Short-term Load Forecasting With Clustered Hybrid Models Based On Hour Granularity

Eleftherios Kouloumpris
elefthenk@csd.auth.gr
Aristotle University and Medoid AI
Thessaloniki, Greece

Athina Konstantinou
Aristotle University
Thessaloniki, Greece

Stamatis Karlos
stkarlos@csd.auth.gr
Aristotle University
Thessaloniki, Greece

Grigorios Tsoumakas
greg@csd.auth.gr
Aristotle University and Medoid AI
Thessaloniki, Greece

Ioannis Vlahavas
vlahavas@csd.auth.gr
Aristotle University
Thessaloniki, Greece

ABSTRACT

Although the recent technological achievements have noticeable impact on several aspects of daily life, more and more challenges are raised in practice. As it concerns the Energy field, the need for accurate predictions over time-dependent use cases of large scale remains high. Deep learning approaches have already found great acceptance in energy time-series signals, but there is still much space for improvement. Contributing to the task of short-term load forecasting we compose a hybrid method; first it exploits the statistical profiling of input raw-signals validating them through various complexity metrics; then a series of feature-engineering processes are applied, before fitting a specified recurrent neural network (RNN) architecture. During the first stage, we use time series clustering to separate time periods in order to capture better temporal patterns. We evaluate our approach using a public dataset that regards the total load consumption of Spain, thus supporting our assumptions about the benefits of leveraging hybrid models for short-term load forecasting. The proposed method outperforms other competitors, including a different RNN architecture and some representative Machine Learning regressors.

CCS CONCEPTS

• **Applied computing** → **Forecasting**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Short-term load forecasting, Long short-term memory networks, Load signal decomposition, Time series clustering, Signal complexity metrics

ACM Reference Format:
Eleftherios Kouloumpris, Athina Konstantinou, Stamatis Karlos, Grigorios Tsoumakas, and Ioannis Vlahavas. 2022. Short-term Load Forecasting With Clustered Hybrid Models Based On Hour Granularity. In *12th Hellenic Conference on Artificial Intelligence (SETN 2022), September 7–9, 2022, Corfu, Greece.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3549737.3549783

SETN 2022, September 7–9, 2022, Corfu, Greece

ACM ISBN 978-1-4503-9597-7/22/09…$15.00
https://doi.org/10.1145/3549737.3549783

1 INTRODUCTION

Every kind of crisis drastically affects our society and induces new priorities for serving human needs. At the present time, the impact of the ecological and financial crises in several aspects of our lives is profound, while the advent of the pandemic has already set new balances. The digital revolution is the most important ally towards tackling this new, unstable reality. The widespread rise in environmental awareness has established the Energy market of particular interest to the corresponding committees and national regulators. In addition, continuous advancement in this diversified field requires the co-operation of policy-makers, stakeholders, researchers and data engineers/scientists.

Naturally, the creation of a more sustainable and resource-efficient future constitutes a central problem and has given rise to popular plans such as the European Green Deal [1] and the National Action Plan for Energy Efficiency [2]. The restrictions and the targets that have been imposed recently in international and/or more local scale settle the development of appropriate predictive tools necessary in order to provide detailed analysis and reduce the needs for new energy supplies[31]. Consequentially, these needs have attracted the interest of the Artificial Intelligence (AI) research community to continuously propose innovative methods for forecasting energy load signals, such as the consumption of individual user(s) or building(s).[3, 18]. The instability that exists in the immediate behaviour of individual consumers is still under research, and constitutes critical issue towards the improvement of short-term forecasting [8, 15].

Despite the non-deterministic character of this kind of tasks, any underlying pattern and/or periodicity can be beneficial, thus facilitating the overall forecasting quality. Considering also the large volume of measurements that are now collected by the corresponding sensors, several deep neural network models (DNNs) have conducted important achievements in the recent literature, often outperforming the more conventional time-series methods that depend on purely statistical methods and/or exploit ML models [10, 14]. However, their effectiveness cannot be overlooked,

[1] https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal_en
[2] https://www.coe.int/en/web/human-rights-intergovernmental-cooperation/national-action-plans

since they can still achieve equivalent performance with reduced complexity when they are properly combined, achieving robust performance even when the amount of the collected data is limited [2, 4].

Although a great deal of attention has focused on these two directions, approaches that either directly or indirectly combine both of them have remained almost untouched. These few works that presented hybrid methods try to decompose the raw-input signal, before applying convolutional or recurrent neural networks (CNN, RNN) for obtaining trustworthy forecasts [11, 21]. In this paper, we investigate a novel combination of RNN and traditional time-series methods. We design and evaluate such a hybrid combination for the task of short-term load forecasting, proceeding also to a more dedicated analysis of how external features are better consumed by specified RNNs. The choice of the latter coincides with that of Long-short term memory (LSTM) networks, which to the best of our knowledge has yet to be combined with profiling methods in that task, as stated in [12], but have achieved competitive behavior in load forecasting [17].

Furthermore, we introduce an unsupervised strategy for capturing more consistent time periods, thus building locally trained models. The evaluation of that split is conducted through simple but insightful complexity metrics. This last stage is implemented through k-means clustering based on hour granularity of the input signal, in conjunction with Dynamic Time Warping (DTW) for distance computation [20]. We also examine when it is best to apply a locally trained model through clustering rather against the globally trained model. An automated parameter tuning stage has also been integrated for optimizing each clustered model based on a validation set.

The rest of this work is structured as follows: Section 2 summarizes some recent work regarding the main scientific directions of the proposed approach. In Section 3, the examined data collection is presented along with our methodology. Section 4 reveals the experimental setting that was followed together with the performance of the proposed algorithm against the selected competitors, presenting also a brief discussion per conducted experiment following an ablation study format. Finally, the last Section concludes the main contributions and how are these supported by our results, followed by future steps.

2 RELATED WORK

This Section initially presents the findings of a few recent survey works in order to outline the research trends in the field of energy forecasting and highlights possible gaps in the literature. After recording the most remarkable points, we review some approaches that concern the three main points over which the proposed work contributes: use of LSTM methods in energy forecasting, decomposition methods of energy signals and time series clustering.

2.1 Survey works on energy forecasting

A large part of world's energy resources are consumed by buildings, according to official recordings. This fact has attracted the interest of [9], which presented an in-depth categorization of the energy planning and forecasting methods that tackle such use cases. In contrast with engineering approaches, which do not explicitly exploit historical data during their design, the AI/ML-based approaches are clearly based on them, where the Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs) have been highlighted through that literature review. Based on the conducted investigation, the current survey reports that ANNs can effectively model non-linear relationships, while their generalization ability is reduced when the specifications or the weather conditions of the unknown use cases are drastically different. On the other hand, SVMs are better at tackling over-fitting phenomena, but cannot still perform appropriate scaling when the amount of data is highly increased, while the choice of kernel function remains an important obstacle.

The energy consumed by buildings is the main interest also of the next survey work [30]. The aspiration of that review is the emerging importance of energy estimation for delivering the wider objective of sustainable development. Hence, 91 published works between 2000 and 2019 were mined through a careful selection stage in order to format a representative pool of research demonstrations. As it concerns the main findings of that survey, three different forecasting model types are recorded: black-box, grey-box and ensembles of those two categories. Three out of four works conduct experiments on short-term forecast horizons, while 85% of the included articles are evaluated on datasets that stem from dedicated use cases, instead of exploiting synthetic data or benchmarks (14% and 1% of those works, respectively).

The most recent of the mentioned surveys also follows the discrimination of demand and supply sides in its context [36]. Furthermore, it reaches to a conclusion about the combination of Convolutional neural networks (CNN) along with LSTM networks for capturing better the underlying temporal patterns before trying to fit over the sequential features. However, they indicate the dependency that exists between the applied models and each use case, suggesting evaluating non-hybrid models before resorting to them during production.

2.2 LSTM-based approaches on load forecasting

LSTMs exploit various gate mechanisms in order to let periodic patterns enrich the learning process by accumulating past information and model better the long-term dependencies. Hence, the issues that occur by back-propagation algorithm are effectively handled, overcoming at the same time the need of fixed size input and output signals. Following the same line as in case of DNNs, stacked or convolutional architectures that are based on LSTMs have been designed for tackling time-dependent problems [32]. A Sequence-to-Sequence choice was proved more accurate against typical LSTM for both hour and minute granularity scale [24]. While the adoption of a recurrent inception CNN offered an intelligent manner of preserving the timing characteristics, it also achieved to tune inherently posed parameters [16].

Another research approach is the adoption of ensemble schemes, constructing usually heterogeneous or hybrid models. We discern the next two publications; a combination between the seasonal auto-regressive integrated moving average (SARIMA) algorithm and the standard LSTM network [19], as well as an evolutionary approach that explores a pool of DDNs, LSTMs and gated recurrent unit models (GRUs) – another one RNN-based variant – [3] for

predicting energy consumption on household level. The former is based on the complementary behavior obtained by assessing a linear and a nonlinear model balancing the contribution of each algorithm through a trade-off parameter, while the latter adopts a genetic algorithm for tuning both the number of the kept models and their configurations.

2.3 Signal preprocess on load forecasting

Our work is also relevant to the literature of signal decomposition and time series clustering. Despite the crucial role of the raw-signal decomposition, plenty of the applied methods have been devised from fields like Signal Theory without necessarily caring about the time dependency, the computational efficiency and/or the interpretability of the underlying mechanism. However, vital boost has been recorded regarding load forecasting approaches after careful adoption of such preprocessing stages per investigated case. A family of ordered weighted averaging operators (OWA) was introduced in [7] achieving to extract smoother variants of the input signal that offered smaller aggregated errors during the evaluation. A variant of the popular empirical model decomposition (EMD) method [37], the ensemble EMD, found great acceptance in large scale use cases by choosing different models for fitting the extracted frequency components; the lower through the multi-variable linear regression regressor and the higher ones through LSTMs [21].

Time series clustering has also been used in the literature for obtaining more accurate predictions based on proper grouping of data patterns. Density based clustering was applied to solar energy production data for capturing the different weather profiles [6]. Afterwards, they applied standard LSTM networks and the Facebook Prophet model and achieved great error reduction in the majority of the examined scenarios. Two works that used partitioning-based clustering methods, k-means and k-medoids respectively, in tasks that concern energy signals are [1, 13]. The former employs the dynamic validity index for defining the number of clusters, while the latter is based on the multivariate Root Mean Square Error (RMSE) computed over the initial and the reconstructed signal.

As it regards our methodology, we used the decomposition of the raw-signal in order to facilitate the stage of the clustering, which later was applied for capturing local consistencies. While our approach follows a strategy similar with [11], that work materialized the applied decomposition through a CNN profiling approach, mentioning at the same time the need of adopting LSTMs for similar scenarios. Furthermore, we use DTW to obtain more robust distances and employ two different measures for selecting the number of clusters.

3 METHODOLOGY

3.1 Data sources and preprocessing

This paper focuses on short-term load forecasting in Spain, a country that is ranked 30th by population and 45th by energy consumption per person worldwide based on measurements during 2020[3]. Specifically, we consider an energy dataset with hourly total load observations that concern the total consumption of Spain [4]. In addition, the dataset includes weather information for temperature, pressure, humidity, and wind speed for 5 cities of Spain. Linear interpolation has been used to impute a small number of outliers, while gaussian noise has been added to historical weather data in order to make them resemble weather forecasts. In the following paragraph, we describe additional feature engineering steps.

Table 1: Spain total consumption dataset formulation

Feature	Load	Weather	Time	Type
Total load	✓			Total (1)
Total load 1-week lag	✓			Total (1)
Total load 2-weeks lag	✓			Total (1)
Total load 3-weeks lag	✓			Total (1)
Total load 4-weeks lag	✓			Total (1)
Temperature		✓		City (5)
Pressure		✓		City (5)
Humidity		✓		City (5)
Wind Speed		✓		City (5)
Weekday			✓	Trig. (2)
Month of year			✓	Trig. (2)
Day of year			✓	Trig. (2)
Day of month			✓	Trig. (2)
Weekday or weekend			✓	Boolean (1)

Regarding total load consumption, we create 4 lagged features by taking the total load consumed 1-4 weeks before the current day. Furthermore, we enrich the dataset with a number of calendar features. Specifically, we consider the weekday, month, day of the year, day of the month, and week- days/weekends indicator. Excepting the last feature which is a boolean indicator, the other calendar features have been encoded with a pair of trigonometric functions (sine and cosine) with regards to their cycle. The complete set of features has been normalized in the range [0,1] by dividing with the maximum value for each column, and can be found in Table 1. Five of them are related to electric load, twenty of them refer to the weather, and nine of them represent the time information.

The historical dataset described previously range from 1/1/2015 to 31/12/2018. Since the first 28-days are required to extract the lagged load features, the actual interval for which all features are available is 29/1/2015 to 31/12/2018. As it is common in the literature of time series forecasting, we proceed to splitting the data into train, validation and test periods. Consequently, training set is from 29/1/2015 to 31/12/2016, validation set is from 1/1/2017 to 31/12/2017, and test set is from 1/1/2018 to 31/12/2018.

The aforementioned dataset, along with an implementation of the learning methods and experiments of the upcoming sections, are available in a (currently anonymized) public GitHub repository.[5]

3.2 k-means Clustering with Dynamic Time Warping

The decomposition of an input signal into multiple components to be predicted by different models has been an effective approach in the literature of time series forecasting [35]. In this work, we split

[3] https://www.eia.gov/international/overview/country/ESP

[4] https://www.kaggle.com/nicholasjhana/energy-consumption-generation-prices-and-weather

[5] https://anonymous.4open.science/r/Short-term-Load-Forecasting-with-LSTM-7870

the hourly total load signal into 24 time series, one for each hour of the day. We proceed to the description of a clustering process, the purpose of which is to group the time series such that a different model can be trained for each cluster.

We apply k-means to cluster the hourly time series, an iterative clustering method that requires the experimenter to set the number of clusters via a parameter k. Furthermore, the application of k-means requires the specification of a distance function. Subsequently, we use k-means in combination with the Dynamic Time Warping (DTW) distance [25]. DTW takes into consideration the time aspect of its inputs, which makes it an appropriate distance function for the comparison of time series. Compared to the Euclidean distance, previous works attribute the superiority of DTW to its sequence-alignment flexibility [5]. While DTW adds additional time complexity to k-means, it is only required in the training phase of our system. Our criterion in the selection of parameter k lies in the maximization of the Silhouette score [38] and the minimization of the Davies-Bouldin index (DBI) [27].

In order to define the Silhouette coefficient, it is helpful to introduce some notation. Assuming a dataset $X = \{x_1, ..., x_N\}$, the Silhouette score of a single example x_i is denoted as Silhouette_i and is a function of the the mean intra-cluster distance (a_i) and the mean nearest-cluster distance (b_i) with respect to the same example. Also, x_i belongs to the cluster $C(x_i) \subseteq X$ and $NC(x_i) \subseteq X$ is the cluster that is closest to x_i excluding $C(x_i)$, while C denotes any cluster and $|C|$ the number of examples in C. The Silhouette coefficient is then defined as follows:

$$NC(x_i) = arg_C min \frac{1}{|C|} \sum_{x_j \in C} dist(x_i, x_j), C \neq C(x_i) \tag{1a}$$

$$a_i = \frac{1}{|C(x_i)|} \sum_{x_i \in C(x_i)} dist(x_i, x_j), x_i \neq x_j \tag{1b}$$

$$b_i = \frac{1}{|NC(x_i)|} \sum_{x_i \in NC(x_i)} dist(x_i, x_j) \tag{1c}$$

$$Shillouete_i = \frac{b_i - a_i}{max(a_i, b_i)} \tag{1d}$$

$$Shillouete = \frac{1}{N} \sum_{x_i \in X} Shillouete_i \tag{1e}$$

DBI is defined in Equation 2, in which s_i denotes the mean distance between all time series in the i_{th} cluster and their cluster's centroid, M_{ij} is the distance between the i_{th} cluster and the j_{th} cluster centroids, and k is the total number of clusters.

$$DBI = \frac{1}{k} \sum_{i=1}^{k} max_{i \neq j} \frac{s_i + s_j}{M_{ij}} \tag{2}$$

Moreover, to reduce the complexity of the method, we also consider the temporal continuity of the partitions created by the clustering result. For instance, we consider the clustering result ([0, .., 12], [13, .., 23]) more cohesive compared to a different clustering ([0, .., 6, 18, .., 23], [7, .., 17]) in which clusters are more interleaved. Although this restriction can be quantified, in the case of 24 different hours this heuristic is easily observable without the need of any human expertise.

3.3 Time series decomposition

To improve the results of $k-$means, we investigate the potential of time series decomposition (TSD) in specifying less complicated components to be used as input for clustering. Specifically, we use additive seasonal decomposition based on moving averages in order to extract trend, season and residual components from the original (integrated) signal. The formula for the decomposition of an input signal y_t is given in Equation 3. In the equation, $\hat{T}_t$ denotes the trend component, $\hat{S}_t$ denotes the seasonal component (computed with the detrended signal y_t^D) and $\hat{R}_t$ denotes the remainder component.

$$\hat{T}_t = \frac{1}{m} \sum_{j=-k}^{k} y_{t+j} \tag{3a}$$

$$y_t^D = y_t - \hat{T}_t \tag{3b}$$

$$\hat{S}_t \underset{\text{j} \leftarrow \text{t\%m}}{=} \hat{S}_j = \frac{T}{m} \sum_{i=0}^{T/m} y_{j+im}^D \tag{3c}$$

$$\hat{R}_t = y_t - \hat{T}_t - \hat{S}_t \tag{3d}$$

We proceed accordingly to experiments of integrated, trend, seasonal and remainder clustering in order to detect the best result. In addition to the selection criteria mentioned in the previous section, we also consider the complexity of the produced components with the following measures: Complexity Estimate (*CE*), Mean Absolute Change (*MAC*) and Approximate Entropy (*ApEn*) [28]. Assuming a time series y_t, the complexity metrics are given in Equations 4, 5 and 6, respectively. In addition, *ApEn* requires the definition of two parameters: the length of the window w in which maximum differences are sought and a threshold r above which differences are not considered. The purpose of such complexity metrics is to capture the irregularity or the degree of unforecastability found in time series data.

$$CE(y) = \sqrt{\sum_{t=1}^{N-1} (y_t - y_{t+1})^2} \tag{4}$$

$$MAC(y) = \sum_{t=1}^{N-1} |y_t - y_{t+1}| \tag{5}$$

$$d_{ijmax} = max_{1 \geq k \geq m}(|y_{i+k-1} - y_{j+k-1}|) \tag{6a}$$

$$Ci^m(r) = \frac{1}{N-m+1} \sum_{j=1}^{N-m+1} \mathbb{1}(d_{ijmax} \leq r) \tag{6b}$$

$$\Phi^m(r) = \frac{1}{N-m+1} \sum_{i=1}^{N-m-1} ln(Ci^m(r)) \tag{6c}$$

$$ApEn(m, r) = \Phi^m(r) - \Phi^{m+1}(r) \tag{6d}$$

3.4 Deep learning architectures

In conjunction with the profiling methods mentioned previously, we implement and evaluate two architectures based on the LSTM model, using two years for training and one year for validation. We describe in detail the two architectures, including the process of hyper-parameter exploration.

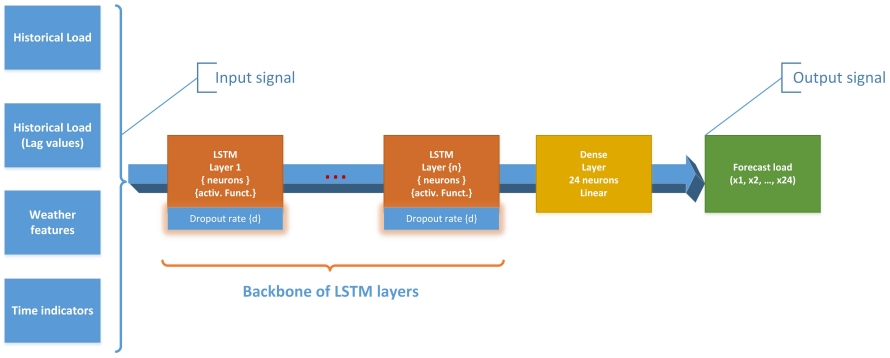

Figure 1: Single-stage-input architecture

In the first architecture, all features are given in the first layer. The architecture begins with n LSTM layers that use the *tanh* activation function, each followed by a dropout layer to reduce over-fitting. The final layer is a fully-connected layer with linear activation function consisting of 24 output neurons, corresponding to the hours of the day. A graphical representation of that architecture is given in Figure 1. We will refer to this architecture as Single-stage-input architecture.

In the second architecture, only historical load data are given in the first layer. The remaining features which describe weekly load lags, weather and time are given after the LSTM layers via a fully connected hidden layer. The output of this layer is concatenated with the output of the last LSTM, and is then passed to the output layer. The output layer is also a fully connected network with 24 outputs. Similarly, with the previous architecture, a graphical representation of the second one is given in Figure 2. We refer to this architecture as Double-stage-input architecture.

Regarding the two architectures, we considered the following hyper-parameters: learning rate, batch size, neurons per LSTM layer, number of LSTM layers, dropout rate, training epochs, optimizer, activation functions and window size. For the second architecture, we also consider the number of neurons and activation function of the hidden fully connected layer. In both cases, our optimisation goal is to improve the MAPE during the validation period. Different combinations of weather features and locations are also investigated during that data pre-processing stage.

As it regards the evaluation stage, we consider a number of forecasting performance metrics; Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Relative Mean Square Error (RRMSE) and R^2, given in Equations 7, 8, 9, 10 and 11, respectively. In addition, we evaluate the percentage of absolute percentage errors that fall within three ranges: [0,10],(10,15] and (15,100].

$$MAE(y, \hat{y}) = \frac{1}{T}\sum_{t=1}^{T}|y_t - \hat{y}_t| \tag{7}$$

$$MAPE(y, \hat{y}) = 100\frac{1}{T}\sum_{t=1}^{T}\frac{|y_t - \hat{y}_t|}{|y_t|} \tag{8}$$

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2} \tag{9}$$

$$RRMSE(y, \hat{y}) = \frac{\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}}{\sum_{t=1}^{T}(y_t)} \tag{10}$$

$$R^2 = 1 - \frac{SSE}{SST} \tag{11a}$$

$$SSE = \sum_{t=1}^{T}(y_t - \hat{y}_t)^2 \tag{11b}$$

$$SST = \sum_{t=1}^{T}(y_t - \bar{y})^2 \tag{11c}$$

3.5 Proposed Model

We propose a hybrid model that combines time series profiling with an appropriate LSTM architecture. The main idea is to perform cluster-based time split forecast which may be based either on the integrated signal or on some signal component extracted with TSD. An algorithm for the general approach is presented in Algorithm 1. In the next paragraph, we describe the simpler approach that is based on the integrated signal.

Using the integrated signal, we first split the total load signal into 24 time series and then apply k-means to group these into clusters and derive k hour groups. For each hour group, we train a dedicated instance of our model architecture to predict only the hours that participate in the same cluster. These steps are present in (Algorithm 1, *TRAINING*), where we assume the parameter *componentSelector* is equal to *integrated*, such that clustering runs on the original input signal.

Then, in the prediction phase (Algorithm 1, *PREDICTION*), the input data split according to the hour groups and the full-day length output of the proposed model is constructed by merging the prediction vectors produced by each time-based cluster.

In the second approach, the difference is that k-means is applied to a component produced by TSD. For instance, it is possible to extract the trend component of the total load signal and perform k-means on the same component to derive a different hour grouping, by passing $componentSelector = trend$ in (Algorithm 1, *TRAINING*).

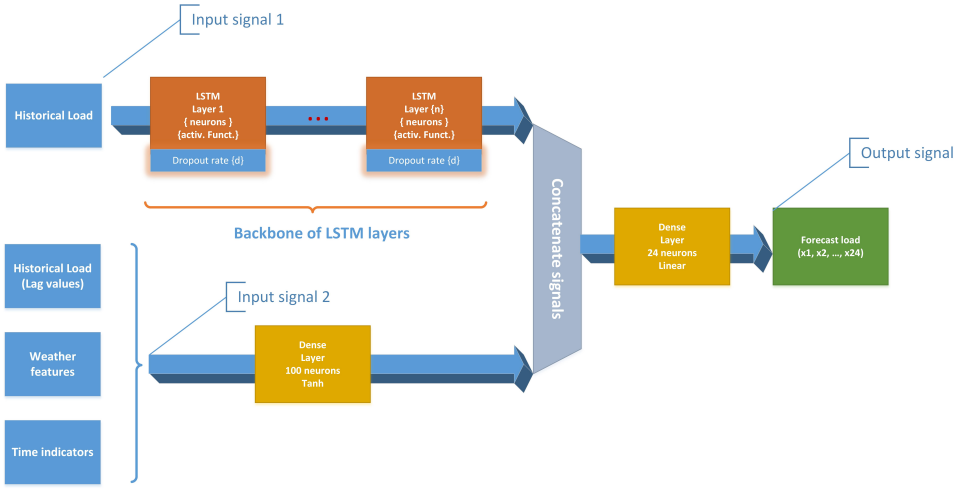

Figure 2: Double-stage-input architecture

Afterwards, the process is the same as described previously; that is the training of a separate instance of our architecture for each hour-based group and the merging of predictions from all cluster models. In the experimental part of this paper, our selection of the best component to use for clustering is solely based on validation set performance.

4 EXPERIMENTAL SETTINGS AND RESULTS

4.1 Time series clustering and decomposition

The current Section presents the results of the profiling procedure, in which we applied k-means clustering and time series decomposition to group the hours of the day.

The k-means algorithm with DTW distance requires the selection of parameter k, which is the number of cluster centers. In order to tune this parameter, we run k-means on the integrated signal for $k \in [2, .., 13]$ and evaluate the results with the Silhouette score and Davies-Bouldin index, as presented in Table 2. The best values for the two evaluation metrics have been highlighted with boldface. While the best performance is achieved by $k = 2$, we choose k-means with $k = 3$ because it split the input signal into intervals with better time continuity.

Table 2: Silhouette score and Davies-Bouldin index for different number of clusters (k)

Clusters	2	3	4	5	6	7
Silhouette	**0.572**	0.437	0.341	0.319	0.331	0.337
DBI	**1.0**	1.674	2.908	3.249	3.378	3.544
Clusters	8	9	10	11	12	13
Silhouette	0.301	0.313	0.315	0.271	0.269	0.266
DBI	3.431	3.248	3.216	3.672	4.591	5.453

In the following step, we apply time series decomposition to derive the following components: trend, season and remainder. We apply k-means on the integrated signal and all the additive components separately. Then, we consider the time continuity that results from the different clusterings. Table 3 presents how the 24 hours were grouped for feeding different signal components to k-means. It is apparent that residual and seasonal clustering produce hours groups of better time continuity compared to the results of integrated and trend clustering.

Thus, hereafter we will consider only residual and seasonal clustering. To choose the better option among the two, we use the *MAC*, *CE* and *ApEn*, introduced in Section 3.3, to evaluate the complexity for each clustering. These results, including also a normalized mean estimation of complexity, are provided in Table 4. Specifically, two out of three complexity metrics regard residual clustering as a better option on average, while it also has the best normalized mean complexity.

Subsequently, in the following stages we proceed with residual clustering due to the fact that it produced low complexity clusters with good time continuity.

4.2 LSTM architecture

As mentioned in Section 3.4, we considered two deep learning architectures. To select among the best architecture, we used a trial-and-error approach to tune each hyper-parameter, while holding the remaining hyper-parameters fixed. The results showed that the Single-stage-input architecture performed better in terms of performance and error distribution. Thus, we proceed with this architecture for the remaining of this work.

Further details regarding the tuning process of our LSTM model, which constitutes the backbone of the proposed architecture, are revealed here. Specifically, we use Bayesian Optimisation to specify and tune the LSTM architecture on the integrated signal with the hyperas library [29]. The search space per hyper-parameter that has been included in that stage is given in Table 5. In addition, the configuration that achieved the optimal performance is depicted into Table 6.

Algorithm 1 Energy forecasting with clustered hybrid models

function CLUSTERING(inputSignal,clusteringParameters)
  hourComponents ← *hourSplit*(*inputSignal*) // creating a dataset with 24 components, one per hour
  hourGroups ← *DTWKmeans*(*hourComponents*, *clusteringParameters*) // partitioning the input dataset into k clusters
  return *hourGroups*
function CLUSTERING_APPLY(inputSignal, hourGroups)
  hourComponents ← *hourSplit*(*inputSignal*)
  clusteredComponents ← *clusterAssign*(*hourComponents*, *hourGroups*)
  return *clusteredComponents*
function DECOMPOSITION(inputSignal, componentSelector)
  trend, *season*, *remainder* ← *decompose*(*inputSignal*) // based on subsection 3.3
  signalComponent ← *componentSelector*(*trend*, *season*, *remainder*)
  return *signalComponent*
function TRAINING(inputSignal, componentSelector, clusteringParameters, lstmParameters)
  if *componentSelector* == *integrated* **then**
    signalComponent ← *inputSignal*
  else
    signalComponent ← *DECOMPOSITION*(*inputSignal*, *componentSelector*)
  hourGroups ← *CLUSTERING*(*signalComponent*, *clusteringParameters*)
  clusteredComponents ← *CLUSTERING_APPLY*(*inputSignal*, *hourGroups*)
  for *cluster in 1, ..., k* **do**
    clusterRNN[*cluster*] ← *trainRNN*(*clusteredComponents*[*cluster*], *lstmParameters*)
  return *clusterRNN*
function PREDICTION(inputSignal, componentSelector, clusterRNN,hourGroups)
  clusteredComponents ← *CLUSTERING_APPLY*(*inputSignal*, *hourGroups*)
  for *cluster in 1, ..., k* **do**
    predictedComponents[*cluster*] ← *clusterRNN*[*cluster*].*predict*(*clusteredComponents*[*cluster*])
  predictions ← *merge*(*predictedComponents*)
  return *predictions*

Table 3: Hour groupings resulting from clustering of different signal components

	Integrated Clustering	**Trend Clustering**
Cluster 1	07:00, 08:00, 15:00, 16:00, 17:00, 18:00, 22:00	09:00, 10:00, 11:00, 12:00, 13:00, 14:00, 15:00, 16:00, 17:00, 18:00, 19:00, 20:00, 21:00, 22:00
Cluster 2	23:00, 00:00, 01:00, 02:00, 03:00, 04:00, 05:00, 06:00	01:00, 02:00, 03:00, 04:00, 05:00, 06:00
Cluster 3	09:00, 10:00, 11:00, 12:00, 13:00, 14:00, 19:00, 20:00, 21:00	23:00, 00:00, 07:00, 08:00
	Seasonal Clustering	**Residual Clustering**
Cluster 1	12:00, 13:00, 14:00, 15:00, 16:00, 17:00, 18:00, 19:00, 20:00	12:00, 13:00, 14:00, 15:00, 16:00, 17:00, 18:00, 19:00, 21:00
Cluster 2	21:00, 22:00, 23:00, 00:00, 01:00, 02:00, 03:00, 04:00, 05:00, 06:00	22:00, 23:00, 00:00, 01:00, 02:00, 03:00, 04:00, 05:00, 06:00
Cluster 3	07:00, 08:00, 09:00, 10:00, 11:00	07:00, 08:00, 09:00, 10:00, 11:00, 20:00

Hereinafter, we will refer to this tuned variant of our selected LSTM architecture as LSTM-HP.

4.3 Cluster vs global model

While the LSTM-HP model is trained on the initial integrated signal that contains all 24 hours, we proceed to the training of three more LSTM models. The first step is to split the 24-hour signal in three hour groups, which are decided by running k-means with DTW distance and $k = 3$ on the residual component of the integrated signal. For each hour group, we train a separate LSTM model which is similar to LSTM-HP in terms of configuration and architecture. The three models for each hour group are denoted as $\text{Model}_{cluster1}$, $\text{Model}_{cluster2}$ and $\text{Model}_{cluster3}$.

Table 4: Complexity metrics for seasonal and residual clustering

Clustering	Seasonal				Residual			
Metric	MAC	CE	ApEn	Norm.	MAC	CE	ApEn	Norm.
Cluster 1	0.0222	4.345	0.957	0.5366	0.0224	3.635	0.976	0.5040
Cluster 2	0.0468	8.363	0.862	0.8013	0.0429	6.003	0.919	0.6991
Cluster 3	0.0523	6.702	0.950	0.7869	0.0478	5.713	1.062	0.7492
Mean	0.0404	6.450	**0.923**	0.7049	**0.0377**	**5.117**	0.985	**0.6507**

Table 5: Hyperas options for LSTM model.

Parameters	Options
LSTM layers	**2**, 3, 4
First LSTM neurons	100, **200**, 300, 400, 600, 800, 1000
Second LSTM neurons	100, 200, 300, **400**, 600
Third LSTM neurons	100, 200 ,300
Fourth LSTM neurons	50, 100 ,150
Epochs	70, 100, 130, **170**, 200
LSTM act. funct.	relu, sigmoid, **linear**, tanh

Table 6: Best hyper-parameters for LSTM model

LSTM layers	2	**LSTM neurons**	200-400
Drop out rate	0.3	**Bach size**	25
Dense layer	1 (24 neurons)	**Epochs**	170
LSTM act. funct.	linear	**kernel init.**	lecun normal

It is possible to derive a 24-hour prediction by merging the predictions of $\text{Model}_{cluster1}$,$\text{Model}_{cluster2}$ and $\text{Model}_{cluster3}$, which all forecast different hours of the day. We denote this combined model as CL-time-LSTM-R. The test set performance evaluation metrics for LSTM-HP, $\text{Model}_{cluster1}$,$\text{Model}_{cluster2}$, $\text{Model}_{cluster3}$ and LSTM-HP are given in Table 7. Overall, we observe that the combined model CL-time-LSTM-R achieves better performance than LSTM-HP in terms of all performance measures.

For the sake of comparison, we also evaluate LSTM-HP separately for each of the three hour-based groups, and denote the results as $\text{LSTM-HP}_{cluster1}$, $\text{LSTM-HP}_{cluster2}$, $\text{LSTM-HP}_{cluster3}$. Interestingly, by comparing these performances with the respective performances of $\text{Model}_{cluster1}$, $\text{Model}_{cluster2}$ and $\text{Model}_{cluster3}$, we observe that the latter models perform better in every hour group. Hence, not only does CL-time-LSTM-R outperform LSTM-HP with regard to the 24-hour set, but is better with regard to any of the three hour-based groups.

Regarding the distribution of errors, we observe that LSTM-HP has a higher percentage in the highest range of (15,100] compared to CL-time-LSTM-R, while CL-time-LSTM-R has lower percentages in the medium and lower ranges of (10,15] and [0,10]. Another observation can be made about the individual cluster models when compared to the performance of LSTM-HP in the corresponding hour ranges. Specifically, $\text{Model}_{cluster1}$, $\text{Model}_{cluster2}$ and $\text{Model}_{cluster3}$ have lower percentages in the (15,100] range compared to $\text{LSTM-HP}_{cluster1}$, $\text{LSTM-HP}_{cluster2}$ and $\text{LSTM-HP}_{cluster3}$.

4.4 Proposed model vs machine learning regressors

The proposed method CL-time-LSTM-R has also been compared with 4 traditional machine learning algorithms for regression. Specifically, these methods are Linear Regression, Ridge Regression, Decision Tree and Support Vector Regression (SVR) [33]. The results are given in Table 8.

Firstly, we observe that LSTM-HP outperforms the traditional methods across the most performance measures, with only exception that Ridge Regression was marginally better in terms of the metrics *RMSE*, *RRMSE* and R^2. More interestingly, the proposed model CL-time-LSTM-R outperformed the traditional methods in every case, with the differences in performance being more noticeable than those of LSTM-HP. In the [0,10] error range, CL-time-LSTM-R achieved a higher percentage compared to the other regressors. This is also true for the (10,15] error range, while LSTM-HP has a marginally lower percentage than. In the (15,100] range, CL-time-LSTM-R has the lower percentage of errors among all competitors, followed by Ridge Regression and LSTM-HP. Based on that error distribution, we can safely reach to the conclusion that the proposed method obtained the most robust performance, recording both accurate predictions and eliminating its deviations.Despite the fact that LSTM-based models demand increased computational resources compared with conventional ML regressors, the improved performance for offline predictions with large lead time compensates that fact.

A violin representation for the absolute percentage error of each distribution is given in Figure 3. It is noticeable that the quartiles Q1, Q2 and Q3 for the LSTM based methods are lower compared to those of the traditional regressors. Moreover, this graphical illustration constitutes an additional qualitative confirmation for the improved error distribution of the LSTM architecture.

5 CONCLUSION

To sum up, we present a dedicated LSTM-based algorithm for tackling the short-term load forecasting problem. Towards that direction, we applied a proper feature engineering stage for integrating weather and time information along with the raw-signal, and exploited signal decomposition for feeding our clustering mechanism in order to capture better time dependencies. The proposed pipeline is totally data-driven, and our results prove the superiority of the clustered-based predictions against the conventional LSTM model, as well as against some popular ML-based regressors. The results obtained by five different performance metrics are presented, along with a more qualitative analysis of the error distribution, revealing

Table 7: Train Cl-time-LSTM model to compare Integrated clustering and Residual clustering.

Model	MAPE	MAE	RMSE	RRMSE	R^2	[0,10]	(10,15]	(15,100]
Model$_{cluster1}$	6.75%	0.051	0.07	9.12%	0.311	75.83%	13.21%	10.96%
Model$_{cluster2}$	4.20%	0.026	0.035	5.75%	0.473	91.51%	6.36%	2.13%
Model$_{cluster3}$	7.102%	0.051	0.072	9.73%	0.4	78.27%	9.82%	11.92%
CL-time-LSTM-R	5.88%	0.042	0.06	8.54%	0.394	82.32%	9.80%	7.89%
LSTM-HP$_{cluster1}$	7.048%	0.054	0.074	9.662%	0.232	76.267%	10.684%	13.047%
LSTM-HP$_{cluster2}$	4.983%	0.031	0.041	6.711%	0.302	88.858%	7.488%	3.652%
LSTM-HP$_{cluster3}$	7.334%	0.055	0.077	10.130%	0.277	77.142%	9.863%	12.994%
LSTM-HP	6.357%	0.045	0.065	9.223%	0.272	81.244%	9.247%	9.509%

Table 8: Compare LSTM models with 4 machine learning algorithms in test set.

Regressors	MAPE	MAE	RMSE	RRMSE	R^2	[0,10]	(10,15]	(15,100]
Linear regression	7.73%	0.055	0.083	11.72%	-0.15	74.06%	14.77%	11.16%
Ridge regression	6.70%	0.048	0.064	9.09%	0.28	78.62%	13.16%	8.22%
Decision tree	7.01%	0.049	0.072	10.23%	0.07	77.14%	10.45%	12.42%
SVR	7.65%	0.055	0.07	9.88%	0.1	70.15%	19.86%	9.99%
LSTM-HP	6.35%	0.045	0.065	9.22%	0.27	81.24%	9.25%	9.51%
CL-time-LSTM-R	5.88%	0.042	0.06	8.53%	0.39	82.32%	9.80%	7.89%

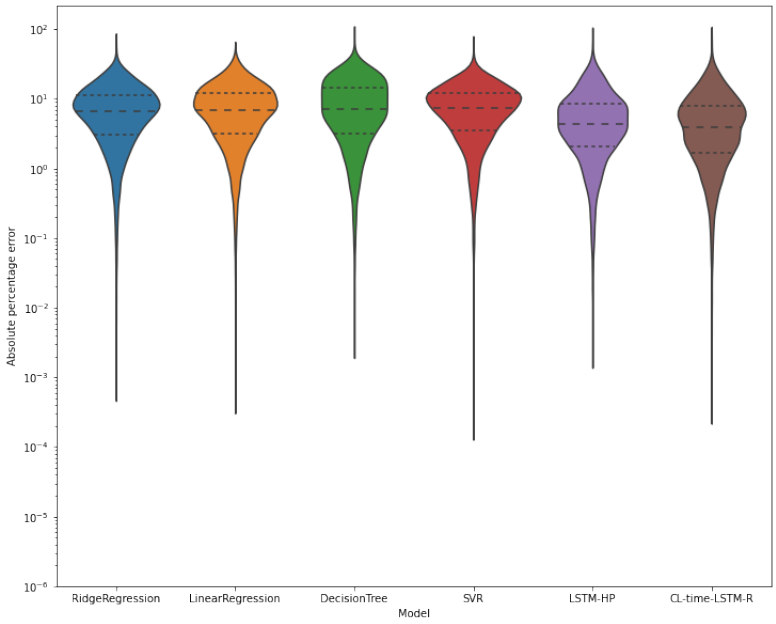

Figure 3: Violin plots for absolute percentage error (%)

the positive effect of adopting a strategy that concatenates forecasts from clustered-based models that stem from different time periods.

In future work, we plan to gradually tune and evaluate the proposed architecture for different time periods and or time horizon, revealing more insights about its learning capacity and validating the coverage of the current decomposition and clustering choices. More recent decomposition strategies should be examined [22, 34], trying at the same time to exploit those exported components also during the inference stage, signal transformations [26], or even apply attention-based mechanisms for better utilising the correlation of the load signal with the rest of collected indicators [23].

ACKNOWLEDGMENTS

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code: T2EDK-03048)

REFERENCES

[1] Juan A Acero, Elliot J K Koh, Gloria Pignatta, and Leslie K Norford. 2020. Clustering weather types for urban outdoor thermal comfort evaluation in a tropical area. *Theoretical and Applied Climatology* 139, 1 (2020), 659–675. https://doi.org/10.1007/s00704-019-02992-9

[2] Ernesto Aguilar Madrid and Nuno Antonio. 2021. Short-Term Electricity Load Forecasting with Machine Learning. *Information* 12, 2 (2021). https://doi.org/10.3390/info12020050

[3] Songpu Ai, Antorweep Chakravorty, and Chunming Rong. 2019. Household Power Demand Prediction Using Evolutionary Ensemble Neural Network Pool with Multiple Network Structures. *Sensors* 19, 3 (2019). https://doi.org/10.3390/s19030721

[4] Mohanad S. Al-Musaylh, Ravinesh C. Deo, Jan F. Adamowski, and Yan Li. 2018. Short-term electricity demand forecasting with MARS, SVR and ARIMA models using aggregated demand data in Queensland, Australia. *Advanced Engineering Informatics* 35 (2018), 1–16. https://doi.org/10.1016/j.aei.2017.11.002

[5] Duong Tuan Anh and Le Huu Thanh. 2015. An efficient implementation of k-means clustering for time series data with DTW distance. *International Journal of Business Intelligence and Data Mining* 10, 3 (2015), 213–232.

[6] Phil Aupke, Andreas Kassler, Andreas Theocharis, Magnus Nilsson, and Michael Uelschen. 2021. Quantifying Uncertainty for Predicting Renewable Energy Time Series Data Using Machine Learning. *Engineering Proceedings* 5, 1 (2021). https://doi.org/10.3390/engproc2021005050

[7] Rosangela Ballini and Ronald R. Yager. 2014. OWA filters and forecasting models applied to electric power load time series. *Evol. Syst.* 5, 3 (2014), 159–173. https://doi.org/10.1007/s12530-014-9112-2

[8] Karol Bot, Antonio Ruano, and Maria da Graça Ruano. 2020. Forecasting Electricity Consumption in Residential Buildings for Home Energy Management Systems. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Marie-Jeanne Lesot, Susana Vieira, Marek Z. Reformat, João Paulo Carvalho, Anna Wilbik, Bernadette Bouchon-Meunier, and Ronald R. Yager (Eds.). Springer International Publishing, Cham, 313–326.

[9] Moulay Larbi Chalal, Medjdoub Benachir, Michael White, and Raid Shrahily. 2016. Energy planning and forecasting approaches for supporting physical improvement strategies in the building sector: A review. *Renewable and Sustainable Energy Reviews* 64, C (2016), 761–776. https://doi.org/10.1016/j.rser.2016.06.04

[10] Cristina Heghedus, Antorweep Chakravorty, and Chunming Rong. 2018. Energy Load Forecasting Using Deep Learning. In *2018 IEEE International Conference on Energy Internet (ICEI)*. 146–151. https://doi.org/10.1109/ICEI.2018.00-23

[11] Benedikt Heidrich, Marian Turowski, Nicole Ludwig, Ralf Mikut, and Veit Hagenmeyer. 2020. Forecasting energy time series with profile neural networks. In *e-Energy '20: The Eleventh ACM International Conference on Future Energy Systems, Virtual Event, Australia, June 22-26, 2020*. ACM, 220–230. https://doi.org/10.1145/3396851.3397683

[12] Benedikt Heidrich, Marian Turowski, Nicole Ludwig, Ralf Mikut, and Veit Hagenmeyer. 2020. Forecasting energy time series with profile neural networks. In *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*. 220–230.

[13] Julia Hidalgo and Renaud Jougla. 2018. On the use of local weather types classification to improve climate understanding: An application on the urban climate of Toulouse. *PLOS ONE* 13, 12 (12 2018), 1–21. https://doi.org/10.1371/journal.pone.0208138

[14] Tareq Hossen, Arun Sukumaran Nair, Radhakrishnan Angamuthu Chinnathambi, and Prakash Ranganathan. 2018. Residential Load Forecasting Using Deep Neural Networks (DNN). In *2018 North American Power Symposium (NAPS)*. 1–5. https://doi.org/10.1109/NAPS.2018.8600549

[15] Jun Huang, Baohua Yu, Cong-Cong Xing, Tomas Cerny, and Zhaolong Ning. 2021. Online Energy Scheduling Policies in Energy Harvesting Enabled D2D Communications. *IEEE Transactions on Industrial Informatics* 17, 8 (2021), 5678–5687. https://doi.org/10.1109/TII.2020.3005440

[16] Junhong Kim, Jihoon Moon, Eenjun Hwang, and Pilsung Kang. 2019. Recurrent inception convolution neural network for multi short-term load forecasting. *Energy and Buildings* 194 (2019), 328–341. https://doi.org/10.1016/j.enbuild.2019.04.034

[17] Weicong Kong, Zhao Yang Dong, Youwei Jia, David J. Hill, Yan Xu, and Yuan Zhang. 2019. Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network. *IEEE Transactions on Smart Grid* 10, 1 (2019), 841–851. https://doi.org/10.1109/TSG.2017.2753802

[18] Irena Koprinska, Dengsong Wu, and Zheng Wang. 2018. Convolutional Neural Networks for Energy Time Series Forecasting. In *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018*. IEEE, 1–8. https://doi.org/10.1109/IJCNN.2018.8489399

[19] Muralitharan Krishnan, Yoon Mo Jung, and Sangwoon Yun. 2020. Prediction of Energy Demand in Smart Grid using Hybrid Approach. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*. 294–298. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00055

[20] Brecht Laperre, Jorge Amaya, and Giovanni Lapenta. 2020. Dynamic Time Warping as a New Evaluation for Dst Forecast with Machine Learning. *CoRR* abs/2006.04667 (2020). arXiv:2006.04667 https://arxiv.org/abs/2006.04667

[21] Jian Li, Daiyu Deng, Junbo Zhao, Dongsheng Cai, Weihao Hu, Man Zhang, and Qi Huang. 2021. A Novel Hybrid Short-Term Load Forecasting Method of Smart Grid Using MLR and LSTM Neural Network. *IEEE Transactions on Industrial Informatics* 17, 4 (2021), 2443–2452. https://doi.org/10.1109/TII.2020.3000184

[22] Yanfei Li, Haiping Wu, and Hui Liu. 2018. Multi-step wind speed forecasting using EWT decomposition, LSTM principal computing, RELM subordinate computing and IEWT reconstruction. *Energy Conversion and Management* 167 (2018), 203–219. https://doi.org/10.1016/j.enconman.2018.04.082

[23] Jun Lin, Jin Ma, Jianguo Zhu, and Yu Cui. 2022. Short-term load forecasting based on LSTM networks considering attention mechanism. *International Journal of Electrical Power & Energy Systems* 137 (2022), 107818. https://doi.org/10.1016/j.ijepes.2021.107818

[24] Daniel L. Marino, Kasun Amarasinghe, and Milos Manic. 2016. Building energy load forecasting using Deep Neural Networks. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. 7046–7051. https://doi.org/10.1109/IECON.2016.7793413

[25] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[26] Christoforos Nalmpantis, Odysseas Krystalakos, and Dimitris Vrakas. 2018. Energy profile representation in vector space. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence, SETN 2018, Patras, Greece, July 09-12, 2018*. ACM, 22:1–22:5. https://doi.org/10.1145/3200947.3201050

[27] Slobodan Petrovic. 2006. A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters. In *Proceedings of the 11th Nordic Workshop of Secure IT Systems*, Vol. 2006. Citeseer, 53–64.

[28] Steven M Pincus. 1991. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences* 88, 6 (1991), 2297–2301.

[29] M Pumperla. 2017. Hyperas: a very simple convenience wrapper around hyperopt for fast prototyping with Keras model. *Accessed: Apr* 30 (2017), 2019.

[30] Jason Runge and Radu Zmeureanu. 2019. Forecasting Energy Use in Buildings Using Artificial Neural Networks: A Review. *Energies* 12, 17 (2019). https://doi.org/10.3390/en12173254

[31] Bikash Kumar Sahu. 2018. Wind energy developments and policies in China: A short review. *Renewable and Sustainable Energy Reviews* 81 (2018), 1393–1405. https://doi.org/10.1016/j.rser.2017.05.183

[32] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett (Eds.). 802–810. https://proceedings.neurips.cc/paper/2015/hash/07563a3fe3bbe7e3ba84431ad9d055af-Abstract.html

[33] Amanpreet Singh, Narina Thakur, and Aakanksha Sharma. 2016. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. Ieee, 1310–1315.

[34] Angela Stallone, Antonio Cicone, and Massimo Materassi. 2020. New insights and best practices for the successful use of Empirical Mode Decomposition, Iterative Filtering and derived algorithms. *Scientific Reports* 10, 1 (2020), 15161. https://doi.org/10.1038/s41598-020-72193-2

[35] Marina Theodosiou. 2011. Forecasting monthly and quarterly time series using STL decomposition. *International Journal of Forecasting* 27, 4 (2011), 1178–1195.

[36] Nicolai Bo Vanting, Zheng Ma, and Bo Nørregaard Jørgensen. 2021. A scoping review of deep neural networks for electric load forecasting. *Energy Informatics* 4, 2 (2021), 49. https://doi.org/10.1186/s42162-021-00148-6

[37] Zhaohua Wu and Norden E. Huang. 2009. Ensemble Empirical Mode Decomposition: a Noise-Assisted Data Analysis Method. *Adv. Data Sci. Adapt. Anal.* 1, 1 (2009), 1–41. https://doi.org/10.1142/S1793536909000047

[38] Hong Bo Zhou and Jun Tao Gao. 2014. Automatic method for determining cluster number based on silhouette coefficient. In *Advanced Materials Research*, Vol. 951. Trans Tech Publ, 227–230.