Forecasting the Regulating Price in the Finnish Energy Market using the Multi-Horizon Quantile Recurrent Neural Network

MSc Thesis in Mathematical Statistics, Lund University

Thomas Hamfelt

June 3, 2020

Abstract

In recent years there has been a large increase in available data from the electric grid in Finland. The availability of both operational as well as financial data enables exploration of forecasting energy prices using deep learning techniques. As a result this thesis implements the Multi-Horizon Quantile Recurrent Neural Network (MQRNN) to forecast the regulating price in the Finnish energy market. The forecast is a rolling window three to eight hours into the future and contains several quantiles. The results suggest that while the central location of the distribution does not change much from the spot price the tails can be long, especially the right tail. Since the model is able to capture changes in the distribution there is indication that the market contains some structure. Finally, after dicussing the results and drawing conclusions some suggestions for future improvements are presented.

Acknowledgements

I want to express my gratitude to Axpo's Malmö, Sweden office and especially the Physical desk. Without their warm welcome and knowledge sharing this thesis would not have been possible. Also, I want to thank my supervisor Alexandros Sopasakis for his support and encouragement.

Contents

Chapter 1

Introduction

Energy is traded like a commodity in the Nordics where supply and demand determines the price. Due to energy's crucial function in society and intangible feature the set up is different in comparison to other commodity markets. In recent time the need for innovation in the energy market has increased in the mission to combat climate change. Today a growing part of power production comes from renewable energy sources such as wind and solar power. But forecasting the production from renewables is challenging and large deviations can occur which result in market volatility. As a result operators of the power grid, in their effort to provide a well functioning market, are making their data more available. Finland is at the forefront of such an initiative and extensive amount of power grid data is provided in real time. With plenty of data available the possibility to use deep learning techniques for forecasting is enabled. While plenty of research has focused on the energy load and spot price, this thesis investigates the regulating price in Finland.

In its aim to forecast the regulating price this thesis implements the Multi-Horizon Quantile Recurrent Neural Network which was introduced in 2017 by Wu et al. [17]. It models the conditional quantiles several time steps into the future and has been used successfully on sales data from Amazon as well as in the energy forecasting competition GEFCom 2014. Forecasting quantiles can be more informative than point forecasting. It captures the shape, scale and central location of the distribution. When the data is skewed the expected value can be less informative than the median. Quantile regression is also less sensitive to outliers [12]. These are desirable traits when forecasting the regulating price.

The topic of the thesis is of value to companies which trade energy since there is plenty of uncertainty associated with the regulating price. The results from modelling the distribution can be used for risk management and trading. Risk measures, such as Value-at-Risk (VaR), are easily computed once the quantiles are obtained.

In the second chapter the dynamics of the energy market in Finland (and the Nordics) are introduced. First some characteristics of energy, as a commodity, are presented. Secondly the three different markets, *Spot, Intraday* and *Regulating*, on which energy is traded are explained. The occurance of up- and down-regulation is also motivated.

The third chapter covers the history and theory of quantile regression. It is showed that by solving a optimization problem the desired quantile can be obtained. On these observations, Roger Koenker and Gilbert Bassett in 1978 introduced the *conditional quantile* function [11]. It enabled quantiles to be modelled as functions of input variables. Later it was explored how nerual networks could be used to approximate the conditional quantile function. This sets the stage for the Multi-Horizon Quantile Recurrent Neural Network which is covered in detail at the end of the chapter.

The fourth chapter covers how the model was implemented. The choice of neural networks and hyperparameters are motivated.

In the fifth chapter the results are presented. The model implemented is visualized for 10 consecutive hours of the validation data. To test the model on a greater scale histograms are also presented of the forecasting errors for 1000 hours of the validation data.

In the final chapter the results are discussed and conclusions put forward. Trade-offs made during the course of the thesis are clarified and the direction of future work is recommended.

The thesis follows the convention where bold-face, lower-case letters refer to vectors and bold-face, capital-case letters refer to matrices. Exceptions are made when presenting the Multi-Horizon Quantile Recurrent Neural Network to make the notation less heavy and what is implied should be evident from context.

Chapter 2

Dynamics of the Power Market

In the early 1990's the the market for trading of electrical energy started to become deregulated in the Nordic countries. Today it is liberalized and energy is traded like a commodity on the power exchange Nord Pool. But due to its importance to society and unique features the setup of the energy market is different in comparison to other commodity markets. Two important features are:

- **Production must equal consumption in the power grid at all times**. The Transmission Grid Operator (TSO) is an independent operator of the power grid and aims to have a frequency of 50Hz in the power grid at all times. In many cases there is one and in Sweden it is Svenska Kraftnät while in Finland it is Fingrid. In the Nordics there is extensive cooperation between the TSOs.

- **Non-storagable**. Batteries do not have the capacity to store great amounts of energy. Thus there is little room for abitrage possibilities. One cannot produce while the price is low, store it, and sell it when the price is high.

Due to this, the power market is split into three sub-markets which are interconnected. These are the *Spot market*, *Intraday market* and *Regulating market*. In Figure 2.1 their relationships are illustrated for two arbitrary days.

The spot market takes place every day at 12.00 (noon). Up until then trading companies (TCs) can submit their production and consumption plans along with prices to Nord Pool. The submissions apply to the hours of the

coming day. The market price is determined by supply and demand in a double-acution. The Finnish market consists of one price area but this is not always the case. Sweden, for example, consists of four price areas.

Two hours after the spot market has taken place the intraday market opens for the hours of the coming day. This is a liquid market where TCs can trade with each other. Any deviations from the original plans submitted in the spot market can be neutralized. Approximately one hour before the operational hour the intraday market for that particular hour closes.

As the operational hour comes into effect, if the TC produces or consumes more than initially reported in the spot market (including any intraday trading), it will be settled in the regulating market. But the price in the regulating market can vary from the spot price. If production does not equal consumption the frequency in the grid will deviate from 50Hz. In such cases the TSO must take action.

- If consumption $>$ production the frequency falls and the TSO must *up-regulate*. Since demand is greater than supply it generates a higher price than the spot price.
- If consumption = production the frequency is 50Hz and the regulating price equals the spot price.
- If consumption $<$ production the frequency increases and the TSO must *down-regulate*. Since demand is smaller than supply it generates a lower price than the spot price.

Along with the expansion of renewable energy sources it has become more challenging to properly forecast energy production. In Finland on a windy day around 16% of all energy production comes from wind power. On a calm day, wind power constitutes $0,2\%$ [6]. This has resulted in Finland exhibiting a very volatile regulating market. In Figure 2.2 the spot price and regulating price is illustrated for some days in January 2016. The top figure shows the prices. The bottom figure clarifies whether the market was regulated or not. It is not uncommon that the regulating price can be several magnitudes above or below the spot price.

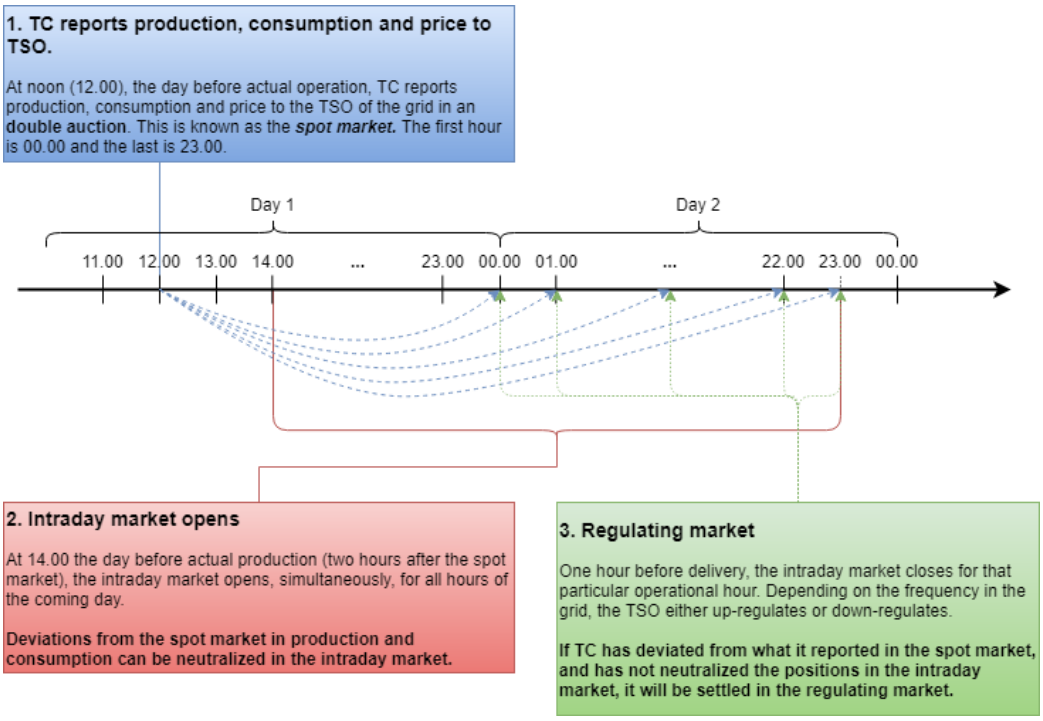

Figure 2.1: The energy market in the Nordics illustrated and explained for two arbitrary days. The first one is the spot market which takes place the day before the operational hours. The second one is the intraday market. For a particular hour it closes approximately one hour before it comes into effect. The third market is the regulating market. Depending on the frequency in the grid the Transmission System Operator (TSO) either up-regulates on down-regulates. Any deviations from the spot market which are not neutralized in the intraday market are settled in the regulating market.

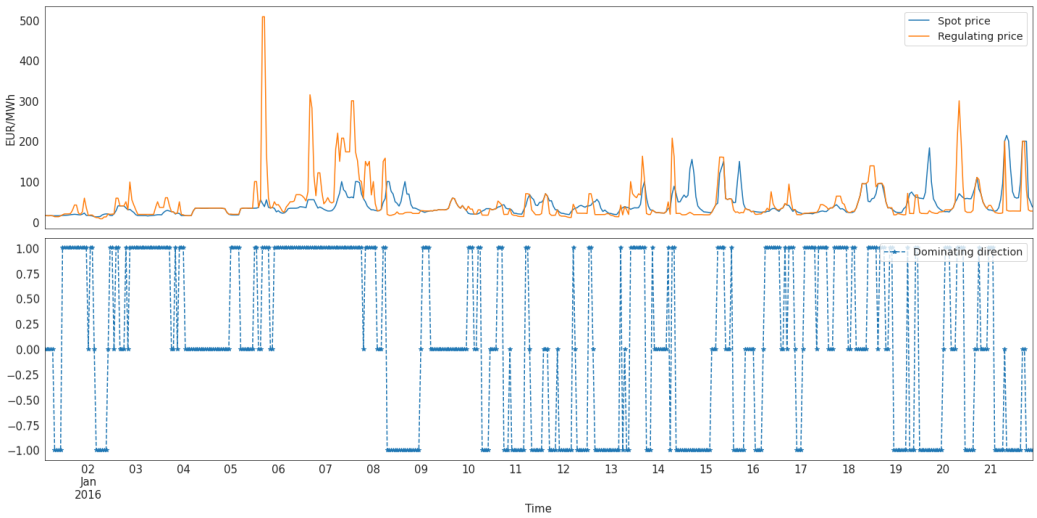

Figure 2.2: In the top figure, the spot price as well as regulating price for some days in January 2016 has been plotted. The regulating price can be very volatile and be many times above or below the spot price. The bottom figure shows whether the market was up-regulated (1), not regulated (0) or down-regulated (−1).

Chapter 3

Theory

3.1 Quantile Regression

Let us denote by F_X the distribution function for some stochastic process X. The inverse to the distribution function is denoted by F_X^{-1} and is often referred to as the *quantile function*. It can be shown, by letting $F_X^{-1}(U) = x$ where U is $U(0,1)$, that F_X is strictly increasing and continuous,

$$\mathbb{P}(X \leq x) = \mathbb{P}(F_X^{-1}(U) \leq x) = \mathbb{P}(U \leq F_X(x)) = F_X(x).$$

In the last equality we use the fact that $\mathbb{P}(U \leq u) = u$. In the case when F_X is discontinuous and/or monotonically increasing, a more robust approach to constructing the quantile is required. This paves the way for the definition of the quantile function.

Definition 3.1.1. Quantile function. For any stochastic variable X with distribution function $F_X(x) = \mathbb{P}(X \leq x)$ we define the inverse distribution function,

$$F_X^{-1}(\tau) = \inf(x \in \mathbb{R} : F_X(x) \geq \tau), \text{ where } \tau \in (0,1).$$

With the quantile function defined, we proceed to quantile regression. This presentation is based on the introduction to the area given in [10] and [12]. We begin by casting the computation of quantiles as an optimization problem. Consider the cost function ρ_τ where $\tau \in (0,1)$ and $\mathbb{1}$ denotes the indicator function,

$$\rho_\tau(u) = u(\tau - \mathbb{1}_{u<0}). \tag{3.1}$$

For a stochastic variable X, we are interested in making a point estimate $\hat{x}$ such that we minimize the cost function ρ_τ. We proceed in the usual manner when computing the expected value,

$$\mathbb{E}\left[\rho_\tau(X-\hat{x})\right] = \int_{\mathbb{R}} \rho_\tau(y-\hat{x}) f_X(y)dy \tag{3.2}$$

$$= (\tau-1)\int_{-\infty}^{\hat{x}} (y-\hat{x}) f_X(y)dy + \tau\int_{\hat{x}}^{\infty} (y-\hat{x}) f_X(y)dy. \tag{3.3}$$

Next we differentiate with respect to $\hat{x}$. The first integral in (3.3) becomes, by the fundamental theorem of calculus and differentiating through the integral,

$$\begin{aligned}(\tau-1)\frac{\partial}{\partial\hat{x}}\int_{-\infty}^{\hat{x}} (y-\hat{x}) f_X(y)dy &= (\tau-1)(y-\hat{x}) f_X(y)\big|_{y=\hat{x}} \\ &\quad + (\tau-1)\int_{-\infty}^{\hat{x}} \frac{\partial}{\partial\hat{x}}(y-\hat{x}) f_X(y)dy \\ &= 0 - (\tau-1)\int_{-\infty}^{\hat{x}} f_X(y)dx \\ &= (1-\tau)F_X(\hat{x}).\end{aligned}$$

And by the same line of argument the second integral in (3.3) becomes,

$$\begin{aligned}\tau\frac{\partial}{\partial\hat{x}}\int_{\hat{x}}^{\infty} (y-\hat{x}) f_X(y)dy &= -\tau(y-\hat{x}) f_X(y)\big|_{y=\hat{x}} \\ &\quad + \tau\int_{\hat{x}}^{\infty} \frac{\partial}{\partial\hat{x}}(y-\hat{x}) f_X(y)dy \\ &= 0 - \tau\int_{\hat{x}}^{\infty} f_X(y)dx \\ &= \tau(F_X(\hat{x})-1).\end{aligned}$$

By collecting the terms and setting the expression equal to zero we obtain

$$(1-\tau)F_X(\hat{x}) + \tau(F_X(\hat{x})-1) = 0 \iff F_X(\hat{x}) = \tau$$

which shows we are able to obtain the quantile τ.

When F_X is replaced by the sample distribution function

$$\hat{F}_X(x) = \frac{1}{n}\sum_{i=1}^{n} 1_{X_i \le x}$$

we may still select $\hat{x}$ to minimize the expected loss. And by doing so we obtain the *sample quantile function*. This is shown by first introducing the expected loss using the sample distribution function,

$$\begin{aligned}\mathbb{E}[p_\tau(X-\hat{x})] &= \frac{1}{n}\sum_{i=1}^{n}(X_i-\hat{x})(\tau - 1_{X_i-\hat{x}<0}) \\ &= \frac{1}{n}\left((\tau-1)\sum_{\hat{x}\geq X_i}(X_i-\hat{x}) + \tau\sum_{\hat{x}<X_i}(X_i-\hat{x})\right).\end{aligned}$$

By differentiating with respect to $\hat{x}$ and setting the expression equal to zero we obtain

$$\begin{aligned}&\frac{1}{n}\frac{\partial}{\partial\hat{x}}\left((\tau-1)\sum_{\hat{x}\geq X_i}(X_i-\hat{x}) + \tau\sum_{\hat{x}<X_i}(X_i-\hat{x})\right) \\ \iff &\frac{1}{n}\left((\tau-1)\sum_{\hat{x}\geq X_i}(-1) + \tau\sum_{\hat{x}<X_i}(-1)\right) = 0 \\ \iff &\frac{1}{n}\sum_{\hat{x}\geq X_i}1 = \tau\end{aligned}$$

where τ is the sample quantile. In 1978, Roger Koenker and Gilbert Bassett [11] were able to generalize these observations and introduce the *conditional quantile* function $Q_y(\tau|\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\beta}(\tau)$ where $\boldsymbol{\beta}(\tau)$ is the solution to the optimization

$$\mathcal{L}_{QR} = \min_{\boldsymbol{\beta}}\sum_{i=1}^{n}\rho_\tau(y_i - \boldsymbol{x}_i^T\boldsymbol{\beta})$$

and y is the variable of interest and $\boldsymbol{x}$ are inputs. The subscript QR stands for Quantile Regression. This opens up the possibility to model quantiles as functions of input variables. Modelling quantiles can be more appealing than doing point estimation since central tendency, scale and shape are distribution measures captured by the quantiles.

With the introduction of artificial neural networks, the interest to model the quantiles with non-linear models grew. In 1995, Halbert White, cited by James W. Taylor [14], provided theoretical support for such as possibility. Using a neural network to map the function with input variables, the

optimization problem becomes

$$\mathcal{L}_{QRNN} = \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \rho_\tau \left(y_i - f(\boldsymbol{\theta}, \boldsymbol{x}_i) \right) \tag{3.4}$$

where $\boldsymbol{\theta}$ represents the weights in the neural network f. The subscript $QRNN$ stands for Quantile Regression Neural Network.

3.2 Multi-Horizon Quantile Recurrent Neural Network

The Multi-Horizon Quantile Recurrent Neural Network (MQRNN), which was introduced in 2017 by Wu et al. [17], expands on the ideas elaborated on above. It is a framework for forecasting quantiles several time steps into the future. And although this thesis forecasts a single variable, for completeness it should be stated that the MQRNN is designed to handle several different time series. It aims to find the conditional distribution,

$$p(y_{t+k,i}, \ldots, y_{t+1,i} | y_{:t,i}, x^{(h)}_{:t,i}, x^{(f)}_{t:}, x^{(s)}_{i})$$

where $y_{\cdot,i}$ is the i'th time series to forecast, $x^{(h)}_{:t,i}$ historical inputs, $x^{(f)}_{t:}$ forecasted inputs and $x^{(s)}_{i}$ static inputs. The static input $x^{(s)}_{i}$ is key to this design as its time-series specific. Translated to the work of this thesis, it opens up the possibility to use *one* MQRNN to forecast the regulating price on several different price areas in the Nordic power market. According to the authors, the network is able to bridge the behavior of different time series on which it has trained. It is also a way to combat the issue of forecasting a variable with little or no historical data to train on. Since this thesis only investigates the regulating price in Finland the i'th notation is dropped and there are no static inputs.

In comparison to (3.4) where f originally was a one hidden layered multilayer perceptron (MLP) the MQRNN incorporates several different neural networks. The model is illustrated in Figure 3.1.

Let K be the horizon (number of timesteps) to forecast and Q the number of quantiles. For any forecast creating time point t, the network will output a $K \times Q$ matrix $\hat{\boldsymbol{Y}} = [\hat{y}^{(q)}_{t+k}]_{k,q}$. The goal is to minimize the *quantile loss function* (note it is the same function as (3.1) but rewritten)

$$L_q(y, \hat{y}) = q \max(y - \hat{y}, 0) + (1 - q) \max(\hat{y} - y, 0). \tag{3.5}$$

To optimize the parameters in the network the following metric is minimized during training

$$\mathcal{L}_{MQRNN} = \sum_{t=1}^{T} \sum_{q=1}^{Q} \sum_{k=1}^{K} L_q \left(y_{t+k}, \hat{y}_{t+k}^{(q)} \right). \tag{3.6}$$

Put in words, the inner sum is a summation over the entire horizon K. The middle sum is over all the quantiles Q. The outer sum is across all forecast creating time points in the training data.

As shown in Figure 3.1 the MQRNN consists of three different neural networks. A recurrent neural network (RNN) acts as *encoder* while the other two, which are part of the *decoder*, are multilayer perceptrons (MLPs) and referred to as the *Local MLP* and *Global MLP*. The equations are,

$$h_t = f(h_{t-1}, x_t, y_t), \tag{3.7}$$

$$[c_{t+1}, \ldots, c_{t+K}, c_a] = m_{Global}(h_t, x_{t:}^{(f)}), \tag{3.8}$$

$$[y_{t+k}^{(q_1)}, \ldots, y_{t+k}^{(q_Q)}] = m_{Local}(c_{t+k}, c_a, x_{t+k}^{(f)}). \tag{3.9}$$

The MQRNN architecture is designed to solve a *sequence-to-sequence (Seq2Seq)* type of problem. In 2013 Alex Graves [7] demonstrated that it was possible to use the Long Short-Term Memory (LSTM) to generate sequences of data with long-range structure. The *encoder-decoder* framework, introduced by Cho et al in 2014 [3] for Statistical Machine Translation (SMT), is a continuation of solving Seq2Seq problems. The main idea is to have a neural network *encode* the input and produce a representation, also known as the *context*, which another neural network then decodes to the *output*. When trained together, the encoder and decoder can approximate more intricate relationships. In addition, when using RNNs as encoder and decoder, it opens up the possibility to have input and output of arbitrary lengths which may not equal.

In the case of the MQRNN, as can be seen in Figure 3.1 and equation (3.7), an RNN is used as encoder. It opens up the possibility to have input vary in length and captures temporal differences in the historical data due to its recursive nature. The decoder of the MQRNN is not an RNN but consists of two MLPs, the Global- and Local MLP.
The Global MLP (3.8) takes the encoding h_t as well as the flattened predictions $x_{t:}^{(f)}$ as input. The output is a vector with elements which either

act as a horizon-specific context $c_{t+1}, \ldots, c_{t+K}$ or a horizon-agnostic context c_a. The contexts must not be of a single element, but can contain several elements.
The Local MLP (3.9) acts on each specific time point in the horizon. For any horizon $t+k$ it takes the horizon specific context c_{t+k}, the horizon independent context c_a as well as the forecasted inputs at that particular horizon $x^{(f)}_{t+k}$. The output at each specific horizon is a vector containing all quantiles. For each time step t, the local MLP iterates through the horizons and thus produces the matrix $\hat{\boldsymbol{Y}} = [\hat{y}^{(q)}_{t+k}]_{k,q}$.

A motivation for having two MLPs in the decoder comes from the authors emphasis on having both horizon-specific context as well as horizon-agnostic context. The horizon-specific context, as stated by the authors, "carries network-structural awareness of the temporal distance between a forecast creation time point and a specific horizon. This is essential to aspects like seasonality mapping." [17, p. 4] Meanwhile, some information may not be time-dependent and is thus captured by the horizon-agnostic context. At a forecast creation time point, it acts on the entire horizon. The authors find, empirically, an improvement in training stability as well as smoother forecasts when including the horizon-agnostic context.

While the MQRNN iterates through the horizons and produces the quantiles there also exists other approaches to generate forecasted sequences. The fairly common *recursive strategy*, at time t, treats the forecasted value $\hat{y}_{t+1}$ as the true value, $y_{t+1} := \hat{y}_{t+1}$, and then uses it as an input in the next time step. This is repeated for the entire forecasting horizon $t+1, t+2, \ldots, t+K$. The MQRNN does not employ the recursive strategy but uses the *Direct Multi-Horizon strategy* instead. By using it, the model is trained on a multivariate target. The authors state the latter strategy is less prone to accumulating errors over time since it does not use forecasted target values as ground truth and is efficient as parameters are shared across predictions.

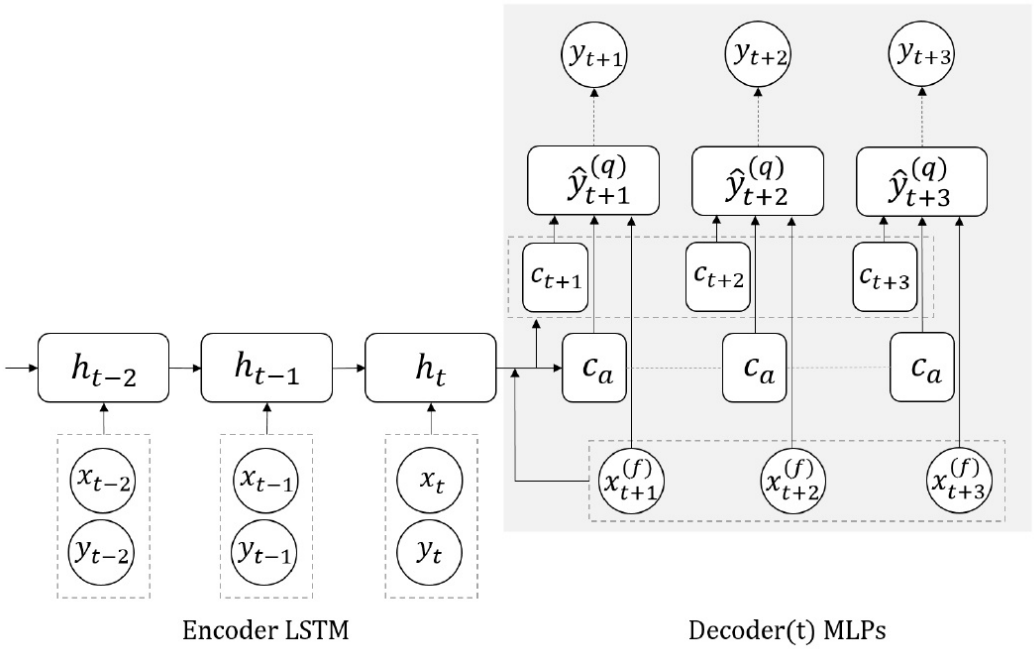

Figure 3.1: The architecture of the MQRNN [17]. Circles represent observed data. Squares are network nodes. First, a recurrent neural network (RNN) acts as encoder and encodes historical data. The gray area is the decoder. A dashed box implies the flattening of a vector. Dashed lines implies replication. The decoder, which consists of two multilayer perceptrons (MLPs), decodes and outputs the quantiles. In the picture the encoder is a Long Short-Term Memory (LSTM) network, but this thesis uses a Gated Recurrent Unit (GRU) network. In comparison to the original illustration, the forking-sequences scheme suggested by the authors is removed since this thesis implements the rolling-window scheme.

Chapter 4

Implementation

This section covers the implementation of the MQRNN. All programming was done in Python 3 in Google Colab. Google Colab enables the limited free use of a GPU which speeds up training and evaluation of neural networks. The Tensorflow package was used to build the model and input pipeline.

4.1 Data Collection and Preprocessing

Two data sources were used for collecting data, the power exchange Nord Pool and Finnish TSO Fingrid. In recent years, both have developed *Application programming interfaces (APIs)* from which data can be downloaded. Nord Pool launched its API in July 2019 and Fingrid made its beta API available in February 2017. While Nordpool provides plenty of market data for several price areas in Northern Europe, Fingrid publishes country unique data for Finland. This includes real-time data on a three minute basis for, among other power generation sources, nuclear-, wind- and hydro power production in Finland. The initative by Fingrid is currently unique in the Nordics but other countries are following similar paths. The Swedish TSO, Svenska Kraftnät, is set to start testing a similar API by 2022.

It was decided that a rolling window of $t + 3h, \ldots, t + 8h$ should be the forecast horizon. The reason for not including $t + 1h$ and $t + 2h$ are due to trading reasons. It also allows all historical data to be published and accessed. Forecasting beyond $t + 8h$ would be useful, but a horizon of eight hours ensures all data used as forecasting inputs to be available. In Table 4.1 the data used can be reviewed. In total, 69 variables are used in the model.

Some descriptions have been provided to clarify the content of a variable. It has also been indicated whether it has been used as historical data for the encoding, forecasted input in the decoder or both.

Data was downloaded from 2016-01-01 until 2020-04-01. Subsequently the data was split such that 80% was used to train, 10% to validate and 10% to test. Since the forecast of the regulating price would be hourly, the input data was also made to be hourly. Any variables which were reported on a more frequent basis were downsampled to hourly averages. As is common practice in deep learning, the data was preprocessed before being modelled. It is often necessary to have the data on similar scales. Thus the data was *normalized* using

$$\frac{X - \mu_{X_{train}}}{\sigma_{X_{train}}}$$

where $\mu_{X_{train}}$ and $\sigma_{X_{train}}$ are the sample mean and sample standard deviation from the training data of any variable X. The preprocessing method of *normalization* is commonly used in time series forecasting within deep learning [15]. One drawback when normalizing a variable with a large variance is that large values influence the normalized data heavily. It shrinks values which are more centrally located in the distribution. This is certainly the case for regulating price data which can be close to 0 but also contains -1000 and 3000. Two exceptions to the preprocessing was made. First, the variable called *Dominating Direction* was already in the set $\{-1, 0, 1\}$. Secondly, all the time variables were in the set $\{0, 1\}$. Thus no preprocessing was required for these variables since their values were close to zero and integers.

Data	Description	Usage
Regulating price	Value to be forcasted	Target and historical
Dominating direction	Dominating direction of regulating volumes $\{1, 0, -1\}$	Historical
Regulating prices SE1, SE3	Regulating prices of neighboring price areas	Historical
Nuclear power production	Power from nuclear power	Historical
Transmission between FIN and SE1	Power transmission in cable between price areas	Historical
Transmission between FIN and SE3	Power transmission in cable between price areas	Historical
Production surplus/deficit	Domestic surplus or deficit of power production	Historical
Industrial cogeneration	Power from industrial cogeneration	Historical
Hydro power production	Power from hydro	Historical
Other small scale production	Power from small scale production and reserve sources	Historical
Cogeneration of district heating	Power from cogeneration of district heating	Historical
Change in temp. in Rovaniemi, Jyväskylä and Helsinki	Differentiated temperature $x_t - x_{t-1}$ for major cities	Historical
Ordered up-regulations from balancing energy	The volume of ordered up-regulations from balancing energy market	Historical

Ordered down-regulations from balancing energy	The volume of ordered down-regulations from balancing energy market	Historical
The sum of the up-regulation bids in the balancing energy	Volume of up-regulating bids	Historical
The sum of the down-regualtion bids in the balancing energy	Volume of down-regulating bids	Historical
Spot prices FIN, SE1, SE3, EE	Spot prices in Finland and neighboring price areas	Forecasted and historical
Total production prognosis	Prognosis of total power production	Forecasted
Total consumption prognosis	Prognosis of total power consumption	Forecasted
Volume of up-regulating bids in regulating market	Sum of bids for up-regulation	Forecasted and historical
Volume of down-regulating bids in regulating market	Sum of bids for down-regulation	Forecasted and historical
Transmission capacity for intraday market (FIN-SE1)	Intraday market capacity is given as free capacity after spot market	Forecasted and historical
Transmission capacity for intraday market (FIN-SE3)	Intraday market capacity is given as free capacity after spot market	Forecasted and historical
Transmission capacity for intraday market (FIN-EE)	Intraday market capacity is given as free capacity after spot market	Forecasted and historical

Adjusted volume of up-regulating bids	Volume of up-regulating bids divided by spot price	Forecasted and historical
Adjusted volume of down-regulating bids	Volume of down-regulating bids divided by spot price	Forecasted and historical
Time	Binary $\{0, 1\}$ variables for hour and weekday	Forecasted and historical

Table 4.1: List of data variables. Some descriptions have been added to clarify the content. The regulating price is the *target* variable to be forecasted. *Historical* indicates that the variable has been used only in the encoder as a historical input. *Forecasted* indicates that the variable has been used only in the decoder as a forecasted input. Some variables have been used for both.

4.2 Multi-Horizon Quantile Recurrent Neural Network

While the authors of the MQRNN provide the framework little is provided in terms of the architecture of the encoder and decoder. In other words, recommendations for hyperparameters such as the number of nodes or the number of layers of the encoder and decoder are not provided. Finding the correct hyperparameters for a neural network is still a trial-and-error process where different combinations are evaluated and compared. It requires plenty of computing power. Some iterating was done and the final model used is described in this section.

4.2.1 Encoder

A RNN is being used as encoder. But in comparison to the orginal paper and Figure 3.1 a *Gated Recurrent Unit (GRU)* was used instead of the Long-Short Term Memory (LSTM). The GRU is inspired by the LSTM but is easier to compute [3]. And in Seq2Seq modelling, the GRU performs on par with the LSTM [4]. The GRU cell is illustrated in Figure 4.1. For one GRU cell the equations are

$$
\begin{aligned}
r_t &= \sigma\left(\boldsymbol{w}_r^T \boldsymbol{x}_t + u_r h_{t-1}\right), \\
z_t &= \sigma\left(\boldsymbol{w}_z^T \boldsymbol{x}_t + u_z h_{t-1}\right), \\
\tilde{h}_t &= \phi\left(\boldsymbol{w}^T \boldsymbol{x}_t + u r_t h_{t-1}\right), \\
h_t &= z_t h_{t-1} + (1 - z_t)\tilde{h}_t.
\end{aligned}
$$

There are two gates, r_t is the *reset gate* and z_t is the *update gate*. The hidden state h_t depends on $\tilde{h}_t$ as well as the previous hidden state. When the reset gate is close to zero, $\tilde{h}_t$ depends on the current input. This is passed on to the hidden unit h_t and tuned by the update gate z_t. The parameters $\boldsymbol{w}_r, \boldsymbol{w}_z, \boldsymbol{w}, u_r, u_z$ and u are to be optimized during training. The activation functions, σ and ϕ are the sigmoid and hyperbolic tangent (tanh).

By using several hidden units together, the RNN can capture dependencies across different time lengths. This is possible since the parameters are initialized randomly and not shared across the units. This implies each individual GRU unit has different gates. To exploit this, 32 GRU units were used in the model.

There is some periodicity in the energy market and to capture this the last 168 hours (one week) of historical data was fed to the RNN at each time point t to create the context for the decoder.

4.2.2 Global Decoder

The purpose of the global decoder, m_{Global}, is to create horizon-specific contexts as well the horizon-agnostic context. In the MQRNN paper it is stated that the contexts can be of an arbitrary dimension. Inspired by this, and given a horizon of six timesteps, each of the horizon-specific contexts (six) as well as the horizon-agnostic context (one) was given five elements. Thus totalling 35. The input, consisting of the encoding and future inputs, totalled 314 when flattened. Thus the decoder was set up as $m_{Global} : \mathbb{R}^{314} \to \mathbb{R}^{35}$.

As activation function, the *Exponential Linear Unit (ELU)* was selected. Introduced in 2016, it improves upon the *Rectified Linear Unit (ReLU)*. Two benefits are quicker learning and better generalization of inputs [5]. The parameter α can be tuned but was set to the default value of 1. The function and its derivative are stated below and illustrated in Figure 4.2.

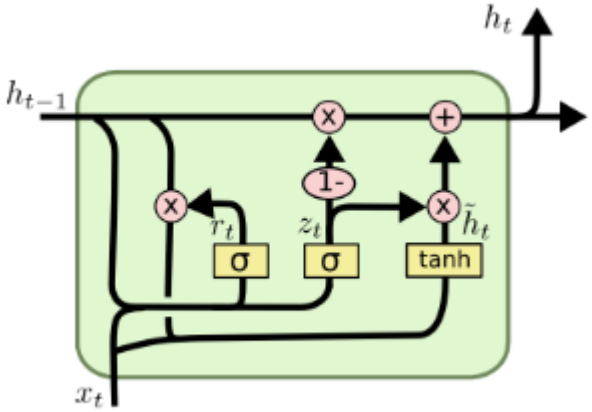

Figure 4.1: One GRU cell illustrated [16]. The reset gate, $r_t = \sigma\left(\boldsymbol{w}_r^T \boldsymbol{x}_t + u_r h_{t-1}\right)$, determines how much previous information should be let through. The update gate, $z_t = \sigma\left(\boldsymbol{w}_z^T \boldsymbol{x}_t + u_z h_{t-1}\right)$, determines how much new information should be let through. It should be noted that in comparison to the equations above the hidden state in the picture is computed as $h_t = z_t \tilde{h}_t + (1 - z_t) h_{t-1}$. The parameters $\boldsymbol{w}_r, \boldsymbol{w}_z, \boldsymbol{w}, u_r, u_z$ and u are to be learned. The activation functions, σ and ϕ are the sigmoid and hyperbolic tangent (tanh).

$$f(x) = \begin{cases} x, & x > 0 \\ \alpha(e^x - 1), & x \leq 0 \end{cases}$$

$$f'(x) = \begin{cases} 1, & x > 0 \\ \alpha e^x, & x \leq 0 \end{cases}$$

4.2.3 Local Decoder

Since the authors of the original paper place such emphasis on the local decoder, m_{Local}, extra care was given to it during implementation. Two experiences are worth shedding additional light on. First, the local decoder was unable to generalize without any activation function. Attempts were made such that only linear combinations of the input to the local decoder were processed without any activation function. The network was more prone

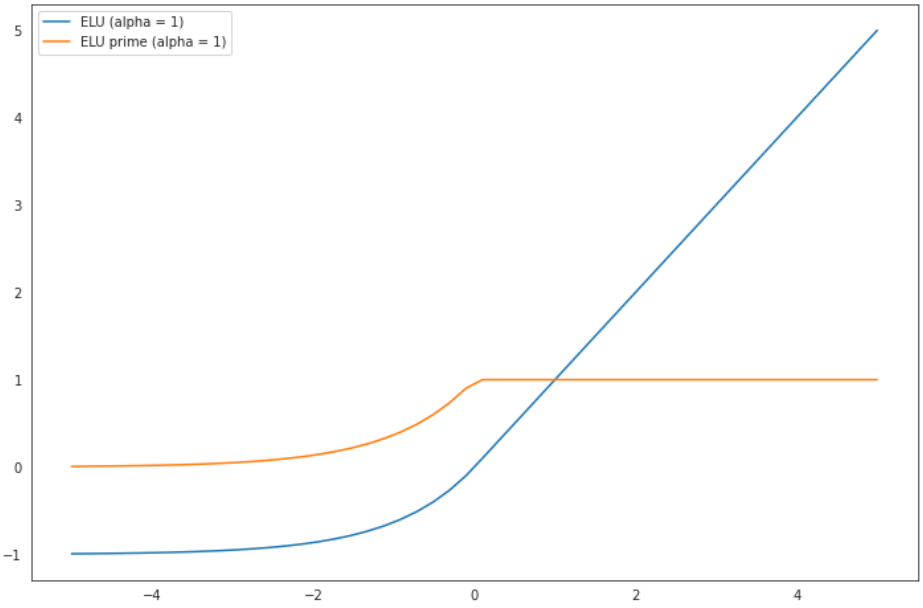

Figure 4.2: The Exponential Linear Unit (ELU) and its derivative. The function is a development of the Rectified Linear Unit and offers quicker learning and better generalization of inputs [5]. The parameter α can be tuned but has been set to the default value of 1.

to experience the issue of crossing quantiles in such attempts. Due to this, the ELU function (see section 4.2.2) was used as activation function. Secondly, the local decoder was sensitive to the number of nodes in the hidden layer. By increasing the number of nodes more information was captured and a better performance in the quantiles observed. As such, a shared hidden layer of 25 nodes was used. Finally to produce the quantiles, at each of the six horizons, a linear function mapped the hidden layer to the quantiles. In total, $m_{Local} : \mathbb{R}^{57} \to \mathbb{R}^{25} \to \mathbb{R}^{11}$. To remind the reader, the dimension of 57 comes from the input to m_{Local}. At each timestep there were 47 forecasted inputs and the horizon-specific as well as the horizon-agnostic contexts were each five dimensional. This totals 57. As for the quantiles, eleven were computed. These were $q_1, q_{10}, q_{20}, q_{30}, q_{40}, q_{50}, q_{60}, q_{70}, q_{80}, q_{90}$ and q_{99}.

4.2.4 Additional Hyperparameters

To prevent the network from depending too much on certain parameters and combat overfitting *dropout* was applied before the output layer. When dropout is applied the network will set a predefined number of random weights equal to zero during training. The fraction of weights to be equal to zero at random was set to 0.5. By also applying dropout the network was able to generalize better and prevent the issue of crossing quantiles.

4.2.5 Optimization

For the optimization procedure a batch size of 32 was used. It is a batch size which is common and recommended as default [2]. As optimizer the *Adaptive Moment Estimation (Adam)* was used. With sparse data it is recommended to use an adaptive optimizer. Another benefit with adaptive optimizers is that they tune the learning rate themselves. As such, the default value can be left as is. Altogether, Adam is considered the best optimizer [13]. The optimization of parameters, also known as *training* within deep learning can be seen in Figure 4.3. The figure shows average loss across batches of equation (3.6). In the legend, Train indicates the loss on data which the network has seen and performed backpropagation on. Test indicates data which the model has not seen and only been evaluated on. A couple of things are worth highlighting from the figure.

First, except for the first epoch, the loss is smaller for the training data compared to the test data. This can be explained by the inner workings of Keras [9]. During training, the loss is reported as the average across batches. Meanwhile during testing, the model uses the weights as they are at the end of an epoch. In later epochs, the training loss continues to decrease suggesting an improvement to the model. This leads to the second notion, which is the curve for the test loss. After epoch three it starts to increase. This behavior is known as *overfitting*. It indicates the model continues to improve on the training data while a decrease in performance on unseen data takes place. This thesis applied two methods to combat this. First, as already mentioned, dropout was applied to prevent the model depending too much on specific input variables. Secondly, *early stopping* was applied. Early stopping makes the model stop training after a decrease in performance on test data has been observed for n epochs. In this case, $n = 2$.

Third, in comparison to other areas where deep learning is being applied,

five epochs is very small. In image- and speech recognition optimization can take place for hours. But a prerequisite is a large amount of data. This is not the case of this thesis since approximately 38000 data samples were used for training.

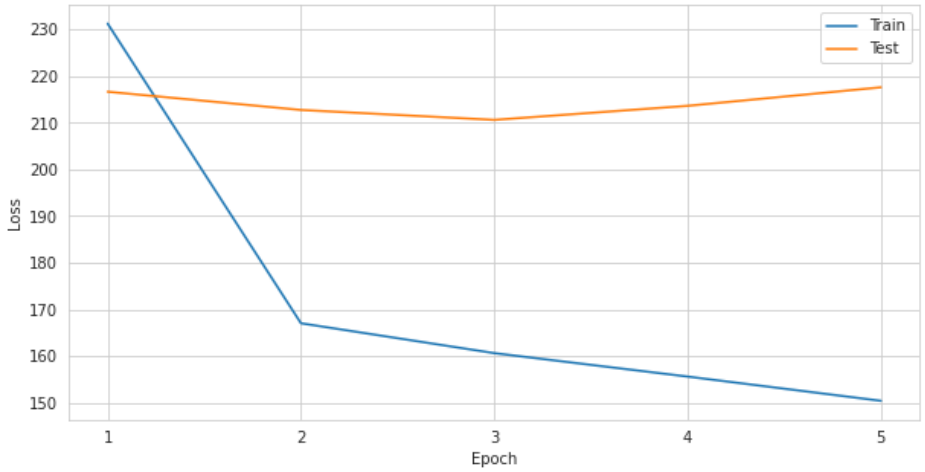

Figure 4.3: Average loss across batches of (3.6) for train and test data. Except for the first epoch the loss is smaller for the training data on which the network performs backpropagation. The loss on training data continues to decrease while the network starts overfitting after epoch three on the test data. Early stopping was applied after two epochs of poorer test performance. Running the model for five epochs is considered small by deep learning standards. The small number of epochs can be explained by relatively little data.

Chapter 5

Results

In this section the results will be presented. The model used is the best performing one (epoch three in Section 4.3). To illustrate the quantiles, the rolling window of $t + 3h, \ldots, t + 8h$ has been illustrated for 10 consecutive hours of the validation data. For comparison, the spot- and the regulating price has been plotted together with the quantiles. The development is easy to see by following the the regulating price (black line) as it shifts by one hour in every plot. All figures are in Appendix A.

Following, to evaluate the model en masse, histograms of the forecasting errors were created for 1000 hours of the validation data. The term forecasting error can be misleading since the quantiles are not point forecasts. But in lack of a better word it will be used. The errors were computed as

$$\hat{y}_{t,j}^{(q)} - y_{t,j} = \epsilon_{t,j}^{(q)}.$$

for all $t = 1, \ldots, 1000$ and $j = t+3, \ldots, t+8$ and $q = 0.01, 0.1, 0.5, 0.9, 0.99$. Theses figures can also be found in Appendix A.

5.1 Quantiles

By looking at Figures A.1 - A.10 it can be seen that the central location of the distribution is around the spot price. Plenty of probability mass is located around the center as the 10% and 90% quantiles are close to the spot price as well. That being said, the model is able to expand and contract the quantiles. During some hours they are wide while during other hours they are close to each other. The quantiles are also smooth over time. For

a particular hour, the quantiles appear to be stable as that hour approaches the forecast creating time point. The tail quantiles show a different behavior as they are further away from the central location of the distribution. Many times the difference between the 99% and 90% quantile is big. This also applies to the 1% and 10% quantile, but not to the same extent.

From the Figures A.1 - A.10 it might look as if the central quantiles are parallel. This is not the case but due to the scale of the y-axis any changes might not stand out.

5.2 Forecasting Errors

While the figures of the quantiles are informative, the histograms of the forecasting errors paint a more holistic picture. Since it already had been indicated plenty of probability mass was located at the center of the distribution the errors of the quantiles further out in the distribution were investigated. For completeness, and importance, the errors for the median were also computed.

By looking at the histograms for the 99% quantile in Figure A.11, it can be seen that in a few cases the error is negative. This indicates the regulting price has surpassed the quantile. The majority of errors are positive and located quite close to 0. Yet, the model appears to overshoot the true regulating price quite freuently since plenty of errors are rather large and positive. For the 90% quantile in Figure A.12 the behavior is similar with plenty of positive errors. Compared to the 99% quantile, the errors are more concentrated and closer to zero. The 50% quantile, also known as the median, is the center of the distribution. In Figure A.13 it can be seen that besides the large negative errors, the distribution is centered around zero. The 10% and 1% quantiles are close to mirror images of the 90% and 99% quantiles. These can be reviewed in Figures A.14 and A.15. The difference is the frequency and magnitude of the regulating price dropping below the quantiles. From the histograms it can be seen that the downside of the regulating price is not as extreme as the upside.

Chapter 6

Discussion

In this section the figures of the quantiles and the forecasting errors will be elaborated upon. Aspects of the implementation will also be taken into consideration. Finally, conclusions will be drawn and some recommendations for future work will be discussed.

6.1 Results Discussion

From the results it appears as if the regulating price has a distribution with plenty of probability mass close to the spot price and long tails. While such behavior can be suspected by merely looking at the raw data, the model aims to quantify this observation. At the onset of this thesis far fewer quantiles were used in the output. It was believed that quantiles close to the center of the distribution would be more prone to change. When this proved not to be the case, more quantiles were added and it was noted the 99% quantile seemed to change more frequently. It raised the question why the 90% quantile was not moving as much. But upon inspecting the forecasting errors it was confirmed that most of the time the 90% quantile was above the regulating price. And the distribution of the errors was centered closer to zero in comparison to the 99% quantile. This gives weight to the notion of the regulating price having plenty of probability mass around the spot price, but the right tail can be very long.

In capturing the right tail behavior the model shows promising results. In some cases the 99% quantile can become very big and thus indicate up-regulation. But in the quantile figures, and confirmed in the error histograms,

the model also overshoots the regulating price quite frequently. This can be seen in Figure A.11. As such one could argue the 99% quantile captures the potential but not always the actual outcome of the regulating price. That being said, the distribution also exhibits a longer left tail. By looking at the histograms of the errors, the model appears to better capture the behavior of the left tail. There are few cases when the regulating price has gone below the 1% quantile. This can be seen in Figure A.15. Extreme up-regulation is however more frequent than extreme down-regulation. Hence it cannot be ruled out if the model was not evaluated on enough extreme down-regulation.

In statistical terminology, the distribution is said to be skewed to the right. And as such, the median does not need to equal the mean. With a right skew, the mean is instead pushed to the right of the median. From a practical point of view, the distribution can be difficult to interpret and act upon. For example, in some of the quantile figures the median is below the spot price. According to the model, there is thus more than 50% probability of down-regulation. At the same time the right tail can be very long. This potentially pushes the expected value above the spot price and another way would be to interpret the results as a case of up-regulation. Attempts were made to quantify the skewness. A robust measure is the quantile skewness γ_Q, also known as the *Bowley skewness* or *Galton skewness* [1]. It is computed as

$$\gamma_Q = \frac{(q_{75} - q_{50}) - (q_{50} - q_{25})}{(q_{75} - q_{25})}.$$

It yields a value between $[-1, 1]$. A positive (negative) number implies a right (left) skew. Complete symmetry yields 0. The 75% and 25% quantiles can be replaced by other quantiles, for example 99% and 1%. The measure was computed with both set of arguments. Due to the the length of the right tail the skewness, when using 99% and 1%, was always right. When using 75% and 25% on the other hand, the skewness could vary more.
One matter which renders this measure useless is the phenomenon called *crossing quantiles*. If, for example, $q_{75} < q_{50}$ the measure breaks down. In addition it violates the fundamental principles of probability theory. But it is a well known problem in quantile regression. Historically it is attributed to estimating the quantiles individually without any restrictions. This can lead to the conditional quantile function no longer being monotonically increasing. Still the problem is limited according to Koenker who write "It is of some comfort to recognize that such crossing is typically confined to outlying regions of the design space." [10, p. 56] During this thesis the problem

was encountered. But by applying dropout the performance was dramatically improved. Alas, in a very few cases the 99% quantile was observed to still experience the problem of crossing quantiles. While not directly touched upon in the orginal MQRNN paper, it is believed that the problem is mitigated by the design of the loss function. Since the loss is not computed independently for each quantile, any quantile crossing is penalized and the errors are backpropagated and captured in the parameters.

So far this thesis has not touched upon how many quantiles should be computed. Certainly many quantiles paint a greater picture of the distribution - but they can also be used to estimate the expected value. While not reported in the thesis, this would be how to proceed using the estimated quantiles. We begin with the definition of expected value of some stochastic variable X

$$\mu = \int_{\mathbb{R}} y f_X(y) dy.$$

Next, we limit the support to $[q_1, q_{99}]$ and note that since the probability mass under f_X is $0,98$ when the support is limited we introduce a normalizing constant

$$\hat{\mu} = \frac{1}{0.98} \int_{q_1}^{q_{99}} y f_X(y) dy.$$

By the substitution method we can set $u = F_X(y)$ and differentiate

$$\frac{du}{dy} = f_X(y) \iff du = f_X(y)dy$$

The new integration limits become

$$\begin{aligned} F_X(q_1) &\iff u = 0.01, \\ F_X(q_{99}) &\iff u = 0.99. \end{aligned}$$

and we obtain the integral

$$\hat{\mu} = \frac{1}{0.98} \int_{0.01}^{0.99} y du.$$

Finally, we note that $y = F_X^{-1}(u) = Q(u)$ and reformulate the integral as

$$\hat{\mu} = \frac{1}{0.98} \int_{0.01}^{0.99} Q(u) du.$$

As such we have an approximation of μ which depends on the quantile function. When forecasting several quantiles these can be used to estimate the integral. For this thesis the domain of integration would be partitioned into the set P with cardinality 11,

$$P = \{0.01, 0.1, 0.2, \ldots, 0.8, 0.9, 0.99\}.$$

The length of the ith subinterval of P is

$$\Delta u_i = u_i - u_{i-1}, \text{ for } 2 \leq i \leq 11.$$

This allows us to write $\hat{\mu}$ as

$$\hat{\mu} = \frac{1}{0.98} \sum_{i=2}^{11} \frac{(q_i + q_{i-1})}{2} \Delta u_i$$

which is an estimate of the expected value based on the quantiles. The estimate can be attractive when we want to summarize the quantiles into one number. But it is flawed in the sense that it discards the distribution before the first quantile and after the last quantile estimated. Naturally it also depends on how well the quantiles capture the true distribution.

For all intents and purposes, the big question is, how well the model can approximate the function which generates the distribution of the regulating price. Arguments both favoring and opposing the success of the model can be made. According to economic theory, if the Efficient Market Hypothesis holds, there should be no structure in the market which can be exploited. All information should already be reflected in the price. To verify this is hard. During the optimization of the model, the loss metric was not minimized very quickly. This speaks in favor of there being some structure in the market. The opposite behavior, where the loss metric decreases very fast directly can be observed when neural networks are tasked with approximating something stochastic. Not seldom the prediction is simply the last observed value [8]. But it should not be forgotten the randomness typical of financial assets also is reflected in the regulating price. During optimization the best performing model was obtained after three epochs and thereafter the model started overfitting. This speaks in favor of little structure in the data. Moreover, it is a bold statement to say that there is structure in the market since it implies the market is inefficient. Nonetheless, the results indicate that the quantiles, especially the ones in the tails, can capture the movements in the regulating price.

6.2 Conclusions

This thesis aimed to implement the MQRNN to forecast the regulating price in the Finnish energy market. This approach has been made possible with the recent availability of data from the Finnish power grid. From the implementation and results this thesis draws the following conclusions.

- **Conclusion 1**: The distribution of the regulating price is centered around the spot price. Plenty of probability mass is located close to the center. Both tails are long, especially the right tail.

This conclusion is supported by the findings in the quantile figures as well as the histogram of the errrors. While the 90% and 10% quantiles are located fairly close to the spot price, the tail quantiles can be further away from the central location of the distribution. Especially the right tail can be very long.

- **Conclusion 2**: There is some structure in the regulating price which the model can approximate.

This conclusion is supported by the behavior of the quantiles in the figures. While little happens to the central location of the distribution, the tail quantiles can change rapidly from hour to hour. This thesis can only speculate in why there would be inefficiencies in the balancing energy market. One reason could be that the energy market is not as liquid as other financial markets since there are fewer actors. Another would be it has a different dynamic and is driven by largely physical variables. For example wind, cable capacity, consumption, ect. While humans have difficulties detecting nonlinear relationships between several variables neural networks are capable of doing so.

6.3 Looking Ahead

To potentially improve the results, some recommendations for the future can be made.

First, this thesis did not implement the forking-sequences scheme introduced in the orginal MQRNN paper. The authors state it improved their results [17]. Building data pipelines for deep learning is not a trivial task

however and implementing tailored schemes is time and knowledge consuming. This was the main reason why the moving-window scheme was used in this thesis and not the forking-sequences.

Secondly, one drawback about this thesis is the absence of weather forecasts. In an age where wind power is a major source of energy, weather forecasts are important. One limitation to this is the lack of historical weather forecasts. While plenty of weather services make their current forecasts available, historical forecasts are not as accessible. And to fully implement weather forecasts, these must be available during the optimization of the model. Consideration must also be made as to what information was available at the forecasting creation time point t. In the interest of time this thesis did not incorporate weather forecasts. But it is believed this would improve the results.

Thirdly, the original intent of the MQRNN is to model several different time series. This opens up the possibility to model the regulating price in all price areas in the Nordics with one model. This could bring about benefits. For example, not all price areas have had the same type of extreme volatility as for example Finland or Denmark. Lack of such observations could potentially be mitigated by a model with shared parameters. The model can borrow statistical features across time series [17]. Another reason for having one model is that it allows the model to train on more data. While some input data could be the same, different price areas would produce different outputs. There is also a resource and computational aspect. Creating and maintaining a model for each price area is time and resource consuming.

Appendix A

Results

A.1 Figures of Quantiles

This appendix contains the quantile figures of 10 consecutive hours of the validation data. By every figure time shifts one hour and the forecast is updated. It can be seen by following the the regulating price (black line).

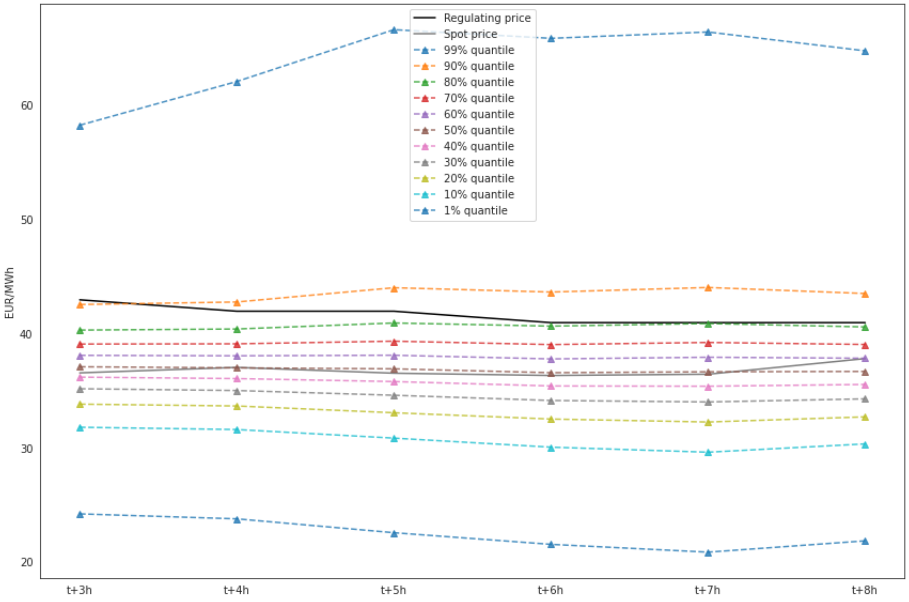

Figure A.1: Forecast $t + 3h, \ldots, t + 8h$.

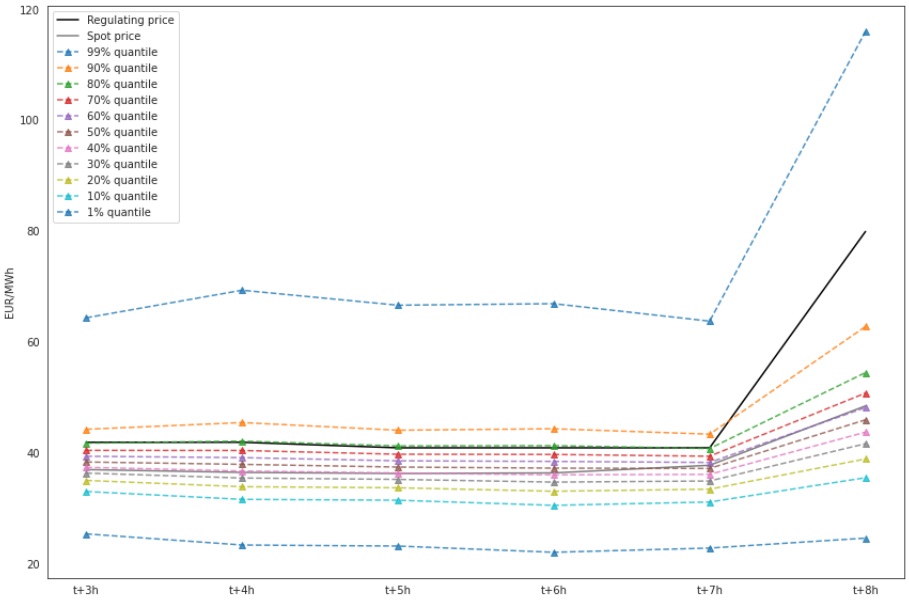

Figure A.2: Forecast $t + 3h, \ldots, t + 8h$.

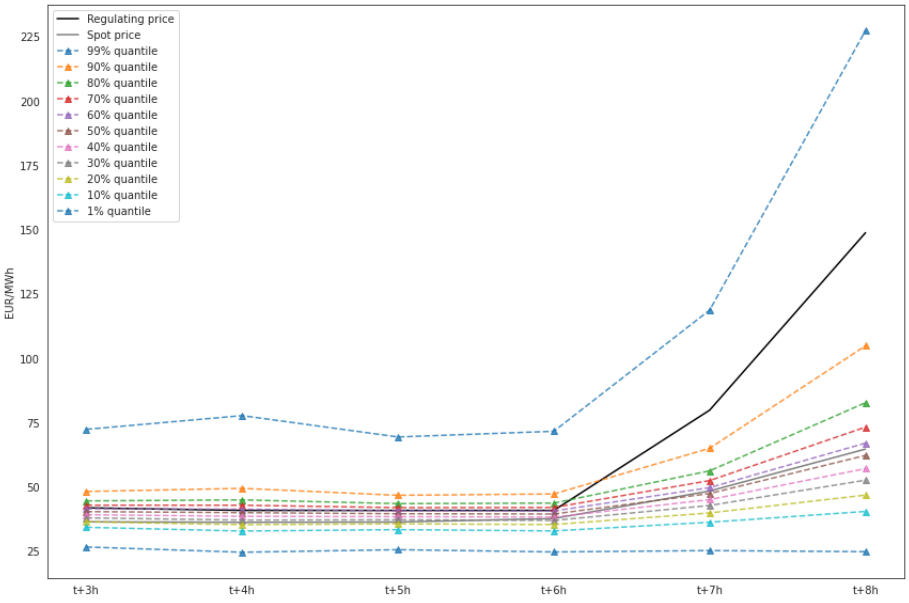

Figure A.3: Forecast $t + 3h, \ldots, t + 8h$.

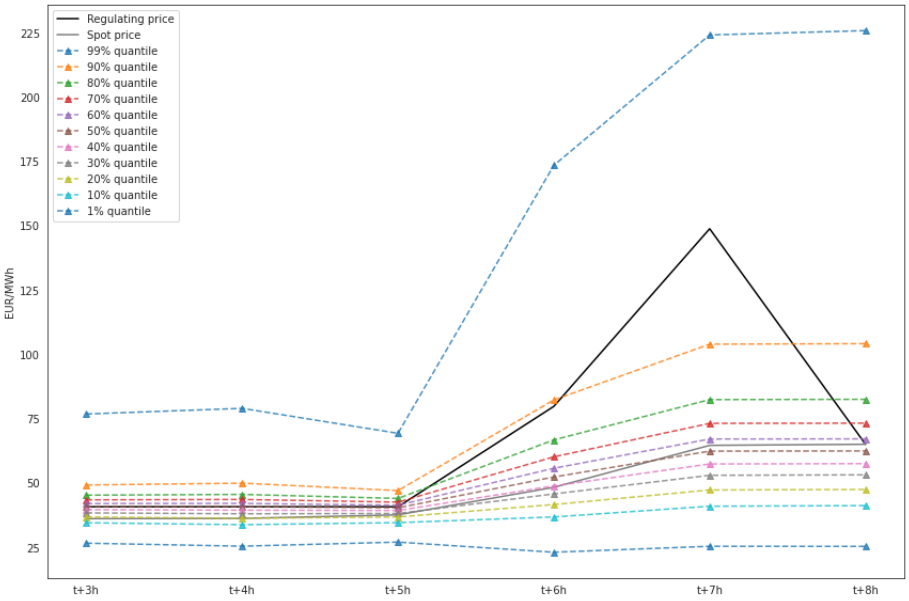

Figure A.4: Forecast $t + 3h, \ldots, t + 8h$.

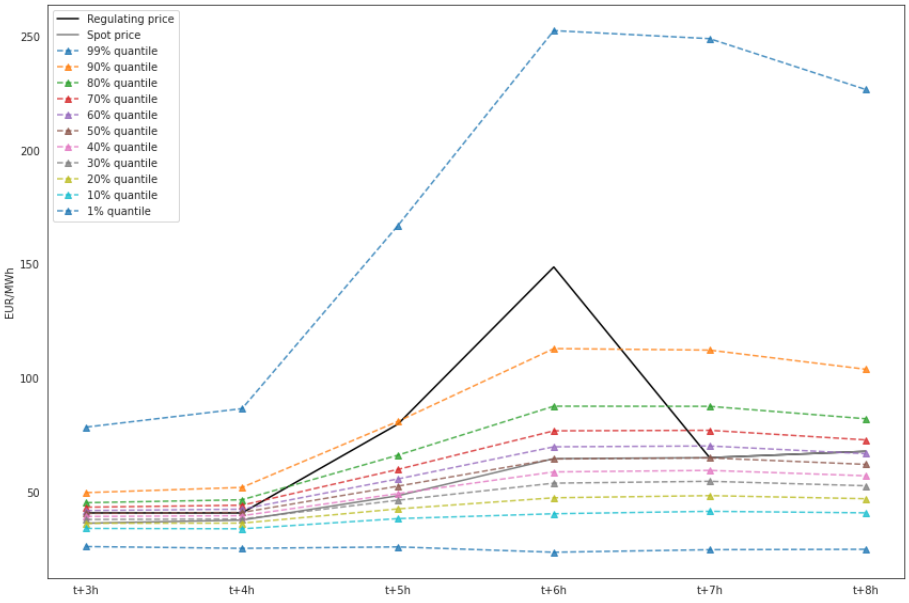

Figure A.5: Forecast $t + 3h, \ldots, t + 8h$.

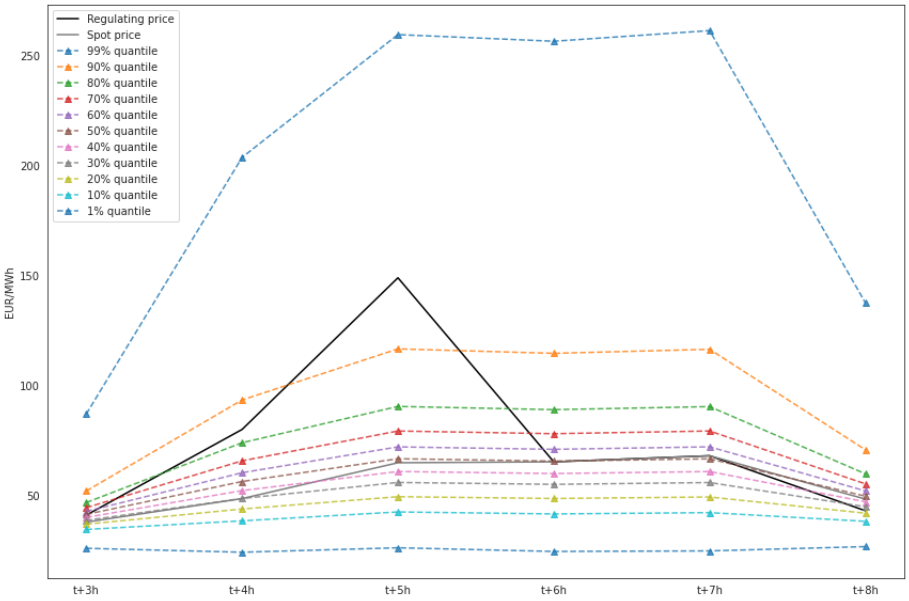

Figure A.6: Forecast $t + 3h, \ldots, t + 8h$.

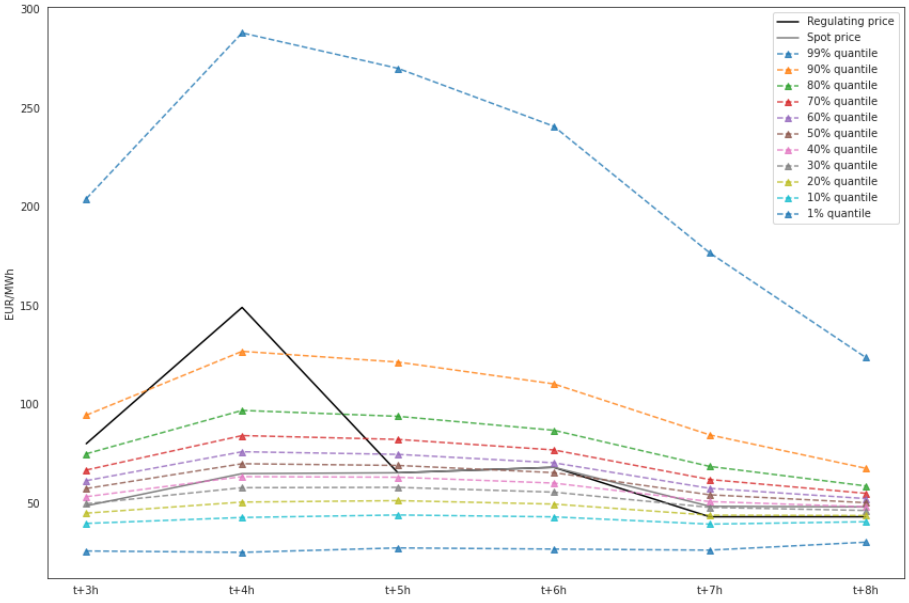

Figure A.7: Forecast $t + 3h, \ldots, t + 8h$.

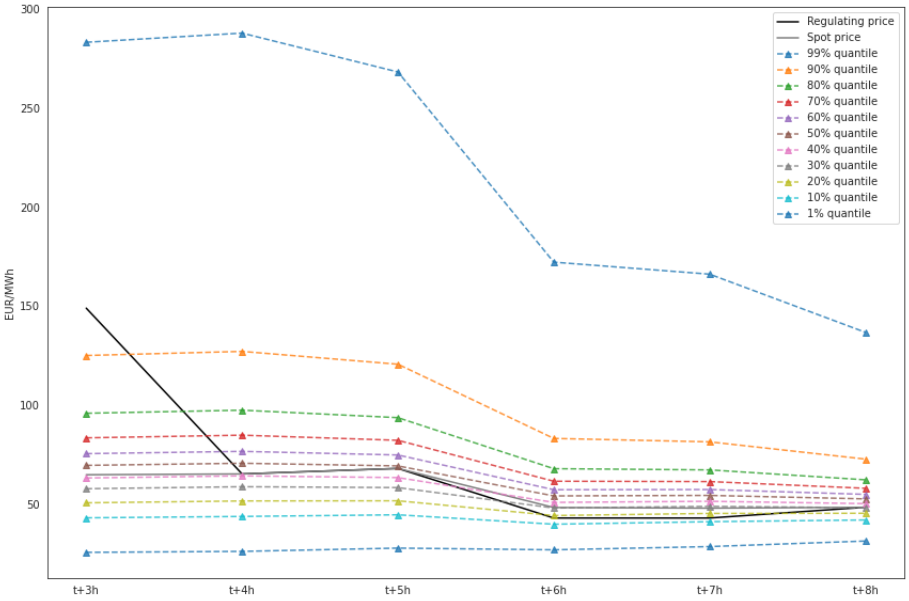

Figure A.8: Forecast $t + 3h, \ldots, t + 8h$.

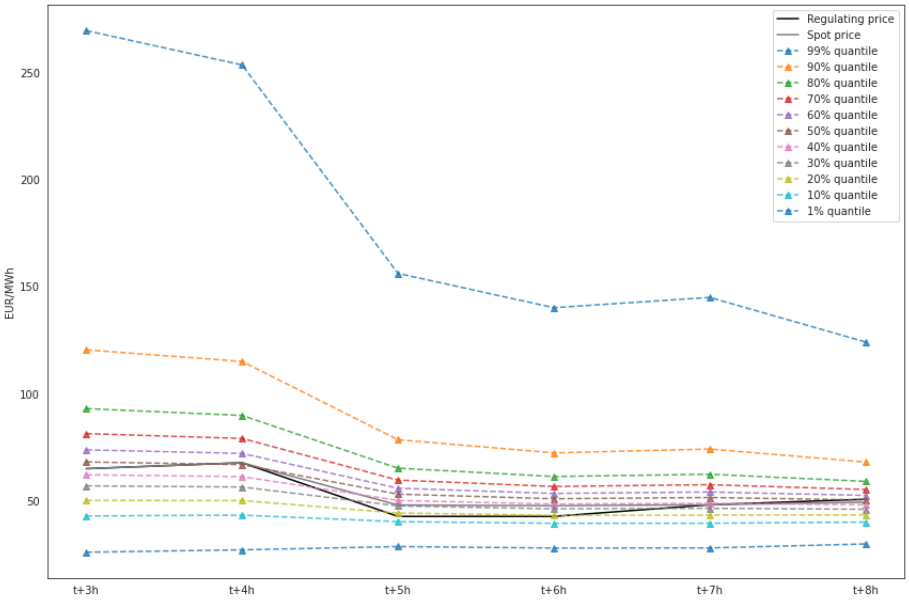

Figure A.9: Forecast $t + 3h, \ldots, t + 8h$.

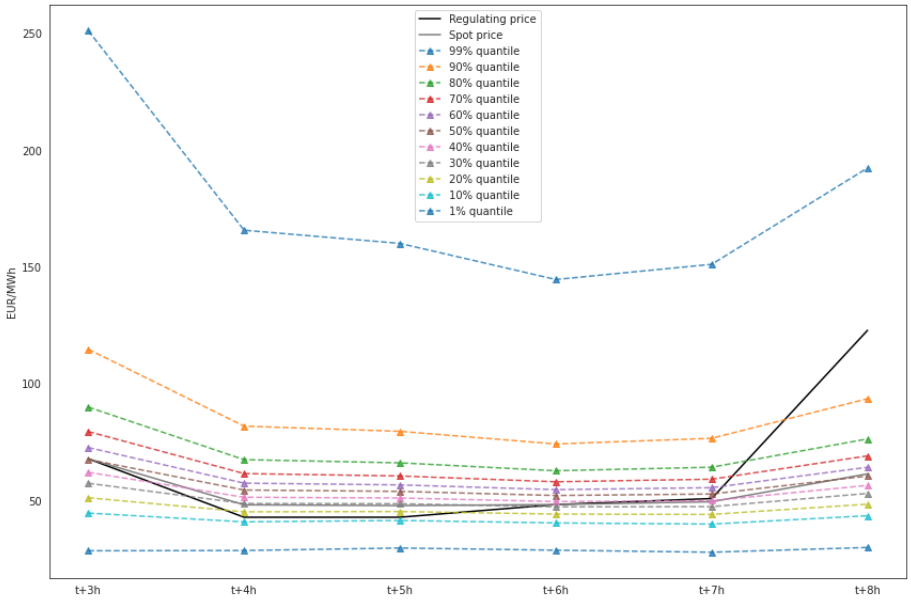

Figure A.10: Forecast $t + 3h, \ldots, t + 8h$.

A.2 Histograms of Forecasting Errors

This appendix contains histograms of the forecasting errors which were computed as

$$\hat{y}_{t,j}^{(q)} - y_{t,j} = \epsilon_{t,j}^{(q)}$$

for all $t = 1, \ldots, 1000$ and $j = t + 3, \ldots, t + 8$ and $q = 0.01, 0.1, 0.5, 0.9, 0.99$. To clarify, each histogram visualizes the errors for a specific timestep in the forecasting horizon for a specific quantile. Each quantile has been provided with a unique color.

The histograms provide a picture of what the error distributions look like. The x-axis consist of bins. The bars of the y-axis count how many errors are contained in a bin.

On the x-axis there are small vertical markers which indicate the location of an error. They provide visualization for extreme and less frequent errors. More dense markers indicate plenty of error occurrences.

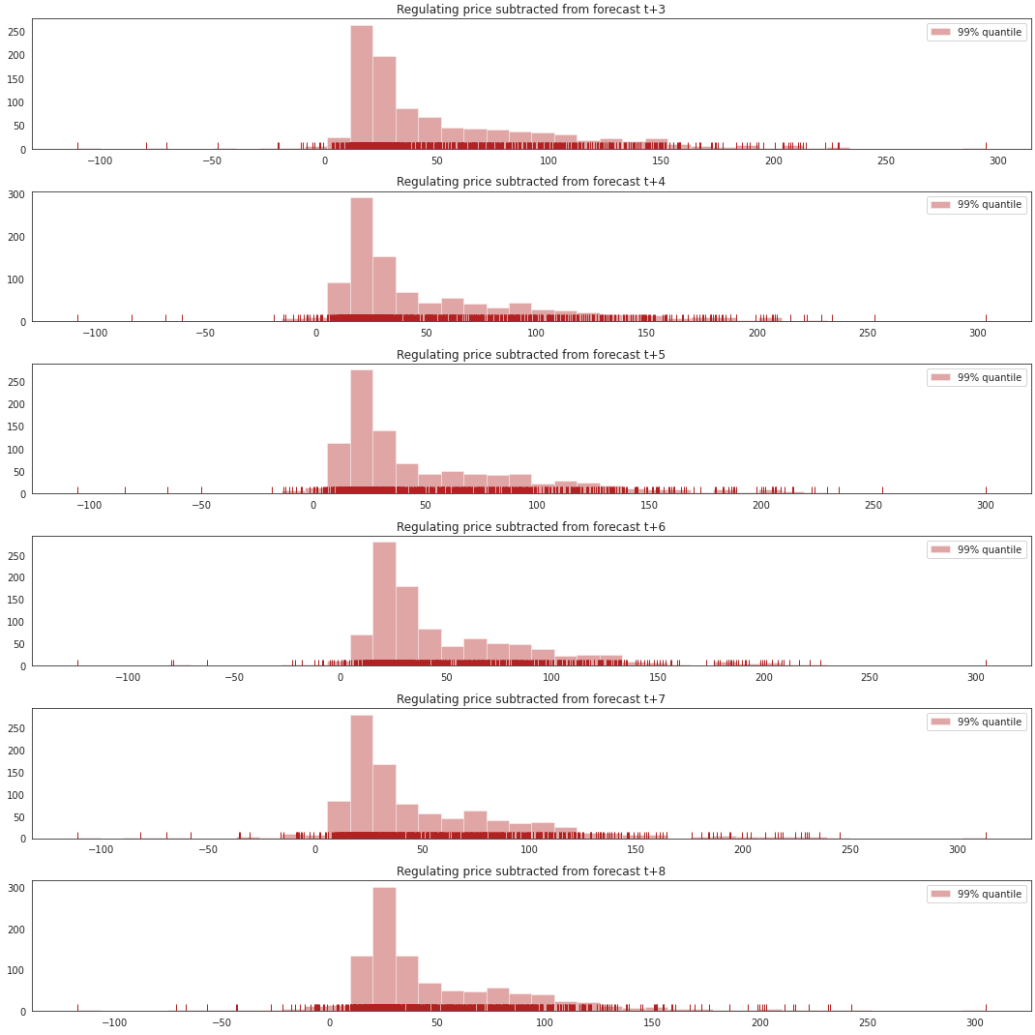

Figure A.11: Forecasting errors of the 99% quantile. Each histogram is a specific horizon in $t + 3h, \ldots, t + 8h$. The x-axis consist of bins. The bars on the y-axis count how many errors are contained in a bin. On the x-axis there are small vertical markers which indicate the location of an error.

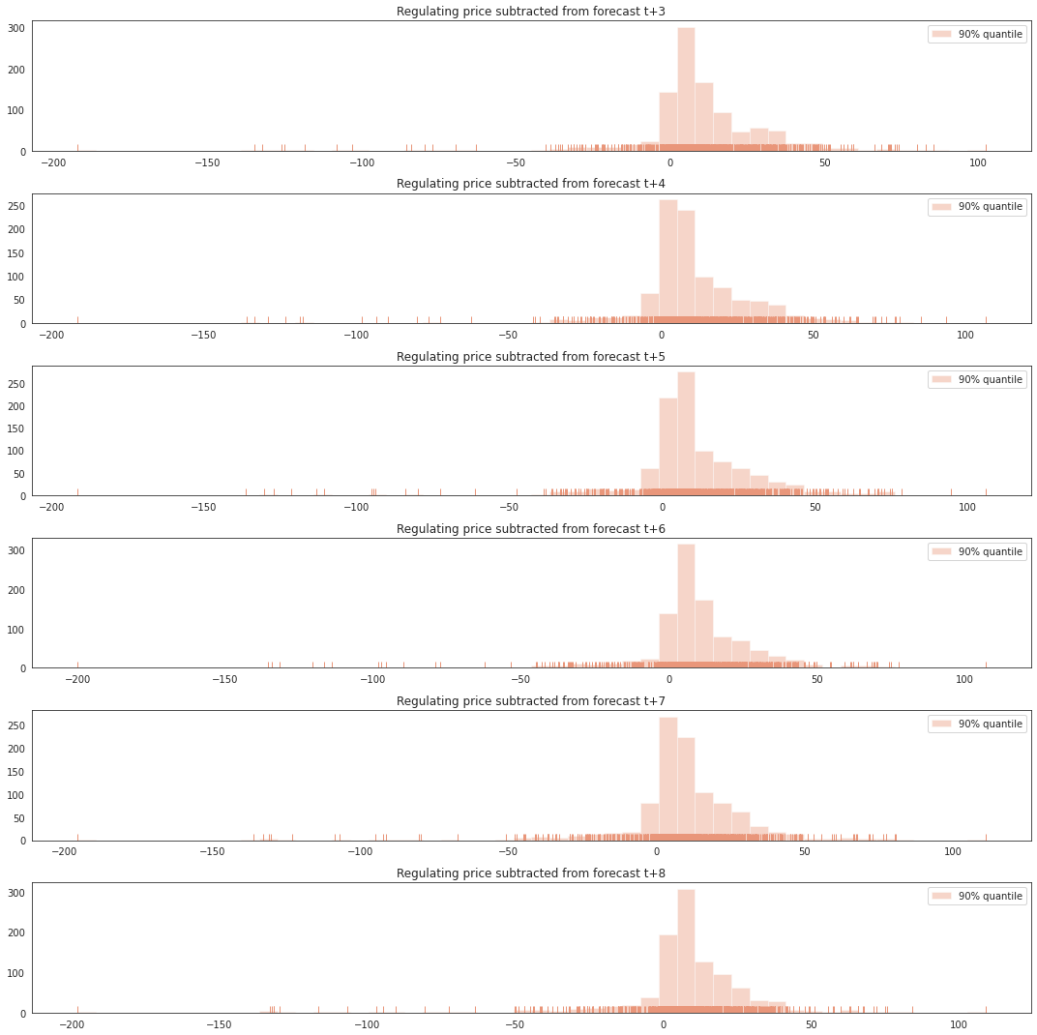

Figure A.12: Forecasting errors of the 90% quantile. Each histogram is a specific horizon in $t + 3h, \ldots, t + 8h$. The x-axis consist of bins. The bars on the y-axis count how many errors are contained in a bin. On the x-axis there are small vertical markers which indicate the location of an error.

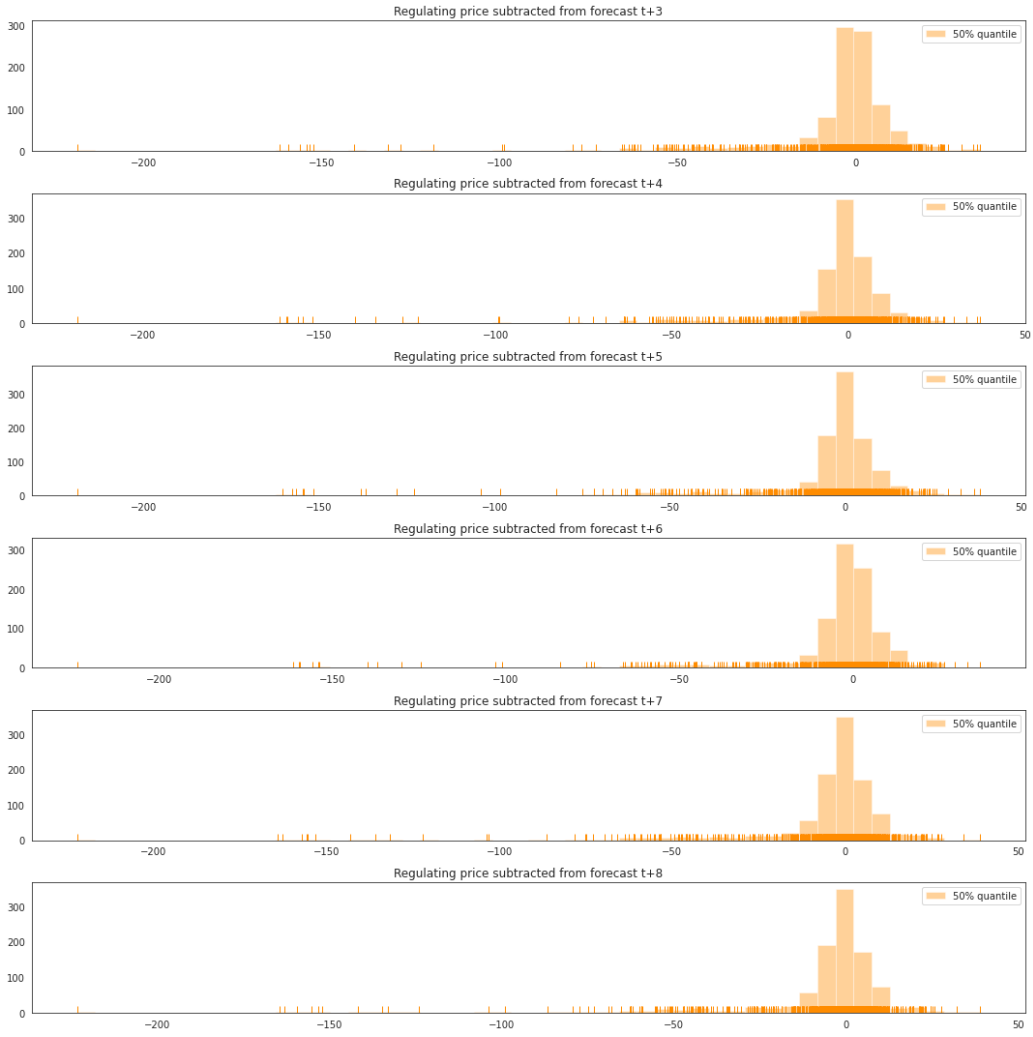

Figure A.13: Forecasting errors of the 50% quantile. Each histogram is a specific horizon in $t + 3h, \ldots, t + 8h$. The x-axis consist of bins. The bars on the y-axis count how many errors are contained in a bin. On the x-axis there are small vertical markers which indicate the location of an error.

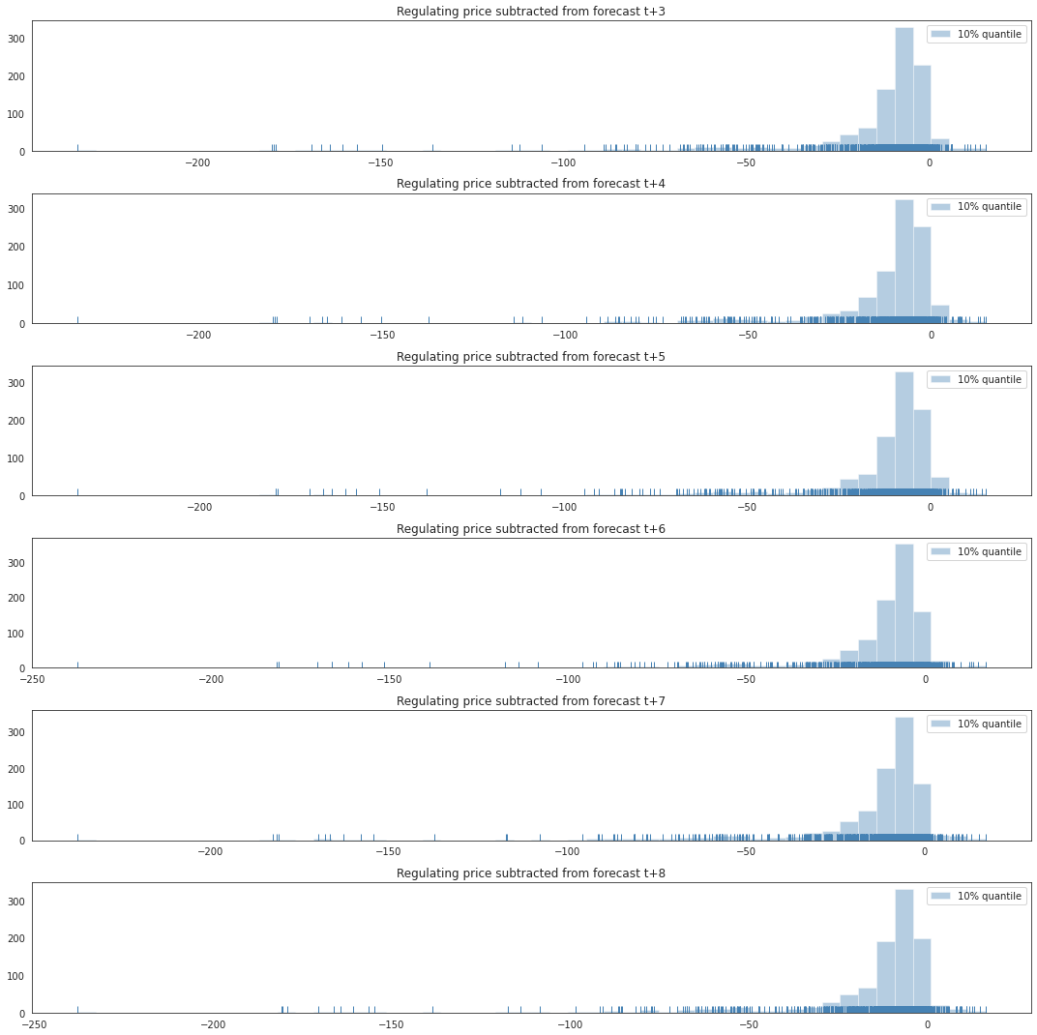

Figure A.14: Forecasting errors of the 10% quantile. Each histogram is a specific horizon in $t + 3h, \ldots, t + 8h$. The x-axis consist of bins. The bars on the y-axis count how many errors are contained in a bin. On the x-axis there are small vertical markers which indicate the location of an error.

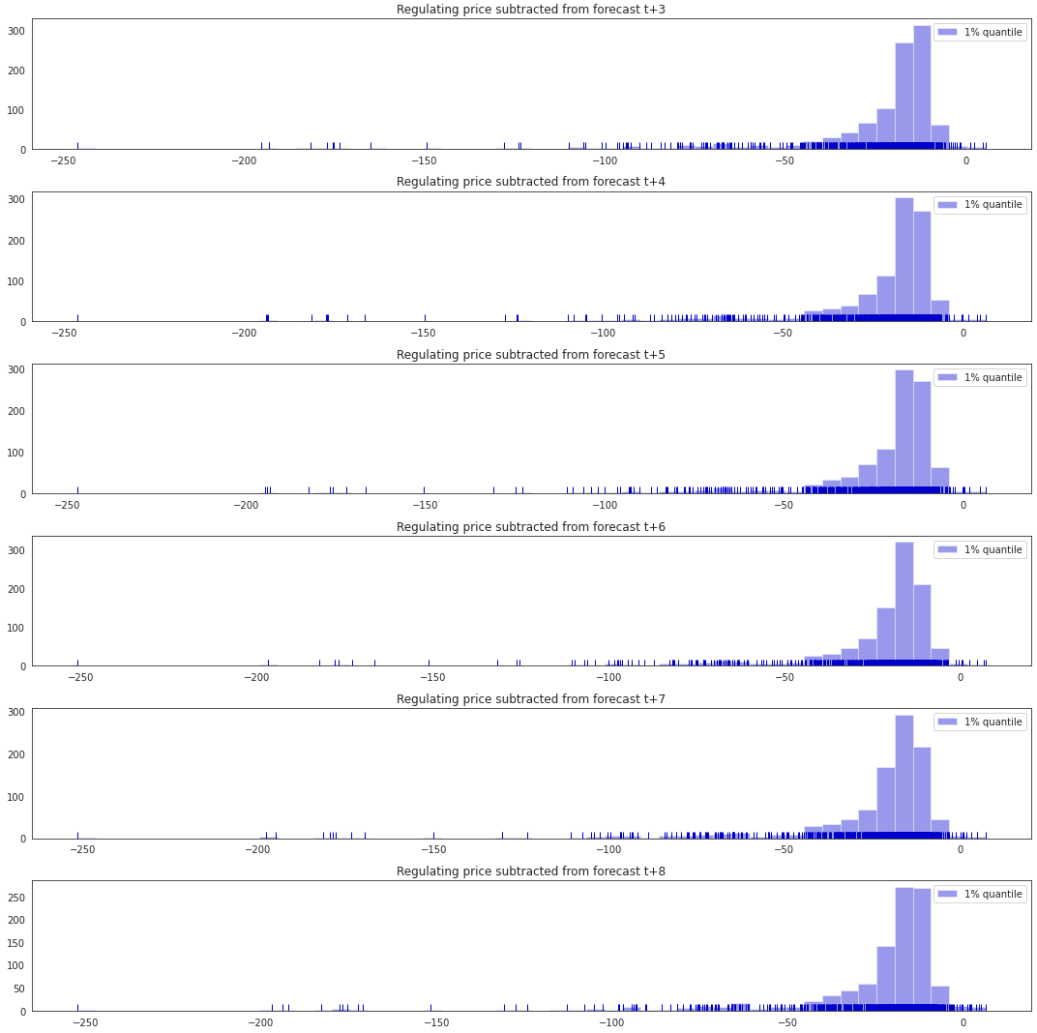

Figure A.15: Forecasting errors of the 1% quantile. Each histogram is a specific horizon in $t + 3h, \ldots, t + 8h$. The x-axis consist of bins. The bars on the y-axis count how many errors are contained in a bin. On the x-axis there are small vertical markers which indicate the location of an error.

Bibliography

[1] *A quantile definition for skewness*. URL: https://blogs.sas.com/content/iml/2017/07/19/quantile-skewness.html (visited on 05/06/2020).

[2] Yoshua Bengio. "Practical recommendations for gradient-based training of deep architectures". In: *CoRR* abs/1206.5533 (2012). arXiv: 1206.5533. URL: http://arxiv.org/abs/1206.5533.

[3] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *CoRR* abs/1406.1078 (2014). arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078.

[4] Junyoung Chung et al. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555 (2014). arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555.

[5] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. 2015. arXiv: 1511.07289 [cs.LG].

[6] *Electricity market explained by Fingrid*. URL: https://www.youtube.com/watch?v=dNU8p71p020 (visited on 05/07/2020).

[7] Alex Graves. "Generating Sequences With Recurrent Neural Networks". In: *CoRR* abs/1308.0850 (2013). arXiv: 1308.0850. URL: http://arxiv.org/abs/1308.0850.

[8] Thomas Hamfelt. "Forecasting the USD/SEK exchange rate using deep neural networks". Bachlor's thesis. Centre for Mathematical Sciences: Lund University, Aug. 2019.

[9] *Keras FAQ: Frequently Asked Keras Questions*. URL: https://keras.io/getting-started/faq/#why-is-the-training-loss-much-higher-than-the-testing-loss (visited on 05/02/2020).

[10] Roger Koenker. *Quantile Regression*. Econometric Society Monographs. Cambridge University Press, 2005. DOI: 10.1017/CBO9780511754098.

[11] Roger W Koenker and Gilbert Bassett. "Regression Quantiles". In: *Econometrica* 46.1 (1978), pp. 33–50. URL: https://people.eecs.berkeley.edu/~jordan/sail/readings/koenker-bassett.pdf.

[12] *Quantile Regression AU - Hao, Lingxin AU - Naiman, Daniel*. Thousand Oaks, California, 2007. DOI: 10.4135/9781412985550. URL: https://methods.sagepub.com/book/quantile-regression.

[13] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *CoRR* abs/1609.04747 (2016). arXiv: 1609.04747. URL: http://arxiv.org/abs/1609.04747.

[14] James W. Taylor. "A quantile regression neural network approach to estimating the conditional density of multiperiod returns". In: *J. Forecasting* (2000), pp. 299–311.

[15] *Time series forecasting*. URL: https://www.tensorflow.org/tutorials/structured_data/time_series (visited on 04/22/2020).

[16] *Understanding LSTM Networks*. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 04/23/2020).

[17] Ruofeng Wen et al. "A Multi-Horizon Quantile Recurrent Forecaster". In: *arXiv e-prints*, arXiv:1711.11053 (Nov. 2017), arXiv:1711.11053. arXiv: 1711.11053 [stat.ML].