Reliability Engineering and System Safety 230 (2023) 108897

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

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

Feature disentanglement and tendency retainment with domain adaptation for Lithium-ion battery capacity estimation

Fujin Wang [a,b], Zhibin Zhao [a,b,*], Zhi Zhai [a,b], Yanjie Guo [a,b], Huan Xi [c], Shibin Wang [a,b], Xuefeng Chen [a,b]

[a] School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, 710049, PR China
[b] The State Key Laboratory for Manufacturing Systems Engineering, Xi'an, 710061, PR China
[c] Key Laboratory of Thermo-Fluid Science and Engineering of Ministry of Education, School of Energy and Power Engineering, Xi'an Jiaotong University, Xi'an, 710049, PR China

ARTICLE INFO

Keywords:
Lithium-ion battery
Capacity estimation
Feature disentanglement
Tendency retainment
Domain adaptation
Contrastive learning

ABSTRACT

Online capacity estimation of lithium-ion batteries plays an important role in battery management systems. Accurate estimation of the current capacity of the battery is helpful for predictive maintenance. Data-driven capacity estimation methods have become increasingly popular in recent years. However, most data-driven approaches assume that the battery degradation data used to train and test obeys the same distribution. In practical applications, there is often a discrepancy in the distributions of the training set and the test set due to differences in the internal chemistry of batteries and various operating conditions. The existing methods for domain adaptive capacity estimation force the global features of the source and target domains to be aligned, but do not take into account the domain-specific information. Thus, models may not extract domain-invariant representation. To address this problem, We propose a feature **D**isentanglement and tendency **R**etainment network (DR-Net) for domain adaptive capacity estimation. Specifically, our DR-Net can disentangle the extracted features from raw data into domain-invariant shared features and domain-specific private features while retaining degradation trend information. Experimental results show that the proposed method achieves more competitive results in both estimation accuracy and robustness than other state-of-the-art methods. Our code is available at: https://github.com/wang-fujin/battery_capacity_estimation.

1. Introduction

Lithium-ion batteries are widely used in electric vehicles, aerospace, electronic equipment and photovoltaic systems due to their advantages of high energy density, low cost, low self-discharge rate, and long service life [1,2]. However, as the number of times when the battery is charged and discharged increases, the performance of the battery will gradually decrease until it fails. To ensure the safe and reliable operation of the equipment and make a predictive maintenance plan, it is significant to conduct real-time online monitoring and evaluation of the state-of-health (SOH) and state-of-charge (SOC) of the battery.

For lithium-ion batteries, capacity is an indicator that reflects the state of battery degradation. The battery life is defined as the number of cycles in which the capacity drops to 80% of the initial capacity [3]. Therefore, accurate capacity estimation of the current state is helpful for the predictive maintenance of batteries. In the battery management system (BMS), the current, voltage, and temperature can usually be collected. The capacity can be evaluated via the data monitored by the BMS, which can provide a reference for health management and achieve true predictive maintenance [4].

Recent studies have reported various methods for capacity estimation and remaining useful life (RUL) prediction of lithium-ion batteries. Generally, these methods can be divided into two categories: model-based methods and data-driven methods. Model-based methods [5–9] require expert knowledge, and battery-related design parameters are required for modeling. The procedures of model-based methods are often complex, which limits their wide applications. Data-driven methods [10] are popular for battery capacity estimation and RUL prediction, which rely only on past degradation data to build battery aging models. Xu et al. [11] proposed a stacked denoising autoencoder method to directly predict battery life using battery features. Ardeshiri et al. [12] proposed a stacked bidirectional long short-term memory network (LSTM) based on extreme gradient boosting to predict capacity degradation trajectory. Li et al. [13] proposed a multi-model framework based on particle filtering and support vector regression

* Corresponding author at: School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, 710049, PR China.
E-mail address: zhaozhibin@xjtu.edu.cn (Z. Zhao).

https://doi.org/10.1016/j.ress.2022.108897
Received 13 July 2022; Received in revised form 7 October 2022; Accepted 8 October 2022
Available online 17 October 2022

to achieve multi-step ahead estimation of battery capacity and RUL. Roman et al. [14] designed a machine learning pipeline for estimating battery capacity degradation and evaluated it on 179 batteries cycled under different conditions. Li et al. [15] proposed a SOH evaluation method for lithium-ion batteries based on incremental capacity analysis to achieve high prediction accuracy. Aiming at the influence of the capacity regeneration phenomenon, some scholars [16–19] have done corresponding research and proposed various SOH estimation models considering the capacity regeneration phenomenon. To achieve higher prediction accuracy, He et al. [20] proposed a hybrid data-driven approach. They used wavelet transform to denoise the initial features, and random forest was used to select important features, and finally proposed a novel mogrifier-LSTM for capacity prediction. What is more, Zhang et al. [21] proposed a hybrid parallel residual convolutional neural network (CNN) model for RUL prediction, which can reliably predict battery RUL based on sparse data corresponding to only 20% charging capacity.

Although data-driven methods can establish capacity estimation models based on historical battery degradation data, they are all based on the assumption that the training data (source domain) and test data (target domain) obey the same distribution [22]. However, such an assumption is often not satisfied in reality, as changes in operating conditions will inevitably lead to different data distributions, also called domain shift, as shown in Fig. 1. Once the operating conditions change, these methods will fail. Therefore, some scholars used transfer learning or domain adaptation to solve the problem of domain shift. These methods have been widely used in the field of image classification and have achieved good results [23–25]. In the field of battery capacity estimation and RUL prediction, some scholars have also done corresponding research.

From the perspective of data preprocessing, Li et al. [26] proposed a data preprocessing method based on the Maximum Mean Discrepancy (MMD) to reduce the discrepancy between different data distributions, thereby realizing knowledge transfer across domains. Tan et al. [27] established a feature expression scoring rule, selecting features with high scores to train the model to have better generalization performance. In addition, pre-training and fine-tuning techniques were also widely used in battery capacity estimation tasks [28–32]. However, these methods require labeled data from both source and target domains, and sometimes it is difficult to obtain labeled data from the target domain in many practical applications. Kim et al. [33] used variational inference to predict the battery RUL and used uncertainty estimation to predict battery degradation patterns, which was also involved with fine-tuning techniques. Ye et al. [34] proposed an unsupervised feature alignment method based on the fusion of MMD and correlation alignment for battery SOH estimation. Wang et al. [35,36] proposed a degradation trend alignment method to align battery degradation data in different domains, thereby realizing knowledge transfer across domains. Han et al. [37] integrated the domain adaptation layer based on MMD into LSTM to achieve the alignment of the degraded features of the source and target batteries, but the proposed method still require a small amount of labeled target domain data. In short, methods reviewed above all require the target domain data to be labeled.

The essential goal of domain adaptation is to find domain-invariant features, so that the knowledge learned in the source domain can be transferred to the target domain. Among the existing domain adaptation methods for battery capacity estimation, some scholars [34,37] used MMD to measure the distance between source and target domains and decreased this distance during the optimization process. Ye et al. [34] achieved distribution alignment via adding a discriminator and making the discriminator unable to distinguish which domain the features come from through adversarial training. Based on these methods, it may make the features extracted from the source and target domains similar with each other, but what is obtained may not be good domain-invariant features. In classification tasks (e.g., image classification), there is no trend information between classes of the same domain. However, battery capacity estimation is a regression problem, and there is degradation trend information within batteries during the performance degradation from health to fault. Forcing the global features of two domains to be aligned may lead the model to extract similar features and weaken the degradation trend information, so the model may fail to learn a perfect domain-invariant representation. However, existing domain-adaptive capacity estimation methods [26,27,33,34,37] all aim to align the global features of the two domains without considering the specific features in each domain.

In the battery degradation data, in addition to the shared features that reflect the battery degradation trend, there is private information in different domains due to various charging/discharging strategies and individual difference of batteries. Therefore, we proposed a model to extract shared features while ignoring the influence of private information. Specifically, we propose a feature **D**isentanglement and tendency **R**etainment network (DR-Net) for unsupervised domain adaptive capacity estimation of lithium-ion batteries, which can disentangle the extracted features into domain-invariant shared features and domain-specific private features while retaining degradation trend information. To be specific, our DR-Net consists of four modules: a Shared Encoder, a Private Encoder, a Decoder, and a Predictor. Instead of aligning global feature distributions, we extract shared and private features via the Shared Encoder and the Private Encoder, respectively, and align only the shared features. In addition, we impose constraint to make the shared features and private features dissimilar. Thus, the purpose of feature disentanglement is achieved. To ensure that the extracted shared and private features can retain useful information of the raw data, the sum of the shared and private features should be able to reconstruct the raw data as much as possible. So the features are decoded by a Decoder, and the mutual information between the decoded data and the raw data is forced to be maximized. Additionally, due to the various data distributions and statistical characteristics between different domains, we use different BatchNormalization layers for the source and target domains to make the model easier to be trained [38]. So a better domain-invariant representation can be learned for domain adaptive capacity estimation. Our contributions are listed as follows:

1. A DR-Net is proposed for unsupervised domain adaptive capacity estimation of lithium-ion batteries. we constrain the DR-Net from structure so that it considers both shared and private information. By imposing different constraints, the extracted features are disentangled into domain-invariant and domain-specific features. To our best knowledge, the domain-specific information is first considered for battery capacity estimation.
2. To make the features more representative, i.e. retaining degradation trend information and ignoring the noise information, the contrastive learning method is used in the proposed model. The domain-invariant shared features which retain degradation trend information are finally used for capacity estimation.

Through comparisons with other state-of-the-art methods, the effectiveness and superiority of the proposed method are verified. The experimental results also show that feature disentanglement and tendency retainment can improve the performance in domain adaptive capacity estimation.

The reminder of this paper is organized as follows. Section 2 gives the problem statement, methodology overview, and the proposed capacity estimation model, respectively. Section 3 presents the designed experiments and compared methods for verification. Section 4 provides the comprehensive results of compared studies and discussions. Finally, the conclusions are drawn in Section 5.

2. Methodology

2.1. Preliminary

To clearly formulate the problem, the basic notations are defined first. Let $D^s = \left\{\mathbf{x}_i^s, y_i^s\right\}_{i=1}^{N_s}$ represents the labeled source domain, where

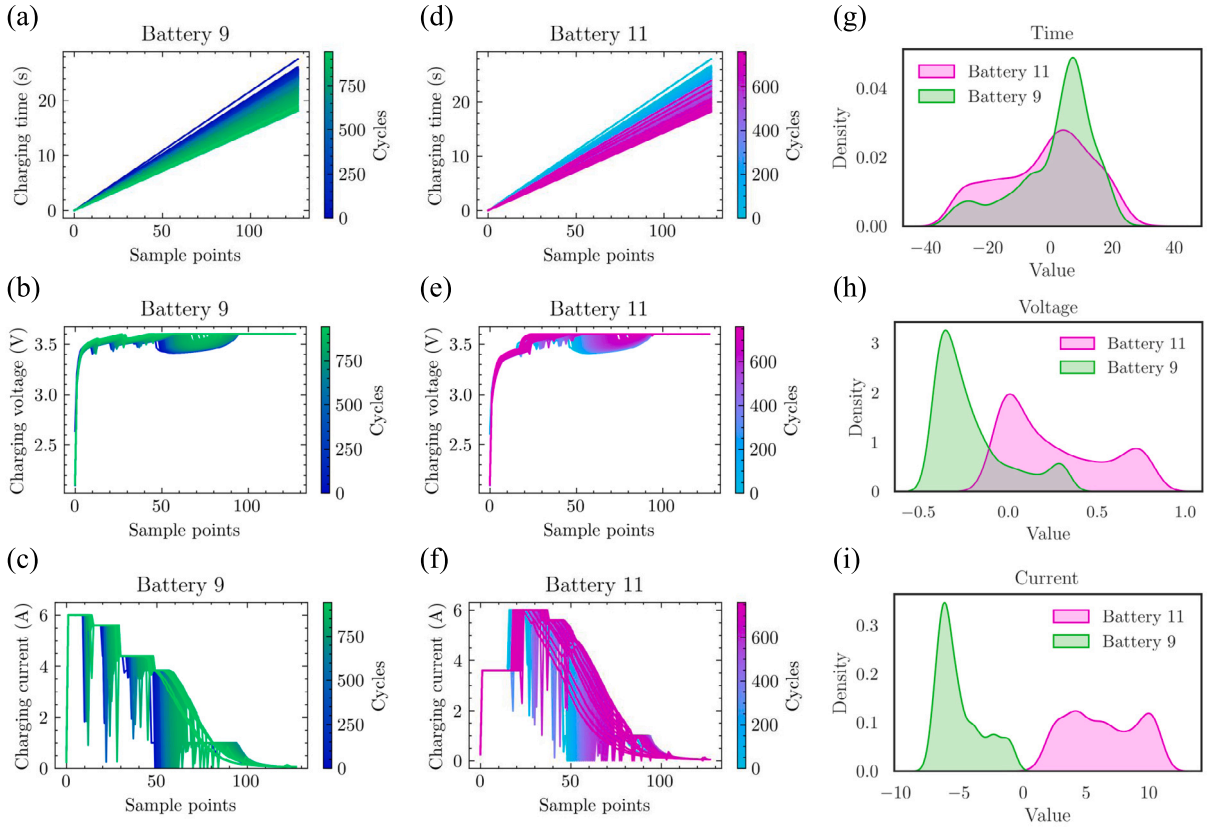


Fig. 1. Visualization of randomly selected data from two batteries (two domains). (a)–(f) are the time, voltage, and current data during the charging process of two batteries, respectively. (g)–(i) are the corresponding kernel density estimations, respectively.

$\mathbf{x}_i^s$ is the ith sample, y_i^s denotes the corresponding capacity of $\mathbf{x}_i^s$, N_s is the number of samples in the source domain. $D^t = \{\mathbf{x}_i^t\}_{i=1}^{N_t}$ represents the unlabeled target domain. $P^s(\mathbf{x}^s)$ and $P^t(\mathbf{x}^t)$ denote the marginal probability distributions of source and target domains, respectively. The battery data distributions in source and target domains are different due to various working conditions, that is $P^s(\mathbf{x}^s) \neq P^t(\mathbf{x}^t)$, as shown in Fig. 1(g)–(i). The unsupervised domain adaptation problem aims to transfer knowledge from the labeled source domain to the unlabeled target domain.

In this paper, the voltage, current, and time data during charging process are used to estimate capacity. So a sample $\mathbf{x}_i$ consists of voltage, current, and time from the ith charging cycle. To take into account the influence of previous charging cycles, a sliding window method with window length L is used to process the sample $\mathbf{x}_i$ as $\mathbf{X}_i = \mathbf{x}_{i:i-L}$. The final capacity estimation model is defined as $\hat{y}_i = f(\mathbf{X}_i)$.

It is worth noting that we use the data from the charging phase to estimate capacity rather than the data from the discharging phase. Such a setting is more in line with real industry application due to the fact that the charging phase is more controllable and data collection is easier.

2.2. Methodology overview

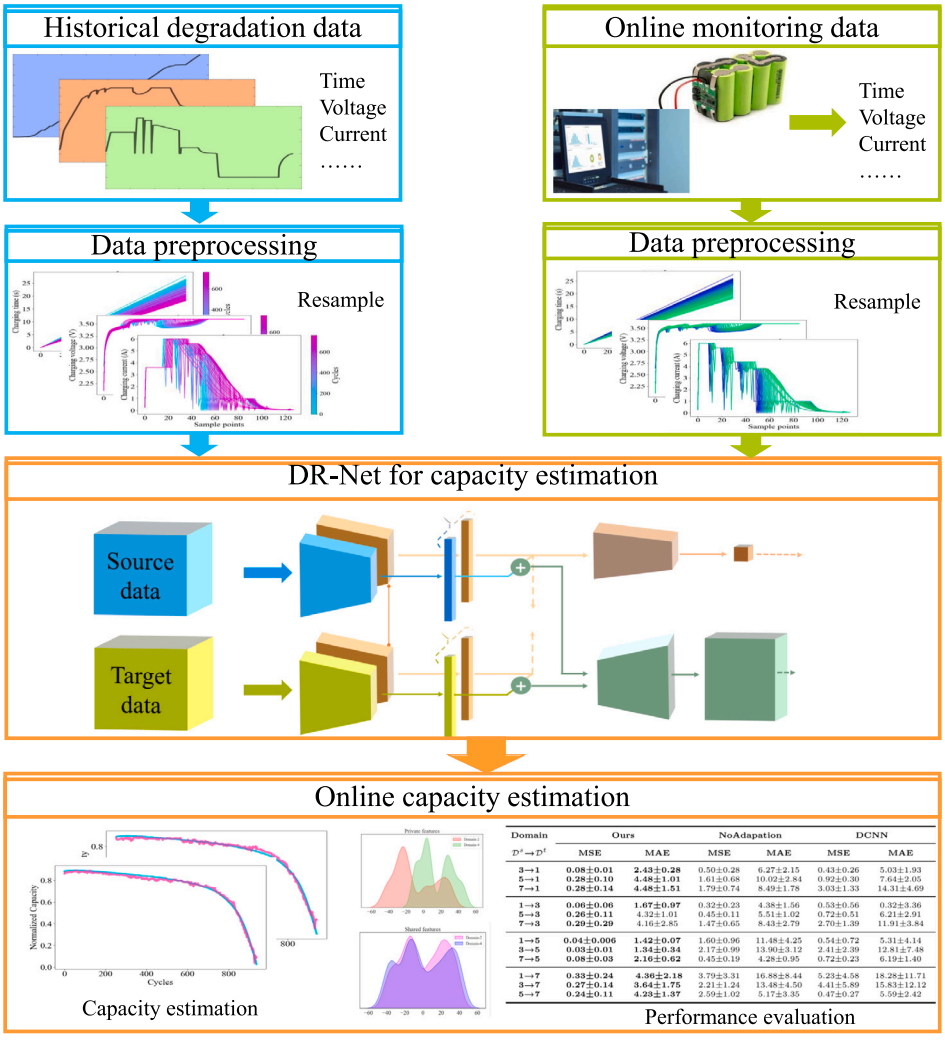
An overview of the main procedure of the proposed method is shown in Fig. 2. First, the historical degradation data is collected. The data for each cycle is resampled to 128 points at equal time intervals since the number of points for each cycle is different. For battery data from the target domain, the same resampling strategy is used. Next, both the labeled source domain data and the unlabeled target domain data are used to train the proposed model simultaneously. Finally, the capacity estimation and performance analysis of the target domain batteries are carried out.

2.3. Proposed structure

To learn a better domain-invariant representation for domain adaptive capacity estimation of lithium-ion batteries, we propose a DR-Net which can consider both shared and private information. Our DR-Net consists of four modules: a Shared Encoder, a Private Encoder, a Decoder, and a Predictor. The Shared Encoder and the Private Encoder is used to extract domain-invariant shared features and domain-specific private features. We constrain the shared and private features of the same domain to make them dissimilar, and align the shared features from different domains to achieve feature disentanglement. To ensure that the extracted shared and private features can well retain useful information of the raw data, the sum of the shared and private features should be able to reconstruct the raw data as much as possible. So the features are decoded by a Decoder, and the mutual information between the decoded data and the raw data is guaranteed to be maximized. The battery capacity is then estimated by a Predictor with the shared features as inputs. The complete network structure is shown in Fig. 3.

2.3.1. Feature extraction and disentanglement

The proposed model adopts a one-dimensional residual CNN similar to ResNet [39] as a deep feature extractor, i.e. Encoder. To ensure that the model could learn domain-invariant and domain-private features, both the Shared Encoder and the Private Encoder are used for source and target domains. The structure and number of parameters of all Encoders are the same, and the Shared Encoders in source and target domains share the same parameters, denoted as $E_s(\cdot)$. $E_p^s(\cdot)$ and $E_p^t(\cdot)$ denote the Private Encoder of source and target domains, respectively.



Domain	Ours		NoAdapation		DCNN	
$\mathcal{D}^s \to \mathcal{D}^t$	MSE	MAE	MSE	MAE	MSE	MAE
3→1	**0.08±0.01**	**2.43±0.28**	0.50±0.28	6.27±2.15	0.43±0.26	5.03±1.93
5→1	**0.28±0.10**	**4.48±1.01**	1.61±0.68	10.02±2.84	0.92±0.30	7.64±2.05
7→1	**0.28±0.14**	**4.48±1.51**	1.79±0.74	8.49±1.78	3.03±1.33	14.31±4.69
1→3	**0.06±0.06**	**1.67±0.97**	0.32±0.23	4.38±1.56	0.53±0.56	0.32±3.36
5→3	**0.26±0.11**	4.32±1.01	0.45±0.11	5.51±1.02	0.72±0.51	6.21±2.91
7→3	**0.29±0.29**	4.16±2.85	1.47±0.65	8.43±2.79	2.70±1.39	11.91±3.84
1→5	**0.04±0.006**	**1.42±0.07**	1.60±0.96	11.48±4.25	0.54±0.72	5.31±4.14
3→5	**0.03±0.01**	**1.34±0.34**	2.17±0.99	13.90±3.12	2.41±2.39	12.81±7.48
7→5	**0.08±0.03**	**2.16±0.62**	0.45±0.19	4.28±0.95	0.72±0.23	6.19±1.40
1→7	**0.33±0.24**	**4.36±2.18**	3.79±3.31	16.88±8.44	5.23±4.58	18.28±11.71
3→7	**0.27±0.14**	**3.64±1.75**	2.21±1.24	13.48±4.50	4.41±5.89	15.83±12.12
5→7	**0.24±0.11**	**4.23±1.37**	2.59±1.02	5.17±3.35	0.47±0.27	5.59±2.42

Fig. 2. The framework of the proposed methodology.

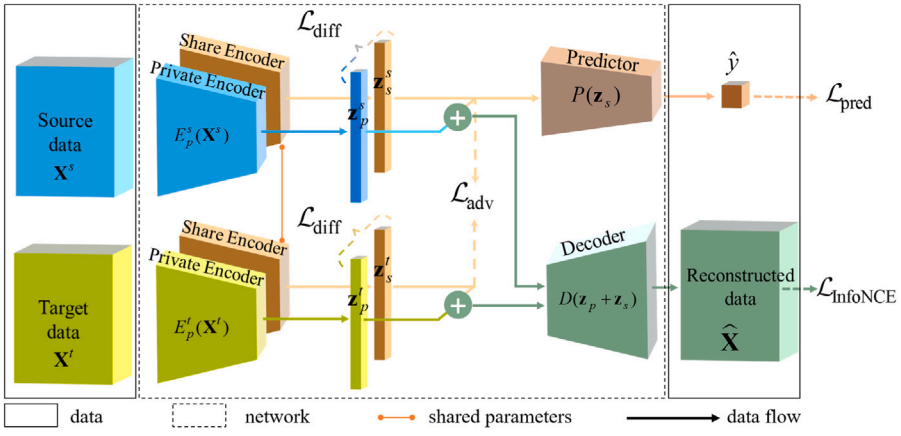


Fig. 3. The structure of the proposed DR-Net. A Shared Encoder $E_s(\cdot)$ is used to extract domain-invariant features for both source and target domains. A Private Encoder (one for each domain) $E_p(\cdot)$ is used to extract domain-specific features. A shared Decoder $D(\cdot)$ is used to reconstruct raw data by both the private and shared features. A Predictor $P(\cdot)$ is used to estimate the capacity $\hat{y}$ of the input $\mathbf{X}$.

The process of feature extraction can be expressed by the following formulas:

$$\begin{cases} \mathbf{z}_s^s = E_s(\mathbf{X}^s) \\ \mathbf{z}_p^s = E_p^s(\mathbf{X}^s), \end{cases} \tag{1}$$

$$\begin{cases} \mathbf{z}_s^t = E_s(\mathbf{X}^t) \\ \mathbf{z}_p^t = E_p^t(\mathbf{X}^t), \end{cases} \tag{2}$$

where $\mathbf{z}_s^s$ and $\mathbf{z}_p^s$ represent shared and private features from the source domain; $\mathbf{z}_s^t$ and $\mathbf{z}_p^t$ represent shared and private features from the target domain; $\mathbf{X}^s$ and $\mathbf{X}^t$ denote source and target data.

To encourage shared and private features to be different, we add orthogonality constraints [40] between them. Let $\mathbf{Z}_s^s$ and $\mathbf{Z}_p^s$ be matrices whose rows consist of $\mathbf{z}_s^s$ and $\mathbf{z}_p^s$, respectively. Similarly, Let $\mathbf{Z}_s^t$ and $\mathbf{Z}_p^t$ be matrices whose rows consist of $\mathbf{z}_s^t$ and $\mathbf{z}_p^t$. The difference loss between shared and private features is defined as:

$$\mathcal{L}_{\text{diff}} = \|\mathbf{Z}_s^{s\top}\mathbf{Z}_p^s\|_F^2 + \|\mathbf{Z}_s^{t\top}\mathbf{Z}_p^t\|_F^2, \tag{3}$$

where $\|\cdot\|_F^2$ represents the squared Frobenius norm. Minimizing $\mathcal{L}_{\text{diff}}$ can make the extracted features dissimilar between the Shared Encoder and the Private Encoder.

Unlike the constraints between shared and private features, we expect that the features extracted by the Shared Encoder from the source and target domains are similar with each other, i.e. preserving degradation trend information. Therefore, the similarity loss is used to achieve the purpose of extracting domain-invariant features. We use an adversarial approach to train the Shared Encoder for feature confusion. Concretely, we train the Shared Encoder to extract features from the source and target domains such that the *domain discriminator* cannot distinguish which domain the features come from. The optimization objective of adversarial training between the *domain discriminator* and the Shared Encoder can be expressed as:

$$\begin{aligned} \min_{E_s}\max_{G_D} \mathcal{L}_{\text{adv}} = &-\mathbb{E}_{\mathbf{X}^s\in D^s}\log\left[G_D\left(E_s\left(\mathbf{X}^s\right)\right)\right] \\ &-\mathbb{E}_{\mathbf{X}^t\in D^t}\log\left[1-G_D\left(E_s\left(\mathbf{X}^t\right)\right)\right] \end{aligned} \tag{4}$$

where $G_D(\cdot)$ denotes a *domain discriminator*. Eq. (4) involves the process of alternating minimum and maximum optimization. For simplicity, according to [41], a *gradient reversal layer* can be added to solve the above optimization problem, which can change the sign of the gradient without adding new parameters.

2.3.2. Tendency retainment via contrastive learning

To ensure that the private features $\mathbf{z}_p$ and shared features $\mathbf{z}_s$ are sufficiently representative, while retaining useful information of the original data as much as possible, the idea of contrastive learning is applied. Specifically, A Decoder is added to the DR-Net, and we rely on the InfoNCE loss [42] to maximize the mutual information between the decoded data and the input data.

Given the input $\mathbf{X}_i$, the shared features $\mathbf{z}_{s,i}$ and private features $\mathbf{z}_{p,i}$ can be obtain by Eqs. (1) and (2). Note that the data in both source and target domains are decoded by the Decoder, and for simplicity, all superscripts representing the domain label in this part are omitted. We applied a Decoder $D(\cdot)$ to decode the features as:

$$\hat{\mathbf{X}}_i = D(\mathbf{z}_{p,i} + \mathbf{z}_{s,i}). \tag{5}$$

The mutual information of the raw data $\mathbf{X}_i$ and the decoded data $\hat{\mathbf{X}}_i$ is defined as:

$$I(\mathbf{X}_i, \hat{\mathbf{X}}_i) = \sum_{\mathbf{X}_i,\hat{\mathbf{X}}_i} p(\mathbf{X}_i, \hat{\mathbf{X}}_i)\log\frac{p(\mathbf{X}_i|\hat{\mathbf{X}}_i)}{p(\mathbf{X}_i)}. \tag{6}$$

It is difficult to directly model mutual information. Followed by [42], we model a density ratio function which can reflect the mutual information between $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$ as follows:

$$f(\mathbf{X}_i, \hat{\mathbf{X}}_i) \propto \frac{p(\mathbf{X}_i|\hat{\mathbf{X}}_i)}{p(\mathbf{X}_i)}. \tag{7}$$

By maximizing the mutual information between $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$, the extracted features can retain the information of the raw data as much as possible. For simplicity, we use Eq. (8) to estimate the density ratio function,

$$f(\mathbf{X}_i, \hat{\mathbf{X}}_i) = \mathbf{X}_i^\top\hat{\mathbf{X}}_i. \tag{8}$$

To maximize the density ratio function, we jointly optimize the Encoder and the Decoder by using a contrastive estimation loss. For one sample $\mathbf{X}_i$ in a mini-batch, that is, $\mathbf{X}_i \in \mathcal{B}$, the positive pairs consist of $(\mathbf{X}_i, \hat{\mathbf{X}}_i)$, and the negative pairs consist of $(\mathbf{X}_j, \hat{\mathbf{X}}_i)$, $i \neq j$. Therefore, the optimization objective in a mini-batch is set as:

$$\mathcal{L}_{\text{InfoNCE}} = -\mathop{\mathbb{E}}_{\mathbf{X}_i\in\mathcal{B}}\left[\log\frac{\exp(f(\mathbf{X}_i, \hat{\mathbf{X}}_i))}{\sum_{i\neq j,\mathbf{X}_j\in\mathcal{B}}\exp(f(\mathbf{X}_j, \hat{\mathbf{X}}_i))}\right]. \tag{9}$$

By minimizing the above InfoNCE loss, the mutual information between $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$ can be maximized, thus ensuring that the extracted features retain the most information of the raw data.

Due to the large difference in data distributions and statistical characteristics between the source and target domains (as shown in Fig. 1), inspired by [38], we further adopt different BatchNormalization (denoted as DBN) layers for source and target domains in the Shared Encoder, respectively. The BatchNormalization layer only counts the global mean and variance of training samples, so it does not add new parameters to the Encoder. Based on this, a better domain-invariant representation which retaining degradation trend information can be learned by the Shared Encoder.

2.3.3. Capacity estimation

As shown in Fig. 3, since only the data from the source domain contains labels, the Predictor is trained by domain-invariant features extracted from the source domain. The predictor $P(\cdot)$ consists of two Linear layers, with a ReLU layer embedded in the middle. The domain-invariant features are used as inputs to the Predictor to obtain the capacity estimate $\hat{y}$, that is, $\hat{y} = P(\mathbf{z}_s^s)$. The loss function is the mean square error loss, which is formulated as:

$$\mathcal{L}_{\text{pred}} = \frac{1}{N_s}\sum_{i=1}^{N_s}(\hat{y}_i - y_i)^2. \tag{10}$$

2.4. Overall and learning

The overall architecture of DR-Net is shown in Fig. 3, and the specific details of each part are given in Table 1. The final optimization objective can be formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \alpha\mathcal{L}_{\text{InfoNCE}} + \beta\mathcal{L}_{\text{diff}} + \gamma\mathcal{L}_{\text{adv}}, \tag{11}$$

where α, β, γ are trade-off parameters. By minimizing Eq. (11), the weight parameters of the DR-Net can be updated. When the training step is complete, the Encoder of the target domain and the Predictor are combined for online capacity estimation.

It is worth noting that the offline training parameters of our DR-Net are 6.4 M, and the online test parameters are only 2.06 M. For intuitive presentation, we give that the parameters of VGG [43] is 138.3 M, and the parameters of ResNet-18 [39] is 11.7 M.

3. Experimental section

In this section, we carried out experiments to verify the effectiveness of the proposed method. And comparative experiments were conducted to compare with other state-of-the-art methods.

Table 1
Details of the DR-Net. The structure of the EncoderBlock is similar to the BasicBlock in ResNet [39], which consists of Conv1d, BatchNorm1d, SiLU, Conv1d, and BatchNorm1d. ConvTranspose1d is used to replace the Conv1d in DecoderBlock, and the structures of other parts are the same as EncoderBlock. A residual connection is added in each EncoderBlock and DecoderBlock.

	Basic block	Input size	Output size
Encoder	Conv1d+BN+SiLU	(24,128)	(16,128)
	EncoderBlock1	(16,128)	(32,128)
	EncoderBlock2	(32,128)	(64,64)
	EncoderBlock3	(64,64)	(128,32)
	EncoderBlock4	(128,32)	(192,16)
	Linear+Dropout+Linear	(192,16)	256
Decoder	Linear	256	1024
	DecoderBlock1	1024	(32,64)
	DecoderBlock2	(32,64)	(32,128)
	Conv1d	(32,128)	(24,128)
Predictor	Linear	256	128
	ReLU		
	Linear	128	1

Table 2
According to different charging protocols, the batteries were divided into different domain.

Domain	Charge protocol 0→ 20%→ 40%→ 60%→ 80%	Battery ID
D_1	3.6→ 6→ 5.6→ 4.755 C	11,12,27,29,38
D_2	4.8→ 5.2→ 5.2→ 4.16 C	1,2,10,20,42
D_3	6→ 5.6→ 4.4→ 3.834 C	9,21,22,31,36
D_4	8→ 4.4→ 4.4→ 3.94 C	13,16,23,24,47

3.1. Battery data

The open-access LiFePO4 battery cycle data in [44] are served as our experimental data. These batteries have a nominal capacity of 1.1 Ah and a nominal voltage of 3.3 V. They were charged to 80% SOC via four stages of fast-charging (constant current charging mode, CC), and then fully charged in constant current and constant voltage (CC-CV) mode. In the first three fast-charging stages, 20% SOC was charged in each of stage (0%–20%, 20%–40%, and 40%–60%, respectively) at a predetermined rate. In the fourth stage, the charging rate was determined by guaranteeing that the total charging time of four stages was 10 min. All batteries were discharged with a constant current of 4 C. More details could be found in [44]. There are a total of five batches of batteries in this data set. The run-to-failure data of 45 batteries from the last batch were selected to validate the proposed method.

Depending on the charging protocol, the batteries were divided into different domains. We finally selected 20 batteries, which were divided into 4 domains, denoted as D_1, D_2, D_3, and D_4. Each domain contains data for 5 batteries. The specifications were listed in Table 2.

3.2. Data preparation

Since the data points of each cycle are different, to facilitate the model processing, we resampled the data of each cycle to the same number of points. Specifically, 128 points were resampled at equal time intervals from the time, current, and voltage data of the charging process for each cycle, that is, $\mathbf{x}_i \in \mathbb{R}^{128\times3}$. At the same time, the window length L was set to 8, so $\mathbf{X}_i \in \mathbb{R}^{128\times24}$. The units and scales of the data are different, and thus the normalization method is often used to transpose the data to the certain scale to make the training process more stable. In this paper, we used min–max normalization to transpose the data to the range of [0, 1].

3.3. Implementation details

Our model was learned by minimizing the loss defined in Eq. (11), and SGD optimizer was used in the training phase. Through grid search technique, the trade-off parameter in Eq. (11) can be quickly obtained. In our DR-Net, the convolution kernel size of Conv1d is 3 and the stride is 1. Other hyperparameters were listed in Table 3. Our DR-Net are implemented in Pytorch platform on GeForce RTX 3090 GPU with 24 GB memory. More details can be found in our source code: code link.

Table 3
Experimental hyperparameters.

Parameters	Value	Parameters	Value
α	0.05	Optimizer	SGD
β	0.001	Momentum	0.9
γ	80	Learning rate	0.002
Epoch	300	Batch size	64
Early stop	20	Dropout rate	0.5

To qualify the performance of the proposed model, two performance metrics were employed in this paper. The mean square error (MSE) and the mean absolute error (MAE) were used to measure the error between the predicted value $\hat{y}$ and the true value y. The two evaluation metrics are defined as follows:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{12}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|, \tag{13}$$

where N denotes the number of samples used for testing in the target domain.

3.4. Compared methods

To clearly illustrate the superiority of the proposed method, we compared the proposed method with the state-of-the-art methods.

(1) **NoAdaptation**: We used the Encoder and Predictor in our DR-Net as a backbone to compose a new model, which takes the data of the source domain as the training set and tests it directly on the target domain. The main purpose of this setting is to compare the capacity estimation results of battery data under different operating conditions without *domain adaptation*. Other experimental conditions are the same as those in the proposed method.
(2) **LSTM**: LSTM, with the ability to model time series, is widely used for battery capacity estimation and RUL prediction [12,45]. In this paper, we compare the proposed method with LSTM. Like **NoAdaptation**, LSTM is trained on the source domain, and then directly tested on the target domain.
(3) **DCNN**: We implemented a network structure similar to the model proposed in [46]. It is worth noting that, expect for the hyperparameters, the other settings of DCNN are the same as in [46].
(4) **DANN**: DANN is a popular structure in domain adaptation, first proposed in [41]. It used the same *feature extractor* to extract features from the source and target domains, and then aligned the features through adversarial training. We implemented a DANN architecture with our Encoder as a feature extractor. Other settings are same as those in the proposed method.
(5) **MMD**: MMD is a distance metric, which is commonly used in domain adaptation. It can effectively measure the first-order distribution divergence in Reproducing Kernel Hilbert Space. During model optimization, the source and target domains are aligned by narrowing this distribution. In previous works, [34] and [37] both used MMD-based methods to achieve domain-adaptive capacity estimation. For comparison, we replaced the adversarial training in **DANN** with MMD distance to update the model.

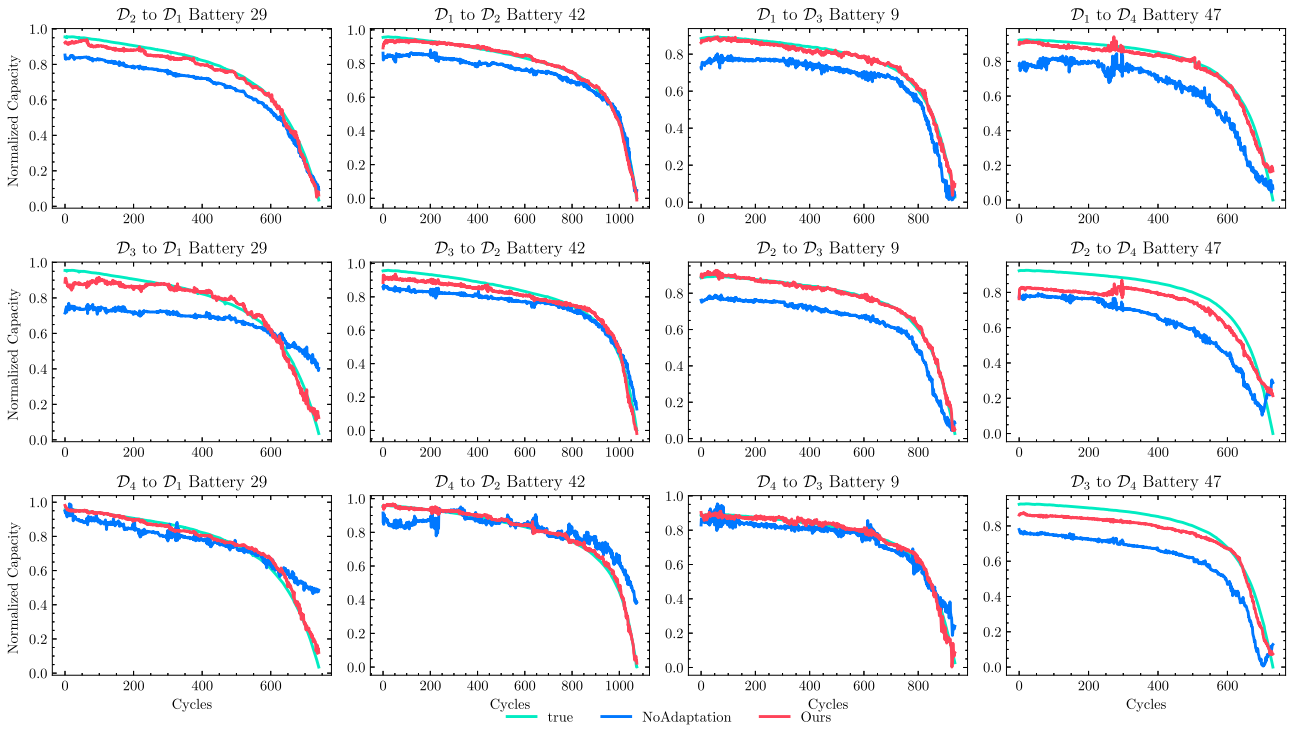


Fig. 4. The results of capacity estimation. $\mathcal{D}_i$ to $\mathcal{D}_j$ means to use $\mathcal{D}_i$ as the source domain and $\mathcal{D}_j$ as the target domain to conduct experiments. Red curves are the capacity estimation results of the **DR-Net (proposed)**, and blue curves are the corresponding results of **NoAdaptation**. Two methods have the same backbone.

To ensure fairness, the above methods all adopted an end-to-end method. That is, the inputs of the model were the original time, current, and voltage data during charging process, which were resampled to 128 points for each cycle. The output was the estimated capacity.

4. Results and discussions

In this section, we analyzed and discussed the experimental results of Section 3, and performed ablation analysis on the proposed method to verify the effectiveness of each module.

4.1. Capacity estimation results

As shown in Table 2, there are a total of battery charging data under 4 operating conditions (domains, $\mathcal{D}$). According to the experimental design, any two conditions can be combined to form a domain adaptation task, so a total of 12 experiments were carried out. Since there are 5 batteries in each task, for simplicity, we choose 12 batteries, one for each domain adaptation task, to show the results of capacity estimation and perform detailed analysis.

Fig. 4 presents the comparison results of capacity estimation between the proposed method and the method without domain adaptation (NoAdaptation). It is worth noting that both methods use the same backbone and the same training parameters. It can be clearly seen that the capacity values predicted by the proposed model is closer to the true capacity value, while there are significant errors between the estimated values by NoAdaptation and the true ones.

Due to the various battery charging and discharging protocols in different domains, that is, domain shift, common data-driven methods often show performance degradation in domain adaptive capacity estimation tasks. The degree of degradation is not easy to quantify, so it is not suitable to directly use models trained in the source domain in domain adaptive capacity estimation tasks. The experimental results also prove that the domain adaptation is a suitable and also necessary method for the domain adaptive capacity estimation task.

4.2. Comparisons with other methods

To further demonstrate the effectiveness and superiority of the proposed method, we compared the proposed method with state-of-the-art methods. The comparison results are given in Table 4. To eliminate accident and improve the credibility of the experiments results, each value in the table was obtained by conducting five experiments and then taking the average. A same strategy was used in Tables 5 and 6.

As shown in Table 4, our proposed method obtained best results in almost all tasks. Among them, NoAdaptation, LSTM, and DCNN got the results without domain adaptation, and it can be seen that the results of these methods are the worst. MMD and DANN were commonly used methods for domain adaptation tasks, and their results were worse than those of our method and better than those of the other three methods. It is worth mentioning that the structure of the backbone is the same for all methods except DCNN and LSTM.

In addition, the difficulty of the domain adaptation task vary across domains due to the different degrees of discrepancy. We found that the larger the discrepancy in data distributions between the source and target domains, the more challenging the knowledge transfer. However, the proposed method can still achieve knowledge transfer between two distant domains well. For example, $\mathcal{D}_1$ and $\mathcal{D}_4$ may be the two domains with the most dissimilar distributions, but our model could still achieve good results.

It can also be observed from Table 4 that the MMD and DANN approaches assume that the Encoder can learn a good domain-invariant representation, and thus simply force the Encoder to extract similar features, which can harm the performance of the model. In the domain adaptive battery capacity estimation task, in addition to the shared features that can reflect battery degradation trends, private information exists in different domains due to various charging and discharging strategies and individual difference of battery. Therefore, simply forcing features extracted by the Encoder to be similar may not get good results. In our proposed method, more competitive results can be obtain due to the features disentanglement and tendency retainment.

Table 4
The results of the proposed method and other state-of-the-art methods. The best results have been **bolded**.

Metrics	Domain $D^s \rightarrow D^t$	(DR-Net) Ours	NoAdapation	LSTM	DCNN	MMD	DANN
MSE	**2 → 1**	**0.08±0.01**	0.50±0.28	0.20±0.06	0.43±0.26	0.20±0.08	0.16±0.08
	3 → 1	**0.28±0.10**	1.61±0.68	1.20±0.90	0.92±0.30	0.45±0.15	0.95±0.25
	4 → 1	**0.28±0.14**	1.79±0.74	1.17±0.80	3.03±1.33	0.66±0.25	0.86±0.47
	1 → 2	**0.06±0.06**	0.32±0.23	0.38±0.03	0.53±0.56	0.13±0.10	0.11±0.05
	3 → 2	**0.26±0.11**	0.45±0.11	0.39±0.02	0.72±0.51	**0.26±0.11**	0.43±0.15
	4 → 2	**0.29±0.29**	1.47±0.65	1.48±0.81	2.70±1.39	0.32±0.10	0.48±0.16
	1 → 3	**0.04±0.006**	1.60±0.96	1.27±0.23	0.54±0.72	0.12±0.09	0.17±0.07
	2 → 3	**0.03±0.01**	2.17±0.99	0.54±0.29	2.41±2.39	0.14±0.05	0.27±0.12
	4 → 3	**0.08±0.03**	0.45±0.19	1.17±0.48	0.72±0.23	0.17±0.04	0.15±0.10
	1 → 4	**0.33±0.24**	3.79±3.31	0.59±0.06	5.23±4.58	0.41±0.28	0.92±0.43
	2 → 4	**0.27±0.14**	2.21±1.24	0.67±0.20	4.41±5.89	0.46±0.29	1.39±1.30
	3 → 4	**0.24±0.11**	2.59±1.02	0.48±0.08	0.47±0.27	0.34±0.17	0.86±0.26
MAE	**2 → 1**	**2.43±0.28**	6.27±2.15	2.64±0.96	5.03±1.93	3.18±0.85	3.19±0.92
	3 → 1	**4.48±1.01**	10.02±2.84	8.19±1.20	7.64±2.05	4.92±1.12	8.63±1.25
	4 → 1	**4.48±1.51**	8.49±1.78	6.46±1.18	14.31±4.69	5.05±0.58	7.10±2.65
	1 → 2	**1.67±0.97**	4.38±1.56	4.28±0.43	0.32±3.36	2.69±1.14	2.43±0.68
	3 → 2	4.32±1.01	5.51±1.02	4.00±0.52	6.21±2.91	**3.84±1.26**	5.50±1.07
	4 → 2	4.16±2.85	8.43±2.79	7.98±2.41	11.91±3.84	**3.82±1.15**	5.25±0.91
	1 → 3	**1.42±0.07**	11.48±4.25	3.56±1.49	5.31±4.14	2.57±1.22	3.29±0.95
	2 → 3	**1.34±0.34**	13.90±3.12	6.25±2.40	12.81±7.48	2.87±0.69	4.52±1.15
	4 → 3	**2.16±0.62**	4.28±0.95	5.93±1.58	6.19±1.40	2.56±0.31	2.33±0.49
	1 → 4	**4.36±2.18**	16.88±8.44	6.18±0.64	18.28±11.71	5.11±2.59	8.36±2.43
	2 → 4	**3.64±1.75**	13.48±4.50	6.07±2.17	15.83±12.12	4.55±1.81	9.84±5.11
	3 → 4	4.23±1.37	5.17±3.35	**4.21±0.27**	5.59±2.42	4.59±2.01	8.47±1.75
Inference parameters (M)		2.06	2.06	0.34	1.09	2.06	2.06
Inference time (s/1000)		0.103	0.103	0.100	0.033	0.103	0.103

Note: All error values in the table are multiplied by 100. The format is: mean ± std.

Table 5
Ablation study. Gradually adding each loss function to the proposed model to prove the effectiveness of each part.

$\mathcal{L}_{pred}$	√	√	√	√	√
$\mathcal{L}_{adv}$		√	√	√	√
$\mathcal{L}_{InfoNCE}$			√	√	√
$\mathcal{L}_{diff}$				√	√
DBN					√
$D^s \rightarrow D^t$	MSE				
2 → 1	0.50±0.28	0.16±0.08	0.23±0.23	0.12±0.09	**0.08±0.01**
3 → 1	1.61±0.68	0.95±0.25	1.04±0.67	0.41±0.32	**0.28±0.10**
4 → 1	1.79±0.74	0.86±0.47	0.86±0.66	0.37±0.25	**0.28±0.14**
1 → 2	0.32±0.23	0.11±0.05	**0.06±0.03**	0.06±0.04	0.06±0.06
3 → 2	0.45±0.11	0.43±0.15	0.30±0.16	**0.23±0.11**	0.26±0.11
4 → 2	1.47±0.65	0.48±0.16	0.74±0.53	0.54±0.30	**0.29±0.29**
1 → 3	1.60±0.96	0.17±0.07	0.25±0.24	0.12±0.141	**0.04±0.006**
2 → 3	2.17±0.99	0.27±0.12	0.32±0.27	0.08±0.07	**0.03±0.01**
4 → 3	0.45±0.19	0.15±0.10	0.11±0.05	0.09±0.03	**0.08±0.03**
1 → 4	3.79±3.31	0.92±0.43	0.81±0.59	0.54±0.40	**0.33±0.24**
2 → 4	2.21±1.24	1.39±1.30	1.22±0.74	0.59±0.31	**0.27±0.14**
3 → 4	2.59±1.02	0.86±0.26	0.76±0.39	0.40±0.17	**0.24±0.11**

Table 6
Ablation study on InfoNCE loss and MSE loss.

Domain	$\mathcal{L}_{InfoNCE}$		$\mathcal{L}_{MSE}$	
$D^s \rightarrow D^t$	MSE	MAE	MSE	MAE
2 → 1	**0.08±0.01**	**2.43±0.28**	0.16±0.11	2.79±1.15
3 → 1	**0.28±0.10**	**4.48±1.01**	0.64±0.47	5.96±3.00
4 → 1	**0.28±0.14**	**4.48±1.51**	0.49±0.26	5.57±2.00
1 → 2	**0.06±0.06**	**1.67±0.97**	0.11±0.04	2.44±0.49
3 → 2	**0.26±0.11**	**4.32±1.01**	0.32±0.30	4.51±2.18
4 → 2	**0.29±0.29**	**4.16±2.85**	0.44±0.22	5.36±1.85
1 → 3	**0.04±0.006**	**1.42±0.07**	0.19±0.11	**1.42±0.07**
2 → 3	**0.03±0.01**	**1.34±0.34**	0.10±0.03	2.41±0.49
4 → 3	**0.08±0.03**	**2.16±0.62**	0.21±0.28	2.66±1.60
1 → 4	**0.33±0.24**	4.36±2.18	0.39±0.20	**3.95±0.82**
2 → 4	**0.27±0.14**	**3.64±1.75**	0.38±019	4.07±1.16
3 → 4	**0.24±0.11**	**4.23±1.37**	0.29±0.16	4.24±1.60

4.3. Visualization of distributions

The proposed model extracts features layer by layer from the time, voltage, and current data during the charging process, and finally maps to the capacity. To verify that both domain-invariant and domain-specific features are learned by the Shared Encoder and Private Encoder, it is necessary to visualize the learned features.

Firstly, the data of two batteries were selected from the two domains, and the time, voltage, and current curves during the charging process were drawn, as shown in Fig. 1(a)–(f). Since the distribution of high-dimensional data could not be directly visualized, we used Principal Component Analysis (PCA) to process the raw data. The first principal component (PC) was analyzed, and the distributions of the first principal components of time, voltage, and current data in different domains were drawn by kernel density estimation (KDE), as shown in Fig. 1(g)–(i). Due to the various charging protocols, it can be seen that the shapes of the voltage and current curves are quite different, and the KDE figures also reflect that the data distributions of the two batteries have a great discrepancy. The charging time curve is a straight line passing through the origin that satisfies the equation $y = kx$, so the distribution is similar, which is also in line with cognition.

The raw data was mapped to a 256-dimensional feature space by the Shared Encoder and the Private Encoder, respectively. Likewise, we used PCA to process the shared features and private features, and plotted the KDE curves of the first PCs of the shared features and private features, respectively. As shown in Fig. 5, the distributions among the shared features are similar, while the distributions among the private features have a great discrepancy, which is what we expected. It also prove that the Shared Encoder has extracted domain-invariant features, and the Private Encoder has extracted domain-specific features. Through feature visualization techniques, it is proved that both Private Encoder and Shared Encoder work as expected.

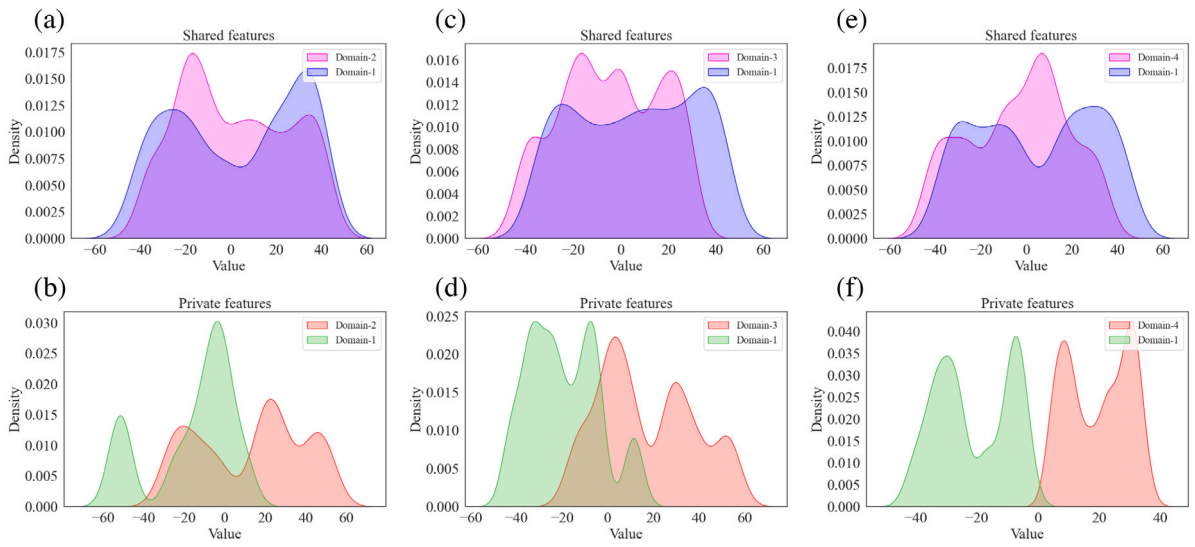


Fig. 5. Kernel density estimation curves for shared and private features. The feature distributions are visualized from 3 different domain adaptation tasks. For example, (a)–(b) show the kernel density estimation curves for shared and private features in tasks with D_2 as the source domain and D_1 as the target domain, respectively. Other subfigures are similar.

4.4. Time cost analysis

In this part, we make statistics and analysis on the time of model inference. The inference parameters of the proposed model and other comparative models are given in Table 4. For a fair comparison, each model was tested for 50 inferences on 1000 samples. The inference time of the each model was recorded and then averaged, shown in Table 4.

The four models, DR-Net, NoAdaptation, MMD, and DANN, have the same amount of inference parameters due to their backbones are the same. Therefore, their inference time is also the same. LSTM has the least amount of parameters, but its inference time is almost the same as DR-Net. Because LSTM performs multiple recurrent calculations in the time dimension, while CNN only does it once. In general, the computational efficiency of these models is very high, which can fully meet the needs of industry applications.

4.5. Sensitivity analysis

In this part, we evaluate the effect of different lengths of training data from target domain on DR-Net performance. To provide a simple and intuitive presentation, we conducted experiments with D_2 as source domain and D_1 as target domain. All the data of the source domain, which contains 4,415 samples, are used during the training. By changing the number of target domain samples participating in training, the performance of the trained DR-Net was tested on the target domain. The result was given in Fig. 6. It can be seen that as the number of target domain samples increased during the training, the test MSE of the DR-Net decreased. It is worth noting that in the case of D_2 as source domain and D_1 as target domain, the second best model is DANN (Table 4), and its MSE is 0.16 ± 0.08.

In addition, we also carried out sensitivity analysis experiments on trade-off parameters α, β, and γ. In the same way, we conducted experiments with D_2 as source domain and D_1 as target domain. Considering the optimal value of each parameter in Table 3, we set each parameter value range as follows: $\alpha \in [0.01, 0.05, 0.1, 0.5, 1]$, $\beta \in [0.0001, 0.001, 0.01, 0.1, 1]$, and $\gamma \in [1, 10, 20, 50, 80, 100]$. Since there are three parameters, we fixed two parameters and varied the other one in each experiment. For example, we fixed $\beta = 0.001$, $\gamma = 80$, and α take the value in $[0.01, 0.05, 0.1, 0.5, 1]$. The result was given in Fig. 7. As can be seen from the figure, the DR-Net performance is most sensitive to the value of γ, while the values of α and β have relatively little impacts on the model performance.

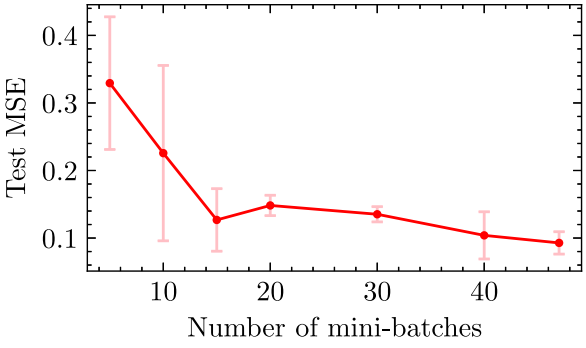


Fig. 6. The effect of different lengths of training data from target domain on DR-Net performance. The x-axis represents the number of mini-batches in the target domain during the training. A mini-batch contains 64 samples (batch size = 64). The y-axis represents test MSE on the test set, and the format is: mean $\pm$ std (%).

4.6. Ablation study

4.6.1. Effect of different loss functions

In this part, we performed ablation analysis to verify the effectiveness of each part in the proposed method. We disassembled each part, and then gradually added each part to the proposed method and conducted experiments. The experimental results were shown in Table 5. It can be observed that as the number of constraints we imposed on the model increases, the performance of the model also gets increasingly better, which proves that each part we proposed is effective. Different loss functions constrain the model into a more reasonable solution space, making it easier for the model to find the optimal solution.

4.6.2. Effect of contrastive learning

In this paper, we use the idea of contrastive learning to achieve degradation trend retainment. The main purpose of contrastive learning during our capacity estimation task is to encourage the model to learn a good representation through the comparison between positive pairs and negative pairs, thereby achieving higher accuracy capacity estimation.

Specifically, we added an InfoNCE loss [42] between the decoded data and the original data to ensure that the mutual information between them was maximized, which forced the Encoder to extract

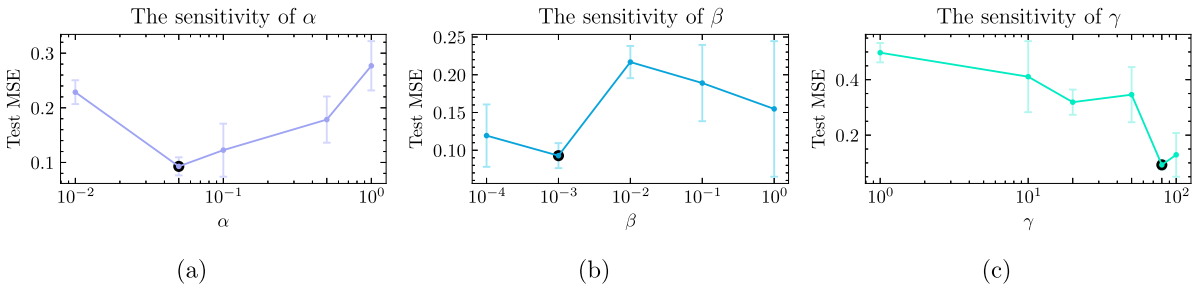


Fig. 7. The sensitivity of trade-off parameters on DR-Net performance. The black dots in each figure is the optimal value point selected by this paper.

better features that retain as much information as possible from the original data. There are also articles [40] that constrain the Encoder by reconstructing the original data, i.e. using MSE loss. Through ablation analysis, we proved that InfoNCE loss is better than MSE loss. We replaced $\mathcal{L}_{\text{InfoNCE}}$ with $\mathcal{L}_{\text{MSE}}$ and conducted experiments with all other conditions being equal, and the experimental results were given in Table 6.

As shown in Table 6, the model using $\mathcal{L}_{\text{InfoNCE}}$ achieves the better results, which also shows that the model has learned better features. Here, **we try to give a plausible explanation:** the MSE loss forces the model to reconstruct the original data and guarantees that the error at each point is small. Such constraints make the model sensitive to outliers. The model may adjust a large number of parameters just to reconstruct an outlier, which makes the model unstable, and the learned features are not the best. On the contrary, InfoNCE loss only pays attention to the mutual information between the decoded data and the original data, and does not pay attention to a specific point. It is not sensitive to outliers, so the model can learn a better representation.

5. Conclusion

In this paper, we proposed a DR-Net for unsupervised domain adaptive capacity estimation of lithium-ion batteries. Considering that the battery degradation data contains not only the degradation trend information, but also the diversity of charging/discharging strategies and individual difference of batteries, a feature disentanglement and tendency retainment network was designed. We constrain our DR-Net from structure so that it considers both shared and private information. By designing the Shared Encoder and the Private Encoder while applying corresponding constraints, the proposed model can disentangle the extracted features into domain-invariant and domain-specific features. To make the features more representative, i.e. retaining degradation trend information and ignoring the noise information, the features are decoded by a Decoder, and the mutual information between the decoded data and the raw data is guaranteed to be maximized. Thus, the model can learn more robust and representative domain-invariant features and achieve high-precision domain adaptive capacity estimation. The effectiveness of the proposed method was verified by conducting experiments on public datasets.

However, despite the promising results, the proposed method still has room for improvement. In practical industrial applications, batteries are not always fully charged and discharged. Our future work will focus on domain adaptive capacity estimation using partial charging/discharging cycling data. Furthermore, domain adaptive capacity estimation under different battery chemistries is also a future focus.

CRediT authorship contribution statement

Fujin Wang: Writing – original draft, Software, Methodology, Formal analysis, Conceptualization. **Zhibin Zhao:** Writing – review & editing, Methodology, Conceptualization. **Zhi Zhai:** Writing – review & editing, Supervision, Resources. **Yanjie Guo:** Supervision. **Huan Xi:** Writing – review & editing, Supervision. **Shibin Wang:** Supervision. **Xuefeng Chen:** Supervision, Funding acquisition.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the Natural Science Foundation of China under Grand 52105116; the China Postdoctoral Science Foundation under Grand 2021M692557 and Grand 2021TQ0263.

References

[1] Nykvist B, Nilsson M. Rapidly falling costs of battery packs for electric vehicles. Nature Clim Change 2015;5(4):329–32.
[2] Schmuch R, Wagner R, Hrpel G, Placke T, Winter M. Performance and cost of materials for lithium-based rechargeable automotive batteries. Nat Energy 2018;3(4):267–78.
[3] Lipu MH, Hannan M, Hussain A, Hoque M, Ker PJ, Saad MM, et al. A review of state of health and remaining useful life estimation methods for lithium-ion battery in electric vehicles: Challenges and recommendations. J Cleaner Prod 2018;205:115–33.
[4] Liu X, Zheng Z, Büyüktahtakın İE, Zhou Z, Wang P. Battery asset management with cycle life prognosis. Reliab Eng Syst Saf 2021;216:107948.
[5] Zou Y, Hu X, Ma H, Li SE. Combined state of charge and state of health estimation over lithium-ion battery cell cycle lifespan for electric vehicles. J Power Sources 2015;273:793–803.
[6] Xu X, Tang S, Yu C, Xie J, Han X, Ouyang M. Remaining useful life prediction of lithium-ion batteries based on wiener process under time-varying temperature condition. Reliab Eng Syst Saf 2021;214:107675.
[7] Zhai Q, Ye Z-S. RUL prediction of deteriorating products using an adaptive Wiener process model. IEEE Trans Ind Inf 2017;13(6):2911–21.
[8] Chen L, Lü Z, Lin W, Li J, Pan H. A new state-of-health estimation method for lithium-ion batteries through the intrinsic relationship between ohmic internal resistance and capacity. Measurement 2018;116:586–95.
[9] Lin CP, Ling MH, Cabrera J, Yang F, Yu DYW, Tsui KL. Prognostics for lithium-ion batteries using a two-phase gamma degradation process model. Reliab Eng Syst Saf 2021;214:107797.
[10] Hu X, Xu L, Lin X, Pecht M. Battery lifetime prognostics. Joule 2020;4(2):310–46.
[11] Xu F, Yang F, Fei Z, Huang Z, Tsui K-L. Life prediction of lithium-ion batteries based on stacked denoising autoencoders. Reliab Eng Syst Saf 2021;208:107396.
[12] Ardeshiri RR, Liu M, Ma C. Multivariate stacked bidirectional long short term memory for lithium-ion battery health management. Reliab Eng Syst Saf 2022;224:108481.
[13] Li S, Fang H, Shi B. Remaining useful life estimation of lithium-ion battery based on interacting multiple model particle filter and support vector regression. Reliab Eng Syst Saf 2021;210:107542.
[14] Roman D, Saxena S, Robu V, Pecht M, Flynn D. Machine learning pipeline for battery state-of-health estimation. Nat Mach Intell 2021;3(5):447–56.
[15] Li Y, Abdel-Monem M, Gopalakrishnan R, Berecibar M, Nanini-Maury E, Omar N, et al. A quick on-line state of health estimation method for li-ion battery with incremental capacity curves processed by Gaussian filter. J Power Sources 2018;373:40–53.
[16] Wang D, Kong J-Z, Zhao Y, Tsui K-L. Piecewise model based intelligent prognostics for state of health prediction of rechargeable batteries with capacity regeneration phenomena. Measurement 2019;147:106836.

[17] Ma Q, Zheng Y, Yang W, Zhang Y, Zhang H. Remaining useful life prediction of lithium battery based on capacity regeneration point detection. Energy 2021;234:121233.
[18] Kong JZ, Wang D. Multi-stage modeling and remaining charge-discharge cycles prediction of rechargeable batteries considering capacity regeneration phenomena. In: Proceedings of the 29th European safety and reliability conference. 2020.
[19] Zhang S-J, Kang R, Lin Y-H. Remaining useful life prediction for degradation with recovery phenomenon based on uncertain process. Reliab Eng Syst Saf 2021;208:107440.
[20] He J, Tian Y, Wu L. A hybrid data-driven method for rapid prediction of lithium-ion battery capacity. Reliab Eng Syst Saf 2022;226:108674.
[21] Zhang Q, Yang L, Guo W, Qiang J, Peng C, Li Q, et al. A deep learning method for lithium-ion battery remaining useful life prediction based on sparse segment data via cloud computing system. Energy 2022;241:122716.
[22] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, et al. A comprehensive survey on transfer learning. Proc IEEE 2020;109(1):43–76.
[23] Tian L, Tang Y, Hu L, Ren Z, Zhang W. Domain adaptation by class centroid matching and local manifold self-learning. IEEE Trans Image Process 2020;29:9703–18.
[24] Taskesen B, Yue M-C, Blanchet J, Kuhn D, Nguyen VA. Sequential domain adaptation by synthesizing distributionally robust experts. In: International conference on machine learning. PMLR; 2021, p. 10162–72.
[25] Sharma A, Kalluri T, Chandraker M. Instance level affinity-based transfer for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 5361–71.
[26] Li Y, Sheng H, Cheng Y, Stroe D-I, Teodorescu R. State-of-health estimation of lithium-ion batteries based on semi-supervised transfer component analysis. Appl Energy 2020;277:115504.
[27] Tan Y, Zhao G. Transfer learning with long short-term memory network for state-of-health prediction of lithium-ion batteries. IEEE Trans Ind Electron 2019;67(10):8723–31.
[28] Shu X, Shen J, Li G, Zhang Y, Chen Z, Liu Y. A flexible state-of-health prediction scheme for lithium-ion battery packs with long short-term memory network and transfer learning. IEEE Trans Transp Electrif 2021;7(4):2238–48.
[29] Li Y, Li K, Liu X, Wang Y, Zhang L. Lithium-ion battery capacity estimation—A pruned convolutional neural network approach assisted with transfer learning. Appl Energy 2021;285:116410.
[30] Liu Y, Shu X, Yu H, Shen J, Zhang Y, Liu Y, et al. State of charge prediction framework for lithium-ion batteries incorporating long short-term memory network and transfer learning. J Energy Storage 2021;37:102494.
[31] Wang Y-X, Chen Z, Zhang W. Lithium-ion battery state-of-charge estimation for small target sample sets using the improved GRU-based transfer learning. Energy 2022;123178.
[32] Ma J, Shang P, Zou X, Ma N, Ding Y, Sun J, et al. A hybrid transfer learning scheme for remaining useful life prediction and cycle life test optimization of different formulation Li-ion power batteries. Appl Energy 2021;282:116167.
[33] Kim S, Choi YY, Kim KJ, Choi J-I. Forecasting state-of-health of lithium-ion batteries using variational long short-term memory with transfer learning. J Energy Storage 2021;41:102893.
[34] Ye Z, Yu J. State-of-health estimation for lithium-ion batteries using domain adversarial transfer learning. IEEE Trans Power Electron 2021;37(3):3528–43.
[35] Wang F, Zhao Z, Ren J, Zhai Z, Wang S, Chen X. A transferable lithium-ion battery remaining useful life prediction method from cycle-consistency of degradation trend. J Power Sources 2022;521:230975.
[36] Wang F, Zhao Z, Zhai Z, Wang S, Ding B, Chen X. Remaining useful life prediction of lithium-ion battery based on cycle-consistency learning. In: 2021 International conference on sensing, measurement & data analytics in the era of artificial intelligence. IEEE; 2021, p. 1–6.
[37] Han T, Wang Z, Meng H. End-to-end capacity estimation of lithium-ion batteries with an enhanced long short-term memory network considering domain adaptation. J Power Sources 2022;520:230823.
[38] Chang W-G, You T, Seo S, Kwak S, Han B. Domain-specific batch normalization for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 7354–62.
[39] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 770–8.
[40] Bousmalis K, Trigeorgis G, Silberman N, Krishnan D, Erhan D. Domain separation networks. Adv Neural Inf Process Syst 2016;29:343–51.
[41] Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In: International conference on machine learning. PMLR; 2015, p. 1180–9.
[42] Van den Oord A, Li Y, Vinyals O. Representation learning with contrastive predictive coding. 2018, arXiv e-prints, arXiv–1807.
[43] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014, arXiv preprint arXiv:1409.1556.
[44] Attia PM, Grover A, Jin N, Severson KA, Markov TM, Liao Y-H, et al. Closed-loop optimization of fast-charging protocols for batteries with machine learning. Nature 2020;578(7795):397–402.
[45] Fu S, Zhang Y, Lin L, Zhao M, Zhong S-s. Deep residual LSTM with domain-invariance for remaining useful life prediction across domains. Reliab Eng Syst Saf 2021;216:108012.
[46] Shen S, Sadoughi M, Li M, Wang Z, Hu C. Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries. Appl Energy 2020;260:114296.