IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 31 October 2023, accepted 22 November 2023, date of publication 27 November 2023,
date of current version 4 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3337215

RESEARCH ARTICLE

Transfer Learning Techniques for the Lithium-Ion Battery State of Charge Estimation

PANAGIOTIS ELEFTHERIADIS, (Student Member, IEEE), SPYRIDON GIAZITZIS, SONIA LEVA, (Senior Member, IEEE), AND EMANUELE OGLIARI, (Member, IEEE)
Department of Energy, Politecnico di Milano, 20156 Milan, Italy

Corresponding author: Panagiotis Eleftheriadis (panagiotis.eleftheriadis@polimi.it)

This work was supported by the MOST–Sustainable Mobility Center through the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR)–MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4–D.D. 1033 17/06/2022) under Grant CN00000023.

ABSTRACT State of Charge (SOC) estimation is vital for battery management systems (BMS), impacting battery efficiency and lifespan. Accurate SOC estimation is challenging due to battery complexity and limited data for training Machine Learning based models. Transfer learning (TL) leverages pre-trained models, reducing training time and improving generalization in SOC estimation. In this paper, 8 different transfer learning techniques are examined, which were applied in four different models (LSTM, GRU, BiLSTM, and BiGRU) for SOC estimation. These transfer learning techniques have been applied to three datasets for re-training the models and results have been compared with the same models defined by Bayesian Hyperparameter Optimization. The TL4 and TL5 techniques consistently stood out as among the most efficient in both accuracy and computational time.

INDEX TERMS Transfer learning, lithium-ion battery, machine learning, state of charge.

NOMENCLATURE

SOC	State of Charge.
BHO	Bayesian Hyperparameter Optimization.
RNN	Recurrent Neural Network.
LSTM	Long-Short Term Memory.
BiLSTM	Bidirectional Long-Short Term Memory.
GRU	Gated Recurrent Unit.
BiGRU	Bidirectional Gated Recurrent Unit.
MMD	Maximum Mean Discrepancy.

I. INTRODUCTION

Batteries, among the advanced energy storage technologies [1], have become an integral part of various applications, from smartphones and laptops to electric vehicles and renewable energy systems [2]. The vital link of batteries, coupled with BMS, ensures the optimal performance, safety, and longevity of battery-powered devices and systems [3]. The accurate estimation of battery performance indicators such as the SOC, which refers to the remaining available energy in the battery relative to its maximum capacity, the State of Health (SOH) [4], the State of Power (SOP) and eventually the Remaining Useful Life (RUL) [5] is extremely important. There are several methods for estimating SOC [6], SOH [7], and SOP [8]. Moreover, SOC estimation prevents overcharging, extends battery life, avoids power failures, and enhances performance. Generally, SOC estimation relied on complex mathematical models based on electrochemical principles, requiring deep understanding, complex calculations, and extensive calibration. However, these methods struggle with real-world complexities, resulting in inaccurate estimations. Machine Learning (ML) has gained attention for SOC estimation, using data-driven algorithms to capture intricate relationships between battery parameters and SOC. By training ML models on large datasets, these algorithms generate accurate real-time estimations. However, the data-driven model may not be working effectively with a small amount of data while a large amount of data may lead to a huge computational burden [9]. The application of Transfer Learning (TL) has gained significant interest in recent

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

years because it allows for solving different tasks starting from a pre-trained Neural Network (NN). It often yields better results and requires less training time. As reported in [10], a Convolutional Neural Network (CNN) model was used as the source domain, with its pre-trained parameters transferred to the target domain. The network structure remained unchanged. The experiments demonstrated that this transferred model achieved higher performance with an accuracy score of 0.80 compared to the non-transferred model, which only achieved an accuracy of 0.64.

Recent publications in the field of SOC estimation for Electric Vehicle BMS address the challenge of accurate SOC estimation amid factors like temperature variations, aging, and inter-cell inconsistencies [11], [12]. In this review [13] the classifications of model-based and data-driven based SOC estimation are explained in terms of estimation model/algorithm, benefits, drawbacks, and estimation error. In [14] a 4^{th} order model-based extended Kalman filter (EKF) was designed and the SOC estimation results overcame the presented non-averaged model. Researchers in [15] propose a TL and domain adaptation method, but note the difficulty in achieving the same target space for online SOC estimation. They developed a new SOC differential processing method and a combined TL approach, successfully validating it with real battery pack data from China First Automobile Work, achieving precise SOC estimation with less than a 1.5% error.

Another research paper [16] focuses on SOC estimation through cloud-based BMS (C-BMS) applications. Feature extraction techniques, such as Empirical Mode Decomposition (EMD) and Long Short-Term Memory (LSTM) were used to construct a SOC estimation model. TL and adaptive weighting strategies improved accuracy, with maximum errors of 2.2% and 1.3%. Furthermore, [17], a research paper, suggests applying TL to reduce the need for extensive training data and improve SOC estimation efficiency. Considering the importance of lithium-ion batteries (LIBs), a study investigated the ML model's ability to predict SOC in cylindrical LIBs under different charge-discharge cycles, aiming to enhance prediction accuracy. Lastly [18], a review emphasises the challenges in accurately estimating the SOC of LIBs, highlighting nonlinear behaviour, environmental variations, and discrepancies between laboratory and real-world conditions in LIB SOC modelling and estimation.

However, choosing the most appropriate TL technique presents a trade-off among computing time, estimation errors, and other characteristics. Therefore, the underlying research question of this paper revolves around identifying the best-fitted TL technique for SOC estimation, considering the aforementioned trade-off.

This paper compares pre-trained Recurrent Neural Network (RNN) models, as reported in a previous study [19], applied to three LIB datasets (two public and one private) for the SOC estimation with different Transfer Learning approaches and source-target datasets combination.

For this reason, the two open datasets publicly available represent the source for the model building and the private one, obtained by the authors, is the target. This is the typical situation when there is a lack of data for training an ML model: open datasets are useful tools and TL is often adopted to accomplish the task on the private target-related problem. A comprehensive comparative analysis was conducted, examining the behavior of TL across various datasets and different RNN solutions. Even though there is extensive research around this topic, differently from what is usually found in literature where only a pair of datasets are considered [20], here three different datasets are used both as a source and target of the adopted TL techniques. Furthermore different combinations of freezing, tuning and minimization of the discrepancy between the target and source dataset, the models were tested dataset-specific and layer-specific aspects. Finally, a comparative analysis between several TL techniques and a Bayesian Hyperparameter Optimization (BHO) algorithm for predicting the battery SOC is presented. Finally, by evaluating computing time, estimation errors, and relevant characteristics, the aim is to identify the most effective approach, enhancing performance and efficiency. All the aforementioned points are a novelty with respect to the current state-of-the-art [20], [21].

The paper is structured as follows: Section II presents the various TL techniques applied, and Section III presents the datasets that were used to train and test the model. Section IV introduces the models used to apply the TL techniques in this paper. In Section V there is a presentation of the results of the models while in Section VI an extensive discussion is included. Lastly, in Section VII, there is a conclusion of the overall paper.

II. TRANSFER LEARNING TECHNIQUES

One significant challenge in the realm of ML pertains to the requirement that data both in the training and the test share a common feature space and distribution. Nonetheless, this assumption often fails in practical scenarios or when data can be easily outdated. In such instances, the successful application of knowledge transfer can yield substantial improvements in the learning process. In recent times, the concept of TL [22] has arisen as a novel learning framework aimed at addressing this intricate challenge [23]. The objective of TL is to transfer the knowledge gained from one dataset (the source) to a new dataset (the target) and to expedite the network-building process improving the computational efficiency, which is crucial in real-time applications.

Most commonly used for our purposes is the TL technique of ''feature extraction'', which involves combinations of freezing pre-trained layers and training new layers on top. A more sophisticated technique uses the Maximum Mean Discrepancy (MMD) method [20] for the training of the dense layers. MMD is a technique used in domain adaptation, which aims to transfer knowledge from a source domain to a target domain. MMD training involves training the NN layers to minimise the discrepancy between the source and target domain distributions. In this case, the training of the RNN

layer and the dense layers occurs in different steps. All the characteristics of the different TL techniques are summarized in Table 1.

A. FINE-TUNING (TL1)

The first technique is a typical fine-tuning technique, where all the layers of the NN, including the recurrent layers and dense layers, are subject to updates [18]. The weights and biases of each layer are initialised with pre-trained weights obtained from a related task or domain. Then, during the training phase, these weights are adjusted using backpropagation and gradient descent, where the error between the predicted SOC values and the ground truth SOC values is minimised. This fine-tuning process enables the network to adapt to the specific SOC estimation task and improve its performance by refining the learned representations in each layer.

B. PARTIAL FINE-TUNING (TL2)

This technique consists of a partial fine-tuning of the NN [21]. The initial recurrent layer and the initial dense layer can be used as static feature extractors in two-layer models by 'freezing' their weights and biases. In the case of three-layer models, this concept extends to the first two recurrent layers and the first two dense layers. These layers have already learned representations from the pre-training phase, and their purpose is to provide useful features to the last recurrent layer and to the last dense layer. The goal is to leverage the knowledge captured by the pre-trained layers while only updating the weights and biases of the third recurrent layer and the third dense layer. This approach can be useful when you have limited training data or want to avoid overfitting [24].

C. FEATURE EXTRACTION 1 (TL3)

This approach is characterized by the utilization of the MMD technique applied to the dense layers [20]. The primary objective is to facilitate the alignment of learned feature representations with the targeted feature distribution for SOC estimation. The process involves minimizing the discrepancy between these distributions, thereby enabling the NN to more effectively encapsulate pertinent data patterns and relationships. This, in turn, enhances the performance of SOC estimation. This technique can be beneficial when you have limited labelled training data but have access to pre-trained models or similar tasks [21]. It allows leveraging the pre-trained representations while fine-tuning specific layers and favouring the alignment of feature distributions using MMD.

D. HYBRID MODEL (TL4)

Similar to TL2, TL4 strikes a balance between freezing and fine-tuning layers, with exclusive training for dense layers [21]. This allows the dense layers to learn pertinent data relationships using pre-trained representations from recurrent layers. Freezing recurrent layers trims trainable parameters, preventing overfitting and lowering computational demands, speeding up training. Nevertheless, there's a trade-off, as fine-tuned dense layers might miss subtle temporal dependencies without adapting recurrent layers. This approach is less suitable if pre-trained recurrent layers don't align well with SOC estimation.

E. ADAPTIVE HIDDEN STATES (TL5)

By limiting the training to the final recurrent layer, this TL technique significantly reduces computational costs but carries the potential drawback of insufficiently capturing the requisite patterns for precise SOC estimation, reducing the model performance [21]. Such methods can also encounter gradient flow issues, potentially leading to vanishing gradients during backpropagation in the third recurrent layer, hindering learning. Additionally, fine-tuning a single layer heightens the risk of overfitting due to limited parameter updates, which may impede generalization to new data. To mitigate these risks, ample representative training data and a well-pre-trained model are essential. In this scenario, as the employed models demonstrated strong SOC predictive capabilities, it is the most advisable technique.

F. FEATURE EXTRACTION 2 (TL6)

This technique for the 2-layer models is the same as the previous feature extraction technique (TL3), but in the case of 3-layer models, the second recurrent layer is frozen, while the dense layers undergo MMD aiming to align learned feature representations with the target feature distribution for the SOC estimation [20]. Here, freezing the second recurrent layer is useful to study how the importance of acquired features (which generally increase at higher levels) influences TL, harnessing the pre-existing representations. This approach proves to be particularly beneficial when there is a scarcity of labelled training data, yet access to pre-trained models or analogous tasks is accessible [21].

G. FEATURE EXTRACTION 3 (TL7)

This technique is also a feature extraction method [20], but unlike the previous cases, the first recurrent layer is frozen while fine-tuning is applied to the remaining recurrent layers. As for the dense layers, similar to the previous feature extraction cases, they undergo MMD tuning, to align learned feature representations with the target feature distribution for SOC estimation. In this context, it is also interesting to observe how the different levels of the recurrent layers influence TL accuracy, offering insights into the effectiveness of feature representation at various NN stages.

H. FEATURE EXTRACTION 4 (TL8)

In this instance, akin to the prior technique, an approach centred on feature extraction is employed [20]. However, in this method, the final recurrent layer remains unaltered, while the lower recurrent layers, which are theoretically presumed to capture less critical features, undergo fine-tuning

TABLE 1. Main features of the adopted TL Techniques.

Technique	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8
Description	Fine-tuning all LS [18]	Partial fine-tuning [21]	MMD alignment [20]	DL training [21]	Last RL training [21]	MMD for 2-LS models [20]	FE with FR LS [20]	FE with lower LS fine-tuning [20]
LS Updated	All LS	3^{rd} LS	MMD specific LS	DLs	Last RL	DL and final RL	RL and DL	Lower RL and DLs
Overfitting Risk	High	Low (LU)	Moderate	Low (FR RL)	High (LU)	Low (FR 2^{nd} RL)	Low (FR 1^{st} RL)	Low (FR final RL)
Computational Cost	High	Low (FR RL)	Moderate	Low (FR RL)	Very low	Low (FR 2^{nd} RL)	Low (FR 1^{st} RL)	Low (FR final RL)

LS: Layers; RL: Recurrent Layer; DL: Dense Layer; FE: Feature Extraction; FR: Frozen; LU: Limited Updates

in conjunction with the dense layers through the MMD technique. This mirrors the TL3 approach, where the MMD method is applied to the dense layers to ensure alignment of the learned feature representations with the targeted feature distribution. By minimizing the differences between these distributions, the NN becomes more adept at capturing essential data patterns and relationships [21]. This configuration also offers an opportunity to examine how the alteration of lower-level recurrent layers, as opposed to the final layer, affects the overall accuracy and effectiveness of TL.

I. TRAINING PROCEDURE

All the selected techniques are summarised schematically for the 2-layer and the 3-layer models in Table 2. There exists a correlation between the outlined TL techniques, as adding one additional layer, both one RNN and one dense, to the 2-layer architecture, would make it correspond to its 3-layer architecture. The additional layer is depicted in Figure 3 highlighted in red. This is the rationale behind considering them as equivalent.

TABLE 2. 2 and 3-layer models layout with TL techniques.

#	Technique	2 Layer Model				3 Layer Model	
		RL 1	RL 2	DL 1	DL 2	RL 3	DL 3
TL1	fine-tuning	ft	ft	ft	ft	ft	ft
TL2	partial fine-tuning	FR	ft	FR	ft	ft	ft
TL3	FE 1	ft	ft	MMD	MMD	ft	MMD
TL4	hybrid model	FR	FR	ft	ft	FR	ft
TL5	adaptive HS	FR	ft	FR	FR	ft	FR
TL6	FE 2	ft	ft	MMD	MMD	FR	MMD
TL7	FE 3	FR	ft	MMD	MMD	ft	MMD
TL8	FE 4	ft	FR	MMD	MMD	FR	MMD

RL: Recurrent Layer; DL: Dense Layer; FE: Feature Extraction
HS: Hidden states; FR: Frozen

The dataset preprocess was the same as in [19], where the network inputs were normalized using the minimum-maximum normalization technique, as described in [25], which scales the measurements to a range between 0 and 1. To apply the TL techniques, the Keras Library [26] was employed to re-train the pre-trained models. The training process utilized the Adam optimization algorithm, as outlined in [27], which is a stochastic optimization method. The learning rate was set to 0.001 as the models were already pre-trained and it should be adapted to the new dataset. Similarly to [19] the Huber loss was used as the loss function [28] for the fine-tuning process, while the network underwent training for a total of 1000 epochs, and an early stopping strategy was implemented, triggering cessation if no improvement in validation loss occurred within 50 consecutive epochs.

For the sake of reproducibility, the TL techniques were executed on a computer equipped with an Intel i-910900KF processor, boasting a clock speed of 3.70 GHz, a memory capacity of 64 GB DDR4, and an NVIDIA GeForce GTX 1650 GPU.

J. EVALUATION METRICS

The purpose of this research is to compare the SOC estimation accuracy of different TL techniques and to analyse how these techniques perform on different datasets and different types of NNs. Together with the execution time, the mean absolute error (MAE) and the root mean square error (RMSE) were used:

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|SOC_i - \hat{SOC_i}| \tag{1}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(SOC_i - \hat{SOC_i})^2} \tag{2}$$

where SOC_i and $\hat{SOC_i}$ are the real value and the predicted value by the model, respectively.

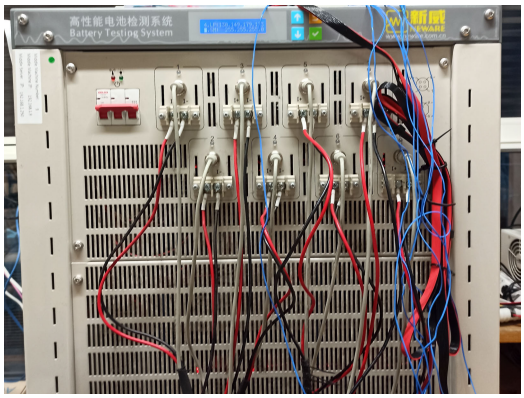
Based on the calculation of MAE and RMSE, the mean μ and the standard deviation σ are calculated for all the layers and datasets presented in the next section.

III. CASE STUDIES

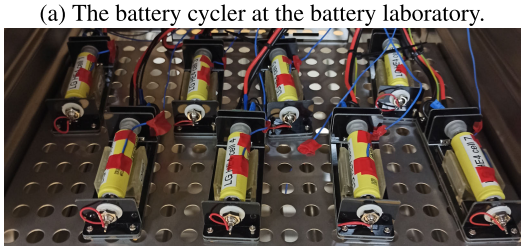
The first two open datasets (McMaster and CALCE) were used both as the "source" and "target", while the third private one (PoliMi), was employed only as the "target". This is the typical scenario, when there is a lack of data and pre-trained models are usually exploiting open data to transfer the knowledge on a limited "target" dataset. The main features of the dataset employed in this analysis, are summarized in Table 3. The three datasets contain dynamic driving cycles, with McMaster having the highest number, followed by PoliMi, and finally CALCE, which has a significantly lower count compared to the first two. It is also crucial to highlight that the PoliMi dataset is derived from the McMaster dataset but has been adjusted to account for the different battery cell ratings used. For example, the difference among the datasets is well depicted in Figure 2, where the distribution of the US06 driving cycle samples at 25°C of the voltage (Figure 2-a) and, for the sake of clarity, the cumulative density function of the standardized current samples (Figure 2-b) are shown.

TABLE 3. Datasets Main Features.

Features	McMaster	CALCE	PoliMi
Battery Type	3Ah 18650 - NMC Li-Ion	2Ah NMC Li-Ion	LG 2.5Ah 18650 NMC
Testing Equipment	Digatron battery tester	Arbin BT2000 battery cycler	Neware battery tester
Temperature Range	-20°C, -10°C, 0°C, 10°C, 25°C, 45°C	0°C, 25°C, 45°C	-20°C, -10°C, 0°C, 10°C, 25°C, 35°C
Driving Cycles	UDDS, US06, LA92, Mixed1-Mixed8	DST, US06, FUDS, BJDST	UDDS, US06, LA92, Mixed1-Mixed6
Sampling rate	0.1 seconds	1 second	0.1 seconds
No. of Samples	3,691,328	209,162	7,676,032



(a) The battery cycler at the battery laboratory.



(b) The 8 Li-Ion battery NMC cells inside the thermal chamber.

FIGURE 1. Two aspects of the experimental setup of the PoliMi dataset.

A. MCMASTER

The McMaster University [29] dataset encompasses the results of an experimental setup involving a 3Ah 18650 - Nickel Manganese Cobalt (NMC) Li-Ion battery cell. The testing took place within a thermal chamber, spanning six distinct ambient temperatures ranging from −20, −10, 0, 10, 25, to 45°C, thereby covering a wide temperature spectrum. Three different driving cycles were employed: the Urban Dynamometer Driving Schedule (UDDS), the Highway Driving Schedule (US06), and the California Unified Cycle (LA92), along with a variety of hybrid cycles identified as Mixed1, Mixed2, Mixed3, Mixed4, Mixed5, Mixed6, Mixed7, and Mixed8. Each of these driving cycles was conducted under the previously mentioned six ambient temperature conditions. Data acquisition was performed at a frequency of 0.1 seconds.

B. CALCE

The Centre for Advanced Life Cycle Engineering (CALCE) [30] dataset involves testing a 2Ah NMC Li-Ion battery cell using an Arbin BT2000 battery cycler within a thermal chamber. The testing occurred at three distinct ambient temperatures: 0, 25, and 45°C, representing low, room, and high-temperature conditions, respectively. Within this dataset, four driving cycles were employed: the Dynamic Stress Test (DST), the US06, the Federal Urban Drive Schedule (FUDS), and the Beijing Dynamic Stress Test (BJDST). All four driving cycles were executed under the three ambient temperature conditions mentioned earlier, with data recorded at a frequency of 1 second.

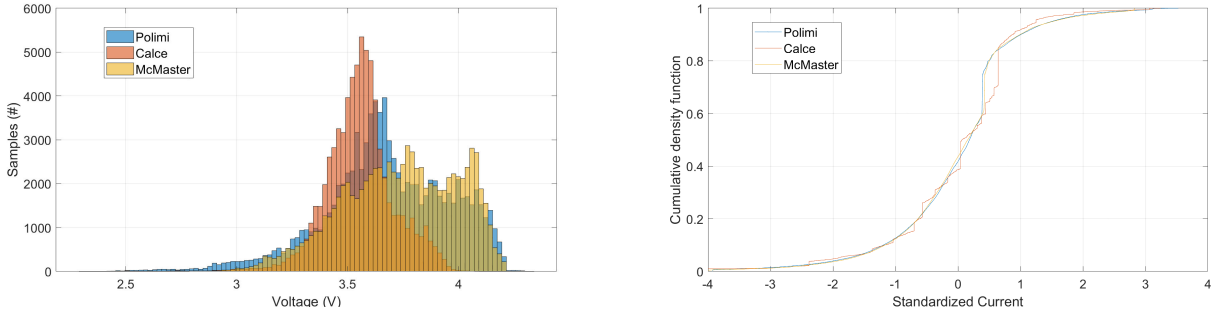
C. POLIMI

The PoliMi dataset was obtained through laboratory experiments conducted at the battery laboratory, involving six LIBs (LG 2.5Ah 18650 NMC) subjected to testing on a Neware battery tester depicted in Figure 1. Multiple driving cycles, namely UDDS, US06, LA92, and Mixed 1-6 (created by randomly combining the initial three driving cycles), were applied to each battery cell. Furthermore, the cells were subjected to various temperatures (−20°C, −10°C, 0°C, 10°C, 25°C, 35°C), with data recorded at a sampling rate of 0.1s.

IV. PRETRAINED MODELS USED

The models used in this study are obtained from a previous work of the authors [19]. These models are composed of an input layer receiving a time series of 30 seconds of the: terminal voltage, current, and surface temperature. In the next step, multiple recurrent layers are stacked on top of each other, where recurrent layers are a type of RNN layer designed to capture sequential dependencies in the data. Four different types of RNNs were used: the LSTM, the GRU, the BiLSTM and the BiGRU. Following the recurrent layers, there are multiple dense layers, also known as ''fully connected layers'', take the features extracted by the recurrent layers and map them to the desired output.

The pre-trained model, as described in [19], was generated for each dataset using BHO, incorporating a recurrent layer. This process resulted in models with hyperparameters close to the optimal parameters (according to BHO). The BHO explored a hyperspace where the number of recurrent and dense layers ranged between 2 and 3, and the number of units within each layer varied between 4 and 64 with a step of 4. The initial dense layers utilized a linear activation function, while the leaky ReLU (Rectified Linear Unit) was employed for the final dense layer. Optimization was carried out using the Adam optimizer with a learning rate of 0.00001. The Huber loss function was chosen as the loss metric, combined with an early stopping technique, which would terminate the training process if no improvement was observed for



(a) Distributions of the voltage.

(b) Cumulative density function of the standardized current.

FIGURE 2. In the three analysed datasets of the US06 driving cycle samples at 25°C.

50 consecutive epochs out of the total 1000 training epochs. Based on the findings in [19], the temporal resolution of the datasets was configured to be 1 second. Then, for a fair comparison, McMaster and PoliMi datasets are downsampled: 1 sample for every 10 was taken to downsample to 1 second and make the data homogeneous with CALCE in terms of granularity.

In the case of the McMaster dataset, as documented in [19], the training phase employed dynamic cycles LA92, UDDS, US06, Mixed1, Mixed2, Mixed7, and Mixed8 at temperatures of −10, 0, 10, and 25 degrees Celsius. Conversely, the testing phase utilized Mixed4, Mixed5, and Mixed6 dynamic cycles. Similarly, for the CALCE dataset, the training dataset consisted of dynamic cycles BJDST and DST, while the testing dataset included FUDS and US06 cycles, all at temperatures of 0, 10, and 25 degrees Celsius. Lastly, the PoliMi dataset underwent training using dynamic cycles UDDS, US06, and Mixed1 to Mixed4, and during the model evaluation, LA92, Mixed5, and Mixed6 cycles were used, covering temperature conditions of −10, 0, 10, and 25 degrees Celsius.

As observed in [19], the model's predictive accuracy is influenced by temperature variations. Notably, the model exhibits larger errors at lower temperatures, specifically at 0 and −10°C, in contrast to its performance at 25°C. This temperature-dependent behavior poses a significant challenge in accurately modeling the battery, highlighting the potential utility of TL in addressing this challenge.

In the McMaster dataset, after BHO, the use of LSTM and GRU layers led to a two-layer architecture, while bidirectional variants required a three-layer model. Similarly, in the CALCE dataset, models with LSTM layers had a two-layer architecture, whereas those with GRU, BiLSTM, and BiGRU layers adopted a three-layer structure. In Figure 3, the architecture of the pre-trained model is shown.

Subsequently, each of the source models underwent eight TL techniques when applied to the remaining datasets.

V. RESULT

In the forthcoming two subsections, the analysis of both the dataset and the layers is analyzed. While the BiLSTM model showcased superior performance in [19] across both the McMaster and CALCE datasets, it is imperative to investigate the performance of TL techniques across various layers and datasets to gain a more comprehensive perspective.

To facilitate the reading of the results, the labels of the generated model combine first the name of the source dataset followed by the target. For example: the "CALCE_McMaster" model, is formulated by applying TL from the CALCE pre-trained model to the McMaster dataset.

A. DATASET ANALYSIS

The results of each (TL) technique and layer were detailed per dataset to determine the most suitable TL approach for each specific dataset. For simplicity, the analysis primarily showcased MAE as an indicator, noting that RMSE displayed a similar trend but wasn't explicitly presented. Throughout the presented tables, the 'BHO' column means the MAE on the target dataset when the model was built following the same procedure as BHO, which was used for training the pre-trained models later applied in TL. The mean and the standard deviation of the complete dataset analysis are presented in Table 4 while the detailed results can be found in Appendix A.

TABLE 4. MAE mean and standard deviation of TL analysis.

	CALCE_McMaster		CALCE_PoliMi		McMaster_CALCE		McMaster_PoliMi	
MAE	μ	σ	μ	σ	μ	σ	μ	σ
TL1	1.709	0.063	3.102	0.218	1.520	0.292	2.773	0.153
TL2	1.548	0.049	2.954	0.075	1.170	0.114	2.200	0.350
TL3	1.646	0.079	2.428	0.289	1.213	0.084	1.926	0.099
TL4	1.547	0.104	2.446	0.385	1.156	0.077	1.933	0.029
TL5	1.499	0.039	2.337	0.438	1.185	0.049	1.951	0.048
TL6	1.566	0.097	2.307	0.499	1.292	0.085	2.062	0.081
TL7	1.604	0.091	2.385	0.477	1.318	0.116	2.063	0.094
TL8	1.549	0.057	2.379	0.483	1.274	0.078	1.994	0.040
BHO	2.286	0.274	2.522	0.160	1.612	0.267	2.522	0.160

Furthermore, the MAE boxplots are shown in Figure 4.

1) SOURCE DATASET: CALCE

In the context of the McMaster dataset as the target dataset (Figure 4 - a), the lowest error was observed with the BiLSTM layer in conjunction with TL4, resulting in an MAE of 1.369% with a reduction of 29.7% with respect to the BHO error, closely followed by the BiLSTM layer with TL5, which yielded an MAE of 1.437% and a reduction of

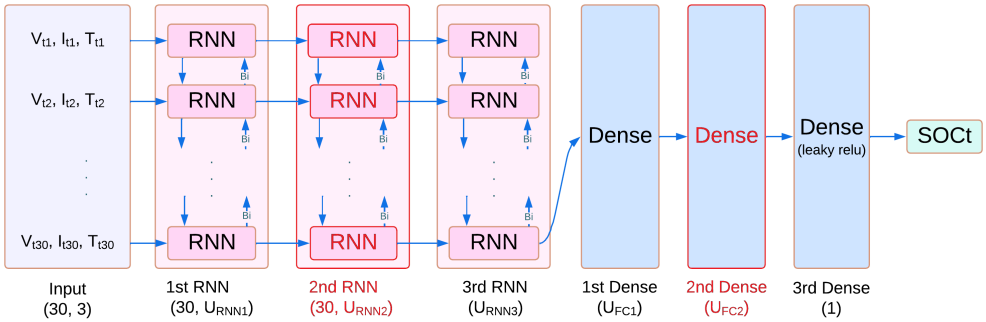


FIGURE 3. Comprehensive pre-trained model architecture; in the 2-layer model the 2nd layer (in red) both of the RNN and the dense layer were omitted. The RNN layer can be one between LSTM and GRU, while the arrow labelled "Bi" represents the bidirectional layer in the relevant models (BiLSTM or BiGRU).

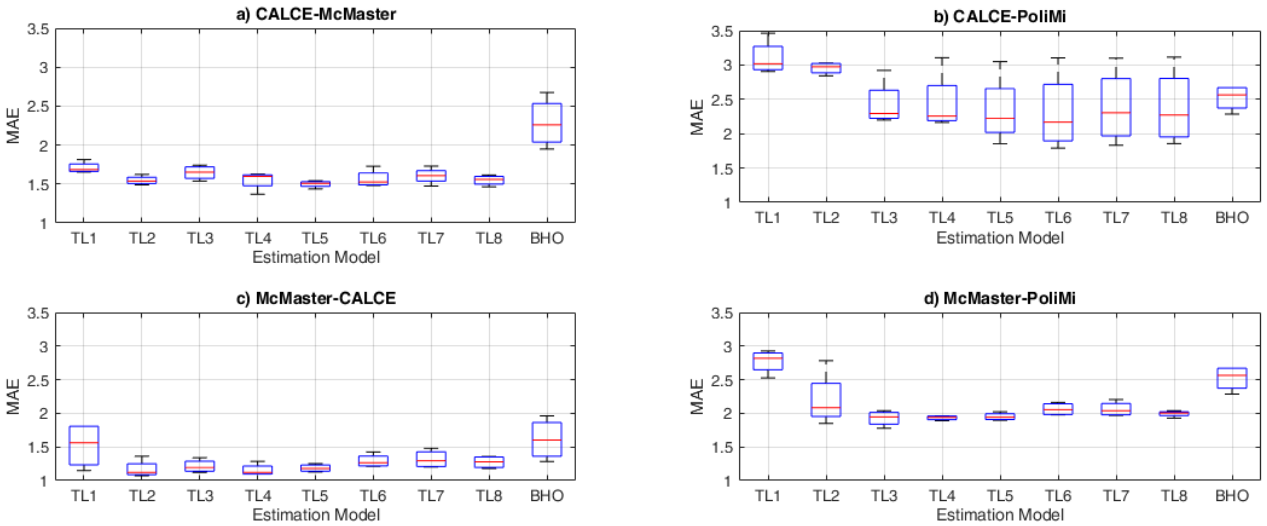


FIGURE 4. Mean Absolute Error of the estimation models, with the different dataset combination.

26.3% in respect to the BHO error. When considering all the layers examined, as it can be seen in Table 4, TL5 exhibited the lowest mean and SD of MAE of 1.499% and 0.039% respectively, with TL4 as the subsequent best-performing technique with 1.547% and 0.104% respectively.

In the same way, when PoliMi dataset is the target (Figure 4 - b), the minimum error was observed with the BiLSTM layer and TL6, resulting in an MAE of 1.789% with a reduction of 27.3% with respect to the BHO error, closely followed by the BiLSTM layer with TL7, which yielded an MAE of 1.832% with a reduction of 25.6% with respect to the BHO error. When assessing all the examined layers, as can be seen in Table 4, it was found that TL6 exhibited the lowest mean and SD of MAE of 2.307% and 0.499%, respectively. TL5 emerged as the subsequent best-performing technique with 2.337% and 0.438%, respectively.

2) SOURCE DATASET: MCMASTER

In the context of the CALCE dataset as the target dataset (Figure 4 - c), results are presented in Appendix A - Table 9. In this case, the lowest error was observed with the BiLSTM layer with TL2, resulting in an MAE of 1.073% with a reduction of 16.3% compared to BHO, closely followed by the LSTM layer with TL4, yielding an MAE of 1.096% with a reduction of 44.1% with respect to BHO. When considering all the layers examined, as it can be seen in Table 4, TL4 exhibited the lowest mean and SD of MAE of 1.156% and 0.077% respectively, with TL2 as the subsequent best-performing technique with 1.170% and 0.114% and TL5 closely performing with 1.185% and 0.049%, respectively.

Similarly, the complete results with the PoliMi dataset as the target (Figure 4 - d) are reported in Appendix A - Table 10. Here, the lowest error was observed when combining the BiGRU layer with TL3, resulting in an MAE of 1.777% with a reduction of 33.4% against BHO, followed closely by the LSTM layer with TL2 scoring an MAE of 1.849% with a reduction of 19.1% compared to the BHO. Upon a comprehensive evaluation of all examined layers, as can be seen in Table 4, it was determined that TL3 displayed the lowest mean and standard deviation of MAE of 1.926% and 0.099%, respectively. Subsequently, TL4 emerged as the second-best performing technique, with mean and standard deviation of MAE values of 1.933% and

0.029%, respectively. TL5 closely followed with values of 1.951% and 0.048%, respectively.

B. LAYER ANALYSIS

In this comprehensive analysis, the results for each TL technique and each dataset across the range of four-layer types (LSTM, BiLSTM, GRU and BiGRU) are presented, allowing to discern the most effective TL technique for each layer. In this case, RMSE is used as an indicator, as MAE follows a similar trend, though MAE is not explicitly displayed. The RMSE mean and standard deviation across the four dataset combinations of the layer analysis are presented in Table 5, and the relevant Figure 5, while the detailed results can be found in Appendix A. As mentioned previously in the Dataset Analysis, the four dataset combinations are: the CALCE_McMaster, CALCE_PoliMi, McMaster_CALCE, and McMaster_PoliMi.

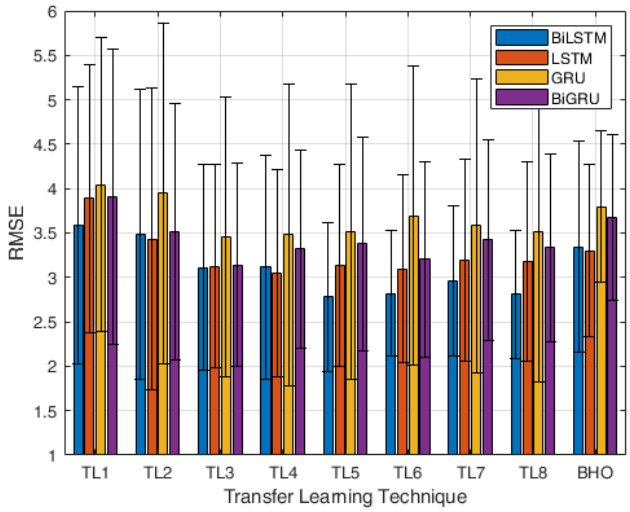


FIGURE 5. RMSE mean and standard deviation in the layer analysis.

TABLE 5. RMSE mean and standard deviation for the layer analysis.

	BiLSTM		LSTM		GRU		BiGRU	
RMSE	μ	σ	μ	σ	μ	σ	μ	σ
TL1	3.590	1.563	3.890	1.506	4.045	1.655	3.907	1.664
TL2	3.488	1.631	3.430	1.699	3.948	1.920	3.516	1.439
TL3	3.111	1.156	3.125	1.141	3.459	1.577	3.139	1.141
TL4	3.116	1.263	3.044	1.167	3.485	1.698	3.324	1.115
TL5	2.779	0.844	3.135	1.143	3.519	1.661	3.382	1.202
TL6	2.820	0.712	3.093	1.058	3.695	1.685	3.202	1.104
TL7	2.960	0.841	3.200	1.136	3.584	1.655	3.420	1.130
TL8	2.808	0.726	3.175	1.120	3.520	1.703	3.333	1.057
BHO	3.345	1.189	3.303	0.963	3.798	0.847	3.673	0.932

Please, note that, as previously stated, being the PoliMi dataset private, it was used only as a ''target'' resembling the most common scenario for TL adoption. In the same way as before, the ''BHO'' row represents the RMSE error scored on the target dataset when the model was developed following the identical procedure employed for BHO, which aligns with the training process of the pre-trained models subsequently utilized for TL. Within the context of each layer typology, varying error levels are observed for the four source-target combinations. More in detail, as reported in Table 5:

1) LSTM

- CALCE_McMaster: the lowest error was observed with the TL2 technique, resulting in an RMSE of 2.302% with a reduction of 25.4% with respect to the BHO error. This was closely followed by TL4 and TL5.
- CALCE_PoliMi: the model generated through the BHO approach displayed slightly superior performance compared to all the TL techniques with an RMSE of 4.155%. This combination achieved its lowest error with the TL6 technique, resulting in an RMSE of 4.164% closely followed by TL3.
- McMaster_CALCE: the lowest error was observed with the TL4 technique, yielding an RMSE of 1.546% with a reduction of 14.9% with respect to the BHO error. This was closely followed by TL2 and TL3.
- McMaster_PoliMi: the lowest error was observed with the TL2 technique, yielding an RMSE of 3.814% with a reduction of 8.17% with respect to the BHO error. This was closely followed by TL4.

TL4 exhibited the lowest RMSE mean and standard deviation of 3.044% and 1.167%, respectively. Subsequently, TL6 emerged as the next best-performing technique with values of 3.093% for the mean and 1.058% for the standard deviation of RMSE.

2) BILSTM

- CALCE_McMaster: TL4 scored the lowest error, with an RMSE of 2.246% and a reduction of 21.2% with respect to the BHO error. TL8, TL7 and TL5 closely followed this.
- CALCE_PoliMi: TL6 achieved the lowest error, with a RMSE of 3.331% and a reduction of 25.2% compared to the BHO error and TL5 closely followed this.
- McMaster_CALCE: the lowest error was observed with the TL2 technique, yielding an RMSE of 1.525% with a reduction of 6.2% with respect to the BHO error. TL4 and TL5 closely followed this.
- McMaster_PoliMi: the lowest error was observed with the TL8 technique, yielding an RMSE of 3.499% with a reduction of 21.4% with respect to the BHO error. This was closely followed by TL3 and TL6.

Here, TL5 exhibited the lowest mean and standard deviation of RMSE values of 2.779% and 0.844%, respectively. Following closely, TL8 emerged as the next best-performing technique with values of 2.808% for the mean RMSE and 0.726% for the standard deviation of RMSE. TL6 also performed well, with mean and standard deviation of RMSE values of 2.820% and 0.712%, respectively.

3) GRU

- CALCE_McMaster: the lowest error was observed with the TL8 technique, resulting in an RMSE of 2.399% with

a reduction of 34.9% with respect to the BHO error. This was closely followed by TL5 and TL4.

- CALCE_PoliMi: the model generated through the BHO approach displayed markedly superior performance compared to all the TL techniques. In this analysis, the lowest error resulted for the TL3 technique, with an RMSE of 5.808% closely followed by TL1. Conversely, this trend is not reflected in the MAE metric, where the error of the BHO model falls among the highest.
- McMaster_CALCE: the lowest error was observed with the TL4 technique, yielding an RMSE of 1.565% with a reduction of 36.3% with respect to the BHO error. This was closely followed by TL3 and TL2.
- McMaster_PoliMi: the lowest error was observed with the TL3 technique, yielding an RMSE of 3.858% with a reduction of 14.7% with respect to the BHO error while TL4 closely followed this.

In this case, TL3 exhibited the lowest RMSE mean and standard deviation of 3.459% and 1.577%. Subsequently, TL4 emerged as the next best-performing technique with values of 3.485% and 1.698%, respectively, followed by TL5 with values of 3.519% and 1.661%, respectively.

4) BIGRU

- CALCE_McMaster: the lowest error was observed with the TL6 technique, resulting in an RMSE of 2.525% with a reduction of 25.1% with respect to the BHO error. This was closely followed by TL3 and TL2.
- CALCE_PoliMi: TL8 achieved its lowest error, resulting in an RMSE of 4.663%. TL6 closely followed this with an RMSE of 4.725%. In a similar way to the GRU layer, the model generated through the BHO approach displayed markedly superior performance compared to all the TL techniques. In the same way, as for the GRU layer, that is not mirrored in the MAE metric where the BHO model error is within the highest.
- McMaster_CALCE: the lowest error was observed with the TL1 technique, yielding an RMSE of 1.750% with a reduction of 23.2% with respect to the BHO error. This was closely followed by TL5 and TL2.
- McMaster_PoliMi: the lowest error was observed with the TL3 technique, yielding an RMSE of 3.187% with a reduction of 29.5% with respect to the BHO error while This was closely followed by TL6.

In this case, TL3 exhibited the lowest RMSE mean and standard deviation of 3.139% and 1.141%, respectively. Subsequently, TL6 emerged as the next best-performing technique with values of 3.202% for the mean of RMSE and 1.104% for the standard deviation of RMSE.

In almost all instances, except in CALCE_PoliMi, nearly all TL techniques consistently outperformed the corresponding BHO model, as evidenced in the preceding tables. This observation underscores the inherent advantages of employing TL techniques in achieving superior results.

C. TRAINING TIME COMPARISON

As mentioned in [19], BHO was used to discover the nearly optimal models and this took between 5 to 8 hours, with the equipment described in subsection II-I. This is a time-intensive procedure, although it ultimately results in favourable outcomes with notably lower errors.

While implementing the TL techniques, the pre-trained model undergoes a re-training process and the framework records the training time in seconds. The resulting training times typically range from 50 to 100 seconds, which are significantly shorter than the time required for BHO tuning. Consequently, for better comparison of the TL techniques the recorded time is reported in Table 6 and it exclusively represents the duration of the network training process and does not encompass data loading and preprocessing.

TABLE 6. Mean and standard deviation Training Time for each TL technique among all layers and all datasets analyses.

Training Time	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8
μ (s)	107.0	53.8	123.8	39.8	46.4	68.8	73.1	70.2
σ (s)	46.2	13.0	34.1	6.3	14.4	18.3	21.6	19.6

Consistently across various layers and datasets, each TL technique exhibited a similar training duration. Consequently, the mean training time was computed and displayed. As it can be seen in Table 6, among the TL techniques, TL4 stands out as the fastest, boasting a mean processing time of 39.8 seconds and an SD of 6.3s seconds. It is closely followed by TL5, which records a mean time of 46.4 seconds and a standard deviation of 14.4 seconds, and TL2, with respective times of 53.8 seconds and 13.0 seconds. The remaining group comprising TL6, TL7, and TL8 is not significantly different in terms of speed, while TL1 and TL3 tend to be the most computationally time-consuming. Nevertheless, it is important to note that none of the techniques pose a significant processing time burden, especially when compared to their counterpart BHO model, which requires hours for tuning. However, for the purpose of the comparison, it's worth noting that TL4 and TL5 are more favourable choices when the training time is critical, especially when related to the dataset size.

VI. DISCUSSION

The comprehensive analysis presented in this study investigated the effectiveness of various TL techniques in the context of SOC estimation. These techniques have been assessed both in terms of accuracy and computational efficiency.

A crucial decision in TL is determining which layers of an NN to freeze or not and which ones to fine-tune. The strategy behind this selection can heavily affect the SOC estimation performance of the model and its training time.

Out of all the transfer learning techniques employed, the training phase for each of them was halted using the

early stopping technique, which was triggered when further training would not lead to improved network performance.

Among all the reviewed TL techniques, TL4 and TL5 consistently emerged as the most effective in terms of accuracy, offering computational advantages at the same time, which makes them particularly noteworthy. The TL4 technique consistently performed well across different datasets and layers. This technique includes freezing the recurrent layers and training only the dense layers. This suggests that dense layers are essential for capturing the relevant patterns in the data, while the recurrent layers' pre-trained representations are already suitable for the tasks. Techniques like TL5, which allows only the last recurrent layer to be trained, suggest that adjusting the final recurrent layer can be beneficial without adding too much computational cost.

TL techniques involving MMD (TL3, TL6, TL7 and TL8) seem to be quite effective in certain scenarios, especially with the BiLSTM layer, given its role in aligning learned feature representations of the source and target distributions.

It is important to highlight that the McMaster dataset, which contains more dynamic driving cycles, seems to have benefitted the most from TL, especially when it's used as a target dataset. The compatibility of McMaster with the PoliMi dataset could also be an influencing factor. While the results are promising, not all combinations of source-target datasets, NN layers, and TL techniques guarantee an improvement. The case of LSTM with the CALCE_PoliMi model underlines this aspect.

When extensive datasets are available for the target task, it's more feasible to unfreeze more layers or even the entire network, allowing for more fine-tuning. Conversely, with limited data available, it is generally a conservative approach to freeze the former layers and only fine-tune the latter ones, minimizing the number of trainable parameters.

In these case studies, since both source and target datasets are on the same task of estimating the SOC of a battery, more layers can be unfrozen. On the contrary, if the tasks were different, it would have been better to retain the initial layers and fine-tune only the last few layers. If the layers are completely retrained, there's a risk that the final model might forget the useful features learned from the source. By freezing some layers, the original learned features are preserved.

It is worth highlighting that in certain instances, a negative transfer learning effect was observed. This was evident in cases such as TL1 and McMaster_PoliMi, as well as in some cases involving CALCE_PoliMi. This observation underscores a potential limitation of transfer learning techniques, and it emphasizes the importance of exercising caution and careful consideration when applying them.

The main finding from this analysis is that strategic freezing of layers can lead to both higher SOC estimation accuracy and computational efficiency. Focusing on fine-tuning either of the dense layers exclusively or of the last recurrent layer seems to represent an optimal approach for the datasets under scrutiny.

TABLE 7. pre-trained model with CALCE on McMaster.

MAE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
LSTM	1.814	1.526	1.697	1.585	1.501	1.556	1.602	1.582	2.128
GRU	1.700	1.623	1.742	1.609	1.517	1.727	1.729	1.535	2.675
BiLSTM	1.655	1.489	1.608	1.369	1.437	1.482	1.473	1.463	1.949
BiGRU	1.669	1.551	1.538	1.627	1.542	1.497	1.613	1.615	2.393
μ	1.709	1.548	1.646	1.547	1.499	1.566	1.604	1.549	-
σ	1.707	1.546	1.643	1.540	1.498	1.560	1.599	1.547	-

TABLE 8. pre-trained model with CALCE on PoliMi.

MAE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
LSTM	2.906	2.933	2.200	2.164	2.263	2.003	2.103	2.051	2.286
GRU	3.463	3.030	2.921	3.107	3.049	3.103	3.099	3.114	2.671
BiLSTM	2.957	3.013	2.246	2.296	1.855	1.789	1.832	1.856	2.462
BiGRU	3.080	2.841	2.343	2.218	2.182	2.334	2.505	2.495	2.669
μ	3.102	2.954	2.428	2.446	2.337	2.307	2.385	2.379	-
σ	3.087	2.952	2.397	2.395	2.264	2.211	2.295	2.288	-

TABLE 9. pre-trained model with McMaster on CALCE.

MAE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
LSTM	1.805	1.138	1.155	1.096	1.211	1.211	1.214	1.213	1.439
GRU	1.808	1.104	1.122	1.097	1.150	1.226	1.202	1.180	1.962
BiLSTM	1.148	1.073	1.234	1.145	1.128	1.425	1.480	1.358	1.283
BiGRU	1.319	1.363	1.340	1.285	1.252	1.305	1.375	1.345	1.765
μ	1.520	1.170	1.213	1.156	1.185	1.292	1.318	1.274	-
σ	1.462	1.160	1.207	1.151	1.183	1.286	1.308	1.269	-

VII. CONCLUSION

This paper presents a comprehensive study on the application of eight Transfer Learning techniques that significantly enhance the accuracy and the computational efficiency of the estimation of the State of Charge of Lithium batteries. The accuracy of four pre-trained RNN models is analysed. In the dataset analysis, these models are built on two public "source" datasets (McMaster and CALCE) and they are subsequently applied on "target" datasets both by crossing them, and on a private dataset (PoliMi). These datasets, which provide comprehensive information on various characteristics and performance measures of LIBs, form the basis of the models used in this research. Furthermore, in a layer typology analysis, the different TL techniques are applied to the four previously defined RNN models: LSTM, BiLSTM, GRU and BiGRU. SOC estimation results are finally compared with the same models which underwent a Bayesian Hyperparameter Optimization.

The comparative analysis of the transfer learning techniques, including feature extraction and MMD (TL3, TL6, TL7 and TL8), reveals the potential of these methods in improving SOC estimation accuracy, especially with the BiLSTM layer. The results showed that the impact of the transfer learning technique on the model performance heavily depends on the characteristics and the extension of the datasets that are employed. In addition, in most cases, the TL4 and TL5 generally yielded better results than employing BHO to find the near-optimal hyperparameters of the model, revealing both higher accuracy and lower computational effort.

An immediate extension of this work is the application of these TL techniques to estimate the SOC of batteries with different chemistries, to validate the robustness and versa-

TABLE 10. pre-trained model with McMaster on PoliMi.

MAE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
LSTM	2.927	1.849	1.989	1.921	1.920	1.987	1.990	2.007	2.286
GRU	2.769	2.782	1.902	1.891	1.962	2.162	1.968	1.929	2.671
BiLSTM	2.868	2.110	2.038	1.963	1.897	2.121	2.206	2.038	2.462
BiGRU	2.527	2.058	1.777	1.959	2.024	1.978	2.089	2.004	2.669
μ	2.773	2.200	1.926	1.933	1.951	2.062	2.063	1.994	-
σ	2.764	2.150	1.921	1.933	1.950	2.059	2.059	1.993	-

TABLE 11. RMSE of LSTM to all four TL models.

RMSE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
CALCE_McMaster	2.532	2.302	2.478	2.324	2.361	2.451	2.546	2.517	3.086
CALCE_PoliMi	5.614	6.020	4.373	4.489	4.565	4.164	4.499	4.391	4.155
McMaster_CALCE	2.260	1.582	1.589	1.546	1.719	1.689	1.686	1.680	1.816
McMaster_PoliMi	5.154	3.814	4.060	3.816	3.895	4.066	4.068	4.110	4.155
μ	3.890	3.430	3.125	3.044	3.135	3.093	3.200	3.175	-
σ	3.307	2.676	2.653	2.561	2.701	2.691	2.751	2.733	-

TABLE 12. RMSE of BiLSTM to all four TL models.

RMSE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
CALCE_McMaster	2.538	2.389	2.427	2.246	2.310	2.319	2.302	2.296	2.851
CALCE_PoliMi	5.435	5.687	4.740	4.783	3.425	3.331	3.461	3.539	4.452
McMaster_CALCE	1.621	1.525	1.695	1.593	1.642	1.945	2.005	1.896	1.626
McMaster_PoliMi	4.766	4.352	3.582	3.841	3.738	3.686	4.072	3.499	4.452
μ	3.590	3.488	3.111	3.116	2.779	2.820	2.960	2.808	-
σ	2.847	2.703	2.681	2.593	2.498	2.637	2.726	2.612	-

TABLE 13. RMSE of GRU to all four TL models.

RMSE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
CALCE_McMaster	2.565	2.524	2.588	2.436	2.411	2.616	2.613	2.399	3.686
CALCE_PoliMi	5.882	5.965	5.808	6.052	5.950	6.043	6.050	6.086	4.524
McMaster_CALCE	2.234	1.593	1.583	1.565	1.629	1.668	1.650	1.633	2.456
McMaster_PoliMi	5.498	5.712	3.858	3.887	4.084	4.452	4.023	3.964	4.524
μ	4.045	3.948	3.459	3.485	3.519	3.695	3.584	3.520	-
σ	3.363	2.927	2.760	2.718	2.775	2.915	2.851	2.767	-

TABLE 14. RMSE of BiGRU to all four TL models.

RMSE	TL1	TL2	TL3	TL4	TL5	TL6	TL7	TL8	BHO
CALCE_McMaster	2.831	2.553	2.540	2.653	2.627	2.525	2.672	2.662	3.369
CALCE_PoliMi	5.677	5.539	4.947	4.693	4.769	4.725	4.777	4.663	4.522
McMaster_CALCE	1.750	1.824	1.883	1.879	1.824	1.847	1.993	1.996	2.279
McMaster_PoliMi	5.368	4.151	3.187	4.070	4.308	3.712	4.239	4.013	4.522
μ	3.907	3.516	3.139	3.324	3.382	3.202	3.420	3.333	-
σ	3.108	2.938	2.776	2.924	2.918	2.820	3.027	2.984	-

tility of these techniques. Additionally, the TL techniques presented in this study will be adapted for estimating other LIB parameters, such as the SOH and SOP. This would further broaden the scope of these methods, enabling more holistic management of battery systems.

APPENDIX A
TABLES

See Tables 7–14.

ACKNOWLEDGMENT

This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

REFERENCES

[1] R. Xiong, S. M. Sharkh, H. Li, H. Bai, W. Shen, P. Bai, and X. Zhou, "IEEE access special section editorial: Advanced energy storage technologies and their applications," *IEEE Access*, vol. 8, pp. 218685–218693, 2020.

[2] P. Venugopal, S. S. Shankar, C. P. Jebakumar, R. Agarwal, H. H. Alhelou, S. S. Reka, and M. E. H. Golshan, "Analysis of optimal machine learning approach for battery life estimation of Li-ion cell," *IEEE Access*, vol. 9, pp. 159616–159626, 2021.

[3] B. Balasingam, P. Kumar, and G. Rankin, "Battery fuel gauge: A crucial element in battery management systems," *IEEE Instrum. Meas. Mag.*, vol. 25, no. 7, pp. 14–20, Oct. 2022.

[4] N. Noura, L. Boulon, and S. Jemeï, "A review of battery state of health estimation methods: Hybrid electric vehicle challenges," *World Electr. Vehicle J.*, vol. 11, no. 4, p. 66, Oct. 2020. [Online]. Available: https://www.mdpi.com/2032-6653/11/4/66

[5] R. Xiong, Y. Zhang, J. Wang, H. He, S. Peng, and M. Pecht, "Lithium-ion battery health prognosis based on a real battery management system used in electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4110–4121, May 2019.

[6] P. Eleftheriadis, S. Giazitzis, S. Leva, and E. Ogliari, "Data-driven methods for the state of charge estimation of lithium-ion batteries: An overview," *Forecasting*, vol. 5, no. 3, pp. 576–599, Sep. 2023. [Online]. Available: https://www.mdpi.com/2571-9394/5/3/32

[7] P. Eleftheriadis, S. Giazitzis, C. Ozmalatyalilar, S. Leva, and R. Zich, "An overview of data-driven methods for the state of health estimation," in *Proc. IEEE Int. Conf. Environ. Electr. Eng. IEEE Ind. Commercial Power Syst. Eur.*, Jun. 2023, pp. 1–6.

[8] R. Guo and W. Shen, "A review of equivalent circuit model based online state of power estimation for lithium-ion batteries in electric vehicles," *Vehicles*, vol. 4, no. 1, pp. 1–31, Dec. 2021.

[9] M. S. H. Lipu, M. A. Hannan, A. Hussain, A. Ayob, M. H. M. Saad, T. F. Karim, and D. N. T. How, "Data-driven state of charge estimation of lithium-ion batteries: Algorithms, implementation factors, limitations and future trends," *J. Cleaner Prod.*, vol. 277, Dec. 2020, Art. no. 124110.

[10] S. Chen, H. Yao, F. Qiao, Y. Ma, Y. Wu, and J. Lu, "Vehicles driving behavior recognition based on transfer learning," *Expert Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 119254.

[11] M. U. Ali, A. Zafar, S. H. Nengroo, S. Hussain, M. J. Alvi, and H.-J. Kim, "Towards a smarter battery management system for electric vehicle applications: A critical review of lithium-ion battery state of charge estimation," *Energies*, vol. 12, no. 3, p. 446, Jan. 2019.

[12] M. Adaikkappan and N. Sathiyamoorthy, "Modeling, state of charge estimation, and charging of lithium-ion battery in electric vehicle: A review," *Int. J. Energy Res.*, vol. 46, no. 3, pp. 2141–2165, Mar. 2022.

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

[14] D. Di Domenico, G. Fiengo, and A. Stefanopoulou, "Lithium-ion battery state of charge estimation with a Kalman filter based on a electrochemical model," in *Proc. IEEE Int. Conf. Control Appl.*, Sep. 2008, pp. 702–707.

[15] M. A. Hannan, M. S. H. Lipu, A. Hussain, and A. Mohamed, "A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations," *Renew. Sustain. Energy Rev.*, vol. 78, pp. 834–854, Oct. 2017.

[16] P. Qin and L. Zhao, "A novel transfer learning-based cell SOC online estimation method for a battery pack in complex application conditions," *IEEE Trans. Ind. Electron.*, vol. 71, no. 2, pp. 1606–1615, Feb. 2024, doi: 10.1109/TIE.2023.3250768.

[17] D.-W. Chung, J.-H. Ko, and K.-Y. Yoon, "State-of-charge estimation of lithium-ion batteries using LSTM deep learning method," *J. Electr. Eng. Technol.*, vol. 17, no. 3, pp. 1931–1945, May 2022.

[18] Z. Zhang, Z. Yang, C. Gan, and H. Karlsen, "State-of-charge estimation of lithium battery based on transfer learning," in *Proc. Int. Conf. Artif. Intell., Big Data Algorithms (CAIBDA)*, May 2021, pp. 50–53.

[19] P. Eleftheriadis, S. Leva, and E. Ogliari, "Bayesian hyperparameter optimization of stacked bidirectional long short-term memory neural network for the state of charge estimation," *Sustain. Energy, Grids Netw.*, vol. 36, Dec. 2023, Art. no. 101160.

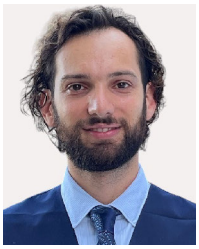
[20] S. Schwendemann, Z. Amjad, and A. Sikora, "Bearing fault diagnosis with intermediate domain based layered maximum mean discrepancy: A new transfer learning approach," *Eng. Appl. Artif. Intell.*, vol. 105, Oct. 2021, Art. no. 104415.

[21] L. Shen, J. Li, J. Liu, L. Zhu, and H. T. Shen, "Temperature adaptive transfer network for cross-domain state-of-charge estimation of Li-ion batteries," *IEEE Trans. Power Electron.*, vol. 38, no. 3, pp. 3857–3869, Mar. 2023.

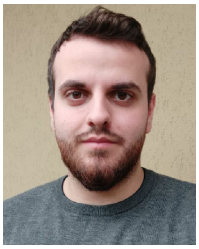
[22] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Jan. 2009.

[23] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, pp. 1–40, 2016.

[24] W. Kang, L. Lin, S. Sun, B. Zhang, X. Shen, and S. Wu, "Three-round learning strategy based on 3D deep convolutional GANs for Alzheimer's disease staging," *SSRN Electron. J.*, vol. 13, no. 1, p. 5750, 2023.
[25] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Jan. 2011.
[26] C. Keras-Team, "Keras-team/keras: Deep learning for humans. [Online]. Available: https://github.com/fchollet/keras
[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[28] P. J. Huber, "Robust estimation of a location parameter," *Breakthroughs Statistics: Methodology Distribution*. Cham, Switzerland: Springer, 1992, pp. 492–518.
[29] M. Naguib. (2020). *Lg 18650hg2 Li-ion Battery Data and Example Deep Neural Network xEV SOC Estimator Script*. [Online]. Available: https://data.mendeley.com/datasets/cp3473x7xv/3
[30] L.-i. b. e. d. University of Maryland. *Calce Battery Data*. [Online]. Available: https://calce.umd.edu/battery-data



PANAGIOTIS ELEFTHERIADIS (Student Member, IEEE) received the Dipl.Eng. degree from the Aristotle University of Thessaloniki, in 2017. He is currently pursuing the Ph.D. degree. His research interests include AI, batteries, and renewable energy sources.



SPYRIDON GIAZITZIS received the Dipl.Eng. degree from the Aristotle University of Thessaloniki, in 2023. He is currently pursuing the Ph.D. degree. His research interests include AI, batteries, and renewable energy sources.

SONIA LEVA (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Politecnico di Milano, Italy, in 2001. She is currently a Full Professor of electrical engineering with the Department of Energy, Politecnico di Milano. She is also the Director of the Solar Tech Laboratory (SolarTechLAB) and the Laboratory of MicroGrids (MG2LAB), Politecnico di Milano. She is a Senior Member of the IEEE Power and Energy Society and a member of the IEEE Working Group Distributed Resources Modeling and Analysis.

EMANUELE OGLIARI (Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from Politecnico di Milano, Italy, in 2016. He has been working on photovoltaic power plant design and their optimization, since 2010, and has also been on RES expected power by means of computational intelligence techniques, since 2012. He is currently an Associate Professor with the Department of Energy, Politecnico di Milano, where he also teaches electrical engineering.

Open Access funding provided by 'Politecnico di Milano' within the CRUI CARE Agreement