applied sciences

MDPI

Article

Novel Data-Driven Models Applied to Short-Term Electric Load Forecasting

Manuel Lopez-Martin [1,*], Antonio Sanchez-Esguevillas [1], Luis Hernandez-Callejo [2,*], Juan Ignacio Arribas [1,3] and Belen Carro [1]

[1] Department TSyCeIT, ETSIT, University of Valladolid, Paseo de Belén 15, 47011 Valladolid, Spain; antoniojavier.sanchez@uva.es (A.S.-E.); jarribas@tel.uva.es (J.I.A.); belcar@tel.uva.es (B.C.)
[2] Department EiFAB, University of Valladolid, Campus Universitario Duques de Soria, 42004 Soria, Spain
[3] Castilla-Leon Neuroscience Institute, University of Salamanca, 37007 Salamanca, Spain
* Correspondence: manuel.lopezm@uva.es (M.L.-M.); luis.hernandez.callejo@uva.es (L.H.-C.)

Abstract: This work brings together and applies a large representation of the most novel forecasting techniques, with origins and applications in other fields, to the short-term electric load forecasting problem. We present a comparison study between different classic machine learning and deep learning techniques and recent methods for data-driven analysis of dynamical models (dynamic mode decomposition) and deep learning ensemble models applied to short-term load forecasting. This work explores the influence of critical parameters when performing time-series forecasting, such as rolling window length, k-step ahead forecast length, and number/nature of features used to characterize the information used as predictors. The deep learning architectures considered include 1D/2D convolutional and recurrent neural networks and their combination, Seq2seq with and without attention mechanisms, and recent ensemble models based on gradient boosting principles. Three groups of models stand out from the rest according to the forecast scenario: (a) deep learning ensemble models for average results, (b) simple linear regression and Seq2seq models for very short-term forecasts, and (c) combinations of convolutional/recurrent models and deep learning ensemble models for longer-term forecasts.

Keywords: short-term electric load forecasting; deep learning; machine learning; dynamic mode decomposition; deep learning ensemble model

Citation: Lopez-Martin, M.; Sanchez-Esguevillas, A.; Hernandez-Callejo, L.; Arribas, J.I.; Carro, B. Novel Data-Driven Models Applied to Short-Term Electric Load Forecasting. *Appl. Sci.* **2021**, *11*, 5708. https://doi.org/10.3390/app11125708

Academic Editors: Matti Lehtonen and Mohsen Soltani

Received: 23 April 2021
Accepted: 18 June 2021
Published: 20 June 2021

1. Introduction

Short-term load forecasting (STLF) is of vital importance to utility companies in many areas, such as maintenance, operations, and reliability. Achieving accurate load forecasts is a difficult task due to the highly non-linear nature of the underlying model. Forecasts in this area have been historically treated with time-series statistical analysis methods, e.g., autoregressive integrated moving average (ARIMA) [1], with a clear trend towards the use of machine learning techniques in current times [2].

STLF is a solid area of research with a significant body of literature exploring the application of specific techniques to specific datasets or the review of techniques applied in different research works to different datasets. Meanwhile, the number of papers proposing new forecasting methods in the areas of dynamic modeling and machine learning (and more specifically deep learning) continues to grow, which makes it very interesting to have a systematic comparison, based on a single dataset, of a significant number of novel techniques.

In this work, we propose a comprehensive study of STLF with a significant number of data-driven forecasting models that are novel or rarely applied to STLF, such as: (a) dynamic mode decomposition (DMD) [3–5], (b) deep learning (DL) models based on specific combinations of convolutional neural networks (CNN) and recurrent neural networks (RNN) [6,7], (c) sequence to sequence (Seq2seq) models with and without soft

attention [8–10], and (d) deep learning ensemble models specially targeted for time-series forecasting [11]; and a set of well-known forecasting models, such as: (e) classic machine learning (ML) models: linear regression, random forest, gradient boosting, k-nearest neighbors, support vector regression and AdaBoost [12–15], (f) multi-layer perceptron [2], and (g) deep learning models based on separate CNN and RNN networks [13,14]. By jointly evaluating the novel or less used techniques with the best known, we can compare their performance in different scenarios.

We carried out the study by applying the different models to a real dataset of power consumption from a Spanish utility for the province capital of Soria (Spain). This dataset has been extensively studied previously [16–23].

This work is different from other studies: (a) by applicability, not being a reference to other studies and applying all the models to the same dataset which allows comparison of results, (b) by novelty, by including, in the same study, the results from DMD [5], new DL ensemble models based on gradient boosting principles [11], specific configurations of CNN/RNN [6,7], and Seq2seq models [8–10], and (c) by extension, by considering a large number of different models under different scenarios.

STLF needs a previous preparation of the load values used as predictors. The preparation consists of aggregating these values in discrete time intervals (time-slots) that can be seconds, minutes, or hours, and the forecast can be based on a different number of previous values (predictors) that are obtained with a rolling window process applied to the past values. The length of the rolling window defines the number of predictors. With these predictors, the forecast can be extended to the following time-slot or to several time-ahead time-slots (k-step ahead forecast). Finally, the value of the predictors can be a scalar (load value), or a vector composed of the load value plus additional information, such as date/time or weather data. Considering the scenario with vector predictors and multiple time-ahead forecasts, we arrive at a challenging multivariate multi-output regression problem. The influence of these parameters: (a) length of rolling window, (b) length of time-ahead forecast, and (c) features used to form the predictors, is analyzed in this research in combination with the different natures exposed by the models.

Time-series statistical analysis algorithms (e.g., ARIMA) have not been contemplated as they are trained to specific past values, while our goal is to have a single algorithm that can be trained once and generalizes to new past values. In addition, they are mainly intended for single-output forecasts and their extension to multi-output scenarios (i.e., forecasting k future values) generates complex models (e.g., VARIMA).

To appraise different possible evaluation objectives associated with STLF, we have obtained six different metrics to assess the forecasting performance of the models: mean square error (MSE), mean absolute error (MAE), median absolute error (MAD), coefficient of determination (R^2), relative root mean squared error (RRMSE), and symmetric mean absolute percentage error (sMAPE). We also analyze the influence of the time-ahead forecast interval and the rolling window length on these metrics.

Results are presented following three different forecast objectives: very short-term forecasts (immediate time-slots in the forecast time horizon), longer-term forecasts (last time-slots in the forecast time horizon), and forecast average (average of all time-slots in the forecast time horizon). Considering these three scenarios, we have obtained the three best groups of models according to the different forecast objectives: (a) deep learning ensemble models for average results, (b) simple linear regression and Seq2seq models for very short-term forecasts, and (c) combinations of convolutional/recurrent models and deep learning ensemble models for longer-term forecasts.

We have explored the use of deep learning blocks based on the ensemble model presented in [11], called gaNet. The gaNet architecture has been used for IoT traffic prediction [11] and classification [24]. This is the first time that this architecture has been applied to STLF. A gaNet architecture is made up of small deep learning networks formed by a few Convolutional (Conv) and/or RNN layers. These networks are organized as repeating blocks whose outputs are combined (after an aggregation function) into a final output.

The input to the architecture is shared by all the blocks. This architecture is connected to gradient boosting models, stacked models [25], and residual networks [26].

It is worth noting the excellent results of the proposed deep learning ensemble model (gaNet) and how it excels in average results and in longer-term (most difficult) forecasts. This good behavior can be connected with having a repeating set of randomly initialized deep learning blocks, in line with other studies that connect the importance of a rich set of random initializations with the behavior of deep learning models [27]. Deep ensembles can also provide an improvement in uncertainty estimates for samples outside the expected data distribution, through appropriate data selection and a specific loss function that maximizes model diversity [28]. This work contributes to provide additional results that confirm the good behavior of deep ensembles under an additional perspective provided by gaNet. In previous works [11,24], gaNet has been applied to time-series forecasts with a panel data structure (a list of entities, each with an associated time-series), whereas this work applies it to a single time-series with different requirements for data preparation and the validation/testing process.

An additional objective of the study is to put forward the availability of accurate forecast results as a valuable tool for identifying new applications, such as: (a) identification of anomalous consumption patterns due to excessive deviations from the forecasts, which can be used as alarms for security or fraud situations and, (b) simulation of what-if non-standard load scenarios and their consequences. These two applications are important for smart grids and their convergence with the IoT (Internet of Things) infrastructure with higher cybersecurity and fraud risks [29,30].

As a summary, **the contributions of this work** are: (1) Extend an ensemble deep learning model (i.e., gaNet) based on the gradient boosting architecture to the particular needs of STLF. (2) Present a thorough analysis of STLF models with a special emphasis on novel methods, e.g., gaNet, DMD, Seq2seq, and combinations of CNN/RNN models. (3) Apply all the models to a previously well-studied dataset of real electricity consumption, allowing comparisons to be made on a single dataset in a homogeneous and structured way, which allows comparison of results and drawing conclusions on common bases. (4) Analyze the impact on forecasts due to: (a) length of rolling window, (b) length of time-ahead forecast, and (c) features used to form the predictors. (5) Present the best groups of models according to different forecast objectives. (6) Apply to STLF, for the first time, as far as we know, the gaNet model and the specific configurations that combine convolutional and recurrent layers as proposed here [6,7].

The organization of the paper is as follows: Section 2 summarizes related works. Section 3 describes the dataset and the forecast models. Section 4 provides the results and Section 5 presents the conclusions.

2. Related Works

There is a large body of recent work on STLF applying many different techniques which show that even when it may be seen as a well-known area of study, it is still an important area of research due to its technical difficulty and economic impact of having an accurate load forecast. The intention of this work is to contribute to this area by providing practical results on the application of some of the newest methods for time-series forecasting applied to STFL in an extensive and homogeneous way.

We will present related works considering the applied methods, global review studies, and some of the anticipated applications that result from having accurate forecasts. The presentation will focus more on adopted methods and processes than on performance metric comparison, since the diversity of datasets, the difference in load magnitudes, the differences in the implementation of the metrics, and the various test/validation procedures make it very difficult to perform a homogeneous comparison of results.

Review of current techniques: The work in [2] presents a comprehensive review of the techniques used to forecast electricity demand, analyzing the different types of forecasts, parameters affected, and techniques used, together with a literature review and a taxonomy

of the main variables involved in the problem. The work in [13] presents a detailed review of recent literature and techniques applied for building energy consumption modeling and forecasting.

A complete review of machine learning for load prediction is given in [31] with emphasis on the following methods: tree-based methods and random forest, support vector regression (SVR), neural networks, ensemble learning, statistical methods (ARMA, ARIMA, ...), gaussian processes, k-nearest neighbors regressor (KNN), and fuzzy time-series algorithms. An alternative review is given in [32] highlighting the following models: linear regression, artificial neural networks (ANN), SVR, statistical methods, restricted Boltzmann machine, and deep learning (CNN and LSTM). A recent review of machine learning models applied to load forecasting focusing on performance metrics results is proposed in [33]. This study presented the following methods as the most frequently implemented in electric power forecasting: statistical methods, artificial neural networks (ANNs), support vector machines (SVMs), decision trees (DTs), adaptive neuro fuzzy inference systems (ANFISs), and recurrent neural networks (RNNs).

The work in [34] presents the application of five machine learning models for STLF: multiple linear regression, KNN, SVR, and gradient boosting. The best performance is obtained for gradient boosting. Authors in [35] present a recent and systematic review of techniques applied to load forecasting with an emphasis on neural networks and hybrid models. The main models presented are: SVM, GBM, random forest (RF), KNN, neural networks, decision trees, statistical methods, CNN, LSTM, and the adaptive neuro-fuzzy inference system (ANFIS). The authors in [36,37] provide an study of methods to use in load forecasting considering different forecasting problems and objectives. The main methods discussed are: statistical methods, AdaBoost, GBM, random forest, generalized additive models (GAM), gaussian processes, KNN, linear regression, and support vector regression. The work in [38] presents an extensive literature review with no mention of dynamical systems modeling or deep learning or ensemble deep learning models. The methods mentioned are: multioutput linear regression, generalized additive models, statistical models, neural networks, support vector machines, gradient boosting, and fuzzy regression. The review in [39] favors the use of artificial neural networks, support vector regression, and fuzzy logic models.

Dynamical systems modeling: There is a growing current interest in the application of dynamical systems analysis tools based on reduced-order models and, in particular, in the use of dynamic mode decomposition to STLF. The work in [5] provides a DMD study applied to electric load data from a utility operator in Queensland, Australia. In this work, the DMD algorithm presents better forecasting results than classic time-series autoregressive approaches. They offer a one day ahead forecast based on the load values of the previous 4 days, presenting a MAPE result of 2.13. A similar application of DMD is carried out in [40] but applying DMD to predict forecast errors followed by an extreme value constraint method to further correct the forecasts. The algorithm is applied to actual load demand data from the grid in Tianjin, China, and the results obtained with DMD are compared with a series of additional techniques (autoregressive moving average, neural networks, support vector machines, extreme learning machines, etc.). According to the authors, the proposed method shows greater accuracy and stability than alternative ones, with a best average root mean squared error (RMSE) of 476.17. In [41], the authors employ an empirical mode decomposition technique to extract different modes from the load signal, and apply an independent deep belief network for each mode prediction, with a subsequent aggregation of results (ensemble) to obtain the final load forecast. More classic ensemble techniques for forecasting electricity consumption in office buildings are investigated in [42], comparing gradient tree boosting (GTB), random forests (RF), and a specifically adapted AdaBoost model that presents the best results.

Classic machine learning: A substantial number of works have presented several classic machine learning models to STLF. A feed-forward artificial neural network (FF-ANN) is used in [43] to forecast the electricity consumption for residential buildings for

24 h. The results are compared with other models including GTB and RF, selecting the best model at each forecast iteration. The best average result for the different test iterations is obtained for the ANN with an RMSE of 2.48. The work in [44] presented a theoretical review of the most commonly used ML methods for STLF, including ANN and a support vector for regression. The work in [45] presents an interesting pipeline method based on combining a feature transformation using a clustering approach followed by a support vector regression. The authors in [46] introduce an additive regression model for STLF.

Statistical analysis methods: Time-series statistical analysis models for STLF are discussed in detail in [1] with forecasts at an hourly interval applied to load data from the Electric Reliability Council of Texas (ERCOT). They present results applying ARIMA and Seasonal Autoregressive Integrated Moving Average (SARIMA) models achieving an average mean absolute percentage error (MAPE) between 4.36% to 12.41%.

Sequence to sequence models: The sequence to sequence (Seq2seq) architecture that originated in the field of natural language processing (NLP) has been applied in recent works to STLF. The authors in [47] apply different Seq2seq architectures, comparing them with other DL models based on recurrent and convolutional layers. The models are applied to two different datasets (scenarios): one for an individual household electric power consumption data set (IHEPC) located in Sceaux, France, and the other for the GEFCom2014 public dataset made available for the global energy forecasting competition 2014. The best results (RMSE between 17.2 and 0.75 depending on the scenario) are obtained with convolutional and recurrent architectures together with deep neural networks with dense layers. Considering average results, the Seq2seq models do not provide the best results. The conclusions obtained in this work are consistent with the results obtained by the present study. A similar study is presented in [48], where research was conducted comparing a Seq2seq model (with and without attention) with alternative DL models based exclusively on different types of recurrent networks, such as long short-term memory network (LSTM) and gated recurrent unit (GRU). In this case, the Seq2seq model presents the best results for short-term forecasting, also in accordance with the results obtained in the present work. A generic Seq2seq with a specific attention mechanism is proposed in [49] for multivariate time-series forecasting.

Deep learning models: Using the same dataset proposed for this work, [16] presents an ANN model that works on a 24 h day-ahead forecasting of electric loads previously aggregated into clusters by consumption patterns. The patterns are obtained with a self-organizing map (SOM) followed by a k-means clustering algorithm. The work in [50] introduces a deep learning architecture based on an ensemble of convolutional blocks acting on segregated subsets of the input data. The model is applied for day-ahead forecasting of individual residential loads with data obtained from a smart metering electricity customer behavior trial (CBTs) in Ireland. The work focuses on achieving low training time and high accuracy, the proposed model being the best in both aspects with an MAE of 0.3469. The authors in [51] present a simple MLP regressor with a sophisticated selection of features based on previous consumptions and weather data. The work in [52] introduces a combined model of CNN and LSTM layers for STLF. A comparison between recurrent neural networks (LSTM) and support vector regression (SVR) is provided in [53], showing that the bidirectional LSTM model outperforms both the unidirectional LSTM and the SVR models.

3. Materials and Methods

This section presents the dataset and forecast models (Sections 3.1 and 3.2, respectively) used for the experiments.

3.1. Selected Dataset

The dataset employed in this research corresponds to real data from a Spanish utility over a period of three years. The load distribution presents similarities to that of a

microgrid [16], with a value range between 7370 and 39,550 kW. The values have annual periodicity, also with strongly defined daily and weekly periodicity (Figure 1).

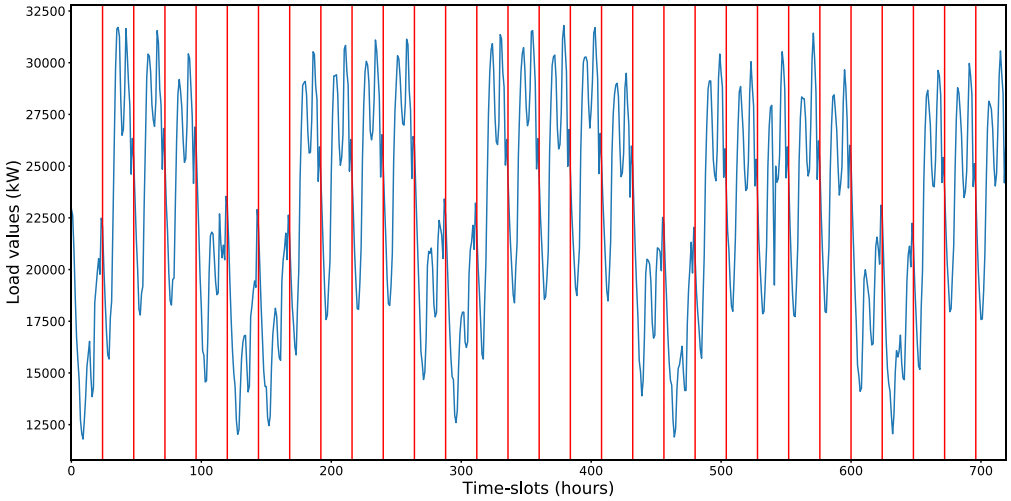

Figure 1. Chart showing the load values for a period of 30 days and time-slots of 1 h. The vertical red lines indicate a 24 h period (1 day). We can observe the strong daily and weekly periodicity.

Load values were aggregated in time-slots of one hour. The total number of time-slots corresponded to 26,302 h. Additional exogenous features were also considered: (a) Date/time features: month, time, day of the week, and weekend indicator. (b) Weather features: mean and standard deviations for the atmospheric pressure, wind speed, wind direction (degrees), humidity, and solar radiation. Continuous features were scaled in the range [0,1]. Categorical features were one-hot-encoded. We considered four different sets of features out of the many possible feature combinations: (a) 1 feature: electricity load, (b) 45 features: date/time (day of week, weekend, hour, month) and load, (c) 57 features: date/time, weather and load, and (d) 76 features: date/time, load, plus day of the month (one-hot-encoded). The number of features for the different feature sets is denoted by f (Figure 2). The weather and day of the month features did not provide a clear improvement in the forecast metrics.

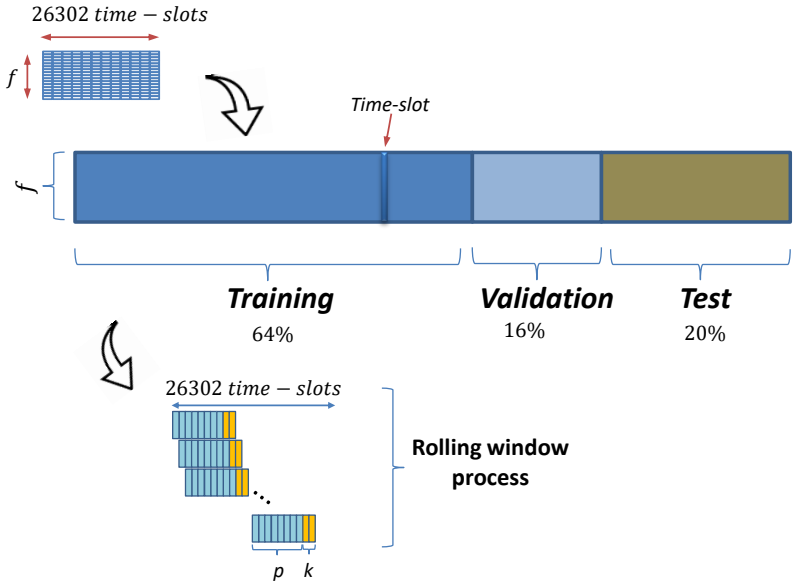

Figure 2. Division of the original dataset into training, validation, and test sets with a data proportion of 64, 16, and 20%, respectively. A rolling window process was used to extract the predictors sequence (p time-slots used as predictors) and the real time-ahead load values (k time-ahead forecast length). Each time-slot is represented by a vector of predictors with f components (load value, date/time features, etc.).

Figure 2 presents how the original dataset was divided into training, validation, and test sets with a data proportion of 64, 16, and 20%, respectively. Figure 2 also presents the rolling window process used to extract the predictors sequence (p time-slots used as predictors) and the real time-ahead values (k time-ahead forecast length). The validation data were used to assess model performance during training. The test set was used to provide all final results in Section 4. As a summary, we consider the following ranges of values for the parameters p, k, and f: (a) p: 24 (using the previous day of data—24 h), 168 (previous week of data—168 h), or 720 (previous month of data—720 h); (b) k: 24 (1-day forecast horizon), 168 (1-week horizon), or 720 (1-month horizon); and (c) f: 1, 45, or 57. The forecast results for different combination for these values (f, p, k) are reported separately in Section 4 and the Appendix A.

3.2. Models Description

This section presents the different models used in our research. The models have been grouped according to their characteristics. The presentation of results in Section 4 will follow the groups described here. The groups were the following:

- Classic machine learning models: We considered the following models: linear regression, random forest, gradient boosting, k-nearest neighbors, support vector regression, and AdaBoost. These models provide a good selection of machine learning models widely applied to STLF.
- Deep learning models: As already mentioned, deep learning models are currently the main trend in STLF. We applied various configurations of convolutional neural networks (CNN) and long short-term memory (LSTM) networks, a type of recurrent neural networks (RNN). The combination of CNN and LSTM networks has provided some of the best results, in accordance with the results obtained in other works applying the same configurations in other fields (network traffic analysis, video quality of experience, etc.) [6,7].
- Deep learning ensemble models: It is well-known that aggregating the capabilities of various estimators can increase their effectiveness in reducing errors and avoiding overfitting. There are several aggregation strategies, and boosting is one of the most important and provides state-of-the-art estimators. Bringing together boosting and deep learning has shown very good results in other forecasting problems [11]. The DL ensemble models included in this work follow the gaNet architecture [11], which is a deep learning boosting ensemble model specifically intended for time-series forecasting.
- Seq2seq models: The sequence to sequence model had its origins in the field of NLP, but it has been extended to many time-series prediction problems. It has rarely been used in STLF even when it provides good results when applied [48].
- Dynamic mode decomposition (DMD) models: These models attempt to approximate the non-linear latent drivers of a system by a linear transformation [4]. They are mainly applied in finance and fluid dynamics for both prediction and characterization of system behavior [3,54]. These models provide important information about the principal modes of the signal and their behavior. DMD is a technique that is currently attracting interest in STLF [5,55].

Figures 3–5 show different instances of the generic regression algorithm needed to transform the input sequence of p predictors into the output forecast sequence of length k. Figure 3 presents details of the models: classic ML, dynamic mode decomposition (DMD) and DL architectures. Figure 4 presents the details for the Seq2seq model with and without attention. Figure 5 presents the details for the ensemble models based on the gaNet architecture [11], which are deep learning ensemble configurations based on gradient boosting principles and are particularly suitable for time-series forecasting. These were the ensemble models used in this work.

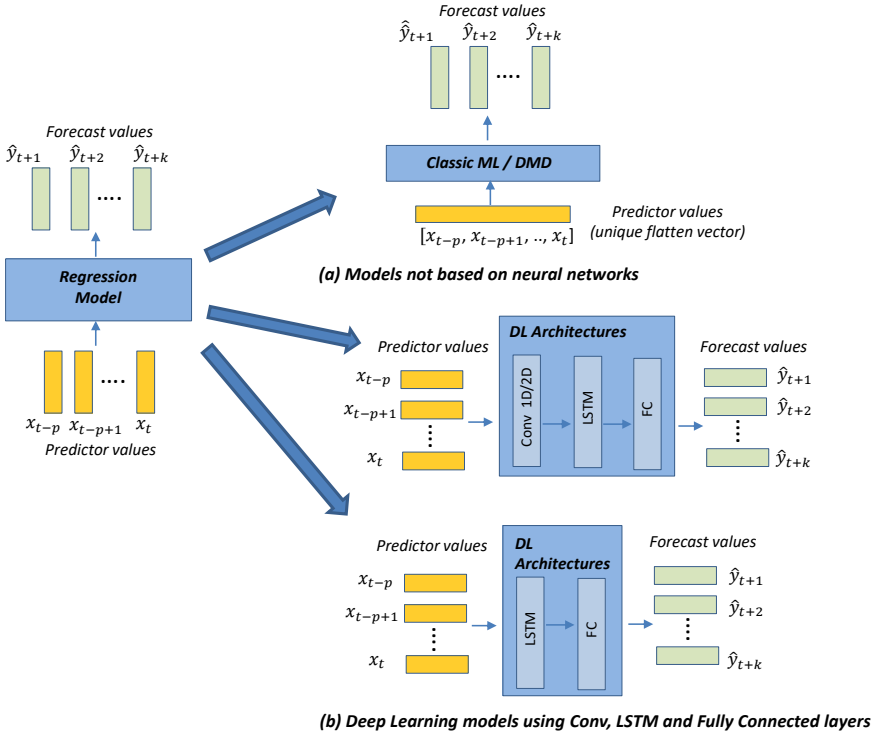

Figure 3. Detail of the models: classic ML, dynamic mode decomposition (DMD), and DL architectures.

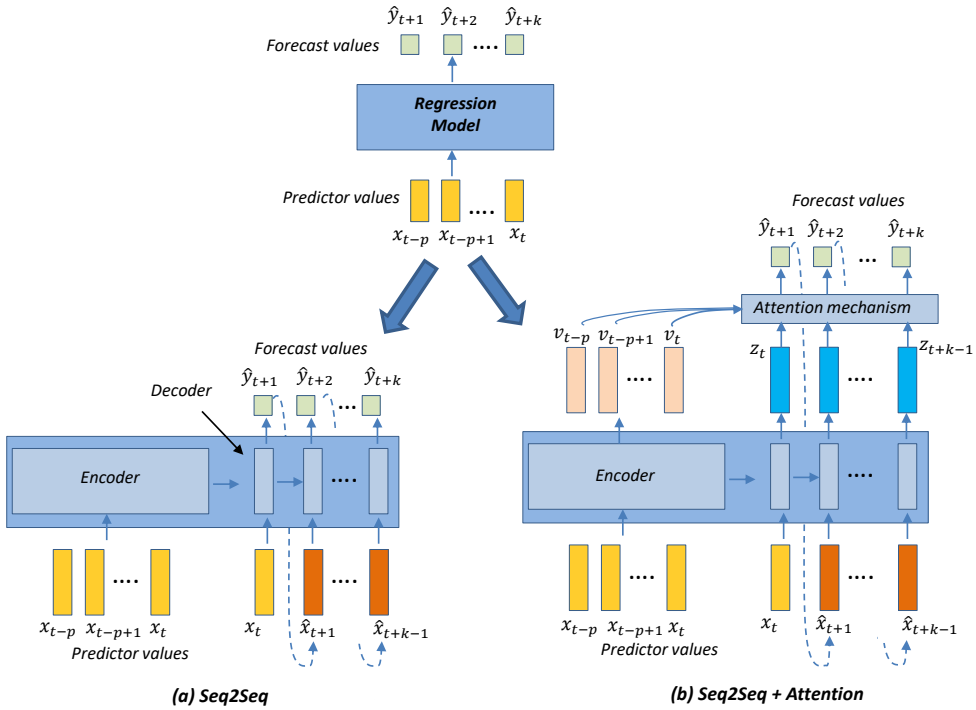

Figure 4. Detail of the Seq2seq and Seq2seq + attention models.

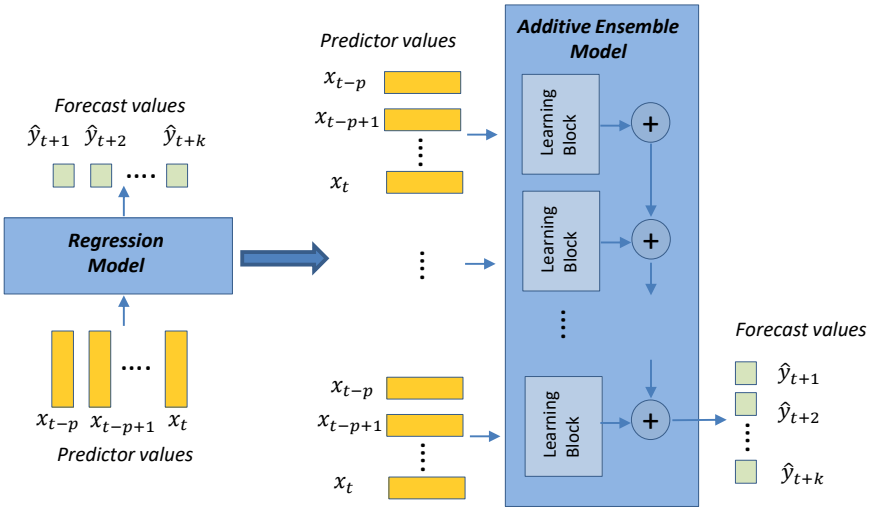

Figure 5. Detail of the deep learning ensemble models used in this work.

Figure 3 presents a schematic view of the ML, DL, and DMD models with an emphasis on showing the inputs received by the models and the generated outputs. We can observe how the inputs for the ML and DMD models are different to the inputs for the DL models, the reason being that DL models can receive vector-valued inputs, i.e., both LSTM [48] and 1D/2D CNN [47] models can receive a vector-valued sequence (with length p) where each timestep is represented by a vector of values. However, the ML and DMD models expect a sequence of scalar values (longitudinal data) as input; the way to transform the input data for these models is to flatten the vectors over all time-steps.

The DMD model [3,4] is different from the rest of the models since it has its origin in the study of dynamical systems, exploring the construction of a linear approximation to the dynamics of the system. It is a data-driven method based on approximating the temporal evolution of the system by a linear transformation given by a matrix (A). The eigenvectors and eigenvalues of this matrix define the dynamic modes of the system and their temporal evolution. The matrix A that defines the mapping between past (x_t) and future (x_{t+1}) snapshots of the system follows Equation (1). This expression can be extended to a complete set of snapshots obtained using the rolling window method presented in Figure 2. The snapshots set can be arranged in a matrix $X \in \mathbb{R}^{p \times N}$, where p is the length of the snapshots and N is the number of snapshots obtained with the rolling window process. Likewise, a new matrix $\hat{X}$ is created by selecting the same columns of X with a lag, i.e., if $\hat{X}_i$ is the i column of matrix $\hat{X}$, then $\hat{X}_i = X_{i+1}$. Once X and $\hat{X}$ are defined, we obtain Equations (2) and (3), where $X^{\dagger}$ is the pseudo inverse of X. The eigenvectors of X define the dynamic modes of the system. The problem with Equation (3) is that the matrices involved are large and their computation is difficult. The DMD method provides a solution to obtain the eigenvectors of A by computing the singular value decomposition of X (Equation (4)) and the matrix $\widetilde{A}$ (Equation (5)). The matrix $\widetilde{A}$ can be considered as the linear best fit for our original system. The matrix $\widetilde{A}$ has a much lower dimensionality than A, which allows us to easily obtain its eigenvalues (Λ) and eigenvectors (W) (Equation (6)). Finally, the eigenvalues of A are the same as the eigenvalues of $\widetilde{A}$ [3] and the eigenvectors of A (denoted as Φ) can be obtained from the previously obtained elements and computed following Equation (7).

$$x_{t+1} = A\, x_t \tag{1}$$

$$\hat{X} = A\, X \tag{2}$$

$$A = \hat{X}\, X^{\dagger} \tag{3}$$

$$X = U \Sigma V^{*}\ ;\ \hat{X} = A U \Sigma V^{*} \tag{4}$$

$$U^{*} \hat{X}\, V \Sigma^{-1} = U^{*} A U = \widetilde{A} \tag{5}$$

$$\widetilde{A}W = W\Lambda \tag{6}$$

$$\Phi = \hat{X}\, V\Sigma^{-1}\mathrm{W} \tag{7}$$

The previous presentation of DMD corresponds to a complete deployment of the algorithm. To further reduce computational needs, we can keep just the leading eigenvalues (Λ) and eigenvectors (W) of $\widetilde{A}$ while preserving most of the decomposition energy [3,4]. In this way, we reduce the dynamics of the system to its main eigenvalues and its corresponding eigenvectors (Φ). In Section 4, we present the main eigenvectors of the DMD decomposition for our data set for different rolling window lengths (p).

DMD allows us to implement forecasts [3,4] using Equation (8), where x_t is a forecast for time t, Λ^t is the power t of the diagonal matrix Λ, which is easily computed, and b_0 is an initial condition obtained by multiplying the inverse of the matrix Φ with the initial snapshot (first column) (X_0) of the matrix X:

$$b_0 = \Phi^{-1}\, X_0;\; x_t = \Phi\Lambda^t b_0 \tag{8}$$

As shown in Section 4, the DMD algorithm is not particularly efficient at forecasting, but it is very useful for identifying the fundamental modes and temporal behavior of a time series.

The classic ML models with best combined performance results, considering both the forecasting metrics and execution times, are: linear regression (LR) and random forest (RF) [13]. They are well-known ML models. They are robust, fast and do not require intensive hyperparameter tuning and are good at preventing overfitting. These models are particularly interesting for very short-term forecasts (Section 4). However, they experience problems with large values of the parameters p and f that produce large input vectors, since the inputs of these models are the predictors arranged as a flat vector. This has an impact on memory problems and computational times. Furthermore, these models allow a single prediction output, thus requiring as many regressors as outputs with a significant impact on training times for multi-step ahead forecasts, as is our case.

The DL models considered follow two configurations, depicted in Figure 3: (a) recurrent networks formed by one LSTM layer or two stacked LSTM layers plus a fully connected (FC) final layer, or (b) networks formed by a combination of 1D-Convolutional (1D-Conv) layers (between one to three) followed by one or two LSTM layers and a final FC layer. These two configurations have shown very good classification and regression performance in other fields [6,7], as well as other time-series forecasting problems [11]. We also explored the use of 2D-Conv layers instead of the 1D-Conv (common for time-series) layers by transforming the input sequence of vectors into a matrix that is interpreted as a pseudo-image, following the approach in [7]. Contrary to the good performance of this approach in other fields, in this case the results obtained by the architecture 2D-CNN + LSTM were not as good as expected (Figure A1—Appendix A).

The number of epochs used for training all models was 100, with an early-stop criteria established in 10 epochs without improvements. The batch size was 20 samples. We used a ReLU activation function for all layers, except the last layer with a linear activation. The loss function used was the mean squared error. There is a growing interest in the automatic hyperparameter optimization using AutoML techniques [56–58]. The extensive search of deep learning architectures proposed in this work could be applied to these new techniques and be the object of possible future research. The high demands for time and computational resources of these techniques must be considered to evaluate their suitability in each case. For this research, we opted for the selection of values based on heuristics and previous experience.

Sequence-to-sequence models (Seq2seq) (Figure 4) [8] originated in the NLP field to handle sequences of categorical values (words). They consist of two blocks: encoder and decoder. The encoder creates a latent representation (embedding) of the input, and the decoder uses the embedding to construct the forecast. The forecast is generally done in an iterative process where the values of the successive forecasts are entered as input for the

next forecast. The structure of the encoder and decoder layers is usually (but not necessarily) similar and employs a recurrent (LSTM) and fully connected layer. The addition of other types of layers is also possible to facilitate representation learning, such as convolutional layers. An attention mechanism [9,10] can be added to the Seq2seq model. The "attention" consists of a distance (similarity) comparison between the forecast and intermediate values connected to the past inputs, that is, the previous p predictor values. When the similarity operation is differentiable, we name it soft attention, and hard attention when it is not. In our case, we used soft attention with a softmax applied to the dot product between the forecast and the p intermediate values produced by the encoder (associated with the p values used as predictors). The result of this dot product plus softmax operation was applied to a fully connected layer that produced the final forecast.

The DL ensemble model used in this work follows the gaNet architecture [11] which is based on the creation of an estimator by aggregating blocks formed by small DL networks (Figure 5). All blocks are arranged in sequence, all sharing the same input. The output from the first block is aggregated to the output from the following blocks until the final output is produced. The aggregation process begins with a fixed value (usually an average of the expected results). The aggregation function used is the sum, but other functions such as the mean or maximum value can also be used. It is important to note that all blocks are trained together end-to-end using gradient descent.

In the basic gaNet model, all blocks have the same layer configuration (architecture) and all blocks in the sequence are similar, but not identical, as random initialization of their weights and end-to-end training will induce different weights in each of the blocks. There are also variants on this basic configuration by allowing different block architectures as well as other training options. In [11], the gaNet model is presented in detail with several variants (types), according to whether: the blocks architecture are all identical or not, if they share their weights, or if the loss function is a unique function or is formed by adding the loss functions of the intermediate outputs. Considering all the variants in [11], we chose only two that we named ensemble Type I and II, and that corresponded to types III and IV in [11]. The ensemble model Type I is made up of identical blocks where each block is formed by small DL networks consisting of 1D-Conv, LSTM, and FCN layers. We also differentiated a subgroup of Type I models when all the blocks shared their weights. In this case, we had blocks not only with identical architecture but with identical weights. It is interesting to investigate this specific subgroup because they are models with very few weights, which can be important to avoid overfitting. In the ensemble Type II model, blocks are separated into groups where each group can have a different architecture, with all blocks in the same groups sharing the same architecture. For this model, we indicate separately the number of repetitions of identical blocks per group. There is freedom in the number of groups and blocks per group. In this case, blocks with similar architectures could also share weights, but it is a possibility that has not been explored.

We deviated from the original Type II models presented in [11] where all blocks shared the same inputs, by allowing blocks with different architectures to receive different inputs. In this case, we used three different types of inputs: (a) a sequence of scalars formed exclusively by the load values that corresponds to a value of the parameter f equal to 1, (b) a sequence of vectors formed by the load values plus the date/time features that corresponds to a value of the parameter f equal to 45, and (c) a sequence of vectors formed by the load values plus the date/time and weather features that corresponds to a value of the parameter f equal to 57. All these options are presented in the results tables in Section 4 and Figure A1 (Appendix A), and, in those cases where there are different inputs per block, we have marked that in the tables by an NA in the f column, reporting the type of input associated with each block by an additional letter within a parenthesis and immediately after the layer type. Possible values for these additional letters are: an (L) for inputs formed exclusively by the load values, a (D) for inputs formed by the load plus the date/time features, and a (A) for inputs formed by the load plus the date/time and weather features.

4. Results and Discussion

The objective of this work was to assess the suitability of the different models for STLF. In this section, we present in detail the forecast performance metrics obtained by the different models considered for this research. An additional aim was to present together (and under homogeneous evaluation criteria) the results obtained by classic ML models and new time-series forecasting methods, with an emphasis on novel techniques, which originated in other fields and which have not been applied, or have rarely been applied, to STLF, i.e., models that integrate ideas from deep learning with gradient boosting and ensemble architectures (gaNet) [11], and models based on dynamical systems analysis techniques (DMD) [3–5]. The analyzed models are presented in detail in Section 3.2.

Several forecast performance metrics were considered: mean square error (MSE), mean absolute error (MAE), median absolute error (MAD), coefficient of determination (R^2), relative root mean squared error (RRMSE), and symmetric mean absolute percentage error (sMAPE). The definition of the performance metrics were based on the following definitions (Equations (9)–(12)), where: Y corresponds to the ground-truth values, $\hat{Y}$ is the predicted values, $\overline{Y}$ is the mean values of Y, Y_i represents each particular ground-truth value and $\hat{Y}_i$ represents each particular predicted value, and N is the number of samples:

$$MSE = Mean\left(\left(Y - \hat{Y}\right)^2\right);\ MAE = Mean\left(\left|Y - \hat{Y}\right|\right);\ MAD = Median\left(\left|Y - \hat{Y}\right|\right) \quad (9)$$

$$RRMSE = \frac{\sqrt{\sum_{i=0}^{N}\left(Y_i - \hat{Y}_i\right)^2}}{\sqrt{\sum_{i=0}^{N}\left(Y_i\right)^2}} \quad (10)$$

$$R^2 = 1 - \frac{\sum_{i=0}^{N}\left(Y_i - \hat{Y}_i\right)^2}{\sum_{i=0}^{N}\left(Y_i - \overline{Y}\right)^2} \quad (11)$$

$$sMAPE = \frac{100}{N}\sum_{i=0}^{N} 2\frac{\left|Y_i - \hat{Y}_i\right|}{|Y_i| + \left|\hat{Y}_i\right|}\ \% \quad (12)$$

All metrics have values greater than zero with no upper limit, except R^2 that has an upper limit of 1 with no lower limit, and sMAPE which has an upper limit of 200%. In all cases, the smaller the value, the better the result, except the R^2 metric, where the relationship is the opposite. R^2 provides an indication of the variance explained by the model. A value of 1 corresponds to a perfect fit of the model to the real data, a value of zero indicates a dummy prediction always using the mean value, and a negative value a worse prediction than always choosing the mean. The other metrics (MSE, MAE, MAD, sMAPE, and RRMSE) are error metrics, and they are always positive, with the best result for a value of zero. We considered especially important the metrics R^2, RRMSE and SMAPE, since they are metrics representing a ratio between the error and the actual values.

All results provided in this section were obtained with the test set presented in Section 3.1. The results are given after inverting the scale (in the range [0,1]) performed for training. Therefore, the results are based on unscaled load values, which can be especially important for metrics based on actual values and not on ratios (e.g., MSE, MAE, MAD). We provide an extensive analysis of results for different values of the parameters: k (number of forecast values), p (number of predictor time-slots), and f (length of features used as predictors).

Figures 6–8 show the forecast metrics for the different models (Section 3.2), and for different values of the parameters f, p , and k. They present the data in two sections, with the upper section presenting the raw data with a color-code format, and the lower section showing a subset of the data in a bar chart format. The bar charts below the tables present the sMAPE metrics (average, T-0 and T-23) for some of the best performing models in the corresponding table. These tables present the metrics using a color code to easily show where the best values are. Color coding uses a color palette where the greenest color

is for the best result and the reddest is for the worst. Color coding was applied column-wise to compare results for the same metric. Figure 6 presents all the results for the Seq2seq and Seq2seq + attention models. Figures 7 and 8 present the results for the rest of the models, Figure 7 for a f value equal to 45, and Figure 8 for an f value equal to 1. Metric values are provided for the forecasts of the first (T-0) and the last (T-23) time-slots, in addition to the average of forecasts for all (24) predicted time-slots. The last two columns provide the training and test times for all models.

				Average						T-0						T-23							
Model	f	p	Model	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	Training Time (min)	Test Time (min)
Seq2Seq	45	24	1 LSTM + 1 FC	6,479,118	1770.5	1261.5	0.801	8.959	0.115	434,997	476.9	353.3	0.987	2.435	0.031	10,543,200	2436.6	1810.2	0.676	12.219	0.152	14.369	0.048
	45	24	2 LSTM + 1 FC	8,289,029	1747.5	969.9	0.745	8.691	0.129	451,045	479.0	352.3	0.986	2.451	0.032	13,433,500	2427.8	1372.3	0.587	11.807	0.172	63.148	0.071
	45	24	2 1D-CNN + 1 LSTM + 1 FC	47,089,200	5315.6	4358.8	-0.448	26.988	0.315	2,358,580	1182.3	940.3	0.927	6.416	0.072	53,410,500	5806.8	4783.1	-0.642	29.515	0.343	16.880	0.068
	45	168	1 LSTM + 1 FC	8,420,570	1879.8	1174.8	0.742	9.199	0.129	434,666	477.9	356.7	0.987	2.438	0.031	14,577,200	2672.0	1687.9	0.553	12.898	0.179	82.606	0.076
	45	168	2 LSTM + 1 FC	6,788,302	1748.8	1176.9	0.792	8.801	0.117	419,015	477.5	371.4	0.987	2.434	0.030	11,131,400	2377.3	1613.9	0.658	11.749	0.157	84.025	0.134
	45	168	2 1D-CNN + 1 LSTM + 1 FC	29,722,340	4474.9	4106.6	0.088	22.049	0.251	3,050,370	1326.3	1034.1	0.906	6.348	0.082	33,555,800	4903.7	4572.9	-0.030	24.233	0.272	35.034	0.102
	45	720	1 LSTM + 1 FC	8,534,025	1890.2	1154.7	0.739	9.360	0.131	462,248	489.2	359.5	0.986	2.489	0.032	14,063,700	2592.6	1628.2	0.570	12.635	0.176	100.725	0.229
	45	720	2 LSTM + 1 FC	6,366,990	1667.3	1077.9	0.805	8.392	0.114	446,108	491.5	375.3	0.986	2.533	0.031	9,796,230	2190.8	1422.8	0.700	10.804	0.147	160.340	0.368
	45	720	2 1D-CNN + 1 LSTM + 1 FC	30,051,510	4128.2	3208.7	0.080	20.318	0.255	8,307,780	2268.5	1907.6	0.746	10.902	0.135	32,701,000	4385.3	3466.6	-0.001	21.663	0.268	115.625	0.297
Seq2Seq + Attention	45	24	1 LSTM + 1 FC	8,384,991	1854.4	1172.6	0.742	9.083	0.130	488,121	503.9	374.0	0.985	2.599	0.033	13,955,300	2544.9	1620.0	0.571	12.321	0.175	13.582	0.067
	45	24	2 LSTM + 1 FC	7,578,606	1698.7	987.7	0.767	8.367	0.123	416,383	464.7	351.1	0.987	2.370	0.030	12,384,800	2373.5	1409.8	0.619	11.478	0.165	17.083	0.067
	45	24	2 1D-CNN + 1 LSTM + 1 FC	159,237,700	11174.2	10988.3	-3.896	70.258	0.592	143,198,000	10482.3	10183.4	-3.402	64.212	0.562	162,892,000	11335.8	11235.5	-4.008	71.692	0.599	12.383	0.067
	45	168	1 LSTM + 1 FC	9,017,546	1936.1	1214.5	0.723	9.651	0.135	528,351	525.2	388.1	0.984	2.715	0.034	13,566,200	2555.7	1646.6	0.584	12.650	0.173	24.848	0.087
	45	168	2 LSTM + 1 FC	6,999,635	1698.1	1045.9	0.785	8.422	0.119	445,945	485.9	360.4	0.986	2.485	0.031	10,888,600	2258.5	1409.4	0.666	11.113	0.155	87.210	0.123
	45	168	2 1D-CNN + 1 LSTM + 1 FC	39,696,310	4883.9	4246.0	-0.218	25.720	0.286	4,081,380	1499.9	1128.3	0.875	7.504	0.095	62,598,700	6520.2	6020.2	-0.921	35.150	0.372	49.909	0.102
	45	720	1 LSTM + 1 FC	16,830,620	2744.8	1849.3	0.485	12.680	0.180	484,231	503.7	371.9	0.985	2.554	0.033	29,492,600	3963.0	2761.7	0.098	17.796	0.255	184.945	0.232
	45	720	2 LSTM + 1 FC	9,215,292	2022.9	1331.0	0.718	9.991	0.135	450,344	496.0	379.3	0.986	2.537	0.031	15,789,300	2842.9	1856.9	0.517	13.798	0.186	167.660	0.334
	45	720	2 1D-CNN + 1 LSTM + 1 FC	26,543,160	3987.9	3370.7	0.188	19.586	0.239	8,106,980	2239.7	1862.6	0.752	11.152	0.134	30,918,400	4356.0	3724.7	0.054	21.361	0.261	109.441	0.248

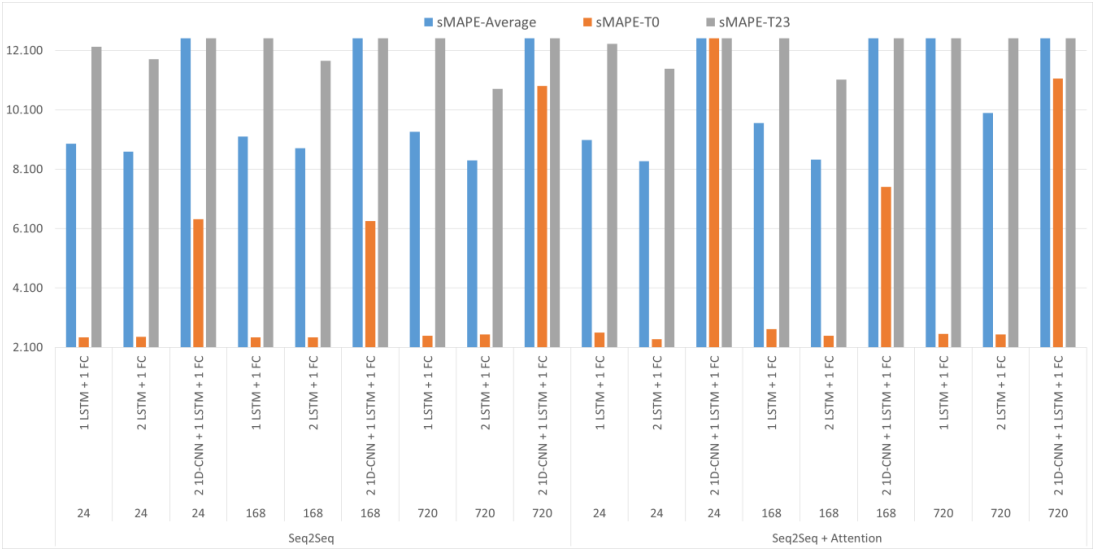

Figure 6. (**Upper section**): forecast metrics for the average, first (T-0), and last (T-23) predicted time-slots using Seq2seq with and without an attention mechanism. The table is color-coded along columns with colors between red and green corresponding to the worst and best results, respectively. (**Lower section**): bar chart for sMAPE metrics for some of the best performing models from the upper table.

				Average						T-0						T-23							
Model	f	p	Model	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	Training Time (min)	Test Time (min)
DL Architectures	45	24	1 LSTM + 1 FC	6,195,447	1698.4	1170.8	0.810	8.506	0.115	1,494,960	937.8	722.7	0.954	4.843	0.057	9,810,780	2117.0	1435.5	0.698	10.555	0.147	4.765	0.008
	45	24	2 LSTM + 1 FC	7,748,237	1802.8	1121.3	0.762	9.056	0.128	1,528,600	913.9	693.6	0.953	4.713	0.058	12,091,800	2310.3	1419.4	0.628	11.388	0.163	19.256	0.015
	45	24	2 2D-CNN + 1 FC	33,094,060	4882.9	4494.4	-0.018	24.165	0.270	33,039,300	4880.1	4482.6	-0.016	24.150	0.270	33,275,800	4891.8	4506.8	-0.023	24.196	0.271	7.766	0.023
	45	24	2 1D-CNN + 1 FC	5,907,817	1743.4	1284.2	0.818	8.846	0.113	2,394,930	1208.6	993.5	0.926	6.393	0.073	8,622,150	2095.9	1485.9	0.735	10.508	0.138	3.201	0.004
	45	24	2 1D-CNN (MaxPooling) + 1 FC	6,126,764	1792.7	1336.6	0.812	9.130	0.115	2,658,230	1263.9	1012.1	0.918	6.580	0.077	8,993,910	2172.0	1568.9	0.724	10.954	0.141	3.788	0.004
	45	24	2 1D-CNN + 1 LSTM + 1 FC	4,975,367	1507.8	1016.4	0.847	7.627	0.103	1,564,870	928.8	715.4	0.952	4.813	0.059	7,307,580	1830.8	1187.4	0.775	9.227	0.127	19.558	0.007
	45	24	2 1D-CNN + 2 LSTM + 1 FC	5,738,281	1629.7	1087.6	0.824	8.264	0.111	2,041,510	1086.1	856.6	0.937	5.610	0.067	8,104,580	1970.8	1330.0	0.751	9.959	0.134	12.020	0.015
	45	168	1 LSTM + 1 FC	6,974,843	1766.6	1170.7	0.786	8.856	0.122	2,428,550	1195.6	962.4	0.925	6.263	0.073	9,739,580	2082.1	1305.0	0.701	10.384	0.147	18.308	0.046
	45	168	2 LSTM + 1 FC	7,798,327	1833.4	1166.8	0.761	9.153	0.129	2,802,340	1228.6	941.5	0.914	6.341	0.079	10,425,300	2170.0	1400.7	0.680	10.934	0.152	31.858	0.067
	45	168	2 1D-CNN + 1 FC	11,667,630	2662.6	2181.5	0.642	13.216	0.160	9,972,840	2524.9	2106.5	0.694	12.685	0.148	13,048,100	2794.2	2236.9	0.600	13.824	0.170	1.921	0.013
	45	168	2 1D-CNN (MaxPooling) + 1 FC	16,910,420	3278.5	2812.9	0.481	16.236	0.192	20,405,200	3713.7	3217.1	0.374	18.473	0.212	24,157,800	4056.4	3695.9	0.259	20.018	0.231	4.178	0.016
	45	168	2 1D-CNN + 1 LSTM + 1 FC	5,372,823	1487.8	995.2	0.835	7.527	0.107	1,576,770	951.4	744.6	0.952	4.931	0.059	8,369,170	1832.4	1158.3	0.743	9.245	0.136	8.235	0.018
	45	168	2 1D-CNN + 2 LSTM + 1 FC	4,189,875	1332.0	882.4	0.871	6.677	0.094	1,247,890	837.3	645.4	0.962	4.371	0.052	5,960,730	1596.7	1009.0	0.817	7.936	0.115	15.313	0.022
	45	720	1 LSTM + 1 FC	6,090,553	1599.7	1081.7	0.814	8.054	0.114	1,894,670	1017.9	776.7	0.942	5.275	0.065	8,479,800	1843.8	1179.6	0.741	9.115	0.137	56.410	0.221
	45	720	2 LSTM + 1 FC	7,241,965	1745.5	1130.5	0.778	8.668	0.124	2,611,450	1191.2	911.1	0.920	6.037	0.076	9,902,940	2008.5	1205.5	0.697	9.879	0.148	69.946	0.370
	45	720	2 1D-CNN + 1 FC	33,224,320	4884.9	4485.8	-0.017	24.145	0.270	33,419,500	4895.9	4503.5	-0.023	24.193	0.271	33,571,700	4903.6	4500.3	-0.027	24.221	0.272	8.410	0.045
	45	720	2 1D-CNN (MaxPooling) + 1 FC	33,134,430	4880.0	4485.1	-0.014	24.125	0.270	33,276,900	4888.4	4490.7	-0.018	24.162	0.270	33,390,600	4894.1	4508.7	-0.022	24.182	0.271	15.823	0.050
	45	720	2 1D-CNN + 1 LSTM + 1 FC	5,059,471	1496.8	1014.4	0.845	7.493	0.104	1,645,400	959.7	743.7	0.950	5.063	0.060	6,419,050	1721.4	1157.2	0.804	8.563	0.119	21.860	0.067
	45	720	3 1D-CNN + 1 LSTM + 1 FC	5,661,417	1612.8	1116.3	0.827	8.066	0.111	2,411,320	1137.5	849.1	0.926	5.880	0.073	7,315,740	1854.9	1268.3	0.776	9.242	0.127	17.143	0.090
	45	720	2 1D-CNN + 2 LSTM + 1 FC	6,291,731	1746.6	1206.9	0.807	8.619	0.117	3,990,250	1471.3	1093.8	0.878	7.417	0.094	7,100,180	1837.5	1236.7	0.783	9.080	0.125	23.993	0.062
Classic ML	45	24	Random Forest	5,463,283	1470.4	857.2	0.832	7.414	0.107	633,968	556.8	398.2	0.981	2.807	0.037	7,913,630	1803.0	1046.0	0.757	9.057	0.132	9.488	0.105
	45	24	Linear Regression	6,314,323	1753.8	1260.5	0.806	8.862	0.115	688,331	588.0	415.5	0.979	2.938	0.039	8,622,780	2124.0	1502.5	0.735	10.664	0.138	0.902	0.001
	45	24	Gradient Boosting	19,925,480	3714.6	3301.1	0.387	18.849	0.209	14,067,000	3155.2	2888.6	0.568	16.218	0.176	20,416,300	3759.5	3307.8	0.372	19.056	0.212	3.271	0.009
	45	24	KNN	13,130,560	2576.7	1652.0	0.596	12.658	0.170	11,975,500	2422.6	1494.0	0.632	11.901	0.162	14,493,100	2734.1	1824.0	0.554	13.401	0.179	1.476	18.022
	45	24	Support Vector Regression	7,051,829	2011.4	1617.9	0.783	10.180	0.123	2,608,470	1314.6	1146.2	0.920	6.691	0.076	9,420,250	2361.8	1922.7	0.710	11.911	0.144	38.132	8.682
	45	24	AdaBoost	11,344,240	2665.9	2276.3	0.651	13.636	0.157	2,720,780	1338.4	1158.9	0.916	6.975	0.077	12,510,100	2773.4	2257.8	0.615	14.152	0.166	6.649	0.067
	45	168	Random Forest	4,959,729	1351.4	788.8	0.848	6.717	0.103	747,048	614.5	443.3	0.977	3.105	0.041	5,707,890	1472.8	843.3	0.825	7.322	0.112	92.524	1.028
	45	168	Linear Regression	4,758,965	1379.6	892.7	0.854	6.941	0.100	406,341	442.2	321.5	0.988	2.261	0.030	6,361,340	1625.5	1041.7	0.805	8.138	0.118	73.650	0.008
	45	168	Gradient Boosting	16,874,050	3388.8	3016.4	0.482	17.295	0.193	13,894,000	3108.7	2870.8	0.574	16.009	0.175	17,038,100	3405.4	3041.5	0.477	17.365	0.194	44.733	0.069
	45	168	KNN	12,102,020	2544.2	1748.8	0.629	12.387	0.163	11,877,000	2509.1	1716.0	0.636	12.214	0.162	12,215,900	2558.9	1760.0	0.625	12.463	0.164	7.530	274.878
	45	168	Support Vector Regression	6,325,536	1890.3	1487.0	0.806	9.536	0.118	4,085,740	1558.7	1249.2	0.875	7.934	0.095	7,465,730	2087.5	1678.4	0.771	10.520	0.128	179.090	42.726
	45	168	AdaBoost	8,299,474	2283.8	1963.3	0.745	11.802	0.134	2,030,910	1133.3	947.5	0.938	5.856	0.067	9,607,530	2459.9	2061.6	0.705	12.743	0.146	55.554	0.478
Ensemble Type I	45	24	(3 FC)*5	7,690,465	1914.9	1321.1	0.764	9.581	0.128	2,235,200	1128.2	891.2	0.931	5.877	0.070	10,874,800	2359.8	1689.7	0.666	11.865	0.155	1.504	0.003
	45	24	(4 FC)*5	8,469,669	1959.6	1258.3	0.740	9.953	0.134	2,134,390	1086.2	841.6	0.934	5.688	0.069	11,740,500	2353.4	1436.9	0.639	11.818	0.161	2.896	0.003
	45	24	(1 LSTM + 1 FC)*10	6,974,140	1701.1	1039.4	0.786	8.600	0.120	723,268	635.2	499.1	0.978	3.278	0.040	10,420,500	2131.0	1256.6	0.680	10.642	0.152	57.917	0.049
	45	24	(2 LSTM + 1 FC)*3	7,176,459	1815.6	1190.7	0.779	9.174	0.123	1,565,160	982.0	818.2	0.952	5.124	0.059	11,087,600	2279.0	1417.5	0.659	11.457	0.156	16.675	0.025
	45	168	(3 FC)*5	6,195,977	1768.8	1290.1	0.810	8.879	0.116	3,703,070	1465.4	1159.3	0.886	7.582	0.090	7,909,510	2052.0	1543.7	0.757	10.291	0.132	13.312	0.012
	45	168	(4 FC)*5	6,660,185	1778.6	1248.9	0.796	8.886	0.120	3,321,780	1345.0	1046.3	0.898	6.857	0.086	8,342,990	2077.2	1492.1	0.744	10.386	0.136	11.200	0.009
	45	168	(1 LSTM + 1 FC)*5	6,689,661	1752.0	1183.0	0.795	8.895	0.119	1,498,140	931.6	728.0	0.954	4.823	0.057	9,546,250	2111.4	1403.9	0.707	10.618	0.145	50.960	0.146
	45	168	(2 LSTM + 1 FC)*3	5,669,071	1552.3	1011.5	0.826	7.780	0.110	1,610,220	960.9	763.7	0.951	4.910	0.060	7,639,600	1846.9	1175.7	0.766	9.246	0.130	49.340	0.146
Ensemble Type I-Weights Sharing	45	24	(3 FC)*5	8,475,861	2196.9	1715.3	0.739	11.094	0.136	5,611,620	1773.3	1354.7	0.827	9.121	0.111	10,887,700	2502.1	1931.9	0.665	12.496	0.155	1.452	0.003
	45	24	(4 FC)*5	8,526,247	2103.0	1552.9	0.738	10.558	0.135	3,404,970	1414.3	1154.4	0.895	7.453	0.087	12,056,400	2485.3	1767.0	0.629	12.378	0.163	1.185	0.003
	45	168	(3 FC)*5	33,547,800	4906.2	4521.8	-0.030	24.272	0.272	33,964,500	4928.3	4540.0	-0.042	24.361	0.274	33,623,700	4909.6	4523.8	-0.032	24.281	0.272	2.432	0.009
	45	168	(4 FC)*5	7,619,318	2063.7	1581.3	0.766	10.344	0.129	6,430,420	1881.8	1418.7	0.803	9.487	0.119	10,722,500	2469.2	1871.0	0.671	12.453	0.154	4.301	0.009
Ensemble Type II	45	24	(1 LSTM + 2 FC)*2 + (4 FC)*2	6,663,891	1700.8	1083.0	0.795	8.660	0.118	954,288	728.7	562.9	0.971	3.779	0.046	10,798,700	2155.7	1252.4	0.668	10.744	0.154	14.514	0.013
	45	168	(1 LSTM + 2 FC)*2 + (4 FC)*2	4,838,068	1461.5	961.9	0.852	7.338	0.102	1,164,250	801.2	611.4	0.964	4.127	0.051	6,988,970	1787.9	1168.9	0.786	8.896	0.124	72.698	0.059

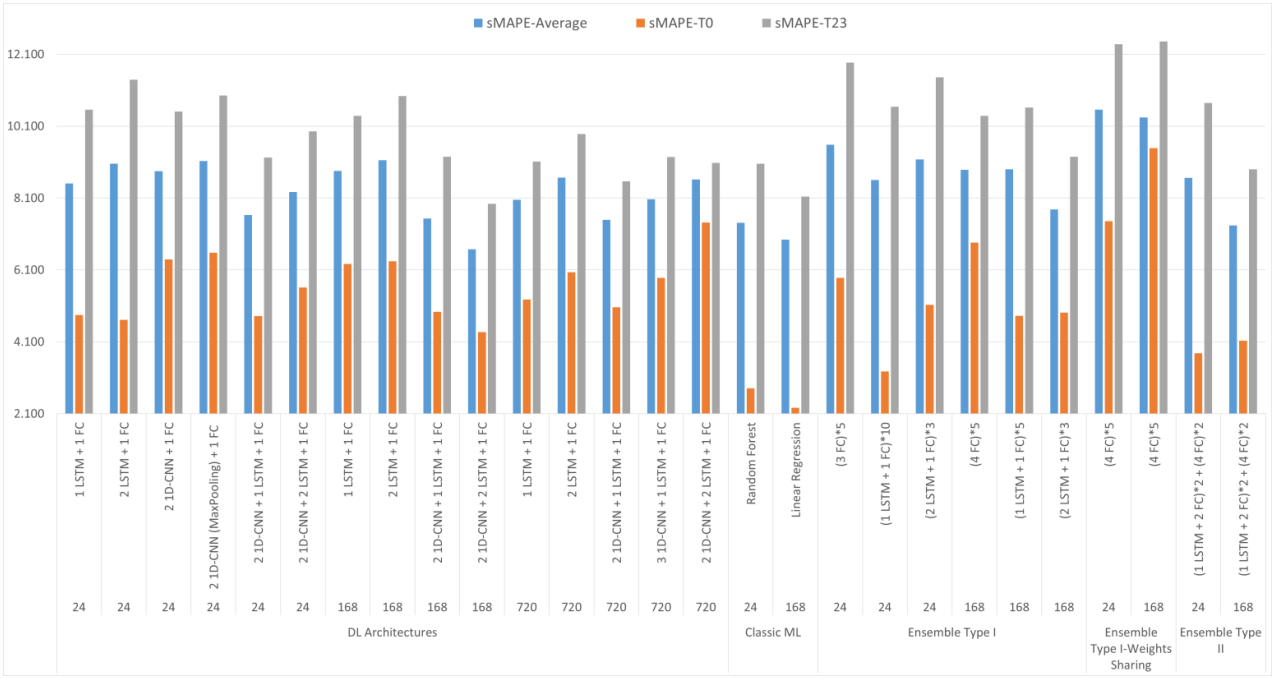

Figure 7. (**Upper section**): forecast metrics for the average, first (T-0), and last (T-23) predicted time-slots using DL, ML, and ensemble models for a fixed number of values per predictor (f = 45). The table is color-coded along columns with colors between red and green corresponding to the worst and best results, respectively. (**Lower section**): bar chart for sMAPE metrics for some of the best performing models from the upper table.

				Average						T-0						T-23							
Class	f	p	Model	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	Training Time (min)	Test Time (min)
DL Achitectures	1	24	1 LSTM + 1 FC	6,022,328	1731	1232	0.8148	8.766	0.112	1,228,310	842	659	0.9622	4.357	0.052	9,384,450	2203	1578	0.7115	11.066	0.144	3.812	0.0091
	1	24	2 LSTM + 1 FC	5,675,340	1694	1219	0.8255	8.542	0.109	1,152,420	815	633	0.9646	4.179	0.050	8,421,500	2110	1508	0.7411	10.673	0.136	4.826	0.0161
	1	24	2 1D-CNN + 1 FC	5,975,028	1734	1272	0.8163	8.837	0.112	1,252,970	878	716	0.9615	4.634	0.053	10,303,400	2260	1487	0.6833	11.404	0.151	1.684	0.0052
	1	24	2 1D-CNN (MaxPooling) + 1 FC	6,924,590	1882	1376	0.7871	9.555	0.120	1,572,820	984	811	0.9516	5.091	0.059	10,800,700	2438	1813	0.6680	12.277	0.154	1.564	0.0058
	1	24	2 1D-CNN + 1 LSTM + 1 FC	5,923,070	1724	1252	0.8179	8.711	0.112	1,340,070	893	712	0.9588	4.588	0.054	9,072,510	2165	1472	0.7211	10.976	0.141	1.805	0.0121
	1	24	2 1D-CNN + 2 LSTM + 1 FC	5,587,690	1660	1181	0.8282	8.337	0.108	1,284,620	894	739	0.9605	4.509	0.053	8,512,330	2096	1464	0.7383	10.556	0.137	3.646	0.0176
	1	168	1 LSTM + 1 FC	3,296,706	1224	867	0.8988	6.151	0.084	1,122,670	823	673	0.9656	4.155	0.050	4,244,780	1373	923	0.8697	6.961	0.097	122.766	0.0431
	1	168	2 LSTM + 1 FC	3,595,281	1265	890	0.8897	6.368	0.088	1,578,830	954	751	0.9516	4.860	0.059	4,316,120	1339	880	0.8675	6.691	0.098	52.734	0.0852
	1	168	2 1D-CNN + 1 FC	7,223,554	2067	1739	0.7783	10.411	0.124	8,383,650	2405	2200	0.7428	12.025	0.136	12,656,700	2909	2623	0.6116	14.587	0.167	4.088	0.0107
	1	168	2 1D-CNN (MaxPooling) + 1 FC	5,344,936	1653	1256	0.8360	8.396	0.107	4,393,440	1626	1330	0.8652	8.313	0.098	8,322,360	2172	1739	0.7446	10.991	0.135	2.336	0.0129
	1	168	2 1D-CNN + 1 LSTM + 1 FC	3,950,512	1372	998	0.8788	6.938	0.092	1,334,120	898	742	0.9591	4.629	0.054	5,206,950	1543	1040	0.8402	7.788	0.107	4.332	0.0153
	1	168	2 1D-CNN + 2 LSTM + 1 FC	3,383,631	1274	912	0.8962	6.424	0.085	1,191,380	844	684	0.9634	4.334	0.051	5,022,610	1537	1050	0.8459	7.682	0.105	8.075	0.0213
	1	720	1 LSTM + 1 FC	4,788,052	1593	1199	0.8535	8.013	0.102	2,197,550	1160	956	0.9328	6.017	0.070	6,004,430	1749	1267	0.8163	8.755	0.115	34.154	0.2102
	1	720	2 LSTM + 1 FC	5,541,973	1732	1319	0.8304	8.810	0.110	3,487,550	1444	1171	0.8933	7.439	0.088	7,219,800	2021	1601	0.7791	10.355	0.126	35.682	0.3469
	1	720	2 1D-CNN + 1 FC	19,444,790	3654	3256	0.4049	18.106	0.206	21,130,200	3885	3484	0.3534	19.295	0.216	22,165,200	3934	3504	0.3218	19.486	0.221	6.670	0.0401
	1	720	2 1D-CNN (MaxPooling) + 1 FC	33,198,780	4884	4484	-0.0161	24.139	0.270	33,122,000	4880	4475	-0.0136	24.129	0.270	33,117,800	4880	4470	-0.0133	24.123	0.270	5.667	0.0291
	1	720	2 1D-CNN + 1 LSTM + 1 FC	2,982,062	1171	837	0.9087	5.873	0.080	986,983	781	656	0.9698	4.060	0.047	4,328,650	1382	947	0.8676	6.874	0.098	9.040	0.0338
	1	720	3 1D-CNN + 1 LSTM + 1 FC	3,340,233	1233	878	0.8978	6.236	0.084	1,203,010	825	650	0.9632	4.196	0.051	4,746,470	1460	966	0.8548	7.359	0.102	16.328	0.0461
	1	720	2 1D-CNN + 2 LSTM + 1 FC	3,316,090	1239	884	0.8985	6.148	0.084	1,268,230	866	693	0.9612	4.429	0.053	4,712,460	1474	1022	0.8558	7.240	0.102	17.393	0.0391
Classic ML	1	24	Random Forest	5,302,593	1479	889	0.8370	7.517	0.105	568,900	534	380	0.9825	2.720	0.035	8,331,450	1954	1213	0.7439	9.851	0.135	1.228	0.0136
	1	24	Linear Regression	10,533,310	2266	1547	0.6761	11.594	0.148	902,373	693	497	0.9723	3.465	0.045	13,948,600	2732	1865	0.5712	13.950	0.175	0.007	0.0001
	1	168	Random Forest	5,020,732	1349	780	0.8459	6.704	0.104	757,942	619	454	0.9767	3.108	0.041	5,815,730	1481	847	0.8215	7.363	0.113	9.665	0.1074
	1	168	Linear Regression	4,240,085	1297	827	0.8699	6.436	0.094	350,885	431	328	0.9892	2.212	0.028	5,533,440	1495	915	0.8302	7.402	0.110	0.050	0.0006
DMD	1	24	NA	18,253,160	3234	2612	0.4409	16.414	0.199	7,035,700	2021	1516	0.7845	10.128	0.125	22,294,500	3647	3024	0.3171	18.565	0.222	0.005	0.0695
	1	168	NA	11,280,890	2580	2062	0.6542	13.301	0.158	8,531,820	2299	1918	0.7388	11.931	0.138	13,465,600	2810	2272	0.5869	14.338	0.173	0.053	0.5600
	1	720	NA	8,297,923	2210	1804	0.7174	11.425	0.140	7,232,500	2077	1698	0.7520	10.755	0.131	8,731,240	2262	1838	0.7064	11.696	0.143	0.065	0.8642
Ensemble Type I	1	24	(3 FC)*5	4,911,157	1524	1062	0.8490	7.713	0.101	868,293	730	612	0.9733	3.831	0.044	8,012,950	1971	1313	0.7537	9.912	0.133	1.820	0.0019
	1	24	(4 FC)*5	4,710,536	1480	1012	0.8552	7.497	0.099	889,754	740	607	0.9726	3.873	0.044	7,636,190	1970	1374	0.7652	9.845	0.130	2.278	0.0022
	1	24	(1 LSTM + 1 FC)*10	5,256,786	1608	1166	0.8384	8.228	0.104	663,536	620	482	0.9796	3.207	0.038	8,867,330	2212	1657	0.7274	11.255	0.140	58.649	0.0568
	1	24	(2 LSTM + 1 FC)*3	5,312,667	1587	1110	0.8367	7.979	0.105	927,148	738	591	0.9715	3.812	0.045	8,037,210	2067	1487	0.7529	10.323	0.133	21.165	0.0226
	1	168	(3 FC)*5	3,161,967	1148	788	0.9030	5.741	0.082	695,745	636	507	0.9787	3.245	0.039	4,338,160	1343	904	0.8669	6.752	0.098	2.413	0.0033
	1	168	(4 FC)*5	3,256,741	1152	771	0.9001	5.724	0.083	790,755	669	524	0.9757	3.449	0.042	4,472,430	1353	890	0.8627	6.784	0.099	1.733	0.0026
	1	168	(1 LSTM + 1 FC)*5	3,615,454	1258	868	0.8890	6.329	0.088	760,056	679	569	0.9767	3.478	0.041	4,858,600	1395	867	0.8509	6.876	0.104	57.717	0.1239
	1	168	(2 LSTM + 1 FC)*3	3,750,715	1343	960	0.8849	6.812	0.090	1,022,650	789	631	0.9686	3.973	0.047	5,022,560	1511	982	0.8459	7.584	0.105	22.735	0.1317
Ensemble Type I-Weights Sharing	1	24	(3 FC)*5	6,025,247	1791	1351	0.8147	9.065	0.112	1,203,380	858	718	0.9630	4.454	0.051	8,690,810	2164	1575	0.7328	10.889	0.138	1.107	0.0017
	1	24	(4 FC)*5	5,670,401	1681	1220	0.8257	8.511	0.109	1,099,780	808	648	0.9662	4.191	0.049	8,551,730	2046	1377	0.7371	10.281	0.137	2.021	0.0017
	1	168	(3 FC)*5	4,354,109	1425	1026	0.8664	7.233	0.097	3,156,330	1347	1044	0.9032	7.115	0.083	6,977,510	1922	1441	0.7859	9.649	0.124	1.121	0.0022
	1	168	(4 FC)*5	3,844,067	1323	942	0.8820	6.560	0.091	1,267,590	852	669	0.9611	4.389	0.053	5,385,760	1503	999	0.8347	7.439	0.109	3.031	0.0033
Ensemble Type II	1	24	(1 LSTM + 2 FC)*2 + (4 FC)*2	5,280,710	1551	1034	0.8376	7.827	0.105	830,195	723	616	0.9745	3.685	0.043	8,266,720	1997	1300	0.7459	10.044	0.135	13.403	0.0131
	1	168	(1 LSTM + 2 FC)*2 + (4 FC)*2	3,064,396	1127	773	0.9060	5.693	0.080	604,466	587	458	0.9815	2.988	0.037	4,364,300	1327	861	0.8661	6.666	0.098	95.918	0.0986

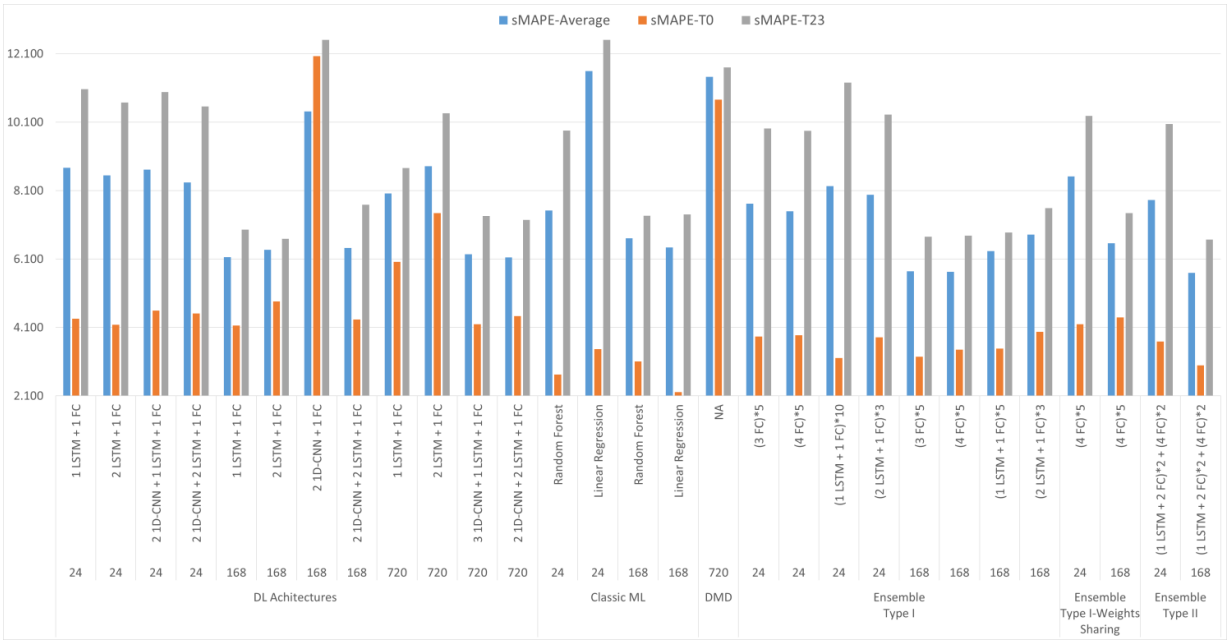

Figure 8. (**Upper section**): forecast metrics for the average, first (T-0), and last (T-23) predicted time-slots using DL, ML, and ensemble models for a fixed number of values per predictor (f = 1). The table is color-coded along columns with colors between red and green corresponding to the worst and best results, respectively. (**Lower section**): bar chart for sMAPE metrics for some of the best performing models from the upper table.

The best results for the Seq2seq models were achieved when using only recurring networks (one or two LSTM layers) while the inclusion of convolutional layers deteriorates the forecast. Seq2seq models (Figures 6 and A1 (Appendix A) present some of the best results for very short-term forecasts and worst for average and long-term forecasts. These models have difficulty converging with f, equal to one with their best results with f equal to 45.

The results for DL, ML, and ensemble models, with an f value equal to 45, are shown in Figure 7 and the Appendix A. For this value of parameter f, the best performing models

were classic ML models. A combination of two CNN layers and two LSTM appeared to provide a performance similar to random forest and linear regression. The ensemble models did not give a good performance either.

The results for DL, ML, and ensemble models, with an f value equal to 1, are shown in Figure 8 and the Appendix A. The results in this figure are very different from the results in Figure 7, despite being the same models with a different value for parameter f. In Figure 8, we obtain most of the best results obtained in this work. The best performing model types were DL and ensemble models for long-term forecasting and average results, while for short-term results, the classic ML and Seq2seq models were best. It is interesting how the combination of CNN and LSTM layers produced some of the best performance models, since this result is very different from what happens in Seq2seq architectures, for which these layer combinations provide very poor results.

A conclusion drawn from the results in Figures 6–8 is the importance of the impact of the rolling window length and the number of features associated with the time-slots. Another is that the best specific sequence of layers for a deep learning architecture depends on the type of model, making it impossible to establish a priori a better architecture for all types of models.

In all the tables in Figures 6–8, the description of the models includes the number and type of layers used: CNN, LSTM, and fully connected layers (FC), forming a sequence separated by the + sign. The ensemble models follow the gaNet architecture [11] which is formed by repeating blocks where the configuration of each repeating block is included in parentheses with an asterisk and a number to the right of the parentheses that indicates the number of repetitions of the block. Ensemble Type II models that are made up of different blocks with a possibly different architecture per block can also have different types of inputs per block; in those cases where there are different inputs per block, we have marked that in the tables by an NA in the f column, reporting the type of input associated with each block by an additional letter within parentheses and immediately after the layer type (LSTM or 1D-CNN). Possible values for these additional letters are: an (L) for inputs formed exclusively by the load values, a (D) for inputs formed by the load plus the date/time features, and a (A) for inputs formed by the load plus the date/time and weather features.

Table 1 provides a summary table with a ranking of the three best models for the three most important metrics (R^2, RRMSE, and SMAPE) plus training and test times. Similar to Figures 6–8, metric values are provided for the forecasts of the first (T-0) and the last (T-23) time-slots, in addition to the average of forecasts for all (24) predicted time-slots.

Table 1. Ranking of the three best models for the three most important metrics (R^2, SMAPE and RRMSE) plus training and test times. Values are provided for the forecasts of the first (T-0) and the last (T-23) time-slots, in addition to the average of forecasts for all (24) predicted time-slots.

	Metric	Rank	Class	f	p	k	Model	Metric Value
Average		1°	DL Achitectures	1	720	24	2 1D-CNN + 1 LSTM + 1 FC	0.9087
	R^2	2°	Ensemble Type II	1	168	24	(1 LSTM + 2 FC) * 2 + (4 FC) * 2	0.9060
		3°	Ensemble Type I	1	168	24	(3 FC) * 5	0.9030
	sMAPE	1°	Ensemble Type II	1	168	24	(1 LSTM + 2 FC) * 2 + (4 FC) * 2	5.6934
		2°	Ensemble Type I	1	168	24	(4 FC) * 5	5.7235
		3°	Ensemble Type I	1	168	24	(3 FC) * 5	5.7410
	RRMSE	1°	DL Achitectures	1	720	24	2 1D-CNN + 1 LSTM + 1 FC	0.0797
		2°	Ensemble Type II	1	168	24	(1 LSTM + 2 FC) * 2 + (4 FC) * 2	0.0803
		3°	Ensemble Type I	1	168	24	(3 FC) * 5	0.0820
T-0	R^2	1°	Classic ML	1	168	24	Linear Regression	0.9892
		2°	Classic ML	45	168	24	Linear Regression	0.9875
		3°	Seq2Seq + Attention	45	24	24	2 LSTM + 1 FC	0.9872
	sMAPE	1°	Classic ML	1	168	24	Linear Regression	2.2120
		2°	Classic ML	45	168	24	Linear Regression	2.2608
		3°	Seq2Seq + Attention	45	24	24	2 LSTM + 1 FC	2.3705
	RRMSE	1°	Classic ML	1	168	24	Linear Regression	0.0278
		2°	Classic ML	45	168	24	Linear Regression	0.0299
		3°	Seq2Seq + Attention	45	24	24	2 LSTM + 1 FC	0.0303
T-23	R^2	1°	DL Achitectures	1	168	24	1 LSTM + 1 FC	0.8697
		2°	DL Achitectures	1	720	24	2 1D-CNN + 1 LSTM + 1 FC	0.8676
		3°	DL Achitectures	1	168	24	2 LSTM + 1 FC	0.8675
	sMAPE	1°	Ensemble Type II	1	168	24	(1 LSTM + 2 FC) * 2 + (4 FC) * 2	6.6660
		2°	DL Achitectures	1	168	24	2 LSTM + 1 FC	6.6912
		3°	Ensemble Type I	1	168	24	(3 FC) * 5	6.7516
	RRMSE	1°	DL Achitectures	1	168	24	1 LSTM + 1 FC	0.0967
		2°	DL Achitectures	1	720	24	2 1D-CNN + 1 LSTM + 1 FC	0.0975
		3°	DL Achitectures	1	168	24	2 LSTM + 1 FC	0.0976
	Training Time (min)	1°	DMD	1	24	24	NA	0.0047
		2°	Classic ML	1	24	24	Linear Regression	0.0066
		3°	Classic ML	1	168	24	Linear Regression	0.0499
	Test Time (min)	1°	Classic ML	1	24	24	Linear Regression	0.0001
		2°	Classic ML	1	168	24	Linear Regression	0.0006
		3°	Ensemble Type I-B	1	24	24	(3 FC) * 5	0.0017

The main findings of the work, as conclusions from Table 1, are:

- The best value for parameter p seems to be 168, i.e., having a rolling window length of 1 week, it appears that shorter and longer values do not generally provide the best results. The exceptions are Seq2seq models, which prefer a value of 24 (1 day), and some deep learning models that combine a CNN with an LSTM, for which 720 (1 month) is the preferred length.
- The best number of features associated with each time-slot seems to be one, i.e., to use the load value exclusively. The exception is again the Seq2seq models which prefer to have the date features additionally. The behavior of logistic regression is confusing in this regard because they obtain best results with both 1 and 45 features/time-slot.

- For near-term forecasts (prediction of the following value), the linear/simple models, as linear regression, provide the best results when using a large enough number of predictors ($p = 128$) and a simple scalar predictor value ($f = 1$), which is the electric load. In this scenario, another model providing the best results is the Seq2seq+attention model with a smaller number of predictors ($p = 24$) and the full vector of predictor values ($f = 45$).
- For longer-term forecasts, the deep learning models with one or two LSTM layers provide the best results. Another model with excellent performance is a model combining 1D-CNN layers and LSTM layers. The model with the best sMAPE metric is an ensemble Type II model with a small number of dense and LSTM blocks. In all these cases, the best results are obtained with a large number of predictors ($p = 168$, 720) and a smaller number of values per predictor ($f = 1$).
- Considering the average results for the 24 predicted values, the best results are obtained for: (1) ensemble Type II models with a small number of dense and LSTM blocks, (2) ensemble Type I models with a small number of dense blocks, and (3) combined 1D-CNN and LSTM layers with $p = 720$. All these results are obtained with the smaller number of values per predictor ($f = 1$).
- Interestingly, multiple output models can produce better results for distant forecasts than single output models (classic ML models) even when the latter models have a specific regressor to predict each output. This is a demonstration of the great correlation between the outputs that a single model can learn, while k independent regressors lose this valuable information.

In Figure A1 (Appendix A), a complete table is provided, integrating the results from Figures 6–8 into a single one. The Table in the Appendix A contains additional results that are not shown in Figures 6–8, with the intention to simplify them by not showing the worst performing models.

Another interesting result is that including additional features (date/time, weather) should provide an improvement in the results due to the strong correlation between weather and electricity consumption. Nevertheless, this influence turned out to be less significant than anticipated, with most models performing better simply by using the past load values. An explanation could be obtained from the results presented in previous works showing that the influence of weather data is more noticeable as the forecast period increases [36,37], and their influence is small for a forecast horizon of a few hours [37].

It is interesting to analyze the evolution of the forecast metrics for the different time-ahead time-slots. This evolution depends mainly on the forecasting model, and it can range from a monotonous decreasing line to a decreasing exponential, and it can have intermediate plateaus where the performance is almost constant regardless of the prediction time distance. In all cases, the performance metrics experience a decline when the forecast is for a longer time interval, as intuitively expected. Figure 9 presents the evolution of the forecast metrics for different k-ahead forecasts for two of the best models for average forecasting (Table 1) and Figure 10 for another two of the best models for near-term forecasting (Table 1).

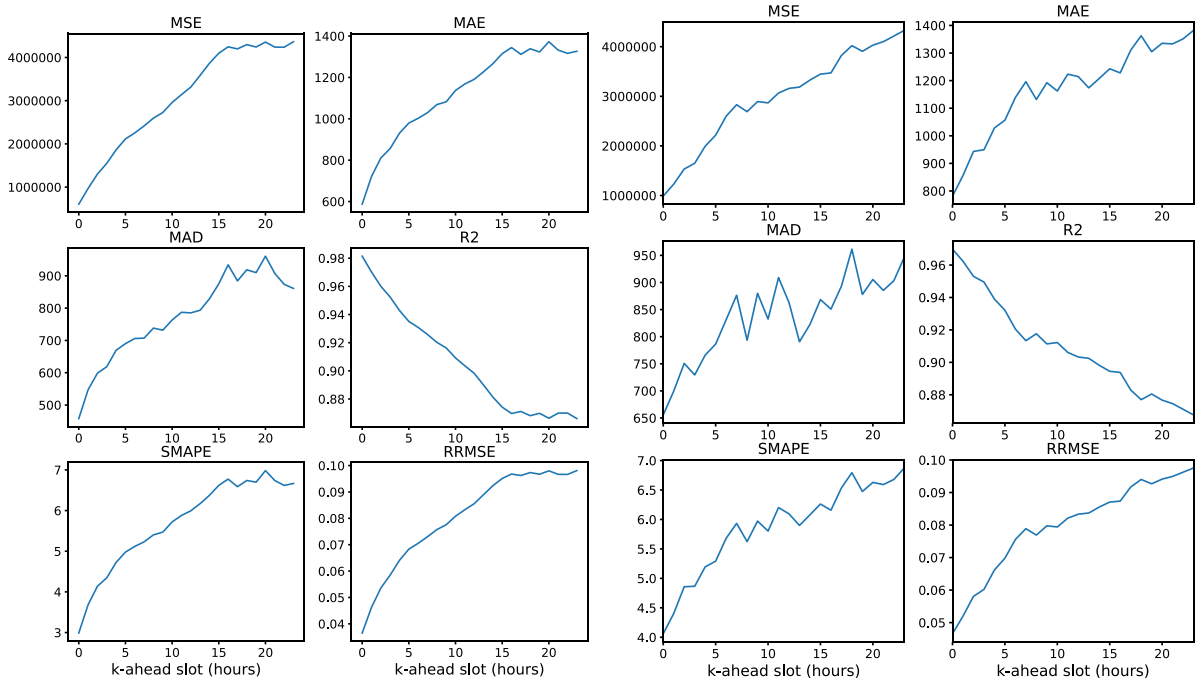

Figure 9. Forecast metrics for successive predicted time-slots for models: ensemble Type II ((1 LSTM + 2 FC) * 2 + (4 FC) * 2) (f = 1, p = 168, k = 24) (**left**) and DL architecture (2 1D-CNN + 1 LSTM + 1 FC) (f = 1, p = 720, k = 24) (**right**).

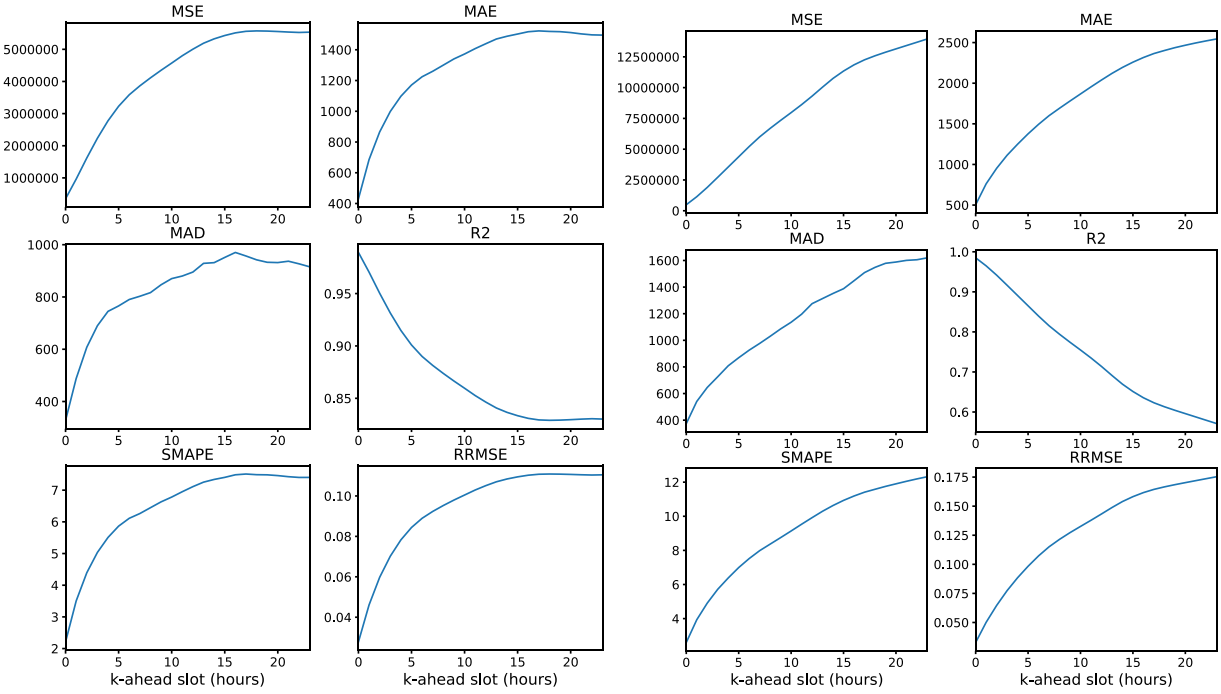

Figure 10. Forecast metrics for successive predicted time-slots for models: classic ML, (linear regression) (f = 1, p = 168, k = 24) (**left**) and Seq2seq + attention (2 LSTM + 1 FC) (f = 45, p = 24, k = 24) (**right**).

It is interesting to analyze the impact of the rolling window length (parameter p) on the forecast performance metrics. Figure 11 presents the impact on the performance metrics with a rolling window length between 24 (1 day) and 1440 h (60 days). In Figure 11, a vertical red line marks an interval of one week (168 h). A performance improvement is obtained by increasing the value of p up to a value of 168 (1 week of rolling window length); from this point, the improvement is much smaller, and even decreases with a

p value greater than 720 (4 weeks). This behavior explains why the best results were obtained with a p value between 168 and 720. The Seq2seq models are an exception in this case due to their difficulties in handling large input vectors.

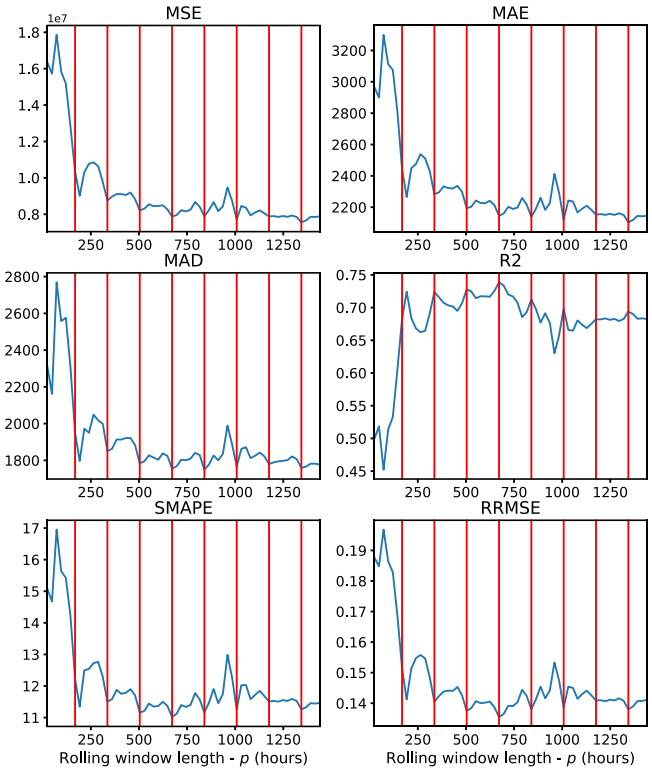

Figure 11. Influence of the rolling window length (p) on different forecast metrics (MSE, MAE, MAD, R^2, SMAPE and RRMSE), obtained with the DMD model. The range of values for p is between 24 (1 day) and 1440 h (60 days).

It may also be of interest to investigate the evolution of the loss function during training. Figure 12 provides this evolution for one of the deep learning architectures ((1 LSTM + 1 FC) (f = 1, p = 168, k = 24)), using the mean squared error (MSE) loss. In general, we observed that after 20–30 epochs, most of the models converged smoothly, with the Seq2seq and ensemble models experiencing the most difficulties in their convergence.

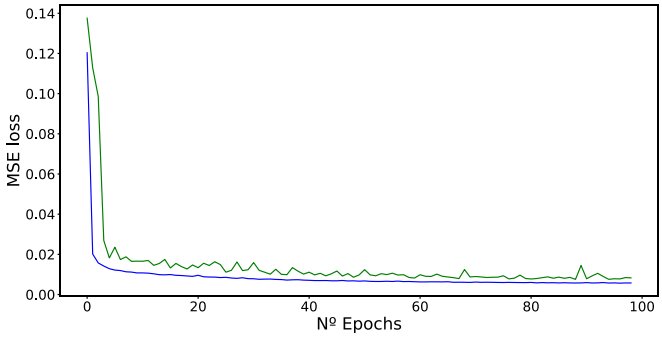

Figure 12. Training (blue line) and validation (green line) MSE loss evolution for successive epochs while training (model: DL architecture (1 LSTM + 1 FC) (f = 1, p = 168, k = 24)).

To provide a visual indication of the quality of the 24 h forecast, Figure 13 shows a comparison between real (ground-truth) load signals and their forecasts as we increase the forecast time horizon. The different diagrams are 24 h time windows taken at random points in the test set. The forecast follows the real signal almost perfectly to a point where it

begins to drift. The drift point is different for different samples, with some signals perfectly followed almost all the time and others beginning to separate in the initial forecasts. In all cases, the forecast signal is a smoothed version of the real one.

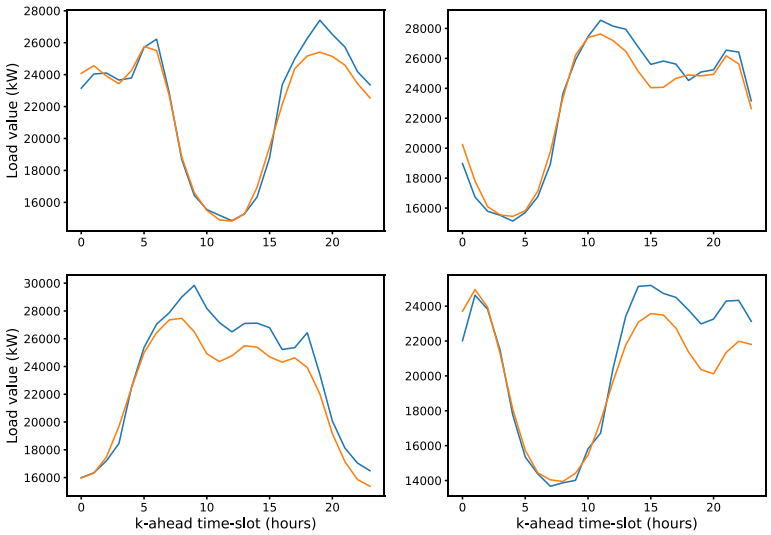

Figure 13. Comparison between real 24 h load signals (blue lines) and their forecasts (orange lines) for data in the test set (model: ensemble Type II ([1 LSTM + 2 FC] * 2 + (4 FC)*2) (f = 1, p = 168, k = 24)).

Figures 14–17 present the ability of the DMD algorithm to identify the structure of the time-series (load values). Figure 14 shows the evolution of the signal energy (sum of squares of principal eigenvalues) in relation to the total signal energy (sum squares of all eigenvalues). We observe that to achieve a 99% total signal energy, generally a few eigenvalues are sufficient. In particular, for our dataset, 10 eigenvalues (modes) are sufficient to capture the 99% of total signal energy for a rolling window of 24 and 168 h, and 16 eigenvalues for a rolling window of 720 h.

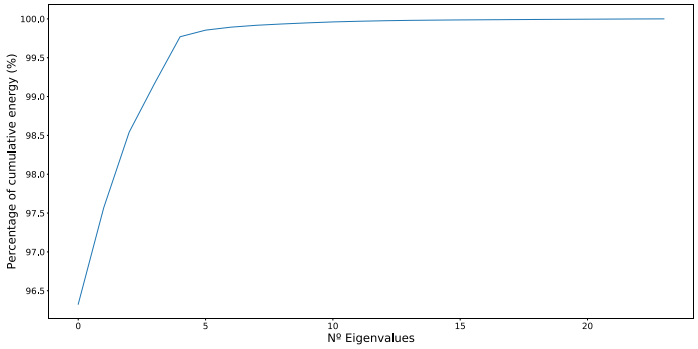

Figure 14. Graph showing the total energy of the signal captured by the linear DMD approximation versus the number of eigenvectors (modes) used to implement the transformation. This graph was obtained with a rolling window of 24 h, and is very similar to the graph for a rolling window of 168 h.

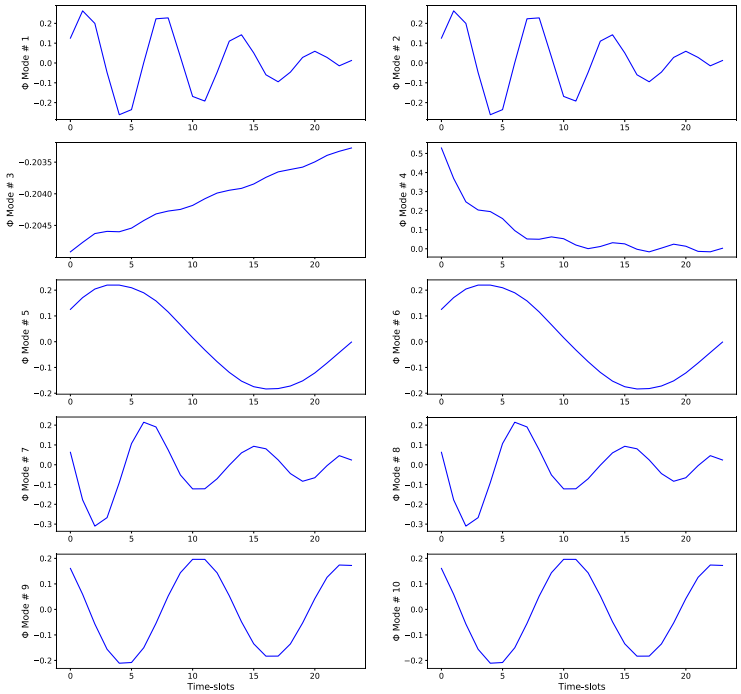

Figure 15. Principal eigenvectors (Φ) of the DMD algorithm (best linear mapping) for the temporal evolution of the load values. The eigenvectors are in order of importance (according to their corresponding eigenvalues) and represent the main modes of temporal evolution of the time-series. They clearly show the strong intra-day periodicity of the load values. The values shown correspond to a rolling window of 24 h (1 day).

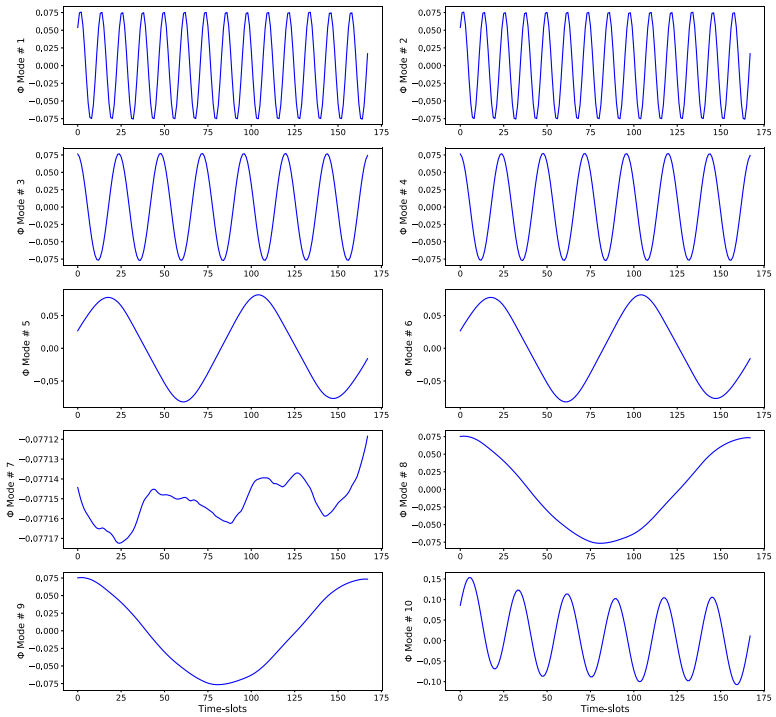

Figure 16. Principal eigenvectors (Φ) of the DMD algorithm (best linear mapping) for the temporal evolution of the load values. The eigenvectors are in order of importance (according to their corresponding eigenvalues) and represent the main modes of temporal evolution of the time-series. They clearly show the strong daily periodicity of the load values. The values shown correspond to a rolling window of 168 h (1 week).

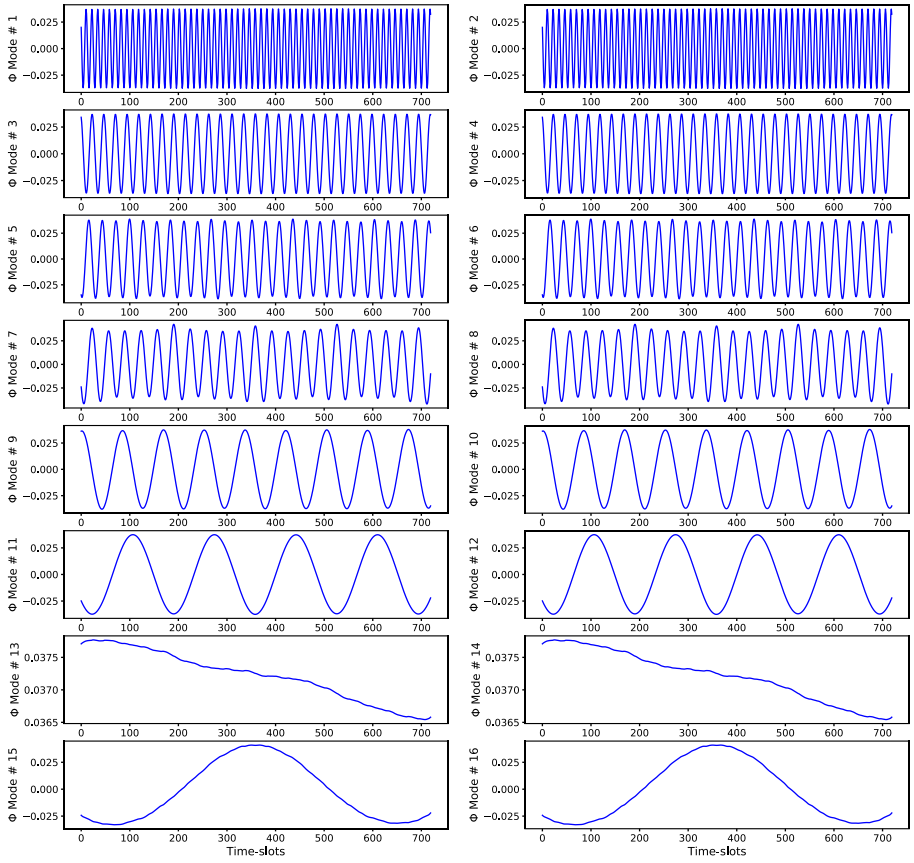

Figure 17. Principal eigenvectors (Φ) of the DMD algorithm (best linear mapping) for the temporal evolution of the load values. The eigenvectors are in order of importance (according to their corresponding eigenvalues) and represent the main modes of temporal evolution of the time-series. They clearly show the strong daily/weekly periodicity of the load values. The values shown correspond to a rolling window of 720 h (1 month). In this case, 16 eigenvectors were needed to capture 99% of the total signal energy.

Figures 15–17 show the principal eigenvectors (Φ) of the DMD algorithm for the temporal evolution of the load values and for different rolling window lengths (24, 168, and 720 h). The eigenvectors are in order of importance (according to their corresponding eigenvalues) and represent the main modes of temporal evolution of the time-series. They show the strong periodicity of the load values at different levels of granularity: intra-day, daily, and weekly.

We implemented all the neural network models (deep learning, Seq2seq, attention, and gaNet) in python using Tensorflow/Keras [59]. For all other models, we used the scikit-learn python package [60].

5. Conclusions

This work provides a comprehensive analysis of a significant number of novel forecasting models from different fields and areas of research, exploring reduced-order dynamical systems techniques (DMD), classic ML models (linear regression, random forest, gradient boosting, k-nearest neighbors, support vector regression, and AdaBoost), DL models (convolutional and recurrent architectures and their combination), DL sequence to sequence models (Seq2seq with and without attention), and specific DL ensemble models (gaNet). All techniques were applied to a unique dataset obtained from real data from a Spanish utility, which implies having a homogeneous set of results that allows a systematic comparison between models.

Some of the models analyzed have been widely applied to STLF (e.g., classical machine learning models), while others have rarely been applied (e.g., sequence to sequence, attention, DMD), and some of them have been applied for the first time to STLF, as far as we know (e.g., deep learning ensemble models specifically suitable for time-series forecasting, and specific configurations combining convolutional and recurring layers).

The main conclusions obtained from this work are the following: (a) A rolling window of 1 week appears to be the best value for most models, and longer and shorter values do not, on average, provide better results. (b) Another interesting result is that including additional features (date/time, weather) could provide an improvement in the results in some scenarios, but not in general, the models that obtain better results simply using the past values of load being more numerous. (c) For very-short term forecasts, it is important to note that simple linear models and Seq2seq architectures provide better results than more complex models. (d) For longer-term forecasts, deep learning models consisting of convolutional layers followed by recurrent layers provide the best results, better than pure convolutional or recurrent networks and similar, in forecasting performance in DL ensemble models (gaNet). (e) When the focus is to have good average forecasts, considering both very short-term and medium-term forecasts, the best models are DL ensemble models (gaNet), followed by CNN/RNN network combinations.

This work also validates the importance of deep ensemble models and how they provide better performance results for predictions with increasing degrees of uncertainty. It corroborates other studies that try to understand the foundations of this behavior, considering different points of view. This work presents a specific deep ensemble model that originated by combining ideas from gradient boosting and residual networks [11] and demonstrates the results from other ensemble architectures [27,61] where emphasis is placed on the random initialization of the different ensemble entities.

As already mentioned, the electrical sector is a critical infrastructure and therefore one of the main industries exposed to security and fraud attacks, i.e., DDoS (distributed denial of service) cybersecurity threats. It could also be vulnerable to so-called ransomware cybersecurity attacks, where the attacker hijacks (typically through encryption) customer data that can only be recovered after payment to the attacker. One specific threat scenario is related to charging and billing. In smart grids, the central billing system receives customer consumptions from so-called smart meters through communication technologies. It also receives aggregated consumptions from intermediate elements of the grid (substations at district level, city distribution level). A cybersecurity attack could target both the billing system and the individual smart meters, preventing the electrical utility from knowing how much electricity has been consumed by the customers. An electrical company would lose a significant amount of money if it was not able to bill its customers, and the attacker could ask for an important ransom. However, thanks to machine learning, the electrical company could estimate the consumption made based on historical data, and therefore, minimize the incentive of the attacker to ask for a ransom, and if the attack is executed in any case, make a provisional billing based on an accurate estimation. With the current dataset, estimations are made at the small city level, because consumption data from individual smart meters are much more restrictive from a General Data Protection Regulation (European Union) data privacy perspective. As a future line of study, research could be extended with anonymized personal identifiable information.

Considering the experience gained with deep learning models applied to time-series forecasting from the perspective of a multivariate multi-output regression problem, we plan to extend the experiments with novel deep learning sequential models, such as transformers [62]. Another future work will be to build controlled perturbation-based experiments to demonstrate the capabilities of STLF as a security/threat indicator.

Author Contributions: Conceptualization, methodology, formal analysis, and software, M.L.-M.; validation, A.S.-E. and L.H.-C.; resources, B.C., J.I.A., A.S.-E. and L.H.-C.; data curation, L.H.-C.; writing—original draft preparation, M.L.-M.; writing—review and editing, L.H.-C., J.I.A. and A.S.-E.; visualization, M.L.-M.; supervision, B.C., J.I.A. and A.S.-E.; project administration, B.C. and J.I.A.;

funding acquisition, B.C. and J.I.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded with grant RTI2018-098958-B-I00 from Proyectos de I+D+i «Retos investigación», Programa Estatal de I+D+i Orientada a los Retos de la Sociedad, Plan Estatal de Investigación Científica, Técnica y de Innovación 2017-2020, Spanish Ministry for Science, Innovation, and Universities; the Agencia Estatal de Investigación (AEI) and the Fondo Europeo de Desarrollo Regional (FEDER).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Restrictions apply to the availability of these data. Data was obtained from Iberdrola and are available from the authors with the permission of Iberdrola. Code will be shared upon properly justified requests.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In this annex is presented the complete set of results from which the most important results and conclusions have been extracted.

The table below shows all the forecast performance metrics for the average, first (T-0), and last (T-23) predicted time-slots using DL, ML, and ensemble models. The average is obtained along the 24 predicted values (k = 24). Different number of predictors (p) and network architectures are considered for a fixed number of values per predictor (f = 1). The table is color coded (column-wise) with a green–red palette corresponding to best–worst results. Section 3.2 provides how to interpret the description of the models and their groups. The last two columns provide the training and test times for all models.

Class	f	p	Model	Average						T-0						T-23						Training Time (min)	Test Time (min)
				MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE	MSE	MAE	MAD	R2	sMAPE	RRMSE		
Seq2Seq	45	24	1 LSTM + 1 FC	6,479,118	1770.5	1261.5	0.801	9.0	0.115	434,997	476.9	353.3	0.987	2.4	0.031	10,543,200	2436.6	1810.2	0.676	12.2	0.152	14.369	0.0479
	45	24	2 LSTM + 1 FC	8,289,029	1747.5	969.9	0.745	8.7	0.129	451,045	479.0	352.3	0.986	2.5	0.032	13,433,500	2427.8	1372.3	0.587	11.8	0.172	63.148	0.0715
	45	24	2 1D-CNN + 1 LSTM + 1 FC	47,089,200	5315.6	4358.8	-0.448	27.0	0.315	2,358,580	1182.3	940.3	0.927	6.4	0.072	53,410,500	5806.8	4783.1	-0.642	29.5	0.343	16.880	0.0684
	45	168	1 LSTM + 1 FC	8,420,570	1879.8	1174.8	0.742	9.2	0.129	434,666	477.9	356.7	0.987	2.4	0.031	14,577,200	2672.0	1687.9	0.553	12.9	0.179	82.606	0.0763
	45	168	2 LSTM + 1 FC	6,788,302	1748.8	1176.9	0.792	8.8	0.117	419,015	477.5	371.4	0.987	2.4	0.030	11,131,400	2377.3	1613.9	0.658	11.7	0.157	84.025	0.1339
	45	168	2 1D-CNN + 1 LSTM + 1 FC	29,722,340	4474.9	4106.6	0.088	22.0	0.251	3,050,370	1326.3	1034.1	0.906	6.3	0.082	33,555,800	4903.7	4572.9	-0.030	24.2	0.272	35.034	0.1019
	45	720	1 LSTM + 1 FC	8,534,025	1890.2	1154.7	0.739	9.4	0.131	462,248	489.2	359.5	0.986	2.5	0.032	14,063,700	2592.6	1628.2	0.570	12.6	0.176	100.725	0.2294
	45	720	2 LSTM + 1 FC	6,366,990	1667.3	1077.9	0.805	8.4	0.114	446,108	491.5	375.3	0.986	2.5	0.031	9,796,230	2190.8	1422.8	0.700	10.8	0.147	160.340	0.3678
	45	720	2 1D-CNN + 1 LSTM + 1 FC	30,051,510	4128.2	3208.7	0.080	20.3	0.255	8,307,780	2268.5	1907.6	0.746	10.9	0.135	32,701,000	4385.3	3466.6	-0.001	21.7	0.268	115.625	0.2975
Seq2Seq + Attention	45	24	1 LSTM + 1 FC	8,384,991	1854.4	1172.6	0.742	9.1	0.130	488,121	503.9	374.0	0.985	2.6	0.033	13,955,300	2544.9	1620.0	0.571	12.3	0.175	13.582	0.0665
	45	24	2 LSTM + 1 FC	7,578,606	1698.7	987.7	0.767	8.4	0.123	416,383	464.7	351.1	0.987	2.4	0.030	12,384,800	2373.5	1409.8	0.619	11.5	0.165	17.083	0.0669
	45	24	2 1D-CNN + 1 LSTM + 1 FC	159,237,700	11174.2	10988.3	-3.896	70.3	0.592	143,198,000	10482.3	10183.4	-3.402	64.2	0.562	162,892,000	11335.8	11235.5	-4.008	71.7	0.599	12.383	0.0675
	45	168	1 LSTM + 1 FC	9,017,546	1936.1	1214.5	0.723	9.7	0.135	528,351	525.2	388.1	0.984	2.7	0.034	13,566,200	2555.7	1646.6	0.584	12.7	0.173	24.848	0.0865
	45	168	2 LSTM + 1 FC	6,999,635	1698.1	1045.9	0.785	8.4	0.119	445,945	485.9	360.4	0.986	2.5	0.031	10,888,600	2258.5	1409.4	0.666	11.1	0.155	87.210	0.1232
	45	168	2 1D-CNN + 1 LSTM + 1 FC	39,696,310	4883.9	4246.0	-0.218	25.7	0.286	4,081,380	1499.9	1128.3	0.875	7.5	0.095	62,598,700	6520.2	6020.2	-0.921	35.2	0.372	49.909	0.1016
	45	720	1 LSTM + 1 FC	16,830,620	2744.8	1849.3	0.485	12.7	0.180	484,231	503.7	371.9	0.985	2.6	0.033	29,492,600	3963.0	2761.7	0.098	17.8	0.255	184.945	0.2322
	45	720	2 LSTM + 1 FC	9,215,292	2022.9	1331.0	0.718	10.0	0.135	450,344	496.0	379.3	0.986	2.5	0.031	15,789,300	2842.9	1856.9	0.517	13.8	0.186	167.660	0.3336
	45	720	2 1D-CNN + 1 LSTM + 1 FC	26,543,160	3987.9	3370.7	0.188	19.6	0.239	8,106,980	2239.7	1862.6	0.752	11.2	0.134	30,918,400	4356.0	3724.7	0.054	21.4	0.261	109.441	0.2477
DL Achitectures	1	24	1 LSTM + 1 FC	6,022,328	1730.5	1232.1	0.815	8.8	0.112	1,228,310	842.4	659.5	0.962	4.4	0.052	9,384,450	2203.3	1578.5	0.712	11.1	0.144	3.812	0.0091
	1	24	2 LSTM + 1 FC	5,675,340	1694.3	1219.1	0.826	8.5	0.109	1,152,420	815.4	633.0	0.965	4.2	0.050	8,421,500	2110.2	1508.4	0.741	10.7	0.136	4.826	0.0161
	1	24	2 1D-CNN + 1 FC	5,975,028	1734.3	1271.7	0.816	8.8	0.112	1,252,970	878.2	716.2	0.961	4.6	0.053	10,303,400	2260.3	1486.7	0.683	11.4	0.151	1.684	0.0052
	1	24	2 1D-CNN (MaxPooling) + 1 FC	6,924,590	1881.6	1376.1	0.787	9.6	0.120	1,572,820	984.0	810.8	0.952	5.1	0.059	10,800,700	2438.0	1813.5	0.668	12.3	0.154	1.564	0.0058
	1	24	2 1D-CNN + 1 LSTM + 1 FC	5,923,070	1724.4	1251.8	0.818	8.7	0.112	1,340,070	892.7	712.1	0.959	4.6	0.054	9,072,510	2164.9	1472.1	0.721	11.0	0.141	1.805	0.0121
	1	24	2 1D-CNN + 2 LSTM + 1 FC	5,587,690	1660.1	1180.9	0.828	8.3	0.108	1,284,620	894.1	738.8	0.961	4.5	0.053	8,512,330	2096.0	1463.7	0.738	10.6	0.137	3.646	0.0176
	1	24	6 FC	6,043,480	1746.9	1256.7	0.814	8.8	0.113	2,106,420	1128.1	897.7	0.935	5.8	0.068	9,780,620	2307.0	1700.4	0.699	11.6	0.147	0.345	0.0023
	1	168	1 LSTM + 1 FC	3,296,706	1224.4	867.3	0.899	6.2	0.084	1,122,670	822.6	673.3	0.966	4.2	0.050	4,244,780	1372.8	923.1	0.870	7.0	0.097	122.766	0.0431
	1	168	2 LSTM + 1 FC	3,595,281	1265.2	889.6	0.890	6.4	0.088	1,578,830	954.4	750.7	0.952	4.9	0.059	4,316,120	1339.5	879.5	0.868	6.7	0.098	52.734	0.0852
	1	168	2 1D-CNN + 1 FC	7,223,554	2067.2	1739.4	0.778	10.4	0.124	8,383,650	2405.4	2200.5	0.743	12.0	0.136	12,656,700	2909.2	2623.3	0.612	14.6	0.167	4.088	0.0107
	1	168	2 1D-CNN (MaxPooling) + 1 FC	5,344,936	1653.3	1256.5	0.836	8.4	0.107	4,393,440	1625.8	1329.7	0.865	8.3	0.098	8,322,360	2171.6	1739.3	0.745	11.0	0.135	2.336	0.0129
	1	168	2 1D-CNN + 1 LSTM + 1 FC	3,950,512	1371.7	997.6	0.879	6.9	0.092	1,334,120	898.3	741.9	0.959	4.6	0.054	5,206,950	1542.8	1039.7	0.840	7.8	0.107	4.332	0.0153
	1	168	2 1D-CNN + 2 LSTM + 1 FC	3,383,631	1274.5	911.8	0.896	6.4	0.085	1,191,380	843.5	683.8	0.963	4.3	0.051	5,022,610	1537.1	1050.3	0.846	7.7	0.105	8.075	0.0213
	1	168	6 FC	3,717,323	1274.2	877.5	0.886	6.3	0.089	1,184,110	832.9	646.5	0.964	4.3	0.051	4,955,080	1464.8	966.5	0.848	7.3	0.105	2.508	0.0030
	1	720	1 LSTM + 1 FC	4,788,052	1592.8	1199.2	0.853	8.0	0.102	2,197,550	1160.0	956.2	0.933	6.0	0.070	6,004,430	1748.9	1267.4	0.816	8.8	0.115	34.154	0.2102
	1	720	2 LSTM + 1 FC	5,541,973	1732.3	1319.1	0.830	8.8	0.110	3,487,550	1444.4	1170.7	0.893	7.4	0.088	7,219,800	2021.4	1601.1	0.779	10.4	0.126	35.682	0.3469
	1	720	2 1D-CNN + 1 FC	19,444,790	3653.8	3255.8	0.405	18.1	0.206	21,130,200	3884.8	3484.3	0.353	19.3	0.216	22,165,200	3934.4	3504.3	0.322	19.5	0.221	6.670	0.0401
	1	720	2 1D-CNN (MaxPooling) + 1 FC	33,198,780	4883.6	4484.2	-0.016	24.1	0.270	33,122,000	4880.2	4475.0	-0.014	24.1	0.270	33,117,800	4879.7	4470.0	-0.013	24.1	0.270	5.667	0.0291
	1	720	2 1D-CNN + 1 LSTM + 1 FC	2,982,062	1171.1	836.6	0.909	5.9	0.080	986,983	781.2	655.7	0.970	4.1	0.047	4,328,650	1382.1	946.6	0.868	6.9	0.098	9.040	0.0338
	1	720	3 1D-CNN + 1 LSTM + 1 FC	3,340,233	1233.4	878.0	0.898	6.2	0.084	1,203,010	825.4	650.4	0.963	4.2	0.051	4,746,470	1459.7	966.2	0.855	7.4	0.102	16.328	0.0461
	1	720	2 1D-CNN + 2 LSTM + 1 FC	3,316,090	1239.0	884.4	0.899	6.1	0.084	1,268,230	866.2	692.8	0.961	4.4	0.053	4,712,460	1474.2	1022.5	0.856	7.2	0.102	17.393	0.0391
	1	720	6 FC	3,566,631	1290.4	944.9	0.891	6.5	0.088	2,102,510	1086.4	873.4	0.936	5.6	0.068	4,663,830	1500.0	1081.6	0.857	7.5	0.101	0.774	0.0036
	45	24	1 LSTM + 1 FC	6,195,447	1698.4	1170.8	0.810	8.5	0.115	1,494,960	937.8	722.7	0.954	4.8	0.057	9,810,780	2117.0	1435.5	0.698	10.6	0.147	4.765	0.0077
	45	24	2 LSTM + 1 FC	7,748,237	1802.8	1121.3	0.762	9.1	0.128	1,528,600	913.9	693.6	0.953	4.7	0.058	12,091,800	2310.3	1419.4	0.628	11.4	0.163	19.256	0.0149
	45	24	2 2D-CNN + 1 FC	33,094,060	4882.9	4494.4	-0.018	24.2	0.270	33,039,300	4880.1	4482.6	-0.016	24.1	0.270	33,275,800	4891.8	4506.8	-0.023	24.2	0.271	7.766	0.0226
	45	24	2 1D-CNN + 1 FC	5,907,817	1743.4	1284.2	0.818	8.8	0.113	2,394,930	1208.6	993.5	0.926	6.4	0.073	8,622,150	2095.9	1485.9	0.735	10.5	0.138	3.201	0.0037
	45	24	2 1D-CNN (MaxPooling) + 1 FC	6,126,764	1792.7	1336.6	0.812	9.1	0.115	2,658,230	1263.9	1012.1	0.918	6.6	0.077	8,993,910	2172.0	1568.9	0.724	11.0	0.141	3.788	0.0037
	45	24	2 1D-CNN + 1 LSTM + 1 FC	4,975,367	1507.8	1016.4	0.847	7.6	0.103	1,564,870	928.8	715.4	0.952	4.8	0.059	7,307,580	1830.8	1187.4	0.775	9.2	0.127	19.558	0.0074
	45	24	2 1D-CNN + 2 LSTM + 1 FC	5,738,281	1629.7	1087.6	0.824	8.3	0.111	2,041,510	1086.1	856.6	0.937	5.6	0.067	8,104,580	1970.8	1330.0	0.751	10.0	0.134	12.020	0.0150
	45	24	6 FC	7,861,525	1911.9	1302.3	0.758	9.6	0.130	2,565,100	1246.3	1025.8	0.921	6.5	0.075	10,000,900	2193.8	1505.5	0.693	10.9	0.148	1.753	0.0019
	45	168	1 LSTM + 1 FC	6,974,843	1766.6	1170.7	0.786	8.9	0.122	2,428,550	1195.6	962.4	0.925	6.3	0.073	9,739,580	2082.1	1305.0	0.701	10.4	0.147	18.308	0.0460
	45	168	2 LSTM + 1 FC	7,798,327	1833.4	1166.8	0.761	9.2	0.129	2,802,340	1228.6	941.5	0.914	6.3	0.079	10,425,300	2170.0	1400.7	0.680	10.9	0.152	31.858	0.0668
	45	168	2 1D-CNN + 1 FC	11,667,630	2662.6	2181.5	0.642	13.2	0.160	9,972,840	2524.9	2106.5	0.694	12.7	0.148	13,048,100	2794.2	2236.9	0.600	13.8	0.170	1.921	0.0131
	45	168	2 1D-CNN (MaxPooling) + 1 FC	16,910,420	3278.5	2812.9	0.481	16.2	0.192	20,405,200	3713.7	3217.1	0.374	18.5	0.212	24,157,800	4056.4	3695.9	0.259	20.0	0.231	4.178	0.0163
	45	168	2 1D-CNN + 1 LSTM + 1 FC	5,372,823	1487.8	995.2	0.835	7.5	0.107	1,576,770	951.4	744.6	0.952	4.9	0.059	8,369,170	1832.4	1158.3	0.743	9.2	0.136	8.235	0.0176
	45	168	2 1D-CNN + 2 LSTM + 1 FC	4,189,875	1332.0	882.4	0.871	6.7	0.094	1,247,890	837.3	645.4	0.962	4.4	0.052	5,960,730	1596.7	1009.0	0.817	7.9	0.115	15.313	0.0219
	45	168	6 FC	8,624,556	2140.4	1547.8	0.735	10.6	0.137	6,972,700	2020.0	1585.7	0.786	10.2	0.124	11,376,000	2526.7	1933.5	0.651	12.5	0.158	0.898	0.0065
	45	720	1 LSTM + 1 FC	6,090,553	1599.7	1081.7	0.814	8.1	0.114	1,894,670	1017.9	776.7	0.942	5.3	0.065	8,479,800	1843.8	1179.6	0.741	9.1	0.137	56.410	0.2215
	45	720	2 LSTM + 1 FC	7,241,965	1745.5	1130.5	0.778	8.7	0.124	2,611,450	1191.2	911.1	0.920	6.0	0.076	9,902,940	2008.5	1205.5	0.697	9.9	0.148	69.946	0.3704
	45	720	2 1D-CNN + 1 FC	33,224,320	4884.9	4485.8	-0.017	24.1	0.270	33,419,500	4895.9	4503.5	-0.023	24.2	0.271	33,571,700	4903.6	4500.3	-0.027	24.2	0.272	8.410	0.0447
	45	720	2 1D-CNN (MaxPooling) + 1 FC	33,134,430	4880.0	4485.1	-0.014	24.1	0.270	33,276,900	4888.4	4490.7	-0.018	24.2	0.270	33,390,600	4894.1	4508.7	-0.022	24.2	0.271	15.823	0.0499
	45	720	2 1D-CNN + 1 LSTM + 1 FC	5,059,471	1496.8	1014.4	0.845	7.5	0.104	1,645,400	959.7	743.7	0.950	5.1	0.060	6,419,050	1721.4	1157.2	0.804	8.6	0.119	21.860	0.0670
	45	720	3 1D-CNN + 1 LSTM + 1 FC	5,661,417	1612.8	1116.3	0.827	8.1	0.111	2,411,320	1137.5	849.1	0.926	5.9	0.073	7,315,740	1854.9	1268.3	0.776	9.2	0.127	17.143	0.0900
	45	720	2 1D-CNN + 2 LSTM + 1 FC	6,291,731	1746.6	1206.9	0.807	8.6	0.117	3,990,250	1471.3	1093.8	0.878	7.4	0.094	7,100,180	1837.5	1236.7	0.783	9.1	0.125	23.993	0.0625
	45	720	6 FC	9,512,350	2313.1	1757.0	0.709	11.4	0.145	10,589,100	2407.2	1778.1	0.676	11.9	0.153	11,515,800	2579.2	2037.7	0.648	12.9	0.159	3.164	0.0255
Classic ML	1	24	Random Forest	5,302,593	1478.8	889.0	0.837	7.5	0.105	568,900	534.5	380.4	0.983	2.7	0.035	8,331,450	1954.0	1213.0	0.744	9.9	0.135	1.228	0.0136
	1	24	Linear Regression	10,533,310	2266.1	1547.2	0.676	11.6	0.148	902,373	692.7	497.0	0.972	3.5	0.045	13,948,600	2731.6	1865.2	0.571	14.0	0.175	0.007	0.0001
	1	24	Gradient Boosting	20,891,520	3810.9	3399.4	0.358	19.3	0.214	14,067,000	3155.2	2888.6	0.568	16.2	0.176	21,707,300	3886.6	3421.5	0.333	19.6	0.219	0.089	0.0009
	1	24	KNN	4,989,475	1418.2	862.2	0.847	7.2	0.102	967,456	698.0	510.0	0.970	3.6	0.046	8,942,990	2001.4	1216.0	0.725	10.0	0.140	0.019	0.1742
	1	24	Support Vector Regression	9,076,027	2198.5	1699.7	0.721	11.2	0.138	1,804,010	1105.8	965.6	0.945	5.7	0.063	12,368,900	2628.5	2062.1	0.620	13.3	0.165	2.525	0.3637
	1	24	AdaBoost	12,802,020	2782.2	2266.8	0.606	14.2	0.166	2,758,610	1348.2	1159.9	0.915	7.0	0.078	14,798,600	2956.5	2320.5	0.545	15.0	0.181	0.507	0.0074
	1	168	Random Forest	5,020,732	1349.2	780.1	0.846	6.7	0.104	757,942	618.7	454.2	0.977	3.1	0.041	5,815,730	1481.1	846.7	0.822	7.4	0.113	9.665	0.1074
	1	168	Linear Regression	4,240,085	1297.5	827.1	0.870	6.4	0.094	350,885	430.9	328.1	0.989	2.2	0.028	5,533,440	1495.4	915.4	0.830	7.4	0.110	0.050	0.0006
	1	168	Gradient Boosting	16,874,050	3388.8	3016.4	0.482	17.3	0.193	13,894,000	3108.7	2870.8	0.574	16.0	0.175	17,038,100	3405.4	3041.5	0.477	17.4	0.194	0.765	0.0049
	1	168	KNN	4,112,377	1282.9	823.5	0.874	6.3	0.095	2,606,990	1055.6	708.0	0.920	5.3	0.076	5,185,820	1444.5	912.0	0.841	7.1	0.107	0.134	3.8043
	1	168	Support Vector Regression	4,366,702	1444.1	1056.4	0.866	7.4	0.097	1,666,740	1044.3	917.9	0.949	5.5	0.061	5,569,000	1614.3	1149.7	0.829	8.2	0.111	3.598	0.7381
	1	168	AdaBoost	8,127,589	2242.5	1898.0	0.751	11.6	0.133	2,203,930	1190.4	1025.6	0.932	6.2	0.070	9,485,050	2465.8	2120.7	0.709	12.6	0.145	3.401	0.0496
	45	24	Random Forest	5,463,283	1470.4	857.2	0.832	7.4	0.107	633,968	556.8	398.2	0.981	2.8	0.037	7,913,630	1803.0	1046.0	0.757	9.1	0.132	9.488	0.1054
	45	24	Linear Regression	6,314,323	1753.8	1260.5	0.806	8.9	0.115	688,331	588.0	415.5	0.979	2.9	0.039	8,622,780	2124.0	1502.5	0.735	10.7	0.138	0.902	0.0010
	45	24	Gradient Boosting	19,925,480	3714.6	3301.1	0.387	18.8	0.209	14,067,000	3155.2	2888.6	0.568	16.2	0.176	20,416,300	3759.5	3307.8	0.372	19.1	0.212	3.271	0.0091
	45	24	KNN	13,130,560	2576.7	1652.0	0.596	12.7	0.170	11,975,500	2422.6	1494.0	0.632	11.9	0.162	14,493,100	2734.1	1824.0	0.554	13.4	0.179	1.476	18.0216
	45	24	Support Vector Regression	7,051,829	2011.4	1617.9	0.783	10.2	0.123	2,608,470	1314.6	1146.2	0.920	6.7	0.076	9,420,250	2361.8	1922.7	0.710	11.9	0.144	38.132	8.6819
	45	24	AdaBoost	11,344,240	2665.9	2276.3	0.651	13.6	0.157	2,720,780	1338.4	1158.9	0.916	7.0	0.077	12,510,100	2773.4	2257.8	0.615	14.2	0.166	6.649	0.0673
	45	168	Random Forest	4,959,729	1351.4	788.8	0.848	6.7	0.103	747,048	614.5	443.3	0.977	3.1	0.041	5,707,890	1472.8	843.3	0.825	7.3	0.112	92.524	1.0280
	45	168	Linear Regression	4,758,965	1379.6	892.7	0.854	6.9	0.100	406,341	442.2	321.5	0.988	2.3	0.030	6,361,340	1625.5	1041.7	0.805	8.1	0.118	73.650	0.0082
	45	168	Gradient Boosting	16,874,050	3388.8	3016.4	0.482	17.3	0.193	13,894,000	3108.7	2870.8	0.574	16.0	0.175	17,038,100	3405.4	3041.5	0.477	17.4	0.194	44.733	0.0694
	45	168	KNN	12,102,020	2544.2	1748.8	0.629	12.4	0.163	11,877,000	2509.1	1716.0	0.636	12.2	0.162	12,215,900	2558.9	1760.0	0.625	12.5	0.164	7.530	274.8777
	45	168	Support Vector Regression	6,325,536	1890.3	1487.0	0.806	9.5	0.118	4,085,740	1558.7	1249.2	0.875	7.9	0.095	7,465,730	2087.5	1678.4	0.771	10.5	0.128	179.090	42.7263
	45	168	AdaBoost	8,299,474	2283.8	1963.3	0.745	11.8	0.134	2,030,910	1133.3	947.5	0.938	5.9	0.067	9,607,530	2459.9	2061.6	0.705	12.7	0.146	55.554	0.4784
DMD	1	24	NA	18,253,160	3234.0	2611.7	0.441	16.4	0.199	7,035,700	2021.0	1515.8	0.785	10.1	0.125	22,294,500	3646.8	3023.9	0.317	18.6	0.222	0.005	0.0695
	1	168	NA	11,280,890	2580.4	2062.0	0.654	13.3	0.158	8,531,820	2298.9	1918.4	0.739	11.9	0.138	13,465,600	2809.6	2272.1	0.587	14.3	0.173	0.053	0.5600
	1	720	NA	8,297,923	2210.2	1803.7	0.717	11.4	0.140	7,232,500	2076.7	1698.1	0.752	10.8	0.131	8,731,240	2261.5	1838.4	0.706	11.7	0.143	0.065	0.8642
Ensemble Type I	1	24	(3 FC)*5	4,911,157	1524.4	1062.0	0.849	7.7	0.101	868,293	730.1	612.4	0.973	3.8	0.044	8,012,950	1970.6	1312.9	0.754	9.9	0.133	1.820	0.0019
	1	24	(4 FC)*5	4,710,536	1480.2	1012.0	0.855	7.5	0.099	889,754	739.5	607.3	0.973	3.9	0.044	7,636,190	1969.6	1374.3	0.765	9.8	0.130	2.278	0.0022
	1	24	(1 LSTM + 1 FC)*10	5,256,786	1608.1	1166.0	0.838	8.2	0.104	663,536	620.4	482.0	0.980	3.2	0.038	8,867,330	2212.0	1656.8	0.727	11.3	0.140	58.649	0.0568
	1	24	(1 LSTM + 1 FC)*10 + dropout(0.3)	23,755,980	4104.2	3837.5	0.270	21.4	0.229	24,718,500	4599.4	4773.9	0.240	26.1	0.233	21,555,400	3823.8	3570.6	0.337	19.3	0.218	34.108	0.0434
	1	24	(2 LSTM + 1 FC)*3	5,312,667	1586.6	1110.1	0.837	8.0	0.105	927,148	738.3	590.6	0.971	3.8	0.045	8,037,210	2067.4	1487.3	0.753	10.3	0.133	21.165	0.0226
	45	24	(3 FC)*5	7,690,465	1914.9	1321.1	0.764	9.6	0.128	2,235,200	1128.2	891.2	0.931	5.9	0.070	10,874,800	2359.8	1689.7	0.666	11.9	0.155	1.504	0.0031
	45	24	(4 FC)*5	8,469,669	1959.6	1258.3	0.740	10.0	0.134	2,134,390	1086.2	841.6	0.934	5.7	0.069	11,740,500	2353.4	1436.9	0.639	11.8	0.161	2.896	0.0028
	45	24	(1 LSTM + 1 FC)*10	6,974,140	1701.1	1039.4	0.786	8.6	0.120	723,268	635.2	499.1	0.978	3.3	0.040	10,420,500	2131.0	1256.6	0.680	10.6	0.152	57.917	0.0492
	45	24	(1 LSTM + 1 FC)*10 + dropout(0.3)	13,571,780	2865.4	2337.1	0.583	14.1	0.173	10,828,000	2593.8	2143.7	0.667	13.0	0.154	14,458,700	2997.8	2532.5	0.556	14.9	0.178	22.944	0.0460
	45	24	(2 LSTM + 1 FC)*3	7,176,459	1815.6	1190.7	0.779	9.2	0.123	1,565,160	982.0	818.2	0.952	5.1	0.059	11,087,600	2279.0	1417.5	0.659	11.5	0.156	16.675	0.0253
	1	168	(3 FC)*5	3,161,967	1147.6	787.7	0.903	5.7	0.082	695,745	635.8	506.8	0.979	3.2	0.039	4,338,160	1342.7	904.1	0.867	6.8	0.098	2.413	0.0033
	1	168	(4 FC)*5	3,256,741	1152.1	771.2	0.900	5.7	0.083	790,755	669.5	524.0	0.976	3.4	0.042	4,472,430	1352.8	890.1	0.863	6.8	0.099	1.733	0.0026
	1	168	(1 LSTM + 1 FC)*5	3,615,454	1258.5	867.5	0.889	6.3	0.088	760,056	678.9	569.1	0.977	3.5	0.041	4,858,600	1394.8	867.2	0.851	6.9	0.104	57.717	0.1239
	1	168	(2 LSTM + 1 FC)*3	3,750,715	1342.7	960.2	0.885	6.8	0.090	1,022,650	788.7	631.5	0.969	4.0	0.047	5,022,560	1511.1	981.7	0.846	7.6	0.105	22.735	0.1317
	45	168	(3 FC)*5	6,195,977	1768.8	1290.1	0.810	8.9	0.116	3,703,070	1465.4	1159.3	0.886	7.6	0.090	7,909,510	2052.0	1543.7	0.757	10.3	0.132	13.312	0.0123
	45	168	(4 FC)*5	6,660,185	1778.6	1248.9	0.796	8.9	0.120	3,321,780	1345.0	1046.3	0.898	6.9	0.086	8,342,990	2077.2	1492.1	0.744	10.4	0.136	11.200	0.0094
	45	168	(1 LSTM + 1 FC)*5	6,689,661	1752.0	1183.0	0.795	8.9	0.119	1,498,140	931.6	728.0	0.954	4.8	0.057	9,546,250	2111.4	1403.9	0.707	10.6	0.145	50.960	0.1463
	45	168	(2 LSTM + 1 FC)*3	5,669,071	1552.3	1011.5	0.826	7.8	0.110	1,610,220	960.9	763.7	0.951	4.9	0.060	7,639,600	1846.9	1175.7	0.766	9.2	0.130	49.340	0.1459
Ensemble Type I-Multiple Losses	1	24	(4 FC)*5	5,049,507	1495.3	993.5	0.845	7.6	0.103	986,533	739.6	556.3	0.970	3.8	0.047	7,830,260	1953.5	1346.9	0.759	9.9	0.131	4.099	0.0031
	1	24	(1 LSTM + 1 FC)*5	5,514,763	1626.0	1135.9	0.830	8.2	0.107	853,369	712.4	569.7	0.974	3.7	0.043	8,177,150	2094.6	1549.5	0.749	10.5	0.134	28.741	0.0267
	1	168	(1 LSTM + 1 FC)*3	3,351,393	1208.6	842.9	0.897	6.0	0.085	969,539	750.9	595.7	0.970	3.9	0.046	4,680,840	1418.6	933.0	0.856	7.0	0.102	1.778	0.0039
	45	24	(4 FC)*5	8,325,770	1948.5	1295.8	0.744	9.8	0.132	1,920,650	1029.0	779.5	0.941	5.3	0.065	10,609,900	2224.6	1429.6	0.674	11.1	0.153	3.452	0.0049
	45	24	(1 LSTM + 1 FC)*5	7,269,773	1721.2	1046.1	0.776	8.7	0.123	1,241,060	829.3	631.2	0.962	4.3	0.052	10,785,800	2094.6	1196.9	0.668	10.5	0.154	28.006	0.0307
	45	168	(1 LSTM + 1 FC)*5	6,949,138	1686.6	1040.4	0.787	8.5	0.121	1,362,150	880.5	691.1	0.958	4.6	0.055	9,728,390	2001.5	1216.1	0.701	9.9	0.146	59.952	0.1442
Ensemble Type I-Weights Sharing	1	24	(3 FC)*5	6,025,247	1791.1	1351.3	0.815	9.1	0.112	1,203,380	857.9	717.8	0.963	4.5	0.051	8,690,810	2163.7	1575.1	0.733	10.9	0.138	1.107	0.0017
	1	24	(4 FC)*5	5,670,401	1680.7	1220.1	0.826	8.5	0.109	1,099,780	807.9	647.6	0.966	4.2	0.049	8,551,730	2045.8	1377.0	0.737	10.3	0.137	2.021	0.0017
	1	168	(3 FC)*5	4,354,109	1425.3	1026.1	0.866	7.2	0.097	3,156,330	1347.3	1043.8	0.903	7.1	0.083	6,977,510	1922.3	1441.4	0.786	9.6	0.124	1.121	0.0022
	1	168	(4 FC)*5	3,844,067	1322.9	941.7	0.882	6.6	0.091	1,267,590	852.3	668.7	0.961	4.4	0.053	5,385,760	1502.9	999.4	0.835	7.4	0.109	3.031	0.0033
	45	24	(3 FC)*5	8,475,861	2196.9	1715.3	0.739	11.1	0.136	5,611,620	1773.3	1354.7	0.827	9.1	0.111	10,887,700	2502.1	1931.9	0.665	12.5	0.155	1.452	0.0034
	45	24	(4 FC)*5	8,526,247	2103.0	1552.9	0.738	10.6	0.135	3,404,970	1414.3	1154.4	0.895	7.5	0.087	12,056,400	2485.3	1767.0	0.629	12.4	0.163	1.185	0.0026
	45	168	(3 FC)*5	33,547,800	4906.2	4521.8	-0.030	24.3	0.272	33,964,500	4928.3	4540.0	-0.042	24.4	0.274	33,623,700	4909.6	4523.8	-0.032	24.3	0.272	2.432	0.0092
	45	168	(4 FC)*5	7,619,318	2063.7	1581.3	0.766	10.3	0.129	6,430,420	1881.8	1418.7	0.803	9.5	0.119	10,722,500	2469.2	1871.0	0.671	12.5	0.154	4.301	0.0091
Ensemble Type II	1	24	(1 LSTM + 2 FC)*2 + (4 FC)*2	5,280,710	1550.9	1033.6	0.838	7.8	0.105	830,195	723.1	615.9	0.974	3.7	0.043	8,266,720	1996.9	1300.2	0.746	10.0	0.135	13.403	0.0131
	45	24	(1 LSTM + 2 FC)*2 + (4 FC)*2	6,663,891	1700.8	1083.0	0.795	8.7	0.118	954,288	728.7	562.9	0.971	3.8	0.046	10,798,700	2155.7	1252.4	0.668	10.7	0.154	14.514	0.0125
	NA	24	(1 LSTM(A) + 2 FC)*1 + (1 LSTM(D) + 2 FC)*1 + (1 LSTM(L) + 2 FC)*1 + (4 FC)*1	7,187,851	1687.9	990.6	0.779	8.4	0.122	573,377	564.9	433.8	0.982	2.9	0.036	10,856,900	2197.1	1237.0	0.666	10.9	0.155	30.576	0.0172
	NA	24	(1 LSTM(A) + 2 FC)*1 + (1 LSTM(D) + 2 FC)*1 + (1 LSTM(L) + 2 FC)*1	7,062,396	1717.1	1071.5	0.783	8.6	0.121	787,803	674.8	528.4	0.976	3.4	0.042	11,162,300	2194.6	1231.7	0.657	11.1	0.157	27.577	0.0294
	NA	24	(1 LSTM(A) + 2 FC)*2 + (1 LSTM(D) + 2 FC)*2 + (1 LSTM(L) + 2 FC)*2 + (4 FC)*2	6,363,914	1649.2	1058.9	0.804	8.4	0.114	570,060	571.3	448.8	0.982	2.9	0.035	10,226,000	2156.0	1330.3	0.686	10.7	0.150	33.828	0.0294
	NA	24	(2 1D-CNN(A) +1 LSTM + 2 FC)*1 + (2 1D-CNN(D) +1 LSTM + 2 FC)*1 + (2 1D-CNN(L) +1 LSTM+ 2 FC)*1 + (4 FC)*1	5,313,763	1625.5	1168.0	0.837	8.3	0.106	1,589,250	952.1	746.2	0.951	4.8	0.059	8,367,550	2002.0	1312.2	0.743	10.1	0.136	9.082	0.0130
	NA	24	(2 1D-CNN(A) +1 LSTM + 2 FC)*2 + (2 1D-CNN(D) +1 LSTM + 2 FC)*2 + (2 1D-CNN(L) +1 LSTM+ 2 FC)*2 + (4 FC)*2	5,364,495	1637.0	1160.4	0.835	8.3	0.107	2,702,450	1351.7	1187.5	0.917	6.9	0.077	8,231,580	2008.4	1348.8	0.747	10.1	0.135	27.541	0.0229
	1	168	(1 LSTM + 2 FC)*2 + (4 FC)*2	3,064,396	1126.6	772.9	0.906	5.7	0.080	604,466	587.1	458.0	0.981	3.0	0.037	4,364,300	1326.7	861.1	0.866	6.7	0.098	95.918	0.0986
	45	168	(1 LSTM + 2 FC)*2 + (4 FC)*2	4,838,068	1461.5	961.9	0.852	7.3	0.102	1,164,250	801.2	611.4	0.964	4.1	0.051	6,988,970	1787.9	1168.9	0.786	8.9	0.124	72.698	0.0586
	NA	168	(1 LSTM(A) + 2 FC)*1 + (1 LSTM(D) + 2 FC)*1 + (1 LSTM(L) + 2 FC)*1 + (4 FC)*1	4,564,873	1392.9	916.9	0.860	7.0	0.098	743,137	650.9	498.2	0.977	3.3	0.040	6,565,770	1711.3	1107.2	0.799	8.6	0.120	37.157	0.0621
	NA	168	(1 LSTM(A) + 2 FC)*1 + (1 LSTM(D) + 2 FC)*1 + (1 LSTM(L) + 2 FC)*1	4,881,112	1484.2	1002.6	0.850	7.5	0.102	1,125,100	795.5	612.2	0.965	4.1	0.050	6,792,180	1759.1	1154.1	0.792	8.7	0.122	36.073	0.0596
	NA	168	(1 LSTM(A) + 2 FC)*2 + (1 LSTM(D) + 2 FC)*2 + (1 LSTM(L) + 2 FC)*2 + (4 FC)*2	4,323,387	1426.0	1003.4	0.867	7.2	0.096	802,789	688.9	561.9	0.975	3.5	0.042	6,515,730	1757.3	1140.2	0.800	8.8	0.120	78.647	0.0980
	NA	168	(2 1D-CNN(A) +1 LSTM + 2 FC)*1 + (2 1D-CNN(D) +1 LSTM + 2 FC)*1 + (2 1D-CNN(L) +1 LSTM+ 2 FC)*1 + (4 FC)*1	4,410,754	1493.2	1070.1	0.865	7.5	0.097	1,293,710	877.8	720.4	0.960	4.5	0.053	5,816,750	1724.5	1205.1	0.821	8.6	0.113	24.410	0.0713
	NA	168	(2 1D-CNN(A) +1 LSTM + 2 FC)*2 + (2 1D-CNN(D) +1 LSTM + 2 FC)*2 + (2 1D-CNN(L) +1 LSTM+ 2 FC)*2 + (4 FC)*2	4,623,346	1549.9	1149.6	0.858	7.9	0.099	1,110,690	815.2	654.9	0.966	4.3	0.049	6,889,680	1924.8	1406.1	0.789	9.7	0.123	34.370	0.1153
	1	720	(1 LSTM + 2 FC)*2 + (4 FC)*2	3,664,349	1316.3	983.3	0.888	6.6	0.088	1,115,960	838.7	699.7	0.966	4.2	0.050	4,391,940	1384.6	939.8	0.866	6.8	0.098	42.145	0.5970
	NA	720	(1 LSTM(A) + 2 FC)*1 + (1 LSTM(D) + 2 FC)*1 + (1 LSTM(L) + 2 FC)*1 + (4 FC)*1	4,443,107	1448.9	1033.2	0.864	7.3	0.096	671,410	627.3	507.5	0.979	3.3	0.038	6,591,150	1812.8	1236.8	0.798	9.2	0.120	107.037	0.7107

Figure A1. Forecast metrics for the average, first (T-0), and last (T-23) predicted time-slots for all models.

References

1. Nguyen, H.; Hansen, C.K. Short-term electricity load forecasting with Time Series Analysis. In Proceedings of the 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), Dallas, TX, USA, 19–21 June 2017; pp. 214–221.
2. Hernández, L.; Baladron, C.; Aguiar, J.M.; Carro, B.; Sanchez-Esguevillas, A.J.; Lloret, J.; Massana, J. A Survey on Electric Power Demand Forecasting: Future Trends in Smart Grids, Microgrids and Smart Buildings. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1460–1495. [CrossRef]
3. Schmid, P.J. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **2010**, *656*, 5–28. [CrossRef]
4. Tirunagari, S.; Kouchaki, S.; Poh, N.; Bober, M.; Windridge, D.; Dynamic, D.W. Dynamic Mode Decomposition for Univariate Time Series: Analysing Trends and Forecasting. 2017. Available online: https://hal.archives-ouvertes.fr/hal-01463744 (accessed on 18 June 2021).
5. Mohan, N.; Soman, K.P.; Sachin Kumar, S. A data-driven strategy for short-term electric load forecasting using dynamic mode decomposition model. *Appl. Energy* **2018**, *232*, 229–244. [CrossRef]
6. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A.; Lloret, J. Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things. *IEEE Access* **2017**, *5*. [CrossRef]
7. Lopez-Martin, M.; Carro, B.; Lloret, J.; Egea, S.; Sanchez-Esguevillas, A. Deep Learning Model for Multimedia Quality of Experience Prediction Based on Network Flow Packets. *IEEE Commun. Mag.* **2018**, *56*. [CrossRef]
8. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. *arXiv* **2014**, arXiv:1409.3215.
9. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv* **2014**, arXiv:1409.0473.
10. Luong, M.-T.; Pham, H.; Manning, C.D. Effective Approaches to Attention-based Neural Machine Translation. *arXiv* **2015**, arXiv:1508.04025.
11. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A. Neural network architecture based on gradient boosting for IoT traffic prediction. *Future Gener. Comput. Syst.* **2019**, *100*, 656–673. [CrossRef]
12. Bontempi, G.; Ben Taieb, S.; Le Borgne, Y.-A. Machine Learning Strategies for Time Series Forecasting BT-Business Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, 15–21 July 2012. In *Tutorial Lectures*; Aufaure, M.-A., Zimányi, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 62–77. ISBN 978-3-642-36318-4.
13. Bourdeau, M.; Zhai, X.Q.; Nefzaoui, E.; Guo, X.; Chatellier, P. Modeling and forecasting building energy consumption: A review of data-driven techniques. *Sustain. Cities Soc.* **2019**, *48*, 101533. [CrossRef]
14. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLoS ONE* **2018**, *13*, e0194889. [CrossRef]
15. Martin, M.L.; Sanchez-Esguevillas, A.; Carro, B.C. Review of Methods to Predict Connectivity of IoT Wireless Devices. In *Ad Hoc and Sensor Wireless Networks*; Old City Publishing, Inc.: Philadelphia, PA, USA, 2017; Volume 38, pp. 125–141.
16. Hernández, L.; Baladrón, C.; Aguiar, J.M.; Carro, B.; Sánchez-Esguevillas, A.; Lloret, J. Artificial neural networks for short-term load forecasting in microgrids environment. *Energy* **2014**, *75*, 252–264. [CrossRef]
17. Hernández, L.; Baladrón, C.; Aguiar, J.; Calavia, L.; Carro, B.; Sánchez-Esguevillas, A.; Pérez, F.; Fernández, Á.; Lloret, J. Artificial Neural Network for Short-Term Load Forecasting in Distribution Systems. *Energies* **2014**, *7*, 1576–1598. [CrossRef]
18. Hernández, L.; Baladrón, C.; Aguiar, J.; Calavia, L.; Carro, B.; Sánchez-Esguevillas, A.; Sanjuán, J.; González, Á.; Lloret, J. Improved Short-Term Load Forecasting Based on Two-Stage Predictions with Artificial Neural Networks in a Microgrid Environment. *Energies* **2013**, *6*, 4489–4507. [CrossRef]
19. Hernández, L.; Baladrón, C.; Aguiar, J.; Calavia, L.; Carro, B.; Sánchez-Esguevillas, A.; García, P.; Lloret, J. Experimental Analysis of the Input Variables' Relevance to Forecast Next Day's Aggregated Electric Demand Using Neural Networks. *Energies* **2013**, *6*, 2927–2948. [CrossRef]
20. Hernández, L.; Baladrón, C.; Aguiar, J.; Carro, B.; Sanchez-Esguevillas, A.; Lloret, J. Short-Term Load Forecasting for Microgrids Based on Artificial Neural Networks. *Energies* **2013**, *6*, 1385–1408. [CrossRef]
21. Hernández, L.; Baladron, C.; Aguiar, J.M.; Carro, B.; Sanchez-Esguevillas, A.; Lloret, J.; Chinarro, D.; Gomez-Sanz, J.J.; Cook, D. A multi-agent system architecture for smart grid management and forecasting of energy demand in virtual power plants. *IEEE Commun. Mag.* **2013**, *51*, 106–113. [CrossRef]
22. Hernández, L.; Baladrón, C.; Aguiar, J.; Carro, B.; Sánchez-Esguevillas, A. Classification and Clustering of Electricity Demand Patterns in Industrial Parks. *Energies* **2012**, *5*, 5215–5228. [CrossRef]
23. Hernández, L.; Baladrón, C.; Aguiar, J.M.; Calavia, L.; Carro, B.; Sánchez-Esguevillas, A.; Cook, D.J.; Chinarro, D.; Gómez, J. A Study of the Relationship between Weather Variables and Electric Power Demand inside a Smart Grid/Smart World Framework. *Sensors* **2012**, *12*, 11571–11591. [CrossRef]
24. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A. IoT type-of-traffic forecasting method based on gradient boosting neural networks. *Future Gener. Comput. Syst.* **2020**, *105*, 331–345. [CrossRef]
25. Wolpert, D.H. Stacked generalization. *Neural Netw.* **1992**, *5*, 241–259. [CrossRef]
26. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.

27. Frankle, J.; Carbin, M. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. *arXiv* **2018**, arXiv:1803.03635.
28. Jain, S.; Liu, G.; Mueller, J.; Gifford, D. Maximizing Overall Diversity for Improved Uncertainty Estimates in Deep Ensembles. *arXiv* **2019**, arXiv:1906.07380. [CrossRef]
29. Wang, W.; Lu, Z. Cyber security in the Smart Grid: Survey and challenges. *Comput. Netw.* **2013**, *57*, 1344–1371. [CrossRef]
30. Cui, L.; Qu, Y.; Gao, L.; Xie, G.; Yu, S. Detecting false data attacks using machine learning techniques in smart grid: A survey. *J. Netw. Comput. Appl.* **2020**, *170*, 102808. [CrossRef]
31. Zhang, L.; Wen, J.; Li, Y.; Chen, J.; Ye, Y.; Fu, Y.; Livingood, W. A review of machine learning in building load prediction. *Appl. Energy* **2021**, *285*, 116452. [CrossRef]
32. Zhao, Y.; Zhang, C.; Zhang, Y.; Wang, Z.; Li, J. A review of data mining technologies in building energy systems: Load prediction, pattern identification, fault detection and diagnosis. *Energy Built Environ.* **2020**, *1*, 149–164. [CrossRef]
33. Vivas, E.; Allende-Cid, H.; Salas, R. A Systematic Review of Statistical and Machine Learning Methods for Electrical Power Forecasting with Reported MAPE Score. *Entropy* **2020**, *22*, 1412. [CrossRef] [PubMed]
34. Aguilar Madrid, E.; Antonio, N. Short-Term Electricity Load Forecasting with Machine Learning. *Information* **2012**, *12*, 50. [CrossRef]
35. Nti, I.K.; Teimeh, M.; Nyarko-Boateng, O.; Adekoya, A.F. Electricity load forecasting: A systematic review. *J. Electr. Syst. Inf. Technol.* **2020**, *7*, 1–19. [CrossRef]
36. Taylor, J.W. An evaluation of methods for very short-term load forecasting using minute-by-minute British data. *Int. J. Forecast.* **2008**, *24*, 645–658. [CrossRef]
37. Dumas, J.; Cornélusse, B. Classification of load forecasting studies by forecasting problem to select load forecasting techniques and methodologies. *arXiv* **2018**, arXiv:1901.05052.
38. Hong, T.; Fan, S. Probabilistic electric load forecasting: A tutorial review. *Int. J. Forecast.* **2016**, *32*, 914–938. [CrossRef]
39. Hammad, M.A.; Jereb, B.; Rosi, B.; Dragan, D. Methods and Models for Electric Load Forecasting: A Comprehensive Review. *Logist. Sustain. Transp.* **2020**, *11*, 51–76. [CrossRef]
40. Kong, X.; Li, C.; Wang, C.; Zhang, Y.; Zhang, J. Short-term electrical load forecasting based on error correction using dynamic mode decomposition. *Appl. Energy* **2020**, *261*, 114368. [CrossRef]
41. Fan, C.; Ding, C.; Zheng, J.; Xiao, L.; Ai, Z. Empirical Mode Decomposition based Multi-objective Deep Belief Network for short-term power load forecasting. *Neurocomputing* **2020**, *388*, 110–123. [CrossRef]
42. Pinto, T.; Praça, I.; Vale, Z.; Silva, J. Ensemble learning for electricity consumption forecasting in office buildings. *Neurocomputing* **2020**, *423*, 747–755. [CrossRef]
43. Oprea, S.; Bâra, A. Machine Learning Algorithms for Short-Term Load Forecast in Residential Buildings Using Smart Meters, Sensors and Big Data Solutions. *IEEE Access* **2019**, *7*, 177874–177889. [CrossRef]
44. Jacob, M.; Neves, C.; Vukadinović Greetham, D. (Eds.) *Short Term Load Forecasting BT-Forecasting and Assessing Risk Ndividual Electricity Peaks*; Springer International Publishing: Cham, Switzerland, 2020; pp. 15–37. ISBN 978-3-030-28669-9.
45. Candelieri, A. Clustering and Support Vector Regression for Water Demand Forecasting and Anomaly Detection. *Water* **2017**, *9*, 224. [CrossRef]
46. Fan, S.; Hyndman, R.J. Short-Term Load Forecasting Based on a Semi-Parametric Additive Model. *IEEE Trans. Power Syst.* **2012**, *27*, 134–141. [CrossRef]
47. Gasparin, A.; Lukovic, S.; Alippi, C. Deep Learning for Time Series Forecasting: The Electric Load Case. *arXiv* **2019**, arXiv:1907.09207.
48. Gong, G.; An, X.; Mahato, N.K.; Sun, S.; Chen, S.; Wen, Y. Research on Short-Term Load Prediction Based on Seq2seq Model. *Energies* **2019**, *12*, 3199. [CrossRef]
49. Du, S.; Li, T.; Yang, Y.; Horng, S.-J. Multivariate time series forecasting via attention-based encoder–decoder framework. *Neurocomputing* **2020**, *388*, 269–279. [CrossRef]
50. Huang, Y.; Wang, N.; Gao, W.; Guo, X.; Huang, C.; Hao, T.; Zhan, J. LoadCNN: A Low Training Cost Deep Learning Model for Day-Ahead Individual Residential Load Forecasting. *arXiv* **2019**, arXiv:1908.00298.
51. Khotanzad, A.; Afkhami-Rohani, R.; Maratukulam, D. ANNSTLF-Artificial Neural Network Short-Term Load Forecaster-generation three. *IEEE Trans. Power Syst.* **1998**, *13*, 1413–1422. [CrossRef]
52. Farsi, B.; Amayri, M.; Bouguila, N.; Eicker, U. On Short-Term Load Forecasting Using Machine Learning Techniques and a Novel Parallel Deep LSTM-CNN Approach. *IEEE Access* **2021**, *9*, 31191–31212. [CrossRef]
53. Atef, S.; Eltawil, A.B. Assessment of stacked unidirectional and bidirectional long short-term memory networks for electricity load forecasting. *Electr. Power Syst. Res.* **2020**, *187*, 106489. [CrossRef]
54. Mann, J.; Kutz, J.N. Dynamic Mode Decomposition for Financial Trading Strategies. *arXiv* **2015**, arXiv:1508.04487. [CrossRef]
55. Dylewsky, D.; Barajas-Solano, D.; Ma, T.; Tartakovsky, A.M.; Kutz, J.N. Dynamic mode decomposition for forecasting and analysis of power grid load data. *arXiv* **2020**, arXiv:2010.04248.
56. He, X.; Zhao, K.; Chu, X. AutoML: A survey of the state-of-the-art. *Knowl. Based Syst.* **2021**, *212*, 106622. [CrossRef]
57. Zöller, M.-A.; Huber, M.F. Benchmark and Survey of Automated Machine Learning Frameworks. *J. Artif. Int. Res.* **2021**, *70*, 409–472. [CrossRef]
58. Hutter, F.; Kotthoff, L.; Vanschoren, J. (Eds.) *Automated Machine Learning: Methods, Systems and Challenges*; The Springer Series on Challenges in Machine Learning; Springer International Publishing: Cham, Switzerland, 2019; ISBN 978-3-030-05317-8.

59. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* **2016**, arXiv:1603.04467.
60. Pedregosa, F.; Michel, V.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Vanderplas, J.; Cournapeau, D.; Pedregosa, F.; Varoquaux, G.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
61. Fort, S.; Hu, H.; Lakshminarayanan, B. Deep Ensembles: A Loss Landscape Perspective. *arXiv* **2019**, arXiv:1912.02757.
62. Vaswani, A.; Brain, G.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.