ПРОГНОЗИРАНЕ НА ЦЕНИТЕ НА ЕЛЕКТРОЕНЕРГИЯТА ДЕН НАПРЕД ЧРЕЗ РЕКУРЕНТНА НЕВРОННА МРЕЖА ЗА ИЗСЛЕДВАНЕ С ДЪЛГОСРОЧНА-КРАТКОСРОЧНА ПАМЕТ

DAY-AHEAD ELECTRICITY PRICE FORECASTING USING LONG-SHORT TERM MEMORY RECURRENT NEURAL NETWORK

Ekaterina Popovska, Galya Georgieva-Tsaneva
Institute of Robotics, Bulgarian Academy of Sciences
E-mail: ekaterina.popovska@gmail.com, galitsaneva@abv.bg

Abstract

The availability of accurate day-ahead electricity price forecasts is very important for electricity market participants and it is an essential challenge to accurately forecast the electricity price. Therefore, this study proposes an efficient method suitable for electricity price forecasting (EPF) and processing time-series data from the Bulgarian day-ahead market based on a long-short term memory (LSTM) recurrent neural network model. The LSTM model is used to forecast the day-ahead electricity price for the Bulgarian day-ahead market. As inputs to the model are used historical hourly prices for the period between 20.01.2016 and 05.03.2022. The output is the electricity price forecasts for hours and days ahead. The future values of prices are forecasted recursively. LSTM can model temporal dependencies in larger Time Series set horizons without forgetting the short-term patterns. LSTM networks are composed of units that are called LSTM memory cells and these cells contain some gates that process the inputs. Since electricity price is affected by various seasonal effects, the model is trained for several years. The effectiveness of the proposed method is verified using real market data.

Keywords: electricity price forecasting; deep learning; day-ahead market; time series forecasting; Long short-term memory (LSTM); machine learning.

1. INTRODUCTION

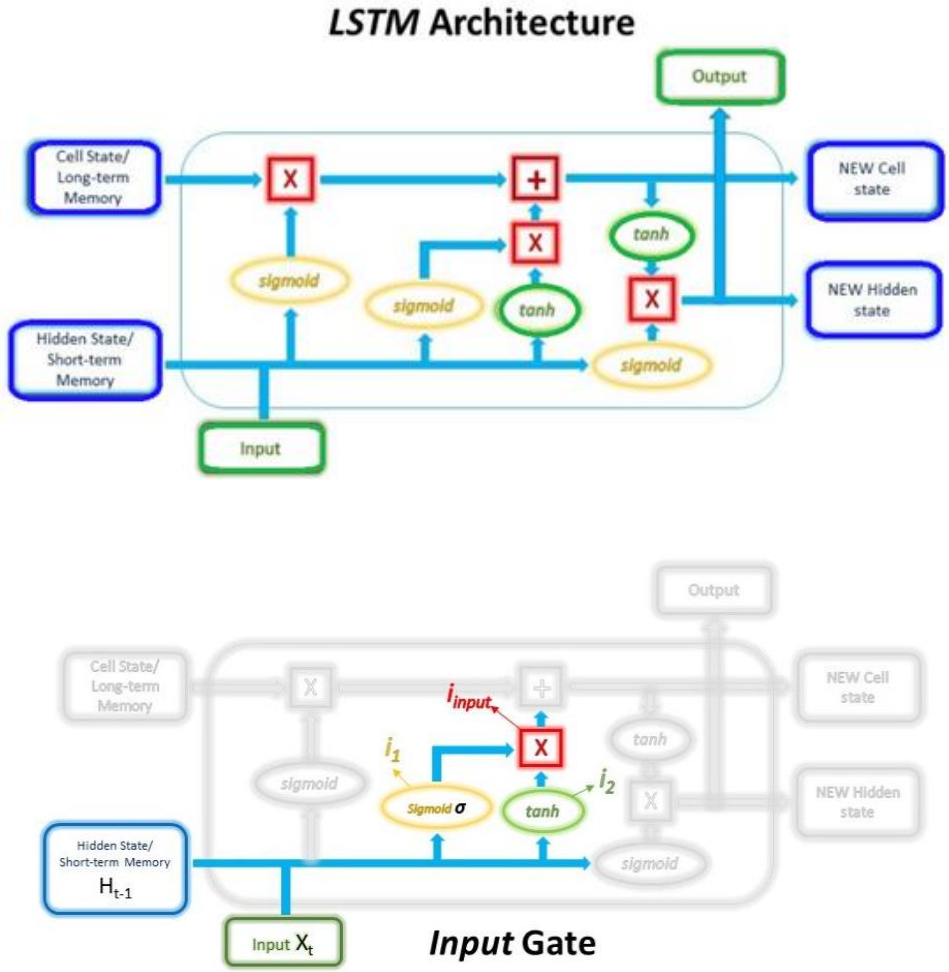
Due to the increased demand after the 2020 and COVID pandemic, during the first half of 2021, the world faced an energy crisis as oil and gas production was still carrying imbalances caused by the distortions in the world economy. At the same time, the levels of gas reserves in Europe did not secure the supply during the winter. The delay in the completion of the construction of the Nord Stream 2 contributed to the rise in prices. This behaviour continued until the end of the first quarter of 2022, when gas prices fell, dragging European electricity markets prices. The combination of many factors such as CO2 tariffs, renewables subsidies and the need for hedging tools for the energy transition, production of renewable sources that are dependable on natural factors, such as insolation, wind speed, and time of the year increased the volatility in the European electricity markets and made the electricity production more stochastic. Accordingly, accurate electricity price forecasting [1, 2] has become a substantial requirement since the liberalization of the electricity markets. Due to the challenging nature of electricity prices [3], which includes high volatility, sharp price spikes, and seasonality, various types of electricity price forecasting models [4, 5] still compete and cannot outperform each other consistently.

All this brought the need of using new methods for accurate day-ahead electricity price forecasting and one of them is artificial neural networks (ANN) which is increasingly explored during the last years. It may be considered that LSTM (long short-term memory) [6, 7] and RNN (recurrent neural network) [8, 9, 10] architectures are good methods for electricity price

forecasting. In this paper, we use the Bulgarian IBEX market day-ahead price dataset for price forecasting using LSTM architecture for capturing nonlinear short-term time dependencies [11, 12]. An accurate forecast may allow gaining a competitive advantage and it may also help traders, investors, and stakeholders to find out the most informative indicators to make a better market prediction, make investment decisions, and hedge the risks. The prediction of the market value is of great importance to help in maximizing the profit on electricity purchases while keeping the risk low.

2. PROPOSED METHODOLOGY

Long Short-Term Memory (LSTM) is an improvement over traditional Recurrent Neural Network (RNN) in the sense that it can effectively remember a long sequence of events in the past. RNN and LSTM tend to derive information from the previous context and chart future actions. The difference between RNN and LSTM [13] is that RNN is used in places where the retention of memory is short, whereas LSTM is capable of connecting events that happened way earlier and the events that followed them. LSTM [14] may be one of the best choices for analysing and predicting temporal-dependent phenomena that span over a long period or multiple instances in the past. In LSTM, each node is used as a memory cell that can store other information in contrast to simple neural networks, where each node is a single activation function. Figure 1 visualized the LSTM structure to define the guiding equations of LSTM. LSTM has three gates: input gate, forget gate, and output gate, as visualized in fig. 1.



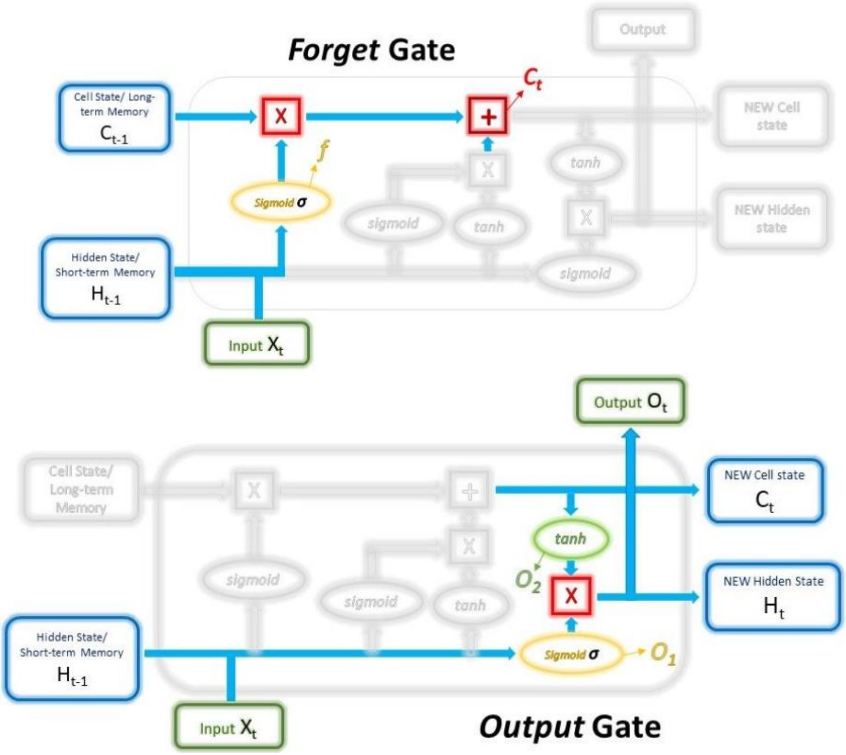


Fig. 1. *Illustration of LSTM architecture (figure adapted from [15], [16] and [17])*

The forget gate layer tells the LSTM cell whether to keep the past information or completely throw it away. The input gate determines what new information should be added to the cell state. The output gate finally gives the output which is a filtered version of the cell state. This gives the inherent ability for the model to retain information that is more relevant and forgot unnecessary information which is a filtered version of the cell state. That is why LSTM [18] requires a huge dataset during its training and validation process for a better result. The sigmoid layer decides the flow of information. Whether it needs to allow all of the information or some of it or nothing, multiple gates perform the role of Forget, Input, and Output. These gates perform the respective operation on the cell state which carries information from the past.
Regarding the input gate, the sigmoid function will transform the values to be between 0 and 1, with 0 indicating that part of the information is unimportant, whereas 1 indicates that the information will be used. This helps to decide the values to be kept and used, and also the values to be discarded. As the layer is being trained through back-propagation, the weights in the sigmoid function will be updated such that it learns to only let the useful pass through while discarding the less critical features. Mathematically, this is achieved using two layers:

$$i_1 = \sigma(W_{i_1} \cdot (H_{t-1}, x_t) + bias_{i_1}) \tag{1}$$

$$i_2 = tanh(W_{i_2} \cdot (H_{t-1}, x_t) + bias_{i_2}) \tag{2}$$

The outputs from these 2 layers are then multiplied, and the outcome represents the information to be kept in the long-term memory and used as the output shown in the formula [19]:

$$i_{input} = i_1 * i_2 \tag{3}$$

Just like the first layer in the Input gate, the forget vector is also a selective filter layer. To obtain the forget vector, the short-term memory, and current input is passed through a sigmoid function, similar to the first layer in the Input Gate above, but with different weights.

$$f = \sigma(W_{forget} \cdot (H_{t-1}, x_t) + bias_{forget}) \tag{4}$$

The outputs from the Input gate and the Forget gate will undergo a pointwise addition to give a new version of the long-term memory, which will be passed on to the next cell. This new long-term memory will also be used in the final gate, the output gate.

$$C_t = C_{t-1} * f + i_{input} \tag{5}$$

The output gate will take the current input, the previous short-term memory, and the newly computed long-term memory to produce the new short-term memory/hidden state which will be passed on to the cell in the next time step. First, the previous short-term memory and current input will be passed into a sigmoid function with different weights yet again to create the third and final filter. Then, we put the new long-term memory through an activation tanh function. The output from these 2 processes will be multiplied to produce the new short-term memory.

$$O_1 = \sigma(W_{output_1} \cdot (H_{t-1}, x_t) + bias_{output_1})$$

$$O_2 = tanh(W_{output_2} \cdot C_t + bias_{output_2})$$

$$H_t, O_t = O_1 * O_2 \tag{6}$$

The output of each time step can be obtained from the short-term memory, also known as the hidden state.

3. DATA ANALYSIS AND MODELING

In this paper, we will attempt to predict the Bulgarian Day-ahead electricity price using LSTM and analyze its accuracy by tweaking its hyper-parameters. In this stage, the historical data is collected from IBEX, https://ibex.bg/en/ and this historical data is used for the prediction of future prices. We have collected the historical data and have collected an hourly dataset and kept a window size of 41 690 hours. Data ranges from 20.02.2016 to 05.03.2022. Electricity prices follow patterns over daily, weekly and seasonal timeframes, and the schedule of the forecast series is on an hourly and daily basis. The target is to predict the price hours and days as the baseline model uses historical prices for future values prediction.

Trained models will be different from the other in the sense they will be trained on different hyperparameters. After data acquisition, the data will be pre-processed and structured and then the LSTM model will be created. Data cleaning is one of the many important stages in creating an efficient model. Data contains discrepancies and if not removed, they might cause hindrance in producing accurate results. Null values are the most common. Not all the data that is downloaded is necessary for training purposes. The extracted data is stored in a 2D array for further processing. Feature scaling is the most important part of data pre-processing as the results on hourly basis scaling are shown at fig. 2. It helps in standardizing/normalizing the

given dataset so that one entity is not given more importance than the other solely based on its quantity.

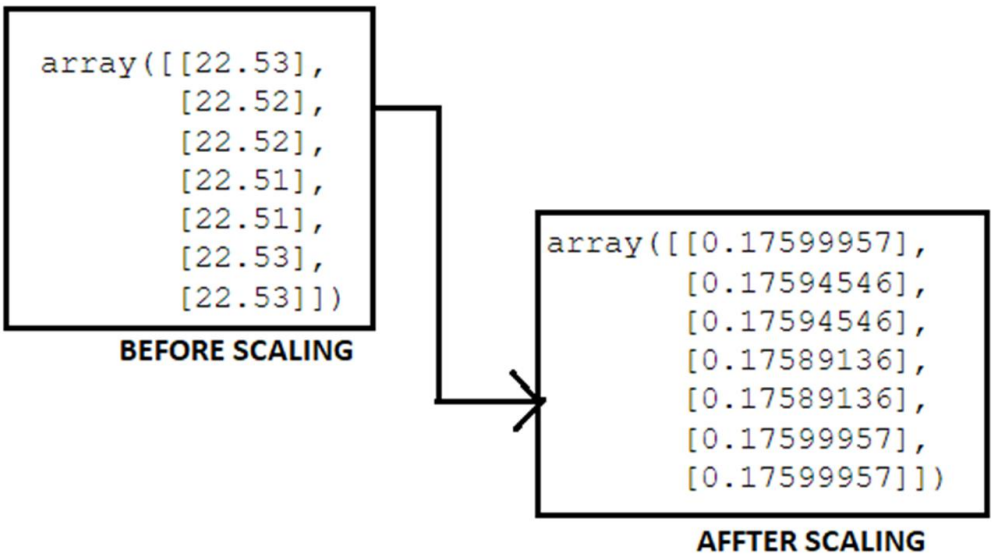


Fig. 2. *Hourly data scaling*

Bringing all the scattered data within the same level for easier analysis should be considered. We used MinMaxScaler() to scale all the values between 0 and 1. The model will be created and trained, and predictions will be made. The line charts of all the models will be plotted, and their accuracy will be observed. The structuring data and price forecasting processes are shown in fig. 3.

The LSTM takes a series as an input. The length of the series is equivalent to the number of previous hours. X_train is the input for the training set, and y_train is the output for a training set. A similar data structure is created for preparing a prediction dataset. Data sets will be structured for training purposes on an hourly and daily basis. The model will use the previous n datasets as the deciding factor and will predict the opening price for the n+1 value.

For LSTM model creation some modules will be imported as a sequential module for initializing the neural network, a dense module for adding a densely connected neural network layer, a Long Short-Term Memory layer, and a dropout module for adding dropout layers that prevent overfitting. Important hyperparameters of LSTM are below:

- Units=96: This means we are creating a layer with ten neurons in it. Each of these five neurons will be receiving the values of inputs.
- Input_shape = The input expected by LSTM is in 3D format. Our training data has a shape of (1689, 50, 1) for daily values and (41640, 50, 1) for hourly values this is in the form of (number of samples, time steps, number of features). This means we have 1689 examples to learn in training data, each example looks back 50-steps in time like what was the price yesterday, the day before yesterday so on till the last 50 days. These are known as Time steps. The last number '1' represents the number of features. Here we are using just one column 'Price' hence its equal to '1'.
- Kernel_initializer =When the Neurons start their computation, some algorithm has to decide the value for each weight.

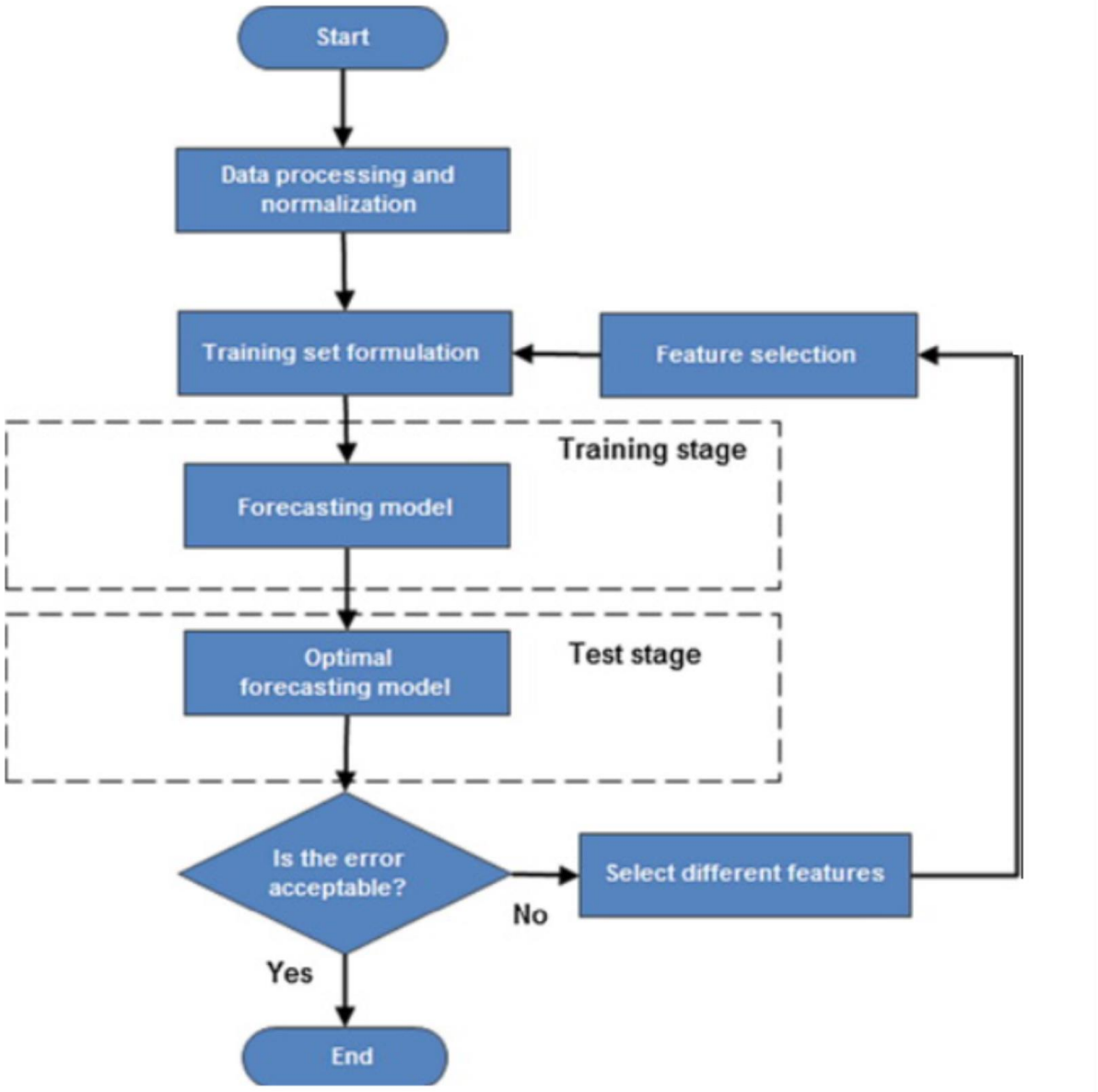


Fig. 3. *Price forecasting processes*

- Return_sequences =True: LSTMs backpropagate thru time, hence they return the values of the output from each time step to the next hidden layer. This keeps the expected input of the next hidden layer in the 3D format. This parameter is False for the last hidden layer because now it does not have to return a 3D output to the final Dense layer.
- Optimizer = This parameter helps to find the optimum values of each weight in the neural network.
- Batch_size =32: This specifies how many rows will be passed to the Network in one go after which the SSE calculation will begin and the neural network will start adjusting its weights based on the errors.
- When all the rows are passed in the batches of 50 rows each as specified in this parameter, then we call that 1-epoch. Or one full data cycle. This is also known as mini-batch gradient descent. A small value of batch_size will make the LSTM look at the data slowly, like 2 rows at a time or 4 rows at a time which could lead to overfitting, as

compared to a large value like 20 or 50 rows at a time, which will make the LSTM look at the data fast which could lead to underfitting. Hence a proper value must be chosen using hyperparameter tuning.

- Epochs=50: The same activity of adjusting weights continues 50 times, as specified by this parameter. In simple terms, the LSTM looks at the full training data 50 times and adjusts its weights.

4. FORECASTING RESULTS

After the model has been trained, the hourly and the daily forecast shall be made. For this purpose, a test dataset is created which contains past hours and days prices so the measuring of the accuracy of the model on hourly and daily basis data errors may be checked. The dataset is structured the same way as done in the data structuring step. These values are fed one by one to the LSTM input layer and the output is collected and stored in a 2D array for chart plotting and accuracy analysis.

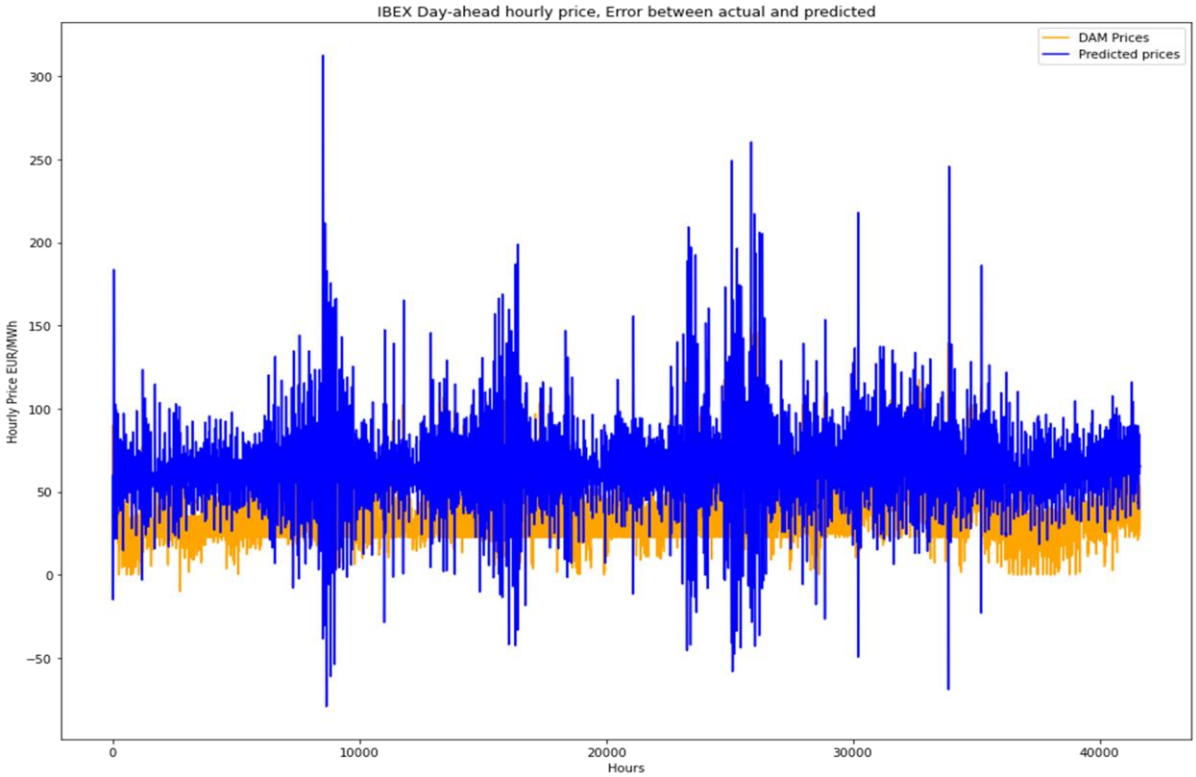


***Fig. 4**. Measuring the Accuracy of the model – hourly data error check*

Figures 4 and 6 show that the model can generally predict the correct direction rather accurately due to the constant seasonality. It demonstrates that training on the difference rather than on the magnitude results in higher accuracy and a better ability to generate a potentially leading model.

From figure 5 and figure 7, we can see the experimental prediction of the model on hourly and daily data respectively predicted with LSTM. Better values can be seen from the hourly data model. On the other hand, LSTM is not able to predict correct prices and give good results with modelled data on daily basis. At figure 7, there are some spikes of rising and falling prices

in the actual price of real prices that are not matched by the model with predicted data curve. It seems like due to inaccurate data. But few incorrect data in the training set or test set cannot affect the prediction model largely or only has the effect that can be ignored.

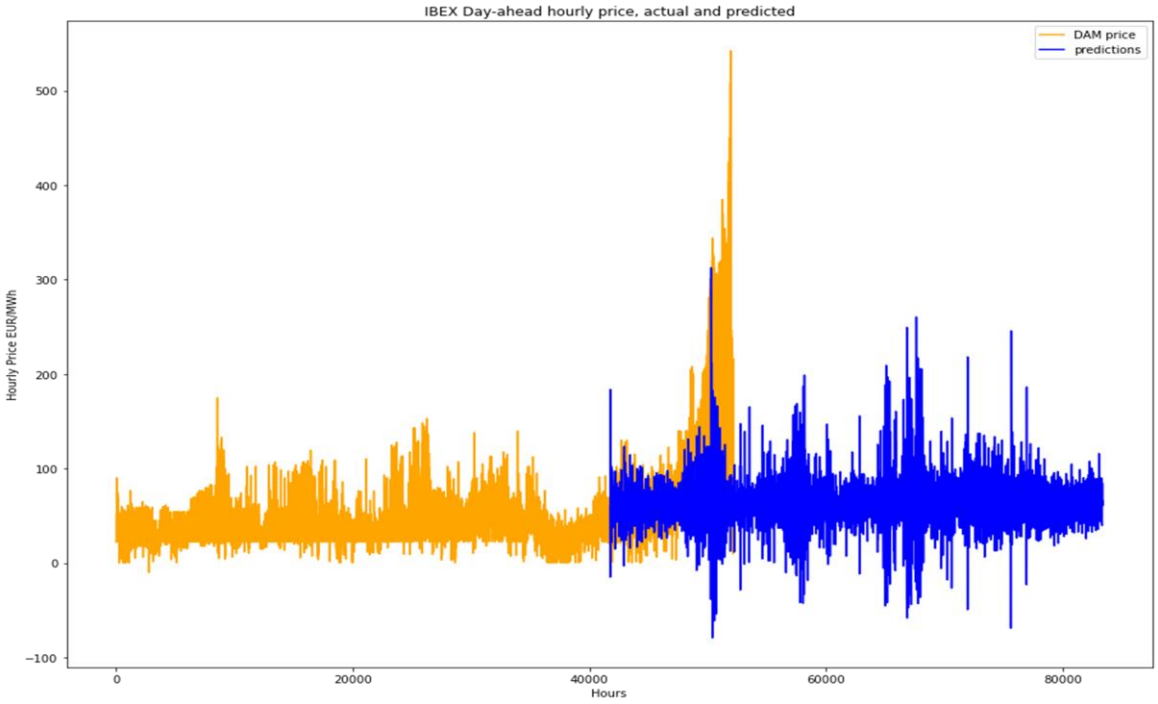


Fig. 5. *Experimental prediction of the model on hourly data*

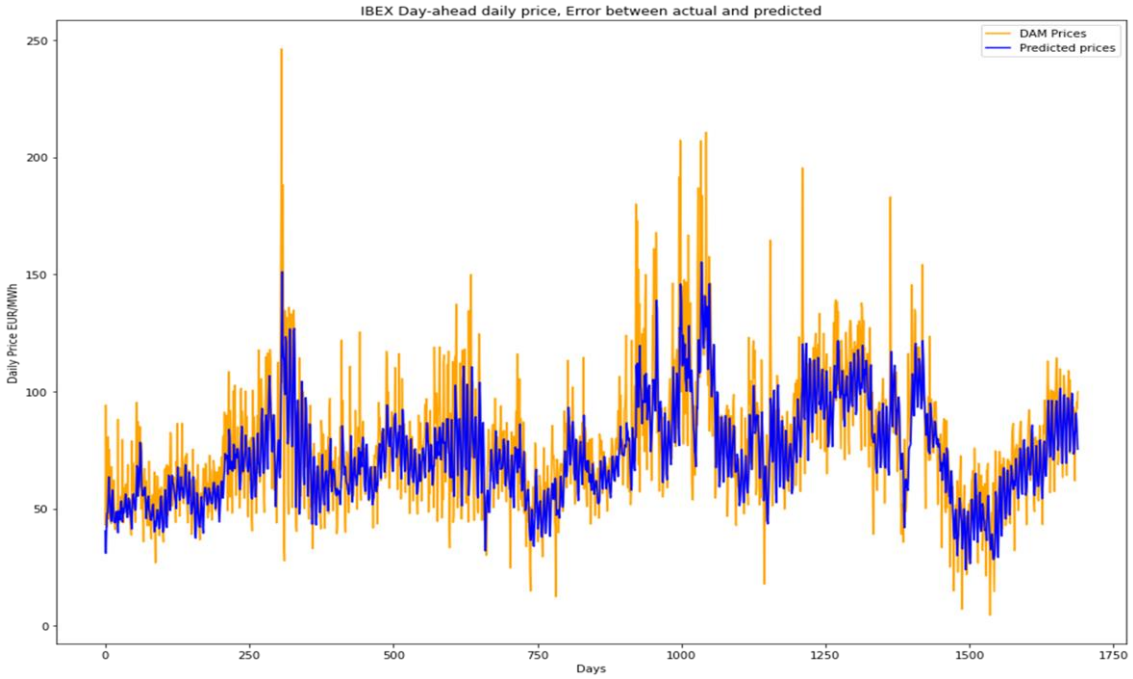


Fig. 6*. Measuring the accuracy of the model on a daily data basis error check*

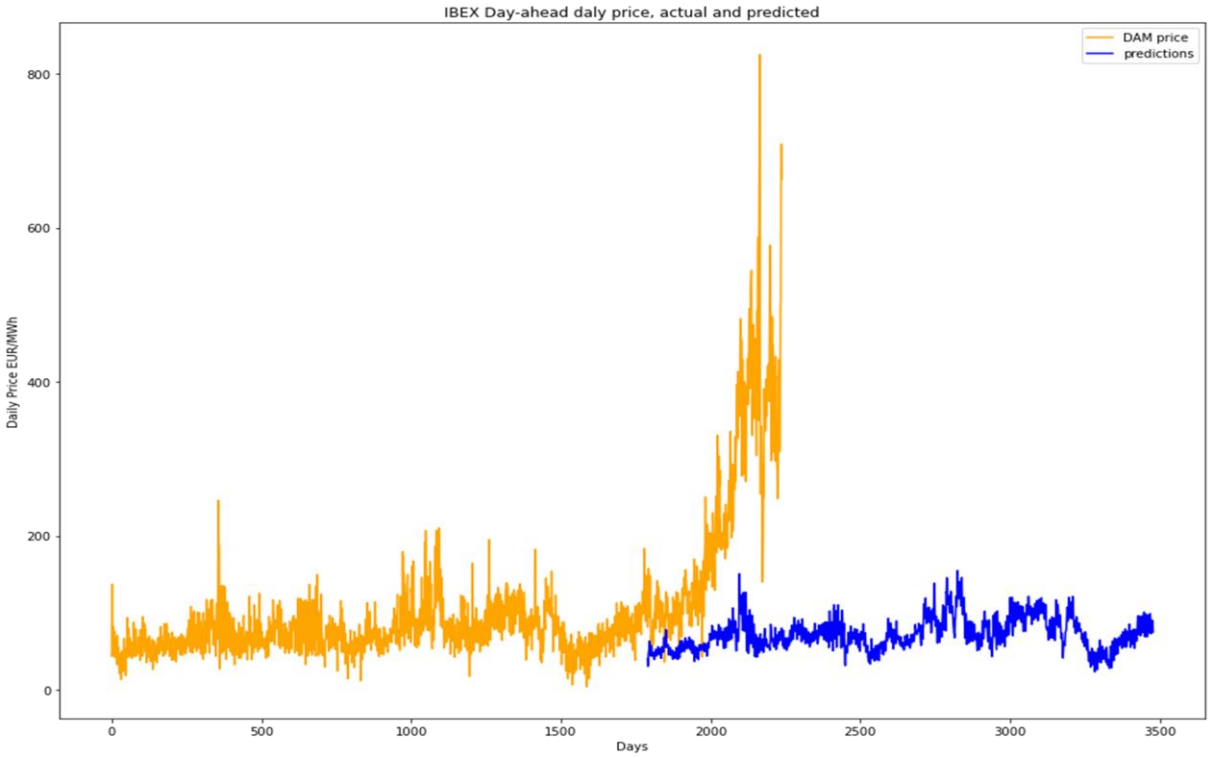


Fig. 7. *Experimental prediction of the model on daily data*

After performing various simulations with a different number of parameters and epochs, we have observed we achieve the best results with training.

CONCLUSIONS

In this paper, we present the LSTM-based architectures for the EPF. Results show that forecasting using the LSTM method may be accurate only for short-term usage. On an hourly basis, the results gave awesome results. However, forecasting for multiple days like the next 30-days or 50 days, the method fails. Failure is not because the LSTM model is bad, but, because electricity markets are highly volatile.

The results show that the benefit of EPF will have spot electricity traders and policymakers, who make decisions based on accurate price predictions. The ability to make predictions based upon historical observations creates a competitive advantage. More testing on other feature selection models can provide more possibilities for researchers and industries to understand how different features affect prediction accuracy. In addition, LSTM may be a useful tool in other areas for example predicting the demand or sales of a product, the count of passengers in an airline, or the closing price of a particular stock, predicting future electrocardiographic **(**ECG), photoplethysmographic (PPG) signals [20, 21, 22] while reducing the non-stationary character of those signals and improving the prediction accuracy.

BIBLIOGRAPHY

[1] C.J. Huang and P.H. Kuo. (2018). "A deep CNN-LSTM model for particulate matter (PM2. 5) forecasting in smart cities". *Sensors*, 18(7), p.2220.

[2] X. Xie, M. Li and D. Zhang. 2021. "A Multiscale Electricity Price Forecasting Model Based on Tensor Fusion and Deep Learning". *Energies*, 14, 7333. https://doi.org/10.3390/en14217333

[3] G. Memarzadeh, F. Keynia. (2021). "Short-term electricity load and price forecasting by a new optimal LSTM-NN based prediction algorithm". *Electr. Power Syst. Res*. 192, 106995.
[4] C.J. Huang, Y. Shen, Y.H. Chen, H.C. Chen. (2021). "A novel hybrid deep neural network model for short-term electricity price forecasting". *Int. J. Energy Res*. 45, 2511–2532.
[5] X. Guo, Q. Zhao, D. Zheng, Y. Ning, Y. Gao. (2020). "A short-term load forecasting model of multi-scale CNN-LSTM hybrid neural network considering the real-time electricity price". *Energy Rep.*, 6, 1046–1053.
[6] M. Sundermeyer, R. Schlüter, and H. Ney. (2012). "LSTM Neural Networks for Language Modeling". In *Interspeech*, pp. 194–197.
[7] Y. Wang. (2017). "A new concept using LSTM neural networks for dynamic system identification". In *ACC*, pp. 5324–5329. IEEE.
[8] A. Sherstinsky. (2020). "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D: Nonlinear Phenomena*, vol. 404, *Special Issue on Machine Learning and Dynamical Systems.*
[9] I. Sutskever, J. Martens, and G. E. Hinton. (2011). "Generating text with recurrent neural networks". In *Lise Getoor and Tobias Scheffer*, *editors*, *ICML*, pp. 1017–1024. Omnipress.
[10] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. Hasegawa-Johnson, T. S. Huang. (2017). "Dilated recurrent neural networks". *Advances in Neural Information Processing Systems*, pp. 77-87.
[11] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, M. Weyrich. (2021). "A survey on long short-term memory networks for time series prediction". *Procedia CIRP*, vol. 99, pp. 650-655. https://www.sciencedirect.com/science/article/pii/S2212827121003796
[12] S. Hochreiter, J. Schmidhuber. (1997). "Long Short-Term Memory". *Neural Comput.* 9 (8), pp. 1735–1780; doi: https://doi.org/10.1162/neco.1997.9.8.1735
[13] X. Tang, Y. Dai, T. Wang, Y. Chen. (2019). "Short-term power load forecasting based on multi-layer bidirectional recurrent neural network", *ET Gen Transm Distrib*, vol. 13(17), pp.3847–3854, https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-gtd.2018.6687
[14] G. R. Waheeb. (2020). "A novel error-output recurrent neural network model for time series forecasting", *Neural Comput Appl*, vol. 32, p. 9621–9647.
[15] J. Chung, C. Gulcehre, K. Cho and Y. Bengio. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv*, p. arXiv:1412.3555.
[16] "Floydhub," LSTMs architecture, [Online]. Available: http://harinisuresh.com/2016/10/09/lstms/. [Accessed 3 march 2022].
[17] G. Loye, "https://blog.floydhub.com/long-short-term-memory-from-zero-to-hero-with-pytorch/," FloydHub. [Online]. [Accessed 3 March 2022].
[18] K.F. Chu, A.Y. Lam, and V.O. Li. (2019). "Deep multi-scale convolutional LSTM network for travel demand and origin-destination predictions". *IEEE Transactions on Intelligent Transportation Systems*, 21(8), pp.3219-3232
[19] G. Loye. "Deep Learning. Long Short-Term Memory: From Zero to Hero with PyTorch". https://blog.floydhub.com/long-short-term-memory-from-zero-to-hero-with-pytorch/
[20] E. Gospodinova, P. Lebamovski, M. Gospodinov. (2021). "Automatic analysis of ECG signals based on their fractal and multifractal properties". *International Conference on Computer Systems and Technologies '21*, *Association for Computing Machinery, New York, NY, United States*, ISBN:978-1-4503-8982-2, DOI:10.1145/3472410.3472421, 136-140.
[21] M. Gospodinov, E. Gospodinova, P. Lebamovski. (2021). "Analysis of Heart Rate Variability using photopletismographic and electrocardiographic signals", *Scientific conference with international participation STEMEDU-2021*, 3, IMI-BAS, ISBN:2683-1333, pp. 7-12.
[22] K. Cheshmedzhiev. (2021). "Registering and Processing of a Photoplethysmography Signals". In *STEMEDU 2021*, pp. 13-19. http://www.math.bas.bg/vt/stemedu/book-3/02-STEMedu-2021.pdf