收件箱　　[IJCNN 2023] Y...

返回　回复　回复全部　转发 ▾　删除　标记为 ▾　移动　备注　举报　分享　更多 ▾

[IJCNN 2023] Your paper #1570884055 ('CyFormer: Accurate State-Of-Health Prediction of Lithium-Ion Batteries via Cyclic Attention')

ToMe

IJCNN 2023 (由 010001875c26a996-b87b2821-5d16-4c7a-abb3-d7a00668399c-000000@mail.edas.info代发)　2023-04-07 22:37

收件人：我 、Jiankun Zhao、Qicheng Li　详情

Dear Mr. Zhiqiang Nie,

Congratulations! Your paper: #1570884055 ('CyFormer: Accurate State-Of-Health Prediction of Lithium-Ion Batteries via Cyclic Attention') has been accepted for presentation at the International Joint Conference on Neural Networks (IJCNN 2023) to be held at the Gold Coast Convention and Exhibition Centre Queensland, Australia, 18-23 June 2023.

The reviews for your paper are below or can be found at https://edas.info/showPaper.php?m=1570884055. We would like to remind you that the authors of all accepted papers at IJCNN 2023 must upload the final versions. Even if no changes are recommended, a final version of the paper MUST BE UPLOADED. The final version should reflect the revisions in response to reviewers' comments and fix ALL format issues. Manuscripts not following the reviewers' comments and format requirement of IJCNN 2023 will be excluded from the conference proceedings, so please revise your paper carefully based on the reviewers' comments. Please note that you must complete and upload your IEEE copyright form prior to submitting your final manuscript. Additionally, all accepted papers will need to submit a pre-recorded video presentation, additional details will be sent shortly.

The final version of your manuscripts must be submitted by April 21, 2023.
It is the responsibility of the submitting author to ensure that the document was viewable and without any errors that would prevent the Conference from including the paper in the Digital Proceedings.

By contributing a manuscript for consideration to IJCNN 2023, the Conference has the expectation that accepted papers will be presented either by the

快捷回复全部　　　　　　　　　好的　收到　太棒了　谢谢　转发

CyFormer: Accurate State-of-Health Prediction of Lithium-Ion Batteries via Cyclic Attention

Zhiqiang Nie[1*], Jiankun Zhao[2*], Qicheng Li[2†], Yong Qin[2†]

[1] College of Cyber Science, Nankai University, Tianjin, China
[2] College of Computer Science, Nankai University, Tianjin, China
{2012307, 2010535}@mail.nankai.edu.cn, {liqicheng, qinyong}@nankai.edu.cn

Abstract—**Predicting the State-of-Health (SoH) of lithium-ion batteries is a fundamental task of battery management systems on electric vehicles. It aims at estimating future SoH based on historical aging data. Most existing deep learning methods rely on filter-based feature extractors (e.g., CNN or Kalman filters) and recurrent time sequence models. Though efficient, they generally ignore cyclic features and the domain gap between training and testing batteries. To address this problem, we present CyFormer, a transformer-based cyclic time sequence model for SoH prediction. Instead of the conventional CNN-RNN structure, we adopt an encoder-decoder architecture. In the encoder, row-wise and column-wise attention blocks effectively capture intra-cycle and inter-cycle connections and extract cyclic features. In the decoder, the SoH queries cross-attend to these features to form the final predictions. We further utilize a transfer learning strategy to narrow the domain gap between the training and testing set. To be specific, we use fine-tuning to shift the model to a target working condition. Finally, we made our model more efficient by pruning. The experiment shows that our method attains an MAE of 0.75% with only 10% data for fine-tuning on a testing battery, surpassing prior methods by a large margin. Effective and robust, our method provides a potential solution for all cyclic time sequence prediction tasks.**

Index Terms—**SoH, time sequence, transformer, cyclic attention, transfer learning**

Fig. 1: Illustration of the SoH prediction task and the row-wise and column-wise attention mechanism. The model takes in a series of historical physical quantities, and outputs SoH values of future cycles. In the experiment, the physical quantities we use are measured current (Cm), measured voltage (Vm), load current (Cl), load voltage (Vl) and temperature (T). Row-wise attention captures intra-cycle connections, while col-wise attention captures inter-cycle connections.

I. INTRODUCTION

Researches on battery management systems (BMS) have received increasing attention for the rapid commercialization of electric vehicles (EVs) [1]. One of the core tasks of BMS is to predict State-of-Health (SoH) of Li-ion batteries. SoH is defined as the ratio of the current releasable battery charge to its rated capacity. It gradually decreases after charging and discharging for a number of cycles, indicating a shrinkage in capacity and maximum power. However, this vital indicator cannot be measured directly due to the complex dynamic behavior and time-varying conditions of Li-ion batteries [2]. The task of predicting SoH is to estimate SoH of future charging-discharging cycles given aging data (current, voltage, temperature, etc.) within every historical cycle (see Fig. 1).

Accuracy guarantees safety, and safety protects life. BMS needs accurate SoH predictions to optimize energy consumption, prevent over-charging and over-discharging, and extend battery life. In contrast, inaccurate estimations of SoH may lead EVs to spontaneous combustion or anchoring.

*Equal contribution.
†Corresponding authors.

Challenges in accurate SoH prediction can be concluded as the following three points: First, SoH is a highly-complicated non-linear function of current, voltage, temperature and other parameters of historical cycles. Theoretically, deep neural network is a perfect choice to fit this function and learn the aging trend from historical data. But practically, it is hard for existing time sequence model to both learn long-term patterns among different cycles and extract battery features within each cycle. Second, different from many application scenarios of artificial intelligence, the aging data of Li-ion batteries is scarce. This usually causes under-fitting on light weight models and over-fitting on larger models. Third, the charging and discharging behavior of one battery may significantly differ from another, even if they are of the same type. A domain gap exists between batteries working in different conditions. Previous works on SoH estimation generally utilize models based on RNN or LSTM [3], [4]. Though more efficient, these models suffer severely from forgetting long-term patterns [5]. More importantly, they quickly forget patterns learnt from training set when being fine-tuned on a test battery, and thus perform poorly when data for fine-tuning is scarce. Recently, several

transformer-based methods have been proposed [6]–[8]. In order to convert initial data into a transformer style input, most of them use a CNN-based feature extractor to extract intra-cycle features. However, the input data within each cycle does not have a hierarchical waveform structure. Therefore, it is hard for convolution filters to capture intra-cycle features effectively. Additionally, this pipeline compresses all sample points within a cycle into one single dimension, which might induce feature loss.

In this work, we present CyFormer, a novel generalized cyclic time sequence model, to address the aforementioned problems. This CNN-free model follows the typical encoder-decoder pipeline of transformer. The encoder first extracts cyclic features from historical data and transmits them into the decoder. Then the SoH queries cross-attend to these features in the decoder and forms the final predictions. At the core of our model lies the cyclic attention (i.e., row-wise and column-wise attention) blocks. Row-wise attention block aims at extracting intra-cycle connections, whereas column-wise attention block aims at extracting inter-cycle connections (Fig. 1). Compared with CNN, these two blocks enable the encoder to capture connections between sample points in different cycles, and preserve intra-cycle features at the same time.

Extensive experiments demonstrate both the effectiveness and the robustness of CyFormer on SoH prediction. Compared with previous works, we adopt a more challenging criterion for testing to fully demonstrate the transfer learning performance of our model. More specifically, we fine-tune the model with only 10% of the SoH data at the beginning, predict SoH of the remaining 90% hidden cycles, and calculate prediction error with the ground truth of these hidden cycles. With this testing method, our model achieved an MAE of 0.75% and an MAPE of 0.90%, surpassing baseline methods by a large margin. Industries may acquire the first few cycles of SoH data in quality control process of every new battery. Our result means that industries can use this SoH data to train a model which could give highly accurate predictions of SoH on BMS.

Our contributions can be concluded as follows:

- We proposed CyFormer, a generalized cyclic time sequence model with row-wise and column-wise attention mechanism. With CyFormer, we gained highly accurate SoH predictions and achieved new SOTA in SoH estimation.
- We adopt a transfer learning style testing criterion, which is closer to the real application scenario. Experiments showed that our model maintains accuracy by this criterion.
- We designed a light weight version of CyFormer for BMS by pruning unnecessary modules. It becomes significantly more efficient than the initial version with only a tiny loss on accuracy.

II. RELATED WORK

a) SoH Prediction: Modern SoH prediction methods can be generally divided into three categories: direct measurement methods, adaptive algorithms and data-driven methods [6].

Direct measurement methods [9], [10] analyze the aging behavior through numerous laboratory tests. These off-line methods require specialized sensors in laboratories. Adaptive algorithms use traditional mathematical models and numerical filters. Lim et al. [11] proposed Fading Kalman filter (FKF), which avoids large estimation errors in conventional Kalman filter. This method induces large computational costs [5], and is not efficient enough to be deployed on BMS.

Data-driven methods can be further divided into machine learning methods and deep learning methods. Machine learning methods typically utilize support vector machine (SVM) [12]–[14] or Gaussian process regression (GPR) [15]–[17]. Most existing SOTAs adopt deep learning methods, such as CNN-LSTM [3], ViT [7] or DynaFormer [8]. Fan et al. [4] proposed a hybrid neural network, which extracts local information with CNN and captures time dependencies with GRU. To better capture global representation, Gu et al. [6] proposed a CNN-Transformer framework that replaces recurrent modules with transformer encoder and decoder. To reduce oscillations, Shen et al. [5] introduced Immersion and Invariance (I&I) adaptive observer into the transformer-based pipeline.

b) Time Series Analysis: In time series forecasting, one of the most prominent models is ARIMA [18]. Flunkert et al. [19] first integrated auto-regression with RNN and proposed DeepAR, a probabilistic forecasting network. Bai et al. [20] discovered that a simple convolutional architecture outperforms canonical recurrent networks (e.g., RNN, LSTM) on a wide spectrum of tasks and datasets. Li et al. [21] proposed LogSparse Transformer with only $O(L(logL)^2)$ memory cost. They also utilized convolutional self-attention so that local context can be better incorporated into attention mechanism.

c) Transfer Learning: In many deep learning tasks, a domain gap exists between training and testing datasets. Therefore, many generalized transfer learning methods have been proposed. Kumar et al. [22] suggested LP-FT, a two-step strategy that first trains linear probing module and then fine-tunes the entire model. Similar strategies have been adopted in SoH predictions. To boost performance on batteries in different working conditions, Fu et al. [7] conducted fine-tuning with SoH data of the first few cycles to shift the model to the testing battery.

III. TASK STATEMENT

As illustrated in Fig. 1, consider that we have a battery working at the end of the t-th charge-discharge cycle. Given size of the prediction window n_{out}, the task of our model is to predict SoH value of cycle $t+1$ to $t+n_{out}$, based on the aging data of cycle 1 to t. To be specific, assume that l_{sample} is the sample point number within each cycle, and c is the number of physical quantities measured at each sample point (i.e., input channel size). The input of this task can be organized as the

following $t \times l_{sample}$ matrix,

$$input = \begin{bmatrix} X_{11}, & X_{12}, & \cdots, & X_{1l_{sample}} \\ X_{21}, & X_{22}, & \cdots, & X_{2l_{sample}} \\ \vdots & \vdots & & \vdots \\ X_{t1}, & X_{t2}, & \cdots, & X_{tl_{sample}} \end{bmatrix} \qquad (1)$$

where each X_{ij} is a vector of size C. It is composed of physical quantities like current, voltage and temperature. The output of this task is a sequence of predicted SoH values,

$$output = \{SoH_i | i = t+1, \cdots, t+n_{out}\} \qquad (2)$$

where SoH_i is a number within [0, 1].

There are two special cases worth considering. The first one is named just-in-time (JIT) prediction, where we set n_{out} as 1 and only predict SoH of the current cycle. The second one is named Remaining Useful Life (RUL) prediction, where we set n_{out} to a pre-defined maximum and predict the cycle number when SoH decreases to a certain threshold. This threshold represents the scrapping point of batteries, and the number of remaining cycles indicates the remaining life of the battery. In this work, we focus on JIT research.

IV. METHOD

An overview of our model is depicted in Fig.2. As described in Section III, our model directly processes two-dimensional cyclic data, rather than the one-dimensional input sequence of the classical transformer architecture. The encoder first encodes the two-dimensional input into a one-dimensional feature sequence, and then feed it into the decoder. In the decoder, the randomly-initialized SoH queries cross-attend to these features to form prediction values of n_{out} future cycles. In the following sections, we will discuss the detailed structure of the encoder and the decoder respectively.

A. Encoder

The encoder is composed of four main parts, namely the input embedding module, the 2D positional encoding module, a stack of encoder layers and the output head.

Both the input embedding and the output head are linear layers. The input embedding module extends the channel size of each input token to $d_{encoder}$. It applies the following affine transformation on each vector in the input matrix 1:

$$X'_{ij} = W^\top X_{ij} + b \qquad (3)$$

where W^T is a $C \times d_{encoder}$ weight matrix and b is a bias vector of size $d_{encoder}$. As the output of the embedding module, X'_{ij} is then fed into the first encoder layer.

The output head fully connects all sample points in each input cycle to form n_{in} feature vectors with the channel size of $d_{decoder}$. It can be described as Formula 4.

$$F_i = \sum_{m=1}^{l_{sample}} O_m^\top X_{im} + b \qquad (4)$$

where F_i is the i-th vector in the cyclic feature sequence. Each W_m^T is a $d_{encoder} \times d_{decoder}$ weight matrix, and b is a bias vector of size $d_{decoder}$.

The 2D positional encoding of each input token is defined as:

$$PE2D_{(cycle,sample)} = PE1D_x + PE1D_y \qquad (5)$$

Where $PE1D_x$ and $PE1D_y$ are both 1D sinusoidal positional encodings calculated by [23]:

$$PE1D_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \qquad (6)$$

$$PE1D_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \qquad (7)$$

Each encoder layer consists of a row-wise attention block, a column-wise attention block and a 3-layer MLP. Each of these blocks is followed by a residual block and a layer normalization block. Both row-wise and column-wise attention blocks derive from self-attention blocks (Alg. 1). They are designed to capture intra-cycle and inter-cycle connections respectively.

a) Row-wise Attention: The upper part of Fig.3 illustrates the structure of a row-wise attention block. Row-wise attention aims at capturing connections between data sampled at different times within a single cycle. To this end, we first split the two-dimensional input into single rows. Each row contains all sample points within a particular cycle. Then each row is regarded as an individual input sequence and goes through a self-attention block with shared weights. To be specific, each row generates its own queries, keys and values. Following the typical multi-head attention mechanism, we first calculate dot-product affinities between queries and keys to form attention weights. Then we multiply the values with their corresponding weights, and feed the result into an output linear layer. After going through self-attention blocks, the outputs of all rows are concatenated to form a final output with the same shape as the original input.

b) Column-wise Attention: The lower part of Fig.3 illustrates the structure of a column-wise attention block. Column-wise attention aims at capturing connections between data sampled at the same time but in different cycles. Similar to row-wise attention, we achieved this goal by slicing the input data into individual columns, feeding them into self-attention blocks with shared weights, and concatenating them into an output matrix.

It should be noticed that the inputs and outputs of row-wise and column-wise blocks all have the same shape, which renders these blocks compatible with the sequential transformer architecture. Additionally, our method preserves intra-cycle features since the second dimension is never squeezed until the end of the encoder.

B. Decoder

The decoder is composed of three main parts, namely the positional encoding module, a stack of decoder layers, and a linear output head. A sequence of randomly-initialized SoH queries goes through these parts to form the final SoH
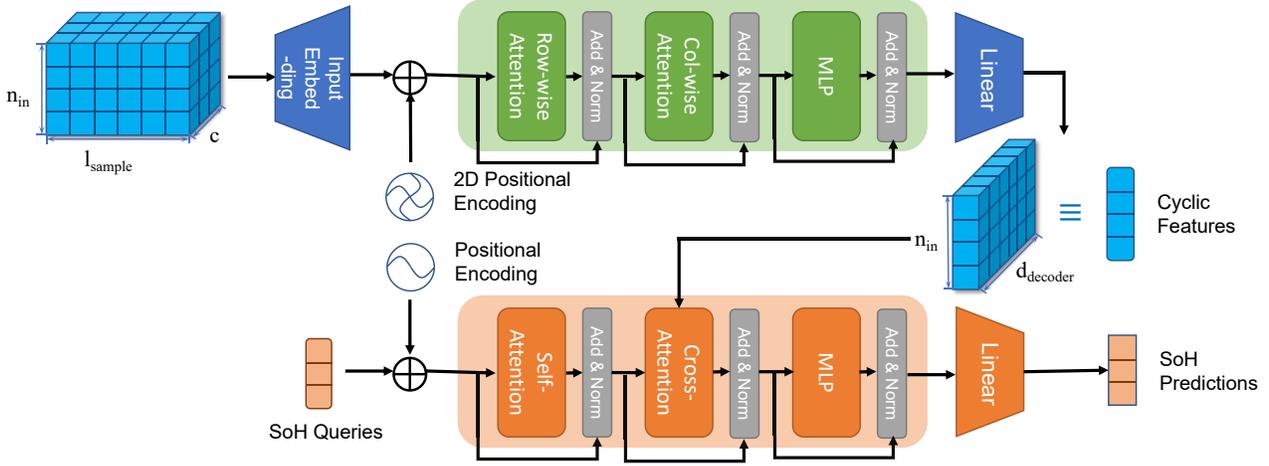
Fig. 2: An overview of our model. Components of the encoder are colored green, whereas componets of the decodered are colored orange. Each cube in the input or square in the output represents a number, while each square in rounded rectangles represents a vector of size $d_{decoder}$.

Algorithm 1 Row-wise and Column-wise Attention

Input: Matrix $[X'_{ij}]$ with channel size c
Output: Matrix $[O_{ij}]$ with channel size c
  for each row R_i in $[X'_{ij}]$ **do**
    // Input projections
    $Q_i, K_i, V_i \leftarrow linear(R_i)$
    // Self-attention
    $A_i \leftarrow softmax\left(\frac{1}{\sqrt{d_{encoder}}}\mathbf{Q}_i^\top \mathbf{K}_i\right)$
    $R_i \leftarrow layerNorm(R_i + V_i A_i^\top)$
  end for
  for each column C_j in $[X'_{ij}]$ **do**
    // Input projections
    $Q_j, K_j, V_j \leftarrow linear(C_j)$
    // Self-attention
    $A_j \leftarrow softmax\left(\frac{1}{\sqrt{d_{encoder}}}\mathbf{Q}_j^\top \mathbf{K}_j\right)$
    $C_j \leftarrow layerNorm(C_j + V_j A_j^\top)$
  end for
  $[O_{ij}] \leftarrow [X'_{ij}]$
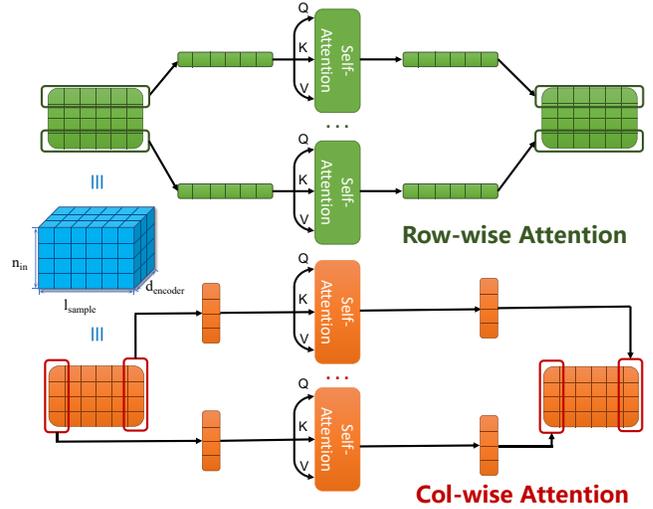
Fig. 3: An illustration of cyclic attention mechanism. The upper part and the lower part show the structure of row-wise and column-wise attention blocks, respectively. Each square in rounded rectangles represents a vector of size $d_{encoder}$.

predictions. The query sequence consists of n_{out} vectors with a channel size of $d_{decoder}$:

$$query = \begin{bmatrix} Q_1, & Q_2, & \cdots, & Q_{n_{out}} \end{bmatrix} \quad (8)$$

The output head is a linear layer. It fully connects all channels of each query vector and outputs n_{out} SoH values:

$$SoH_{t+i} = W^\top Q_i + b \quad (9)$$

In the decoder, we use the 1D sinusoidal positional encoding defined in Formula 6 and 7.

Following the typical transformer decoder architecture, each decoder layer consists of a self-attention block, a cross-attention block and a 3-layer MLP. Similar to the encoder, each of these blocks is followed by a residual block and a layer normalization block. The cross-attention block informs queries of the historical features $\{F_i\}$, while the self-attention block keeps the historical trend among all the predictions.

V. EXPERIMENT

In this section, we first introduce the dataset and the data pre-processing procedure (Sec. V-A). Then, We introduce evaluation metrics (Sec. V-B) and implementation details (Sec. V-C). We conducted a comparative experiment with CNN-LSTM and CNN-Transformer (Sec. V-D) to demonstrate the accuracy of CyFormer. To better validate each component of

our model, we provide detailed ablation studies (Sec. V-E). Finally, we present a light weight version of CyFormer, striking a balance between accuracy and computational costs(Sec. V-F).

A. Dataset

We carry out the experiment with the Battery Data Set provided by NASA Ames Prognostics Center of Excellence [24], [25]. We removed batteries that are extremely inconsistent with the common aging patterns of batteries. Fig. 4 shows the aging curve of the nineteen batteries we used. These battery cells worked in different ambient temperatures(4℃, 24℃, 43℃).In each cycle, they were first charged through a constant current - constant voltage (CC-CV) procedure with the upper voltage at 4.2V until the current decayed to 20 mA. Then they were discharged with constant or pulse current waveforms until each of the cells reached its cut-off voltage. We use load voltage (Vl), load current (Cl), measured voltage (Vm), measured current (Cm) and temperature (T) curves of the discharge process (see Fig. 1). The SoH of a batteryis defined as the ratio of the maximum charge to its rated capacity:

$$\text{SOH} = \frac{Q_{\max}}{C_r} \times 100\% \qquad (10)$$

where $Q_{\max}$ is the maximum charge available from the current battery and C_r is the rated capacity. To align sample points in different cycles, we linearly interpolated and re-sampled intra-cycle data.
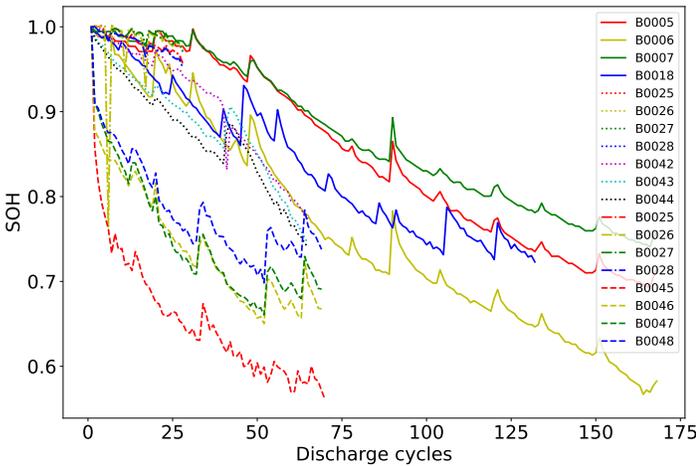
Fig. 4: The SoH decay curves of the 19 batteries in the Battery Data Set. The same linestyle indicates the same group of batteries. Batteries are marked with different colors in the same group.

Among all nineteen batteries, one is selected as the target battery for testing, and others are used as the source dataset for training. To this end, aging data of the target battery is further divided into the fine-tuning segment(10%) and the hidden segment(90%). We adopt a two-stage transfer learning strategy to narrow the domain gap between the source and the target batteries. We first train the model on the source dataset to fit general working conditions, and then fine-tune the model on the fine-tuning segment of the target battery to shift to a new working condition. Finally, evaluation of the model is conducted on the hidden segment of the target battery.

B. Evaluation Metrics

To evaluate the performance of CyFormer, three different evaluation metrics are employed: mean absolute percentage error (MAPE), mean absolute error (MAE) and root mean square error (RMSE). MAPE represents the relative percentage error between the prediction and the actual value. MAE is the average of the absolute difference between the estimation and the actual value of SOH. It aims at measuring the average magnitude of errors of the proposed method. RMSE indicates the deviation of the estimation value and the actual value, and thus represents the quality of estimation. MAE, MAPE, RMSE are defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\widehat{y}_i - y_i| \qquad (11)$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\widehat{y}_i - y_i}{y_i}\right| \qquad (12)$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widehat{y}_i - y_i)^2} \qquad (13)$$

where $y_1, y_2, \cdots, y_n$ are the actual values, $\widehat{y}_1, \widehat{y}_2, \cdots, \widehat{y}_n$ are the predicted values, and n is the number of testing samples.

C. Implementation Details

We choose MAE as the loss function. The network is trained with Adam optimizer with a learning rate of 0.0001. We set β_1 as 0.9 and β_2 as 0.999. Grid search is used to obtain the optimal model parameters. The selected parameters are shown in Table I.

TABLE I: Hyperparameter settings

Hyperparameters	value
l_{sample}	32
$d_{encoder}$	16
$d_{decoder}$	16
n_{in}	16
Gamma	0.1
Batch size	32
Epochs	1500
Encoder layers	4
Decoder layers	4
Attention heads h	8
Learning rate(training)	0.0001
Learning rate(fine-tuning)	0.0002

The input window size n_{in} is defined as the number of cycles contained in the input sequence. The SoH prediction performances with different input window size n_{in} are shown in II. When n_{in} increases, the accuracy increases

simultaneously, whereas the computational cost rises as well. Additionally, ground-truth data for fine-tuning would be scarce if n_{in} were too large. Therefore, we set $n_{in} = 16$ as it strikes a balance between accuracy and efficiency.

TABLE II: The SoH result of different n_{in}

n_{in}	FLOPs	Params	MAE	MAPE	RMSE
8	0.09	0.32	2.68%	2.95%	3.13%
12	0.13	0.33	1.56%	1.66%	1.90%
16	0.17	0.35	0.75%	0.89%	0.95%
32	0.34	0.41	0.73%	0.90%	0.95%

We adopt a transfer learning style testing criterion. The fine-tuning segment of the target battery only contains a small amount (10%) of data at the beginning of the ageing phase. The feature extraction modules are mainly trained on the source dataset. In the fine-tuning process, parameters in the decoder are mainly modified, and the model quickly shifts to the target domain with the 10% fine-tuning segment.

D. Comparison with Other Methods

As mentioned in section V-A, Cell #B0007 is randomly selected as the target battery. Other batteries are selected as the source dataset. The first 10%, 30%, or 70% of #B0007 were adopted for offline fine-tuning, while the remaining part (90%, 70%, or 30%) were used for online evaluation. As typical baselines, CNN-LSTM and CNN-Transformer were employed to estimate battery SoH with the same testing criteria.

As shown in Tab. III, the MAEs, MAPEs and RMSEs of the CyFormer are within 1%, while for CNN-LSTM and CNN-Transformer, the highest errors are about 4% and 3%, respectively. CyFormer achieves the lowest loss among all three methods under all three circumstances, demonstrating its effectiveness and robustness. It should be noticed that CyFormer can achieve accurate prediction using only 10% or less fine-tuning data, while CNN-Transformer need at least 70% to reach a comparable result. This can also prove the transfer learning efficiency of CyFormer.

TABLE III: Comparison of estimation errors among CyFormer and other methods

Battery	Method	MAE	MAPE	RMSE
B0007(10%)	CNN-LSTM	2.69%	2.98%	3.30%
	CNN-Transformer	2.01%	2.41%	2.33%
	CyFormer	**0.75%**	**0.89%**	**0.95%**
B0007(30%)	CNN-LSTM	1.74%	2.19%	2.17%
	CNN-Transformer	1.12%	1.56%	1.67%
	CyFormer	**0.66%**	**0.87%**	**0.96%**
B0007(70%)	CNN-LSTM	1.70%	2.11%	1.89%
	CNN-Transformer	0.66%	0.86%	0.82%
	CyFormer	**0.38%**	**0.52%**	**0.49%**

Fig. 5 (a)-(c) show the SoH prediction results on the target battery (#B0007). The SoH prediction accuracy of CyFormer surpasses other methods by a large margin, especially when
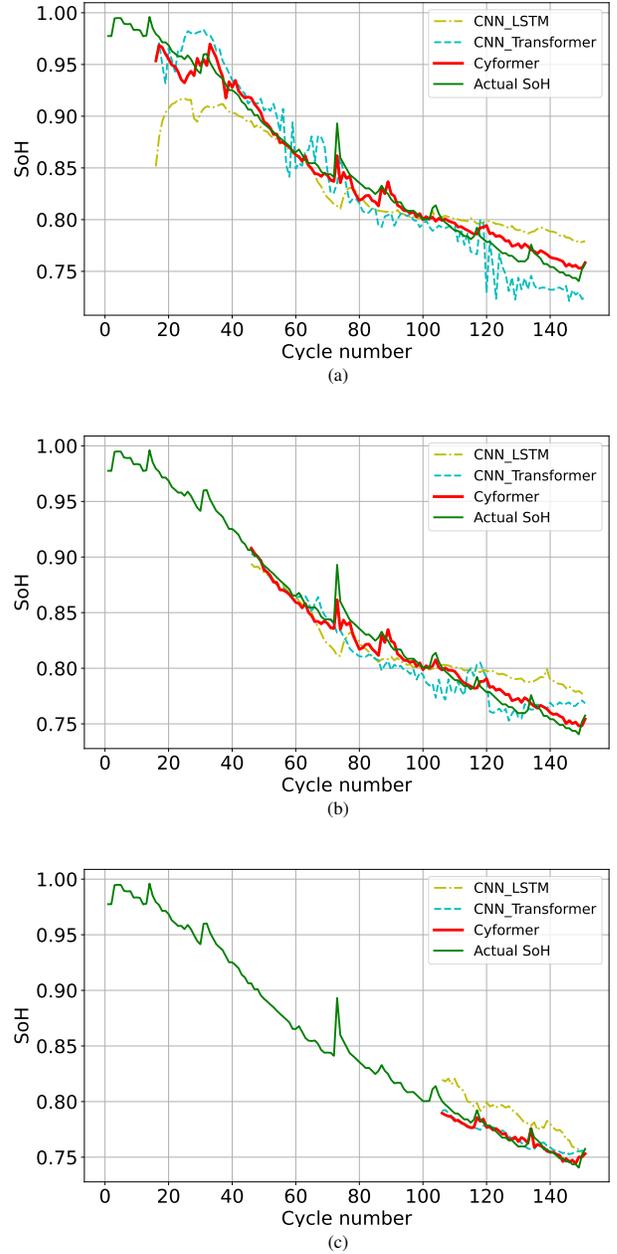
Fig. 5: The result of SoH estimation based on 10% (a), 30% (b), 70% (c) transfer learning dataset.

the fine-tuning proportion is 10% (Fig. 5(a)). When the fine-tuning proportion expands to 30%, CNN-LSTM and CNN-Transformer models closely follow the battery ageing trend only in the first twenty cycles. Significant improvement on accuracy of CNN-Transformer has not appeared until the fine-tuning proportion reaches 70%. In contrast, the CNN-LSTM model still jitters significantly even if the the fine-tuning proportion reaches 70%.

E. Ablation Study

In order to validate the effect of row-wise and column-wise attention blocks, we conducted abalation studies on following conditions.

- w/o Row-wise: CyFormer without row-wise structure
- w/o Col-wise: CyFormer without column-wise structure
- w/o Row-wise + Col-wise: CyFormer without row-wise and column-wise structure

According to Fig. 6, cutting off row-wise and column-wise attention blocks decreases accuracy, verifying the effectiveness of the two structures. As shown in Tab. IV, the column-wise and row-wise atttention blocks reduce MAE by 3.06%, MAPE by 3.82%, and RMSE by 3.54%.
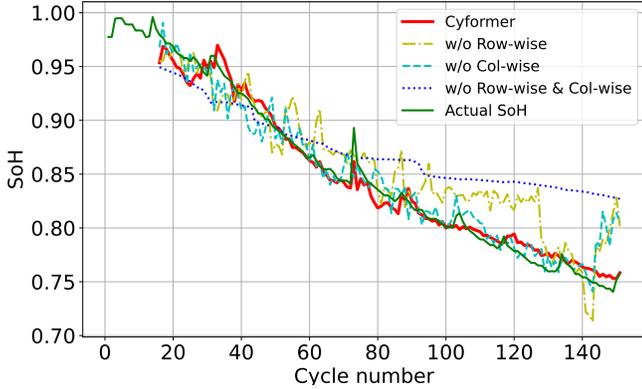
Fig. 6: Ablation Study

TABLE IV: Ablation study of column-wise and row-wise attention blocks

Col-wise.	Row-wise.	MAE	MAPE	RMSE
		3.81%	4.72%	4.53%
	✓	1.87%	2.18%	3.42%
✓		2.93%	3.58%	4.02%
✓	✓	**0.75%**	**0.89%**	**0.95%**

F. Pruning

In this section, we designed a light weight version of CyFormer by pruning. To be specific, we reduced the number of encoders (i.e., depth), and the number of sampling points. We only change one component each time to observe how that affects performance and efficiency.

a) Depth: The depth of the model is defined as the number of CyFormer encoders. It is highly related to the effectiveness of feature extraction. The initial depth is set as 4. To make it more efficient, we carried out four groups of experiments that set depth to 1-4, respectively. The RMSEs, MAEs, MAPEs and FLOPs of SoH prediction are shown in Fig. 7.

It can be seen from Fig. 7 that the feature extraction ability becomes stronger as the model depth increases. Each of the
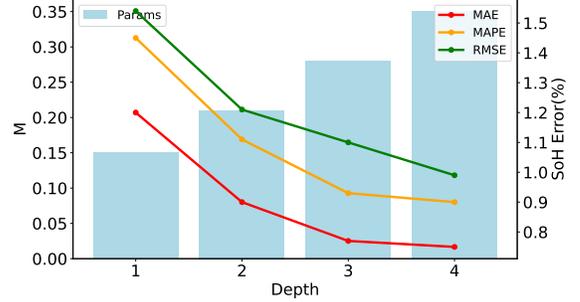
Fig. 7: The results of pruning study on model depth.

first three layers improves performance significantly, while additional layers brings minor improvement. In consideration of prediction accuracy and inference speed, we set the depth of the network as 3 in the light weight model.

b) Sampling Rate: Fig. 8 shows the loss and FLOPs when using different numbers of sampling points. We linearly interpolated and re-sampled from each cycle to form l_{sample} sample points. Initially, we set l_{sample} to 32. As l_{sample} increases, the prediction accuracy augments, but FLOPs rises as well. Thus, we choose the elbow point 24 as l_{sample}, striking a balance between performance and computational costs.
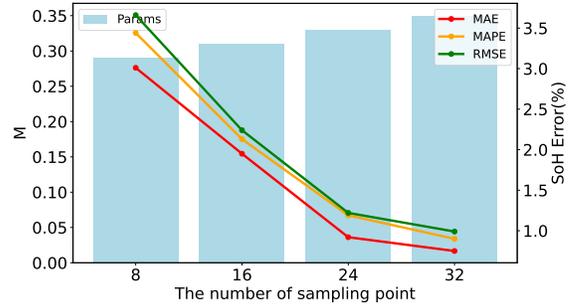
Fig. 8: The results of pruning study on model depth.

The pruning results of each module are shown in Tab. V. The joint effect of pruning both depth and sampling rate leads FLOPs and quantity of parameters to reduce by 41% and 26%, respectively. At the same time, the accuracy of the model is hardly affected.

TABLE V: Pruning Experiment

Methods	FLOPs	Params	MAE	MAPE	RMSE
Initial	0.17	0.35	0.75%	0.89%	0.95%
Depth	0.13	0.28	0.77%	0.93%	1.10%
Sampling point	0.13	0.33	0.92%	1.19%	1.22%
Overall	0.10	0.26	0.95%	1.17%	1.26%

VI. Conclusion

In this work, we present CyFormer, a generalized cyclic time sequence model with row-wise and column-wise attention mechanism. Via cyclic attention, our model effectively captures inter-cycle and intra-cycle connections. To narrow the domain gap among different working conditions, we adopt a two-stage transfer learning strategy. We also designed a light weight version of CyFormer for embedding systems by pruning. Experiments show that our model produces accurate SoH predictions using only 10% data for fine-tuning, demonstrating the effectiveness and robustness of our model. CyFormer provides a potential solution for all cyclic time sequence prediction tasts, and we expect to see more applications of our method.

References

[1] Y. Wang, Z. Sun, and Z. Chen, "Energy management strategy for battery/supercapacitor/fuel cell hybrid source vehicles based on finite state machine," *Applied energy*, vol. 254, p. 113707, 2019.

[2] R. Zou, Y. Duan, Y. Wang, J. Pang, F. Liu, and S. R. Sheikh, "A novel convolutional informer network for deterministic and probabilistic state-of-charge estimation of lithium-ion batteries," *Journal of Energy Storage*, vol. 57, p. 106298, 2023.

[3] L. Ren, J. Dong, X. Wang, Z. Meng, L. Zhao, and M. J. Deen, "A data-driven auto-cnn-lstm prediction model for lithium-ion battery remaining useful life," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3478–3487, 2021.

[4] Y. Fan, F. Xiao, C. Li, G. Yang, and X. Tang, "A novel deep learning framework for state of health estimation of lithium-ion battery," *Journal of Energy Storage*, vol. 32, p. 101741, 2020.

[5] H. Shen, X. Zhou, Z. Wang, and J. Wang, "State of charge estimation for lithium-ion battery using Transformer with immersion and invariance adaptive observer," *Journal of Energy Storage*, vol. 45, p. 103768, 2022.

[6] X. Gu, K. W. See, P. Li, K. Shan, Y. Wang, L. Zhao, K. C. Lim, and N. Zhang, "A novel state-of-health estimation for the lithium-ion battery using a convolutional neural network and transformer model," *Energy*, vol. 262, p. 125501, 2023.

[7] P. Fu, L. Chu, Z. Hou, J. Hu, Y. Huang, and Y. Zhang, "Transfer learning and vision transformer based state-of-health prediction of lithium-ion batteries," *arXiv preprint arXiv:2209.05253*, 2022.

[8] L. Biggio, T. Bendinelli, C. Kulkarni, and O. Fink, "Dynaformer: A deep learning model for ageing-aware battery discharge prediction," *arXiv preprint arXiv:2206.02555*, 2022.

[9] X. Hu, J. Jiang, D. Cao, and B. Egardt, "Battery Health Prognosis for Electric Vehicles Using Sample Entropy and Sparse Bayesian Predictive Modeling," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 4, pp. 2645–2656, Apr. 2016.

[10] Y. Zheng, J. Wang, C. Qin, L. Lu, X. Han, and M. Ouyang, "A novel capacity estimation method based on charging curve sections for lithium-ion batteries in electric vehicles," *Energy*, vol. 185, pp. 361–371, 2019.

[11] K. Lim, H. A. Bastawrous, V.-H. Duong, K. W. See, P. Zhang, and S. X. Dou, "Fading Kalman filter-based real-time state of charge estimation in LiFePO4 battery-powered electric vehicles," *Applied Energy*, vol. 169, pp. 40–48, 2016.

[12] J. Meng, L. Cai, G. Luo, D.-I. Stroe, and R. Teodorescu, "Lithium-ion battery state of health estimation with short-term current pulse test and support vector machine," *Microelectronics Reliability*, vol. 88-90, pp. 1216–1220, 2018.

[13] X. Tan, D. Zhan, P. Lyu, J. Rao, and Y. Fan, "Online state-of-health estimation of lithium-ion battery based on dynamic parameter identification at multi timescale and support vector regression," *Journal of Power Sources*, vol. 484, p. 229233, 2021.

[14] L. Cai, J. Meng, D.-I. Stroe, J. Peng, G. Luo, and R. Teodorescu, "Multiobjective Optimization of Data-Driven Model for Lithium-Ion Battery SOH Estimation With Short-Term Feature," *IEEE Transactions on Power Electronics*, vol. 35, no. 11, pp. 11 855–11 864, Nov. 2020.

[15] D. Yang, X. Zhang, R. Pan, Y. Wang, and Z. Chen, "A novel Gaussian process regression model for state-of-health estimation of lithium-ion battery using charging curve," *Journal of Power Sources*, vol. 384, pp. 387–395, 2018.

[16] J. Liu and Z. Chen, "Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Health Indicator and Gaussian Process Regression Model," *IEEE Access*, vol. 7, pp. 39 474–39 484, 2019.

[17] K. Liu, Y. Shang, Q. Ouyang, and W. D. Widanage, "A Data-Driven Approach With Uncertainty Quantification for Predicting Future Capacities and Remaining Useful Life of Lithium-ion Battery," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3170–3180, Apr. 2021.

[18] G. E. Box and G. M. Jenkins, "Some recent advances in forecasting and control," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 17, no. 2, pp. 91–109, 1968.

[19] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[20] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *CoRR*, vol. abs/1803.01271, 2018, arXiv: 1803.01271.

[21] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," *Advances in neural information processing systems*, vol. 32, 2019.

[22] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution," *arXiv preprint arXiv:2202.10054*, 2022.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[24] "battery data set", nasa ames prognostics data repository, 2007.

[25] https://c3.nasa.gov/dashlink/resources/133/.