Inductive Transfer and Deep Neural Network Learning-Based Cross-Model Method for Short-Term Load Forecasting in Smarts Grids
Méthode de modèle croisé basée sur le transfert inductif et l'apprentissage par réseau neuronal profond pour la prévision de la charge à court terme dans les réseaux intelligents

Dabeeruddin Syed, *Member, IEEE*, Ameema Zainab, *Member, IEEE*, Shady S. Refaat, *Senior Member, IEEE*, Haitham Abu-Rub, *Fellow, IEEE*, Othmane Bouhali, *Member, IEEE*, Ali Ghrayeb, *Fellow, IEEE*, Mahdi Houchati, *Member, IEEE*, and Santiago Bañales, *Member, IEEE*

***Abstract*— In a real-world scenario of load forecasting, it is crucial to determine the energy consumption in electrical networks. The energy consumption data exhibit high variability between historical data and newly arriving data streams. To keep the forecasting models updated with the current trends, it is important to fine-tune the models in a timely manner. This article proposes a reliable inductive transfer learning (ITL) method, to use the knowledge from existing deep learning (DL) load forecasting models, to innovatively develop highly accurate ITL models at a large number of other distribution nodes reducing model training time. The outlier-insensitive clustering-based technique is adopted to group similar distribution nodes into clusters. ITL is considered in the setting of homogeneous inductive transfer. To solve overfitting that exists with ITL, a novel weight regularized optimization approach is implemented. The proposed novel cross-model methodology is evaluated on a real-world case study of 1000 distribution nodes of an electrical grid for one-day ahead hourly forecasting. Experimental results demonstrate that overfitting and negative learning in ITL can be avoided by the dissociated weight regularization (DWR) optimizer and that the proposed methodology delivers a reduction in training time by almost 85.6% and has no noticeable accuracy losses.**

***Résumé*— Dans un scénario réel de prévision de la charge, il est crucial de déterminer la consommation d'énergie dans les réseaux électriques. Les données relatives à la consommation d'énergie présentent une grande variabilité entre les données historiques et les nouveaux flux de données. Afin de maintenir les modèles de prévision à jour avec les tendances actuelles, il est important d'affiner les modèles en temps voulu. Cet article propose une méthode fiable d'apprentissage par transfert inductif (ITL), pour utiliser les connaissances des modèles de prévision de la charge par apprentissage profond (DL) existants, afin de développer de manière innovante des modèles ITL très précis à un grand nombre d'autres nœuds de distribution, en réduisant le temps d'apprentissage du modèle. La technique de regroupement insensible aux valeurs aberrantes est adoptée pour regrouper les nœuds de distribution similaires en grappes. L'ITL est considérée dans le cadre d'un transfert inductif homogène. Pour résoudre le problème de surajustement qui existe avec l'ITL, une nouvelle approche d'optimisation régularisée par le poids est mise en œuvre. La nouvelle méthodologie de modèle croisé proposée est évaluée sur une étude de cas réelle de 1000 nœuds de distribution d'un réseau électrique pour la prévision horaire à un jour. Les résultats expérimentaux démontrent que le surajustement et l'apprentissage négatif dans l'ITL peuvent être évités par l'optimiseur de régularisation de poids dissocié (DWR) et que la méthodologie proposée permet de réduire le temps de formation de près de 85,6 % et n'entraîne pas de perte de précision notable.**

***Index Terms*— Clustering models, inductive transfer learning (ITL), load forecasting, predictive models, smart grids.**

Manuscript received 3 April 2022; revised 16 July 2022 and 3 October 2022; accepted 3 March 2023. Date of publication 23 May 2023; date of current version 29 May 2023. This publication was made possible by NPRP12C-33905-SP-220 from the Qatar National Research Fund (a member of Qatar Foundation), and co-funding by IBERDROLA QSTP LLC. The authors gratefully acknowledge to project "Research Impact Initiative SP5" and the support of the Texas A&M University at Qatar. The open access funding is provided by Qatar National Library. *(Corresponding author: Shady S. Refaat.)*

Please see the Acknowledgment section of this article for the author affiliations.

Associate Editor managing this article's review: Waleed Ejaz.

Digital Object Identifier 10.1109/ICJECE.2023.3253547

I. Introduction

RECENTLY, electrical energy forecasting has received significant attention with developments in the areas of computational sciences and machine learning (ML). Accurate energy forecasting is crucial to the long- and short-term capacity planning of an electrical utility. It also provides benefits such as avoiding overgeneration and undergeneration of energy and assisting in efficient and sustainable energy generation. It helps utilities in operational decisions such as load switching, infrastructure development, enhancing reliability, providing

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

predictability, and scheduling maintenance of power systems such that there is minimal effect on the services delivered to the customers.

Data-driven methodologies have been used in different works to forecast energy with different time horizons leading to three branches: long-, medium-, and short-term forecasting [1]. The training of the ML models and achieving high accuracy of predictions requires a huge amount of historical energy consumption data. ML algorithms are mainly categorized into three types: supervised, unsupervised, and reinforcement learning models [2].

Short-term load forecasting (STLF) in smart grids has employed models, such as autoregressive integrated moving average (ARIMA) [3], linear regression (LR) [4], neural networks (NNs) [5], [6], [7], support vector machines (SVMs) [8], and random forests [9] in supervised learning. In unsupervised learning, dimensionality reduction models [10], [11], such as principal component analysis (PCA) and linear discriminant analysis (LDA), and clustering models [6], [12], such as k-Means and k-Medoids, have been used.

In smart grids, the data are generated at a very high frame rate [13]. At the distribution level of a nationwide grid, there are more than hundreds of thousands of distribution transformers. To provide hourly STLF, it is important to train these hundreds of thousands of ML models within the forecasting horizon of STLF. The proposed methodology aims to tackle this challenge with the clustering and inductive transfer learning (ITL) framework. In addition, at newly installed distribution nodes, an adequate amount of historical data may not be available. In cases of unavailability of large amounts of historical energy consumption data, it is required that the prediction models are trained with limited amounts of data to achieve sufficiently high accuracy. Furthermore, it is important to note that the supervised ML algorithms commonly presume that the training points and testing points belong to the same statistical data distribution and that large amounts of historical data are available [14]. However, the statistical data distribution and patterns of energy consumption have high variability between historical and future data points. Hence, it is crucial to transfer the knowledge obtained from models that are trained on historical data to develop and train ML models on current energy consumption data points. In this work, a methodology with the aim of knowledge transfer is presented. The methodology uses inductive transfer ML to transfer the knowledge from existing trained models to newer models or newer applications. Transfer learning (TL), in cases of low data availability, increases data variance and completes the voids due to missing records leading to more accurate predictions.

With the use of TL, a model trained on data following a statistical distribution can be improved to test with high accuracy on data following different distributions, unlike conventional ML models which perform effectively only when training and testing data follow the same statistical data distribution. The TL leverages the knowledge from past experience to use it with a different and new domain or with a new statistical distribution. The capabilities of TL have previously been utilized in diverse fields and have also been introduced in works on time series forecasting [15], [16], [17]. Nevertheless, these works did not consider the possibility of overfitting in TL models. When TL is applied, it is generally observed that the optimizer converges to a local minimum rather than a global minimum or a local minimum that provides near-true solutions. It is known that NN optimization is nonconvex. Although it is not always possible to converge to a global minimum, convergence to a near optimum solution is a must. The application of TL in models is more prone to overfitting and poor generalization. In this work, we have implemented dissociated weight regularization (DWR) in the weight update rule to break out of local minima, which in turn eliminates negative learning between different models. In addition, TL is integrated with an unsupervised clustering technique with a key reason to reduce the model training time by a large factor.

To the best of our knowledge, there has been no previous work that proposed the hybrid multistage approach involving outlier-insensitive clustering and overfitting-eliminating ITL. This work proposes a weight-regularized technique to eliminate negative learning and avoid overfitting while applying TL. The contributions of this work are summarized as follows.

1) STLF in very large electrical systems, such as nationwide grids, requires hundreds of thousands of models to be trained in a very short time. To overcome this challenge, a novel hybrid deep-learning (DL) and clustering-based ITL methodology is proposed to forecast short-term energy consumption at distribution nodes with faster convergence and in short times. This methodology aims to identify the distribution nodes that have similar trends of energy consumption, cluster these nodes together, and execute TL across different clusters.
2) TL models are prone to overfitting. The possibility of curbing the negative transfer of knowledge has been investigated. It was observed that the clustering-based approach and the proposed ITL between similar distribution units within a cluster eliminate the negative transfer of knowledge.
3) Furthermore, to avoid overfitting of TL models and to eliminate negative TL between dissimilar distribution units or across clusters, a novel DWR technique is proposed. DWR during optimizing the cost function while training DL models eliminates overfitting.
4) Different from the conventional method of developing models one each for a large number of distribution nodes, the proposed multistage methodology provides enhanced scalability with reduced training time and no loss in accuracy. The proposed approach decreases the count of models required for forecasting in a large grid network. The methodology can be scaled to any larger sized grid.
5) The ITL-based forecasting approach aims to alleviate the data absence problem that exists only at newly installed electric distribution nodes.

TABLE I
RELATED WORK

Category	Reference	Method	Evaluation Metrics	TL	Comments on merits or demerits
Statistics	[40]	Time series	MAE	No	Applicable to only small datasets. Low accuracy for larger datasets.
Machine Learning	[41]	SVM	MAPE	No	SVM and other machine learning methods are not appropriate for big data. In addition, the SVM model's training time increases super linearly with data size.
Deep Learning	[19]	Deep RNN	RMSE	No	Recurrent connections in the network tend to increase training time. Accuracy-time trade-off is not justified with RNN.
	This paper	Clus-TL-DWR-DNN	RMSE, MAPE, nRMSE, Train time	Yes	Clustering and TL contributes towards reduction of train time whilst DWR and DNN provide accuracy in forecasting yielding best accuracy-time trade-off.

6) The performance evaluation demonstrates that there is an 85.6% reduction in train time accomplished with the presented approach.

The remainder of this article is structured as follows. Section II discusses the related work in DL, clustering-based load forecasting, and TL. Section III presents the proposed methodology for utilizing transfer ML on DL models. Section IV presents a case study that has been conducted to validate the proposed methodology and discusses the results. Finally, Section V presents the conclusion of the research and future work.

II. RELATED WORK

A wide range of approaches used to anticipate short-term load is presented in the literature. They may be roughly categorized into three groups: statistics, ML, and DL techniques. Table I includes a selection of the most noteworthy publications from each category, outlining the proposed forecasting methodologies, as well as the evaluation metrics to validate the proposed methodology and the forecasting accuracies. Table I also shows whether TL was used in the actual research or not in addition to the limitations of the work.

In addition, the subsequent parts of this section present the related work in two divisions. The first division discusses the STLF methods based on DL and clustering approaches. The second division presents the literature on TL for load forecasting applications.

A. Load Forecasting

The need for highly accurate forecasting models and the advent of smart meters led to sensor-based forecasting models. The load forecasting models have been trained at various levels of a grid, such as household level, substation level, feeder level, and distribution nodes level. Different features, such as lag hour values of power demand, season, and weather variables, including temperature, cloud cover, humidity, and precipitation intensity, have been used to forecast energy consumption for short-, medium-, or long-term levels [18]. Since the models are data-driven, they require large volumes of data to generate highly accurate models. With the smart meters being installed and having been installed recently, huge volumes of data are not available at newly installed nodes of the smart grid. Hence, it is required to explore how the knowledge from trained models at nodes with huge historical data can be transferred to models at nodes with fewer data available.

Deep neural networks (DNNs) have been the forerunner in the generation of highly accurate forecasting models. Shi et al. [19] presented that the uncertainty in energy consumption could be modeled by the use of DL. It is also crucial that overfitting is avoided that generally prevails with a high number of layers in the DNNs. The authors proposed a novel pooling-based deep recurrent NN (RNN) to address the overfitting by increasing data variety and size. The case study was performed at the household level after developing a bespoke DL application with the TensorFlow framework and they reported that their proposed model performs up to 6.5% better in terms of root-mean-square error (RMSE) compared to classical deep RNNs.

Kong et al. [12] addressed the issue of uncertainties of load at the household level by the use of DNNs called long short-term memory (LSTM) with inherent long-term memory capabilities. Their work also included the electrical appliances' energy consumption in the training data and found that the accuracy improved curbing the uncertainty in load predictions. In addition, DL models have been used as part of ensemble models in various works to forecast energy consumption with higher accuracy. Cao et al. [20] used a deep belief network with bagging and boosting variants in an ensemble model.

Moreover, clustering techniques have been utilized in previous works to group similar customers, days, or weather conditions. The clustering techniques provide merits of reducing the variance of uncertainties within each cluster and, also, these decrease the count of models to be built for the same number of units when compared to nonclustering techniques. Goehry et al. [21] presented a methodology based on random forests and sequential expert aggregation showing that their proposed methodology performs better than the classical hierarchical clustering strategy. Wang et al. [22] employed a k-means clustering algorithm with better results when compared to nonclustering strategies. Other clustering algorithms employed for load forecasting include k-Medoids clustering [23] for similar day clustering, expectation–maximization clustering [24], Gaussian mixture clustering [25], density-based spatial clustering of applications with noise (DBSCAN) [26], and hierarchical clustering [25].

B. Transfer Learning

In the past decade, TL has gained widespread research interest from researchers in different fields of study due to its inherent capability of transferring the knowledge gained while training from one application to another. Ribeiro et al. [28] used TL with seasonal and trend adjustment to enhance the

forecasts of energy used in a building with the aid of models trained on data from similar buildings. An improvement of 11.2% in mean absolute percentage error (MAPE) of predictions was reported after the use of TL. Their work assumes the similarity of buildings in terms of energy consumption to apply TL and did not employ clustering-based techniques to group different buildings. Their case study also limits the application of TL to similar buildings. In this article, the clustering-based techniques are employed, and in addition, TL is applied to similar distribution nodes with an improvement of training time and testing accuracy and between dissimilar clusters using weight regularization with an improvement of time and accuracy.

In [15], energy predictive models based on convolutional NNs (CNNs) and TL are proposed. In this article, energy predictive models were tested on a case study of 23 customers against the seasonal ARIMA (SARIMA) model and fresh CNN model. The results proved that the performance in terms of accuracy is improved when the models are pretrained using TL.

Ye and Dai [16] proposed an ensemble model of online TL kernel-based extreme learning machines. The results presented in their work depict that the use of TL improves the performance in terms of accuracy compared to standard ML models. Their work utilizes extreme learning machines that are basically NNs with one hidden layer. The developed approach using extreme learning machines provided many benefits such as eliminating the need for optimizing the number of hidden layers and optimizing a smaller number of parameters [29]. Evident from diverse and numerous research works, the DL models display high accuracy while dealing with the time-dependent energy forecasting problems if the tendency to overfitting is controlled [30]. Hence, in our work, the use of TL is extended to DL models.

Qureshi et al. [17] proposed a two-stage prediction model for wind power based on an ensemble of nine deep autoencoders in the first phase and deep belief networks in the second phase. The work was based on five datasets from wind farms. The TL was utilized in the training of deep autoencoders from two to nine using the knowledge obtained during the training of the first deep autoencoder. Their results indicate that the use of TL overperforms the baseline regression models based on ARIMA and support vector regression (SVR). However, the performance of their ensemble model without the use of TL for autoencoders 2–9 has not been discussed. It is unclear if the improvement in performance is due to the ensemble of the optimized deep autoencoders or due to TL. In our work, the comparison is performed between the same DL model with and without TL to comment and discuss the accuracy and train time improvement due to the technique of TL specifically. Besides, the performance of our proposed methodology is evaluated against several benchmark forecasting models.

III. Proposed Methodology

In this section, a detailed introduction of the proposed methodology for STLF using ITL on clustering-based DL models is presented.

The aim of the methodology is multifold. The main objective of this work is to increase the accuracy of predictions of hourly energy consumption in a reasonable time frame. The methodology is applicable to make many predictions such as Photovoltaic (PV) power forecasting, wind energy forecasting, and energy consumption. Importantly, the aim of the methodology is to apply TL so that the knowledge, network structure, and network parameters are transferred from already existing trained models to newer models or tasks. For tasks with insufficient data to efficiently train a model, TL provides enhanced accuracy in forecasting. For other tasks, TL provides faster convergence of models reducing model training time.

A. Data Acquisition and Processing

1) Data Acquisition: In this work, real-world datasets from the nationwide Spanish Electrical Grid are utilized. The acquired data are power consumption records at the 1000 distribution nodes in the grid. The data consist of 24 072 709 time series hourly energy consumption samples between the period of 1 January 2017 and 28 September 2019. The lag hour values of energy, i.e., past energy consumption values, and season are added as features in the dataset for all of the models that are developed. In addition, the time series features, such as year, month, day, and hour, are appended as data attributes. The feature domain across all the tasks remains the same. The optimal number of lag hour values, to predict the hourly load consumption one day ahead in the future, is realized to be 24 from our previous work [6] on the same dataset. The dataset of the target model or target task is split into 80% for training, 10% for validation, and 10% for testing. Once the sliding window lag hour features and time series features are added to the dataset, it eliminates the autocorrelation between the consecutive recordings and generates a possibility of cross validation on this time series energy consumption data. Tenfold cross validation has been utilized in this work for performance evaluation.

For benchmarking case study, an open-source electricity load diagrams dataset [31] is utilized to provide a comparative evaluation of our proposed methodology against benchmark and state-of-the-art models. The data are available for 370 users and contain 140 256 power usage instances for each user.

2) Data Processing: For the given attributes A_1 and A_2, the normalization function ϕ is a linear transformation such that $\phi(A_1)$ and $\phi(A_2)$ values are in the same domain and possess a similar scale. The normalization function changes the values recorded at distinct scales to an identical scale and within a uniform domain such that these values can be compared and processed in conjunction. Data normalization enhances the accuracy of ML models, and hence, it is considered a crucial preprocessing technique. In this work, minimum–maximum (min–max) feature scaling is incorporated to bring attribute values within the range [0, 1]. The min–max scaling in the range [0, 1] is represented by the following equation:

$$a' = \frac{a - A_{\min}}{A_{\max} - A_{\min}} \tag{1}$$

where a' is the transformed value, a is the primitive value, $A_{\min}$ is the minimum value, and $A_{\max}$ is the maximum value of the attribute. If unnormalized input features are fed to ML models, the loss function is likely to have elongated valleys [32]. Optimizing a cost function raises an issue as the gradient steeps with respect to a few parameters. This in turn causes large oscillations in the search space of weights due to steep slope bounces. One way to compensate for this is optimization with a small learning rate. This in turn raises another problem of slower convergence, larger training time, and disproportionate weight assessment. Normalizing inputs makes loss function more symmetrical and, in turn, makes optimization easier to achieve. The gradients tend to point toward a global minimum even with a larger learning rate, thereby increasing accuracy, achieving faster convergence, and reducing training time.

B. Model Construction Stage

The proposed solution utilizes the outlier-insensitive clustering and ITL-based DNN model with weight regularization to improve the accuracy of predictions. The details of the proposed clustering-based and DWR ITL methodology for load forecasting are presented in Algorithm 1.

1) Clustering Phase: As depicted in Algorithm 1, the first phase of the proposed methodology is the clustering phase. Initially, the energy matrix E is constructed using the energy consumption data available from the smart meter at each distribution transformer. The constructed energy matrix E can be notated as follows:

$$\boldsymbol{E}_{\mathrm{TF}\times H} = \left(e_{i,j}\right) \in \mathbb{R}^{\tau,h} \tag{2}$$

where $\left(e_{i,j}\right)$ is a matrix of size tf $\times$ h, h denotes the number of hours for which the data are available, and τ denotes the number of distribution transformers.

For the 1000 distribution transformers dataset, the size of the matrix $\boldsymbol{E}$ is 1000 $\times$ 24 024. In the next step of Algorithm 1, the dissimilarity matrix is constructed based on the criterion function of Minkowski dissimilarity. The constructed square and symmetric dissimilarity matrix are notated as follows:

$$\mathrm{DM}_{\mathrm{TF}x\mathrm{TF}} = \left(d_{i,k}\right) \in \mathbb{R}^{\tau,\tau}. \tag{3}$$

The pairwise dissimilarity between any two distribution transformers i and k is calculated as follows [6]. As an attempt to reduce the NP-hardness of the optimization, the Minkowski order (q) is fixed to first order

$$d_{i,k} = \left(\sum_{j=1}^{h} \left|E_{i,j} - E_{k,j}\right|^{q}\right)^{\frac{1}{q}}. \tag{4}$$

Once constructed, the dissimilarity matrix is passed as an argument to the clustering algorithm along with the optimized hyperparameter k_{opt} that represents the optimal number of clusters. The hyperparameter k_{opt} in the k-Medoids algorithm cannot be learned directly, and hence, the elbow curve method is employed to discover the optimal value of k, which yields the least within-cluster error [6]. The elbow curve is the illustration analysis between the number of clusters (k) and the within-cluster sum of squares error and the elbow or the dip in the curve reveals the optimal number of clusters k_{opt}. An outlier-insensitive k-Medoid clustering algorithm is adapted to group similar distribution nodes into clusters. At the low level, the clustered models for the grouped profiles and the individual models for each transformer are devised using a DNN framework.

Algorithm 1 Proposed Methodology

Input:
D: distribution node energy datasets, δ: weight decay factor,
i_{max}: maximum number of iterations while training, L: no. of layers
Initialize Cluster number$k = 1$, $a = 1$.
Outputs:
Phase 1: Clustering phase
1: Compute Energy matrix E of all transformers.
2: Compute Dissimilarity Matrix DM using (4).
3: Determine clusters of similar transformers using k-Medoid clustering and elbow curve method [6]. Return optimal number of clusters as k_{opt}.
Method 1: sourceModel $(\boldsymbol{D_S^1})$
4: Train a base model (M_S^1) on D_S^1 using proposed weight regularized optimizer and return the parameters of trained model. Return M_S^1.
Method 2: TLmodelDevelopment $(\boldsymbol{M_S^1}, \boldsymbol{D_T})$
5: Initialize target model attributes & parameters with those of M_S^1.
6: for $i = 1, 2, \ldots, i_{max}$ do the following on Data D_T
7: for $l = 1, 2, 3, \ldots, L$ do
8: do forward propagation as follows:
9: compute $\underline{S}^l = \left(\Theta^l\right)^T \underline{x}^{l-1}$
10: compute $\underline{x}^{(l)} = \begin{bmatrix} 1 \\ \sigma\left(\underline{S}^l\right) \end{bmatrix}$
11: end for
12: compute $h\left(\underline{x}\right) = \underline{x}^{(L)}$
13: for $l = L, L-1, L-2, \ldots, 1$ do
14: do backward propagation as follows:
15: compute f_j^l in all unfrozen layers
16: end for
17: update weights: $\theta_j^{(l)} \xleftarrow{w} (\delta) * \theta_j^{(l)} \eta x_i^{l-1} f_j^l$.
18: Iterate till variations of $\theta_j^{(l)} \leq \varepsilon$.
19: end for
20: Model (M_T) is developed. return M_T.
Phase 2: Transfer learning within cluster
21: for $k = 1$ to k_{opt} do
22: A t/f 1 in cluster k is selected as source domain i.e., $D_S^1 = D_{k[a]}$
23: Invoke *sourceModel* method with argument D_S^1. M_S^1 is returned.
24: for τ in cluster k from $k[a+1]$ to $k[-1]$ do
25: *TLmodelDevelopment* is invoked with arguments M_S^1 & D_τ.
26: Model (τ, k) = returned model M_T
27: increment τ.
28: end for
29: end for
Phase 3: Transfer learning between dissimilar clusters
30: Cluster 1 is selected as source domain i.e., $D_S^1 = D_{k=1}$
31: Invoke *sourceModel* method with argument D_S^1. M_S^1 is returned.
32: for $k = 2$ to k_{opt} do
33: $TLmodelDevelopment$ is invoked with argument $M_S^1 \& D_k$.
34: Model(k) = returned modelM_T.
35: Increment k.
36: end for

2) Transfer Learning: TL is a technique of ML in which the knowledge gained during the training of a model on a domain of features is leveraged to improve the performance of training another model or task on the same or different domain of features [33]. TL eliminates the assumption that the training data and testing data observe the same data distribution. The merits of TL are the following: training is done with less or little data, training gets faster, and model accuracy increases.

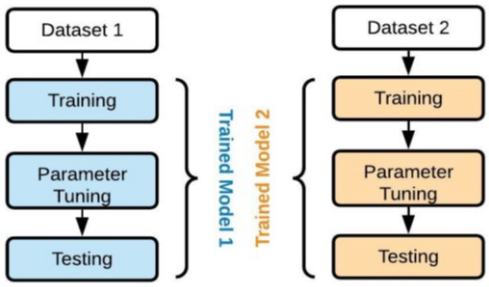

Fig. 1. Traditional learning.

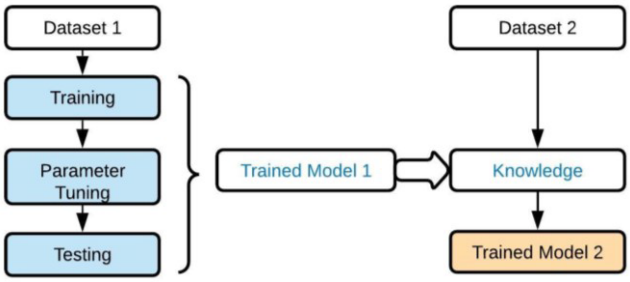

Fig. 2. TL.

Consider that feature domain F_s, label V_s, and task T_s correspond to the source application, and feature domain F_t, label V_t, and task T_t correspond to the target application. The TL aims to improve the performance of task T_t using the knowledge obtained in task T_s, where $T_s \neq T_t$.

Fig. 1 shows the process of traditional ML where the knowledge gained after training one model is not retained or reused in further models. The training of a newer model or task is executed from scratch. Fig. 2 shows the process of transfer ML where the knowledge gained after training one model (trained model 1 in Fig. 2) is transferred to further models (model 2). The weights, knowledge of features, and the network structure are transferred to the training stage for the new task.

The TL process has the benefits of improving the baseline performance of predictions and improving the time to train an ML model [14]. The following are the multiple types of TL algorithms.

1) *Transductive TL (Data Features Are Not the Same Between the Different Tasks) [34]:* If the tasks T_s and T_t that are different infer that the source domain F_s and the target domain F_t are also different, then it is called transductive TL.
2) *ITL (Data Features Are the Same Between Different Tasks) [35]:* If the tasks T_s and T_t that are different infer that the source domain F_s and target domain F_t are the same, then it is called ITL. If the source label V_s exists, then this learning is called multitask learning. The learning is unsupervised in the absence of labels in the tasks, and in such cases, the algorithm is called self-taught TL.
3) *Unsupervised TL [36]:* In this type of learning, the source tasks T_s and T_t are different, the domains F_s and F_t are similar, and the labels are not available in both tasks.

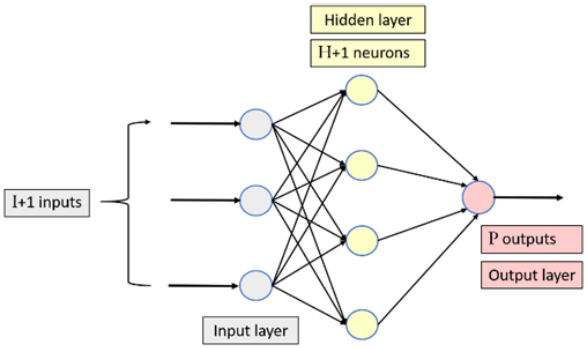

Fig. 3. NN perceptron.

a) Theoretical perspective of TL in cross-model load forecasting using neural networks: Consider a trained NN structure with three layers, as shown in Fig. 3. The input layer with $I+1$ inputs with $(I+1)$th node as bias node, $H+1$ hidden units with $(H+1)$th node as bias node, and P outputs. Consider that the NN model is already trained on training data with N records, i.e., $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$. Since the training is complete, it is safe to assume that the optimal weights have been determined with objective function on minimum training error. Consider that the weights between the input-hidden connections and hidden-output connections are w_{ih} and v_{hp}, respectively, where $1 \leq i \leq I+1$, $1 \leq h \leq H+1$, and $1 \leq p \leq P$. With TL, it is expected to train the model with training record $N+1$ (refers to training record from the new dataset) input x_{N+1} such that the predicted value from the model is equal to the true value of output, i.e., $y_{N+1} = \hat{y}_{N+1}$. The transfer of training with data from a new dataset should minimize the effect on training errors $E_n(1 \leq n \leq N)$ of previous historical data, i.e., minimize the weight sensitivity. The cost objective for weight sensitivity can be given by $T \triangleq (1/8)\sum_{n=1}^{N}\sum_{p=1}^{P} E_{np}^2$. The goal of TL is to determine the weights $4w_{ih}(N+1)$ and $v_{hp}(N+1)$ such that these do not have any effect on weight sensitivity represented by the objective function (S) that balances the tradeoff between weight sensitivity objective function T and error of prediction, for instance, $N+1$. The objective function S is given by the following:

$$S \triangleq T + \frac{\lambda}{2}\sum_{p=1}^{P}\left(y_{(N+1)(p)} - \hat{y}_{(N+1)(p)}\right) \tag{5}$$

where λ is the tradeoff coefficient to balance the evolutionary training error and preevolutionary training error

$$T \triangleq \frac{1}{8}\sum_{n=1}^{N}\sum_{p=1}^{P} E_{np}^2 \tag{6}$$

$$T \triangleq \frac{1}{8}\sum_{n=1}^{N}\sum_{p=1}^{P}\left(\sum_{ih}^{(I+1)H}\frac{\delta E_{np}}{\delta w_{ih}} w_{ih}(N+1) + \sum_{h}^{(H+1)}\frac{\delta E_{np}}{\delta v_{hp}} v_{hp}(N+1)\right)^2. \tag{7}$$

The weight sensitivities of change in error can be given by (10) and (13)

$$\frac{\delta E_{np}}{\delta w_{ih}} = \frac{\delta}{\delta w_{ih}}\left(y_p(n) - \widehat{y_p}(n)\right)^2 \tag{8}$$

$$\frac{\delta E_{np}}{\delta w_{ih}} = 2 * \left(y_p(n) - \widehat{y_p}(n)\right)\left(0 - \frac{\delta \widehat{y_p}(n)}{\delta w_{ih}}\right) \quad (9)$$

$$\frac{\delta E_{np}}{\delta w_{ih}} = -2 * \left(y_p(n) - \widehat{y_p}(n)\right)\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} \quad (10)$$

$$\frac{\delta E_{np}}{\delta v_{hp}} = \frac{\delta}{\delta v_{hp}}\left(y_p(n) - \widehat{y_p}(n)\right)^2 \quad (11)$$

$$\frac{\delta E_{np}}{\delta v_{hp}} = 2 * \left(y_p(n) - \widehat{y_p}(n)\right)\left(0 - \frac{\delta \widehat{y_p}(n)}{\delta v_{hp}}\right) \quad (12)$$

$$\frac{\delta E_{np}}{\delta v_{hp}} = -2 * \left(y_p(n) - \widehat{y_p}(n)\right)\frac{\delta \widehat{y_p}(n)}{\delta v_{hp}}. \quad (13)$$

From (10) and (13), we modify (7) as follows:

$$T \triangleq \sum_{n=1}^{N}\sum_{p=1}^{P}\left[\sum_{ih}\left[-\left(y_p(n) - \widehat{y_p}(n)\right)\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} w_{ih}(N+1)\right]\right] + \sum_{h}\left[-\left(y_p(n) - \widehat{y_p}(n)\right)\frac{\delta \widehat{y_p}(n)}{\delta v_{hp}} v_{hp}(N+1)\right]^2. \quad (14)$$

Let $y_p(n) \triangleq \sum_h^{H+1} u_h(n) v_{hp}$ and $u_h(n) \triangleq f\left(u_h^*(n)\right)$, where $f(\cdot)$ is the activation function of the hidden layer neuron and $u_h^*(n) \triangleq \sum_i^{I+1} x_i(n) w_{ih}$. It is important to note that $x_{I+1}(n) = 1$ and $u_{H+1}(n) = 1$ since the input node $I+1$ and hidden node $H+1$ are bias neurons in the artificial NN considered. Therefore, the change of prediction with respect to the weights in the hidden-to-output layer connections is given by the following:

$$\frac{\delta \widehat{y_p}(n)}{\delta v_{hp}} = u_h(n) \quad (15)$$

where $1 \le h \le H+1$ and $1 \le p \le P$.

The change of prediction with respect to the weights in the input-to-hidden layer connections is given by the following:

$$\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} = \left[\frac{\delta \widehat{y_p}(n)}{\delta u_h(n)}\right]\left[\frac{\delta u_h(n)}{\delta w_{ih}}\right] \quad (16)$$

$$\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} = \left[\frac{\delta \widehat{y_p}(n)}{\delta u_h(n)}\right]\left[\frac{\delta u_h(n)}{\delta u_h^*(n)}\right]\left[\frac{\delta u_h^*(n)}{\delta w_{ih}}\right] \quad (17)$$

$$\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} = v_{hp}(n)\frac{\delta f(x)}{\delta x} x_i(n) | \{x = u_h\} \quad (18)$$

$$\frac{\delta \widehat{y_p}(n)}{\delta w_{ih}} = v_{hp}(n) u_h(n)(1 - u_h(n)) x_i(n) \quad (19)$$

where $1 \le i \le I+1$, $1 \le h < H$, and $1 \le p \le P$.

It implies that

$$T \triangleq \frac{1}{8}\sum_{n=1}^{N}\sum_{p=1}^{P}\left[-\left(y_p(n) - \widehat{y_p}(n)\right)\sum_{ih}^{(I+1)H} \times \left[v_{hp}(n) u_h(n)(1 - u_h(n)) x_i(n) w_{ih}(N+1)\right] - \left(y_p(n) - \widehat{y_p}(n)\right)\sum_{ih}^{(I+1)H}\left[u_h(n) v_{hp}(N+1)\right]\right]^2. \quad (20)$$

In TL, we try to minimize the objective function S that balances the tradeoff between minimizing weight sensitivity T on a historically trained model and the error of predictions on data from a new dataset, i.e., $S \triangleq T + (\lambda/2)(\sum_{p=1}^{P}[(y_p(n+1) - \widehat{y_p}(n+1))]$.

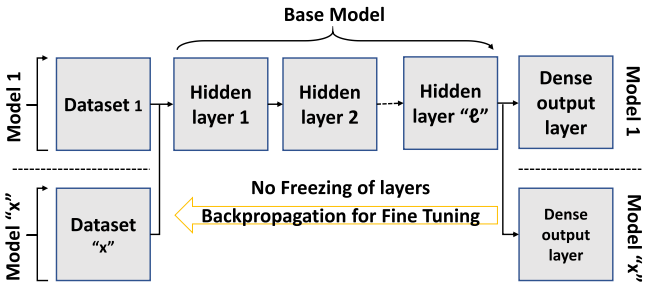

Fig. 4. Homogeneous ITL through fine-tuning.

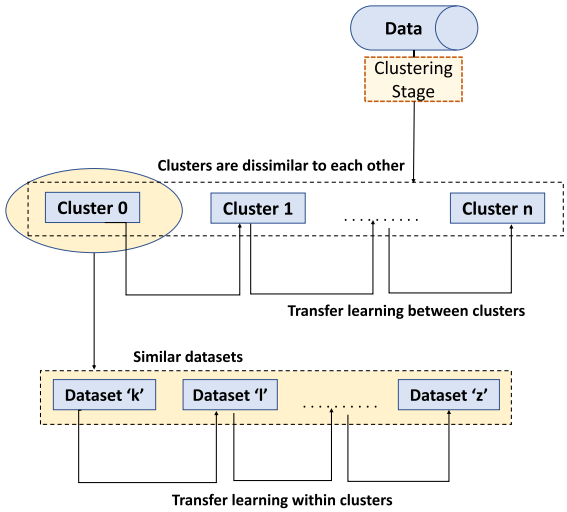

Fig. 5. Clustering-based methodology with TL.

b) Inductive TL in the proposed methodology: In this work, we use homogeneous ITL by fine-tuning through all layers for target tasks. The homogeneous TL is shown in Fig. 4. As shown in Fig. 4, dataset 1 is employed to train model 1 from scratch, i.e., the weights of hidden layers in the base model are optimized. During the development of model x, the base layers from model 1 are utilized without freezing and the fine-tuning is performed through all layers.

The overall methodology of the construction of load forecasting models is shown in Fig. 5. The data of 1000 distribution nodes are passed through the clustering stage to form a group of similar distribution nodes into clusters. The optimal number of clusters is determined to be 93 clusters [6]. Similar distribution nodes are formed into clusters.

In the next stage of methodology, a forecasting model one each for a cluster is developed using TL, that is, a forecasting model (model 0) is first trained from scratch on the source dataset (cluster 0). Second, the model is retrained on target datasets (cluster 1, cluster 2, ..., cluster n) through fine-tuning all the layers in the NN. For convenience, the clustered models formed using TL are denoted as Clus-TL-DNN and the clustered models formed without TL framework are denoted as Clus-DNN where DNN indicates the inherent DL NN model. The accuracies of Clus-TL-DNN are compared with those of Clus-DNN.

TABLE II
TUNED HYPERPARAMETERS OF THE BEST PERFORMING DL MODEL

Hyperparameter	Value
Batch size	128
Epochs	50
Hidden layer activation	ReLU
Weight initialization	Xavier normal
Loss function	Mean-squared error
Model optimizer	DWR with Adam

The next stage of the proposed methodology involves the creation of models within clusters. These are individual models developed for each dataset. Already, the datasets, which are similar in energy consumption patterns, have been clustered together in the previous stage. Now, the knowledge transfer is performed only between the distribution datasets within the same clusters to eliminate any negative transfer of knowledge. In the first subset of experiments, TL is used to construct the subsequent models within a cluster using knowledge transfer from the source domain within the same cluster. For convenience, these models are denoted as Ind-TL-DNN. To develop source domains from cluster 1 onward, we utilize a weight regularization optimizer to transfer knowledge from source domain within cluster 0. The use of weight regularization eliminates negative learning when knowledge transfer occurs between clusters. In another subset of experiments, the individual models are developed without the use of any TL. For convenience, these models are denoted as Ind-DNN.

The models are compared using the RMSE or MAPE for accuracy and training time for execution time. RMSE is a metric of forecasting accuracy in statistics and is given by (21). Also, MAPE is represented by (22)

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(P_i' - P_i\right)^2} \tag{21}$$

$$\text{MAPE} = \frac{100}{M}\sum_{i=1}^{M}\left|\frac{P_i' - P_i}{P_i}\right| \tag{22}$$

where P_i' is the forecast load demand, P_i is the actual load, and M denotes the number of data points.

3) Deep Neural Network: To obtain an efficient and accurate DNN model, a search space for hyperparameters, such as weight initialization strategy, number of hidden layers, number of neurons, activation function, batch size, training epoch, and learning rate, was defined. After a search space was defined, a halving randomized search CV method was employed for hyperparameter tuning. Multiple comparative experiments were performed to confirm the model hyperparameters that are mentioned in Table II. The best performing DL model determined considering training time and accuracy was a DNN consisting of one input layer, four hidden layers, and one output layer. The number of neurons in the input layer equaled the number of independent data attributes. The number of neurons in the hidden layers was set to 75, 50, 40, and 30. The output layer consisted of one node because the model tackled regression. Rectified linear unit (ReLU) activation function [37] was used as activation in the hidden layers, whereas the identity function was used as activation in the output layer. The loss function utilized was the mean-squared error. Adam optimizer and its proposed DWR invariant had been employed for optimization. The models were implemented on a Keras framework. The training process used the Xavier normal weight initialization strategy. Based on the epoch-convergence history graph, the optimal number of epochs was set to 50 with a batch size of 128. After training, the models are saved as .pkl files for later use. The old models are used as starting points for training newer models with the help of TL to achieve faster convergence.

TABLE III
TESTING OF CLUSTERED MODELS ON CLUSTER DATA WITH AND WITHOUT TL FRAMEWORK APPLIED BETWEEN CLUSTERS

	RMSE (kWh)		Improvement (%)
	Clus-DNN	Clus-Tr-DNN	
Cluster 0	4.35	-	-
Cluster 1	12.86	9.92	+22.86 %
Cluster 2	17.10	23.68	-34.47 %

IV. EXPERIMENTAL RESULTS

Extensive experiments were performed to evaluate the performance of the ITL-based methodology. The utilized datasets are the power consumption records at ten and 1000 distribution nodes in the electrical network.

In one set of experiments, individual models are developed using the individual datasets, and in another set of experiments, the clustering approach is applied to group the similar distribution nodes into groups depending on the similarity metric of hourly power usage.

The employed approach is the k-Medoid clustering technique to eliminate the sensitivity to outliers in data analytics. According to the within-cluster error elbow curve, the ideal count of clusters is determined as 3 for ten distribution nodes data and as 93 for 1000 distribution nodes dataset [6].

The initial cluster (cluster 0) is trained using the conventional way without any TL. The other clusters are trained with the help of TL from cluster 0 and the fine-tuning is performed using the corresponding dataset of the cluster. The knowledge from the training of cluster 0 is used for training cluster 1, cluster 2, and so on.

A. Results on Ten Distribution Nodes Dataset

The performance of traditional learning and TL between dissimilar clusters on clustered models for ten distribution nodes dataset is shown in Table III. The RMSE of cluster 1 shows significant improvement after the transfer of knowledge. However, the performance of the model for cluster 2 shows a negative transfer of learning, indicating that the model converged to a local minimum rather than a global optimization point. The negative learning can be explained because the TL is performed between the dissimilar distribution nodes

TABLE IV
TESTING OF CLUSTERED MODELS ON INDIVIDUAL DISTRIBUTION NODE DATASETS WITH AND WITHOUT TL FRAMEWORK APPLIED BETWEEN CLUSTERS

	RMSE (kWh)		Improvement (%)
	Clus-DNN	Clus-TL-DNN	
tf 0	27.95	-	-
tf 1	31.39	-	-
tf 2	29.91	-	-
tf 3	27.50	-	-
tf 4	21.15	22.29	5.39
tf 5	18.49	12.97	29.85
tf 6	21.75	20.920	3.81
tf 7	16.52	20.423	-23.62
tf 8	16.19	19.32	-19.33
tf 9	17.47	20.52	-17.45

belonging to different clusters. A few potential solutions that can be considered to avoid convergence to local minima are the following [38], [39]: 1) considering cyclic learning rate; 2) using stochastic gradient descent (SGD) with warm restarts; 3) considering high values for learning rate; 4) using metaheuristic algorithms such as gray-wolf algorithm, ant colony optimization, and harmony search; and 5) variants of optimizers such as vanilla gradient descent, QHAdam, YellowFin, AggMo, QHM, and Demon. The negative TL can be removed when the transfer of knowledge happens between the distribution nodes that are similar. This is observed in subsequent tables. Moreover, the improvement with TL is more pronounced when the data for target tasks are not sufficiently large. In this work, weight regularization is utilized along with Adam optimizer to eliminate negative learning when the knowledge transfer is to occur between dissimilar clusters.

The ten distribution nodes are clustered into three clusters. With the k-Medoid clustering algorithm, it was determined that the three clusters of distribution nodes are: {0, 1, 2, 6}, {5}, and {3, 4, 7, 8, 9}. One clustered model based on DNNs was developed for each cluster. Thus, the three clustered models have been developed and these have been tested on the individual datasets of the distribution nodes and the results of the performance with and without the use of TL are shown in Table IV. The first column in Table IV represents the distribution node number or transformer number (tf). The similar distribution nodes are grouped into the same clusters; however, any two clusters are assumed to be dissimilar. With the transfer of knowledge between dissimilar clusters, it is possible that the transfer is either positive or a little on the negative side. However, the gain in the execution or training time is always positive. The gain in time is shown in Table V. From Table V, it is clear that the time to train the models with TL is much less than the time to train the models without TL.

If TL is between similar distribution nodes, there is no negative knowledge transfer. The k-Medoid clustering algorithm depending on the criterion of similar energy usage clustered the ten distribution nodes into the clusters {0, 1, 2, 6}, {5}, and {3, 4, 7, 8, 9}. To eliminate the negative TL, the knowledge

TABLE V
CLUSTER TRAINING TIMES AFTER TESTING OF CLUSTERED MODELS WITH TL APPLIED BETWEEN CLUSTERS

	Training time (s)		Improvement (%)
	Clus-DNN	Clus-TL-DNN	
Cluster 0	92	-	-
Cluster 1	49	40	9
Cluster 2	79	50	36.7

TABLE VI
TESTING OF INDIVIDUAL MODELS ON INDIVIDUAL DISTRIBUTION NODE DATASETS WITH TL APPLIED WITHIN CLUSTERS

	RMSE (kWh)		Improvement (%)
	Ind-DNN	Ind-TL-DNN	
tf 0	13.60	-	-
tf 1	10.32	7.90	23.44
tf 2	7.35	5.03	31.56
tf 3	6.42	1.09	83.02
tf 4	18.68	-	-
tf 5	2.18	-	-
tf 6	2.30	1.26	45.21
tf 7	14.91	10.34	30.65
tf 8	3.81	2.22	41.73
tf 9	3.70	2.52	31.89

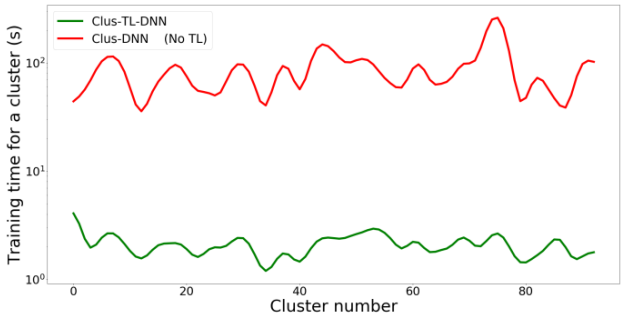

Fig. 6. Cluster training times for 1000 distribution nodes with TL applied between clusters.

from the model of dataset 0 should be transferred only to develop models on datasets 1, 2, and 6. Dataset 5 should have its model developed from scratch. The knowledge from the model of dataset 3 should be transferred to develop models on datasets 4, 7, 8, and 9. The use of the clustering-based methodology eliminated any negative TL and the results are described in Table VI. The negative TL between dissimilar clusters is eliminated by the weight regularization technique proposed in Section IV-C.

B. Results on 1000 Distribution Nodes Dataset

The performance of TL with respect to training time has also been verified with a second case study on 1000 distribution nodes that, according to elbow curve and k-Medoid clustering, were grouped into 93 clusters, and the models were developed using DNNs. As shown in Fig. 6, the time to train the clustered models using TL is always less when compared to the time taken to train the clustered models without TL. This confirms that the TL allows for faster convergence of models.

TABLE VII
PERFORMANCE OF TL ON 1000 DISTRIBUTION NODES DATASET

Model	Training time (min)	Average MAPE (%)	Average RMSE (kWh)
Ind-LR [6]	0.23	20.32	54.04
Clus-LR [6]	0.28	20.84	62.35
Ind-ARIMA [6]	8.55	37.78	59.41
Clus-ARIMA [6]	5.28	39.45	67.87
Ind-LSTM [6]	1890	7.27	22.52
Clus-LSTM [6]	485	11.11	37.06
Ind-DNN	140.15	7.18	19.82
Clus-DNN	77	7.22	21.25
Clus-TL-DNN	**3.23**	14.37	31.96
Clus-TL-DWR-DNN (Proposed)	20.17	**7.20**	**22.10**

The performance of TL in coalition with the clustering layer on the 1000 distribution nodes dataset is shown in Table VII. It takes 3.23 min to develop 93 clustered models using TL when compared to 2.20 h of training time without TL. However, the MAPE varies from 7.22% to 14.37% when TL is employed between dissimilar clusters.

C. Weight Regularization to Eliminate Negative Learning Between Dissimilar Datasets

For TL between dissimilar clusters, utilization of an improved Adam optimizer was proposed to eliminate any negative learning and to break out from local convergence. The first optimization step involves the use of a cyclical learning rate in which the learning rate is initialized to a larger value and is scheduled to decrease subsequently to prevent the avoidance of global minima. The proposed optimizer invariant is utilized with DWR and cyclical learning rate to eliminate overfitting and to break out from local minima toward the global minimum.

The weight update rule in the general Adam optimizer is given by the following:

$$\theta(t) = \theta(t-1) - \alpha f \tag{23}$$

where f is the effective gradient term and α is the learning rate.

The general Adam optimizer is characterized by a large step size when gradient change is less and a smaller step size when gradient change is rapid, and the adaptability in step size is performed by maintaining moving averages (called moments) of gradient over the steps.

The implemented optimizer invariant employs DWR. This allows for weight regularization without the association of hyperparameters such as learning rate (α) and weight decay factor (δ).

The weight update rule in the proposed DWR-Adam optimizer invariant is given by the following:

$$\theta(t) = (\delta)\theta(t-1) - \alpha f. \tag{24}$$

The weight decay factor is introduced as a coefficient to the weight of the previous iteration and lies between 0 and 1.

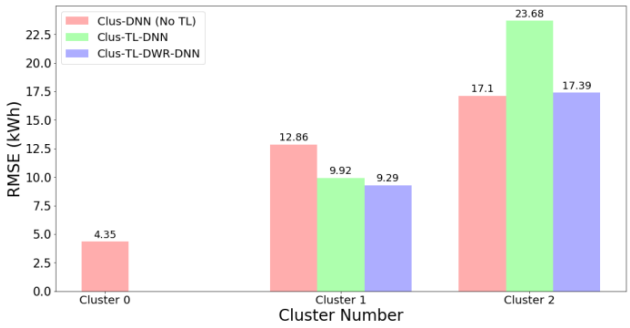

Fig. 7. TL between clusters—testing on cluster data.

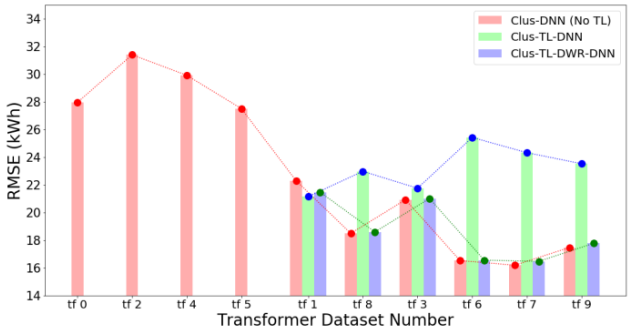

Fig. 8. TL between clusters—testing on individual transformer data.

This forces the weights learned to be small, and thus, the model generalizes better. For convenience, the clustered models using DWR are denoted by Clus-TL-DWR-DNN.

1) Weight Regularization on Ten Nodes Dataset: Figs. 7 and 8 show the performance of TL after weight regularization on ten distribution nodes dataset. The results, obtained after the testing of clustered models is performed on cluster data, are shown in Fig. 7. The graph of TL with weight decay regularization is at the lower bound of error when compared to the model development without TL for clusters 1 and 2. At no point, the error is high in the case of model development after TL. This indicates that the negative learning has been eliminated by the use of weight regularization in the optimizer.

The results, obtained after the testing of clustered models on individual transformers' data, are shown in Fig. 8. The graph of TL with weight decay regularization is at the lower bound of error when compared to the model development without TL for all the transformers, including tf 1, tf 8, tf 3, tf 6, tf 7, and tf 9. At no point, the error is high in the case of model development after TL. This corroborates that the negative learning has been eliminated by the use of weight regularization in the optimizer.

2) Weight Regularization on 1000 Nodes Dataset: The performance of TL after weight regularization on the 1000 distribution nodes dataset is presented in Table VII. To analyze the performance of the proposed weight regularization TL modeling (Clus-TL-DWR-DNN), several state-of-the-art benchmark models, including LR, ARIMA, and deep LSTMs, are selected as comparative methods, as shown in Table VII. Weight regularization utilized during objective function optimization in the proposed model eliminates negative knowledge transfer.

TABLE VIII
PERFORMANCE OF TL WHEN THE DATA AVAILABILITY IS LOW

Datasize	RMSE (kWh)		Improvement (%)
	Ind-DNN	Ind-TL-DNN	
5%	33.9412	20.9255	38.34
20%	33.8498	20.3398	39.91
30%	33.6460	19.9956	40.57
40%	33.0107	18.5012	43.95
50%	30.1865	17.7643	41.15
60%	17.5656	13.4901	23.20
70%	14.3533	12.4246	13.43
80%	14.0287	11.8557	15.48
95%	11.5013	8.8494	23.05

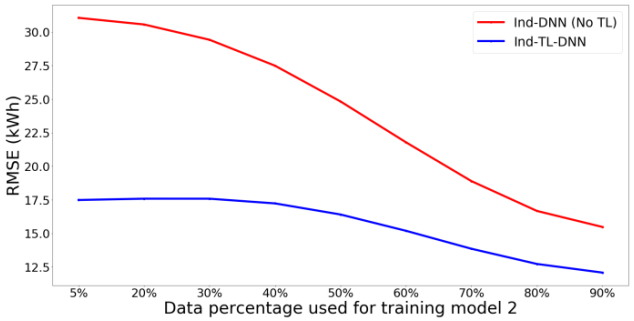

Fig. 9. TL results when the data availability is low.

TABLE IX
COMPETITIVE EVALUATION AGAINST STATE-OF-THE-ART MODELS

Model	Total Train time (s)	Total Test time (s)	Avg. nRMSE	Average MSE (10^{-2} kWh)
ELM	5205.8540	15.3339	0.1068	5.395
Encoder-decoder LSTM	24566.0800	52.3850	**0.0659**	**3.026**
Hybrid CNN-LSTM	18197.2700	235.3232	0.1098	5.403
ConvLSTM	1424480.25	582.9565	0.1073	5.701
Stacked Bi-LSTM	27099.8500	60.9294	0.0880	4.341
Clus-DNN	210.6670	0.6662	0.1563	6.306
Clus-TL-DWR-DNN (Proposed)	**10.2508**	**0.6431**	0.1057	5.326

The proposed Clus-TL-DWR-DNN has a higher overall development time of 20.17 min while maintaining an average MAPE error to a minimum of 7.20% when compared to clustering-based TL modeling that has 3.23 min as development time and an average MAPE of 31.96%.

D. Results on Targets With Smaller Datasets

Besides, the effect of TL has been analyzed with smaller datasets. As observed in Fig. 9, for smaller datasets, the model developed from scratch has low accuracy when compared to the model with knowledge transferred from a similar distribution point. As the size of the dataset increases, the accuracy of both the models, with and without TL, increases, and when a threshold size is reached, these models have very close accuracy values. The results of the performance of TL, when the data availability is low, are verified on the available dataset (see Table VIII). As shown in Table VIII, the model with TL performs 38% better than the model without TL when the data size for the second model is 5% of the original dataset. In all the cases of data availability, the TL model outperforms the conventional model by 13%–43%.

E. Benchmark Case Study for Competitive Evaluation

The proposed multilayer methodology is compared against the state-of-the-art models generated on a normalized benchmark dataset and the results are tabulated in Table IX. As shown in Table IX, the proposed model has the least training time of 10.2506 s and a highly competitive accuracy with an nRMSE of 0.1057.

V. CONCLUSION AND FUTURE WORK

This article proposed a methodology to develop highly accurate trained models even in case of the unavailability of historical data in large quantities. The methodology employs an ITL mechanism to improve the accuracy of the newer models from the knowledge gained during the training of a similar task in the past. The proposed TL model not only improves the accuracy for smaller datasets but also improves the execution time to reach convergence for any size of training data. The effectiveness of the proposed methodology is verified through a case study of hourly energy forecasting where the model predicts hourly load 24 h ahead of time and the used features are 24 past lag values, season, and time series extracted features. The set of experiments was executed for multiple distribution energy datasets while using clustering and additionally, without clustering. The DNNs are used for training the forecasting regression models. The proposed methodology enables the use of the trained TL models from the scenario where large quantities of historical energy consumption data are available to the scenario where the available data are small. To eliminate the negative transfer of knowledge, the TL is employed between datasets with similar energy consumption patterns and similar datasets are determined by the first stage of clustering in the proposed methodology. In cases of knowledge transfer between dissimilar clusters, the proposed weight regularization-based TL approach eliminates negative learning. The overall results indicate that the knowledge transfer using the proposed methodology improves the accuracy of newer models, reduces the time of convergence, and reduces training time for DL models compared to that of models without TL.

In future studies, we plan to utilize different correlation coefficients instead of clustering techniques to determine the similarity between distribution nodes before employing TL between similar nodes. In this work, only one dataset is considered as a source dataset disregarding the fact that the other datasets may contain useful patterns for the target task. Hence, in the future, we plan to perform multisource TL to enhance the accuracy performance.

ACKNOWLEDGMENT

Dabeeruddin Syed is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA, and also with the Department of Electrical and Computer Engineering, Texas A&M University at Qatar, Doha, Qatar (e-mail: syed.dabeeruddin@qatar.tamu.edu).

Ameema Zainab is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: azain@tamu.edu).

Shady S. Refaat, Haitham Abu-Rub, and Ali Ghrayeb are with the Department of Electrical and Computer Engineering, Texas A&M University at Qatar, Doha, Qatar (e-mail: shady.khalil@qatar.tamu.edu; haitham.abu-rub@qatar.tamu.edu; ali.ghrayeb@qatar.tamu.edu).

Othmane Bouhali is with the Research Computing, Texas A&M University at Qatar, Doha, Qatar, and also with the Qatar Computing Research Institute, Hamad Bin Khalifa University, Doha, Qatar (e-mail: othmane.bouhali@qatar.tamu.edu).

Mahdi Houchati and Santiago Bañales are with Iberdrola Innovation Middle East, Doha, Qatar (e-mail: mhouchati@iberdrola.com; sbanales@iberdrola.com).

REFERENCES

[1] A. K. Srivastava, A. S. Pandey, and D. Singh, "Short-term load forecasting methods: A review," in *Proc. Int. Conf. Emerg. Trends Elect. Electron. Sustain. Energy Syst. (ICETEESES)*, Mar. 2016, pp. 130–138.

[2] O. H. Abu-Rub, Q. Khan, and S. S. Refaat, "Multi-level defects classification of partial discharge activity in electric power cables using machine learning," in *Proc. Int. Conf. Artif. Intell.*, 2019, pp. 86–91.

[3] C. N. Yu, P. Mirowski, and T. K. Ho, "A sparse coding approach to household electricity demand forecasting in smart grids," *IEEE Trans. Smart Grid*, vol. 8, no. 2, pp. 738–748, Mar. 2016.

[4] Z. A. Khan and D. Jayaweera, "Approach for forecasting smart customer demand with significant energy demand variability," in *Proc. 1st Int. Conf. Power, Energy Smart Grid (ICPESG)*, Apr. 2018, pp. 1–5.

[5] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep LSTM-RNN," *Neural Comput. Appl.*, vol. 31, no. 7, pp. 2727–2740, Oct. 2019, doi: 10.1007/s00521-017-3225-z.

[6] D. Syed et al., "Deep learning-based short-term load forecasting approach in smart grid with clustering and consumption pattern recognition," *IEEE Access*, vol. 9, pp. 54992–55008, 2021, doi: 10.1109/ACCESS.2021.3071654.

[7] D. Syed, H. Abu-Rub, A. Ghrayeb, and S. S. Refaat, "Household-level energy forecasting in smart buildings using a novel hybrid deep learning model," *IEEE Access*, vol. 9, pp. 33498–33511, 2021, doi: 10.1109/ACCESS.2021.3061370.

[8] A. Ghasemi, H. Shayeghi, M. Moradzadeh, and M. Nooshyar, "A novel hybrid algorithm for electricity price and load forecasting in smart grids with demand-side management," *Appl. Energy*, vol. 177, pp. 40–59, Sep. 2016.

[9] Y. Yang, W. Hong, and S. Li, "Deep ensemble learning based probabilistic load forecasting in smart grids," *Energy*, vol. 189, Dec. 2019, Art. no. 116324.

[10] Y. Lu, T. Zhang, Z. Zeng, and J. Loo, "An improved RBF neural network for short-term load forecast in smart grids," in *Proc. IEEE Int. Conf. Commun. Syst. (ICCS)*, Dec. 2016, pp. 1–6.

[11] D. Syed, S. S. Refaat, H. Abu-Rub, and O. Bouhali, "Short-term power forecasting model based on dimensionality reduction and deep learning techniques for smart grid," in *Proc. IEEE Kansas Power Energy Conf. (KPEC)*, Jul. 2020, pp. 1–6, doi: 10.1109/KPEC47870.2020.9167560.

[12] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, Jan. 2019, doi: 10.1109/TSG.2017.2753802.

[13] A. Zainab, A. Ghrayeb, D. Syed, H. Abu-Rub, S. S. Refaat, and O. Bouhali, "Big data management in smart grids: Technologies and challenges," *IEEE Access*, vol. 9, pp. 73046–73059, 2021, doi: 10.1109/ACCESS.2021.3080433.

[14] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big data*, vol. 3, no. 1, p. 9, 2016.

[15] A. Hooshmand and R. Sharma, "Energy predictive models with limited data using transfer learning," in *Proc. 10th ACM Int. Conf. Future Energy Syst.*, Jun. 2019, pp. 12–16.

[16] R. Ye and Q. Dai, "A novel transfer learning framework for time series forecasting," *Knowl.-Based Syst.*, vol. 156, pp. 74–99, Sep. 2018.

[17] A. S. Qureshi, A. Khan, A. Zameer, and A. Usman, "Wind power prediction using deep neural network based meta regression and transfer learning," *Appl. Soft Comput.*, vol. 58, pp. 742–755, Sep. 2017.

[18] K. Chapagain, T. Sato, and S. Kittipiyakul, "Performance analysis of short-term electricity demand with meteorological parameters," in *Proc. 14th Int. Conf. Electr. Eng./Electron., Comput., Telecommun. Inf. Technol. (ECTI-CON)*, Jun. 2017, pp. 330–333.

[19] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—A novel pooling deep RNN," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5271–5280, Sep. 2018.

[20] Z. Cao, C. Wan, Z. Zhang, F. Li, and Y. Song, "Hybrid ensemble deep learning for deterministic and probabilistic low-voltage load forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 3, pp. 1881–1897, May 2020.

[21] B. Goehry, Y. Goude, P. Massart, and J.-M. Poggi, "Aggregation of multi-scale experts for bottom-up load forecasting," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 1895–1904, May 2020.

[22] X. Wang, W. J. Lee, H. Huang, R. L. Szabados, D. Y. Wang, and P. V. Olinda, "Factors that impact the accuracy of clustering-based load forecasting," *IEEE Trans. Ind. Appl.*, vol. 52, no. 5, pp. 3625–3630, Sep. 2016, doi: 10.1109/TIA.2016.2558563.

[23] H. Wu, D. Niu, and Z. Song, "Short-term electric load forecasting based on data mining," *Int. J. Eng. Technol.*, vol. 9, no. 3, pp. 250–253, 2017.

[24] A. Ganjavi, E. Christopher, C. M. Johnson, and J. Clare, "A study on probability of distribution loads based on expectation maximization algorithm," in *Proc. IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf. (ISGT)*, Apr. 2017, pp. 1–5.

[25] K. Li, Z. Ma, D. Robinson, and J. Ma, "Identification of typical building daily electricity usage profiles using Gaussian mixture model-based clustering and hierarchical clustering," *Appl. Energy*, vol. 231, pp. 331–342, Dec. 2018.

[26] Z. Han, M. Cheng, F. Chen, Y. Wang, and Z. Deng, "A spatial load forecasting method based on DBSCAN clustering and NAR neural network," *J. Phys., Conf. Ser.*, vol. 1449, no. 1, p. 12032, 2020, doi: 10.1088/1742-6596/1449/1/012032.

[27] Y. Li, Y. Sun, K. Horoshenkov, and S. M. Naqvi, "Domain adaptation and autoencoder-based unsupervised speech enhancement," *IEEE Trans. Artif. Intell.*, vol. 3, no. 1, pp. 43–52, Feb. 2022, doi: 10.1109/TAI.2021.3119927.

[28] M. Ribeiro, K. Grolinger, H. F. ElYamany, W. A. Higashino, and M. A. M. Capretz, "Transfer learning with seasonal and trend adjustment for cross-building energy forecasting," *Energy Buildings*, vol. 165, pp. 352–363, Apr. 2018.

[29] D. Syed, S. S. Refaat, H. Abu-Rub, O. Bouhali, A. Zainab, and L. Xie, "Averaging ensembles model for forecasting of short-term load in smart grids," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Los Angeles, CA, USA, Dec. 2019, pp. 2931–2938, doi: 10.1109/BIGDATA47090.2019.9006183.

[30] J. Lago, F. De Ridder, and B. De Schutter, "Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms," *Appl. Energy*, vol. 221, pp. 386–405, Jul. 2018.

[31] (2015). *UCI Machine Learning Repository: Electricity Load Diagrams 2011–2014 Data Set*. Accessed: Jul. 6, 2021. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/

[32] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade* (Lecture Notes in Computer Science), vol. 7700. Berlin, Germany: Springer, 2012, pp. 9–48, doi: 10.1007/978-3-642-35289-8_3.

[33] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowl.-Based Syst.*, vol. 80, pp. 14–23, May 2015.

[34] M. Rohrbach, S. Ebert, and B. Schiele, "Transfer learning in a transductive setting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 46–54.

[35] Z. Deng, K.-S. Choi, Y. Jiang, and S. Wang, "Generalized hidden-mapping ridge regression, knowledge-leveraged inductive transfer learning for neural networks, fuzzy systems and kernel methods," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2585–2599, Dec. 2014.

[36] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn.*, Jun. 2007, pp. 759–766.

[37] S. Sharma, "Activation functions in neural networks," *Towards Data Sci.*, vol. 6, no. 12, pp. 310–316, 2017.

[38] L. N. Smith, "Cyclical learning rates for training neural networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 464–472.

[39] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[40] E. Paparoditis and T. Sapatinas, "Short-term load forecasting: The similar shape functional time-series predictor," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 3818–3825, Nov. 2013, doi: 10.1109/TPWRS.2013.2272326.

[41] Y. Wang, Q. Xia, and C. Kang, "Secondary forecasting based on deviation analysis for short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 26, no. 2, pp. 500–507, May 2011, doi: 10.1109/TPWRS.2010.2052638.

Dabeeruddin Syed (Member, IEEE) is currently pursuing the Ph.D. degree in electrical and computer engineering (ECEN) with Texas A&M University (TAMU), College Station, TX, USA.

His research interests include smart grids (SGs), big data analytics (BDA), and distributed computing.

Ameema Zainab (Member, IEEE) received the B.E. degree in ECE from the University College of Engineering, Osmania University (UCEOU), Hyderabad, India, in 2013, and the M.S. degree in DSE from Hamad Bin Khalifa University (HBKU), Doha, Qatar, in 2018. She is currently pursuing the Ph.D. degree in electrical and computer engineering (ECEN) with Texas A&M University (TAMU), College Station, TX, USA.

Her research interests include domain of source (DS), big data analytics (BDA), and machine learning (ML), in the smart grid (SG).

Shady S. Refaat (Senior Member, IEEE) received the B.A.Sc., M.A.Sc., and Ph.D. degrees in EE from Cairo University, Giza, Egypt, in 2002, 2007, and 2013, respectively.

He has authored in the research areas of electrical machines, power systems reliability, fault-tolerant systems, and SG.

Haitham Abu-Rub (Fellow, IEEE) received the two Ph.D. degrees in electrical engineering and political science from the Gdansk University of Technology, Gdańsk, Poland, in 1995 and 2004, respectively.

He is currently a Professor in electrical engineering program at Texas A&M, Qatar, and also he is a Managing Director of the Smart Grid Center – Extension in Qatar. He attained research and teaching experience in countries of Germany, Palestine, Poland, Qatar, and USA. His research interests include renewable energy, power electronic converters, SG, and electric drives.

Othmane Bouhali (Member, IEEE) is currently a Physics Research Professor with Texas A&M University at Qatar (TAMUQ), Doha, Qatar. His essential research interests include large-scale modeling, medical physics, high-performance computing, and detector technologies for radiation.

Ali Ghrayeb (Fellow, IEEE) received the Ph.D. degree in electrical engineering (EE) from The University of Arizona, Tucson, AZ, USA, in 2000.

He is currently a Professor in electrical and computer engineering (ECEN) with Texas A&M University at Qatar (TAMUQ), Doha, Qatar. His research interests include massive MIMO, wireless communications, physical layer security, and visible light communications.

Mahdi Houchati (Member, IEEE) received the B.Sc. degree in EE from the National Engineering School of Sfax, Sfax, Tunisia, in 2008, and the M.Sc. degree from Qatar University (QU), Doha, Qatar, in 2017.

He is currently a Senior Engineer with Iberdrola Innovation Middle East QSTP, Doha, with 11 years of experience in industry and research institutes.

Santiago Bañales (Member, IEEE) received the M.Sc. degree from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1998, and the dual Engineering degree from the Ecole Centrale Paris, Gif-sur-Yvette, France, and the Universidad Politecnica de Madrid, Madrid, Spain, in 1994.

He is currently the MD of Iberdrola Innovation Middle East, Iberdrola's Digital Utility Innovation Centre, Doha, Qatar. He has a 21-year experience in the energy industry.