KU LEUVEN

FACULTY OF ECONOMICS AND BUSINESS

Deep Learning and Electricity Price Forecasting on the Belgian day-ahead market

Using recursive multi-step and probabilistic forecasts to improve accuracy and quantify uncertainty.

Abel KEMPYNCK

r0776327

Vic VANDEPUT

r0809678

Thesis submitted to obtain the degree of

MASTER OF BUSINESS ENGINEERING

Data Science and Business Analytics

Promoter: Prof. Dr. Christophe Croux

Co-Promoter: Dr. Ir. Hussain Kazmi

Work Leader: Joris Depoortere

Academic year: 2023-2024

Abstract

Accurate electricity price forecasting is critical in today's volatile energy markets, particularly with the increasing penetration of renewable energy sources. This thesis aims to compare the increasingly popular deep learning techniques with established statistical models for electricity price forecasting in the Belgian day-ahead market. The comparative study uses a deep learning method called Long Short-Term Memory (LSTM) and a statistical LASSO-Estimated Auto-Regressive (LEAR) model both in predicting price values as well as quantifying their uncertainty. The main findings reveal that the LEAR model outperforms the LSTM model in terms of accuracy and computation time. The study emphasizes the importance of recalibration for improving forecasts and challenges the notion that deep learning models always outperform statistical methods. The research contributes to the literature by providing insights into the effectiveness of different forecasting models in the field of electricity price prediction.

Acknowledgements

We would like to thank our promoters, Hussain Kazmi and Christophe Croux, for guiding us through the process of making this thesis, sitting down with us regularly to provide us with valuable feedback and setting the next goals to attain. Likewise, we wish to express our gratitude towards the assistants Joris Depoortere and Margarida Mascarenhas because of their vital support in the writing and coding of the thesis. Lastly, since this thesis is the work of two, we are also thankful to each other for the smooth collaboration and atmosphere in the realization of the thesis.

Contents

Chapter 1

Introduction

Accurate price forecasting is crucial for various stakeholders in the dynamic environment of energy markets. Accurate forecasts empower all participants, including producers, consumers and traders, enabling them to make informed and strategic decisions [16]. In a context in which electricity prices have become increasingly volatile in addition to an increasing share of renewable energy sources which are volatile by design, forecasting thereof poses a real challenge. Unlike other commodities, electricity can't be easily stored, supply and demand have to be in constant balance, making it a unique challenge for the market. Historically, statistical methods have shown great strength in forecasting electricity prices and time series in general. However, more recently, deep learning based techniques have gained popularity given the sharp increase in both the dimensionality and quantity of data, lending itself to these techniques capable of capturing deep latent patterns and features.

1.1 Background

Electricity price forecasting first gained attraction with the deregulation of monopolistic power sectors and the creation of competitive electricity markets when it became essential to provide accurate information about the future load, generation and price levels, around 30 years ago. In recent years electricity prices have been behaving more unpredictable than ever before. This is due to the energy transition currently present both in consumption and generation. The share of renewables in the generation of electricity has increased over the last few years. Because wind speed and solar energy are inherently volatile, the generation of electricity is less predictable than before. This makes the forecasts of energy data more valuable than they used to be.

Apart from the recent developments in the energy sector another trend has come up in the field of forecasting. Because of the increased efficiency in computation, deep learning, a kind of machine learning technique, has gained attraction in providing models for predicting time series. This thesis aims to examine the performance of deep learning techniques within electricity price forecasting.

Previous work includes the master's thesis of Jilles De Blauwe [9]. The work of De Blauwe examined, amongst others, the performance of an advanced statistical LASSO-estimated regression model called LEAR. While showing promising results, the model wasn't compared against deep learning methods. Part of this thesis was inspired to try and compare the results of a LEAR model against a state-of-the-art deep learning model.

1.2 Objectives

This master's thesis aims to address two key research questions within the field of smart grids and electricity price forecasting (EPF) on the Belgian day-ahead market. The primary objectives are as follows:

1. **Comparative Analysis between Statistical Models and DNN-based Architectures**
   One of the principal objectives of this study is to conduct a comprehensive comparison between state-of-the-art statistical models and Deep Neural Networks (DNN)-based architectures for electricity price prediction in the Belgian day-ahead market. By leveraging advanced deep learning methodologies, more specifically Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, this research endeavors to evaluate the efficacy of DNN-based architectures in comparison to traditional statistical models. Through careful experimentation and comparative analysis, the aim is to provide insights into the effectiveness and efficiency of each method in capturing the complex temporal dynamics and inherent volatility of electricity prices.

2. **Improvement of the uncertainty issue through usage of distribution forecasts on BELPEX Market**
   Another crucial objective of this research is to mitigate the uncertainty inherent in electricity price forecasting by leveraging distribution forecasting techniques on the BELPEX market, i.e. the Belgian day-ahead electricity market. In recent years, the importance of quantifying uncertainty in energy market predictions has become increasingly evident, especially in light of the growing penetration of renewable energy sources and the resulting market volatility. By employing distributional forecasts, such as Distributional Deep Neural Networks and conformal prediction, this study aims to provide probabilistic forecasts that not only capture the central tendency of price movements but also quantify the associated uncertainty. Through the implementation and evaluation of quantile forecasting models, the objective is to enhance the decision-making capabilities of market participants and stakeholders by providing more informative and actionable forecasts.

1.3 Outline and Scope

This work is divided into six chapters. Chapter 2 provides a background in electricity price forecasting and some of its models used in previous literature, as well as evaluation methods necessary to perform EPF research. Chapter 3 discusses the dataset that was used for the case study. The methods used for constructing the forecasting models as well as the evaluation metrics used in this thesis are described in Chapter

4. In Chapter 5 the results of the case study on the Belgian day-ahead market are presented. Lastly, Chapter 6 contains the conclusions and future work.

Chapter 2

Background

This chapter gives an overview of the theoretical background required to conduct the EPF (Electricity Price Forecasting) research of this thesis. An overview of the day-ahead electricity market itself in Belgium and Europe is given as well as a literature overview and theoretical elaborations on the various models and evaluation methods in the EPF field. The first section discusses the European day-ahead market for electricity while the second section explains different forecasting models and reviews the existing literature concerning these models. The last section describes various options on how to evaluate and compare these models.

2.1 EPEX SPOT market

The Belgian electricity market is part of the European Power Exchange (EPEX) [41]. Founded in 2008, following the deregulation of electricity markets in Northern and Western Europe, it is a merger of multiple national power exchanges. EPEX SPOT operates in Austria, Belgium, Denmark, Germany, Finland, France, Luxembourg, the Netherlands, Norway, Poland, Sweden, the UK and Switzerland. To gain insights on how to construct practical price forecasting models it is useful to first understand the workings of the EPEX SPOT market in Europe. This section briefly explains who the main traders are on this spot market, the timeline of the bidding process, as well as how the day-ahead electricity market price comes about.

2.1.1 Traders on the EPEX SPOT market

Trading on the spot market in Europe is done by major electricity generators and suppliers, or third parties that trade on behalf of companies. Members of the exchange market include [43]:

- Utilities (e.g. Luminus or ENGIE). These companies buy and sell electricity to supply to their customers or compensate for imbalances in their power plants.

- Banks and financial service providers. They play a role in adding liquidity to the market. They don't have generation units or are major suppliers but do trade on the market.

- Trading companies. They have a similar role as the banks by adding liquidity in the market without having their own power assets and are specialized in managing electricity portfolios.
- Other traders include transmission and distribution system operators, regional suppliers and aggregators.

There are more than 800 companies active on the exchange, most of which are utilities, regional suppliers and trading companies. Figure 2.1, from [43], gives an overview of what kind of companies trade on the market.

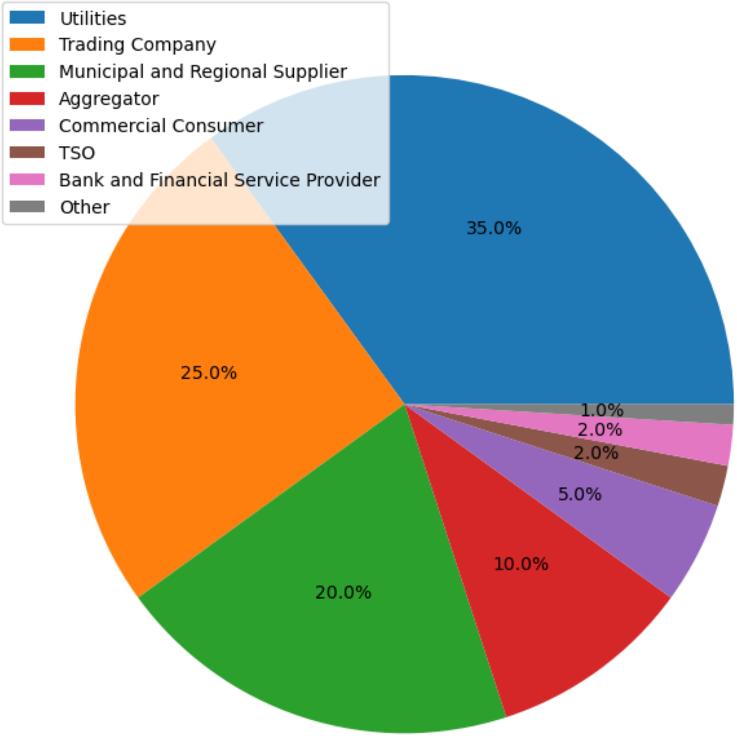


Figure 2.1: Exchange members of EPEX SPOT, by category, from [43]

2.1.2 Timeline of the Day Ahead electricity market

When a company wants to trade on the spot market it has to submit a bid for buying or selling power for the next day, at a specific time (hour) and a specific area. All bids have to be submitted at 12:00 CET, except for Switzerland and the UK, whose markets close even earlier at 11:00 and 10:20 respectively. Each order contains their willingness to buy or sell, in volume, for each price. The aggregated buy orders generate a demand curve while the sell-orders form a supply curve for each hour of the day [42]. In this way, a price is cleared for each hour at the intersection of the demand and supply curves. The price obtained is called the Market Clearing Price and is the same for all market members in the same country/region. Hourly prices, supply and demand curves, and other market data can all be found on the website of EPEX SPOT [44].

A company will never pay more for a certain volume than its willingness to buy, and a seller will never sell below its own pre-specified minimum price [42]. Further imbalances in the market due to generation unit failures or customers dropping out are settled on the intra-day market, which falls outside of the scope of this thesis.

After the market is cleared, the prices for the next 24 hours are published soon after the market has closed. Prices are published at 13:00 CET for most European markets including Belgium, 11:10 CET for the Swiss market, and 10:30 CET for the UK. A company trading on the Belgian spot market will therefore have to have its bids ready at around 11:00 to avoid missing the submission of the bids due to communication delays. An interesting note is that the Swiss prices are already published 50 minutes before the rest of the markets close. Because of the high correlation between the Swiss market and the Belgian market, [9] looked into the possibility of integrating Swiss day-ahead prices into a model to predict the Belgian prices. The study gave promising results but the feasibility of incorporating Swiss prices into the model due to the time constraints of only 50 minutes is not guaranteed.

2.2 Current state-of-the-art models in EPF

In order to evaluate the performance of a forecasting model in a meaningful way the model has to be compared against a benchmark of state-of-the-art models, including both statistical models as well as deep learning models [31]. This section gives a summarized overview of some of the current state-of-the-art models found in the recent literature on the EPF field.

2.2.1 Statistical methods

Similar Day models

Similar Day Models are a very simple technique for predicting prices on the day-ahead market. It takes the prices of another day similar to the one that is being predicted and uses it as a forecast. This can be the same day in the previous week when $\hat{p}_{d,h} = p_{d-7,h}$. Or the model can just use the prices from the day before as the prices for the predicted day, in that case, $\hat{p}_{d,h} = p_{d-1,h}$. The naive similar day forecast used in this thesis, as well as in [9], is a combination of using the previous day and the previous week. Equation (2.1) shows the formula used to predict the prices. On a Saturday, Sunday, or Monday, the prices of the previous week are taken. While on a Tuesday to Friday, yesterday's prices are used.

$$\hat{p}_{d,h}^{\text{naive}} = \begin{cases} p_{d-1,h}, & \text{if } d \text{ is Tue, Wed, Thu, or Fri,} \\ p_{d-7,h}, & \text{if } d \text{ is Sat, Sun, or Mon.} \end{cases} \tag{2.1}$$

These naive models can be used in the calculation of evaluation metrics such as rMAE, discussed in Section 2.4.1.

Auto-Regressive models

Auto-Regressive (AR) models are widely utilized for time series forecasting. In AR modeling, it is assumed that the next price is determined by its previous values in the time series. These historical data points, often referred to as ”lags,” are used to predict future data points. An AR model is trained by applying a linear regression on these lagged variables. Equation (2.2) shows what an AR model would be like. The number of lags included is called the order of the AR model.

$$y_t = \hat{y}_t + \epsilon_t = \alpha_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \beta_3 y_{t-3} + ... + \epsilon_t \tag{2.2}$$

ARX

An extension of the simple AR model is the ARX (Auto-Regressive eXogenous) model. This uses not only regression on historical prices to estimate the model but also includes exogenous variables and their lags as explanatory variables. The paper [29], by J. Lago et al. is a large benchmark study comparing all types of models, including multiple ARX models, although these seemed to perform worse than DL models or more advanced AR models. Possible exogenous variables can be weather forecasts (wind speed, temperature), generation forecasts (overall generation, solar, wind), load forecasts, etc.

farX and LEAR

To gain more accuracy in EPF, a new extension of the ARX was proposed [46], called the fARX (full ARX) model. This is an AR type model that only includes the price lags $d-1$, $d-2$, $d-3$ and $d-7$, as well as the same lags for the exogenous variables. In the already mentioned benchmark study [29] by J. Lago, fARX performs significantly better.

Another extension to this fARX model, also developed in [46], is the LEAR model, also called the fARX-LASSO or LASSOX. This model uses the regularization technique of LASSO (Least Absolute Shrinkage and Selection Operator), to automatically select the input features. In this model, the LASSO operator applies Occam’s razor principle, assigning weights to features and automatically reducing the contribution of some while increasing others. This parsimonious approach enhances the model’s simplicity without requiring the manual selection of input features. Out of all AR-based models in the benchmark study [29], LEAR, alongside a similar model fARX-EN (Elastic-Net), achieved the highest accuracy, producing results comparable to those of DL models. For this reason, the LEAR model will be used as a benchmark in this study. It is used as as a state-of-the-art statistical method to compare against the DL model, as should be done in good EPF research [31].

2.2.2 Deep learning methods

Deep Neural Networks (DNN)

A simple yet effective approach to forecasting electricity prices using deep learning is DNNs. A DNN consists of three components, an input layer, an output layer and multiple hidden layers. Each layer itself consists of a certain number of nodes, called neurons, that are all connected to the previous layer and the next layer. The input layer has as many neurons as there are inputs, in this case, historic prices and all other covariates including their lags that would be included in the model, the output neurons represent the forecasted prices. While the hidden layers are used to capture complex nonlinear relationships in the data [7]. Figure 2.2, from [30] shows how such a DNN would graphically look like.

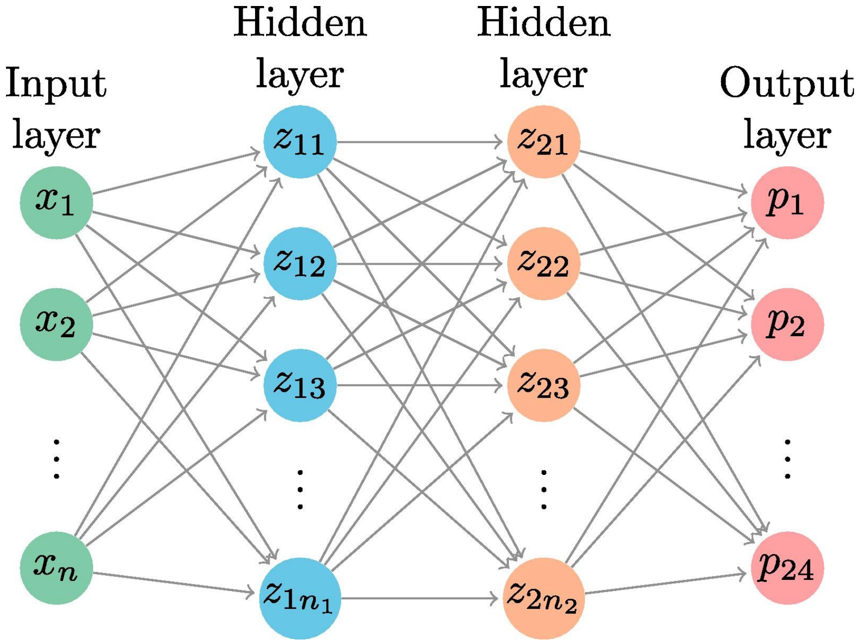


Figure 2.2: Example of Deep Neural Network with 2 hidden layers, from [30]

Each neuron in the hidden layer takes on a value that depends on the previous layer until the output layer determines the final output of the model.
The value of the i_{th} neuron in layer l, $z_i^{(l)}$, is determined by three kinds of parameters.

- The weight vector W_i, which consists of individual weights w_{ij} for each connection between all neurons of the previous layer and the neuron $z_i^{(l)}$. These weights are multiplied by the values of the neurons of the previous layer $Z^{(l-1)}$.
- A bias b_i, which is added to the product of the weights and the values of layer $l-1$.
- An activation function f, which transforms each input to a value between 0 and 1. Examples are the sigmoid function, softmax, or ReLU.

Equation (2.3) is a mathematical representation of the method to calculate one specific neuron in one specific layer. This is done for each neuron in each layer, with each connection having a separate weight and each neuron a different bias, to obtain an output vector Y_d, consisting of the predicted prices for each hour on day d. If there are N hidden layers, each comprising n neurons, n_x number of inputs and n_y outputs, then the total number of calculations required amounts to $N \times n + n_y$.

$$z_i^{(l)} = f\big(W_i^T Z^{(l-1)} + b_i\big) = f\Bigg(\sum_{j=1}^{n} w_{ij} z_j^{(l-1)} + b_i\Bigg) \tag{2.3}$$

To obtain the optimal weights and biases for the network, random values are taken as weights and biases in the first step. Through the optimization techniques of backpropagation and gradient descent, the parameters are iteratively tuned in a manner that minimizes the loss function of the output, often RMSE or MAE.

DNNs used to be hard to implement in EPF due to the high computational complexity. Thanks to the advances in computational technology it has become viable to train and test these complex networks with hundreds of neurons and multiple hidden layers, inputs and outputs [21]. This caused DNNs to attract a lot of attention in the EPF field in recent years. Paper [29] demonstrates a simple 2 hidden layer DNN outperforms state-of-the-art statistical models. [30], [8] and [33] all proposed models with DNNs that performed well, although only comparing them to other DL techniques.

RNN (LSTM)

Recurrent neural networks are different from the neural network because they incorporate feedback loops that make it possible for previous outputs to be used as input for the next timestep. This makes RNNs especially useful for time series forecasting since time series are often correlated with their previous values. However, standard RNNs suffer from the vanishing gradient problem. This is where gradients, which are vectors used in the training part of a NN that point in the direction of the greatest decrease in the loss function, can diminish over long sequences. This limits their effectiveness in capturing long-range dependencies. This limitation led to the development of more advanced architectures like LSTMs and gated recurrent units (GRUs) that address this issue by incorporating explicit mechanisms to control information flow and preserve memory over time.

The most commonly used RNN, and also the one used in this thesis is the LSTM, [18]. Unlike a DNN, which is a feedforward network that transforms the inputs into outputs through several transformations going from one layer to the next, an LSTM network consists of a cell for each timestep where it is able to temporarily store and forget information to capture these time dependencies in the data.

A graphical representation, taken from [29], of such an LSTM cell is shown in Figure 2.3. An LSTM cell at time t consists of three gates. A forget gate F, an input gate I, and an output gate O. The fundamental concept behind LSTM lies in its ability to maintain and selectively update a memory cell state thanks to these gates, which is crucial for capturing long-term dependencies in sequential data.

At each time step t, the LSTM cell uses the previous hidden state z_{t-1} and the current input x_t as decision variables through the input gate (I) and forget gate (F). The forget gate F_t, determined by the sigmoid function σ, selectively chooses which information to retain from the previous cell state c_{t-1} based on z_{t-1} and x_t. Similarly, the input gate I_t and the tanh (hyperbolic tangent function) unit decide which new information is relevant to update the cell state.

While the forget gate and input gate, F_t and I_t, create decision vectors to determine which old information to forget and which new information to store, the tanh unit's role is to create the vector $\bar{c}_t$ containing this new information to be added to the cell state.

Equation (2.4), (2.5) and (2.6) present the way how the input and hidden state are processed through the forget gate and input gate, with W_F, W_I and W_c representing the weights associated with these gates and b_F, b_I and b_c respectively the biases, similar to how DNN processes input throughout its layers.

$$F_t = \sigma\big(W_F \begin{bmatrix} z_{t-1} \\ x_{t-1} \end{bmatrix} + b_F\big) \tag{2.4}$$

$$I_t = \sigma\big(W_I \begin{bmatrix} z_{t-1} \\ x_{t-1} \end{bmatrix} + b_I\big) \tag{2.5}$$

$$\bar{c}_t = \text{tanh}\big(W_c \begin{bmatrix} z_{t-1} \\ x_{t-1} \end{bmatrix} + b_c\big) \tag{2.6}$$

When the decision vectors F_t, I_t and $\bar{c}_t$ are calculated, the new cell state c_t is then determined by integrating these vectors using equation (2.7). The symbol $\odot$ denotes element-wise multiplication.

$$c_t = F_t \odot c_{t-1} + I_t \odot \bar{c}_t \tag{2.7}$$

Finally, the output gate (O) regulates the flow of information from the updated cell state to the new hidden state z_t. The output gate O_t determines which information from c_t, will contribute to the output z_t, encapsulating the LSTM's predictive power. Equation (2.8) shows how the output gate vector is calculated, while equation (2.9) gives the method for updating the hidden state from z_{t-1} to z_t.

$$O_t = \sigma(W_O \cdot [z_{t-1}, x_t] + b_O) \tag{2.8}$$

$$z_t = O_t \odot \tanh(\bar{c}_t) \tag{2.9}$$

These equations explain the inner workings of a single LSTM cell, individual LSTM cells are then organized sequentially. Each LSTM cell operates at a specific time step, taking input from both the current time step t and the output of the preceding time step $t-1$. To further enhance the LSTM's ability to capture certain patterns and long-range relationships within the data, multiple LSTM layers can be stacked on top of each other. Each layer consists of a sequence of LSTM cells. The output of one LSTM layer serves as the input to the next layer, propagating information hierarchically through the network.

LSTM models have gained a lot of popularity in EPF research due to their usefulness in predicting time series and capturing nonlinear relationships throughout time, thanks

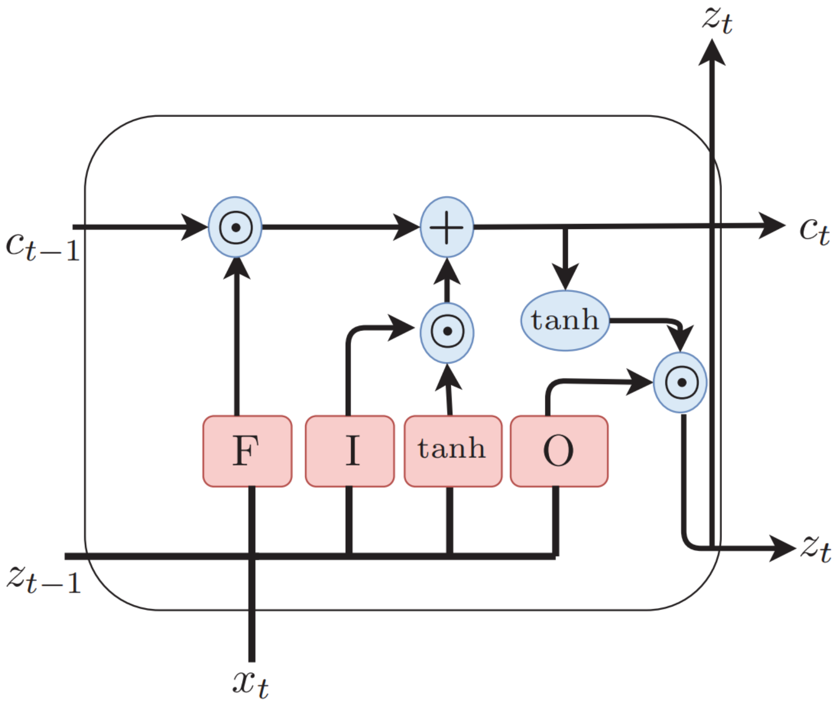


Figure 2.3: Graphical representation of a basic LSTM cell, from [29]

to their memory cells and forget gates. Some papers have shown that LSTM can outperform more simple neural networks like DNN or ANN, [23] and [4]. Other models used a special version of LSTM models, by combining it with other DL techniques to make hybrid models [5], or making the LSTM bi-directional by adding available future information [6]. These also outperform the more simple DNN technique as well as statistical ARIMA models and naive models.
More recent research [24], however, demonstrates that LSTMs in their most basic configuration fail to surpass the performance of state-of-the-art models such as GARCH or LEAR models. The extensive benchmark study [29], on the other hand, provided results showing the LSTM performed as one of the best models, outperforming LEAR but not the simple DNN model.

The existing research on LSTM models is not conclusive in proving if this technique is consistently better than benchmark statistical models. It is therefore useful to make a case study in order to check the performance of LSTM on the Belgian market with more recent data.

CNN

Convolutional neural networks (CNNs) are neural networks particularly used in image and signal processing tasks. While DNNs excel in capturing non-linear relationships in data, CNNs specialize in extracting hierarchical features from spatial data such as images or sequential data like time series [29].

A CNN comprises of 3 main parts, each designed to perform specific operations on the input data. These parts are the convolutional layers, a pooling operation, and a fully connected layer.

The convolutional layer, the core of a CNN, applies a set of learnable filters, also called kernels, to the input data. Each filter scans across the input, performing element-wise multiplication with local regions, producing feature maps that highlight relevant patterns such as edges, textures, or shapes. Each filter captures different properties of the data. Through this process, the network can detect complex patterns at different scales.

Following the convolutional layers, the pooling layers reduce the size of the feature maps, reducing their spatial dimensions while retaining essential information. Common pooling operations include max pooling, which selects the maximum value from each local region, and average pooling, which computes the average value.

After several consecutive convolutions and pooling operations, the fully connected layer classifies/predicts the data. Similar to those in DNNs, the fully connected layer integrates the extracted features to make predictions or classifications. These layers connect every neuron in one layer to every neuron in the next layer, allowing for high-level abstraction and complex decision-making based on the learned features. Finally, an activation function is applied to the output of each neuron, introducing non-linearity and enabling the network to approximate complex functions. Similar to DNNs, the parameters of a CNN, including filter weights and biases, are optimized through iterative processes such as backpropagation and gradient descent. These techniques adjust the network's parameters to minimize a certain loss function such as MAE.

Figure 2.4, taken from [29], gives an example of how a CNN can work. With data arrays of dimension 50 × 50 as inputs to produce 16 output variables. CNNs are traditionally used for image classifications but are also feasible for time series forecasts, for example in [2].

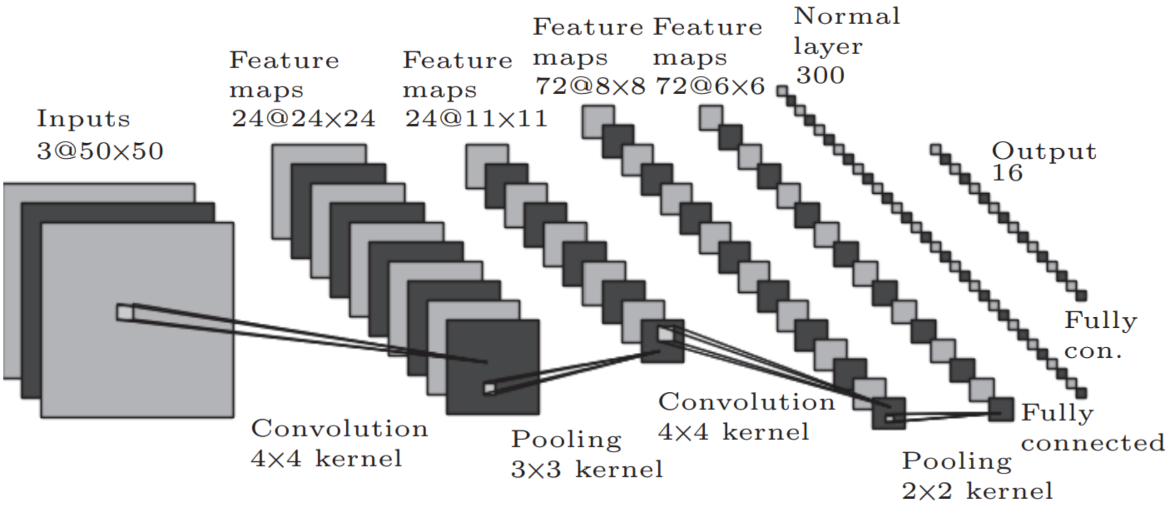


Figure 2.4: An example of a CNN, from [29]

CNN seems to perform worse than other DL techniques or even statistical auto-regressive models according to [49] and [29]. Most of the CNN models in EPF are used in a hybrid form with either LSTM or GRU (Gated Recurrent Unit). CNN when used in a hybrid form is more promising, with [26], [1] and [48] all proposing successful hybrid versions of CNN models. Because of the limited evidence of CNN being a better DL forecast than DNN or LSTM, CNNs are considered beyond the scope of this thesis.

2.2.3 Probabilistic forecasts

The models discussed up until now were all used to forecast prices as a single value, i.e. point forecasts. However, it can be more useful to forecast a range of prices at a certain hour with a certain probability. By adding uncertainty to the model, the forecast carries more information that may be used to further optimize the strategy of the market players, both on the supply and demand side. The models discussed in this overview will be Conformal Prediction (CP), a simple model-agnostic method of creating probabilistic forecasts, Distributional Deep Neural Networks (DDNN), a more sophisticated DL method predicting distributions of hourly prices, and Quantile Regression Averaging (QRA), a statistical method using quantiles,

Conformal Prediction

Conformal prediction [13] is a method of probabilistic forecasting where prediction intervals are constructed around point forecasts using only the errors or residuals as indicators for a forecast's uncertainty.

The dataset containing the historical observation is split into a training set and a calibration set. The training set is used to train the point forecast, while the calibration set is used for determining the prediction intervals. Each residual in the calibration set is given a simple non-conformity score $\lambda_i = |y_i - \hat{y}_i|$. Then, based on a pre-defined confidence level (α), the $(1 - \alpha)^{\text{th}}$ quantile of the sorted conformity scores is taken as the width of the prediction interval. Figure 2.5, from [25] shows the steps taken to obtain prediction intervals with conformal prediction.
Conformal Prediction has many advantages.

- CP is model-agnostic, which means that uncertainty can be added to any point forecast no matter which model.
- CP's only assumption is the exchangeability of errors. No assumptions on the distribution of the errors have to be made, only the fact they are i.i.d.
- CP always gives valid prediction intervals, which means that the percentage of observations outside the PI will always match the defined confidence level.

CP has recently caught on in EPF. The paper [25] and [40] show comparable or even better results than established methods such as QRA or Linear Quantile Regression. CP can also be used to improve the bidding strategy on the electricity market, with the addition of forecasted uncertainty more information is available in the decision-making process [40].

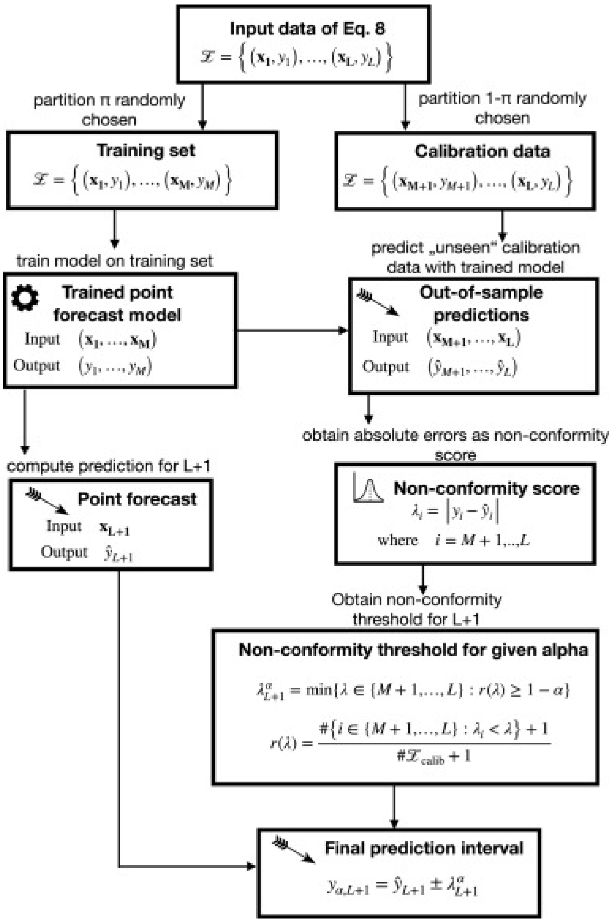
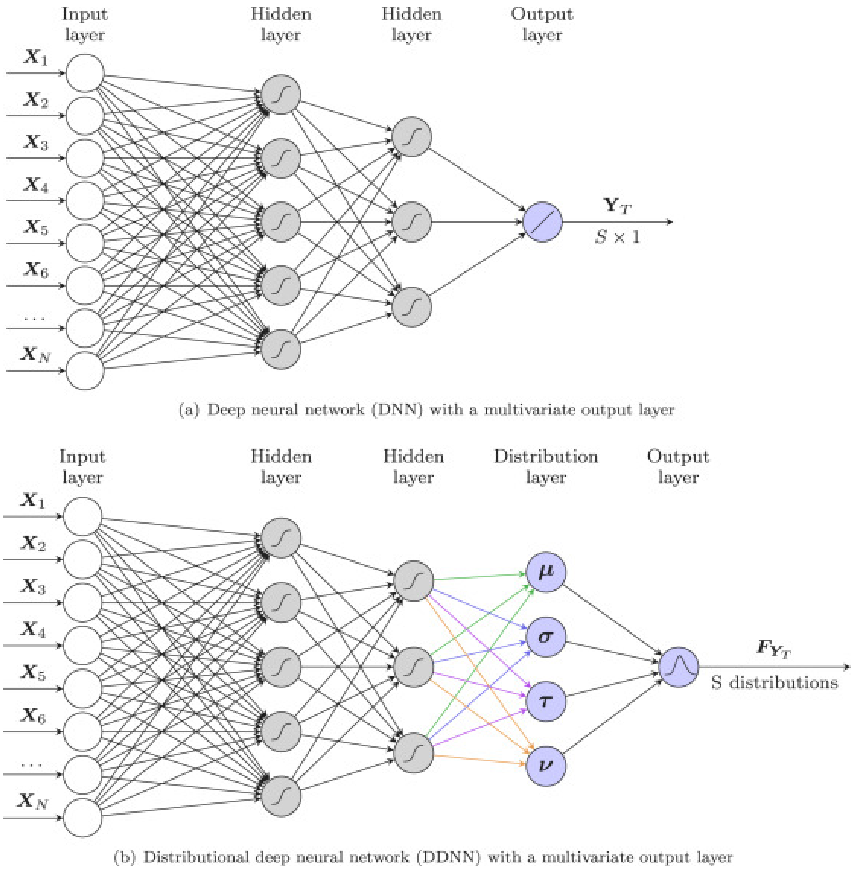

Input data of Eq. 8
$\mathcal{I} = \left\{ (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_L, y_L) \right\}$
partition π randomly chosen
partition 1-π randomly chosen
Training set
$\mathcal{I} = \left\{ (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M) \right\}$
Calibration data
$\mathcal{I} = \left\{ (\mathbf{x}_{M+1}, y_{M+1}), \ldots, (\mathbf{x}_L, y_L) \right\}$
train model on training set
predict „unseen" calibration data with trained model
Trained point forecast model
Input $(\mathbf{x}_1, \ldots, \mathbf{x}_M)$
Output $(y_1, \ldots, y_M)$
Out-of-sample predictions
Input $(\mathbf{x}_{M+1}, \ldots, \mathbf{x}_L)$
Output $(\hat{y}_{M+1}, \ldots, \hat{y}_L)$
obtain absolute errors as non-conformity score
compute prediction for L+1
Point forecast
Input $\mathbf{x}_{L+1}$
Output $\hat{y}_{L+1}$
Non-conformity score
$\lambda_i = \left| y_i - \hat{y}_i \right|$
where $i = M+1, \ldots, L$
Obtain non-conformity threshold for L+1
Non-conformity threshold for given alpha
$\lambda^{\alpha}_{L+1} = \min\{\lambda \in \{M+1, \ldots, L\} : r(\lambda) \geq 1-\alpha\}$
$r(\lambda) = \frac{\#\{i \in \{M+1, \ldots, L\} : \lambda_i < \lambda\} + 1}{\#\mathcal{I}_{\text{calib}} + 1}$
Final prediction interval
$y_{\alpha, L+1} = \hat{y}_{L+1} \pm \lambda^{\alpha}_{L+1}$

Figure 2.5: Conformal Prediction, from [25]

DDNN

The final probabilistic forecasting model under discussion is the Distributional Deep Neural Network (DDNN). Unlike the preceding methods, the DDNN employs neural networks to generate distributions of future prices rather than single price values. The operational mechanism of a DDNN closely resembles that of a DNN, with the exception of the output layer. Figure 2.6 illustrates the distinction between a DNN and a DDNN.

The first part of a DDNN is the same as a DNN, the inputs go through several hidden layers, each with its own weight, biases and activation functions. Each output of a previous layer is the input of the next layer. However, instead of providing an output of, for instance, 24 hourly prices for the following day, the DDNN generates parameters representing the assumed distribution of prices. For example, it may forecast a mean and standard deviation of the distribution. In this context, the loss function is not calculated based on Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE); instead, a loss function such as the Continuously Ranked Probability Score (CRPS) may be utilized. The CRPS evaluates the proximity of the forecasted distribution to the observed values. Alternatively, weights and biases can be optimized by maximizing the likelihood of a parametric distribution, as discussed in [35].



(a) Deep neural network (DNN) with a multivariate output layer

(b) Distributional deep neural network (DDNN) with a multivariate output layer

Figure 2.6: DNN compared to a DDNN, from [35]

The DDNN technique is a relatively recent model, introduced in 2023 by G. Marcjasz et al. [35]. This paper analysed the performance of DDNN against a naive forecast, a LEAR model combined with QRA, and a DNN model also combined with QRA. The DDNN outperformed the LEAR model both in point and interval forecasting, while the comparison against DNN was less conclusive.

Quantile Regression Averaging

Quantile regression averaging is a fairly new method developed by Nowotarski and Weron [38], where prediction intervals of the prices can be computed by applying quantile regression on a pool of point forecasts of individual forecasting models. QRA considers the point forecasts of these individual models as the regressors and the observed price as the dependent variable. The method works as follows, write $Q_p(q|\hat{p}_t)$ as the q^{th} quantile of the price, while $\hat{p}_t$, the forecasted prices, are the regressors. Then try to estimate the weights w_q by minimizing the loss function in Equation (2.11)

$$Q_p(q|\widehat{p}_t) = \widehat{p}_t w_q \tag{2.10}$$

$$\min_{w_t}\left[\sum_{\{t:p_t\geq\widehat{p}_t w_t\}} q|p_t-\widehat{p}_t w_t| + \sum_{\{t:p_t<\widehat{p}_t w_t\}} (1-q)|p_t-\widehat{p}_t w_t|\right]$$
$$=\min_{w_t}\left[\sum_t (q-\mathbb{1}_{p_t<\widehat{p}_t w_t})(p_t-\widehat{p}_t w_t)\right].$$

To make a prediction interval with confidence level c, the $(c/2)^{\text{th}}$ quantile and the $((1-c)/2)^{\text{th}}$ quantile need to be estimated for every point prediction to obtain an upper and lower bound for the interval.

According to [39] QRA has proven successful in probabilistic forecasting of electricity prices. With the QRA models in [34] and [37] outperforming the then state-of-the-art auto-regressive models. The recent papers [47] and [22] propose regularized variants of QRA that significantly outperform statistical benchmark and normal QRA models.

In light of the study's primary focus on the impact of DL techniques on EPF, QRA will not be incorporated into the case study.

2.3 Multi-step Forecasting

One of the problems behind day-ahead price prediction is that not only the next price needs to be predicted, but multiple prices at the same time, in this case, 24 hours. This section explains different strategies on how to model such a multi-step forecast. Different methods on how to estimate and train a model were discussed in Section 2.2, here strategies are explained on which inputs to consider and the amount of models to be estimated. The article of Brownlee [3] discusses 4 different strategies that are listed below.

2.3.1 Direct Multi-step

In the direct multi-step forecast strategy a separate model has to be constructed for each time step that has to be predicted. This means that $M_1(X)$ outputs the price for hour 1 with input vector X, $M_2(X)$ outputs for hour 2 with the same inputs all the way until $M_{24}(X)$ outputs the predicted price at hour 24. While a very straight-forward approach, direct multistep forecasting comes with some disadvantages. It is computationally expensive to estimate and train 24 different models. Another disadvantage of having a separate model on each hour is that each price on a certain hour is seen as independent from other prices, which is often not the case in time series forecasting.

2.3.2 Recursive Strategy

In the recursive strategy only one model is used, but the inputs change for each hour predicted. This strategy takes the predicted price of the previous hour as input for predicting the next price, and so on, hence the name recursive. This strategy is able to capture the correlation that might be present in the time series, as the prediction itself is included for the next. A disadvantage is that this strategy can cause the errors of a prediction to accumulate because they are present as inputs for the next forecasts.

2.3.3 Direct-Recursive Hybrid

The direct-recursive hybrid is a combination of the direct strategy and the recursive one. Multiple models are used for each hour, and also the predicted prices of previous hours are included as inputs for the next hour. This means that model $M_1(X)$ predicts the price $\hat{p}_1$ at hour 1 with input vector X, model $M_2(X, \hat{p}_1)$ does the same for $\hat{p}_2$ but the predicted price at hour 1 is included in the estimation of the model. Finally, model $M_{24}(X, \hat{p}_1, .., \hat{p}_23)$ outputs the price at hour 24 and all time-steps are forecasted.

2.3.4 Multiple Output

The multiple output strategy outputs multiple prices at once by the same model. This means that the output is not a single value, but a vector of values, in this case, 24 hourly prices. These models are harder to train as they need more data to avoid overfitting, but are interesting to use as they can capture relations in the data for both the inputs and the outputs.

2.4 Evaluation of EPF models

2.4.1 Accuracy

There exist several error metrics indicating the accuracy of point forecasts. Equations (2.11) - (2.14) show the various evaluation metrics discussed in [31].

$$\text{MAE} = \frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{h=1}^{24}|p_{d,h} - \widehat{p}_{d,h}| \tag{2.11}$$

$$\text{RMSE} = \sqrt{\frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{h=1}^{24}(p_{d,h} - \widehat{p}_{d,h})^2} \tag{2.12}$$

$$\text{MAPE} = \frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{h=1}^{24}\frac{|p_{d,h} - \widehat{p}_{d,h}|}{|p_{d,h}|} \tag{2.13}$$

$$\text{rMAE} = \frac{\frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{h=1}^{24}|p_{d,h} - \widehat{p}_{d,h}|}{\frac{1}{24N_d}\sum_{d=1}^{N_d}\sum_{h=1}^{24}|p_{d,h} - \widehat{p}_{d,h}^{naive}|} \tag{2.14}$$

Equation (2.11) represents the Mean Absolute Error (MAE) and is the most obvious, it captures the average error in absolute units. While this is very easy to interpret it is not easy to be used as a comparative metric against other forecasts from other datasets. The same goes for RMSE (Root Mean Squared Error), which is also represented in absolute units, euro per MWh in the case of EPF. The MAE is given in almost every EPF study but is mostly accompanied by other metrics such as RMSE, MAPE or rMAE.

MAPE (Mean Absolute Percentage Error), in equation (2.13) is not dependent on units as it is a percentage error, so it can be compared across datasets. However, these percentages become very high when the actual electricity price goes close to zero, which is often the case in the EPEX-BE market.

The rMAE, proposed in [19] uses the relative difference between the evaluated model and a naive forecast, for example, the similar day model discussed in 2.2.1. This is a ratio and not a unit so it can be compared across datasets, unlike MAE and RMSE. It also doesn't have the issue MAPE faces with near-zero prices. It is therefore also recommended as an evaluation metric in [29] and [31].

2.4.2 Test for statistical significance

Diebold-Mariano

When one model outperforms another in terms of accuracy metrics, it remains essential to assess whether these differences possess statistical significance. A commonly employed test for evaluating statistical significance is the Diebold-Mariano (DM) test [10].

To compare two models M_1 and M_2, the DM test constructs a covariance stationary loss function $L(\varepsilon_k^{M_i})$, usually the absolute value of the error or the squared errors, for each model M_i. Here $\varepsilon_k^{M_i}$ stands for the forecast error of model M_i at hour k. The loss differential $d_k^{M_1,M_2}$ is defined as follows.

$$d_k^{M_1,M_2} = L(\varepsilon_k^{M_1}) - L(\varepsilon_k^{M_2}) \tag{2.15}$$

The one-sided DM test then evaluates the null hypothesis that the expected loss differential between M_1 and M_2 is greater than or equal to zero. In other words, the null hypothesis states that M_1 has an accuracy equal or worse than M_2. If H_0 is rejected, it means that M_1 performs statistically significantly better on the predicted prices than M_2. The equation for the DM-test, taken from [29], is found in equation (2.16).

$$\begin{aligned} H_0 &: E[d_k^{M_1,M_2}] \geq 0, \\ H_1 &: E[d_k^{M_1,M_2}] < 0. \end{aligned} \tag{2.16}$$

Giacomini-White

The Giacomini-White (GW) test [14] offers an alternative method to assess the conditional predictive accuracy of forecasting models, differing from the unconditional evaluation provided by the DM test. While the DM test focuses on comparing forecasts directly, the GW test incorporates lagged loss differential values to assess how past errors impact future predictions.
The GW test is formulated in equation (2.17), where X_{d-1} represents the lagged loss differential values.

$$d_k^{M_1,M_2} = \phi' X_{d-1} + \varepsilon_k \tag{2.17}$$

The one-sided version of the GW test has a null hypothesis that $\phi' \leq 0$, suggesting that the error of Model 1 is expected to be smaller compared to Model 2, thus performing better. Interpretation of the GW test aligns closely with that of the DM test, focusing on statistical significance to determine which model demonstrates better predictive accuracy. The GW test can be executed either in a univariate way, yielding individual p-values for each hour, or in a multivariate manner, producing a single p-value encompassing all 24 hours. This flexibility allows for a comprehensive evaluation of conditional predictive performance. Overall, the GW test complements the DM test by providing insights into how past forecasting errors influence current predictions, offering a different perspective on model effectiveness in capturing conditional dependencies within time series data.

2.4.3 Reliability

Reliability is a characteristic of a probabilistic forecast that can be utilized to assess models of this nature.

Assessing the reliability of a probabilistic forecast involves verifying whether the observed values align with the nominal coverage guaranteed by the model. When a prediction interval with a nominal coverage of e.g. 90 percent is forecasted, it would be expected that 90 percent of the observations in the test set are within the prediction interval. To test reliability means essentially to check if the given uncertainty matches the empirical observations.

The nominal coverage is decided upon independently of the model, while the empirical coverage uses the observations to calculate its ratio. Call I_t the indicator that is 1 observation *t* falls within the forecasted prediction interval (a 'hit') and 0 otherwise (a 'miss').

$$\mathrm{I}_t = \begin{cases} 1 & \text{if } P_t \in [\hat{L}^t, \hat{U}^t] \rightarrow \text{'hit'}, \\ 0 & \text{if } P_t \notin [\hat{L}^t, \hat{U}^t] \rightarrow \text{'miss'} \end{cases} \tag{2.18}$$

To test if the empirical coverage equals the nominal coverage, the Kupiec test can be used [27]. This test checks whether the probability that $I_t = 1$, also known as the *unconditional coverage* equals the pre-set confidence level. If this probability significantly differs from the nominal confidence level, the null hypothesis that the two coverages are the same is rejected, suggesting that the probabilistic forecast may not be reliable.

The Kupiec test uses as a test statistic a Likelihood Ratio statistic presented in equation (2.19) and has a χ^2 distribution. In the LR statistic, c is the confidence level (90 percent in our example), $\pi = n_1/(n_0 + n_1)$ is the ratio of 'hits' to the total number of observations, with n_1 being the number of '1's in I_t and n_0 the number of '0's.

$$\mathrm{LR}_{\mathrm{UC}} = -2\log\left\{\frac{(1-c)^{n_0} \cdot c^{n_1}}{(1-\pi)^{n_0} \cdot \pi^{n_1}}\right\} \tag{2.19}$$

2.4.4 Sharpness

The sharpness of a forecast refers to how dense the predicted intervals are. In other words, how concentrated is the forecasted uncertainty? To measure the sharpness a simple sharpness score can be considered, by taking the average width of the forecasted prediction interval. More sophisticated measures include the Pinball Loss or the CRPS.

Average width

One type of sharpness score is the average width of the forecasted prediction interval. This score simply takes the width of each predicted price interval accross all hours predicted and calculates the average. This means that probabilistic models with a high sharpness score have more uncertainty than the models with low scores. Equation (2.20) shows how the average width $\overline{\delta}^{(c)}$ is calculated across a dataset with T timestamps. The upper bound of the prediction interval at time t is presented as $\overline{\alpha_t}$, and the lower bound as $\underline{\alpha_t}$.

$$\begin{aligned} \delta_t^{(c)} &= \overline{\alpha_t} - \underline{\alpha_t} \\ \overline{\delta}^{(c)} &= \frac{1}{T}\sum_{t=1}^{T} \delta_t^{(c)} \end{aligned} \tag{2.20}$$

Pinball Loss

Pinball loss is a type of piecewise linear loss function that is calculated using Equation (2.21) from [39].

$$\textit{Pinball}\left(\widehat{Q}_{P_t}(q), P_t, q\right) = \begin{cases} (1-q)\left(\widehat{Q}_{P_t}(q) - P_t\right), & \text{for } P_t < \widehat{Q}_{P_t}(q), \\ q\left(P_t - \widehat{Q}_{P_t}(q)\right), & \text{for } P_t \geq \widehat{Q}_{P_t}(q), \end{cases} \tag{2.21}$$

Here q is the quantile the probabilistic forecast is trying to predict, $\widehat{Q}_{P_t}(q)$ is the price forecast at quantile q and P_t is the observed price. This is a function that penalizes when the predicted price differs too much from the observed price, which indicates a wider interval, thus measuring the sharpness of a model. The disadvantage of the pinball loss score is that it can only be calculated for one specific quantile and not for a prediction interval. The pinball loss is used in QRA, discussed in 2.2.3, as the loss function when performing the regression [38].

CRPS

CRPS, or Continuous Ranked Probability Score, is a technique measuring the sharpness of density forecasts. Density forecasts predict the entire distribution density of the error at time t instead of just a prediction interval. Density forecasts can be achieved by predicting the parameters of the distribution, or by generating multiple prediction intervals at different quantiles.

The concept underlying CRPS [15] is to contrast the predicted density with the density of an ideal forecast, which possesses a 100 percent probability of matching the observation [17]. The CRPS is essentially just an integral of the squared difference between this perfect forecast's density and the model's predicted density. Equation (2.22) shows how the CRPS is calculated for a density forecast.

$$\mathit{CRPS}(\widehat{F}_{P_t}, P_t) = \int_{-\infty}^{\infty} \left(\widehat{F}_{P_t}(x) - 1_{\{P_t \leq x\}}\right)^2 dx \tag{2.22}$$

Here $\widehat{F}_{P_t}$ represents the cumulative distribution function (CDF) of the forecasted price uncertainty, while $1_{\{P_t \leq x\}}$ represents the CDF of an observation, which would be the perfect distribution. Figure 2.7 from [12] visualises CRPS, the integral from the CRPS function means essentially to calculate the squared area (in red) between the CDF of the forecasted price and the observation. A smaller area means a sharper forecast.

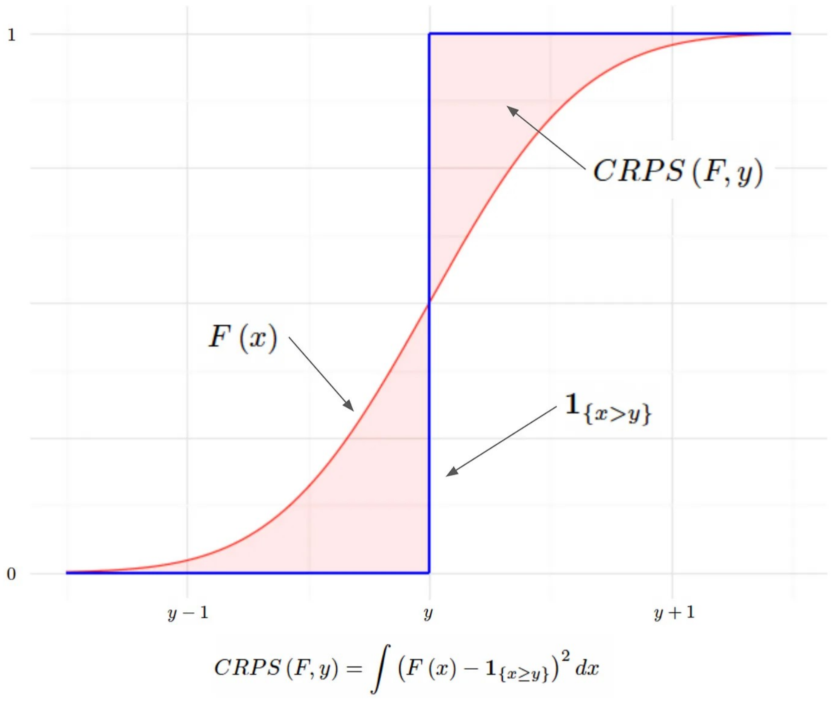


Figure 2.7: Visual representation of CRPS, from [12]

Chapter 3

Dataset

To perform accurate EPF research, it is important to have enough historical data to be able to both train and test the models and to include the most recent data [31], as well as the inclusion of exogenous variables to improve the forecast's accuracy. This chapter explains each dataset that was added to one of the forecasts as an input feature to predict the day-ahead prices on the Belgian spot market.

In the context of this thesis, a dataset comprising six years of hourly BELPEX spot prices spanning from January 1st, 2018, to December 31st, 2023, is employed. Additionally, for multivariate forecasting models, various exogenous variables are incorporated alongside the historical price data. These include other time series data such as the Belgian load and generation forecast for that same period and the Belgian wind and solar forecasts. Using forecasts as inputs instead of actual values of covariates can be justified by the fact that, during prediction time, the model lacks access to the real-time actual values of these covariates. A variable indicating the day of the week is also used in each model.

3.1 Belgian EPEX spot prices

As described in Section 2.1, electricity on the day-ahead market in Belgium is traded together with many other European markets on the EPEX spot market with EPEX as the market regulator. This means that neighbouring markets such as the German or the French markets have an influence on the Belgian electricity price. This study will only take into account the historical prices in the Belgian market to save the model from having too many variables and remain parsimonious. All price data was extracted from the European ENTSO-e transparency platform [11]. Figure 3.1 shows the evolution of said price from 2018 to the end of 2023 in €/MWh. The initial 4.5 years of prices will serve as training data to estimate the model's parameters, depicted in blue. The subsequent 0.5 years (equating to 10% of the training data) will be allocated for validation purposes. Finally, the entire year of 2023 will constitute the test set, utilized to evaluate the model's performance, highlighted in red.
Notice that the variability of the data in the training set is much larger than that of the test set. This discrepancy in data variability between the training and test sets can potentially impact the model's generalization performance. The greater variability within

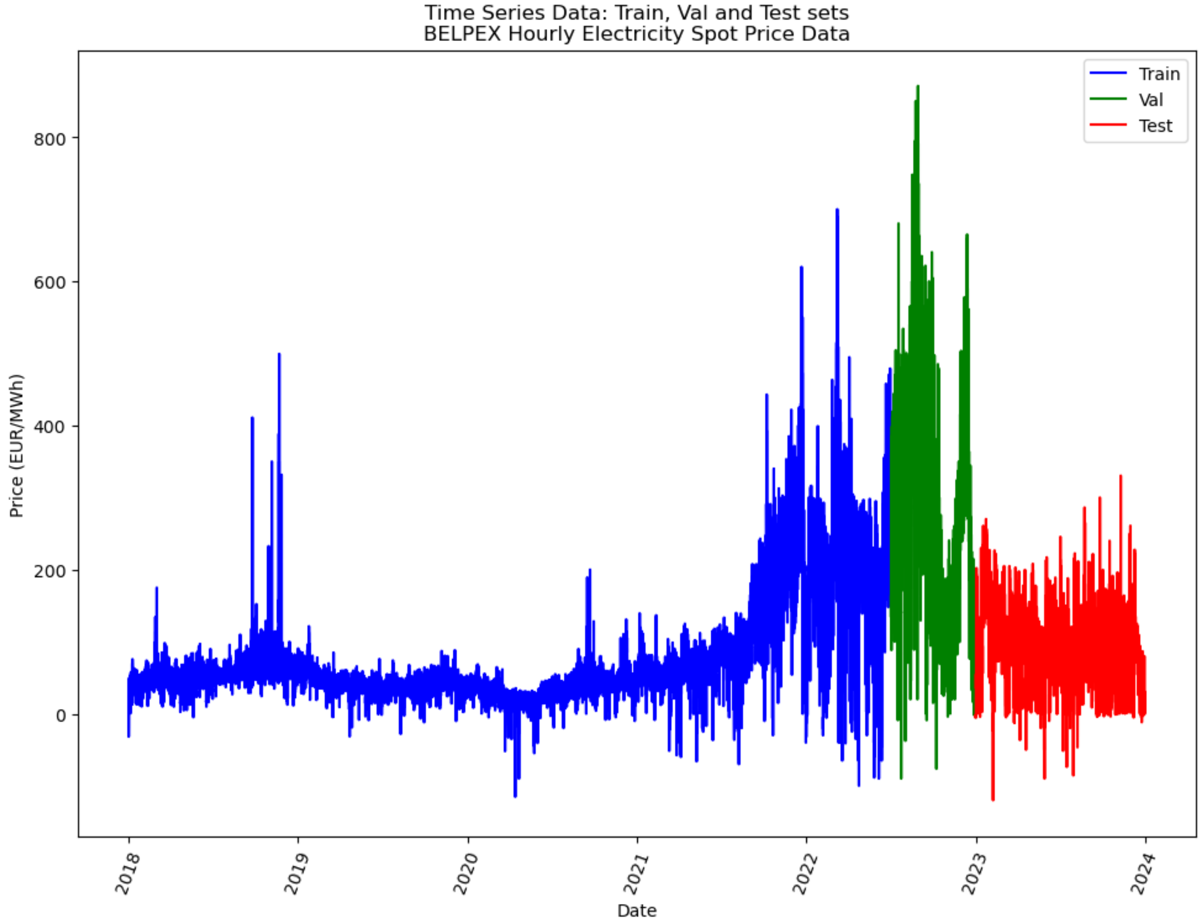


Figure 3.1: Belgian electricity prices 2018-2023 with train, val and test split

the training set implies that the model is exposed to a wider range of scenarios during training, which may aid in its ability to capture diverse patterns and relationships in the data. However, the model's performance on the test set, which typically represents unseen data, may be influenced by this variability and may require robust generalization capabilities to effectively handle such variations.

3.2 Other covariates

3.2.1 Load and generation forecasts

To explore the effect of adding exogenous variables to the model, the Belgian load and generation day-ahead forecasts are clear candidates because of their direct effect on the price clearing. One would expect the accuracy to increase when the load and generation of the next day are known. In figures 3.2 and 3.3, the load and generation forecasts in MW are plotted from 2018 until the end of 2023. Both forecasts exhibit similarities, as the generation and load profiles of an electricity market inherently mirror the prevailing demand for electricity, characterized by heightened demand during winter months and diminished demand during summer periods.

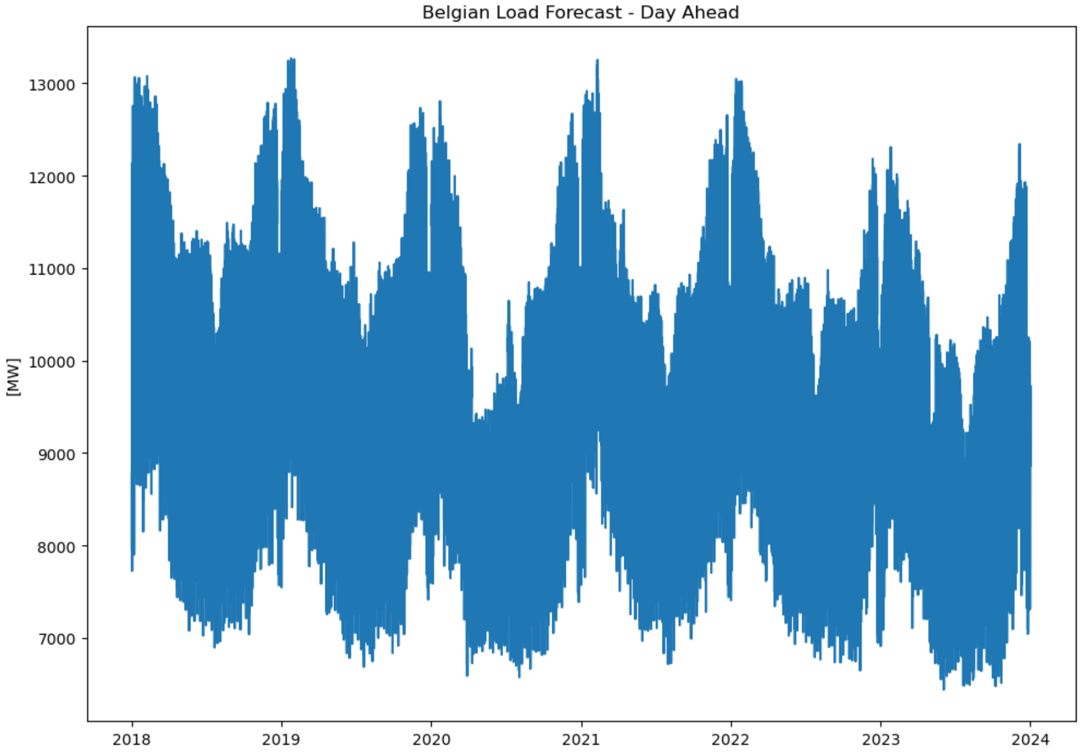


Figure 3.2: Belgian DA load forecast 2018-2023

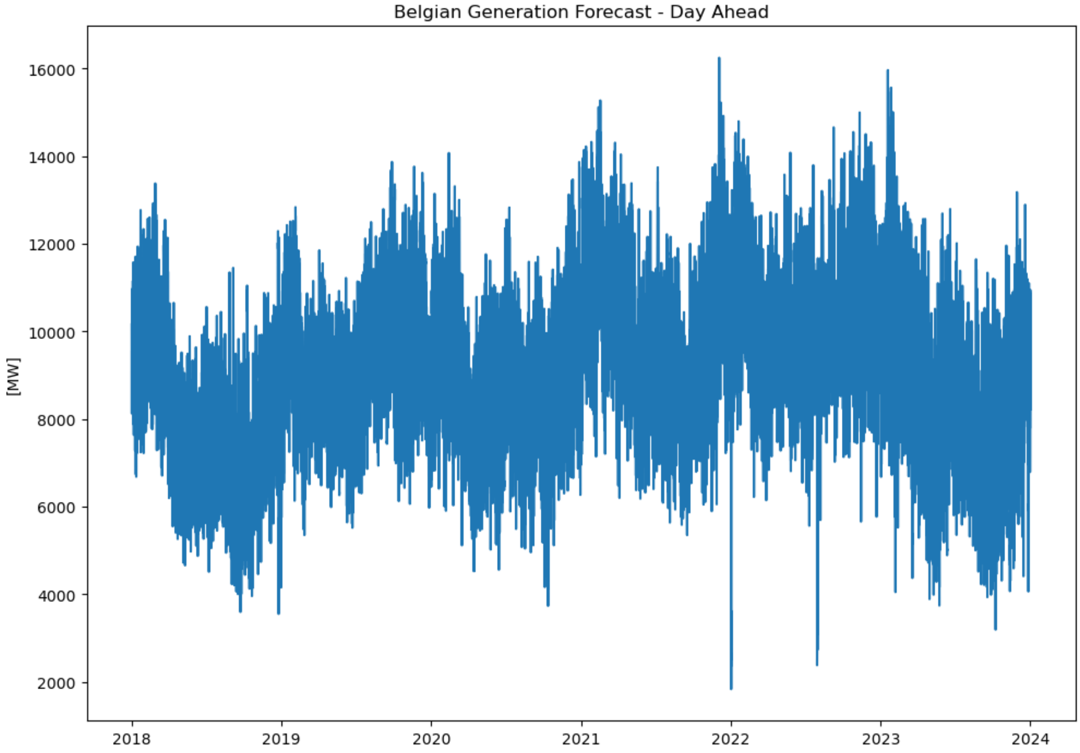


Figure 3.3: Belgian DA generation forecast 2018-2023

3.2.2 Wind and solar forecasts

The day-ahead forecasts of the Belgian wind and solar generation can also be added to the model. Because of the high uncertainty that these types of generations bring with them it is useful to include them in the multivariate model. Moreover, wind and solar generation are often highly correlated to electricity prices. Figure 3.4 and 3.5 show the solar and wind forecasts in Belgium in MW. The solar generation forecasts are more predictable and follow a clear seasonal pattern increasing in amplitude, while wind generation tends to be more volatile although also increasing its peak generation over time.

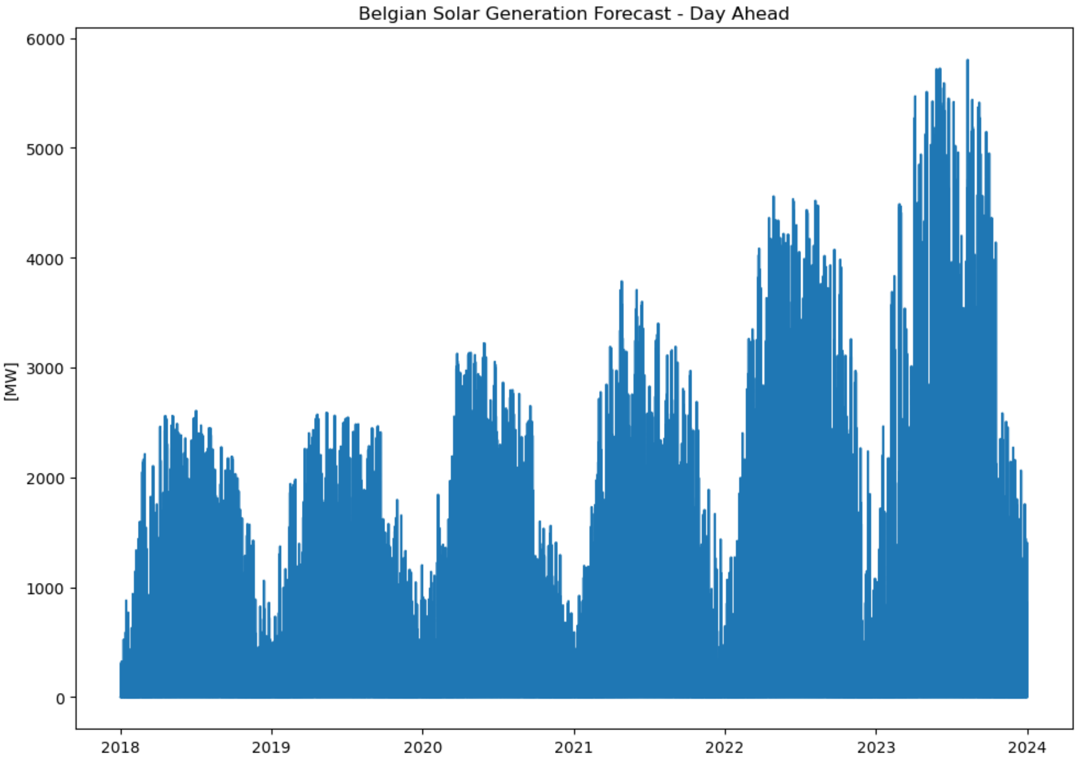


Figure 3.4: Belgian DA solar generation forecast 2018-2023

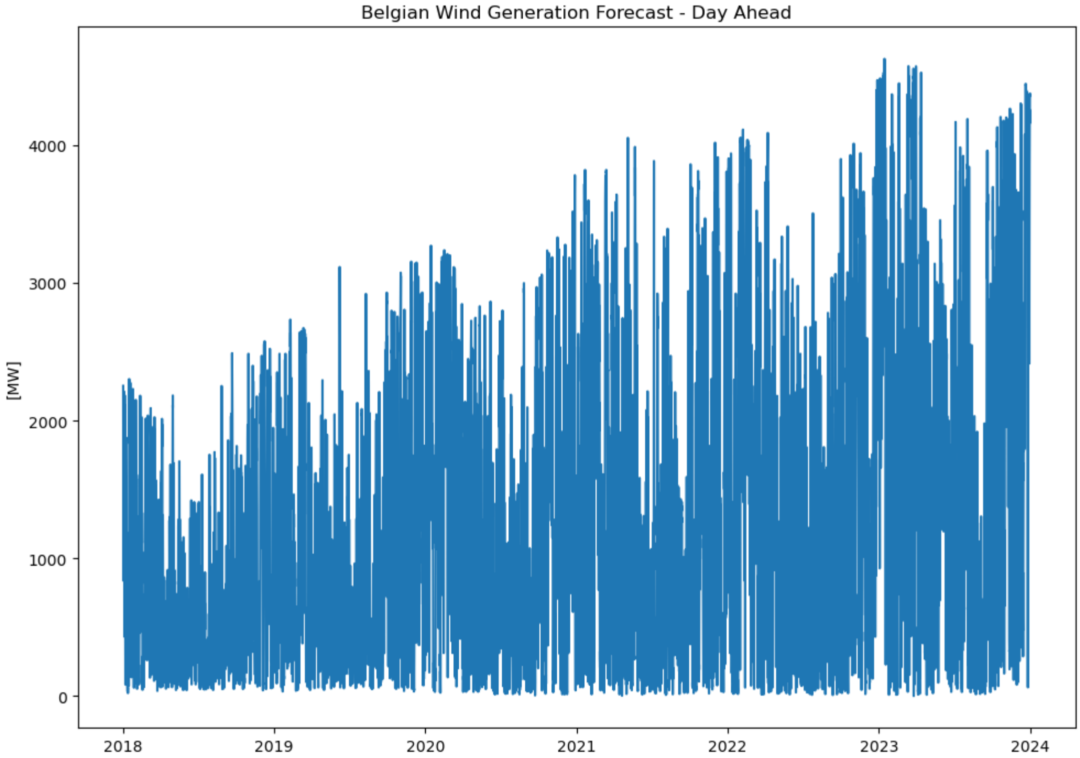


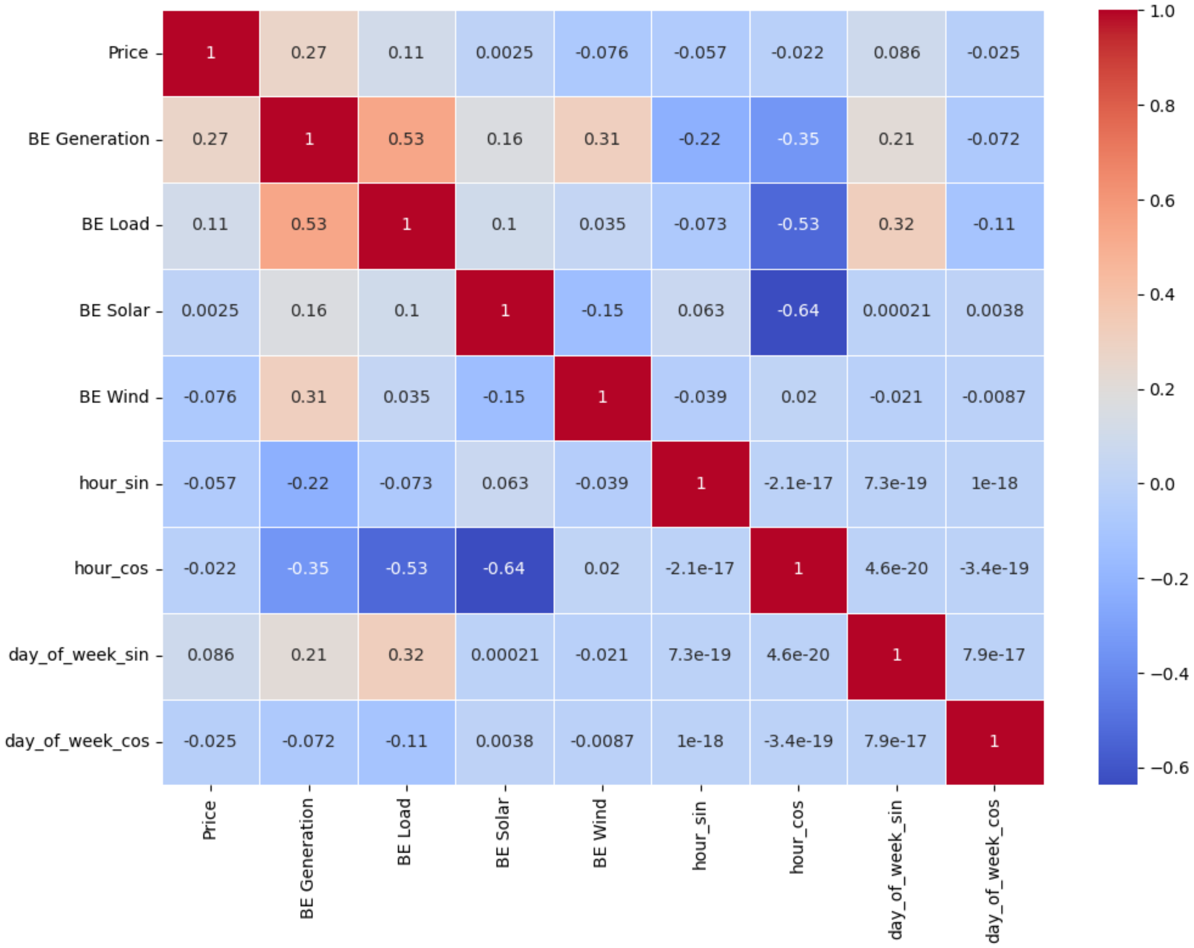
Figure 3.5: Belgian DA wind generation forecast 2018-2023

3.2.3 Day of the week

As the last input feature, weekday variables were used to indicate the day of the week. Variables for the hour of the day are omitted from the model due to the high correlation with the already implemented solar forecasts. Electricity demand can vary strongly throughout the week. It is therefore possible that models with weekday and hour-of-the-day dummies improve the forecast's accuracy. To represent the cyclic nature of weekdays (from Monday to Sunday), cyclical encoding was used [36]. Cyclical encoding transforms these weekday indicators into a continuous space by using sine and cosine functions. This approach ensures that the difference between, for example, Sunday and Monday in the encoding is similar to the difference between Monday and Tuesday, thus maintaining a consistent representation of the cyclic weekday pattern within the model.

3.2.4 Correlation of the input data

Figure 3.6 shows the correlation plot of all the input features discussed in this section. Most of the features are independent. There are some correlations visible, such as the cyclical variables indicating the hour and the solar generation, which makes sense since solar energy is directly related to the hour of the day. The hourly variables are also correlated to the load and to a lesser extent to the generation, which is another reason to omit the hour of the day from the model. Load and generation are also positively correlated because they both reflect the demand for energy at a certain point in time.



	Price	BE Generation	BE Load	BE Solar	BE Wind	hour_sin	hour_cos	day_of_week_sin	day_of_week_cos
Price	1	0.27	0.11	0.0025	-0.076	-0.057	-0.022	0.086	-0.025
BE Generation	0.27	1	0.53	0.16	0.31	-0.22	-0.35	0.21	-0.072
BE Load	0.11	0.53	1	0.1	0.035	-0.073	-0.53	0.32	-0.11
BE Solar	0.0025	0.16	0.1	1	-0.15	0.063	-0.64	0.00021	0.0038
BE Wind	-0.076	0.31	0.035	-0.15	1	-0.039	0.02	-0.021	-0.0087
hour_sin	-0.057	-0.22	-0.073	0.063	-0.039	1	-2.1e-17	7.3e-19	1e-18
hour_cos	-0.022	-0.35	-0.53	-0.64	0.02	-2.1e-17	1	4.6e-20	-3.4e-19
day_of_week_sin	0.086	0.21	0.32	0.00021	-0.021	7.3e-19	4.6e-20	1	7.9e-17
day_of_week_cos	-0.025	-0.072	-0.11	0.0038	-0.0087	1e-18	-3.4e-19	7.9e-17	1

Figure 3.6: Correlation plot full dataset

Chapter 4

Methods

The following chapter presents the methodology used to analyse the effects of deep learning on electricity price forecasting. In line with the first objective of the thesis, which concerns the improvement in forecast accuracy, three types of point forecasts were used to make various price predictions.

- An LSTM model that uses deep learning to predict future prices.
- A LEAR model, an auto-regressive model that uses LASSO for feature selection. This model represents the state-of-the-art statistical method to compare against deep learning methods.
- A Similar-day model which is a naive forecast used as baseline model to calculate certain error metrics and make comparisons.

To achieve the other objective which focuses on the quantification of uncertainty within EPF, two types of interval forecasts were generated.

- A conformal predictor, which is a distribution-free and model-agnostic model that uses the point forecast's errors to construct prediction intervals.
- A Distributional Deep Neural Network (DDNN). A deep learning method that generates a predicted distribution of prices instead of single point values.

The methodology for constructing, evaluating and comparing the various forecasts is explained in detail in the following distinct sections.

Section 4.1 discusses the point forecasts, how the recursive multi-step forecast is built for both the LSTM method, as well as the statistical LEAR model and the naive model. Section 4.2 focuses on how the interval forecasts were constructed to quantify uncertainty, both for conformal prediction and for the DDNN. Lastly, in Section 4.3 the structure of the analysis together with the evaluation methods applied on the various forecasts are described.

4.1 Construction of point forecasts

The LEAR forecast model was made using the EPF toolbox library in Python developed by J. Lago [28] in 2020. The LSTM models, and probabilistic forecasts were made with

Python using the tensorflow library. Further detail of the point forecasts are explained in the following section.

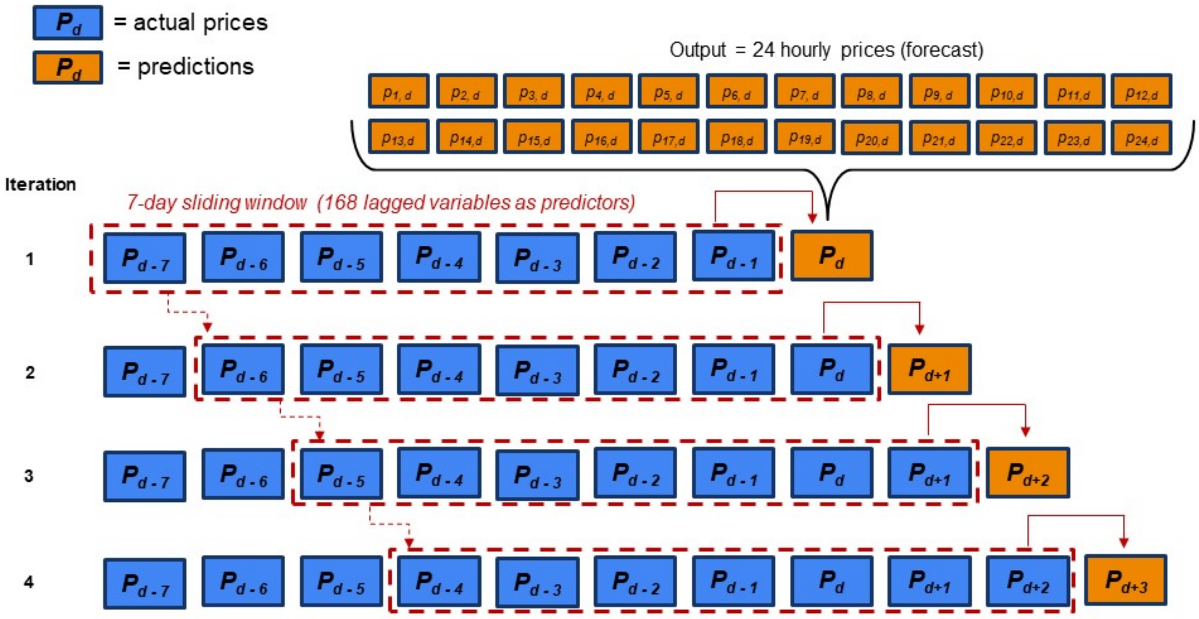
4.1.1 Recursive Multi-Step Multi-Output forecast

The forecasting methodology employed in this thesis is referred to as a recursive multi-step forecast. The different multi-step strategies were described in Section 2.3. This model can be seen as a combination of the recursive strategy and the multiple output strategy.

In LSTM data modeling, the sliding window technique involves creating a set of predictions based on a window of consecutive samples from the dataset. The *recursive* aspect of this approach entails iteratively moving through the dataset, where with each time step, new data is added to the model as input features, while the oldest data is removed from the predictor set. This ensures that newly predicted values are used as inputs with each iteration, forming a so-called sliding window predicting the next set of *H* values.

Multi-Step means that each time-step, multiple prices are predicted at once, in this case, 24 hours each iteration. The sliding window therefore moves in steps of 24 hours.

Figure 4.1 visualises how such a forecast would work. The hourly data of the past 7 days is used as input to predict the next 24 hourly prices. The window then slides on to the next day, when new actual data is added into the sliding window to create a new forecast.



P_d = actual prices
P_d = predictions
Output = 24 hourly prices (forecast)
$p_{1,d}$ $p_{2,d}$ $p_{3,d}$ $p_{4,d}$ $p_{5,d}$ $p_{6,d}$ $p_{7,d}$ $p_{8,d}$ $p_{9,d}$ $p_{10,d}$ $p_{11,d}$ $p_{12,d}$
$p_{13,d}$ $p_{14,d}$ $p_{15,d}$ $p_{16,d}$ $p_{17,d}$ $p_{18,d}$ $p_{19,d}$ $p_{20,d}$ $p_{21,d}$ $p_{22,d}$ $p_{23,d}$ $p_{24,d}$
Iteration
7-day sliding window (168 lagged variables as predictors)
1 P_{d-7} P_{d-6} P_{d-5} P_{d-4} P_{d-3} P_{d-2} P_{d-1} P_d
2 P_{d-7} P_{d-6} P_{d-5} P_{d-4} P_{d-3} P_{d-2} P_{d-1} P_d P_{d+1}
3 P_{d-7} P_{d-6} P_{d-5} P_{d-4} P_{d-3} P_{d-2} P_{d-1} P_d P_{d+1} P_{d+2}
4 P_{d-7} P_{d-6} P_{d-5} P_{d-4} P_{d-3} P_{d-2} P_{d-1} P_d P_{d+1} P_{d+2} P_{d+3}

Figure 4.1: Visual representation of a recursive multi-step forecast

4.1.2 LSTM

As described in Section 2.2.2, LSTM is a recurrent neural network that can be useful especially for predicting future values of a time series since it is able to capture long-term relationships.

The LSTM neural network in this thesis is used as a recursive multi-step forecast and takes the 168 lagged prices and other exogenous variables to generate a forecast of the next 24 hours.

Hyperparameters

An LSTM network has several hyperparameters that need to be tuned while training the model [20]. These variables include

- **Horizon**. The forecasting horizon is a hyperparameter which can also be tuned. However, in the situation of day-ahead price forecasting, it is set to 24 hours because of practical reasons that require at least a 24-hour forecast to place bids on the spot market.
- **Loss function**. The metric that defines the performance of a model. The parameters (weights) of the model are optimized with the objective of minimizing the associated loss function. In this case, MAE will be used as a loss function.

The horizon and the loss function were chosen in advance to build the model. The hyperparameters below were optimized through hyperparameter tuning.

- **Layers**. The amount of stacked LSTM layers on top of each other.
- **Learning rate**. The learning rate is the hyperparameter that determines how much the model weights are adjusted with respect to the loss gradient during training.
- **Batch size**. The batch size refers to the number of training examples used in one iteration of gradient descent, the optimization algorithm, before updating the model's weights.
- **Number of epochs**. The number of epochs refers to the number of times the LSTM model will iterate over the entire training dataset during the training process. Each epoch consists of one forward pass (computing loss) and one backward pass (updating model parameters) of all the training examples.

These were optimized using random search [45], an algorithm for hyperparameter optimization that takes random values in a predefined parameter space. The algorithm trains the model for these random values and chooses the hyperparameters with the best performance. This process iterates a certain number of times to end up with the tuned hyperparameters, in this study 20 different configurations, called samples, for random search were used.

Early Stopping

Early stopping is a technique employed during training to halt the process when a monitored metric, in our case Mean Absolute Error (MAE) based on validation set performance, ceases to improve. Specifically, we terminated the training process prematurely after observing no improvement in validation loss for two consecutive epochs. This strategy mitigates the risk of overfitting and promotes optimal model generalization.

Feature normalization

After experimenting with various scalers (MinMax 0 to 1, MinMax -1 to 1, Robust, Standard scaling), it was found that MinMax 0 to 1 consistently yielded the best performance in terms of our evaluation metrics and thus adopted it for subsequent work. Scaling data is important to ensure that all features contribute equally to model training. Additionally, it is a good practice to fit the scaler on the training data and then use it to transform the testing data, thus preventing data leakage during the model testing process and this is exactly what was done.

4.1.3 LEAR

The LEAR model, discussed in 2.2.1 is a statistical regression model which uses LASSO for feature selection. Different to the model developed by [46] and used in the preceding thesis [9], all 7 days lagged are used as inputs. This means there are 168 input variables for each time series included in the model.

Input features

The input features, discussed in Section 3 include the hourly BELPEX prices of the previous 7 days, alongside the hourly day-ahead load forecast, generation forecast as well as wind and solar generation forecasts. Which variants of exogenous variables were used to compare the influence of those exogenous variables are discussed in Section 4.3. This means that to predict a price at hour h and day d with n exogenous variables, a vector of $168 \times (n+1)$ elements is used as input.

Before the parameters are estimated a LASSO feature selection takes place. This method works by imposing a penalty on the absolute size of the coefficients in the regression model, encouraging some coefficients to be exactly zero. The optimal regularization parameter λ is chosen based on the Akaike Information Criterion (AIC), ensuring a balance between model complexity and predictive accuracy.

Model estimation

Equation (4.1) summarizes how such a LEAR regression would look like.

$$\begin{aligned}\hat{p}_{d,h} = \alpha^0_{d,h} &+ \beta^1_{d-1,1}p_{d-1,1} + ... + \beta^1_{d-1,24}p_{d-1,24} + ... + \beta^1_{d-7,0}p_{d-7,0} + ... + \beta^1_{d-7,24}p_{d-7,24} \\ &+\beta^2_{d-1,1}X^1_{d-1,1} + ... + \beta^2_{d-1,24}X^1_{d-1,24} + ... + \beta^2_{d-7,1}X^1_{d-7,1} + ... + \beta^2_{d-7,1}X^1_{d-7,1} + ... \\ &+\beta^{n+1}_{d-1,1}X^n_{d-1,1} + ... + \beta^{n+1}_{d-1,24}X^n_{d-1,24} + ... + \beta^{n+1}_{d-7,1}X^n_{d-7,1} + ... + \beta^{n+1}_{d-7,24}X^n_{d-7,24}\end{aligned} \tag{4.1}$$

Here, $\hat{p}_{d,h}$ is the predicted price at hour h on day d, β is a regression parameter estimated to minimize MAE, X^i represents an exogenous variable and p is an actual historic price. The full model with the most exogenous variables had four exogenous variables, which means the LEAR model had at most 840 $(168 \times (4+1))$ parameters to estimate for every hour. Although the LASSO feature selection can significantly reduce the number of parameters estimated.

4.1.4 Naive forecast

The case study will implement a naive forecast for several reasons. One reason is to compare with the proposed LSTM model to see if it outperforms a simple baseline model as a minimum requirement. Another reason is that naive predictions are used in the calculation of rMAE, one of the error metrics used in the case study. The naive model chosen is a similar-day forecast as described in 2.2.1, where the prices of a day are equal to the prices of last week in the case of Saturday, Sunday and Monday. In the case of Tuesday to Friday, the prices of the day before are taken as the predicted price. Equation (4.2) describes the naive model.

$$\hat{p}^{\text{naive}}_{d,h} = \begin{cases} p_{d-1,h}, & \text{if } d \text{ is Tue, Wed, Thu, or Fri,} \\ p_{d-7,h}, & \text{if } d \text{ is Sat, Sun, or Mon.} \end{cases} \tag{4.2}$$

4.2 Construction of interval forecasts

This section discusses the methods used to construct the interval forecasts. In line with the second objective of the thesis, these probabilistic forecasts were then compared against each other in terms of reliability and sharpness to assess the performance of the sophisticated deep learning technique against the more simple conformal prediction approach in terms of quantification of uncertainty.

4.2.1 Conformal Predictor

Conformal Prediction (CP) is a simple yet effective way of constructing prediction intervals around point prediction. To perform CP, the test dataset is split further into a calibration set and a test set. The absolute errors of the calibration set are then sorted to make non-conformity scores. These scores give an indication of how well the point forecast fits the actual data. In this case, the absolute value of an error equals its non-conformity score.

To make a prediction interval of level *q*, the *q*$^{\text{th}}$ percentile of non-conformity scores is taken and that value becomes the width of the prediction interval. This guarantees statistically correct coverage, where the confidence level actually equals the percentage of observations that fall outside of the prediction interval. To avoid having a prediction interval with a constant width over the entire test set, the errors are put into $7 \times 24 = 168$ separate lists of equal length according to their specific timestamp. This makes the intervals more variable depending on which day of the week and which hour of the day is being predicted and carries more information about the prediction itself.

4.2.2 Distributional Deep Neural Network

The Distributional Deep Neural Network was constructed using one of the LSTM models with an additional distributional layer added. The prices are assumed to be normally distributed, which means that the distributional layer outputs a mean and a standard deviation in order to predict the density of each of the 24 hourly day-ahead prices.

The hyperparameters of the DDNN are the same as the LSTM model described in Section 4.1.2, with the addition of a loss function to optimize the weights and biases of the distributional layer. This loss function is the negative log-likelihood and is shown in equation (4.3), taken from [32]. This loss function has to be minimized in order to yield the best results for the DDNN.

$$l(\theta) = -\sum_{i=1}^{n} \big(y_i \log \hat{y}_{\theta,i} + (1 - y_i) \log(1 - \hat{y}_{\theta,i}\big) \tag{4.3}$$

Here θ presents one of the distributional parameters, with $\hat{y}_{\theta,i}$ the predicted price at time i using parameter θ and y_i is the actual price at time i.

Section 4.3 discusses the different configurations of LSTM models: the different calibration windows, number of lags and recalibration frequencies. The LSTM model on which the DDNN was based, is the model with the highest accuracy. The best LSTM model, shown in Section 5.1.5 turned out to be the model with the load and generation forecasts included as exogenous variables, a calibration window of 5 years (full history) and a weekly recalibration.

4.3 Evaluation and analysis

The following section describes the structure of the analysis on the different forecasts, as well as the error metrics used to obtain the results of this thesis. The section is split into two main parts, firstly the analysis and evaluation of the point forecast, and secondly those of the interval forecasts.

4.3.1 Point forecasts

The first objective of this thesis is to study the impact of deep learning on EPF. For this purpose, the LSTM forecast will be compared against a naive benchmark and a statistical state-of-the-art forecast (LEAR) used in [9]. Multiple parameters were tested on the forecast to analyse which parametric settings have the best impact. The parameters that will be tested are calibration window, recalibration frequency and number of lagged variables.

Default Model

When comparing the different parametric values there is a need for a default setup for all other variables to have a ceteris paribus comparison. The default consists of a full history CW, 7 days of lagged prices and no recalibration frequency. These parameters are discussed in further detail in this section.

MAE and rMAE

For the point forecasts Mean Absolute Error (MAE) and relative Mean Absolute Error (rMAE) will be used as error metrics to compare the different models. The lower the error metric, the better the model performs.

The reason behind MAE is that it is very easily understood and interpreted, it is just the average absolute price difference of the predicted and the actual values and can be expressed in €/MWh. Besides this advantage, MAE makes it hard to compare models tested on different datasets, which uses other time periods or electricity markets. A bad model tested on a relatively stable price period might have a better MAE than a very good model tested on a very volatile period in the same market. rMAE is arguably a better error metric as it is based on the relative difference in performance with a simple naive forecast. In this study, the naive model with daily and weekly seasonality described in 4.1.4 will be used as the naive forecast.

Equations (1) and (2) represent the method the MAE and rMAE are calculated respectively.

$$MAE = \frac{1}{24N_d} \sum_{d=1}^{N_d} \sum_{h=1}^{24} |p_{d,h} - \widehat{p}_{d,h}| \tag{4.4}$$

$$rMAE = \frac{\frac{1}{24N_d} \sum_{d=1}^{N_d} \sum_{h=1}^{24} |p_{d,h} - \widehat{p}_{d,h}|}{\frac{1}{24N_d} \sum_{d=1}^{N_d} \sum_{h=1}^{24} \left|p_{d,h} - \widehat{p}_{d,h}^{naive}\right|} \tag{4.5}$$

Multivariate vs Univariate

Adding exogenous variables as input features can both increase or decrease the model's accuracy. To test for the effects of exogenous variables on accuracy the aforementioned MAE and rMAE were calculated first for the univariate LSTM and LEAR model (the ”Base” models), only considering the price lags and day of the week. After which the covariates described in 3.2 were added one by one, these models were used to generate comparisons of the effect on accuracy with varying calibration windows, recalibration frequencies and number lagged variables. Table 4.1 gives an overview of which features are included in each model, and which names the models were given to present the results in the next chapter. The selection of exogenous variables in each multivariate model is strategically designed to facilitate comprehensive analysis, enabling examination of

- The difference in accuracy between univariate and multivariate models.
- The impact of incorporating Belgian load and generation data compared to wind and solar data.
- The potential relationship between model complexity and predictive accuracy, particularly whether the largest model yields the most precise forecasts.

Type of Model	**Features included in model**			
	Price lags only + weekday dummy	Wind & Solar forecasts [MW] + price lags +day of week	Load & Generation forecasts [MW] + price lags + day of week	All covariates + price lags + day of week
LSTM	**LSTM Base**	**LSTM W&S**	**LSTM L&G**	**LSTM Full**
LEAR	**LEAR Base**	**LEAR W&S**	**LEAR L&G**	**LEAR Full**

Table 4.1: Overview of Models used for analysis with their respective covariates

Calibration Window

The calibration window of a model is the size of data that is used to train the model. A calibration window (CW) that is too short might result in poorly trained models due to the fact that the model has too little data to capture certain trends in the price. A longer CW can improve accuracy but might lead to overfitting of the data. Three sizes of CW were used to test accuracy.

- 56 days (8 weeks)
- 728 days (2 years)
- Full history of training set (5 years)

Additionally, with longer calibration windows, there may be challenges in capturing moving averages, as recent shocks may not exert sufficient influence on subsequent predictions due to the overwhelming influence of a lengthy historical dataset.

Recalibration frequency

The recalibration frequency of a model is the frequency at which the model is retrained on new data. If there is no recalibration, it corresponds to just training the model once on the training set and then estimating the prices for the full test set. Recalibrating the model can simulate how the forecasting in real time would work. When new prices are available, it makes sense to use this data to recalibrate the model. Although it is computationally more expensive, one would expect accuracy to increase with the recalibration frequency. Four types of frequencies were tested on both the LSTM and the LEAR models: no recalibration, 14 days (bi-weekly), 7 days (weekly) and daily frequencies.

Number of lags

The number of price lags included translates to the size of the sliding window described in 4.1.1. In the default model, the inputs used in the LSTM and LEAR models are the previous 168 hours or 7 days of data. It is possible to change the size of the window to check for the optimal number of lags. The examined number of lags includes intervals of 1 day, 7 days, and 14 days of lagged variables.

Test for statistical significance

To determine whether the distinct models employed in the analysis exhibit statistically significant differences, the Diebold-Mariano test was deployed.

4.3.2 Interval forecasts

There are various ways to assess probabilistic forecasts. This section provides the evaluation methods used to verify the probabilistic forecasts used in the analysis. The two properties that were investigated are the reliability and sharpness of interval forecasts.

Reliability

To assess this reliability the Kupiec test will be used. The Kupiec test checks if the *nominal coverage* is statistically different than the *empirical coverage*. The test is described in detail in Section 2.4.3.

Sharpness

The sharpness of a probabilistic forecast refers to how well the forecasted uncertainty is concentrated around its mean. A probabilistic forecast with high sharpness will have tight prediction intervals, while low sharpness means that the intervals tend to be wider. To assess the sharpness of the interval forecasts, a simple method was used which calculates the average width of the prediction intervals at each forecasted datapoint of the test set. Equation (4.6) presents the way the sharpness score $\overline{\delta}^{(c)}$ for a given confidence level c is calculated. Here $\overline{\alpha_t}$ represents the upper bound of the prediction interval at time t and $\underline{\alpha_t}$ represents the lower bound.

$$\begin{aligned} \delta_t^{(c)} &= \overline{\alpha_t} - \underline{\alpha_t} \\ \overline{\delta}^{(c)} &= \frac{1}{T}\sum_{t=1}^{T} \delta_t^{(c)} \end{aligned} \tag{4.6}$$

The simple metric of sharpness score by average interval width was chosen because of the high interpretability that comes with such a score. More advanced methods like CRPS, described in 2.4.4, require density forecasts, while the CP model used in this thesis only generates prediction intervals.

Chapter 5

Results

The next section will give the main results and findings with respect to each of the two primary objectives of the thesis. The first part represents the results of the point forecasts. The comparison of a deep learning model, LSTM and a LASSO-estimated statistical is analysed in the first part. In the second part, the performance of a distributional neural network is analysed and compared against conformal prediction.

5.1 Point forecasts

5.1.1 Overall Performance

Figure 5.1 shows the predictions of the default models (7-day lags, full history CW and no recalibration) for the entirety of the test set for the LSTM, LEAR and naive model. The orange lines represent the predicted prices while the blue lines show the actual day-ahead prices for 2023. While no clear difference can be seen between the LSTM and the LEAR model, both models seem to have trouble predicting negative prices.The naive model is just a forward shift of the prices.

The lack of negative forecasts also becomes clear in Figure 5.2, which shows the joint plots of the LSTM (left) and LEAR model (right). These plots place the predicted prices on the y-axis and the actuals on the x-axis. The LEAR model is slightly more concentrated around the ideal diagonal, representing perfect forecasts, suggesting better performance of the LASSO estimated model.

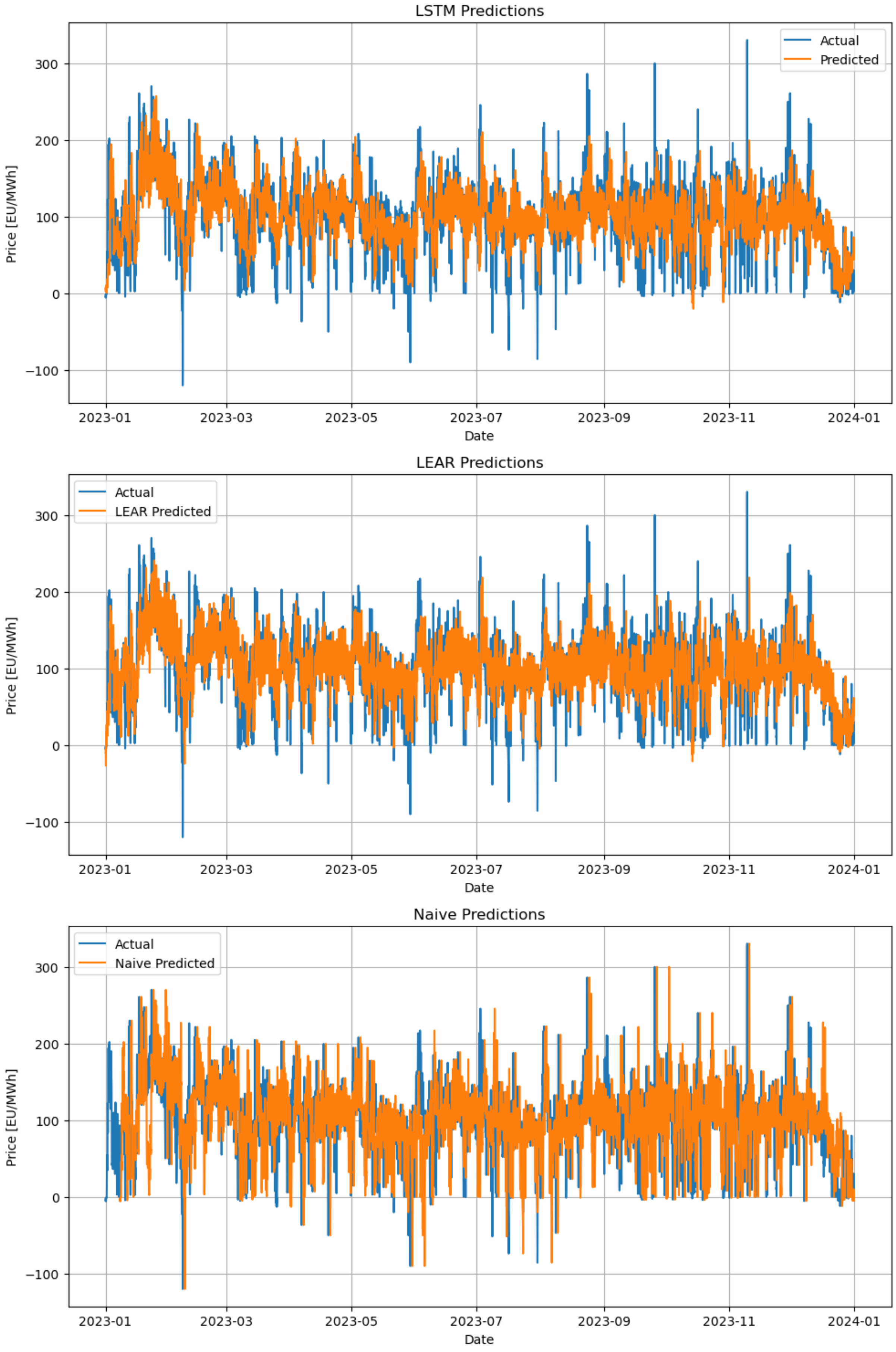


Figure 5.1: Predictions of LSTM, LEAR and Naive model on test set.

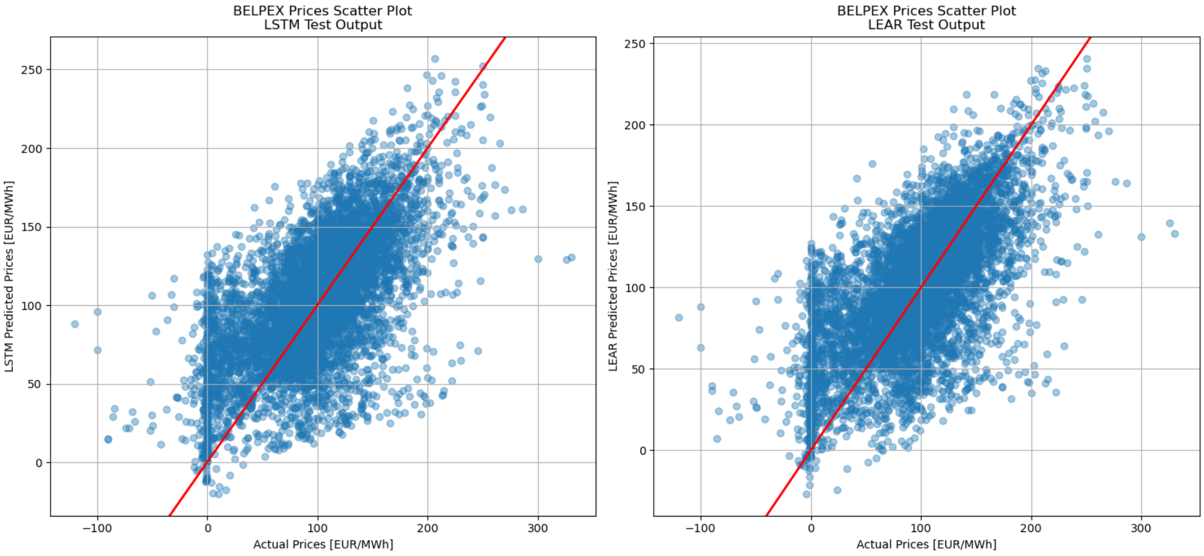


Figure 5.2: Joint plots of LSTM and LEAR models

Figure 5.3 visualises how the model's performance differs within the day. Each line on the clock represents a model, the closer the line is to the center the better its MAE is at that hour. Overall, the predicted prices during the day were less accurate than nightly prices. Further, the naive model clearly performs worse than the more advanced LSTM and LEAR methods. LEAR and LSTM make similar errors during most of the day except during morning hours the LEAR model seems to forecast more accurate prices than the deep learning model.

A scatterplot of the LSTM predictions against the LEAR predictions is shown in Figure 5.4. The x=y diagonal is present as well as the regression line of the scatter plot. The regression line is defined by $LSTM = 0.85 * LEAR + 17.37$, this indicates that on average the LEAR model predicts higher prices than the LSTM model.

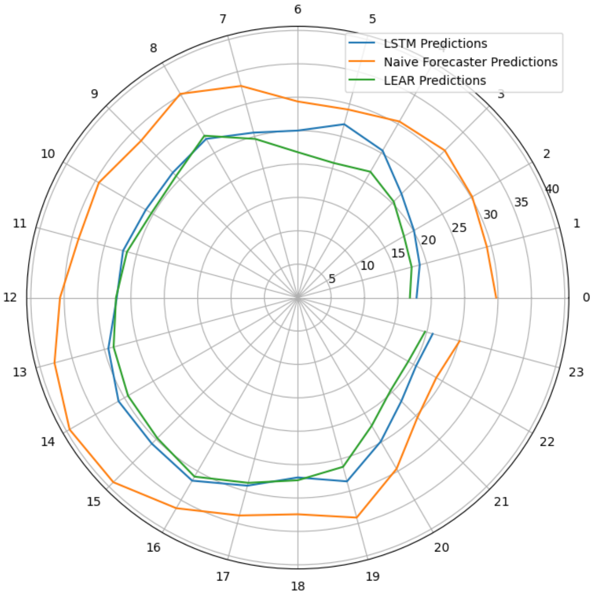


Figure 5.3: Clock graph of averaged hourly MAE values for different forecasts

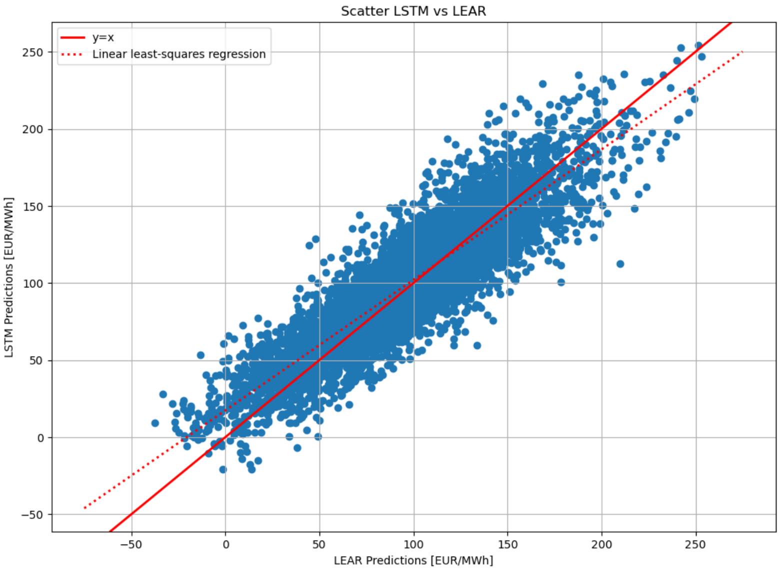


Figure 5.4: Scatter plot of LSTM and LEAR predictions

5.1.2 Influence of exogenous variables

Table 5.1 shows the performance of the default models for each of the exogenous variables added in terms of MAE (in €/MWh) and rMAE. With an exception for the full model, the LEAR model seems to outperform LSTM for each exogenous variable added. Surprisingly, the LSTM model performs best on the univariate model with only lagged prices involved, while the worst LEAR model is the full model, not even outperforming the naive benchmark. A reason for the small differences between models - especially between the Wind and Solar model, the Load and Generation model and the Full model - could be the fact that wind and solar generation data are also included in the overall generation data, resulting in multicollinearity issues. While it seems LSTM did not succeed in outperforming LEAR, no major conclusions can be drawn for this default model. The following sections will make modifications in terms of lags, calibration window and recalibration frequencies to check for possible improvements in accuracy.

		Base	**W&S**	**L&G**	**Full**
LSTM	MAE	29.27	30.19	31.16	29.85
	rMAE	0.91	0.94	0.97	0.93
LEAR	MAE	27.36	26.75	25.52	31.61
	rMAE	0.86	0.84	0.81	1.0

Table 5.1: MAE and rMAE of default models with various exogenous variables added.

5.1.3 Influence of different lags

Table 5.2 gives the MAE and rMAE values of the multivariate models in terms of the number of lagged variables included in the model, which is the same as the size of the sliding window explained in 4.1. Most models outperform the naive benchmark except two. Here again, the LSTM models do not beat the LEAR models. A LEAR model with seven days of lagged variables with the inclusion of the load and generation forecasts remains the strongest model in terms of MAE and rMAE. A larger sliding window doesn't necessarily increase the accuracy of a LEAR model while slightly improving the LSTM models in the case of the Load and Generation model.

In this table, the computation time needed to output the models is also given. There is a strong difference between the computational efficiency of the LEAR versus the LSTM model. While the LEAR model needs at most 11 seconds to compute the model, the LSTM model often takes more than a minute, especially for the models with 14 days of lagged variables.

5.1.4 Impact of calibration window

Table 5.3 shows the results when different calibration windows were used to train the model. Too small calibration windows might not capture all the relationships in the data while a CW that is too large could overfit the data. The 56-day calibration window provides very poor results with both the LSTM and LEAR models, with all of them underperforming the simple naive model.

		Base		**Wind and Solar**		**Load and Gen.**		**Full**	
		LSTM	LEAR	LSTM	LEAR	LSTM	LEAR	LSTM	LEAR
1 day	MAE	34.97	27.75	30.16	29.26	31.05	26.77	29.78	28.78
	rMAE	1.09	0.88	0.94	0.92	0.97	0.84	0.93	0.91
	Time[s]	30.74	2.02	30.61	2.6	32.5	3.07	34.76	4.34
7 day	MAE	29.27	27.36	30.19	26.75	31.16	25.52	29.85	31.61
	rMAE	0.91	0.86	0.94	0.84	0.97	0.81	0.93	1.0
	Time[s]	83.79	4.21	50.54	4.41	54.45	5.35	50.54	9.34
14 day	MAE	30.61	26.51	30.91	28.0	29.65	26.09	30.37	30.81
	rMAE	0.95	0.84	0.96	0.88	0.92	0.82	0.94	0.97
	Time[s]	78.44	4.19	99.23	6.34	76.83	6.62	81.36	11.0

Table 5.2: Results of LSTM and LEAR models based on different number of lags included.

When taking larger CW's into consideration, not all models outperform the naive models. In five cases the rMAE is higher than or equal to one, most notably in the Full LSTM model with 728-day CW. Taking a larger CW, going from 56 days to 728 days to full history is consistently better than the previous model except for the LEAR Base model and the LEAR Full model. Another thing to notice is that the LEAR Base model with 728-day CW prices performs better comparable to the full history LEAR Load and Generation model while having fewer variables and resulting in very short computation times. Overall, the LEAR models again show better results than the LSTM counterparts with a few exceptions.

In short, changing the calibration window of the models doesn't have significant positive effects on performance. Sometimes the smaller calibration windows even have negative effects on performance, especially in the case of the small 56-day windows.

		Base		**Wind and Sol.**		**Load and Gen.**		**Full**	
		LSTM	LEAR	LSTM	LEAR	LSTM	LEAR	LSTM	LEAR
56 day	MAE	73.6	30.94	74.43	45.2	79.2	40.1	76.29	41.3
	rMAE	2.29	0.97	2.31	1.41	2.46	1.26	2.37	1.29
	Time[s]	32.59	1.7	33.36	1.99	31.34	2.05	33.13	3.35
728 day	MAE	32.24	26.89	31.54	33.87	32.85	27.41	39.86	28.02
	rMAE	1.0	0.85	0.98	1.07	1.02	0.86	1.24	0.88
	Time[s]	42.66	1.91	48.18	2.48	58.77	2.46	43.23	3.79
Full history	MAE	29.27	27.36	30.19	26.75	31.16	25.52	29.85	31.61
	rMAE	0.91	0.86	0.94	0.84	0.97	0.81	0.93	1.0
	Time[s]	83.79	4.21	50.54	4.41	54.45	5.35	50.54	9.34

Table 5.3: Results of LSTM and LEAR models based on different sizes of the calibration window.

5.1.5 Impact of recalibration frequency

The last parameter that was compared is the recalibration frequency, which signifies how often the weights of the model are updated to the new data. The results of this comparison are shown in Table 5.4. Recalibrating is computationally costly because models have to be updated on a frequent basis. However, changing the recalibration frequency does result in the best models so far in this study.

Both LSTM and LEAR models significantly improve in accuracy, with a weekly recalibration yielding the best results overall. When recalibration is added, all rMAE's drop below 1 and outperform the naive model. The Full model in this comparison is consistently better than the other models. The Full LEAR model with weekly recalibration is the best one in this comparison, producing an MAE of 24.04 and an MAE of 0.75. LEAR models still do better in this comparison than LSTM. The computation times of the LSTM models take on very large values, making these models practically less feasible than the statistical LEAR method.

These results show that without recalibration the models run into problems reflected in the poor MAE. A reason for this is the fact that the time series data, shown in Figure 3.1 from Section 3, behaves very differently in the evaluated year 2023 than it did in the test set of 2018 to 2022. This means that the model is trained on data that is first very low and then becomes very volatile with a lot of price spikes. To then evaluate it on a test set with more stable prices means that it has been trained on the wrong price behaviors.

		Base		**Wind and Sol.**		**Load and Gen.**		**Full**	
		LSTM	LEAR	LSTM	LEAR	LSTM	LEAR	LSTM	LEAR
None	MAE	29.27	27.36	30.19	26.75	31.16	25.52	29.85	31.61
	rMAE	0.91	0.86	0.94	0.84	0.97	0.81	0.93	1.0
	Time[s]	83.79	4.21	50.54	4.41	54.45	5.35	50.54	9.34
1 day	MAE	25.33	25.16	26.81	24.48	26.79	24.45	28.83	24.22
	rMAE	0.79	0.78	0.83	0.76	0.83	0.76	0.9	0.75
	Time[s]	29816	2524	17699	2.108	17373	1631	46998	2889
7 day	MAE	26.26	25.08	26.14	24.43	25.82	24.29	26.1	24.04
	rMAE	0.82	0.78	0.81	0.76	0.8	0.76	0.81	0.75
	Time[s]	2206	391.07	1694	531.85	1397	421,15	2206	851.66
14 day	MAE	26.8	25.12	26.69	24.74	26.66	24.67	26.15	24.51
	rMAE	0.83	0.78	0.83	0.77	0.83	0.77	0.81	0.76
	Time[s]	1733.43	516.25	1695	781.84	1462	749.14	1904	1055

Table 5.4: Results of LSTM and LEAR models based on different recalibration frequencies

5.1.6 Test for statistical significance

Overall, the LEAR models seemed to outperform the LSTM model and these two models both did better than the naive model in terms of accuracy. To check for statistically significant differences in accuracy, a Diebold-Mariano test was performed on the naive model and the best model for LSTM and LEAR, which was the Load and Generation model with weekly recalibration in the case of LSTM, the best model for LEAR was the Full model also with weekly recalibration. Figure 5.5 shows the results in a grid. A green square in column i and row j means that the p-value of the DM test is small enough to be statistically significant. This means that the model in column i statistically performs better than the model in row j.

As was noticed in the tabular results, LEAR is statistically more accurate than LSTM and the naive model. LSTM does outperform the naive model, but it is not better than the LEAR model. This confirms the observations from the previous section, which made clear that the LEAR model outperforms the recursive multi-step LSTM model.

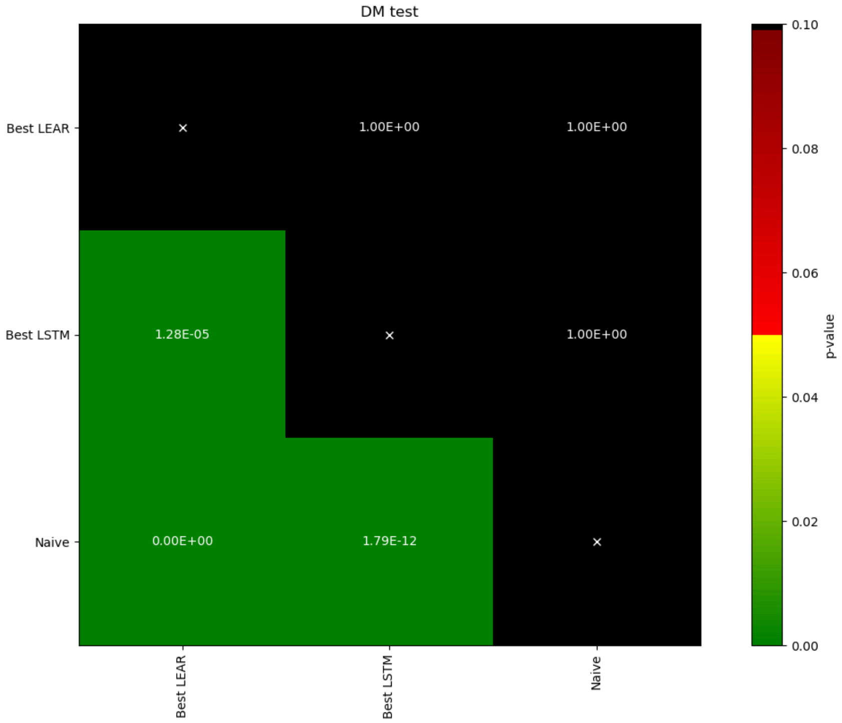


Figure 5.5: Diebold-Mariano test for LSTM versus LEAR versus Naive model.

5.2 Interval forecast

5.2.1 Plot of Conformal predictions

Figure 5.6 shows several conformal prediction intervals for the last two weeks of test data for both the best LSTM and the best LEAR model from Section 5.1.5. For LSTM, this is the Load and Generation model with weekly recalibration, for LEAR the best model was the Full model with weekly recalibration. The naive model is also included as a benchmark.

Only the first two weeks of test data are shown to make the intervals more visible. The actual data is displayed as well as the point forecasts used to construct the prediction intervals. We overlay four different levels to gauge the density of these intervals. The levels chosen are the 99, 95, 90 and 80 percent prediction intervals.

The conformal predictions for the naive model clearly have more uncertainty than the other two models: the 80% interval is as wide as the 99% intervals of the other models. The LEAR model's intervals tend to be slightly sharper than the LSTM model, which makes sense when the point forecasts of LEAR were also more accurate. This translates into smaller errors on average, and since conformal prediction is based on errors, one would expect to see sharper prediction intervals around more accurate point forecasts.

Throughout this small sample of two weeks, the density of the uncertainty varies in the LSTM and LEAR models. When prices are low the uncertainty grows, most notably in the 99% prediction intervals, while the intervals are more concentrated in higher price ranges. Some observations lie outside all the prediction intervals, if the forecast is reliable only 1% of predicted prices should fall outside of the light blue intervals. These statistics will be evaluated in Section 5.2.3.

The analysis of 90th quantile errors per day of the week and hour of the day for different forecasting models in Figure 5.7 reveals intriguing insights. LSTM intervals exhibit remarkable stability compared to other models, while LEAR model intervals demonstrate significant variability, potentially reflecting the inherent uncertainty in the power system. Furthermore, specific patterns emerge, such as lower and stable errors on Wednesday, Thursday, and Friday across all models, indicating robust performance during these days. Additionally, LSTM tends to outperform LEAR on Tuesdays, while LEAR exhibits more extreme forecasts, suggesting varying levels of conservatism in the predictions.

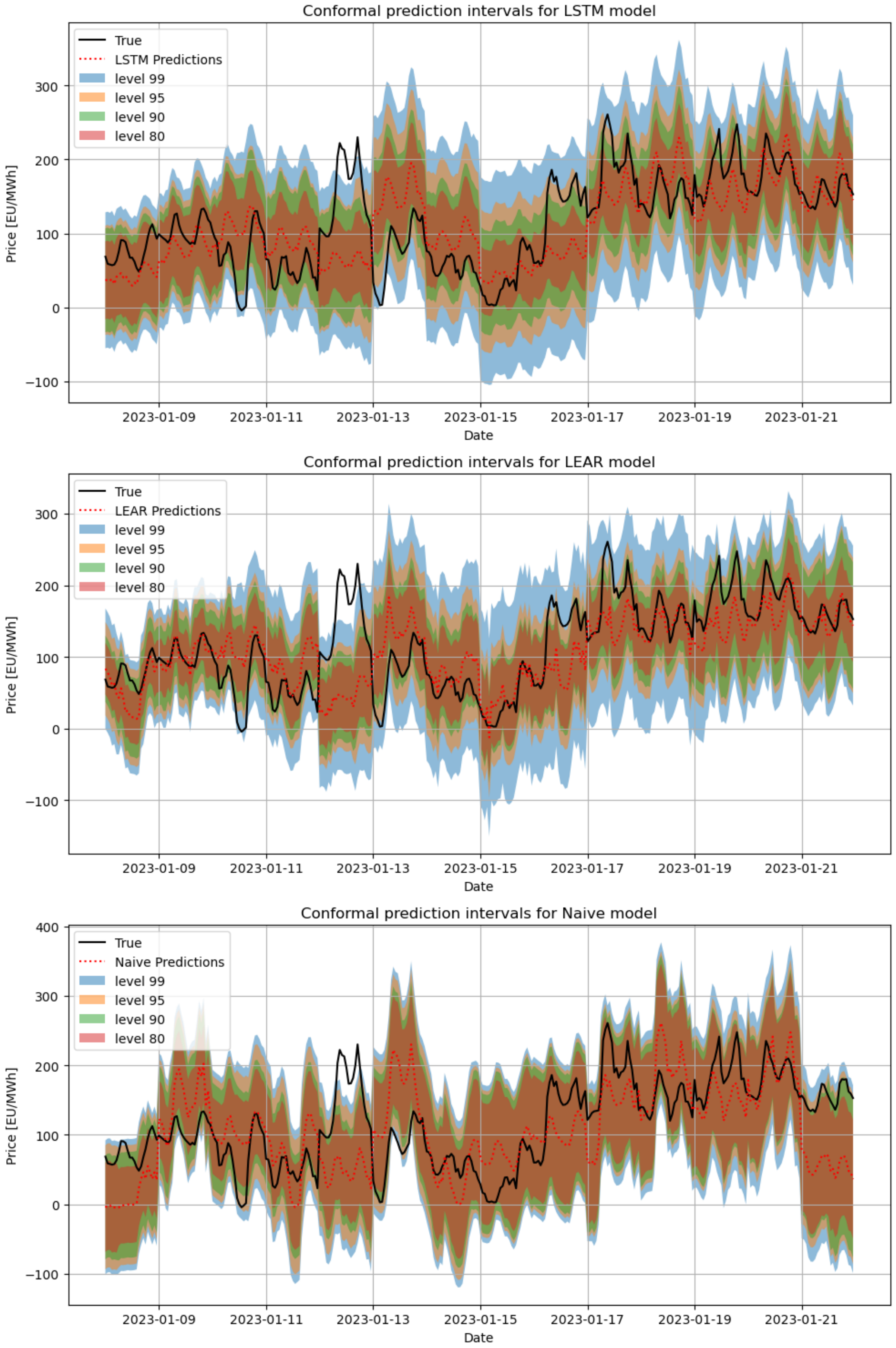


Figure 5.6: Conformal prediction intervals of LSTM, LEAR and Naive model for levels 99%, 95%, 90% and 80%.

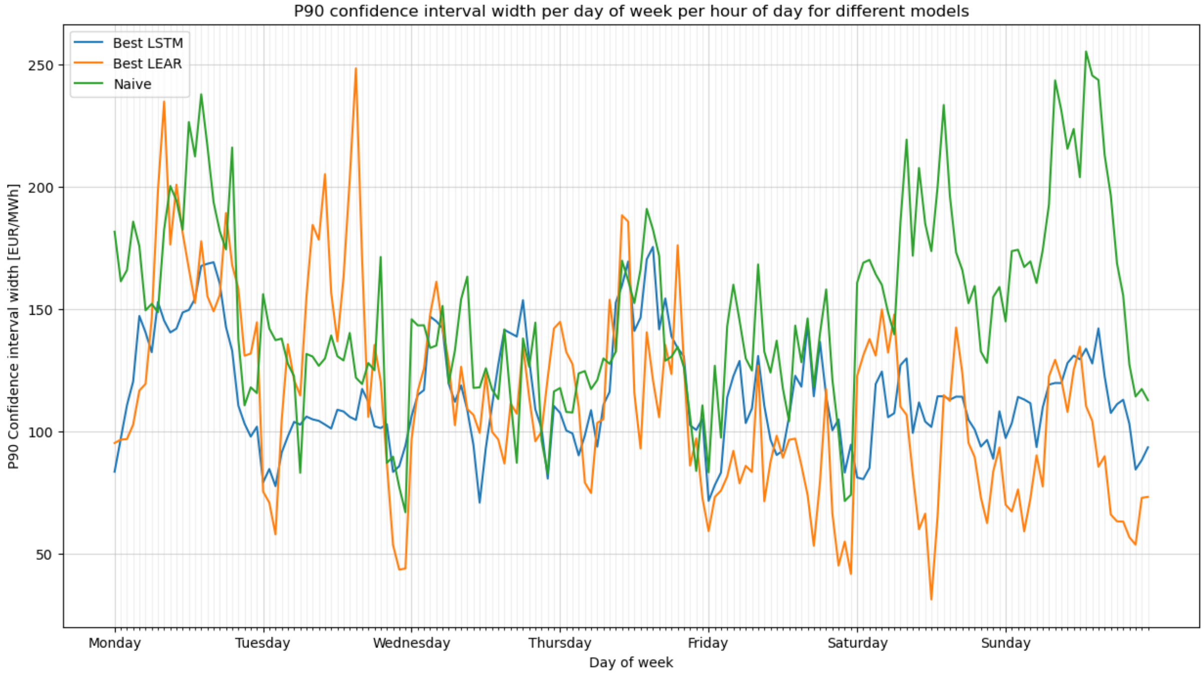


Figure 5.7: Confidence interval width (90%) per day of week per hour of day for different models.

5.2.2 Plot of DDNN

Figure 5.8 gives the prediction intervals generated by the Distributional Deep Neural Network. For the DDNN, the LSTM Load and Generation model with weekly calibration with an additional distributional layer was used (see Section 2.2.3).

The intervals shown in this plot are even wider than the naive model in most cases and seem to have a more constant width. The red dotted line in the plot refers to the mean of the predicted distribution. The mean as predicted parameter doesn't differ too much from the actual observations. The model seems to overestimate the standard deviation of the distribution of the price. All observations lie within the 99% prediction interval and very few lie outside the 95% prediction interval, while it can be expected that for two weeks of test data $(0.05 \times 168 \times 2 = 16.8)$ around 17 observations should miss the 95 percent interval.

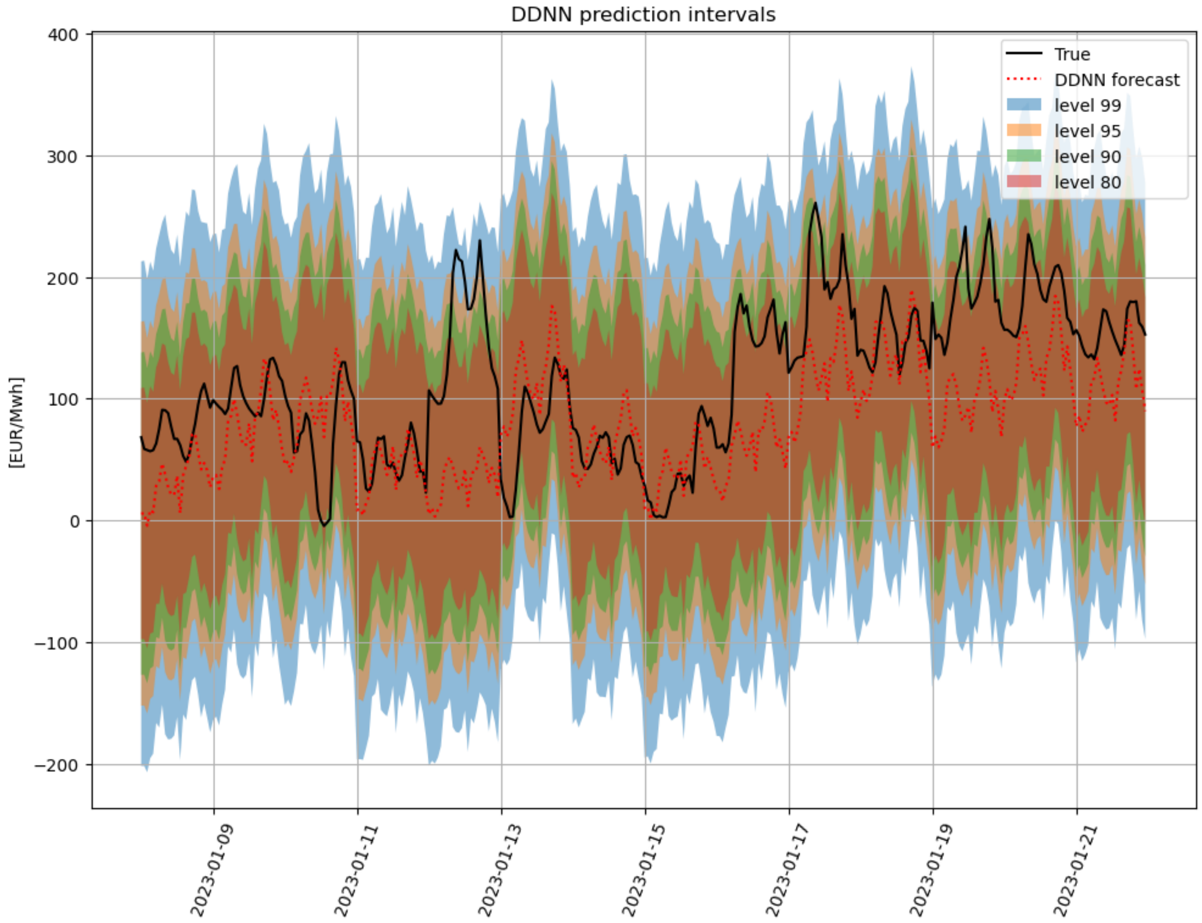


Figure 5.8: Prediction intervals made with Distributional Neural Network for levels 99%, 95%, 90% and 80%.

5.2.3 Evaluation of sharpness and reliability.

This subsection compares all probabilistic forecasts with each other in terms of sharpness and reliability. The exact methods and choice of evaluation metrics are explained in Section 4.3.2.

Sharpness

Table 5.5 shows the sharpness score of each probabilistic forecast for each level that was predicted. The DDNN forecast performs worse in terms of sharpness by a factor of around 2 when compared against the naive conformal predictions. The sharpness of LSTM and LEAR are comparable for each level, with LEAR having slightly sharper intervals than the LSTM model.

To interpret such a sharpness score, let's take as example the 90% sharpness score for the LEAR conformal prediction, which is 110.9 €/MWh. This means that on average, when the LEAR model outputs a predicted price, we can be 90% sure that the actual price is no bigger than 55.45 €/MWh or smaller than 55.45 €/MWh (110.9/2 = 55.45). It is important to note that both the conformal predictions as well as the DDNN output symmetrical intervals. These intervals are an approximation of the density of the errors. However, in reality, electricity prices aren't perfectly symmetrical. Positive electricity prices can spike up to 4000 €/MWh while the negative spikes are around -600€/MWh. This is important to take into consideration when analysing these kinds of intervals.

	DDNN	**LSTM Conformal**	**LEAR Conformal**	**Naive Conformal**
99%	411.91	200.59	194.12	183.34
95%	313.41	144.91	141.62	183.34
90%	263.04	114.96	110.89	147.97
80%	205.0	81.26	76.38	105.19

Table 5.5: Sharpness scores of the DDNN model and the conformal predictions for LSTM, LEAR and the Naive model.

Reliability

The empirical coverage of each probabilistic forecast for each confidence level is shown in Table 5.6. The conformal predictions all seem quite reliable, presenting empirical coverages close to their nominal level. The LEAR model's coverage levels are the closest to their nominal levels, but only slightly differing from the LSTM or the Naive models. The reason behind this high reliability lies in the property of conformal prediction itself. Conformal prediction doesn't assume any distribution of the errors and is not dependent on which model was used for the initial point predictions, it only takes a percentile of the errors according to the nominal level that was chosen.

However, the DDNN forecast cannot be considered reliable, as at each nominal level, the coverage is above 97%. This means that the intervals of the DDNN forecast include almost every actual price of the test set. When a Kupiec test is executed to check whether the empirical coverage statistically significantly differs from the nominal coverage, it is found that the null-hypothesis is rejected for each model on each level. Although the conformal predictions' coverage levels are close to their nominal level, these test results mean that statistically, they can't be called a reliable forecast.

	DDNN	**LSTM Conformal**	**LEAR Conformal**	**Naive Conformal**
99%	99.97	97.79	97.82	97.15
95%	99.77	93.65	93.44	92.3
90%	99.35	87.87	88.02	86.98
80%	97.21	77.43	77.78	77.5

Table 5.6: Empirical coverage of the DDNN model and the conformal predictions for LSTM, LEAR and the Naive model.

5.3 Conclusion

As final section, the main takeaways from the results are summarised below.

In terms of point forecasts, the LEAR and LSTM models both outperformed the naive model in terms of accuracy in most cases, which gives an indication that the models add value. The LEAR model also outperformed the LSTM model in almost every case, which was confirmed in the Diebold-Mariano test. When looking at the different hyperparameters, the recalibration model had the most influence on the models' accuracy. Weekly recalibration had the strongest influence, bringing forth the best models both for LSTM and for LEAR. The calibration window and number of lags included had little effect. The full history CW seemed to be the best window to use and the number of lags didn't have a clear effect.

For the probabilistic forecasts, the DDNN model severely underperformed, both in sharpness and reliability. The conformal predictions did produce reliable intervals. The LEAR model logically had the sharpest forecasts of all the models, slightly more than the LSTM model. Both the conformal LSTM and LEAR models also had smaller uncertainty than the conformal naive model.

Chapter 6

Conclusion

This final chapter contains the overall findings of the thesis. The results from Chapter 5 are used to draw conclusions about the two main objectives of the thesis and possible future work is discussed. The managerial implications of improving electricity price forecasts are also discussed in this chapter.

The LEAR model outperforms the LSTM model

The first objective was to perform a comparative analysis of a deep learning technique against statistical state-of-the-art electricity price forecasting. The models chosen were an LSTM network representing a deep learning model and the LASSO estimated LEAR model as a statistical method. Both the LSTM models and the LEAR models had similar setups. They both had the same exogenous variables added to them and worked as a recursive multi-step forecast. The models were then compared for different lags, calibration windows and recalibration frequencies. A naive similar-day model was added as a benchmark. It was noted that recalibration is necessary to improve forecasts while the other adjustments had little effect on the point forecasts' accuracy.

The LEAR model consistently produced better results in accuracy and the performance was statistically significantly better than the LSTM model. Both models outperformed the naive benchmark, but ultimately, in this specific case, LEAR would be recommended. Not only in accuracy but also in computation time LEAR was more efficient. This is a finding contradictory to many recent literature suggesting that deep learning models always outperform even the state-of-the-art statistical models. This thesis proves that LEAR can still be better in this case study.

Both models produced a high rMAE

When comparing the rMAE - a metric suitable to test across different datasets - to other studies done in recent years. A lot of studies were able to produce rMAE's below 0.5, while the models tested in this thesis reached around 0.8. This means that the models tested here had an overall worse performance than should be expected.

This high rMAE can be attributed to the fact that the data the model was trained on behaved very differently from the set it was evaluated on. It appears that 2023 is a very tricky year to evaluate models on, because of the unusually volatile years of 2022 and 2021 followed by a more stable but still relatively high price level.

Another possible reason is the fact that the data that was used all came from the Belgian electricity market. The Belgian Day-Ahead market, part of the European EPEX spot market, is highly interconnected with its neighbouring markets like France or Germany. The Belgian wind and solar generation was used as well as the Belgian generation forecasts, which overlap in a certain way. Future work could include using more market integration, by adding French or German load forecasts, prices, renewable energy data, etc.

In addition to the observed high rMAE values, it's worth noting that the 2023 test set exhibited a notably stable pattern without significant price spikes. This stability can be advantageous for a naive model that simply predicts a previous value from the same day or week, as the previous value is likely to be very close to the new prediction. Consequently, the stability of the 2023 test set may have contributed to the relatively high performance of the naive model compared to more sophisticated forecasting approaches.

Conformal prediction is more reliable and sharper than the DDNN

The second objective was to quantify the uncertainty of price forecasts. Conformal prediction, a relatively simple statistical method, was compared against the sophisticated Distributional Deep Neural Network, a deep-learning method predicting entire distributions of prices. Prediction intervals for several confidence levels were constructed and the sharpness and reliability of these probabilistic forecasts were evaluated. Conformal prediction proved to have the most reliable forecasts, while the DDNN produced dissatisfactory results. The prediction intervals of DDNN included almost all observations for each level predicted, which carries little to no information when presenting a probabilistic forecast.

The poor performance of the DDNN can be attributed to the fact it was actually an extension of the LSTM model used for point forecasts, the only difference being the addition of a distributional layer. And since the LSTM model also performed below expectation, the DDNN model might have underperformed as well.

Managerial implications

Improved electricity price forecasts hold significant managerial implications, particularly in enhancing revenue generation through more informed bidding strategies and operational decisions. When using more precise day-ahead price predictions, electricity suppliers can strategically adjust their generation and procurement plans, minimizing costs associated with imbalance penalties and maximizing profits. Not only generation companies but in essence every market player on the EPEX spot market can benefit from an improved forecast model.

While not researched in this thesis, future work could explore the implications of adding uncertainty forecasts to certain bidding strategies and the resulting profit connected to it. Bidding on the day-ahead comes down to a price-based unit commitment solvable with mixed linear programming. Adding uncertainty creates an extra stochastic dimension that could heavily influence the optimal bidding strategy.

Individual contribution to the thesis and division of work

Abel: Led the coding efforts and implementation of various research questions. This involved developing algorithms, conducting experiments, and analyzing results. Additionally, Abel played a key role in the technical aspects of the project, ensuring the proper functioning of models and methodologies.

Vic: Primarily focused on researching electricity price forecasting and contributed significantly to the writing of the thesis. This included conducting literature reviews, synthesizing research findings, and drafting thesis chapters. Vic also played a crucial role in shaping the conceptual framework of the study and ensuring clarity and coherence in the presentation of ideas.

Bibliography

[1] Mousa Afrasiabi et al. “Multi-agent microgrid energy management based on deep learning forecaster”. In: *Energy* 186 (Nov. 1, 2019), p. 115873. ISSN: 0360-5442. DOI: 10.1016/j.energy.2019.115873. URL: https://www.sciencedirect.com/science/article/pii/S0360544219315452 (visited on 04/17/2024).

[2] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. *Conditional Time Series Forecasting with Convolutional Neural Networks*. Sept. 17, 2018. DOI: 10.48550/arXiv.1703.04691. arXiv: 1703.04691[stat]. URL: http://arxiv.org/abs/1703.04691 (visited on 04/26/2024).

[3] Jason Brownlee. *4 Strategies for Multi-Step Time Series Forecasting*. MachineLearningMastery.com. Mar. 7, 2017. URL: https://machinelearningmastery.com/multi-step-time-series-forecasting/ (visited on 04/09/2024).

[4] Zihan Chang, Yang Zhang, and Wenbo Chen. “Effective Adam-Optimized LSTM Neural Network for Electricity Price Forecasting”. In: *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*. 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS). ISSN: 2327-0594. Nov. 2018, pp. 245–248. DOI: 10.1109/ICSESS.2018.8663710. URL: https://ieeexplore.ieee.org/document/8663710 (visited on 04/17/2024).

[5] Zihan Chang, Yang Zhang, and Wenbo Chen. “Electricity price prediction based on hybrid model of adam optimized LSTM neural network and wavelet transform”. In: *Energy* 187 (Nov. 15, 2019), p. 115804. ISSN: 0360-5442. DOI: 10.1016/j.energy.2019.07.134. URL: https://www.sciencedirect.com/science/article/pii/S0360544219314768 (visited on 04/17/2024).

[6] Yiyuan Chen et al. “BRIM: An Accurate Electricity Spot Price Prediction Scheme-Based Bidirectional Recurrent Neural Network and Integrated Market”. In: *Energies* 12.12 (Jan. 2019). Number: 12 Publisher: Multidisciplinary Digital Publishing Institute, p. 2241. ISSN: 1996-1073. DOI: 10.3390/en12122241. URL: https://www.mdpi.com/1996-1073/12/12/2241 (visited on 04/17/2024).

[7] Jen-Tzung Chien. “Chapter 2 - Model-Based Source Separation”. In: *Source Separation and Machine Learning*. Ed. by Jen-Tzung Chien. Academic Press, Jan. 1, 2019, pp. 21–52. ISBN: 978-0-12-817796-9. DOI: 10.1016/B978-0-12-804566-4.00013-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128045664000139 (visited on 04/15/2024).

[8] Radhakrishnan Angamuthu Chinnathambi et al. “Deep Neural Networks (DNN) for Day-Ahead Electricity Price Markets”. In: *2018 IEEE Electrical Power and Energy Conference (EPEC)*. 2018 IEEE Electrical Power and Energy Conference (EPEC). ISSN: 2381-2842. Oct. 2018, pp. 1–6. DOI: 10.1109/EPEC.2018.8598327. URL: https://ieeexplore.ieee.org/abstract/document/8598327 (visited on 04/15/2024).

[9] Jilles De Blauwe. “Day-ahead Belgian electricity price forecasting, while accounting for volatility, extreme spikes and downstream utility”. PhD thesis.

[10] Francis X Diebold and Robert S Mariano. “Comparing Predictive Accuracy”. In: *Journal of Business & Economic Statistics* 20.1 (Jan. 1, 2002). Publisher: Taylor & Francis _eprint: https://doi.org/10.1198/073500102753410444, pp. 134–144. ISSN: 0735-0015. DOI: 10.1198/073500102753410444. URL: https://doi.org/10.1198/073500102753410444 (visited on 04/29/2024).

[11] *ENTSO-E Transparency Platform*. URL: https://transparency.entsoe.eu/ (visited on 04/30/2024).

[12] Itamar Faran. *CRPS — A Scoring Function for Bayesian Machine Learning Models*. Medium. Jan. 28, 2023. URL: https://towardsdatascience.com/crps-a-scoring-function-for-bayesian-machine-learning-models-dd55a7a337a8 (visited on 04/25/2024).

[13] Alex Gammerman, Volodya Vovk, and Vladimir Vapnik. *Learning by Transduction*. Jan. 30, 2013. DOI: 10.48550/arXiv.1301.7375. arXiv: 1301.7375[cs, stat]. URL: http://arxiv.org/abs/1301.7375 (visited on 04/18/2024).

[14] Raffaella Giacomini and Halbert White. “Tests of Conditional Predictive Ability”. In: *Econometrica* 74.6 (2006). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1468-0262.2006.00718.x, pp. 1545–1578. ISSN: 1468-0262. DOI: 10.1111/j.1468-0262.2006.00718.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0262.2006.00718.x (visited on 04/29/2024).

[15] Tilmann Gneiting and Adrian E Raftery. “Strictly Proper Scoring Rules, Prediction, and Estimation”. In: *Journal of the American Statistical Association* 102.477 (Mar. 2007), pp. 359–378. ISSN: 0162-1459, 1537-274X. DOI: 10.1198/016214506000001437. URL: http://www.tandfonline.com/doi/abs/10.1198/016214506000001437 (visited on 04/25/2024).

[16] Prospero Events Group. *Exploring the Bright Future of Electricity Price Forecasting Technology*. URL: https://www.linkedin.com/pulse/exploring-bright-future-electricity-price-forecasting-technology-sptrc (visited on 05/05/2024).

[17] Hans Hersbach. “Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems”. In: *Weather and Forecasting* 15.5 (Oct. 1, 2000). Publisher: American Meteorological Society Section: Weather and Forecasting, pp. 559–570. ISSN: 1520-0434, 0882-8156. DOI: 10.1175/1520-0434(2000)015<0559:DOTCRP>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/wefo/15/5/1520-0434_2000_015_0559_dotcrp_2_0_co_2.xml (visited on 04/23/2024).

[18] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (Nov. 1997). Conference Name: Neural Computation, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://ieeexplore.ieee.org/abstract/document/6795963 (visited on 04/16/2024).

[19] Rob J. Hyndman and Anne B. Koehler. “Another look at measures of forecast accuracy”. In: *International Journal of Forecasting* 22.4 (Oct. 1, 2006), pp. 679–688. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2006.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000239 (visited on 04/23/2024).

[20] *Hyperparameter Optimization*. Nixtla. URL: https://nixtlaverse.nixtla.io/neuralforecast/examples/automatic_hyperparameter_tuning.html (visited on 05/03/2024).

[21] Arkadiusz Jedrzejewski et al. “Electricity Price Forecasting: The Dawn of Machine Learning”. In: *IEEE Power and Energy Magazine* 20.3 (May 2022). Conference Name: IEEE Power and Energy Magazine, pp. 24–31. ISSN: 1558-4216. DOI: 10.1109/MPE.2022.3150809. URL: https://ieeexplore.ieee.org/document/9761111/citations#citations (visited on 01/04/2024).

[22] He Jiang, Yao Dong, and Jianzhou Wang. “Electricity price forecasting using quantile regression averaging with nonconvex regularization”. In: *Journal of Forecasting* n/a (n/a). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.3103. ISSN: 1099-131X. DOI: 10.1002/for.3103. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3103 (visited on 04/18/2024).

[23] LianLian Jiang and Guoqiang Hu. “Day-Ahead Price Forecasting for Electricity Market using Long-Short Term Memory Recurrent Neural Network”. In: *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV). Nov. 2018, pp. 949–954. DOI: 10.1109/ICARCV.2018.8581235. URL: https://ieeexplore.ieee.org/document/8581235 (visited on 04/17/2024).

[24] Gaurav Kapoor and Nuttanan Wichitaksorn. “Electricity price forecasting in New Zealand: A comparative analysis of statistical and machine learning models with feature selection”. In: *Applied Energy* 347 (Oct. 1, 2023), p. 121446. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2023.121446. URL: https://www.sciencedirect.com/science/article/pii/S0306261923008103 (visited on 04/17/2024).

[25] Christopher Kath and Florian Ziel. “Conformal prediction interval estimation and applications to day-ahead and intraday power markets”. In: *International Journal of Forecasting* 37.2 (Apr. 1, 2021), pp. 777–799. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2020.09.006. URL: https://www.sciencedirect.com/science/article/pii/S0169207020301473 (visited on 04/18/2024).

[26] Ping-Huan Kuo and Chiou-Jye Huang. “An Electricity Price Forecasting Model by Hybrid Structured Deep Neural Networks”. In: *Sustainability* 10.4 (Apr. 2018). Number: 4 Publisher: Multidisciplinary Digital Publishing Institute, p. 1280. ISSN: 2071-1050. DOI: 10.3390/su10041280. URL: https://www.mdpi.com/2071-1050/10/4/1280 (visited on 04/17/2024).

[27] Paul H. Kupiec. "Techniques for Verifying the Accuracy of Risk Measurement Models". In: *The Journal of Derivatives* 3.2 (Nov. 30, 1995), pp. 73–84. ISSN: 1074-1240, 2168-8524. DOI: 10.3905/jod.1995.407942. URL: http://pm-research.com/lookup/doi/10.3905/jod.1995.407942 (visited on 04/23/2024).

[28] Jesus Lago. *EPF Toolbox*. 2020. URL: https://epftoolbox.readthedocs.io/en/latest/modules/models.html.

[29] Jesus Lago, Fjo De Ridder, and Bart De Schutter. "Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms". In: *Applied Energy* 221 (July 1, 2018), pp. 386–405. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2018.02.069. URL: https://www.sciencedirect.com/science/article/pii/S030626191830196X (visited on 01/05/2024).

[30] Jesus Lago et al. "Forecasting day-ahead electricity prices in Europe: The importance of considering market integration". In: *Applied Energy* 211 (Feb. 1, 2018), pp. 890–903. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2017.11.098. URL: https://www.sciencedirect.com/science/article/pii/S0306261917316999 (visited on 04/16/2024).

[31] Jesus Lago et al. "Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark". In: *Applied Energy* 293 (July 1, 2021), p. 116983. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2021.116983. URL: https://www.sciencedirect.com/science/article/pii/S0306261921004529 (visited on 01/04/2024).

[32] Remy Lau. *Cross-Entropy, Negative Log-Likelihood, and All That Jazz*. Medium. Mar. 10, 2022. URL: https://towardsdatascience.com/cross-entropy-negative-log-likelihood-and-all-that-jazz-47a95bd2e81 (visited on 05/09/2024).

[33] Shuman Luo and Yang Weng. "A two-stage supervised learning approach for electricity price forecasting by leveraging different data sources". In: *Applied Energy* 242 (May 15, 2019), pp. 1497–1512. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2019.03.129. URL: https://www.sciencedirect.com/science/article/pii/S0306261919305380 (visited on 04/16/2024).

[34] Katarzyna Maciejowska, Jakub Nowotarski, and Rafał Weron. "Probabilistic forecasting of electricity spot prices using Factor Quantile Regression Averaging". In: *International Journal of Forecasting* 32.3 (July 1, 2016), pp. 957–965. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2014.12.004. URL: https://www.sciencedirect.com/science/article/pii/S0169207014001848 (visited on 04/18/2024).

[35] Grzegorz Marcjasz et al. "Distributional neural networks for electricity price forecasting". In: *Energy Economics* 125 (Sept. 1, 2023), p. 106843. ISSN: 0140-9883. DOI: 10.1016/j.eneco.2023.106843. URL: https://www.sciencedirect.com/science/article/pii/S0140988323003419 (visited on 04/18/2024).

[36] Shiro Matsumoto. *Understand the capabilities of cyclic encoding*. Medium. Jan. 18, 2024. URL: https://shrmtmt.medium.com/understand-the-capabilities-of-cyclic-encoding-5b68f831387e (visited on 04/30/2024).

[37] Jakub Nowotarski and Rafal Weron. “Merging quantile regression with forecast averaging to obtain more accurate interval forecasts of Nord Pool spot prices”. In: *11th International Conference on the European Energy Market (EEM14)*. 2014 11th International Conference on the European Energy Market (EEM). Krakow, Poland: IEEE, May 2014, pp. 1–5. ISBN: 978-1-4799-6095-8. DOI: 10.1109/EEM.2014.6861285. URL: http://ieeexplore.ieee.org/document/6861285/ (visited on 04/18/2024).

[38] Jakub Nowotarski and Rafał Weron. “Computing electricity spot price prediction intervals using quantile regression and forecast averaging”. In: *Computational Statistics* 30.3 (Sept. 1, 2015), pp. 791–803. ISSN: 1613-9658. DOI: 10.1007/s00180-014-0523-0. URL: https://doi.org/10.1007/s00180-014-0523-0 (visited on 04/17/2024).

[39] Jakub Nowotarski and Rafał Weron. “Recent advances in electricity price forecasting: A review of probabilistic forecasting”. In: *Renewable and Sustainable Energy Reviews* 81 (Jan. 1, 2018), pp. 1548–1568. ISSN: 1364-0321. DOI: 10.1016/j.rser.2017.05.234. URL: https://www.sciencedirect.com/science/article/pii/S1364032117308808 (visited on 04/12/2024).

[40] Yvet Renkema, Nico Brinkel, and Tarek Alskaif. *Conformal Prediction for Stochastic Decision-Making of PV Power in Electricity Markets*. Mar. 29, 2024. arXiv: 2403.20149[cs,eess,stat]. URL: http://arxiv.org/abs/2403.20149 (visited on 04/09/2024).

[41] EPEX SPOT. *About us — EPEX SPOT*. URL: https://www.epexspot.com/en/about (visited on 04/25/2024).

[42] EPEX SPOT. *Basics of the Power Market — EPEX SPOT*. URL: https://www.epexspot.com/en/basicspowermarket (visited on 04/25/2024).

[43] EPEX SPOT. *Exchange Members — EPEX SPOT*. URL: https://www.epexspot.com/en/exchangemembers (visited on 04/25/2024).

[44] EPEX SPOT. *Market Data — EPEX SPOT*. URL: https://www.epexspot.com/en/market-data?market_area=AT&trading_date=2024-04-25&delivery_date=2024-04-26&underlying_year=&modality=Auction&sub_modality=DayAhead&technology=&product=60&data_mode=aggregated&period=&production_period= (visited on 04/25/2024).

[45] *Tune Search Algorithms (tune.search) — Ray 2.20.0*. URL: https://docs.ray.io/en/latest/tune/api/suggestion.html (visited on 05/03/2024).

[46] Bartosz Uniejewski, Jakub Nowotarski, and Rafał Weron. “Automated Variable Selection and Shrinkage for Day-Ahead Electricity Price Forecasting”. In: *Energies* 9.8 (Aug. 2016). Number: 8 Publisher: Multidisciplinary Digital Publishing Institute, p. 621. ISSN: 1996-1073. DOI: 10.3390/en9080621. URL: https://www.mdpi.com/1996-1073/9/8/621 (visited on 04/22/2024).

[47] Bartosz Uniejewski and Rafał Weron. “Regularized quantile regression averaging for probabilistic electricity price forecasting”. In: *Energy Economics* 95 (Mar. 1, 2021), p. 105121. ISSN: 0140-9883. DOI: 10.1016/j.eneco.2021.105121. URL: https://www.sciencedirect.com/science/article/pii/S0140988321000268 (visited on 04/18/2024).

[48] Xiaolong Xie, Wei Xu, and Hongzhi Tan. “The Day-Ahead Electricity Price Forecasting Based on Stacked CNN and LSTM”. In: *Intelligence Science and Big Data Engineering*. Ed. by Yuxin Peng et al. Cham: Springer International Publishing, 2018, pp. 216–230. ISBN: 978-3-030-02698-1. DOI: 10.1007/978-3-030-02698-1_19.

[49] Maheen Zahid et al. “Electricity Price and Load Forecasting using Enhanced Convolutional Neural Network and Enhanced Support Vector Regression in Smart Grids”. In: *Electronics* 8.2 (Feb. 2019). Number: 2 Publisher: Multidisciplinary Digital Publishing Institute, p. 122. ISSN: 2079-9292. DOI: 10.3390/electronics8020122. URL: https://www.mdpi.com/2079-9292/8/2/122 (visited on 04/17/2024).

FACULTY OF BUSINESS AND ECONOMICS
Naamsestraat 69 bus 3500
3000 LEUVEN, België
feb.leuven@kuleuven.be
www.feb.kuleuven.be