Deep4Ener: Energy demand forecasting for unseen consumers with scarce data using a single deep learning model

SPIROS CHADOULOS, IORDANIS KOUTSOPOULOS, and GEORGE C. POLYZOS, Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, Greece

Forecasting the energy demand of individual consumers is a vital component of future smart energy grids since it enables energy-saving mechanisms such as Demand Response, activity scheduling, and prosumer energy markets. However, training a separate model with each consumer's available smart meter data can raise significant cold-start and scalability issues, despite the fact that personalization can be achieved in cases where the respective training sets have adequate data. Namely, making accurate forecasts for new consumers with limited historical data is challenging since a machine learning model requires a significant volume of data to be trained adequately, while scalability becomes an issue when the number of consumers increases. Training a single model on multiple consumers can mitigate these issues, hence we propose a single-model RNN-based deep learning architecture named Deep4Ener, for consumer-level energy demand forecasting, trained on multiple users and capable of making predictions for unseen consumers with scarce historical data that were not included in the training phase. Deep4Ener learns common energy demand characteristics among different consumers, by utilizing a novel architecture for energy profiling, including clustering, and an encoder neural network for feature extraction. Experiments with data from two open datasets show that Deep4Ener achieves high predictive performance both for known and completely new consumers, while outperforming the current state-of-the-art, namely one-model-per-consumer, standalone RNN, and Amazon's DeepAR approaches. Finally, we demonstrate that Deep4Ener shines when combined with Transfer Learning to further improve its forecasting performance on different energy demand consumers with limited data available.

CCS Concepts: • **Computing methodologies → Machine learning**; • **Hardware → Smart grid**.

Additional Key Words and Phrases: Smart grids, Energy consumption forecasting, Deep learning

1 INTRODUCTION

Consumer consumption forecasting plays a vital role in multiple smart grid applications, such as Demand Response (DR) and hour/day-ahead activity scheduling, where Short-Term Load Forecasting (STLF) is utilized [14]. DR initiatives are used to engage energy consumers into energy efficient consumption behavior by adjusting their demand profiles to mitigate imbalances between supply and demand, through various incentive mechanisms, such as dynamic energy tariffs based on consumer-level demand forecasts [7]. Moreover, STLF is crucial for energy market players who need to provide bids for the day-ahead and real-time markets, based on supply and demand forecasts [10].

Training one model per consumer has major disadvantages: A straightforward approach is to train one Machine Learning (ML) model per consumer with historical smart meter data to achieve personalization, which can raise a number of issues. Namely, a model trained on data from a single consumer cannot generalize and conduct predictions for new consumers with different consumption distribution and patterns [25], hence a new model needs to be trained from scratch for each new consumer. Additionally, if a new model is trained for a user with limited historical data, the resulting predictor will have a poor performance, which is known as the cold-start problem. Furthermore, consumer-level demand forecasting includes considerable energy consumption uncertainty, due to the erratic behavior of consumers, especially residential ones, which can lead to low prediction performance when training a separate model per consumer.

As far as scalability is concerned, there are cases where an electricity retailer would like to make consumer-level forecasts for its entire customer base, which can include thousands of consumers, in order to improve the results of a DR program by choosing personalized incentives and DR actions for each consumer. In such cases, training one model for each user would be computationally demanding due to the need for resource-consuming Deep Learning (DL) models, that need individualized hyper-parameter tuning, to tackle energy demand variability and uncertainty [25].

All these issues, raise the need for rethinking the classic energy demand forecasting problem, by transforming it into a consumer-level consumption forecasting problem, where accurate and scalable predictions are needed even for consumers with scarce or non-existent historical data. In this variation of the problem, the goal is not only to achieve accurate forecasts for each consumer in the training set (generalization), but also to make predictions for entirely new consumers with adequate accuracy (representativeness), to mitigate the cold-start problem. If a consumer has adequate historical data, the one-model-per-consumer approach can be applied to achieve higher levels of personalization. However, the available smart meter historical data can be limited in realistic scenarios, while training a separate model for each consumer can lead to problems related to generalization, scalability, and the cold-start problem as discussed earlier.

Advantages of training a single model on multiple consumers: Consumer energy demand forecasting is a problem where personalization is not the first priority since, despite having significant differences, energy consumers share common energy demand characteristics (e.g. most buildings increase their consumption during summer for cooling purposes). Hence, a single DL model trained on multiple consumers can capture shared patterns among them and conduct accurate predictions even for users with erratic consumption characteristics since a broader set of consumer types is utilized for training, thus enhancing the generalization of the DL model, instead of focusing on personalization.

Authors' address: Spiros Chadoulos, spiroscha@aueb.gr; Iordanis Koutsopoulos, jordan@aueb.gr; George C. Polyzos, polyzos@aueb.gr, Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, 76, Patission Str., Athens, Greece, GR-10434.

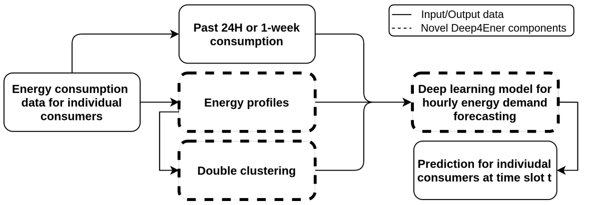
Input/Output data
Novel Deep4Ener components
Past 24H or 1-week consumption
Energy consumption data for individual consumers
Energy profiles
Deep learning model for hourly energy demand forecasting
Double clustering
Prediction for indiviudal consumers at time slot t

Fig. 1. High-level design of the proposed Deep4Ener approach.

Moreover, a single DL model is capable of conducting forecasts for individual consumers with limited historical consumption data, or even for completely new consumers, a scenario where the one-model-per-consumer approach fails. Namely, the single-model approach can utilize past knowledge from multiple consumers and apply it to new ones that have limited historical data to identify their energy demand characteristics and conduct accurate predictions without re-training, a need that is also evident in the literature [3], [25]. This poses a major advantage of the single-model approach since in real use cases, energy consumers might have scarce historical data, e.g. they recently installed smart meters, changed energy providers, or do not want to share their data for privacy reasons. Hence, the main objective of this approach is to train a model that achieves generalization regarding test data from observed consumers (i.e. consumers having data in the training set), while also accomplishing representativeness by making satisfactory predictions even for completely unseen consumers with limited data (i.e. consumers not included in the training set).

The proposed approach: In this paper, we design and validate a single DL model trained on consumption data from multiple consumers, termed Deep4Ener. The proposed approach distinguishes energy demand patterns among different consumers and conducts forecasts for them individually, by constructing their energy profiles. The novel combination of the proposed components in our approach is depicted in Fig. 1. The proposed architecture utilizes a double clustering component specially designed to cluster energy profiles consisting of both time-series and non-time-series features, with the resulting distances from cluster centroids acting as additional inputs for the demand prediction model.

In particular, our architecture features a Recurrent Neural Network (RNN) encoder with Gated Recurrent Unit (GRU) cells that captures the impact of past demand and a Multilayer Perceptron (MLP) that derives the effect that energy profiles have on the next time-slot's energy consumption. This leads to higher predictive performance compared to the state-of-the-art approaches in hourly demand forecasting for individual consumers, even for those not included in the training set, a scenario where the one-model-per-consumer approach completely fails. Additionally, we employ regularization and Transfer Learning (TL) to boost the forecasting performance of Deep4Ener on new consumers with scarce historical data available, by training a base model on a dataset with multiple consumers, and then fine-tuning it on the new target dataset to mitigate the cold-start problem. To sum up, the contributions of this work are the following:

- We introduce Deep4Ener, a consumer-level, single model, energy demand forecasting pipeline, capable of conducting accurate consumption predictions both for known consumers and for new consumers with scarce data not included in the training set.
- We propose a novel Neural Network (NN) architecture that incorporates a GRU RNN encoder and an MLP, for individual consumer energy demand forecasting. The model utilizes energy profiles in the input feature vector, hence enabling it to identify fundamental differences among the consumption patterns of individual consumers.
- We further enhance the model's capability of capturing diverse consumption patterns from individual consumers by introducing a novel double clustering pipeline, designed to group consumers based on both their energy profiles' time-series and non-time-series features.
- We validate the proposed approach using real energy consumption data from two publicly available datasets, with 310 and 368 consumers respectively, and show through experiments that our model outperforms the current state-of-the-art in terms of multiple prediction error metrics, such as R^2, MSE, RMSE, and MAE, while also having the additional advantages of generalization and representativeness.

Compared to our prior work [8], we enrich the proposed approach with regularization and Transfer Learning to improve the model's ability to make forecasts for new consumers originating from different datasets and with diverse characteristics. Furthermore, we enhance the experimental evaluation of the proposed approach by utilizing a second open dataset [29] with consumers from both households and large buildings. A thorough evaluation is also conducted, with more experiments for each dataset separately, as well as cross-dataset experiments to validate the real-world transferability and representativeness of the model. Further to our prior work, we compare our approach with Amazon's state-of-the-art probabilistic time-series forecasting model DeepAR [25]. Our approach makes predictions for new consumers belonging to different categories, with few data available, hence extending the current state-of-the-art and comprising a valuable tool at the hands of energy retailers and operators who can potentially incorporate Deep4Ener in their DR pipelines to target new consumers with limited data.

The rest of the paper is organized as follows: in Section 2, we discuss prior works and highlight the novelty of our work compared to the current state-of-the-art, while in Section 3 we explain the proposed approach in detail. In Section 4, the experiment results are presented and interpreted, while finally in Section 5 this work's contributions are summarized and conclusions are drawn.

2 RELATED WORK

One model per consumer: The most straightforward approach to forecast the energy consumption of a consumer is to train a model for each consumer with the available historical data. Multiple ML models have been studied for this purpose [1], [2], [12], [22], [23], [24], [31], [34], showing promising forecasting capabilities for a single building, with NNs achieving the lowest error in most studies. In [21], TL and specifically few-shot learning is utilized for individualized building energy demand forecasting, by training a baseline neural network on a large dataset and fine-tuning a separate model per building with limited data. However, a model trained to make predictions for a specific building cannot conduct forecasts for new ones. To make things worse, the cold-start problem will emerge in the case of insufficient historical data for other buildings, in case one wants to train new individual models for each of them.

A single model for multiple consumers: An alternative approach is to train a single NN using historical consumption data from multiple consumers. The literature is significantly sparser regarding this approach since the high differentiation of consumer-level energy demand makes the problem more challenging [33]. In [28], Dynamic Time Warping (DTW) is used to cluster 24-hour consumer load curves for 22 days, leading to 20 clusters. Each load curve is encoded as the nearest cluster centroid and Markov models are used for next-day load curve prediction. In [26], a pooling-based RNN is utilized for consumer demand forecasting, which improves the predictive performance by around 7%, while also avoiding overfitting. In [19], TL is utilized to tackle energy demand forecasting for multiple houses. Namely, apartments are clustered based on their daily load profiles, while an individual base RNN is trained on each cluster centroid's profile. Each trained RNN is used as a base model to train an individual model for each apartment, which goes beyond the scenario of one model trained on multiple houses. In [30], few-shot learning is utilized to make energy demand forecasts for buildings with very limited data, i.e. from 12 up to 192 shots/slots. Ensemble clustering is used to cluster buildings from a large dataset and a prototype time-series is generated for each cluster by averaging the time-series of all buildings from that cluster. Then, during few-shot learning, a base LSTM is trained with the prototype series from the cluster that is closer to the target building which is then fine-tuned with the limited target building data, hence going beyond the approach of having one generalized model for multiple building types. The results showed a significant improvement compared to a classic LSTM trained on the limited target data.

In [25], the authors propose a general-purpose time-series probabilistic forecasting model named DeepAR using Autoregressive Recurrent Networks, which can forecast time-series with a single encoder-decoder NN. This approach presumes a time-series probability distribution and learns the mean and standard deviation of consumption at each time slot, while also utilizing a set of time-dependent variables as input. The authors evaluate the model with datasets from multiple domains, including energy consumption forecasting, where it reached a Normalized Root Mean Squared Error (NRMSE) of 1.0 kWh on predicting hourly energy demand for the next day. Probabilistic forecasting is also studied in [33] for household-level load forecasting by extracting load demand scenarios for each house and training a regression model for each scenario, which is then combined with a consumption scenario predictor to obtain the final probabilistic forecast for each household. In [3], the authors utilize an LSTM architecture for household-level electricity time-series forecasting, taking advantage of weather data and the available geo-demographic segmentation data from the "Smart Meters in London" dataset. Namely, the residential smart meters are organized in 19 groups according to the survey-based geo-demographic classification conducted by the dataset creators, and a separate model is trained for each consumer group to make 24-hour predictions based on the past 24-hour consumption and weather data.

Our approach differs from [25] since we propose a lighter NN architecture with an RNN encoder and an MLP that is specifically designed for deterministic and individualized consumer energy demand forecasting, incorporating energy profiles and clustering in the model's inputs. While [25] and [33] predict the energy consumption's probability distributions for a time window, our model focuses on forecasting energy demand on a single time slot, without assuming a probability distribution for the data. This leads to a lighter architecture, which however seems capable of achieving higher performance in terms of single prediction point error, since we report lower NRMSE compared to [25].

Our work also differs from [28], [26], [19], and [3] since we incorporate consumer energy profiles along with distances from cluster centroids as inputs in a single demand forecasting model, enabling it to distinguish consumption patterns and characteristics between different consumers. In addition, compared to [3], we utilize a novel double clustering pipeline to group consumers based on their energy consumption characteristics instead of using geo-demographic attributes that are only available for a limited number of consumers through surveys. Hence, our model contributes towards transferability and replicability since as we show through experiments, it can make accurate energy demand forecasts for new consumers with limited data and it can be even transferred to entirely different datasets without a re-training phase, or with just a fine-tuning procedure on limited data from the target dataset.

3 THE PROPOSED DEEP4ENER APPROACH

3.1 Model

We denote by $\mathcal{C}$ the set of energy consumers available, consisting of C consumers in total, with each consumer $c \in \mathcal{C}$ having a historical energy consumption time-series denoted as $\boldsymbol{\mathcal{P}^c}$ consisting of T energy consumption measurements. $\boldsymbol{\mathcal{P}^c}$ is defined as follows: $\boldsymbol{\mathcal{P}^c} = (P_0^c, \ldots, P_t^c, \ldots, P_T^c)$ with P_t^c denoting the energy consumption measurement in kWh for consumer c during time slot $t \in \{0, \ldots, T\}$.

For each consumer c, $\boldsymbol{\mathcal{P}^c}$ is used to calculate the energy profile $\boldsymbol{E^c} = (\epsilon_1^c, \ldots, \epsilon_{36}^c)$ consisting of 36 energy profile features in total as described in subsection 3.3. These profile features can be adapted according to the use case at hand, e.g. consumer or building metadata can be included. However, in this work we keep the energy profiles as general as possible, so that they can be calculated for any consumer just using historical energy demand measurements, in order to enhance the transferability and replicability of the methodology. In addition, the time-series data need to be transformed using the sliding window method in order to be utilized as an input for the RNN encoder component of the neural network architecture described in detail in subsection 3.5.

We aim to train a single model on energy consumption time-series data from multiple consumers, which will then be capable of conducting consumer-level energy demand forecasts for the next time slots, even for new consumers with limited data. The proposed architecture utilizes energy profiles to distinguish different demand patterns and characteristics between individual consumers. A double clustering process is applied to group consumers with similar energy profiles, hence each profile is encoded according to its distances from all cluster centroids. Additionally, a Deep Neural Network (DNN) architecture that includes an RNN encoder is utilized to help the model capture the impact of past consumption time-series.

3.2 Machine Learning Background

3.2.1 Deep Learning. DNNs are designed to capture complex multi-level data correlations and abstractions for a wide variety of applications, such as speech-recognition, image processing, and time-series forecasting [20]. DNNs include multiple layers which themselves incorporate a number of neurons. Each neuron h_n is comprised from an input feature vector $\boldsymbol{x}$ from the previous layer, a weight vector $\boldsymbol{w_n}$, a bias b_n, and an activation function $f(\cdot)$ as: $h_n = f(\boldsymbol{w_n}^T\boldsymbol{x}+b_n)$. An algorithm known as backpropagation [32] is utilized to learn the weights and biases of a neural network from a set of training data over several epochs. An RNN is a neural network variant which also keeps an internal memory that captures the temporal characteristics of input feature sequences. The most popular RNN cell versions are Long Short-Term Memory networks (LSTMs) and Gated Recurrent Units (GRUs). The two variants have shown similar performance in many problem settings, while GRUs are more efficient since they use fewer parameters [9].

3.2.2 Clustering. Clustering algorithms assign objects into groups according to their characteristics, so that similar objects belong to the same cluster, based on a distance measure. k-means with Euclidean distance metric is a widely-used clustering algorithm, that iteratively assigns objects into the nearest cluster, with a predefined number of clusters k. A variant of k-means , TimeSeriesKMeans, that is utilized to cluster time-series data based on their curve shape, uses Dynamic Time Warping (DTW) distance as a similarity measure, instead of Euclidean distance [5], [28]. A *warping path* on a $n \times n$ matrix is defined as a sequence $p = (p_1, \dots, p_L)$ where $p_l = (i_l, j_l)$ and $l \in [1 : L]$, with $p_1 = (1, 1)$, $p_L = (n, n)$, $i_1 \leq i_2 \leq \dots \leq i_L$, and $j_1 \leq j_2 \leq \dots \leq j_L$. Furthermore, the cost of a warping path p for two feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as: $c_p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{l=1}^{L}(x_{i_l} - y_{j_l})^2$, and the DTW distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as: $DTW(\boldsymbol{x}, \boldsymbol{y}) = c_{p^*}(\boldsymbol{x}, \boldsymbol{y})$, where $p^* = \arg\min c_p(\boldsymbol{x}, \boldsymbol{y})$ is the warping path with the lowest possible cost, which is found using Dynamic Programming [5], [28]. Hence, the DTW distance measures the similarity between two time-series vectors by "stretching" them appropriately to eliminate time-shifts. Thus, using DTW instead of Euclidean distance improves the clustering performance of TimeSeriesKMeans compared to standard k-means regarding time-series data.

3.3 Consumer Energy Profiles

An energy profile is a vector containing characteristics and statistics derived from available consumer energy demand data, which can be classified as either time-series or non-time-series features. The time-series segment of a profile includes ordered statistics regarding specific time periods, e.g. average hourly energy demand. The remaining non-time-series segment of features in a profile can include any other statistics calculated from the consumer's historical consumption data, as proposed by [4], [15], and [16]. Namely, consumption aggregates for different periods of the day, consumption ratios, calculated as the ratio of two consumption figures, and statistical features.

The energy profiles utilized also include a set of features proposed by [13], namely relative average consumption in each period of the day, mean relative standard deviation, and weekend vs. weekday score. For any consumer c, $\bar{P}^c_j$ is defined as the mean consumption and σ^c_j as the standard deviation for each time period $j \in [1, 2, 3, 4]$ (each day is divided into four periods). Additionally, mean consumption for weekends and weekdays is defined as PWE^c_j and PWD^c_j respectively. The detailed energy profile feature table is presented and discussed in Appendix A.

3.4 Consumer Double Clustering

Clustering is often utilized to group consumers with similar energy consumption characteristics, while in most cases algorithms such as k-means with Euclidean distance are used. However, this approach might not be ideal for inputs that include time-series. Namely, k-means with Euclidean distance is susceptible to minor time shifts since it measures point-to-point distance, thus two 24-hour load curves with similar shape will be far in terms of Euclidean distance if one is shifted by just an hour [28].

Hence, in our approach we use k-means with DTW as a distance measure to cluster consumers based only on the time-series segment of their energy profiles. The non-time-series segment is utilized as an input for a second k-means model with Euclidean distance, resulting to a double cluster membership for each consumer. Namely, k^* and k clusters for the non-time-series and time-series clustering procedures respectively. The cluster centroid distances are utilized as additional input features for the consumption forecasting NN. Experiments conducted with real data showed an improvement with this double clustering approach in terms of both cluster quality and predictive performance of the demand forecasting model when the cluster centroid distances are included in the input feature vector. Experiments that demonstrate why the proposed double clustering approach is superior compared to regular clustering for this use case can be found in Appendix B.

3.5 The Proposed Neural Network Approach

The energy profiles calculated from the available historical data and the distances from all cluster centroids regarding each consumer c are combined into an input feature vector to train a model that outputs the consumer's demand P^c_t for time slot t, while 1-hour time slots are used. Additionally, this input vector includes the following time-related features: Hour (0-23), Weekday (0-6), DayOfYear (1-365), and Month (1-12), while it can also contain other available consumer metadata such as building size in m^2, and solar panel integration, as well as past energy demand. The main rationale is that by combining the aforementioned input features, we enable the NN to learn differences between individual consumer types, hence distinguishing certain attributes through the hidden layers, which can lead to more accurate individual consumer forecasts.

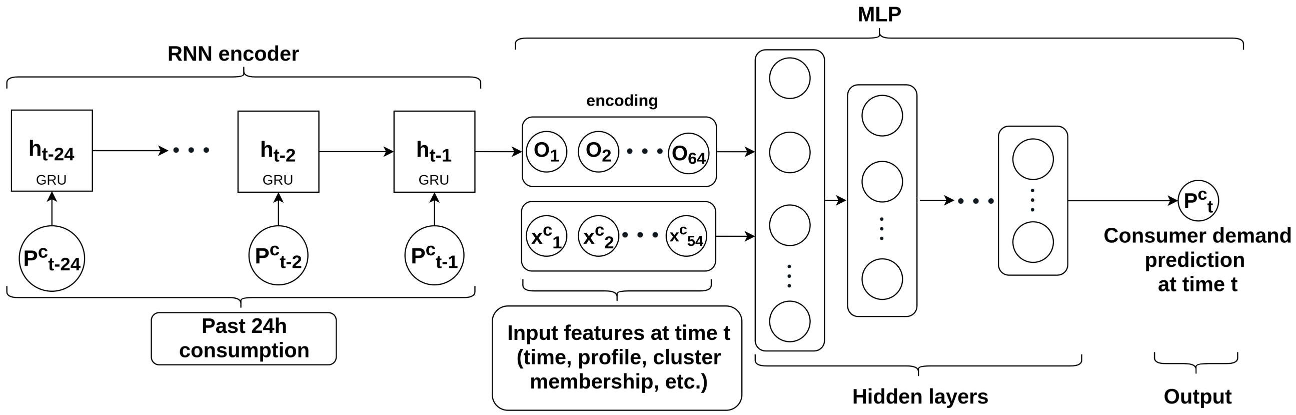
MLP
RNN encoder
encoding
h_{t-24} GRU
h_{t-2} GRU
h_{t-1} GRU
O_1 O_2 O_{64}
x^c_1 x^c_2 x^c_{54}
P^c_{t-24} P^c_{t-2} P^c_{t-1}
P^c_t
Consumer demand prediction at time t
Past 24h consumption
Input features at time t (time, profile, cluster membership, etc.)
Hidden layers
Output

Fig. 2. The proposed Deep4Ener neural network architecture.

The energy demand of previous time-slots improves the model's prediction of energy consumption P_t^c for the next time slot t. However, if past consumption is directly used as part of the NN's input feature vector, the time dimension of the time-series sequence will be ignored. Namely, the time-series features will be treated as independent and not as a sequence of measurements. On the flip side, if an RNN is trained only on the past energy measurements, the advantages of using all the remaining features presented earlier (e.g. energy profiles) will be missed.

For those reasons, we propose a novel architecture depicted in Fig. 2, consisting of an RNN encoder that uses the past consumption time-series as input, and an MLP on top that also includes the rest of the calculated features. Additionally, the MLP incorporates as an input the encoding vector output of the RNN encoder's last cell. The rationale of our approach is that it combines the advantages of a GRU RNN trained on energy time-series data and an MLP trained on crucial features for each consumer, such as energy profiles and cluster distances. Namely, when a prediction for time slot t takes place, the model also incorporates a memory-aware encoding that represents the past energy demand time-series.

The input vector regarding an individual consumer c at time slot t with a 24-hour look-back is: $\boldsymbol{I_t^c} = (P_{t-24}^c, \ldots, P_{t-1}^c, x_1^c, \ldots, x_m^c)$, where $(P_{t-24}^c, \ldots, P_{t-1}^c)$ is the past 24-h energy consumption for consumer c, and $(x_1^c, \ldots, x_m^c)$ includes the energy profile features along with the following features (m features in total): a) Hour, Weekday, Month, DayOfYear; b) Profile distances from non-time-series cluster centroids (k^* distances for k^* clusters); c) Profile distances from time-series cluster centroids (k distances for k clusters); d) `pv` (Boolean feature for photovoltaics - if available - if not, this feature is omitted); e) `total_square_footage` (area in m^2 - if available - if not, this feature is omitted). The `pv` and `total_square_footage` parameters are used only if they are available from consumer data. The past consumption part of $\boldsymbol{I_t^c}$, i.e. $(P_{t-24}^c, \ldots, P_{t-1}^c)$, is used as an input for an RNN encoder with GRU cells [9]. We use a GRU RNN since it is computationally more efficient than LSTMs, while it preserves similar performance in our experiments. More details on the exact neural network architecture utilized in our experiments are presented in Appendix C.

4 DATA EXPERIMENTS

In this section, a thorough set of experiments with real data is carried out in order to evaluate the proposed approach and compare it to the state-of-the-art in terms of predictive performance with different datasets. The experiments' setup and data preprocessing pipelines are presented along with the evaluation metrics used. Furthermore, the results and main takeaways from the experiments are discussed regarding multiple factors, such as model forecasting error, regularization, and transferability.

In subsection 4.3.1, different variations of the proposed architecture are compared with state-of-the-art machine learning models using the Pecan Street dataset, while in subsection 4.3.2, experiments with the UCI-Elergone dataset are conducted to study the performance of the proposed approach when transferred to a different dataset and with the addition of regularization. Additionally, in subsection 4.3.3 the proposed approach is compared against Amazon's DeepAR model and in subsection 4.3.4, its cross-dataset transferability is validated. Finally, in subsection 4.3.5 experiments using TL are conducted to further enhance the model's performance.

4.1 Datasets

The first dataset we employ comes from Pecan Street Dataport [27], and consists of smart meter energy demand data from U.S. households, with 310 consumers being used for training with energy consumption measurements from 2018 to 2019. Additionally, the house area and a Boolean feature about solar panel existence are used for some of the experiments, while $k^* = 5$ and $k = 6$ for the non-time-series and time-series clustering procedures respectively after running the elbow method with the Pecan Street consumers.

We also utilize a second dataset from Elergone and UCI [29] containing electricity demand data for both residential and industrial consumers. The same dataset was used in Amazon's DeepAR paper [25]. Energy consumption measurements from 2014 are utilized for training, with $k^* = 4$ and $k = 7$ for the non-time-series and time-series clustering procedures respectively after running the elbow method with the Elergone consumers. Detailed statistics and characteristics for the datasets are presented in Appendix D.

4.2 Data Preprocessing and Experimental Setup

NNs are designed to minimize an error metric on the training set on average, thus energy peaks are underestimated since they appear with a lower frequency compared to low consumption measurements. For this reason, we apply a Box-Cox transformation [6] on the demand data prior to the training phase, in order to transform them into a normal distribution. The Box-Cox transformation is defined as follows:

$$Y_i = \begin{cases} \frac{Y_i^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0 \\ \log Y_i, & \text{if } \lambda = 0 \end{cases} \tag{1}$$

where Y_i refers to the target variables, which in our case are the energy consumption measurement data, and λ is a parameter selected in order to approximate a normal distribution curve. Furthermore, all input and output features are normalized into $[0, 1]$ using a MinMaxScaler from the Scikit-Learn Python library[1]. Both transformations are inversed after a prediction is made by the model, so that the system outputs an energy consumption forecast value in kWh.

A 80-20 training-test split is applied for the DL models with both datasets. The Pecan Street dataset consisting of 310 households with measurements from 2018 to 2019 is randomly split into training and test sets. This means that the DL models presented are trained on all of the 310 houses, but only with 80% of the measurements. In addition, a set of 100 different unseen consumers is used to test the model's predictive performance on new consumers, while the same is the case for the Elergone-UCI dataset. The loss function used for n data points is Mean Squared Error (MSE): $MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$, where Y_i refer to real/target values and $\hat{Y}_i$ are the model predictions. The optimizer used for training is Adam [17], along with early stopping based on a validation set consisting of 10% of the training set. The experiments were conducted using an NVIDIA RTX 3080 10GB GPU.

4.2.1 *Performance metrics.*

The metrics used for the energy consumption prediction model evaluation are the R^2, MSE (the loss function used for training), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Normalized Root Mean Squared Error (NRMSE) and Normalized Deviation (ND). The R^2 metric measures the variance that the trained model explains and is defined as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \frac{1}{n}\sum_{i=1}^{n}Y_i)^2}. \tag{2}$$

Namely, R^2 measures the closeness of the predicted regression values to the real measurements. The RMSE metric is: $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$. RMSE is a helpful estimator to measure the standard deviation of the model's prediction errors, which is particularly important in case of building energy demand forecasting since different error standard deviations are observed during different hour slots. Another error metric we use is MAE: $MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$. We also employ two normalized error metrics, NRMSE and ND, since they are utilized in Amazon's DeepAR paper [25], while they are also useful to compare the performance of models on different datasets.

Table 1. Experiment Results for Different Variants of the Proposed Approach with the Pecan Street Dataset

	MLP per consumer*	**GRU (7-day look-back)****	**Deep4-Ener (24-hour look-back)**	**Deep4Ener (7-day look-back)**	**Deep4Ener (7-day look-back)**
# of models	310 models for 310 houses	1 model for 310 houses	1 model for 310 houses	1 model for 310 houses	1 model for 100 unseen houses
R^2	66.0%	74.9%	76.3%	76.4%	68.4%
MSE	0.29	0.29	0.27	0.27	0.50
RMSE	0.54	0.54	0.52	0.52	0.71
MAE	0.31	0.30	0.29	0.29	0.37

*With 4 hidden layers (500, 100, 50, 10 neurons respectively). **With 1 GRU layer with 64 hidden units.

NRMSE and ND are defined as follows: $NRMSE = \frac{RMSE}{\frac{1}{n}\sum_{i=1}^{n}Y_i}$, and $ND = \frac{\sum_{i=1}^{n}|Y_i - \hat{Y}_i|}{\sum_{i=1}^{n}Y_i}$, with ND (also known as weighted Mean Absolute Percentage Error - wMAPE) being used instead of Mean Absolute Percentage Error (MAPE) since it is more appropriate for experiments with different datasets that have different energy consumption magnitudes.

The evaluation metrics are calculated after reversing the transformations performed on the data, i.e. Box-Cox and normalization, and are used to compare the following approaches:

- Our proposed Deep4Ener approach with a 24-hour look-back, as depicted in Fig. 1 and Fig. 2.
- Our proposed Deep4Ener approach with a 168-hour (1-week) look-back.
- A regularized version of Deep4Ener with a dropout of 0.5 applied on every layer of the MLP component, except for its input layer $\boldsymbol{I_{MLP}}$, where a dropout of 0.7 is applied.
- A GRU RNN with 1-week look-back trained on multiple consumers, without using energy profiles and cluster distances.
- One MLP per consumer, having the same architecture with the MLP part of the model described in Section 3.5 using the past day's consumption directly as an input.
- Amazon's DeepAR model [25].

4.3 Experiments and Discussion

4.3.1 *Experiments with the Pecan Street dataset.*

Table 1 depicts experiment results for different variations of Deep4Ener compared with other state-of-the-art DL approaches for hourly demand prediction using the Pecan Street dataset. In the first results column, the evaluation metrics for a group of individual models trained separately for each consumer are presented. A 80-20 training-test split is conducted for each of the consumer individual datasets. The significant values of the Standard Deviation (SD) of the error metrics observed for the 310 trained models shows that the limited amount of historical data for many of the consumers negatively affected the respective models. Namely, the SD of the MSE, RMSE, and MAE was 0.22, 0.20, and 0.15 respectively.

[1] https://scikit-learn.org

Additionally, the resulting metrics show that the inherent uncertainty and in some cases arbitrary nature of residential consumer behavior substantially limit the forecasting ability of some models, leading to poor prediction performance for many consumers. This occurred since the available data volume for many households was inadequate to train an ML model (e.g. a few days or weeks), hence leading to the cold-start problem and enforcing one of the main motivations of the proposed Deep4Ener approach. The three middle columns of Table 1 present three models trained on multiple consumers (80% of the combined dataset) and evaluated on the left-out test set. In the second column, the single GRU RNN approach with a 168-hour window achieved slight gains in terms of forecasting error compared to training an individual model per consumer.

The last three columns of Table 1 present the results for two variants of Deep4Ener as described in Section 3.5. The Deep4Ener architecture (Fig. 2) with a 24-hour look-back made forecasts with lower errors and higher R^2 than the GRU RNN model, while also having the additional advantage of using a 24-hour window instead of 168-hour one. Namely, **Deep4Ener achieved a 6.9% reduction of MSE compared to the simple GRU RNN approach**, showing that the approach of utilizing energy profiling and clustering coupled with the neural network depicted in Fig. 2 brings significant value when forecasting the energy demand of individual consumers.

Hence, in the second to last column of Table 1 we depict a Deep4Ener architecture that integrates characteristics from all the previously described models, using an RNN encoder with 1-week look-back window. The results show that this Deep4Ener variant does not achieve any significant performance gains compared to the 24-hour window version of Deep4Ener, hence it might not be worth to extend its look-back window in similar cases. Thus, it is evident that the proposed Deep4Ener approach achieves generalization since it outperforms the approach of training a separate model per consumer, while it also scored lower error metrics compared to a classic GRU RNN model, when tested on a held-out test set.

In the last column of Table 1, we evaluated the trained Deep4Ener model on a new set of consumers that the model had never seen before (100 houses for 2017), to determine if representativeness can be achieved. Despite worse predictive performance when evaluated on unseen consumers, the error metrics are still acceptable, especially considering that training an accurate new model for unseen consumers with inadequate data is not feasible. Thus, Deep4Ener achieves representativeness while also tackling the cold-start problem.

4.3.2 *Experiments with the Elergone-UCI dataset.*

Table 2 presents an evaluation of Deep4Ener on the second dataset utilized for this work, as described in Section 4.1. Similarly to Table 1, the first column of Table 2 shows a set of 368 models trained for 368 consumers separately, with a 80-20 training-test split for each consumer's dataset. The respective evaluation metric values and their standard deviations across the 368 models highlight the emergence of the cold-start problem, where consumers with scarce historical data lead to high forecasting errors for the respective models. For instance, the Standard Deviations of RMSE and MAE were more than their means, i.e. 354 and 328 respectively.

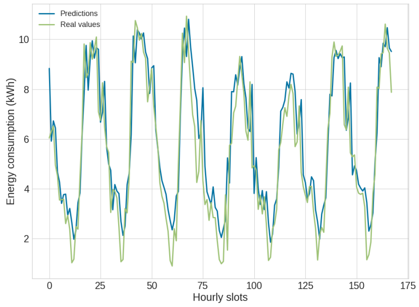

Fig. 3. Deep4Ener trained with UCI data and tested on Pecan Street data.

In the second column of Table 2, Deep4Ener is trained on the UCI dataset consisting of 368 consumers and is compared against the one-model-per-consumer approach (first column of Table 2). Again, a 80-20 train-test split is used on a combined dataset with measurements from all the consumers. We observe that Deep4Ener significantly outperforms the one-model-per-consumer approach for all the evaluation metrics. Namely, **the proposed Deep4Ener approach manages to lower the MSE by 63.5 %, the RMSE by 39.6 % and the MAE by 12 % compared to the one-model-per-consumer approach**, while also achieving representativeness and predictions for new unseen consumers.

In the third column of Table 2, a regularized version of Deep4Ener is trained on the same UCI data. Dropout is utilized as a regularization measure since it constitutes one of the most popular neural network regularization techniques. Other regularization approaches, such as L1 or L2 regularization, can also be adopted. However, the goal of the experiment is to validate the effect that regularization has on the model's transferability and not to identify the most suitable regularization measure for the specific use case and dataset. As expected, the error metrics suffer from a an increase when regularization is applied and validated on the UCI test set. However, regularizing Deep4Ener leads to higher levels of representativeness when we validated it with new consumers from the Pecan Street dataset.

In the last two columns of Table 2, the proposed Deep4Ener and regularized Deep4Ener models trained on the UCI data are tested on 100 houses from the Pecan Street dataset (the same houses as in the last column of Table 1). The error metrics in these last two columns have different magnitudes compared to the rest of the table columns since they are calculated on a different dataset. The regularized version of Deep4Ener (trained with the UCI data) is also presented in Fig. 3 while making forecasts for 1 week of Pecan data. The results show that Deep4Ener can conduct accurate hourly energy consumption forecasts for new consumers from an entirely different dataset with diverse consumer characteristics and patterns. Fig. 3 shows that the model struggles to capture the "ramp downs" which makes sense since in this particular example, the ramp downs are severe and the model cannot react quickly enough, given the fact that it was trained in an entirely different dataset of buildings.

Table 2. Validation of Deep4Ener Trained with the Elergone-UCI Dataset

	MLP per consumer*	**Deep4Ener**	**Regularized Deep4Ener**	**Deep4Ener trained with UCI data and tested on Pecan houses**	**Regularized Deep4Ener trained with UCI data and tested on Pecan houses**
Number of models	368 models for 368 consumers	1 model for 368 consumers	1 model for 368 consumers	1 model trained on 368 UCI consumers and tested on 100 Pecan consumers	1 model trained on 368 UCI consumers and tested on 100 Pecan consumers
R^2	88.0%	99.3%	97.9%	50.2%	62.5%
MSE	10275.3	3749.9	11305.9	0.80	0.61
RMSE	101.4	61.2	106.3	0.89	0.78
MAE	21.7	19.1	28.1	0.62	0.53

*With 4 hidden layers (500, 100, 50, 10 neurons respectively).

Table 3. Comparison Between the Proposed Approach and DeepAR

	DeepAR [25]	**Deep4Ener**
NRMSE	1.00	0.18
ND	0.07	0.06

Table 4. Transferability of Deep4Ener with NRMSE

	Trained on Pecan*	**Trained on UCI**
Tested on Pecan	0.47	0.63
Tested on UCI	2.4	0.30

* Without pv and total_square_footage in the input feature vector.

Furthermore, adding regularization to Deep4Ener led to higher predictive performance when tested on unseen consumers from Pecan Street, further enforcing the motivation for utilizing regularization when representativeness on new consumers is desired.

4.3.3 Comparison with DeepAR. In Table 3, we compare Deep4Ener against Amazon's DeepAR [25] using the UCI dataset with the same set of consumers described in the previous section. Deep4Ener outperforms DeepAR in terms of NRMSE and ND, which are the two metrics used in [25] for this dataset. This improvement in prediction performance probably occurs due the fact that our model is specially designed for deterministic energy demand forecasting, incorporating energy profiles and double clustering, while DeepAR is designed to tackle general-purpose probabilistic time-series forecasting. In the previous sections, Deep4Ener was compared to other popular neural network forecasting models, i.e. MLP per consumer and RNN-GRU approaches, nevertheless a direct comparison with other specific models from the literature is difficult, due to the lack of code, and/or model parameter definition in order to reproduce the results on different datasets.

4.3.4 Cross-dataset transferability. In Table 4, we further examine the cross-dataset transferability of regularized Deep4Ener using NRMSE. The regularized version of Deep4Ener is trained on the two datasets separately to produce two individual models, which are then tested on the held-out test sets of both datasets respectively. It is evident from the primary diagonal of the table that generalization is achieved when the model is tested on a held-out test set with data consisting of consumers that are also included in its training set. From the secondary diagonal, it is evident that the model also achieves representativeness when trained on the UCI dataset and tested on Pecan consumers, while this is not the case for the opposite setup. This is expected since the Pecan-trained model has seen values up to a certain maximum consumption for houses, while the UCI dataset includes much higher values. Hence, the Pecan-trained normalizer will transform most of the UCI measurements to 1.

Table 5. Transfer Learning (TL) Evaluation on Pecan Data

	Deep4Ener trained with UCI data (No TL)	**Deep4Ener base model trained with UCI and fine-tuned on 3-month data from 100 Pecan consumers**	**Deep4Ener base model trained with UCI and fine-tuned on 12-month data from 100 Pecan consumers**
NRMSE	0.63	0.57 (-9.5%)	0.55 (-12.6%)

4.3.5 Transfer Learning experiments. Finally, in Table 5 we employ TL to further improve the transferability and representativeness of regularized Deep4Ener by fully leveraging the available energy consumption data from buildings with limited historical measurements. Consequently, the improved transferability of Deep4Ener with TL can improve its forecasting performance on the target buildings. TL is utilized in the literature to use knowledge acquired from training a model on one problem in the solution of a similar problem. In our case, a base model is first trained on 368 consumers from the UCI dataset and all weights of the model are saved. Then the same NN architecture is initialized with the saved weights from the base model and the training procedure continues on a set of 100 Pecan consumers, a methodology called fine-tuning. More about TL background and motivation can be found in Appendix E.

It is evident from Table 5 that when the base model is finedtuned with 3-month data from Pecan Street, a 9.5% reduction of NRMSE is achieved on the Pecan Street test set compared to not using TL at all. The results in the third column show that when 12-month data are used for fine-tuning the NRMSE is further decreased.

Hence, employing TL to fine-tune Deep4Ener on the target set of consumers can further improve its predictive performance, even if few data are available for them, in which case a new model trained from scratch would lead to poor forecasts, which is known as the cold-start problem.

5 CONCLUSION

In this paper we tackle the problem of consumer-level energy demand forecasting with limited data, utilizing a single novel deep learning approach we named Deep4Ener. The proposed approach leverages an RNN encoder and an MLP paired with energy profiles and double clustering to discover different patterns among electricity consumers and provide accurate predictions even for completely new ones with scarce data available, hence contributing to the state-of-the-art. Experiment results with real data from two datasets show that the proposed approach outperforms the current state-of-the-art in terms of all the prediction error metrics used. The proposed Deep4Ener approach also makes accurate forecasts for consumers from an entirely different dataset with diverse demand characteristics, with the experiment results being further improved when Transfer Learning is employed.

Possible future research directions can include multiple approaches, e.g. the integration of Deep4Ener in a DR system to demonstrate and study its impact on the resulted DR actions in terms of energy savings compared to other prediction mechanisms. It would be interesting to study the real-world impact of such a forecasting methodology, in terms of kWh and carbon emission reduction achieved through a DR program.

ACKNOWLEDGEMENTS

This work was supported by project InterConnect (Interoperable Solutions Connecting Smart Homes, Buildings and Grids), which has received funding from European Union's Horizon 2020 Research and Innovation Programme under Grant 857237.

REFERENCES

[1] Muhammad Waseem Ahmad, Monjur Mourshed, and Yacine Rezgui. 2017. Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. *Energy and Buildings* 147 (2017), 77–89.

[2] Abdulaziz Almalaq and Jun Jason Zhang. 2018. Evolutionary deep learning-based energy consumption prediction for buildings. *IEEE Access* 7 (2018), 1520–1531.

[3] Andrés M Alonso, Francisco J Nogales, and Carlos Ruiz. 2020. A single scalable LSTM model for short-term forecasting of massive electricity time series. *Energies* 13, 20 (2020), 5328.

[4] Christian Beckel, Leyna Sadamori, and Silvia Santini. 2013. Automatic socio-economic classification of households using electricity consumption data. In *Proceedings of the fourth international conference on Future energy systems*. ACM, California, USA, 75–86.

[5] Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series. In *KDD workshop*, Vol. 10. Seattle, WA, USA, 359–370.

[6] George EP Box and David R Cox. 1964. An analysis of transformations. *Journal of the Royal Statistical Society: Series B (Methodological)* 26, 2 (1964), 211–243.

[7] Spiros Chadoulos, Iordanis Koutsopoulos, and George C Polyzos. 2020. Mobile Apps Meet the Smart Energy Grid: A Survey on Consumer Engagement and Machine Learning Applications. *IEEE Access* 8 (2020), 219632–219655.

[8] Spiros Chadoulos, Iordanis Koutsopoulos, and George C Polyzos. 2021. One model fits all: Individualized household energy demand forecasting with a single deep learning model. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*. 466–474.

[9] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

[10] Peter Cramton. 2017. Electricity market design. *Oxford Review of Economic Policy* 33, 4 (2017), 589–612.

[11] David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* PAMI-1, 2 (1979), 224–227.

[12] Cheng Fan, Fu Xiao, and Shengwei Wang. 2014. Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. *Applied Energy* 127 (2014), 1–10.

[13] Stephen Haben, Colin Singleton, and Peter Grindrod. 2015. Analysis and clustering of residential customers energy behavioral demand using smart meter data. *IEEE Transactions on Smart Grid* 7, 1 (2015), 136–144.

[14] Tao Hong and Shu Fan. 2016. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting* 32, 3 (2016), 914–938.

[15] Konstantin Hopf, Mariya Sodenkamp, Ilya Kozlovkiy, and Thorsten Staake. 2016. Feature extraction and filtering for household classification based on smart electricity meter data. *Computer Science-Research and Development* 31, 3 (2016), 141–148.

[16] Konstantin Hopf, Mariya Sodenkamp, and Thorsten Staake. 2018. Enhancing energy efficiency in the residential sector with smart meter data analytics. *Electronic Markets* 28, 4 (2018), 453–473.

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Richard G Lawson and Peter C Jurs. 1990. New index for clustering tendency and its application to chemical problems. *Journal of chemical information and computer sciences* 30, 1 (1990), 36–41.

[19] Tuong Le, Minh Thanh Vo, Tung Kieu, Eenjun Hwang, Seungmin Rho, and Sung Wook Baik. 2020. Multiple electric energy consumption forecasting using a cluster-based strategy for transfer learning in smart building. *Sensors* 20, 9 (2020), 2668.

[20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[21] Eunjung Lee and Wonjong Rhee. 2021. Individualized short-term electric load forecasting with deep neural network based transfer learning and meta learning. *IEEE Access* 9 (2021), 15413–15425.

[22] Chengdong Li, Zixiang Ding, Dongbin Zhao, Jianqiang Yi, and Guiqing Zhang. 2017. Building energy consumption prediction: An extreme deep learning approach. *Energies* 10, 10 (2017), 1525.

[23] Elena Mocanu, Phuong H Nguyen, Madeleine Gibescu, and Wil L Kling. 2016. Deep learning for estimating building energy consumption. *Sustainable Energy, Grids and Networks* 6 (2016), 91–99.

[24] K Muralitharan, Rathinasamy Sakthivel, and R Vishnuvarthan. 2018. Neural network based optimization approach for energy demand prediction in smart grid. *Neurocomputing* 273 (2018), 199–208.

[25] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[26] Heng Shi, Minghao Xu, and Ran Li. 2017. Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Transactions on Smart Grid* 9, 5 (2017), 5271–5280.

[27] Pecan Street. 2022. *Pecan Street Dataport*. pecanstreet.org. https://www.pecanstreet.org/dataport/

[28] Thanchanok Teeraratkul, Daniel O'Neill, and Sanjay Lall. 2017. Shape-based approach to household electric load curve clustering and prediction. *IEEE Transactions on Smart Grid* 9, 5 (2017), 5196–5206.

[29] Artur Trindade. 2022. *Elergone electricity load dataset from UCI machine learning repository*. Elergone. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[30] Qiyuan Wang, Zhihui Chen, and Chenye Wu. 2021. Clustering Enabled Few-Shot Load Forecasting. In *2021 IEEE Sustainable Power and Energy Conference (iSPEC)*. IEEE, 2417–2424.

[31] Zeyu Wang, Yueren Wang, Ruochen Zeng, Ravi S Srinivasan, and Sherry Ahrentzen. 2018. Random Forest based hourly building energy prediction. *Energy and Buildings* 171 (2018), 11–25.

[32] Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proc. IEEE* 78, 10 (1990), 1550–1560.

[33] Zhong Xia, Hui Ma, Tapan Kumar Saha, and Ruiyuan Zhang. 2021. Consumption Scenario-based Probabilistic Load Forecasting of Single Household. *IEEE Transactions on Smart Grid* (2021).

[34] Junjing Yang, Chao Ning, Chirag Deb, Fan Zhang, David Cheong, Siew Eang Lee, Chandra Sekhar, and Kwok Wai Tham. 2017. k-Shape clustering algorithm for building energy usage patterns analysis and forecasting model accuracy improvement. *Energy and Buildings* 146 (2017), 27–37.

A DETAILED ENERGY PROFILE DEFINITION AND NOTATION

Table 6 presents in detail the energy profile features utilized in the proposed approach, after thorough analysis regarding the importance and impact of each feature using the Pearson correlation coefficient. All the features of Table 6 are the same ones as in our prior work [8], except for the seasonal score feature, which is omitted in this extended version since it requires a full year of data to be calculated. Our analysis showed that the seasonal score feature has a small correlation with the consumed energy, hence we believe that by removing it we can apply our approach to greater number of consumers that have less than a year of historical data available. This slight modification makes it possible to calculate the energy profile of a consumer with just a week of historical data, while obviously more data (if available) will lead to a more accurate representation of the consumer's consumption characteristics.

Table 6. Energy Profile Features Used for consumer c

IDs	Profile feature	Description	Type
1-24	24h load profile	Hourly average normalized consumption (24 features)	ts*
25	$\bar{P}^c$	Mean consumption	non-ts
26	σ_c^2	Consumption variance	non-ts
27	max^c	Maximum consumption	non-ts
28	min^c	Minimum consumption	non-ts
29	min_over_mean	min^c / $\bar{P}^c$	non-ts
30	mean_over_max	$\bar{P}^c$ / max^c	non-ts
31	PR_1 (Relative average consumption 1**)	$\bar{P}_1^c$ / $\bar{P}^c$	non-ts
32	PR_2 (Relative average consumption 2)	$\bar{P}_2^c$ / $\bar{P}^c$	non-ts
33	PR_3 (Relative average consumption 3)	$\bar{P}_3^c$ / $\bar{P}^c$	non-ts
34	PR_4 (Relative average consumption 4)	$\bar{P}_4^c$ / $\bar{P}^c$	non-ts
35	weekend_ weekday_ difference_score	$\frac{1}{4}\sum_{j=1}^{4}\frac{\lvert PWD_j^c - PWE_j^c\rvert}{\bar{P}_j^c}$	non-ts
36	mean_relative_std	$\frac{1}{4}\sum_{j=1}^{4}\frac{\sigma_j^c}{\bar{P}_j^c}$	non-ts

*ts stands for time-series. **Each day is divided in the following periods: overnight (period 1, 22:00-6:00), breakfast (period 2, 6:00-9:00), daytime (period 3, 09:00-15:00), and evening (period 4, 15:00-22:00).

B DOUBLE CLUSTERING EXPERIMENTS

To further evaluate and compare the proposed double clustering pipeline against regular clustering, the Hopkins statistic and the Davies–Bouldin index are utilized. The Hopkins statistic represents the cluster tendency of the dataset, i.e. the probability that the data points were derived from a uniform distribution. For that reason, a null hypothesis H_0 and an alternate hypothesis H_a are used, where H_0 implies that the data points are derived by a uniform distribution, and H_a assumes that they were randomly generated. Let $\mathcal{D}$ be the examined dataset, where m points $(p_1, \ldots, p_m)$ are sampled from it, and m other points $(q_1, \ldots, q_m)$ are derived from a random uniform distribution. Then, the Hopkins statistic [18] is defined as follows:

Table 7. Clustering evaluation on the Pecan data

Metrics	Full profile	Time-series	Non-time-series
H	84.5%	86.8%	83.5%
DB	1.81	1.65	1.07
Clusters	5	6	5

$$H = \frac{\sum_{i=1}^{m} u_i}{\sum_{i=1}^{m} u_i + \sum_{i=1}^{m} w_i}, \tag{3}$$

where u_i is the distance between each random point and the nearest point from $\mathcal{D}$, and w_i is the distance between each point in $(p_1, \ldots, p_m)$ and its nearest neighbor from $\mathcal{D}$. Values of H closer to 1 indicate that the dataset has a high clustering tendency, while values closer to 0 indicate that the dataset is uniformly distributed.

The Davies–Bouldin index (DB index) [11] measures the average similarity of the derived clusters. Values closer to 0 indicate a clear cluster partition, with 0 being the lowest possible value. Let $C = \{C_1, C_2, \ldots, C_k\}$ be a partition of n data points into k clusters and $d(c_i, c_j)$ the distance between the centroids of clusters C_i and C_j (the centroids are defined as c_i and c_j respectively). Also, $d(\cdot, \cdot)$ is the Euclidean distance measure. The DB index is defined as follows:

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{j \neq i}\frac{s_i + s_j}{d(c_i, c_j)} \tag{4}$$

where $i, j \in \{1, \ldots, k\}$ and s_i is the average distance of the points in cluster C_i to their centroid:

$$s_i = \frac{1}{|C_i|}\sum_{x \in C_i} d(x, c_i). \tag{5}$$

An evaluation and comparison of the double clustering pipeline against regular clustering is conducted on the full Pecan Street dataset using the Hopkins statistic (H) and the Davies–Bouldin index (DB), with the optimal number of clusters being selected using the elbow method. As presented in Table 7, using the full energy profile to train k-means with Euclidean distance is compared against splitting the profile with the double clustering procedure. The results show a small improvement in terms of H and a significant improvement regarding DB when using the double clustering pipeline, i.e. splitting the profile into time-series and non-time-series features and applying TimeSeriesKMeans with DTW and k-means with Euclidean distance respectively. Thus, it is evident that the proposed double clustering approach results in a better clustering of buildings compared to directly applying k-means to the energy profiles.

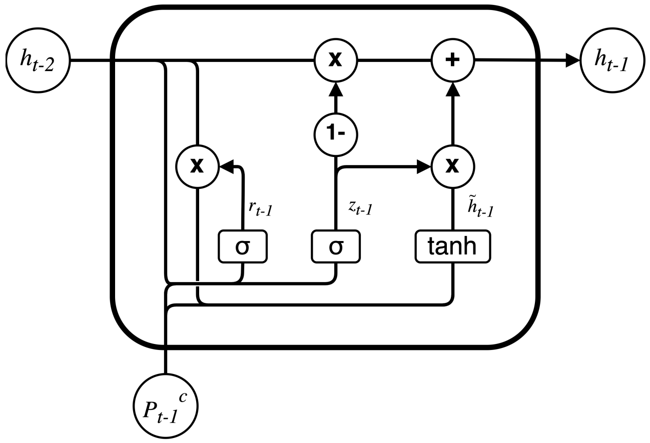

Fig. 4. Gated Recurrent Unit (GRU)

C DEEP4ENER NEURAL NETWORK NOTATION AND DETAILED ARCHITECTURE

The detailed architecture and neural network notation utilized for Deep4Ener are explained in the following paragraphs. An example of the last GRU cell $\boldsymbol{h_{t-1}}$ as depicted in Fig. 4, is presented below:

$$\boldsymbol{z_{t-1}} = \sigma(\boldsymbol{U_z} P^c_{t-1} + \boldsymbol{W_z} \boldsymbol{h_{t-2}} + \boldsymbol{b_z}), \tag{6}$$

$$\boldsymbol{r_{t-1}} = \sigma(\boldsymbol{U_r} P^c_{t-1} + \boldsymbol{W_r} \boldsymbol{h_{t-2}} + \boldsymbol{b_r}), \tag{7}$$

$$\boldsymbol{\tilde{h}_{t-1}} = \tanh(\boldsymbol{U_h} P^c_{t-1} + \boldsymbol{W_h}(\boldsymbol{r_{t-1}} \odot \boldsymbol{h_{t-2}}) + \boldsymbol{b_h}), \tag{8}$$

$$\boldsymbol{h_{t-1}} = \boldsymbol{z_{t-1}} \odot \boldsymbol{\tilde{h}_{t-1}} + (1 - \boldsymbol{z_{t-1}}) \odot \boldsymbol{h_{t-2}}, \tag{9}$$

where $\boldsymbol{z_{t-1}}$ is the update gate vector, $\boldsymbol{r_{t-1}}$ is the reset gate vector, $\boldsymbol{\tilde{h}_{t-1}}$ is the candidate activation vector, $\boldsymbol{h_{t-1}}$ is the output vector, and $\sigma(\cdot)$ refers to the sigmoid activation function $\sigma(x) = \frac{1}{1+e^{-x}}$, while $\boldsymbol{U_z}, \boldsymbol{U_r}, \boldsymbol{U_h}, \boldsymbol{W_z}, \boldsymbol{W_r}, \boldsymbol{W_h}, \boldsymbol{b_z}, \boldsymbol{b_r}$, and $\boldsymbol{b_h}$ are the weight and bias parameters of the NN cell [9].

The RNN encoder output $\boldsymbol{h_{t-1}}$ is an encoding of the past consumption that the NN learned during training. The size of the encoding vector is a hyper-parameter derived from the parameters of the RNN encoder. In our work, we use an encoding with a size equal to 64, after hyper-parameter tuning, which is the number of neurons each GRU cell contains, i.e. $\boldsymbol{h_{t-1}} = (O_1, \ldots, O_{64})$ as depicted in Fig. 2. The encoding along with the rest of the features of $\boldsymbol{I^c_t}$, $\boldsymbol{x^c} = (x^c_1, \ldots, x^c_m)$ in our case, are used as an input feature vector for an MLP with 4 hidden layers. Apart from the input layer $\boldsymbol{I_{MLP}} = (\boldsymbol{h_{t-1}}, \boldsymbol{x^c})$, the MLP consists of a number of hidden layers, and an output neuron which is the energy demand prediction P^c_t for time slot t, regarding the specific consumer. Each hidden layer includes multiple neurons, with the number of hidden layers and neurons being hyper-parameters, and each neuron using the previous layer outputs as an input:

$$\boldsymbol{H_1} = ELU(\boldsymbol{w_1}^T \boldsymbol{I_{MLP}} + \boldsymbol{b_1}), \tag{10}$$

$$\boldsymbol{H_n} = ELU(\boldsymbol{w_n}^T \boldsymbol{H_{n-1}} + \boldsymbol{b_n}),\ n = 2, \ldots, 4, \tag{11}$$

$$P^c_t = \sigma(\boldsymbol{w_5}^T \boldsymbol{H_4} + \boldsymbol{b_5}), \tag{12}$$

Table 8. Dataset Descriptive Statistics

	Pecan Street Energy (kWh)	**Elergone-UCI Energy (kWh)**
mean	1.22	323.98
SD	1.39	772.45
min	0	0
25% percentile*	0.36	19.79
50% percentile*	0.74	98.32
75% percentile*	1.57	268.62
max	19.15	10,163.86
skewness	1.56	5.56

*Percentage of the measurements which are lower than the respective percentile value.

where $ELU(\cdot)$ refers to the Exponential Linear Unit activation function, which is defined as:

$$ELU(x) = \begin{cases} x, & x \geq 0 \\ \alpha(e^x - 1), & x < 0. \end{cases} \tag{13}$$

In our case, it is $\alpha = 1$ and we use *ELU* instead of *ReLU* since it does not face the dying *ReLU* problem and minimizes the cost faster while producing more accurate results. Furthermore, the MLP layers have 500, 100, 50, 10, and 1 neurons respectively after hyper-parameter tuning, and dropout regularization of 0.5 (applied to every layer) is utilized for some of the model variations.

D DATASET STATISTICS

Table 8 presents the statistics of the datasets. It shows that both datasets suffer from heavy positive skewness, which is expected due to the nature of energy consumption and its patterns throughout the day. The statistics of the two datasets show that their consumers are diverse, both in terms of energy demand magnitude and intra-day patterns since the first one only contains houses and the second one also includes buildings.

E TRANSFER LEARNING BACKGROUND AND MOTIVATION

Transfer Learning (TL) is a Machine Learning paradigm that stores knowledge (in the form of neural network weight and bias parameters) acquired during model training for one problem setting and transferring it to a related problem. The saved parameters of the whole network or just a part of it, are utilized as the initial weights for a new training phase with a different dataset on a related problem. In the case of the single-model building energy demand forecasting problem, diverse datasets from different types of buildings exist, e.g. commercial buildings, industrial buildings, offices, households, etc., hence constituting different sub-problems with unique characteristics. Therefore, as presented in section 4.3.5, we utilize TL to fully leverage the available data from datasets that have limited historical measurements, in order to enhance the transferability of the trained model and its performance on the target buildings. Specifically, loading a model that is already trained on a rich dataset and fine-tuning it with a few weeks of data from a scarce target dataset significantly improves its performance in regard to the buildings of the latter.