be obtained, as shown in FIGURE 8. FIGURE 8 shows predicted performance of different types of input data. In FIGURE 8 (a), the raw data of the temperature and the voltage is used for model input. Compared with FIGURE 8 (a), FIGURE 8 (c) increases the current input. In FIGURE 8 (b), feature vectors of the temperature and the voltage are used for model input. Compared with FIGURE 8 (b), FIGURE 8 (d) increases current input. As previously mentioned, adding the current parameter will waste computing resources. In addition, compared with FIGURE 8 (a), FIGURE 8 (b) has better predictive performance. Analogously, compared with FIGURE 8 (c), FIGURE 8 (d) has better predictive performance. The reason is that in FIGURE 8 (b) and FIGURE 8 (d), the input of the model is feature vectors which are adequately represented by establishing the RNNs-based vector representation model. That is, the prediction ability in the vector space is obviously stronger than that in the non-vector space.

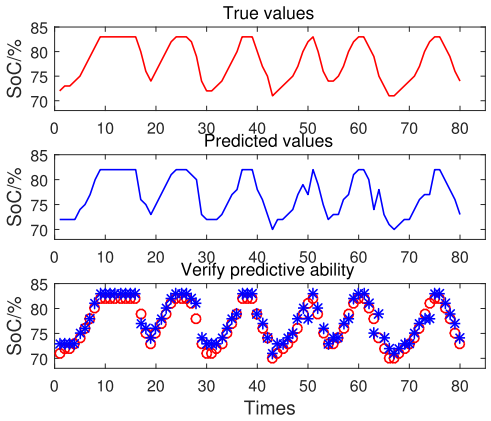
TABLE 2 shows the relationship between mean absolute error and resource allocation in the prediction process. (Note that, in TABLE 2, the input length is represented by N and MAE means mean absolute error.) In the case of using the raw data as input, mean absolute error is relatively high (e.g., 0.135), which cannot meet accuracy requirement of the prediction model. However, under the vector space, mean absolute error is less than 0.036. Analogously, adding current parameters increases memory consumption and overhead of computing resources (namely CPU), which affects model performance. In a nutshell, using feature vectors of the



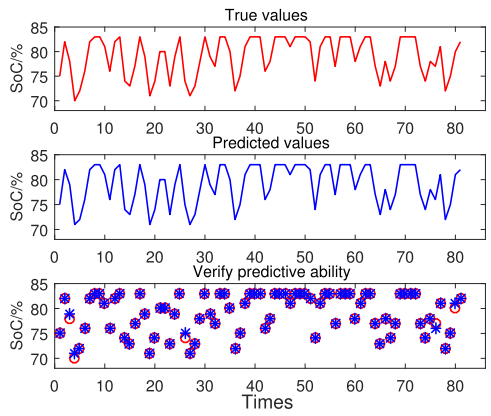
(a) The raw data of the temperature and voltage as input.



(b) Feature vectors of the temperature and voltage as input.



(c) The raw data of the temperature, voltage and current as input.



(d) Feature vectors of the temperature, voltage and current as input.

FIGURE 8. Testing the SOC prediction model based on CNNs.

TABLE 2. Contrast test of SOC prediction model based on Recursive-NNs + CNNs.

	TV			TV-Vec			TVC			TVC-Vec		
N	Memory/MB	CPU/%	MAE	Memory/MB	CPU/%	MAE	Memory/MB	CPU/%	MAE	Memory/MB	CPU/%	MAE
80	527	7.83	0.107	1273	10.32	0.011	835	9.25	0.128	1762	13.48	0.030
120	635	8.18	0.107	1358	11.95	0.014	925	10.14	0.128	1894	15.14	0.029
160	693	9.37	0.096	1427	13.83	0.019	983	11.15	0.135	1998	16.86	0.031
200	748	9.91	0.091	1479	14.97	0.009	1073	11.41	0.122	2035	18.59	0.035
240	836	10.13	0.104	1526	15.72	0.015	1143	12.43	0.121	2274	20.58	0.027
280	897	10.94	0.094	1536	18.19	0.013	1256	12.85	0.125	2368	22.82	0.024

temperature and voltage as input (namely $TV - Vector$) can obtain better prediction performance where the optimal value is 0.009. The simulation results demonstrate that, compared with the non-vector space (namely TV and TVC), the vector space (namely $TV - Vec$ and $TVC - Vec$) improves 12.6% in prediction performance.

C. PERFORMANCE RESULTS OF DIFFERENT ALGORITHMS

TABLE 3 shows mean absolute error of our model in comparison with baseline methods. The first thing to note is that the best-performing recursive-NNs+CNNs method outperforms baseline methods on all dataset, which demonstrates effectiveness of our method. After affirming effectiveness of our model in comparison with Ah counting method [8], OCV method [10], recurrent neural networks [12] and CNNs [36], TABLE 3 summarizes experimental results of different algorithms. (Note that in TABLE 3, t represents time, OCV represents open-circuit voltage, $Recurrent - NNs$ represents recurrent neural networks, and our scheme represents recursive neural networks + CNNs.) When input of DNNs-based methods (such as recurrent neural networks and CNNs) is the raw data (namely TV and TVC), Ah counting method and OCV method outperform DNNs-based methods. However, compared with Ah counting method and OCV method,

TABLE 3. Mean absolute error.

Methods \ Input data	Ct	OCV	TV	TV-Vec	TVC	TVC-Vec
Ah counting method	0.127	/	/	/	/	/
OCV method	/	0.131	/	/	/	/
Recurrent-NNs	/	/	0.138	0.057	0.172	0.082
CNNs	/	/	0.147	0.046	0.193	0.076
Our scheme	/	/	0.091	0.014	0.128	0.029

performance of DNNs-based methods is greatly improved on vector space (namely $TV - Vector$ and $TVC - Vector$). Moreover, our scheme outperforms recurrent neural networks and CNNs on all dataset, which demonstrates effectiveness of our method. Simulation results demonstrate that, in terms of mean absolute error, our proposed method improves 4.3% compared with recurrent neural networks. By comparision with classical methods such as Ah counting method, our proposed method improves 11.3% in terms of mean absolute error.

VI. CONCLUSION AND FUTURE WORK

To improve battery data representation and extract sufficient feature information hidden in battery vector, a novel DNNs-based method is proposed, which aims to improve the prediction performance of LIBs SOC. To be more specific, a RNNs-based data representation model is designed to obtain the appropriate vector representation of battery data, and then a multi-channel extended CNNs-based model, which is fed with the well-trained battery vector, is proposed. By integrating these well-trained vectors into multi-channel extended CNNs, it is proved that this practice can improve battery's SOC prediction performance obviously. Extensive simulation over real dataset is conducted, and our method is compared with some state-of-the-art methods. Simulation results demonstrate that our method improves prediction performance of 4.3% and 11.3% compared with recurrent neural networks and Ah counting method, respectively. With the development of Edge Computing, it is possible for our proposed RNNs-CNNs method to work in a real-time environment. In real-time environment, battery information is collected and sent to the edge platform where high-performance servers are deployed for our algorithm strategy, and then the calculation results are returned to the vehicle platform. In the implementation of the real-time environment, how to ensure the timely upload of data and how to upload large amounts of data with minimal resource overhead remain an open question that we hope to pursue in our further work.

REFERENCES

[1] P. Shen, M. Ouyang, L. Lu, J. Li, and X. Feng, "The co-estimation of state of charge, state of health, and state of function for lithium-ion batteries in electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 92–103, Jan. 2018.

[2] J. Duan, X. Tang, H. Dai, Y. Yang, W. Wu, X. Wei, and Y. Huang, "Building safe lithium-ion batteries for electric vehicles: A review," *Electrochem. Energy Rev.*, vol. 3, no. 1, pp. 1–42, Mar. 2020.

[3] Y. Ding, Z. P. Cano, A. Yu, J. Lu, and Z. Chen, "Automotive li-ion batteries: Current status and future perspectives," *Electrochem. Energy Rev.*, vol. 2, no. 1, pp. 1–28, Mar. 2019.

[4] K. W. E. Cheng, B. P. Divakar, H. Wu, K. Ding, and H. F. Ho, "Battery-management system (BMS) and SOC development for electrical vehicles," *IEEE Trans. Veh. Technol.*, vol. 60, no. 1, pp. 76–88, Jan. 2011.

[5] Y. Zhang, X. Rui, H. He, and M. Pecht, "Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5695–5705, Jul. 2018.

[6] N. Denis, M. R. Dubois, J. P. F. Trovao, and A. Desrochers, "Power split strategy optimization of a plug-in parallel hybrid electric vehicle," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 315–326, Jan. 2018.

[7] D. N. T. How, M. A. Hannan, M. S. Hossain Lipu, and P. J. Ker, "State of charge estimation for lithium-ion batteries using model-based and data-driven methods: A review," *IEEE Access*, vol. 7, pp. 136116–136136, 2019.

[8] A. Fotouhi, D. J. Auger, K. Propp, and S. Longo, "Lithium–sulfur battery state-of-charge observability analysis and estimation," *IEEE Trans. Power Electron.*, vol. 33, no. 7, pp. 5847–5859, Jul. 2018.

[9] G. Blaj, C. J. Kenney, A. Dragone, G. Carini, S. Herrmann, P. Hart, A. Tomada, J. Koglin, G. Haller, S. Boutet, M. Messerschmidt, G. Williams, M. Chollet, G. Dakovski, S. Nelson, J. Pines, S. Song, and J. Thayer, "Optimal pulse processing, pile-up decomposition, and applications of silicon drift detectors at LCLS," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 11, pp. 2854–2868, Nov. 2017.

[10] Y. Li, J. Hu, M. Liu, Y. Chen, K. W. Chan, Z. He, and R. Mai, "Reconfigurable intermediate resonant circuit based WPT system with load-independent constant output current and voltage for charging battery," *IEEE Trans. Power Electron.*, vol. 34, no. 3, pp. 1988–1992, Mar. 2019.

[11] X. Li, L. Zhao, L. Wei, M.-H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang, "DeepSaliency: Multi-task deep neural network model for salient object detection," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3919–3930, Aug. 2016.

[12] H. Chaoui and C. C. Ibe-Ekeocha, "State of charge and state of health estimation for lithium batteries using recurrent neural networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 8773–8783, Oct. 2017.

[13] Y. Han, Q. Li, T. Wang, W. Chen, and L. Ma, "Multisource coordination energy management strategy based on SOC consensus for a PEMFC–battery–supercapacitor hybrid tramway," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 296–305, Jan. 2018.

[14] S. Dey, B. Ayalew, and P. Pisu, "Nonlinear robust observers for state-of-charge estimation of lithium-ion cells based on a reduced electrochemical model," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 5, pp. 1935–1942, Sep. 2015.

[15] Z. Zhang, X. Cheng, Z.-Y. Lu, and D.-J. Gu, "SOC estimation of lithium-ion battery pack considering balancing current," *IEEE Trans. Power Electron.*, vol. 33, no. 3, pp. 2216–2226, Mar. 2018.

[16] H. Yang, H. Wen, Y. Qi, and J. Fan, "An equivalent circuit model to analyze passive intermodulation of loose contact coaxial connectors," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 5, pp. 1180–1189, Oct. 2018.

[17] E. Chemali, P. J. Kollmeyer, M. Preindl, R. Ahmed, and A. Emadi, "Long short-term memory networks for accurate State-of-Charge estimation of li-ion batteries," *IEEE Trans. Ind. Electron.*, vol. 65, no. 8, pp. 6730–6739, Aug. 2018.

[18] A. Bartlett, J. Marcicki, S. Onori, G. Rizzoni, X. Guang Yang, and T. Miller, "Electrochemical model-based state of charge and capacity estimation for a composite electrode lithium-ion battery," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 384–399, Mar. 2016.

[19] E. Chatzinikolaou and D. J. Rogers, "Hierarchical distributed balancing control for large-scale reconfigurable AC battery packs," *IEEE Trans. Power Electron.*, vol. 33, no. 7, pp. 5592–5602, Jul. 2018.

[20] S. Zhao, S. R. Duncan, and D. A. Howey, "Observability analysis and state estimation of lithium-ion batteries in the presence of sensor biases," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 1, pp. 326–333, Jan. 2017.

[21] H. Tian, X. Wang, Z. Lu, Y. Huang, and G. Tian, "Adaptive fuzzy logic energy management strategy based on reasonable SOC reference curve for online control of plug-in hybrid electric city bus," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1607–1617, May 2018.

[22] T. Chen, L. Lin, X. Wu, N. Xiao, and X. Luo, "Learning to segment object candidates via recursive neural networks," *IEEE Trans. Image Process.*, vol. 27, no. 12, pp. 5827–5839, Dec. 2018.

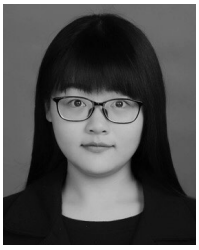
[23] J. Wei, G. Dong, and Z. Chen, "Remaining useful life prediction and state of health diagnosis for lithium-ion batteries using particle filter and support vector regression," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5634–5643, Jul. 2018.

[24] W. Li, L. Liang, W. Liu, and X. Wu, "State of charge estimation of lithium-ion batteries using a discrete-time nonlinear observer," *IEEE Trans. Ind. Electron.*, vol. 64, no. 11, pp. 8557–8565, Nov. 2017.
[25] J. Chen, Q. Ouyang, C. Xu, and H. Su, "Neural network-based state of charge observer design for lithium-ion batteries," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 1, pp. 313–320, Jan. 2018.
[26] Š. Papáček, B. Macdonald, and C. Matonoha, "Closed-form formulas vs. PDE based numerical solution for the FRAP data processing: Theoretical and practical comparison," *Comput. Math. Appl.*, vol. 73, no. 8, pp. 1673–1683, Apr. 2017.
[27] Q. Yu, R. Xiong, C. Lin, W. Shen, and J. Deng, "Lithium-ion battery parameters and State-of-Charge joint estimation based on H-Infinity and unscented Kalman filters," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 8693–8701, Oct. 2017.
[28] N. Mukherjee and D. De, "A new State-of-Charge control derivation method for hybrid battery type integration," *IEEE Trans. Energy Convers.*, vol. 32, no. 3, pp. 866–875, Sep. 2017.
[29] W. Choi, K. Duraisamy, R. G. Kim, J. R. Doppa, P. P. Pande, D. Marculescu, and R. Marculescu, "On-chip communication network for efficient training of deep convolutional networks on heterogeneous manycore systems," *IEEE Trans. Comput.*, vol. 67, no. 5, pp. 672–686, May 2018.
[30] Y. Zhou, L. Liu, L. Shao, and M. Mellor, "Fast automatic vehicle annotation for urban traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 6, pp. 1973–1984, Jun. 2018.
[31] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1100–1111, Apr. 2018.
[32] A. Dominguez-Sanchez, M. Cazorla, and S. Orts-Escolano, "Pedestrian movement direction recognition using convolutional neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3540–3548, Dec. 2017.
[33] Q. Hu, H. Wang, T. Li, and C. Shen, "Deep CNNs with spatially weighted pooling for fine-grained car recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3147–3156, Nov. 2017.
[34] M. Saha and C. Chakraborty, "Her2Net: A deep framework for semantic segmentation and classification of cell membranes and nuclei in breast cancer evaluation," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2189–2200, May 2018.
[35] A. Farag, L. Lu, H. R. Roth, J. Liu, E. Turkbey, and R. M. Summers, "A bottom-up approach for pancreas segmentation using cascaded superpixels and (Deep) image patch labeling," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 386–399, Jan. 2017.
[36] J. An, L. Fu, M. Hu, W. Chen, and J. Zhan, "A novel fuzzy-based convolutional neural network method to traffic flow prediction with uncertain traffic accident information," *IEEE Access*, vol. 7, pp. 20708–20722, 2019.

FEN ZHAO was born in Zaozhuang, Shandong, China, in 1991. She is currently pursuing the Ph.D. degree with the Chongqing University of Posts and Telecommunications, China. Her main research interests include principle and application of neural computation, natural language processing, recognition of speech signals, intelligent control theory, and pattern recognition.

YINGUO LI was born in Huangmei, Hubei, China, in 1955. He received the Ph.D. degree in control science and engineering from Chongqing University, Chongqing, in 2003. Since 2007, he has been a Professor with the Automation Department, Chongqing University of Posts and Telecommunications. His research interests include computer vision on autonomous vehicle, advance driving assistant systems, intelligent control theory, and pattern recognition. He received the first and second prize of the Chongqing Science and Technology Progress, in 2009 and 2014, respectively.

XINHENG WANG was born in Zaozhuang, Shandong, China, in 1988. He is currently pursuing the Ph.D. degree with the Chongqing University of Posts and Telecommunications, China. His research interests include natural language processing, recognition of speech signals, and intelligent control theory.



LING BAI was born in Shijiazhuang, Hebei, China, in 1991. She is currently pursuing the Ph.D. degree with the Chongqing University of Posts and Telecommunications, China. Her research interests include principle and application of neural computation, recognition of image and speech signals, intelligent control theory, and pattern recognition.

TAILIN LIU was born in Youyang, Chongqing, China, in 1992. He received the B.Eng. and M.Eng. degrees from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2015 and 2018, respectively. He is currently with The 32nd Institute of China Electronics Technology Corporation. His main research interests include natural language processing and image recognition.