Electric Power Systems Research 190 (2021) 106756

Contents lists available at ScienceDirect

Electric Power Systems Research

journal homepage: www.elsevier.com/locate/epsr

A deep learning method for forecasting residual market curves[☆]

Alex Coronati, José R. Andrade, Ricardo J. Bessa[*]

INESC TEC, Institute for Systems and Computer Engineering, Technology and Science Porto, Portugal

ARTICLE INFO

Keywords:
Residual demand curve
Forecasting
Deep learning
Electricity market

ABSTRACT

Forecasts of residual demand curves (RDCs) are valuable information for price-maker market agents since it enables an assessment of their bidding strategy in the market-clearing price. This paper describes the application of deep learning techniques, namely long short-term memory (LSTM) network that combines past RDCs and exogenous variables (e.g., renewable energy forecasts). The main contribution is to build up on the idea of transforming the temporal sequence of RDCs into a sequence of images, avoiding any feature reduction and exploiting the capability of LSTM in handling image data. The proposed method was tested with data from the Iberian day-ahead electricity market and outperformed machine learning models with an improvement of above 35% in both root mean square error and Frèchet distance.

1. Introduction

Electricity price forecasting, in different time horizons (intraday, day-ahead, mid-term, etc.), is fundamental for advanced bidding strategies. This motivated research in the last 20-years about point and probabilistic forecasts for the market-clearing price [1]. Moreover, modelling and forecasting competitors' bidding curves is also very valuable to optimize bidding strategies. This information is captured by the residual demand curve (RDC), which, for a generation company, is the difference between the demand curve and the aggregated curve of all the competitors' supply offers [2]. RDC expresses the market-clearing price as a function of the quantity of energy that the agent is willing to buy or sell, which is particularly valuable for price-maker companies since it enables an assessment of their bidding strategy in the market-clearing price

Different authors explored information from forecasted or estimated RDC in order to build advanced bidding strategies [3]. Regarding the supply side, Wolak proposed an optimal bidding strategy that used an analytical function for the RDC [4]; Contreras et al. proposed a Cournot-based model to find the optimal bids of a company with thermal and hydro units, using an RDC represented by a Taylor's linear approximation around the clearing price [5]; Ugedo et al. described a stochastic programming formulation that used RDCs constructed through clustering and approximated by a linear regression [6]. In the demand side, Herranz et al. applied genetic algorithm in order to optimize the demand bids using a sliding-window and Monte Carlo simulation of residual offer curves – ROC (i.e., RDC from the energy buyer viewpoint) [7]; Philpott et al. studied different bidding strategies by using an analytical ROC that provides the probability of having a demand offer accepted [8].

All these works either applied a naive estimation of the RDC or extrapolated future RDC by clustering past data and do not solve a typical forecasting problem. More advanced RDC forecasting strategies were proposed in the literature. Aneiros et al. described two functional data analysis (FDA) models to forecast hourly RDC [9], one non-parametric autoregressive model and a semi-functional partial linear model that integrates exogenous scalar variables. Portela et al. extended the seasonal ARIMA model to the FDA framework by using parametric Hilbert operators with kernel surfaces to integrate exogenous variables [10]. Other models based on FDA for residual curve forecasting can be found in [11]. Portela *et al.*, for the Iberian electricity market data, showed that principal component analysis combined with transfer function outperformed FDA in terms of forecasting accuracy [12].

In this context, the present paper provides the following original contributions: (a) builds up on the idea of transforming the temporal sequence of RDC into a sequence of images, avoiding any feature reduction; (b) proposes a hybrid approach that combines one dimensional and convolutional long short-term memory (LSTM) networks to produce highly accurate day-ahead forecasts and to outperform traditional machine learning models (e.g., gradient boosting trees).

[☆] This work is financed by the ERDF - European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT – Fundação para a Ciência e a Tecnologia, within project ESGRIDS – Desenvolvimento Sustentável da Rede Elétrica Inteligente/SAICTPAC/0004/2015-POCI-01-0145-FEDER-016434.

[*] Corresponding author. fax: +351 22 209 4050.
E-mail addresses: alex.coronati@inesctec.pt (A. Coronati), jose.r.andrade@inesctec.pt (J.R. Andrade), ricardo.j.bessa@inesctec.pt (R.J. Bessa).

https://doi.org/10.1016/j.epsr.2020.106756
Received 27 September 2019; Received in revised form 27 March 2020; Accepted 2 August 2020
Available online 08 August 2020
0378-7796/ © 2020 Elsevier B.V. All rights reserved.

The rest of the paper proceeds as follows. Section 2 details the formulation and forecasting methodology, while Section 3 validates the forecasting model for the Iberian market. Finally, Section 4 presents the conclusions and future work.

2. Deep learning forecasting methodology

This section describes an RDC day-ahead forecasting framework based on recent developments in the deep learning field, namely long short-term memory (LSTM) networks capable of handling input time series of data with one or two-dimensional representation.

2.1. Formulation of the RDC forecasting problem

The publication of the regulation 543/2013 in the European Union increased the amount of publicly available data from the electricity market, including access to individual offers from market players (in general, available with a few months delay). Moreover, aggregated supply and demand curves are publicly available on a daily basis and can be forecasted as RDC (i.e., difference between the two curves).

The goal of the RDC forecasting problem is to use the past sequence of RDCs observations from the day-ahead market, combined with exogenous variables (e.g., renewable energy sources – RES generation, load), in order to forecast the 24 RDCs for the next day ($D + 1$). Mathematically, it can be represented as follows:

$$\hat{Y}_{t+1|t}, \ldots, \hat{Y}_{t+L|t} = f\,(Y_t,\ Y_{t-1}, \ldots, \mathrm{X}_{t-24},\ \mathrm{X}_{t-25}, \ldots, \hat{Z}_{t+1|t},\ \ldots \hat{Z}_{t+L|t}) \tag{1}$$

where Y_t, Y_{t-1}, ..., is an input sequence from the RDC past data, X_{t-24}, ..., is a time series of past exogenous variables, $\hat{Z}_{t+1|t}$, ...$\hat{Z}_{t+L|t}$ is the vector of forecasted exogenous variables and $\hat{Y}_{t+1|t}$, ..., $\hat{Y}_{t+L|t}$ is the output (forecasted) sequence up to time horizon L. It is important to underline that Eq. 1 represents a sequence forecast (or structural forecast), and not a single-step forecast like in traditional time series forecasting problems.

In contrast to the spot price variable, an RDC is represented in a two-dimensional variable space (energy quantity in the x-axis, price in the y-axis), which has two alternative representations from a machine learning perspective: (a) 1-Dimensional (1D), discretization of the RDC, using a fixed range for the price, into a set of K independent time series; (b) 2-Dimensional (2D), convert the full RDC curve into a black and white image represented by a single matrix of pixel values that contains the "spatial" dependency structure between energy and price.

The 1D representation is a matrix of K numerical time series, while in the 2D we have a tensor $Y \in \Re^{M \times J \times P}$ where M and J are rows and columns of grid representing the RDC image and P is the pixel value that only takes 0 or 1 values and vary over time.

2.2. RDC forecasting structure

This section starts by discussing the advantages and disadvantages of the 1D and 2D representations.

In the 1D, the time series are forecasted independently by a multi-output model. This means that, while temporal dependencies can be captured during the forecasting process (i.e., dependency between past and future RDCs), the "spatial" shape of the curve is not properly modelled. Through an increase in the value of K, it is possible to represent each curve with a higher number of samples, thus enhancing the representation quality. However, this increases the number of time-series and consequently the computational requirements. The main advantages of the 1D representation are: (a) simplicity of the forecasting process (e.g., traditional machine learning algorithms can be applied); (b) it is easy to include multiple exogenous variables besides the past RDC; (c) the forecasted curve can be obtained with a linear interpolation of the output vector.

In the 2D, the quantity-price relationships of RDC are embedded in 2D images through an initial pre-processing phase (explained in Section 2.4). With this representation, it is possible to structure the forecasting problem as a video (or motion) forecasting problem where every RDC image represents a video frame and multi-frame sequences can be used to predict the subsequent video frames. Thus, it is possible to exploit the superior capacity of Convolutional LSTM (ConvLSTM) [13] and 3D convolutional neural-networks (3D CNN) [14] to capture both temporal and spatial dependencies encoded in sequences of RDC images, in contrast to the independent time-series processing provided by the 1D representation. The main disadvantages of this representation are: a) processing steps required to convert the original sequences of RDC matrices into sequences of images (and vice-versa) which might introduce additional noise in the final forecasts; b) the inclusion of relevant exogenous scalar variables (e.g., forecasted wind power) is not straightforward; c) higher computational requirements and processing times compared to the 1D approach. In this work, the LSTM processing times were reduced by taking advantage of a graphics processing unit (GPU) through computational frameworks like Keras [15] and TensorFlow [16].

Fig. 1 depicts the two methods employed to produce day-ahead forecasts of RDCs. It considers two alternative paths (compared in Section 3): a) 1D path (left) where past RDC observations are combined with exogenous variables in a sequence-to-sequence learning framework (i.e., 1D-LSTM described in Section 2.3) to produce RDC forecasts; b) 2D path (right) where temporal and spatial correlations of contiguous RDCs images are first modeled by a stack of ConvLSTM and 3D CNN. The output tensor of this intermediate model is then converted into the 1D representation and combined with the past RDCs data and exogenous information to produce high quality forecasts (2D-ConvLSTM described in Section 2.4);

The main motivation for the final hybrid model (i.e., 2D path of Fig. 1) is to first exploit the superior capability of LSTM cells enhanced by embedded convolution operations to handle the quantity-prices relationships of RDC images sequences and then use the 1D-LSTM architecture in order to combine the past and future RDCs with exogenous variables, which have a relevant impact in the forecasting skill (as illustrated in Section 3).

2.3. 1-Dimensional LSTM model (1D-LSTM)

As mentioned before, RDCs are temporal sequences and this motivates the use of LSTM as a natural extension of deep neural networks to time series data since, like a traditional recurrent neural network (RNN), it employs loops that allow information from previous temporal intervals to persist. This section will present the 1D-LSTM forecasting methodology as intuitively as possible and a more detailed description of the mathematics behind LSTM can be found in [17] and Appendix A.

A major limitation assigned to RNN is that, in practice, when the gap between past relevant information and each forecast time grows, the performance of RNN tends to decline, thus resulting in a short-term memory that is known as vanishing gradient problem [18]. LSTM were proposed in [17] in order to overcome the long-term dependency problem and have mechanisms (i.e., called gates) to regulate the flow of information and that are able to learn which data in the sequence are important to keep or discard. In order to apply LSTM, as illustrated in Fig. 1, a certain number of reference prices are selected as fixed quantities making it possible to sample the curves in the selected points while reducing them to one-dimensional vectors of the corresponding energy values.

The LSTM model is capable of exploring the information encompassed in the autocorrelation structure of these RDC time series. However, as shown in [19], spot price is dependent on different variables, such as system load, weather variables and conventional generation. For instance, wind and solar energy can bid at very low prices

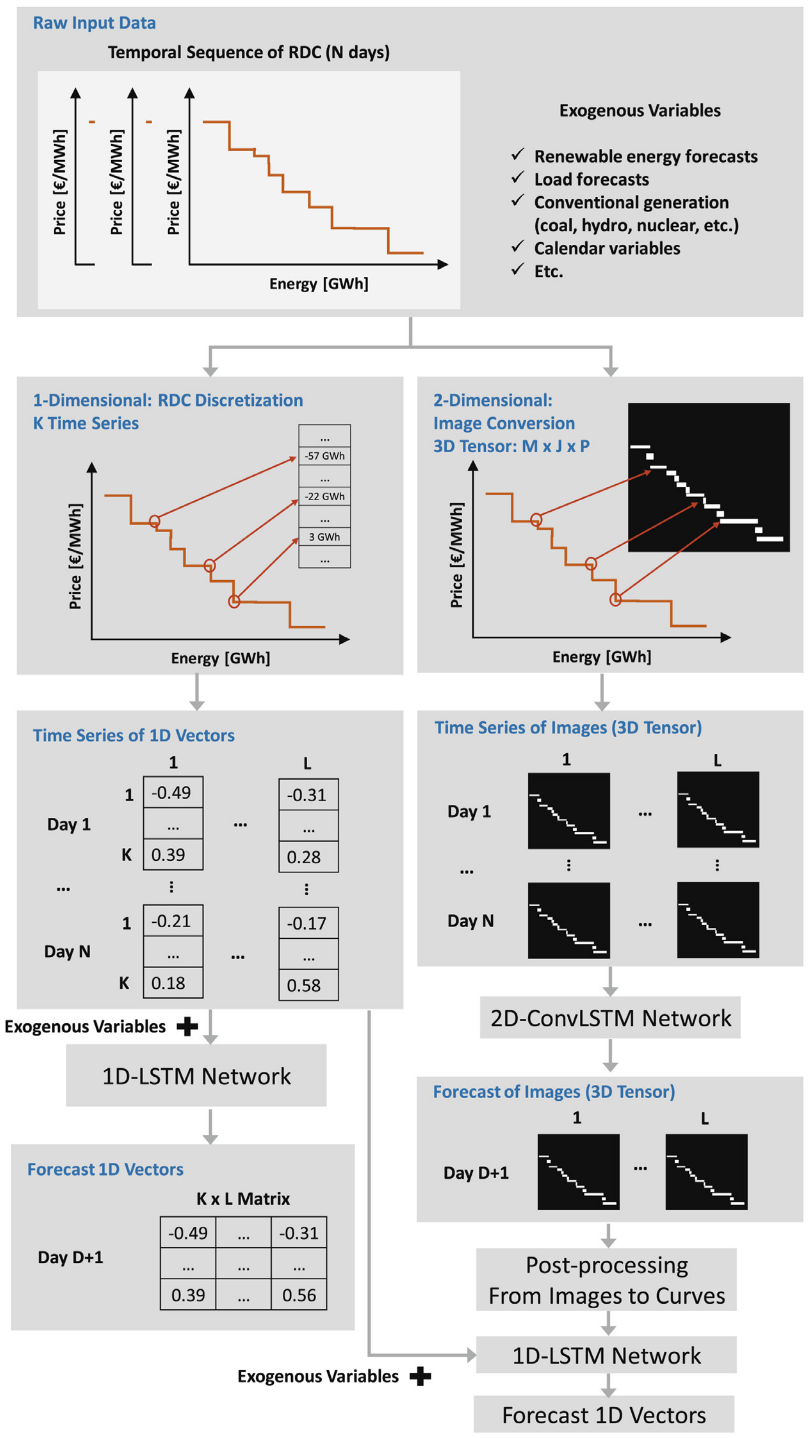

Fig. 1. RDC forecasting framework with long short-term memory (LSTM) as core forecasting method.

(in most cases zero), creating a horizontal displacement in the RDC. Thus, LSTM should be able to process not only the past RDC, but also the exogenous variables. Moreover, in order to capture the daily, weekly and yearly seasonality from spot prices, the following calendar features have been included and decomposed in sine and cosine components: a) month of the year (i.e., values from 1 to 12); b) day of the month; c) hour of the day; d) day of the week (i.e., values from 0 to 6, where 0 is Monday).

These three groups of input data (i.e., RDC time series, exogenous variables and calendar variables) are combined in the 1D-LSTM network structure, as depicted in Fig. 2.

In the upper part, the 1D-LSTM is composed of three parallel branches (one for each group of variables) followed by a stack of three LSTM layers and a fully-connected set of feedforward units as the final output layer.

Processing different input variables in independent branches has been previously used in order to improve the performance of deep learning models since it allows the network to capture intermediate representations of each original group of variables [20]. In this case, each branch is characterized by a set of independent weights (i.e., weights connected to one branch do not influence the activation of neurons on neighbor branches) that are adjusted to map the relations

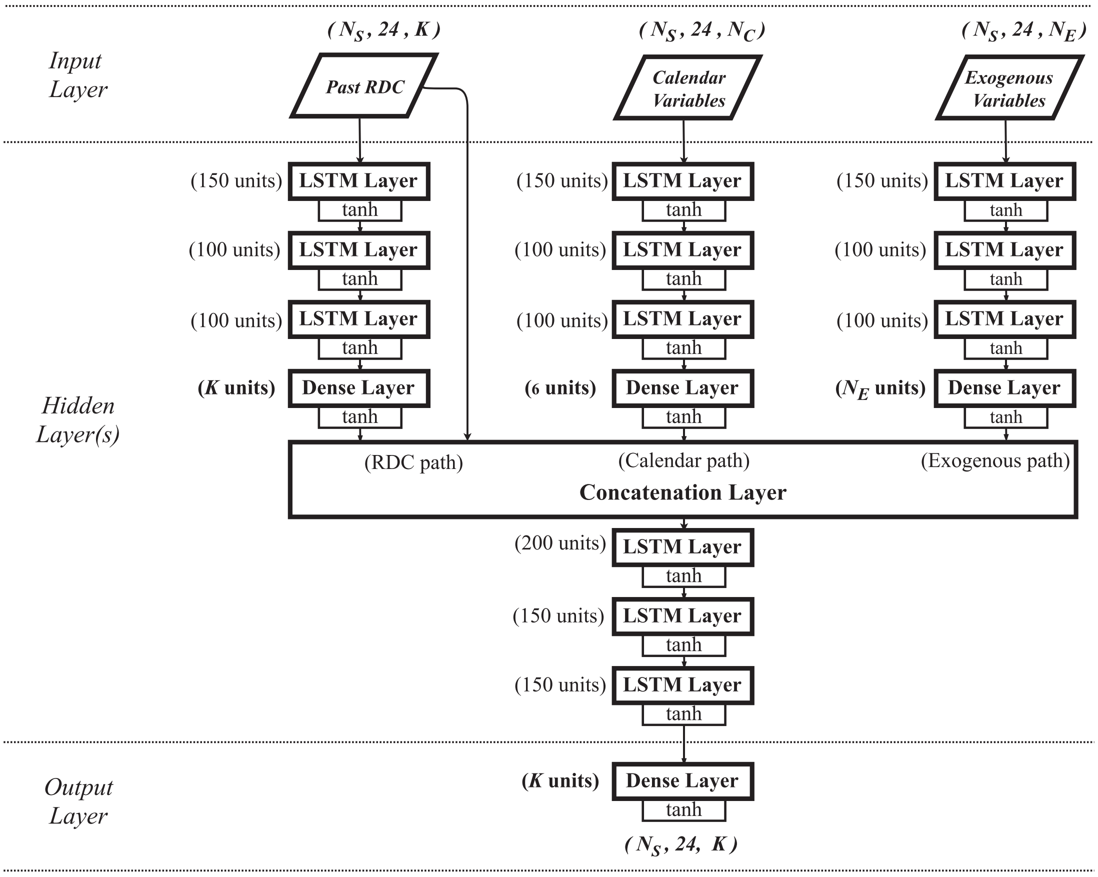
Input Layer
(N_S , 24 , K)
Past RDC
(N_S , 24 , N_C)
Calendar Variables
(N_S , 24 , N_E)
Exogenous Variables
Hidden Layer(s)
(150 units) LSTM Layer tanh
(100 units) LSTM Layer tanh
(100 units) LSTM Layer tanh
(K units) Dense Layer tanh
(6 units) Dense Layer tanh
(N_E units) Dense Layer tanh
(RDC path)
(Calendar path)
(Exogenous path)
Concatenation Layer
(200 units) LSTM Layer tanh
(150 units) LSTM Layer tanh
(150 units) LSTM Layer tanh
Output Layer
(K units) Dense Layer tanh
(N_S , 24, K)

Fig. 2. Inner structure of 1D-LSTM network; N_S represents the total number of samples (days) in each tensor and K, N_C and N_E represents the number of input features of each branch. The second dimension of each tensor represents the length of each sample sequence.

between their own input data and the common desired output.

As depicted in Fig. 2, the output from each branch has the same dimensionality as the corresponding input structures, i.e., tensors with N_S samples, each sample composed of 24 sequences of K, N_C or N_E features, depending on the group of variables. These intermediate output structures are then combined in order to create a single tensor $Y \in \Re^{N_S \times 24 \times T}$ where $T = K + N_C + N_E$. This information is then processed by a stack of three LSTM layers and a fully connected layer with K feedforward units in order to match the number of time series of the observed RDC and to compute the loss function.

The number of neurons for each layer was fine-tuned by an exhaustive exploratory process combining grid-search techniques with a sensitivity analysis to fine-tune the number of units for specific intermediate layers and to enhance the feature learning capabilities of the network. All neurons used a hyperbolic tangent activation function and a dropout regularization value of 0.2 has been chosen to prevent overfitting. The LSTM weights of every layer were initialized from a Grolot uniform initializer [21].

The LSTM weights and bias were optimized using the adaptive learning rate method “RMSprop” [22]. A batch train that was performed over the full training samples and the number of training epochs was determined via early stopping with a validation set (20% from the training sample). Then, an online learning method was used to update internal parameters of the LSTM by retraining it over each new observed sample of 24 RDCs, once they became available.

2.4. 2-Dimensional LSTM model (2D-ConvLSTM)

In the 2D presentation, the first step is to define the image dimensions, and consequently the resolution that the pixel matrices should have. Since all RDC-based images need to have the same size in order to be processed inside the 2D-ConvLSTM network (mathematical details can be found in [13] and Appendix B), the global maximum and minimum energy of all the curves was calculated to define the RDC energy span in GWh and the adopted resolution is one pixel per 1 GWh. The image height is analogously defined by the price span on which all the RDC lie and the adopted resolution is one pixel per 1 € /MWh.

After transforming the original RDCs into a 2D image, it is possible to use complex structures such as convolutional layer to extract low dimensional features from the original image data and properly capture the “spatial” shape of the RDCs.

The CNN architecture used in this work is illustrated in Fig. 3 and is composed of a stack of three convolutional LSTM layers followed by a single 3D CNN layer. Equal-sized convolution windows (11x11 kernels) were defined for all the ConvLSTM layers with the number of kernels varying (i.e., 40 for the first layer and 30 for the second and third layers). Finally, the 3D CNN layer applies a 3 dimensional filter that is able to move over the 3-axis so that both spatial and temporal dimensions are captured.

It is important to underline that an half padding technique (see [23] for more details) is applied to the feature map output of each convolutional layer in order to preserve the same input/output dimensions. This facilitates the loss computation and avoids the use of extra layers to reshape the output tensors to the same dimensionality of the target.

Since the 2D-ConvLSTM training is extremely time-consuming (described in Section 3.6), only minimal tuning of the LSTM parameters and number of ConvLSTM layers were performed, exclusively based on a trial-error analysis.

The weights and biases were optimized using the adaptive learning rate method “ADADELTA” [24] and a binary cross-entropy loss. A batch train was applied (and only once) to the 2D-ConvLSTM and online learning process is only considered for the final 1D-LSTM model

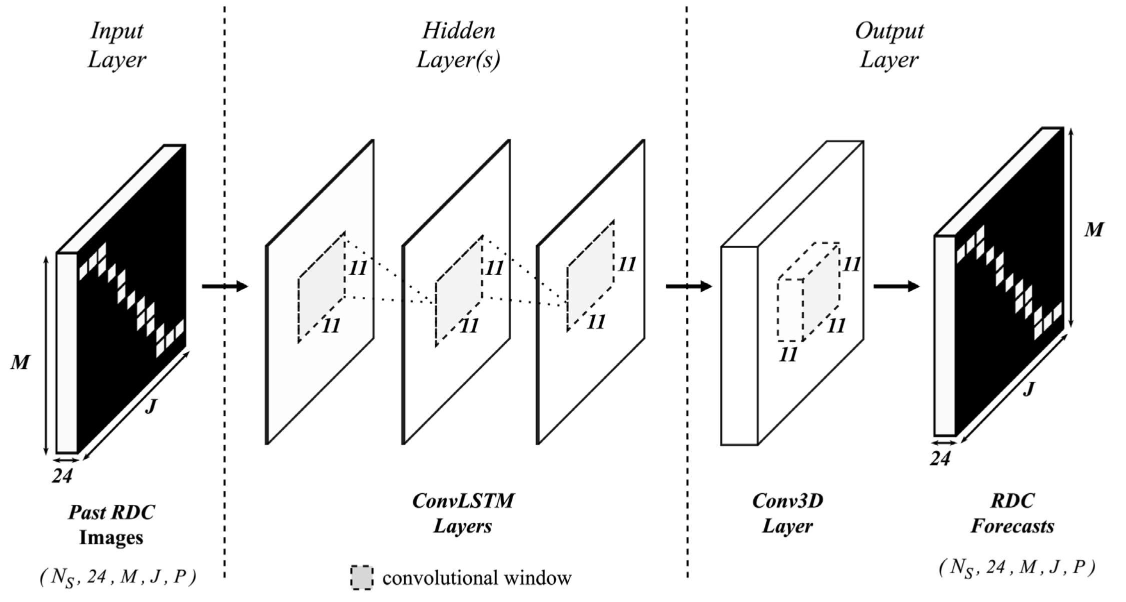

Fig. 3. Inner structure of 2D-ConvLSTM network. Hyperbolic tangent functions are used in each ConvLSTM layer and sigmoid in the final 3D CNN layer.

(trained independently).

As previously mentioned, in 2D-ConvLSTM model, a post-processing step is required to convert the forecasted RDC image back to a point-wise representation. Besides, this post-processing also includes a correction for the 3D CNN output, as illustrated in Fig. 4. This figure shows that pixel matrices obtained directly from this convolutional layer, do not translate into a neat pixel curve, but into a ”blurry” version of it. Thus, in order to obtain point-wise representation, a correction is performed to every row of each matrix (i.e., price pixel axis) that keeps and rounds the value of pixel with the highest intensity to one and sets the remaining pixels value to zero.

As interesting development for future work is to study if this “blurry” image output can be used to construct an uncertainty forecast for the RDC, e.g. multiple short-term scenarios for the curves, quantile curves.

Finally, as mentioned in Section 2.2, the RDCs forecasted by the 2D-ConvLSTM are then used by a final 1D-LSTM with a structure similar to Fig. 2, but with a fourth branch to process these forecasts as input data.

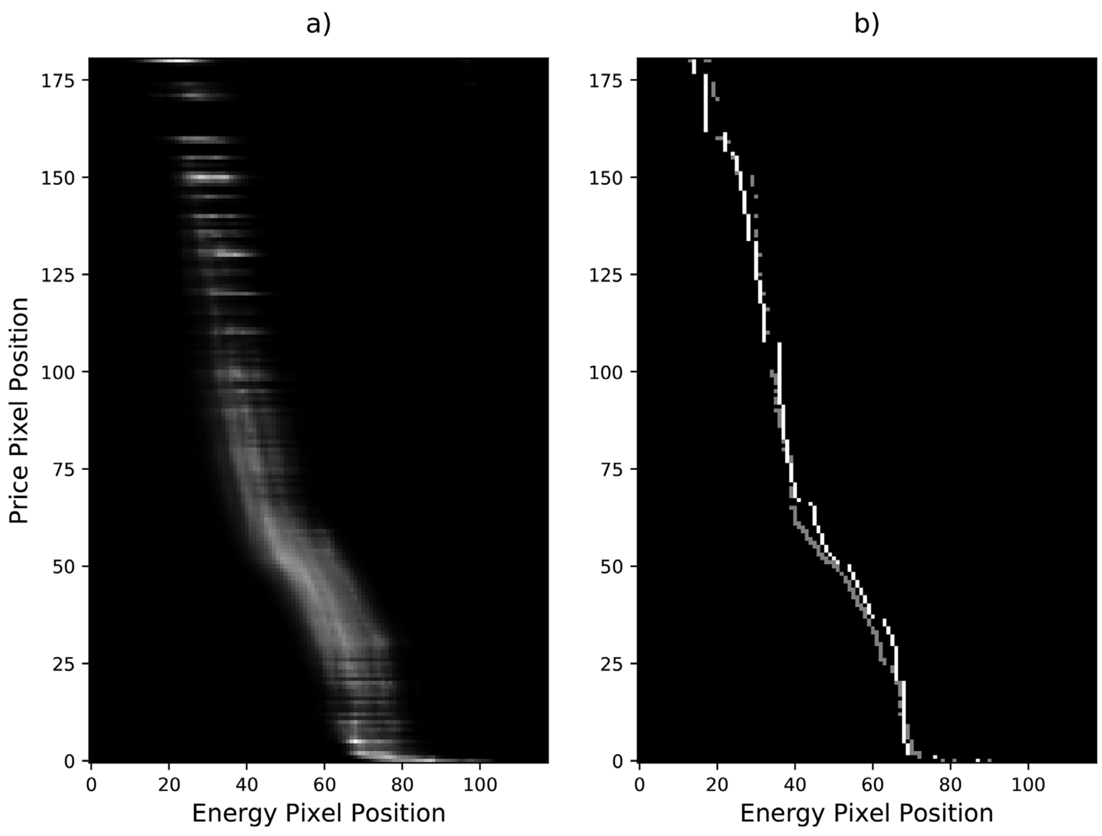

Fig. 4. 2D-ConvLSTM post-processing: (a) original forecast; (b) post-processed forecast (white) and observed curves (grey).

3. Numerical results

3.1. Dataset description: Iberian electricity market

The RDC forecasting methods are tested in data from the Iberian day-ahead electricity market (MIBEL), which makes aggregated supply and demand curves available in a public repository every day[1]. Since the day-ahead session receives offers for day D + 1 until noon, the RDC are forecasted with the information available until 11h00 in day D, which are the exogenous variables listed in Table 1.

Historical data was gathered from the ENTSO-E Transparency platform[2]. RES and load forecasts were extracted from the REN and REE data platforms[3]. The historical period considered in this work ranges between 1st November 2015 and 30th November 2018: (a) the period between 1st November 2015 and 24th March 2018 is divided in 70% for training and the other 20% for validation; (b) the period between 25th March 2018 and 30th November 2018 is used for testing.

The price in MIBEL is between 0 and 180.3 € /MWh, which was used to set $K = 182$ in the 1D approach (i.e., 1 € /MWh increment between 1D time series). For the historical period under analysis, the RDC energy values range between -69.1 GWh and 41.6 GWh, which means that, in the 2D approach, an energy range of 110.8 GWh is used to define the images resolution of 1 pixel per 1€ /MWh and 1 GWh.

3.2. Forecasting benchmark models

In order to evaluate the forecasting skill of the proposed deep learning approach, the following benchmark models of different complexity were implemented:

- Naïve 1: forecast equal to the RDC observed for day D.
- Naïve 2: average RDC from the three previous days.
- Artificial Neural Network (ANN): single-layer neural network that uses the 1D input data.
- Principal Components Analysis (PCA) and ANN (PCA + ANN): applies PCA to the RDCs by reducing them to only three principal components (95% of the variance), which are combined with calendar and exogenous variables in an ANN.
- PCA and Gradient Boosting Trees Regressor (PCA + GBTR): applies GBTR over the PCA representation of the RDC.

The grid search function with cross-validation from scikit-learn machine learning Python library [25] was applied to optimize the hyper-parameters (number of layers, number of neurons, etc.) of all these models.

3.3. Forecasting skill metrics

The RDCs forecasting skill is evaluated through two metrics: root mean square error (RMSE) and Frèchet distance.

The RMSE, traditionally used to measure forecast accuracy, corresponds to the square root of the mean square error between forecasted and observed values. Since the vector of prices is fixed between 1 and 182 € /MWh (1 € /MWh increment), RMSE is only applied between the real and the forecasted energy values and normalized by the maximum energy value.

The Frèchet distance measures the similarity between observed and forecasted RDC by taking into account the location and ordering of the points along the curves. In this metric, RDCs are piecewise linear curves with K vertices.

[1] OMIE website: www.omie.es

[2] ENTSO-E (transparency.entsoe.eu)

[3] REN Centro De Informação (www.mercado.ren.pt), REE ESIOS (www.esios.ree.es)

Table 1
Exogenous variables for Portugal and Spain. $D - 1$ are values observed in the previous day and $D + 1$ are day-ahead forecasts.

Availability	ID	Variables	Country
D-1	C	Coal Generation	ES
	CC	Combined Cycle Generation	ES
	NG	Nuclear Generation	ES
	H_{RC}	Hydro Reservoir Capacity	PT, ES
	H_D	Hydro Dam Generation	PT, ES
	H_{RR}	Hydro Run-river Generation	PT, ES
D + 1	SL	System Load Forecasts	PT, ES
	FW	Wind Power Forecasts	PT, ES
	FW_{Pen}	Wind Power Penetration Forecasts	PT, ES
	FS_{PV}	Solar Power Forecasts	PT, ES
	FS_{Pen}	Solar Power Penetration Forecasts	PT, ES

Considering two RDCs A and B, the vertices in A are represented by $\alpha \in [0, K]$ and the ones in B by $\beta \in [0, K]$. Thus, the associated Frèchet distance will be the smallest distance among all the maximum distances found by varying the position of α or β, between [0,K], while keeping the other one constant. The mathematical formulation between A and B is as follows:

$$\delta_F = \min \max_{\alpha \in [0,K], \beta \in [0,K]} \left\{ d\left(A(\alpha),\ B(\beta) \right) \right\} \tag{2}$$

where $d(A(\alpha), B(\beta))$ is the Euclidean distance between $A(\alpha)$ and $B(\beta)$.

For both metrics, a percentage of improvement (Imp) over a reference model (REF) is computed as follows:

$$Imp = (RMSE_{REF} - RMSE_{LSTM})/RMSE_{REF} \tag{3}$$

3.4. Sensitivity analysis with exogenous variables

A sensitivity analysis was conducted to select the group of exogenous variables with higher explanatory power to forecast RDC, besides past RDCs and calendar variables. Due to computational time limitations, it was not possible to test all the combinations of exogenous variables like in [19]. Thus, the variables from Table 1 were gathered in five groups: (1) System load forecasts – SL; (2) RES forecasts (RES_f) – FW, FS_{PV}; (3) RES penetration forecasts (RES_{Pen}) – FW_{Pen}, FS_{Pen}; (4) hydropower generation (HY) – H_{RC}, H_D, H_{RR}; (5) conventional generation (Conv) – C, CC and NG.

Table 2 shows the RMSE improvement obtained with the top five combinations of exogenous variables groups when compared to an LSTM model without exogenous information. These results show that the most relevant exogenous variables are RES forecasts and hydropower generation.

Fig. 5 depicts the RDC forecast obtained by a 1D-LSTM model without exogenous variables (blue curve) and by an LSTM with load and RES forecasts (green curve), and the observed RDC curve (red curve) for hour 13h00 of 26th October 2018. This day was selected since the RES penetration was 73% and the low bid price from RES shifted the whole RDC to the left. From the figure, it is clear the influence from exogenous information in the RDC forecast, which contrasts with Fig. 6 for the first hour of the 13th November 2018 characterized by a RES penetration of 22% and a higher spot price. In the case of Fig. 6, the difference between the RDC forecasted by the two models is marginal.

3.5. Benchmark models results

The average RMSE, Frèchet distance and the improvement of the 1D and hybrid approaches over the benchmark models are presented in Tables 3 and 4 for the eight months testing period. The average RMSE and Frèchet distance of the 1D approach are 1.45% and 1.47% correspondingly, and for the hybrid approach are 1.33% and 1.41%. The 1D-

Table 2
Sensitivity analysis results.

Groups of features	RMSE Improvement
SL + $\mathbf{RES_f}$ + $\mathbf{RES_{Pen}}$ + HY + Conv	**37.6%**
RES_f + RES_{Pen} + HY + Conv	36.7%
SL + RES_f + RES_{Pen} + HY	33.9%
RES_f + RES_{Pen} + HY	33.7%
RES_f + HY	33.5%

Table 3
Benchmark models forecasting skill - RMSE.

Model	RMSE	1D Imp.	Hybrid Imp.
Naïve 1	2.91%	40.1%	44.4%
Naïve 2	3.2%	45.5%	49.4%
ANN	2.75%	36.2%	40.1%
PCA + ANN	2.84%	38.7%	43.1%
PCA + GBTR	2,78%	37.4%	41.9%

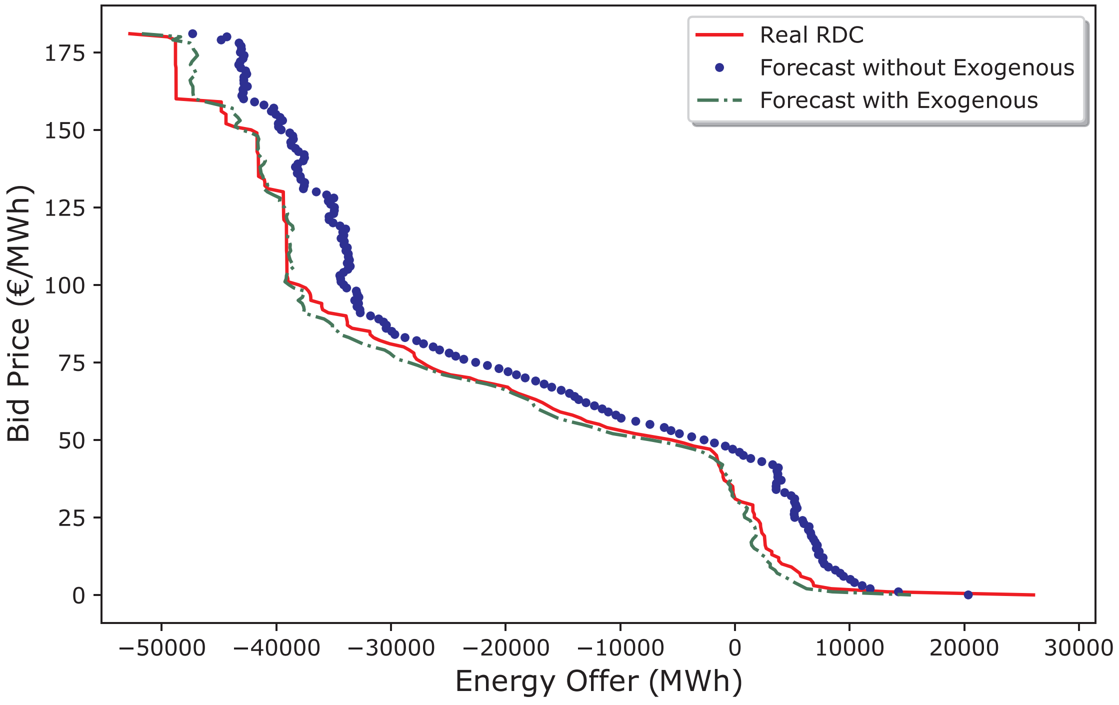

Fig. 5. Forecasts comparison for hour 13h00 of 26th October 2018.

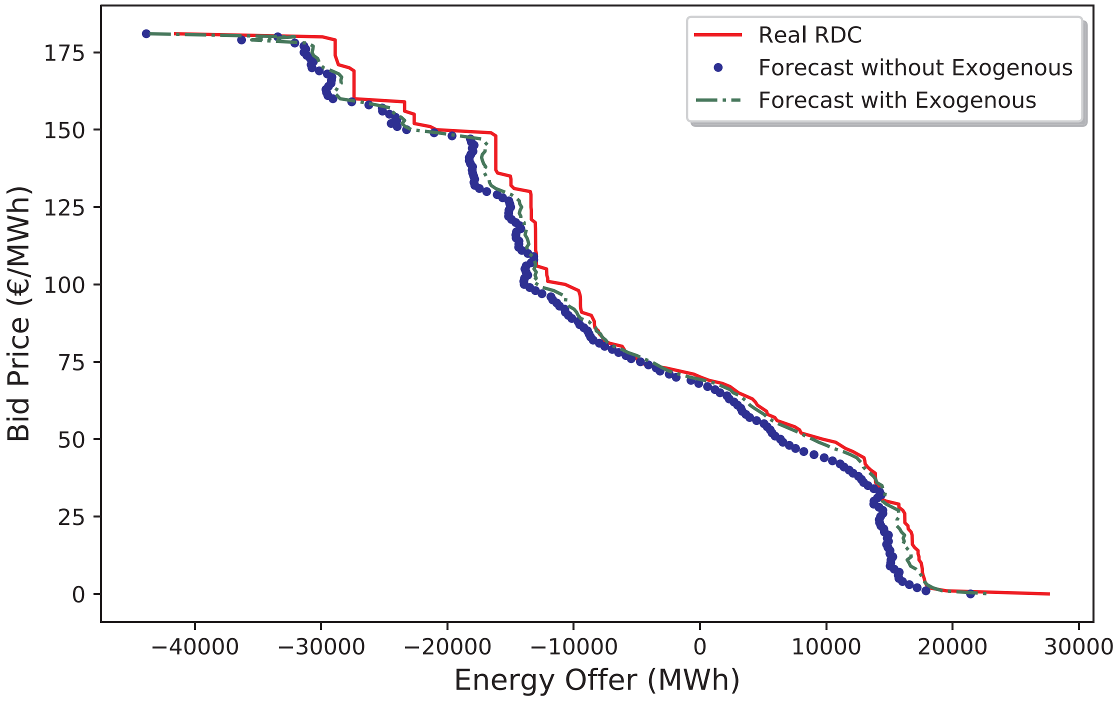

Fig. 6. Forecasts comparison of the 1st hour of 13th November 2018.

Table 4
Benchmark models forecasting skill - Frèchet distance.

Model	Frèchet	1D Imp.	Hybrid Imp.
Naïve 1	3.64%	53.7%	54.8%
Naïve 2	4.15%	59.4%	60.3%
ANN	3.92%	56.9%	58%
PCA + ANN	3.53%	52.2%	53.3%
PCA + GBTR	3.22%	47.5%	48.8%

LSTM and hybrid models largely outperformed the benchmark models, with improvements above 35% in both metrics.

It is important to note that the performance of the Naïve 1 model is close to the one obtained with advanced models, such as ANN, which confirms the results from [9]. Moreover, the models' ranking changes with the evaluation metric. For instance, the ANN model shows the lowest RMSE value, but presents the highest Frèchet distance. The main reason resides in the Frèchet distance ability to assess curve shape similarly, thus penalizing the oscillating behavior present in the ANN forecasts. The results also show that the dimension reduction with PCA does not improve RMSE, but it improves the similarity between forecasted and observed RDC. In fact, the RDCs reduction to three principal components prevents the oscillating behavior observed with K independent time series.

3.6. Comparison between 1D-LSTM and hybrid models

As showed in the previous section, the 1D-LSTM approach produced high quality day-ahead RDCs forecasts. However, the missing modelling of the dependency structure between energy and price bid led to a non-perfect replication of the typical RDCs "stair" shape. Indeed, in the real RDCs, the energy values with higher prices are always higher than the ones with lower prices, while in the forecasted RDC this phenomenon is fully captured due to the forecasts' oscillation behavior observed in Figs. 5–6.

The 2D-LSTM models does not include exogenous variables, which are shown in Section 3.4 to contain relevant information that improves forecasting accuracy. Thus, the RMSE and Frèchet distance values are 4.7% and 4.44%, which do not outperform other benchmark models. Nevertheless, an important feature is the capacity of accurately forecasting the typical RDCs "stair" shape.

The hybrid approach was proposed to overcome the limitations of 2D-LSTM and presented the best forecasting skill. Figs. 7 and 8 show the RMSE and Frèchet monthly improvement obtained with the 1D-LSTM and hybrid approaches over the best benchmark model. It is important to note that the RMSE improvement obtained by the hybrid model is higher than the Frèchet one. This occurs because the hybrid model reduces the distance between real and forecasted energy points (measured by the RMSE), yet it does not improve so much the curve shape forecast (measured by the Frèchet distance), which keeps having a slight oscillating behavior.

In terms of computational time, the 1D-LSTM model's fitting was 34,920 seconds in a server with 96 GB RAM, Intel Xeon CPU X5680 @ 3.33GHz and 6 cores. During the testing period online learning is used to update the model, which means an operational time of just a few seconds. Conversely, the computational time of the 2D-LSTM was 8 days and 23 hours in a computer with 12GB RAM and a NVIDIA GeForceGTX TITAN X GPU.

4. Conclusions

This work, as a proof-of-concept, demonstrates that deep learning techniques like LSTM can be applied to forecast day-ahead RDCs by exploring its capability to project the RDC data onto a 2-dimension space and interpret it as image objects. In order to include exogenous information (e.g., RES forecasts), which has a high impact in the RDC shape, a hybrid approach combining 1D and 2D LSTM networks was proposed. The computational time was decreased by adopting GPUs to run the training of 2D-LSTM and applying online learning, which makes the proposed method feasible in an industrial environment.

The proposed hybrid approach obtained excellent day-ahead forecasts with an RMSE below 1.5% and outperformed five benchmarks models with an improvement over 35% in both Frèchet distance and RMSE. These results were obtained by using publicly available data (i.e., aggregated market curves and not offers at unit level), which shows that highly accurate RDCs forecasts can be obtained from open-data sources. Hence, in order to replicate this method in other electricity markets only aggregated curves from day D are required.

Future work consists in: (a) offer interpretability to decision-makers, e.g. exogenous variables effect on the curve shape; (b) model the effect of complex offers (e.g., minimum income condition) in the clearing price and curves.

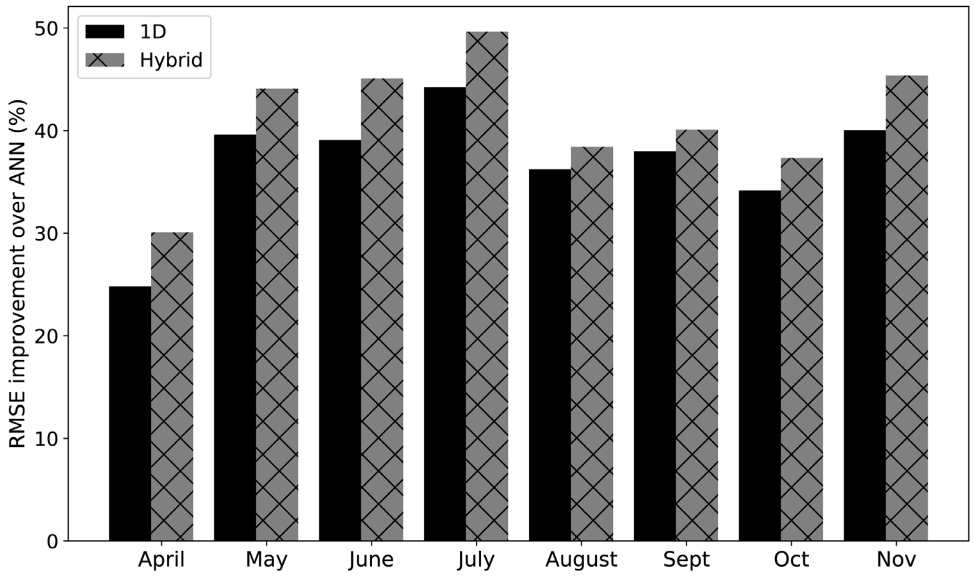

Fig. 7. RMSE monthly improvement of the 1D-LSTM and hybrid models over the ANN benchmark model.

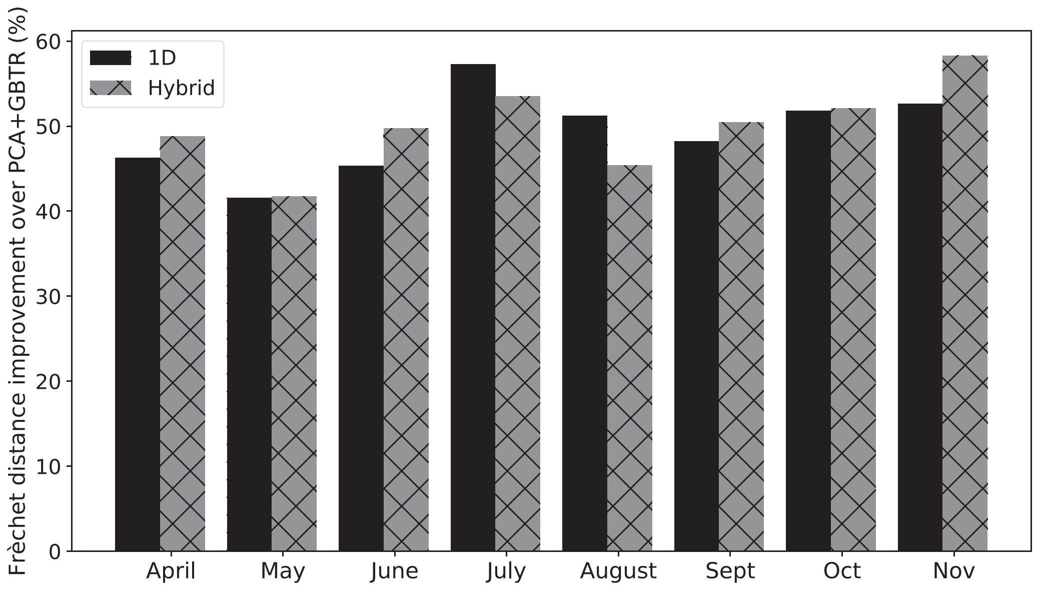

Fig. 8. Frèchet distance monthly improvement of the 1D-LSTM and hybrid models over the PCA + GBTR benchmark model.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors acknowledge the contribution of the NVIDIA Corporation, in providing the GeForce GTX TITAN X GPU used for this research.

Appendix A. Long Short Term Memory Networks (LSTM)

The description of the LSTM follows the formulation of LSTM as in [26].

Compared to RNN, a LSTM network is characterized by a different memory cell with self loops allowing it to store temporal information on the cells state (C_t, i.e. memory from the current block). The LSTM is composed by one input layers with number of neurons equal to the size of the input vector x_t, one or more hidden layers and one output layer with the number of neurons equal to the size of the expected output.

The C_t runs straight down the entire chain, with only some minor linear operations, leaving the information to flow along it unchanged until required. In fact, when new information pass through the block, the LSTM module can remove or add them to the cell state by activating specific structures, called *forget gate*, in successive steps.

The first step takes place in the *forget gate* that "decides" which information needs to be discarded from the cell state, Eq. 4. The gate reads the current input, x_t, and previous output from the LSTM layer, $h_{(t-1)}$, then, applies a sigmoid function σ that outputs a value between 0 (completely discard) and 1 (full hold).

$$f_t = \sigma(W_{f,x}x_t + W_{f,h}h_{t-1} + W_{f,c}\circ C_{t-1} + b_f) \tag{4}$$

where $\circ$ denotes the Hadamard product, f_t are activation values for the *forget gates, W_f* is a weights matrix and b_f is the bias term for the *forget gate*.

The second step is to determine which information will be added to the memory from the current block, i.e. cell state C_t. A sigmoid layer, called the *input gate*, is used to select which values will be updated:

$$i_t = \sigma(W_{i,x}x_t + W_{i,h}h_{t-1} + W_{i,c}\circ C_{t-1} + b_i) \tag{5}$$

where i_t are activation values of the *input gates*.

Meanwhile, an hyperbolic tangent function (tanh) function creates a vector of new candidate values $\tilde{C}_t$ (Eq. 12) to be added to the cell's state.

$$\tilde{C}_t = \tanh(W_{c,x}x_t + W_{c,h}\circ h_{t-1} + b_i). \tag{6}$$

Only by merging these two in the next step, the state update (from C_{t-1} to C_t) is actually performed as follows:

$$C_t = f_t\circ C_{t-1} + i_t\circ\tilde{C}_t. \tag{7}$$

Note that $_t\circ C_{t-1}$ erases the useless information and $i_t\circ\tilde{C}_t$ is added to store new useful information to the cell's state.

The final step consist in deriving the output value h_t that needs to be extrapolated from the last version of C_t. Firstly, an *output gate* composed of a sigmoid function is used to select which parts of the cell state will form the output:

$$o_t = \sigma(W_{o,x}x_t + W_{o,h}h_{t-1} + W_{o,c}\circ C_t + b_0). \tag{8}$$

Secondly, the cell state is also converted into a value between -1 and 1 by passing C_t through a tanh layer and then multiplied by the sigmoid output as follows:

$$h_t = o_t\circ\tanh(C_t). \tag{9}$$

Appendix B. Convolutional LSTM Network

The distinguishing feature of the Convolutional LSTM (ConvLSTM) is that all the inputs x, cell states C, hidden states h and gates (i, f, o) are 3D tensors with spatial dimensions [13] and the main difference to Eq. 4–8 is a convolution operator * in the state-to-state and input-to-state transitions as follows:

$$f_t = \sigma(W_{f,x} * x_t + W_{f,h} * h_{t-1} + W_{f,c} \circ C_{t-1} + b_f) \tag{10}$$

$$i_t = \sigma(W_{i,x} * x_t + W_{i,h} * h_{t-1} + W_{i,c} \circ C_{t-1} + b_i) \tag{11}$$

$$\tilde{C}_t = \tanh(W_{c,x} * x_t + W_{c,h} * h_{t-1} + b_i) \tag{12}$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t \tag{13}$$

$$o_t = \sigma(W_{o,x} * x_t + W_{o,h} * h_{t-1} + W_{o,c} \circ C_t + b_0) \tag{14}$$

$$h_t = o_t \circ \tanh(C_t) \tag{15}$$

References

[1] R. Weron, Electricity price forecasting: a review of the state-of-the-art with a look into the future, Int J Forecast 30 (4) (2014) 1030–1081.
[2] P. Klemperer, M. Meyer, Price competition vs. quantity competition: the role of uncertainty, Rand J Econ 17 (4) (1986) 618–638.
[3] M. Ventosa, A. Baíllo, A. Ramos, M. Rivier, Electricity market modeling trends, Energy Policy 33 (7) (2005) 897–913.
[4] F.A. Wolak, An empirical analysis of the impact of hedge contracts on bidding behavior in a competitive electricity market, Int Econ J 14 (2) (2000) 1–39.
[5] J. Contreras, O. Candiles, J.I. de la Fuente, T. Gómez, A cobweb bidding model for competitive electricity markets, IEEE Transcations on Power Systems 17 (1) (2002) 148–153.
[6] A. Ugedo, E. Lobato, A. Franco, L. Rouco, J. Fernandez-Caro, J. Chofre, Strategic bidding in sequential electricity markets, IEE Proceedings - Generation, Transmission and Distribution 153 (4) (2006) 431–442.
[7] R. Herranz, A.M.S. Roque, J. Villar, F.A. Campos, Optimal demand-side bidding strategies in electricity spot markets, IEEE Trans. Power Syst. 27 (3) (2012) 1204–1213.
[8] A. Philpott, E. Pettersen, Optimizing demand-side bids in day-ahead electricity markets, IEEE Trans. Power Syst. 21 (2) (2006) 488–498.
[9] G. Aneiros, J.M. Vilar, R. Cao, A.M.S. Roque, Functional prediction for the residual demand in electricity spot markets, IEEE Trans. Power Syst. 28 (4) (2013) 4201–4208.
[10] J. Portela, A.M.S. Roque, E.F. Sánchez-Úbeda, J. García-González, R. González, Residual demand curves for modeling the effect of complex offering conditions on day-ahead electricity markets, IEEE Trans. Power Syst. 32 (1) (2017) 50–61.
[11] J.P. González, Functional time series forecasting in electricity markets, Universidad Pontificia Comillas, Madrid, 2017 Ph.D. thesis.
[12] J. Portela, A. Muñoz, E. Alonso, Day-ahead residual demand curve forecasting in electricity markets, ISF 2012 - 32nd International Symposium on Forecasting,Boston, USA, (2012).
[13] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. kin Wong, W. chun Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, Advances in Neural Information Processing Systems 28 (NIPS 2015), Montreal, Canada, (2015).
[14] S. Ji, W. Xu, M. Yang, K. Yu, 3D Convolutional neural networks for human action recognition, IEEE Trans Pattern Anal Mach Intell 35 (1) (2013) 221–231.
[15] Keras: Deep Learning for humans, 2015, [Online] https://github.com/fchollet/keras (accessed on July 2020).
[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, et al., TensorFlow: Large-scale Machine Learning on Heterogeneous Systems, 2015.
[17] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput 9 (8) (1997) 1735–1780.
[18] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Networks 5 (2) (1994) 157–166.
[19] J. Andrade, J. Filipe, M. Reis, R. Bessa, Probabilistic price forecasting for day-ahead and intraday markets: beyond the statistical model, Sustainability 9 (11) (2017) 1990.
[20] A. Founta, D. Chatzakou, N. Kourtellis, J. Blackburn, A. Vakali, I. Leontiadis, A unified deep learning architecture for abuse detection, 10th ACM Conference on Web Science, Boston, USA, (2019), pp. 105–114.
[21] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Journal of Machine Learning Research - Proceedings Track 9 (2010) 249–256.
[22] T. Tieleman, G. Hinton, Lecture 6.5 - RMSprop: Divide the gradient by a running average of its recent magnitude, 2012, (Geoffrey Hinton Neural Networks for machine learning online course).
[23] V. Dumoulin, F. Visin, A guide to convolution arithmetic for deep learning, arXiv preprint arXiv:1603.07285, (2016).
[24] M.D. Zeiler, ADADELTA: An adaptive learning rate method, arXiv preprint arXiv:1212.5701, (2012).
[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al., Scikit-learn: machine learning in python, Journal of Machine Learning Research 12 (2011) 2825–2830.
[26] A. Graves, Generating sequences with recurrent neural networks, CoRR (2013) 1–43 arXiv preprint arXiv:1308.0850.