TU WIEN Informatics

Zeitreihenvorhersage mit aufmerksamkeitsbasierenden Transformer-Modellen

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Computational Science and Engineering

eingereicht von

BSc Alexander Kund-Leitner
Matrikelnummer 1525882

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Associate Prof. Dipl.-Ing. Dr.techn. Clemens Heitzinger

Wien, 17. Jänner 2025

Alexander Kund-Leitner	Clemens Heitzinger

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Time Series Prediction with Attention-Based Transformer Models

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Computational Science and Engineering

by

BSc Alexander Kund-Leitner
Registration Number 1525882

to the Faculty of Informatics

at the TU Wien

Advisor: Associate Prof. Dipl.-Ing. Dr.techn. Clemens Heitzinger

Vienna, 17th January, 2025

Alexander Kund-Leitner	Clemens Heitzinger

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

BSc Alexander Kund-Leitner

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 17. Jänner 2025

Alexander Kund-Leitner

Danksagung

Als Erstes möchte ich mich bei meinem Betreuer, Herrn Professor Heitzinger, für seine Geduld, seine Zeit und die Fähigkeit, mich durch ein so kompliziertes Thema zu führen, bedanken. Er antwortete immer schnell auf meine Fragen und gab mir wichtige Hinweise. Meinen tiefsten Dank möchte ich meiner Familie aussprechen, insbesondere meiner Frau und meinem Hund. Es war immer möglich, meine Ideen und Gedanken mitzuteilen, und indem ich sie ihnen erzählt habe, habe ich mir meine Fragen schon meistens selbst beantwortet. Ich bedanke mich schon im Vorhinein, falls diese Themen der Arbeit von weiteren Personen genauer analysiert werden. Abschließend möchte ich mich bei allen Personen bedanken, die mich in den letzten Jahren unterstützt haben, um meinen Masterabschluss zu erlangen.

Acknowledgements

I would like to thank my supervisor Professor Heitzinger for his patience, his time and the ability to help to guide me through such a complicated topic. He responded to my questions always quickly and gave me important hints. I would like to express my deepest gratitude to my family, especially my wife and my loyal dog. It was always possible to share my ideas and thoughts and by telling them I always answered the questions myself. I extend my thanks to those participants who will share and work on my study to go deeper in this material and research. Finally, I would like to thank all the people that supported my over the last years to finally obtain my master's degree.

Kurzfassung

Diese Arbeit erforscht die Anwendung von Transformer-basierten Modellen zur Vorhersage von zwei sehr unterschiedlichen Datensätzen. Zum besseren Vergleich wurden zusätzlich zwei weitere Modelle erstellt und die Vorhersageergebnisse mithilfe eines statistischen Tests kontrolliert. Die geringsten RMSE-Werte wurden vom SARIMA-Modell erreicht, knapp dahinter folgten die Transformer-basierten Modelle. Ein weiteres zentrales Thema dieser Arbeit war die visuelle Untersuchung der Aufmerksamkeit des Transformers. Die Ergebnisse zeigen, dass es einen Zusammenhang zwischen der Anzahl der Aufmerksamkeitsköpfe und der Verteilung der Aufmerksamkeit gibt. Zusätzlich wurde eine Periodizität in der Aufmerksamkeit nachgewiesen, wenn diese bereits in den Quelldaten vorhanden war. Am Ende der Arbeit wurden diese Modelle in einer Handelssimulation mit realen Regeln getestet, bei der es darum ging, Aktien zu kaufen und zu verkaufen. Alle Modelle erzielten einen negativen Gewinn, was auf die Schwierigkeit derart nicht-linearer und komplexer Finanzdaten hinweist.

Abstract

This thesis explores the application of Transformer models for forecasting two different data sets. For a comparison, two other state-of-the-art (SARIMA and LSTM) models were created and compared with help of statistical tests. The results show, that the SARIMA-model achieves the lowest RMSE-scores for both data sets, while the Transformer performs close to it. Another big topic was to analyze the attention of the Transformer model by a visual inspection. Additionally, the results show that there is a relationship between the number of attention-heads and the distribution of the attention. They also show if there exists a periodicity in the source data there is also a measurable periodicity in the attention. At the end of the study tested these models were tested in a training simulation with real-world rules. All models produce a negative profit after one month of trading indicating the difficulty of such non-linear and complex finance data.

Contents

CHAPTER 1

Introduction

Time-series forecasting (TSF) remains a focal point within the scientific field of computer science, and many papers have been written on this topic. Many real-world problems, like the prediction of energy supply, forecasting weather, and forecasting financial trends, etc., can be defined as prediction tasks for machine learning. The basis for TSF is that a model learns a trend based on historical data to predict the future. Some models fix those windows to look-back for the prediction, while other models vary those windows based on the input. Traditional models like LSTM (Long Short-Term Memory) or ARIMA (AutoRegressive Integrated Moving Average) have been investigated enough to state that there are limitations to them. The limitations of the ARIMA model include the inability to handle multiple non-linear variables as input features and the non-constant standard deviation of those input sequences in real-world problems. On the other hand, the LSTM model suffers from its inability to quickly recognize and extract complex features, especially for a stock prediction task where multiple features have to be considered for an accurate prediction. A relatively new model type named the Transformer, initially used for Natural Language Processing (NLP), has emerged in the field of sequence data forecasting. With its attention mechanism, a revolutionary new way was opened to apply this to the time-series field. This attention mechanism is able to learn and memorize complex feature patterns and use them for prediction. The mechanism pays attention to all sequences, and the training is easy to perform in parallel. In contrast to LSTM, the Transformer is a model type where it is possible to analyze the mechanism of the actual prediction, whereas the LSTM or the ARIMA model acts like a black box. By visualizing the attention, it is possible to detect bias in the data to avoid it and make the prediction more general and better [19] [25] [15].
The motivation for the first chosen dataset, which consists of building energy data, is that in 2019, existing buildings consumed over 40% of the world's energy. Therefore, there is a big interest in minimizing this amount of energy with new, innovative ideas. A solution could be to build more and more smart buildings or upgrade existing ones.

These smart buildings are capable of learning the building's energy consumption history and training an efficient machine learning model based on the data collected by the sensors. Additionally, these resources will be used more efficiently, and the reliability will rise too. The energy consumption dataset provided continuous and detailed data with a strong periodic pattern, which allows for building an accurate model [16]. A good opportunity to generate the amounts of data needed to train these machine learning models is to use the Internet of Things (IoT). The sensors provide a solid foundation for collecting data to detect anomalies and predict the building's energy consumption over a period of time [6]. An energy prediction task could be used as a basis for a more accurate virtual building simulation. This would allow for testing various parameters of the building, such as heating, air conditioning, or lighting in the rooms. With these simulations, it is possible to find the best combination of those parameters to achieve a smaller total energy consumption for the building.
On the other hand, getting a passive income in addition to the normal working salary is a goal many employees want to achieve and perhaps one day quit their jobs. Automatic trading bots could be one option to earn a lot of passive income, since the bots work 24/7. There are many approaches to that, such as different trading strategies or using complex machine learning indicators that have good precision in making those trading decisions. Scalping is a trading strategy where the time frame is set to minutes and hundreds of trades are made per day. For those trades, a very small profit per trade is produced, but in the end, a substantial amount of money could be produced with this method over time [26]. These machine learning models also have the potential to forecast not only the decision on when a trade will be executed but also the future trend of the price, helping traders find the optimal point to buy or sell a position [13].
In this thesis, an attention-based Transformer model is tested on its prediction power for two different time-series datasets. For comparison, traditional models will be created as well to compare the prediction power. To better understand the attention mechanism, multiple attention visualizations will be presented. Additionally, for the second dataset, which consists of Bitcoin close prices, a trading simulation will be created where the model decides whether to invest or not over a very short time. Due to its high volatility, the Bitcoin dataset provided a high difficulty for reading patterns and predicting prices. Additionally, the influence of market events, other external factors, and rapid price changes make this data the perfect choice for training a difficult time-series model.

1.1 State of the Art

Jesse Vig and his team developed a visualization tool for a multi-head attention provided by the BERT and the OpenAI GPT-2 models. The attention-head view shows patterns based on the layers of the attention for different input sequences for word embedding. They also present two other use cases, such as the model view and the neuron view. The model view has one important use case, such as identifying recurring patterns. This is done by looking at the attention across all heads from a bird's-eye perspective [24].
The topic of long range forecasting with multi-variable time series data investigates

Jake Gribsby and his team for multiple types of data, such as weather or traffic light information. They solve those problems with a new "spatiotemporal sequence" method where the Transformer gets an input token that represents a single value. [9]
One of the first attempts at a time series forecasting with Transformers presents Neo Wu and his team by using influenza-like illness (ILI) for a case study. They develop a general Transformer-based model for time series forecasting. By comparing their results with other models like LSTM, ARIMA, and Seq2Seq+attn models, the Transformer is a new competitor by achieving the lowest RMSE score at their tests. These papers covered promising results from a Transformer-based models but not fully any relationship between periodic patterns in the data and the attention mechanism itself. To fill this gap, this thesis explores how the attention responds while varying input parameters of the model [25]. This was the first time that a Transformer-based model could effectively translate text into another language. By introducing the self-attention mechanism, which is the basis of many papers for the future [23]. To extend the idea of visualizing attention, this thesis investigates this approach on time-series data. With this, the attention could be analysed more to see how it reacts based on the given input sequence from the Transformer and possible patterns in it. If there exists a periodicity in the attention [14] or how the attention reacts to multiple features as input sequence or simply how the attention distributes for a variation in the number of input channels, referring to the number of heads for the Transformer or analyse a possible connection between a visualization of attention and a forecasting task [24]. On the other hand, how the actual prediction performs based on the variation of the number of heads will also be investigated for two different datasets [23, 20].

1.2 Research Questions

1.2.1 Predicting Building Energy Consumption

RQa1: Is a multivariable attention-based Transformer model capable of predict building energy consumption?
This question aims to explore the predictive power of the Transformer model by changing the following hyperparameters: input decoder length and number of heads. The different input lengths are [24, 48, 72, 96, 168], which represent he time unit 24 hours and the other, respectively, and the different numbers of heads are [1, 2, 4, 8]. For each pair of the two hyperparameters, a model is going to be trained and tested over a given time period. The predicted sequence length is always 169 hours, which represents a whole week of energy consumption prediction. The metric used to analyzethe prediction will be the root mean square error (RMSE).
RQ1b: Which one of the following models (LSTM, SARIMA or Transformer) can best predict energy consumption?
This question tests the upper three named models on the same time period as described in **RQ1a**. The metric used to measure the error between the predicted and true values is the RMSE for all three models. The number of heads used to train the Transformer will

be chosen based on the result in **RQ1a**. All three models will be tested with multiple input lengths, such as [24, 48, 72, 96, 168], and the results will also be tested for their statistical significance.

1.2.2 Predicting the Close Price of Bitcoin

RQ2a: Is a multivariable Transformer-based model capable of predicting the close price of Bitcoin data?
The input range for this question will be set to [24, 48, 72] in comparison to the model parameters in **RQ1**, but the number of heads will still be the same. The test will be done for the three models mentioned in **RQ1a** with the same metric.
RQ2b: Which one of the following models, LSTM, SARIMA or Transformer can predict the close price of Bitcoin the best?
The process is the same as shown in **RQ1b**, this time for the Bitcoin close price dataset.
RQ2b: Is a machine learning model profitable to use as a trading instrument?
Only the models with reasonably good predictive performance will be considered for this question. The trading rules will be taken from a real-world trading platform, and a backtest will be performed by analyzing the evolution of a virtual wallet. At the end of the question, a detailed profit analysis will be presented, and the calculation of the value of risk over multiple time frames will be shown.

1.2.3 Visualizing the Attention for Time Series Data

RQ3a: How does the attention react to the additional features?
The aim of this question is to find a way to illustrate the attention from a Transformer model with the actual input sequence. Then the attention will be measured in the area of the appearance of the additional features and the relative percentage calculated. This percentage value is compared with the model results with no additional features.
RQ3b: How does the attention react to a strong periodicity in the input sequence? The calculation of the attention is the same as in **RQ3a**, in addition, the attention pattern is compared to different input sequences with a high periodicity, in this case a weekly pattern from building energy data. This will be done and analyzed for a large number of heads. This time, the R^2-score is calculated by comparing the attention and the postulated weekly pattern.
RQ3c: How does the attention distribute over a variation in the number of heads?
This question aims to analyze the impact of the different numbers of heads on the distribution of the attention itself. The metric is the standard deviation and measures how evenly the attention is distributed.

1.3 Overview

In the next chapter, the Transformer will be explained in detail. Also, the way from the first appearance of attention to the generalized attention used in this thesis.
The results chapter will first show the data analysis and the data preparation from the

two datasets that were used later, and then a detailed description of the results from the research questions defined before. Following this is the discussion part, where the findings from the results are discussed in detail. Finally, in the conclusion part, the most important things will be summarized shortly.

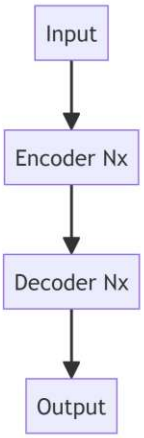
CHAPTER 2

Methods

In this chapter, the most important methods will be explained to form the Transformer architecture. Especially the individual parts will be investigated closer such as Encoder, Decoder, Attention-Mechanism, Positional Encoding, Multi-Head Attention and Masked Multi-Head Attention. Additionally, the process will be explained which dimensions the individual parts of the Transformer expects

2.1 Transformer Architecture

. The Transformer is a sequence-to-sequence model mostly used on natural language processing NLP tasks for translation. Before the Transformer was discovered, a RNN-based architecture was used for NLP tasks. A RNN processes a sentence sequentially and passes word by word into the system for training. The input is named Encoder, and the output part is named Decoder. The model creates a context vector, which contains the meaning of the sentence. The downside of this method is that for long sentences, the model cannot handle the context, and for more complicated sentences, it is hard to figure out the translation [4]. In contrast, the Transformer processes the input and the target sequence as a a whole sequence, which makes it easier to do it in parallel. With this procedure, larger datasets could be used in a reasonable amount of time to train the Transformer. The Encoder and Decoder blocks could also be stacked a couple of times, where every layer recognizes a different meaning of the input. It has been shown that these layers discover the NLP pipeline. The individual layers focus on the part of speech text, semantic roles, or co-references [21] [10].



(a) Transformer flowchart

The architecture is separated into four different parts, namely Input, Encoder, Decoder and Output. The Nx represents the number of Encoder and Decoder blocks which can be referred to a number of layers. The output from the first Encoder layer is the input from the second Encoder layer sequentially and so on until the number of total layers is reached vice versa for Decoder part. The number of layers is also a hyperparameter of the model. All Transformer based models in this thesis were designed with four layers.

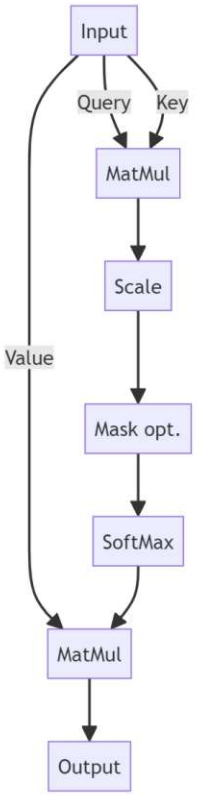
2.2 Attention

The attention mechanism was first inspired by human visual processing system where the brain focuses more on the words it actually reads in comparison to the other words in its vision. This was done by defining weights to each element of a text input to a model. With this, the model will be guided to critical steps for a translation task.
The first appearance of the attention mechanism was proposed by Bahdanau et al.(2014) to solve a bottleneck problem where a Decoder had limited access to the information provided by the input. His attention mechanism consists out of these important parts namely: hidden Decoder states s_{t-1}, where t is a time step, the context vector c_t, the annotation h_i which captures the important information from the input text, the weights $\alpha_{t,i}$ which are assigned to each time step t, a annotation h_i and the attention score $e_{t,i}$ calculated with a given alignment function $a(x, y) = \tanh W[x; y]$. A bidirectional RNN was used in the Encoder part to generate the h_i in both directions to capture both perspectives of the words. On the other side the Decoder produces target words focusing on the most relevant information from the input sequence. To get the alignment scores a alignment function is used with the hidden Decoder state and the annotation $e_{t,i} = a(s_t - 1, h)$. By applying a soft-max operation on the previous computed alignment scores $e_{t,i}$ the weights are calculated $\alpha_{t,i} = \text{softmax}(e_{t,i})$. Finally the context vector c_t which is unique is computed by forming the weighted sum of all Encoder hidden states T $c_t = \sum_{i=1}^{T} \alpha_{t,i} h_i$. At each time step this context vector c_t is fed into the Decoder. By

reformulation the attention mechanism could be applied to any sequence-to-sequence task [4].

2.2.1 Generalisation of Attention

There are three main components to generalise attention namely queries Q, keys K and values V.
Queries: They contain the context or information needed to compute attention. In NLP tasks, queries are usually input tokens or embeddings, which represent words for a translation task. For a prediction task, mainly these queries will represent energy values or price data.
Keys: Keys have the same type of data as the queries and are used to match the content with the queries. Based on the similarity, keys help to calculate the attention score by determining the relevance of queries to them.
Values: The actual information or content is contained by the values and were associated with the keys. To calculate the weighted averages the values are used. These averages capture the relevant information from the values corresponding to the queries. In comparison to the attention mechanism from Bahdanau et al. the queries refer to the previous Decoder output s_{t-1}, while the values and the keys refer to the Encoder inputs h_i. In the paper "Attention is all you need" they refer to this calculation as a scaled dot-product attention. The queries, keys and values are calculated as follow. Take the input sequence and multiply it with the initial weights W sets by the model.
$Q = XW^q$ $K = XW^k$ $V = XW^v$ where X is the input sequence after applying the positional encoding, W^j are the weights with $j = (q, k, v)$ representing the individual types quires, keys and values. W_j has the dimension of $d_{\text{model}} \cdot d_{\text{model}}$. The resulting Q, K, V have the following dimensions $s_{\text{in}} \cdot d_{\text{model}}$ where s_{in} represents the input sequence length and d_{model} the dimension of the model.
The following steps needs to compute the general attention:
First calculate the attention score $\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V$. To compute the alignment scores $e_{q,k} = QK^T$ Then the score is divided by the square root of the d_{model} for scaling and numerical stability. A softmax function is applied to form the weights. The weights are then multiplied with the values to form the attention matrix. The attention matrix has the following dimension $s_{\text{in}} \cdot d_{\text{model}}$ [18] [23].



(a) Attention flowchart

2.3 Input

The input part of the Transformer mainly prepares the data for the training as well as for the prediction task later. For a prediction task with pure numbers, input/output embedding or tokenising is not needed due to the leak of using words. For a NLP task, first the sentences would be tokenised by separating the sentences into words or even smaller or larger tokens. Then an embedding layer transforms these words into vectors for the model.

2.3.1 Data Indexing

Suppose a sequence S of distinct numbers with a timestamp and the length of N is given. The next step is to separate it into an input sequence and an output sequence. The input sequence length sets how long the Transformer is able to look back in the past to predict the future. The output sequence lengths sets how long the Transformer could predict the future of the sequence. The sequence step size s_{st} sets how many points the Transformer predicts in each prediction step. Suppose the input data has an hourly resolution and the three parameters were set like this [24, 168]. That means that the Transformer gets

the past 24 hours to predict the next hour in one prediction step with a prediction length of 168 hours which represents one week. For the experiments, these parameters were changed to test the Transformer how it predicts the output [23].
With these parameters a vector is created which helps to separate the values form the input sequence and creates a tensor for the input of the Decoder and the Encoder. This algorithm creates pairs of indices based on the three parameters where the distance between the pairs is always the input sequence s_{in} with respect to the step size s_{st}.

Algorithm 2.1: Indexing

Input: Length of sequence N, window size w, step size s_{st}
Output: List of pairs of indices y

$sub_f \leftarrow 0$;
$sub_l \leftarrow w$;
$y \leftarrow []$;
while $sub_l \leq N - 1$ **do**
    $y_n \leftarrow (sub_f, sub_l)$;
    $sub_f \leftarrow sub_f + s_{\text{st}}$;
    $sub_l \leftarrow sub_l + s_{\text{st}}$;
end
return y_n;

With this list of pairs the input sequence S is then divided into smaller sub sequences namely source, target and predict target. The source is the Decoder input, target is the Encoder input which is shifted by one against the source and predict target is the model output important for the computation of the loss.
After applying the provided Dataset class from Python to the three sequences the final dimension of those are batch size, input sequence length and number of features which will be changed trough the different experimental approaches [3].

2.3.2 Positional Encoding

Positional Encoding takes care of the order of the input sequence for the Transformer. Without it the model could not identify similar words or values with a different order in the input sequence for the training. Doe to that the training is done in parallel the positional encoding explicitly defines a order in the input sequence. It is then added to the input sequence before the Encoder input. There are two important rules that the positional encoding must fulfill: [23]
First, values have not to be too large because then the actual value looses the semantically meaning for the model. Secondly every position have to have the same identifier in respect of the sequence. While the sequence change the values of the positional encoding stay the same. Linear functions will not suit to this purpose doe to their lack of boundaries. On the other hand sin and cos functions are bounded in an interval of -1 to 1 and are also be periodic. In the original paper "Attention is all you need" they used this functions

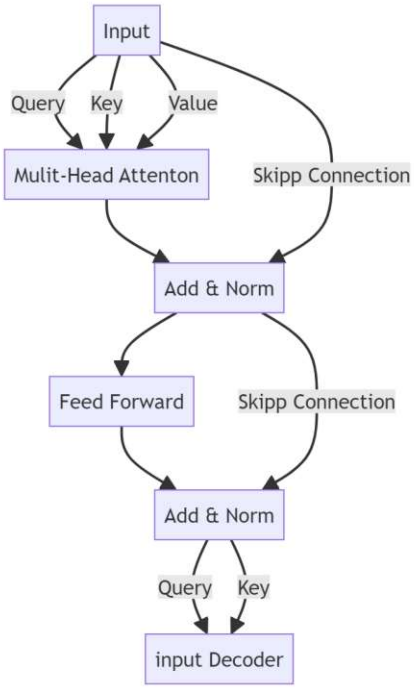
with different frequencies and this technique is also used in this work [23].

$$L := \begin{cases} PE_{(\text{pos},2i)} = \sin\left(\frac{\text{pos}}{10000^{\frac{2i}{d_{\text{model}}}}}\right), \\ PE_{(\text{pos},2i+1)} = \cos\left(\frac{\text{pos}}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \end{cases}$$

The variable pos represents the position and i the dimension. The wavelength ranges from 2π to $10000 \cdot 2\pi$. Where the even positions correspond to a sine function and the odd positions to a cosine function. Thus, the first values in the sequence have the largest frequency of the wave function increasing with the position in the sequence. With this unique values the positional encoding is created [23].

2.4 Encoder

The main task of the Encoder is to turn the input sequence into a machine-readable representation. In total there are three components a Decoder has which are Add&Nome, Feed Forward and Multi-Head Attention.



(a) Encoder flowchart

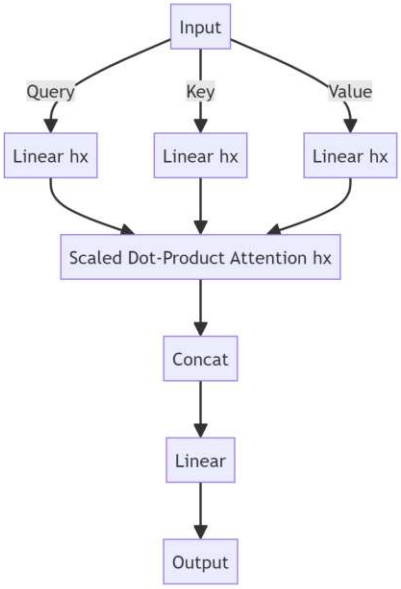
The input of the Decoder is the output from the positional encoding which then is fed into the Mulit-Head Attention block where all three inputs are the same. This indicates as the self-attention mechanism.

2.4.1 Self attention

Self-attention refers to the attention calculation shown in the upper part but all inputs are the same. Attention(X, X, X), where X is a input sequence. This process allows that the model learns relationship between all pairs of the input sequence.

2.4.2 Multi-Head Attention

In the original paper "Attention is all you need" they stretch the idea further and divided the d_{model} dimensional queries, keys and with values into h attention heads. On each



(a) Multi-Head Attention flowchart

head a separate attention calculation is done in parallel. Then these attention heads are concatenated and multiplied with the overall weights W_O. MultiHead$(Q, K, V, \text{head}_\text{i}) =$ Concat$(\text{head}_1, ..., \text{head}_\text{h})W^O where\ \text{head}_\text{i} = \text{Attention}(Q_i, K_i, V_i)$ where $Q_i \in R^{d_{\text{model}} \times d_Q}$, $K_i \in R^{d_{\text{model}} \times d_K}$, $V_i \in R^{d_{\text{model}} \times d_V}$ and $W^O \in R^{hd_v \times d_{\text{model}}}$ [23]. For simplicity the dimension for quires, keys and values were set $d_K = d_V = d_Q$. For the experiments the model was trained with different numbers of heads from the set of heads $H = [1, 2, 4, 8]$ to see how the output reacts to the different number of input channels [11].

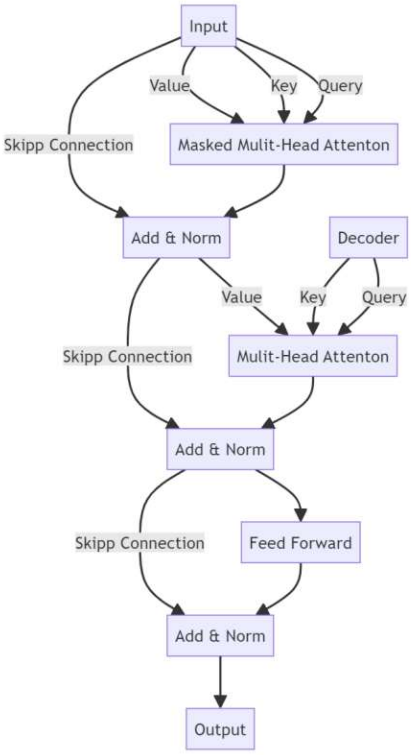
2.4.3 Procedure

Assume the Decoder is initialized with the following parameters such as the model dimension d_{model}, the input sequence length s_{in}, the target sequence s_{tar}, the number of features f and a batch size b. With the preparation from the input part of the Transformer the input from the Decoder D_I has the following dimension: $[b, s_{\text{in}}, f]$. First a linear layer maps D_I to the following dimension: $[d_{\text{model}}, s_{\text{in}}, f]$. This input is then

put in the Multi-Heads Attention where three different weight matrices where multiplied to the same input to form the matrix Q', K' and V'. These weights are the learning parameters that the model changes at the training and they have the same dimension described above [24]. The Multi-Head Attention splits the three matrices into the number of heads. Important to mention is that the length of the dimension of the model must be a multiple of the number of heads. An additional weight is added which holds the weights of all matrices for the multi head attention. After that the output is added with the skip connection and then layer normalized. Finally a feed forward block is implemented by two linear modules with a ReLU activation function in between and the output is normed again [23].

2.5 Decoder

The difference to the Encoder is for the Decoder a Masked Multi-Head Attention mechanism is used for the input of the target sequence input. The queries and the values were taken from the output of the Encoder and incorporated with the Decoder.



(a) Decoder flowchart

2.5.1 Masked Attention

For the masked Attention a mask is added to the attention calculation. The mask is incorporated in the original attention equation like this

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T + \text{Mask}}{\sqrt{d_{\text{model}}}}V\right)$$

The mask itself consists of a matrix with the same dimension as the query, key and value matrix and has zeros on the diagonal and the lower parts of the matrix where the upper part is set to $-\infty$. After applying a softmax to the $-\infty$ entries these entries are then set to 0, which indicates they are ignored by the model [5].

2.5.2 Procedure

The input for the Decoder is also defined and is the input shifted to the right by the step size. In this case, the dimension is defined as $[b, s_{\text{tar}}, f]$, where s_{tar} is the target sequence length, after applying the target positional encoding. The target Decoder layer maps this input to the following matrix dimension $[s_{\text{tar}}, d_{\text{model}}]$ and the three different weight matrices are also applied for the target input. The next Multi-Head Attention block has different inputs where the queries and the keys are taken from the Encoder and the values from the Decoder. This configuration does not satisfy the condition of a self Attention mechanism because the inputs were not the same. After that two norm layers were added with a skip connection and another Feed Forward block is applied [23].

2.6 Output

The output consist of a linear layer and a softmax which calculates the probabilities of the next value for the prediction. During the training the applied weights were updated via back propagation using in this work the Mean Squared Error Loss $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$. For the training a separate vector is stored with the true values to have a reference value for the loss function. Based on that all weights are updated and the model learns the train-set [8].

CHAPTER 3

Results

3.1 Building Data Analysis

For the energy prediction task, an open dataset provided by Kaggle [17] is used. It consists of 507 non-residential buildings, containing energy consumption in kWh over one year with a measurement frequency of one hour. For each building various metadata such as building area, weather and property use type are provided. The buildings are distributed across various time zones, building types such as office, college, or dormitory. For the future work, one office building is selected in the New York, USA (GMT-4) time zone.

3.1.1 Raw Data Analysis

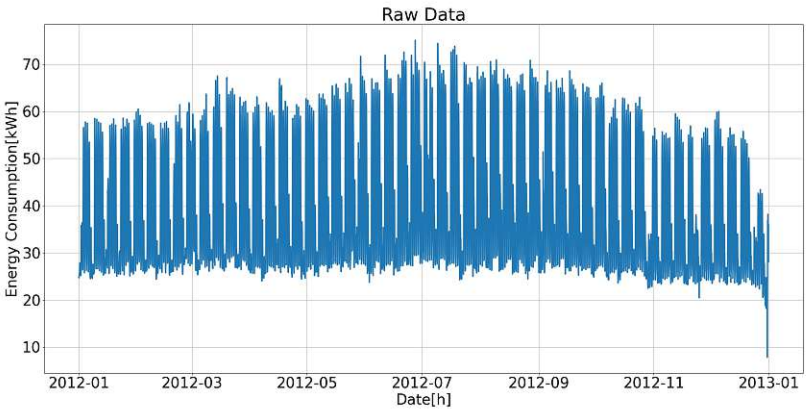


Figure 3.1: Raw data plot energy dataset

In this plot the raw data from the chosen building, Eddy, is shown. The date range goes from January 2012 to January 2013. In total there are 8,784 data-points which represent the energy consumption of the building in hours. The last few points will be filtered out for the training for the models because of the sudden drop in the energy consumption. After filtering the data consists out of 8,593 points. A first rough statistically analysis shows that the mean value is 39.73 with a standard deviation of 13.94, showing a stable trend over the year. The total data range is now from January 1, 2012 to December 24, 2012.

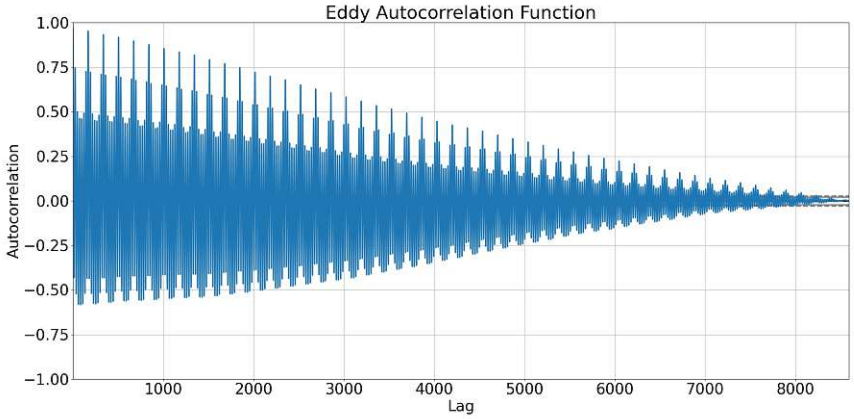


Figure 3.2: Autocorrelation plot energy dataset

To check if there is a weekly pattern in the data which is possible for a energy data like tend an auto-correlation plot is made to analyze the data. The x-axis represents the number of data-points and the y-axis represents the auto-correlation. The cone-like shape represents a strong temporal pattern in the data while the positive peaks indicate a positive correlation and the negative peaks indicate negative correlation.

3.1.2 Data Preparation

The raw-data will be split into three parts. The first part will be used as training-data with a date range from January 1, 2012 to September 30, 2012 for the models, the second part will be used as test-data with a date range from October 2, 2012 to October 21, 2012 and the third one will be used as validation-data with a date range from October 8, 2012 to December 24, 2012. The following holidays in the USA will appear will be appear in the training-data such as New Year, Martin Luther King Day, Memorial Day, Juneteenth, Independence Day, Labor Day and Veterans Day. These days have a different energy consumption trend and to avoid confusing the models training they will be filtered out from the training-data. An additional feature will be added to the dataset which indicates the model when a weekend is or not. This is implemented by defining a signal with the following conditions. The signal is -1 for a regular day during the week and the 1 on the weekend. For the training the training-set will be scaled from -1 to 1. With

this chosen scaler, the change from a weekday to the beginning of the weekend and vice versa exhibits the most significant difference.

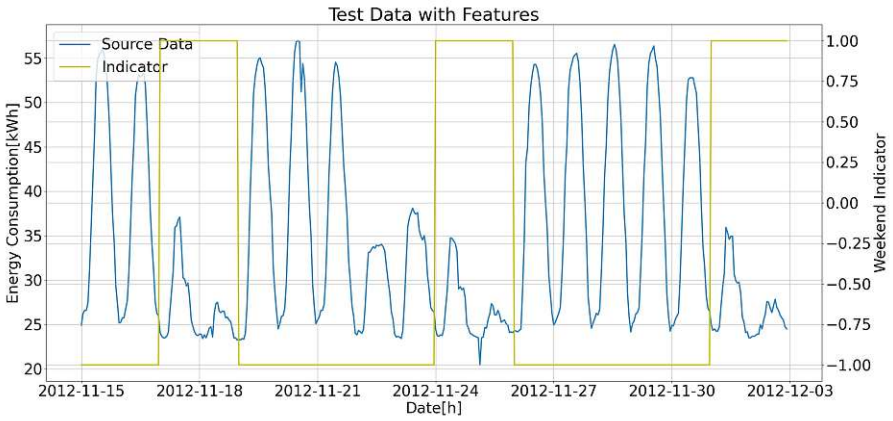


Figure 3.3: Feature plot energy test dataset

In this plot the test-data with a date range of two and a half weeks is plotted. Here the weekly energy pattern is clearly shown with two days with a smaller and five days with a higher energy consumption. The last week follows the clear trend but the second week not. On Thursday and Friday there is much smaller energy consumption value which indicates an anomaly in the pattern. The yellow line indicates the weekend indicator and serves as an additional feature. With this indicator two different models will be created one named Basic and the second one Weekend. The indicator should help the model to identify which weekday it is to improve the prediction performance.

3.2 BTC Data Analysis

The second dataset contains prices from the cryptocurrency Bitcoin (BTC) provided by Kaggle [12]. This dataset is chosen to have a counterpart to the first one. This dataset was chosen to have a counterpart example to see how the Transformer reacts to clearly non-periodic data. The close price column is chosen as a target for the prediction.

3.2.1 Raw Data Analysis

The price measurement frequency for that dataset is one minute and the time range from 2021 was chosen. In total there are over 610,782 data points with the following columns: date, open, high, low, close, Volume BTC and Volume USD. The date column contains values with date-times in a minute time frame, additionally,there are four columns which represents all relevant prices and the last two columns are the volume data for each minute. The values for this range is complete and there are no missing values. A short statistical analysis gives that the mean close price is 46,482.57€, the median price is 46,667.29€ and

the standard deviation is 9,454.11€. The similar median and the mean value indicates a symmetry in the data which we could see in the next plot. For completeness the minimal price is 28,073.03€ and the maximal price is 69,000€.

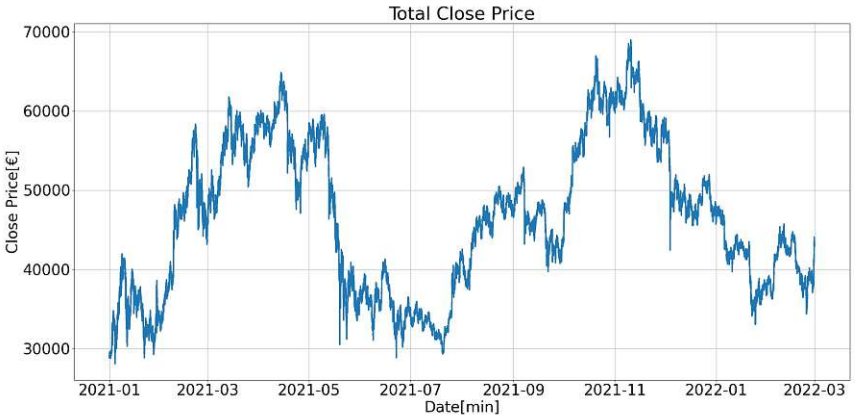


Figure 3.4: Raw data plot BTC dataset

This plot represents the close price for the total time-range for the year 2021. There definitely exists a movement in the market over the chosen time-range with two peaks and three floors. The symmetry line over the whole range is roughly located in the middle of the range.

3.2.2 Data Preparation

The total close price range will be separated into three parts namely train, test and validation dataset. The training dataset goes from January 1, 2021 to November 1, 2021 so the total data gathered data goes over the period of 11 months. This dataset will be used to train the different models. The test-data contains prices from one month after the train dataset without any gap. It is used to test the models at the training to measures the training performance. Finally the range from the validation dataset goes over two months to the end of the total dataset. This part is used to perform a back-test to test the models with data that the model has never seen before. The prediction target for the BTC close price of all prediction tasks in this thesis will be 10 minutes.

Figure 3.5: Raw data plot BTC price changes

Fixing the target length of the prediction to 10 minutes the next plot will analyze the close price change in this defined time range. This plot represents the price change from the whole dataset. On the right-hand side of the plot there are the positive price changes and the negative price changes on the left side. On the y-axis the count of the different price changes is shown and on the x-axis the price change range is shown. The distribution from the train-set to the validation-set is nearly equal, so the choice of the validation-set is valid.

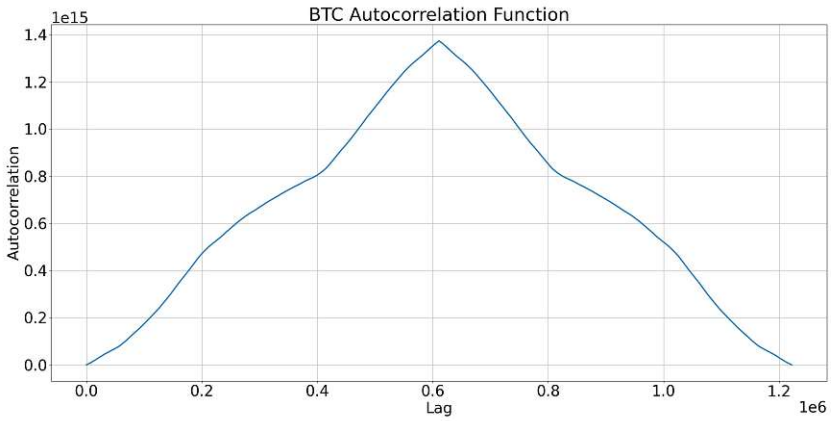


Figure 3.6: Autocorrelation plot BTC dataset

In this plot the auto-correlated function is shown to get further information about the data. The fact that there is only one maximum, is a indication that there exists a strong dominant periodicity or a cycle. There appears a time lag where the price data is highly correlated. The auto-correlation is positive, this means that if the price is high or low at a specific time it tends to follow by periods. The asymmetry of the auto-correlation plot

indicates a trend or seasonal periodicity in the data. In comparison to the dataset used before the periodicity is only limited to the total time-range and not on the input from the model.

3.2.3 Feature Engineering

Two additional features will be added to the dataset derived from the date column, the first one acts the same as shown in the energy building dataset namely weekend indicator. The second feature indicates when the market is open which is on every working day in a week from 08:00 to 16:00. Each indicator was implemented the same way -1 if the market is closed and 1 if the market is open and -1 for a normal working day and 1 on the weekend. The additional features derived from the close price column encompasses three distinct Exponential Moving Averages (EMAs), a buy signal, and a sell signal. The EMAs were calculated by considering the most recent 5, 10, and 20 price data points, respectively. The buy and sell signals follow the defined criteria

$$\text{Buy signal} := \begin{cases} 1, & \text{if EMA5} > \text{EMA10 and EMA10} > \text{EMA20}, \\ 0, & \text{else}, \end{cases} \tag{3.1}$$

$$\text{Sell signal} := \begin{cases} -1, & \text{if EMA5} < \text{EMA10 and EMA10} < \text{EMA20}, \\ 0, & \text{else}. \end{cases} \tag{3.2}$$

These rules define a trading strategy known as the 3EMAs strategy. Due to the 1 minute price frequency the parameters to calculate the three EMAs were also chosen to be short. The common values for the 3EMA trading strategy for hourly prices are 50, 100 and 200. These additional features were incorporated into the models to provide them with temporal price movements and relevant information to perform an overall better prediction for the future price.

Figure 3.7: 3EMAs trading strategy

This plot shows a small segment of the overall close price data. In the plot, buy signals are represented by green arrows, while sell signals are denoted by red arrows. Shortly after the cross-section of the three EMAs the chosen strategy works more efficient and the buy respectively the sell signals indicate the predicted price movement more accurately.

3.3 Energy Prediction

The source code for all Transformer-based models was provided by the following link [22], all SARIMA models from this link [1], and all LSTM-based models from this link [2]. In this section the different models are going to be tested on the energy validation dataset. For all tests with the energy dataset a length of 168 hours is chosen which represents the next week from the input sequence values. The basis to indicate how the model performs the RMSE score is calculated by comparing the predicted energy for the next week with the true values. A time shift window of 24 hours was chosen to gather multiple results for the statistical tests. After prediction the energy the window for the input for the models is shifted by 24h until the end of the validation dataset is reached. Additionally the tests will show how the different models react to the different day combinations for the input sequences.

3.3.1 Transformer Varying Number of Heads

In this section, the results will be shown while varying the number of heads for the Transformer model and the following hypothesis is formulated. By varying the number of heads for the Transformer model a notable periodicity in the prediction performance is noticed. In order to check the periodicity the RMSE score is calculated for the test range and plotted for every model configuration to find the postulated patterns. Only results from the Weekend model will be shown via a boxplot, the results from the basic model will be displayed with a table with the most important scores.

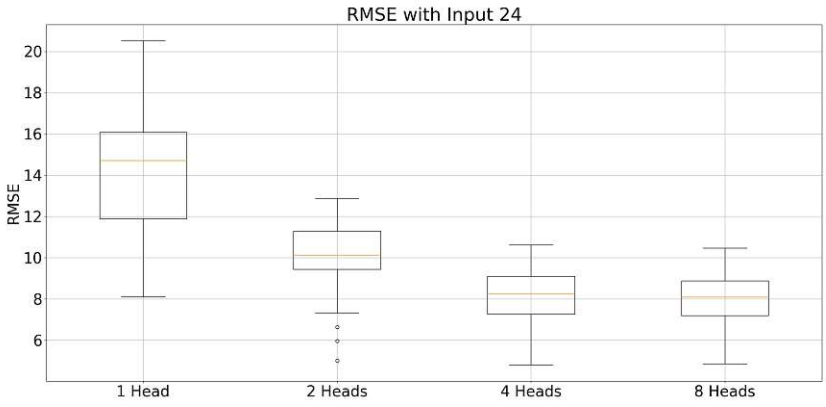


Figure 3.8: Weekend model configuration with fixed input of 24

This plot represents the results with a fixed input length of 24 and the variation of the number of heads with the Weekend model. With the increasing number of heads the median of the RMSE score decreases. For the last two head configurations the median stays at the same value. For this input sequence length the two and the eight head configuration achieves almost the same scores by comparing the mean values. In total for this input sequence length no periodicity is noticed.

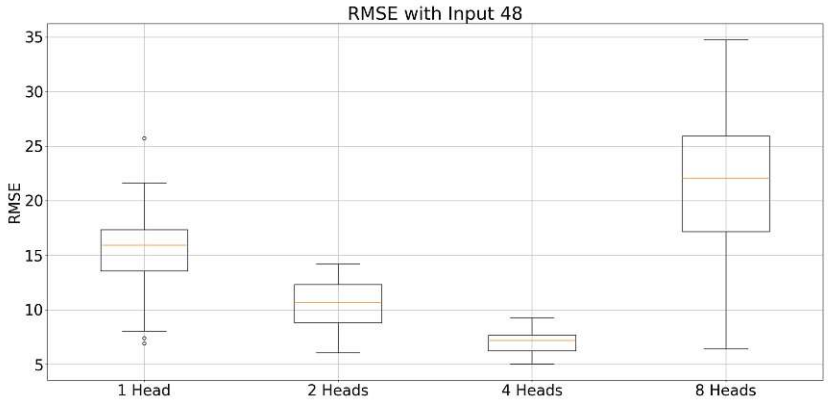


Figure 3.9: Weekend model configuration with fixed input of 48

For this input sequence length the worst performance is achieved with the largest number of heads while the best performance is reached with the four head configuration. In the next section this model will be then compared with the LSTM and the SARIMA model.

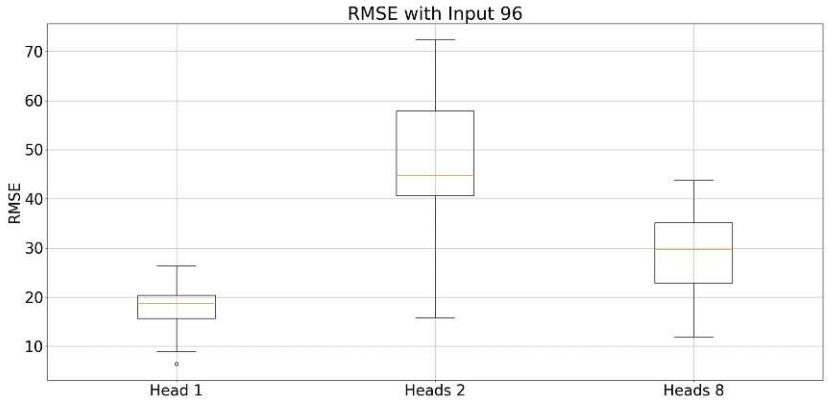


Figure 3.10: Weekend model configuration with fixed input of 96

For this input sequence length there exists a periodicity with the one, two and eight head configuration. The scores for the four head configuration is in the range of 10^5 and so the results are not very representative and so they were filtered out.

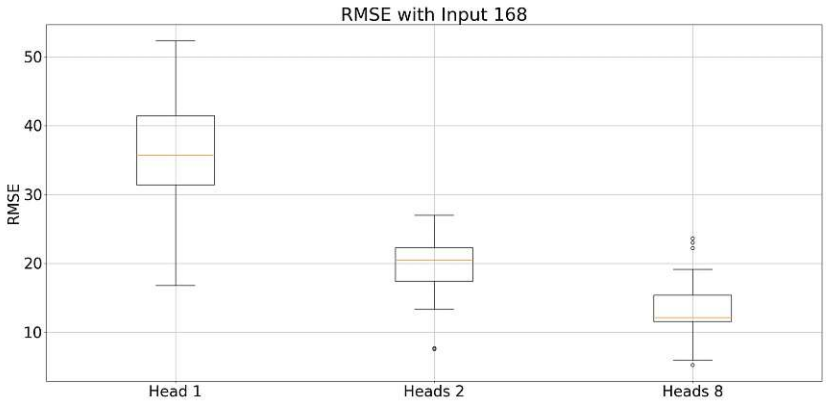


Figure 3.11: Weekend model configuration with fixed input of 168

For the longest input sequence the results have the same tendency to the lowest input sequence length of 24 hours. The results from the four heads configuration were also filtered out because the Transformer does not perform well with that configuration.

Model	mean	std	min	max
1Head	13.22	2.38	7.61	17.11
2Heads	8.03	1.16	4.98	10.35
4Heads	9.94	1.02	7.03	11.43
8Heads	7.78	0.95	5.96	10.05

Figure 3.12: Result table from basic model with 24 input

Model	mean	std	min	max
1Head	96.06	39.32	7.94	162.66
2Heads	12.03	2.42	7.13	16.13
4Heads	10.11	1.87	5.51	13.23
8Heads	156.35	71.25	36.63	325.36

Figure 3.13: Result table from basic model with 48 input

Model	mean	std	min	max
1Head	32.12	8.16	12.94	49.01
2Heads	22.76	7.54	6.61	40.38
4Heads	10.26	2.39	5.65	15.15
8Heads	13.51	3.92	5.34	21.61

Figure 3.14: Result table from basic model with 72 input

Model	mean	std	min	max
1Head	52.11	18.46	11.63	94.18
2Heads	11.07	2.63	6.28	16.31
4Heads	194.83	75.28	30.41	363.92
8Heads	28.92	6.91	6.21	42.96

Figure 3.15: Result table from basic model with 96 input

The displayed tables show the results from the basic model from the Transformer. For the input of 48 hours a valley is formed with a lower mean RMSE score for the two and four head configuration. For the 96 h input sequence length the same periodicity is noticed than as with Weekend model. One difference is that the four head configuration score is in a reasonable range. The scores from the other two models decrease with the

increase of the number of heads and so the tendency is the same as for the Weekend model.

Model	mean	std	min	max
1Head	909.73	361.28	153.99	1871.64
2Heads	14.09	5.08	5.21	27.48
4Heads	1007.59	581.72	6.23	2317.13
8Heads	9.79	2.86	4.67	16.64

Table 3.1: Result table from basic model with 168 input

A clear periodicity is noticed for the longest input sequence length of 168 hours. This model configuration has the clearest periodicity of all the different configurations.

3.3.2 LSTM Varying Input Sequence Length

This section analyses the results from the LSTM model for the same test data which was used before. The basic and the Weekend model were implemented the same way as the Transformer in case of the number and type of the additional features. The LSTM model was configured with four layers with 96 neurons and a batch size of 32. To get a trained model with comparable results to the Transformer, a total number of 100 epochs was chosen for the LSTM.

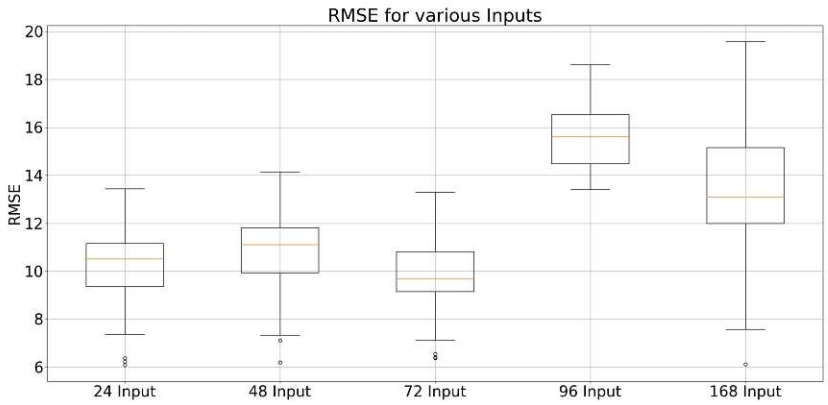


Figure 3.16: RMSE results from the LSTM basic model

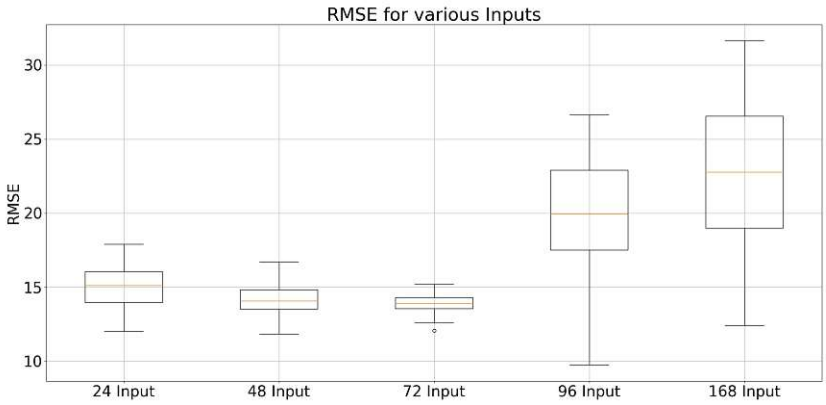


Figure 3.17: RMSE results from the LSTM weekend model

Overall the LSTM model proves that the model is able to predict the energy value with a reasonable low RMSE. In total the basic model produces a lower RMSE score than the Weekend model. The lowest mean RMSE score is reached with the input of 72 for the basic model. This model is chosen to compare the prediction performance with the other two types of models which will be discussed in the last section of this chapter.

3.3.3 SARIMA Varying Input Sequence Length

This section analyses the results from the SARIMA model with a basic and a Weekend configuration. The ARIMA model is built up by using the default values for the order (p, d, q) parameter. These three parameters where chosen to be one which is the default setting. By adding the seasonal order $= (1, 1, 1, 7)$ parameter creating the SARIMA model. The 7 indicates a weekly data for the model.

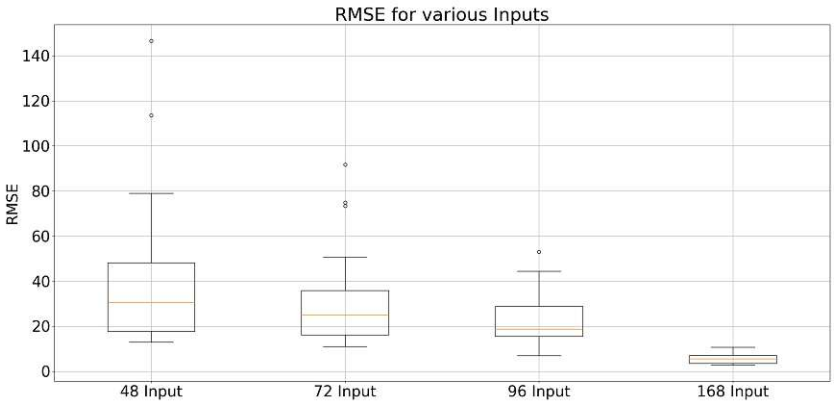


Figure 3.18: RMSE results from the SARIMA weekend model

This plot represents the RMSE scores from the SARIMA model. The lowest mean RMSE score is achieved with the model that has the largest input sequence of 168 hours. The SARIMA needs a hole week to predict the next week. This model is chosen to compare the prediction performance with the other models such as LSTM and Transformer. The results from the basic model has the same tendency but with a higher mean RMSE score for the input length of 168 hours.

3.3.4 Transformer vs. SARIMA vs. LSTM Model Comparison

In this section the three chosen models from the previous chapter will analyse and compete against each other. For the LSTM the Weekend configuration with an input length of 72 hours was chosen, for the SARIMA the Weekend model with an input length of 168 was chosen and for the Transformer model four heads and the input length of 72 was chosen.

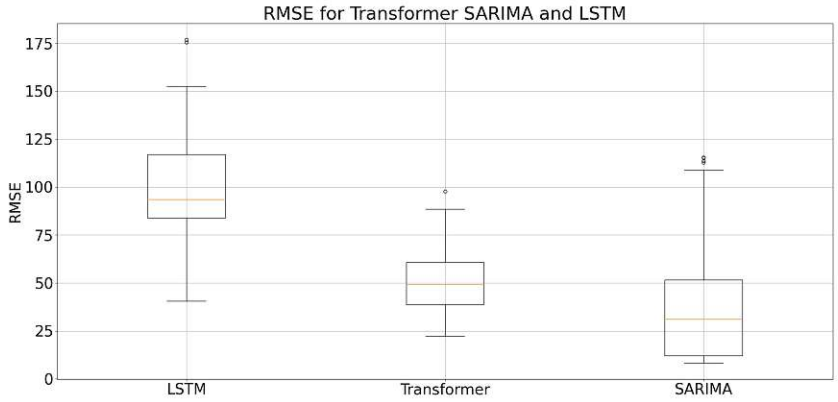


Figure 3.19: Transformer vs SARIMA vs LSTM RMSE results

Overall the results from the LSTM model gets the highest RMSE scores with a median of 10. The median from the best performing Transformer model is smaller than 8. The lowest median score was achieved with the SARIMA model with a score of approximately 5.7. On the other hand the SARIMA results have the largest boxplot, indicating that the model reacts different on the same input sequence than the other models. An additional statistical test will be used to provide statistically relevant analysis for the comparison from the three RMSE scores. The following hypothesis will be formulated:
There are no statistical differences between the three models.

Model1	Model2	meandiff	p-adj	reject
LSTM-24-Basic	SARIMA-168-Weekend	67.47	0.02	True
LSTM-24-Basic	Transformer-72-4Heads-Weekend	56.49	0.05	True
SARIMA-168-Weekend	Transformer-72-4Heads-Weekend	10.97	0.06	False

Table 3.2: Result table Tukey HSD for 3 model comparison

In this table the Tukey HSD test results are shown. The SARIMA and the Transformer model get statistically relevant lower RMSE-scores and the hypothesis is rejected. On the other hand the difference of the SARIMA and the Transformer model is too low to have any statistical relevant differences. This means that the performance from the two models is roughly the same.

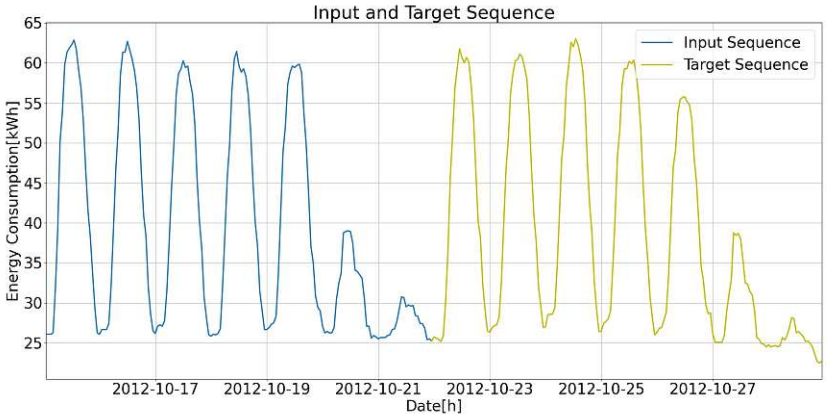


Figure 3.20: Input and target sequence for the Transformer

This plot shows the input and target sequences from the energy dataset. The length of the two sequences is 168, which corresponds to one week. The two sequences are very similar, which shows the weekly periodicity of the dataset. There is also a clear indication of the weekly and daily pattern. The energy consumption is much lower at the weekend than on a typical working day, and there are no public holidays in this week.

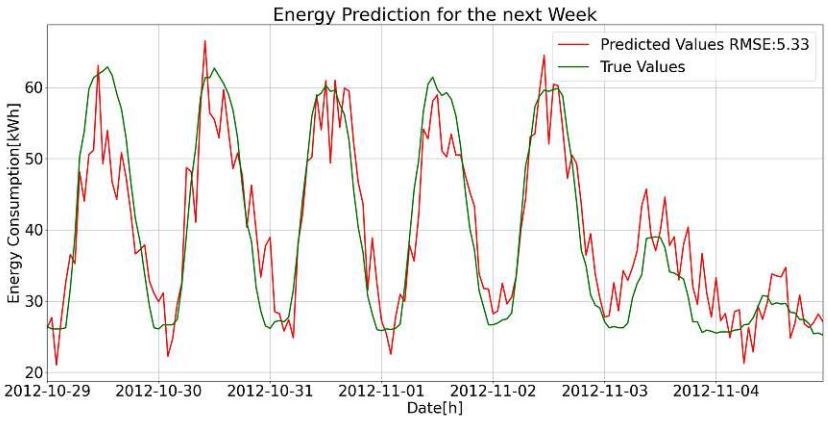


Figure 3.21: Transformer prediction for the next week

This plot represents the energy prediction from the Transformer model with the Weekend configuration, 8 Heads and an input sequence length of 168. The green line represents the true values for the next 7 days in the range from Monday to Sunday. This time range was chosen to find out how the Transformer will react to the jump from the weekend to a normal working day with the end of the input sequence and how it could predict the next weekend. This model configuration indicates the frequency from the whole week reasonably good. The amplitude from Monday and Thursday prediction reaches not the values from the true energy values and there is a small overshoot on Thursday and Friday. Indicating that the jump from the weekend to the working week was not predicted well. Overall the predicted line jumps more up and down and so the smoothness is not the same as for the true values. At the weekend, the frequency was not predicted quite right and the prediction was stretched out.

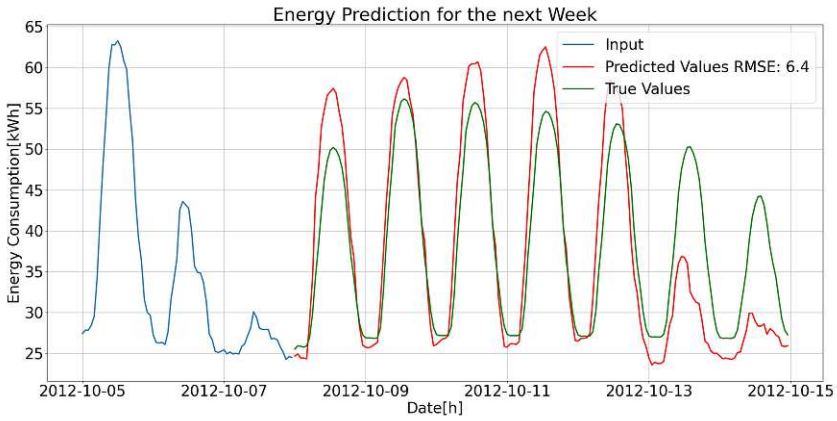


Figure 3.22: LSTM prediction for the next week

The next plot represents the energy prediction from the LSTM model with the configuration of an input of 72 and the basic model. The blue line shows the input sequence which consists of three days starting with a Friday and ending with a Sunday. This point was chosen to see how the model predicts the first day of the week after a weekend. Noticing that the model has almost a perfect frequency over the whole predicted week. On the other hand the amplitude from the prediction never reaches the same level of the true values especially for the weekend. Also the amplitude is rising and falling with a maximum in the middle of the week.

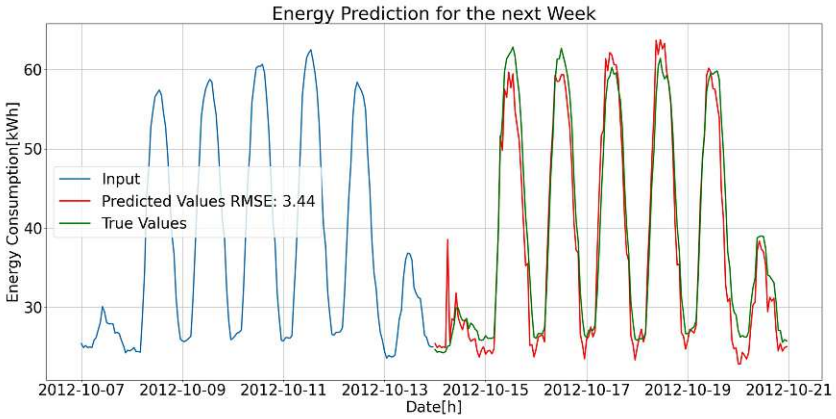


Figure 3.23: SARIMA prediction for the next week

The last plot represents the results from the SARIMA model with an input length of 168h and different weekday input sequence configurations from Sunday to Saturday. The SARIMA model achieves the lowest RMSE score from all models counting 3.4. The predicted line has almost the same smoothness as well as the frequency. At the beginning there is a small peak but then the prediction is very close to the true values.

3.4 Close Price Prediction

In this section, the trained models from the BTC dataset will be tested. The prediction length is set to 10 minutes, which is a very short time window. The RMSE score is chosen to measure the prediction performance by comparing the true values with the predicted values. A time shift of 10 minutes is applied to cover the entire validation range by shifting this time window after a prediction and calculating the RMSE score. The first part of the results for the BTC closing price dataset focuses on a time range of one month, which yields 240 values.

3.4.1 Transformer Varying Number of Heads

In the first part of this section, the Transformer model is tested with a different number of heads to determine whether there are patterns in the RMSE value. The hypothesis is the same as in the section before for the energy dataset.

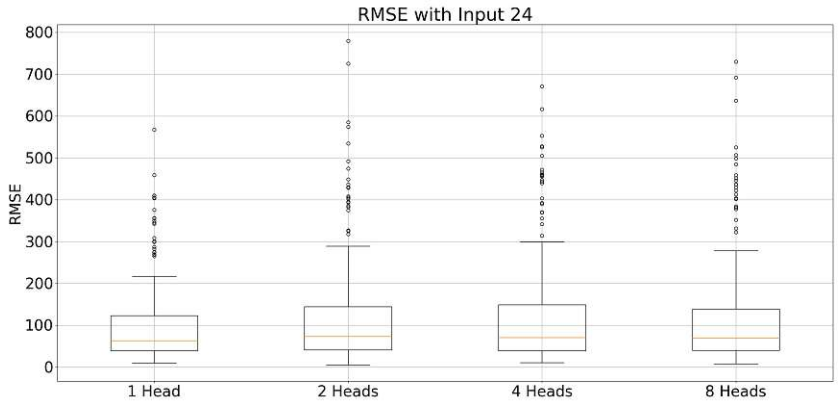


Figure 3.24: RMSE score from the 3EMA model with an input of 24

This plot shows the results for a fixed input of 24 minutes. Overall, there is no difference between these numbers of heads for this input sequence length. Only the number of outliers is different, but no patterns were detected with the different numbers of heads.

Model	mean	std	min	max
1Head	154.86	160.22	9.87	770.85
2Heads	157.50	168.92	15.76	714.60
4Heads	157.22	163.33	10.97	754.02
8Heads	141.23	144.20	14.84	653.82

Figure 3.25: Result table from the 3EMA model with an input of 48

Model	mean	std	min	max
1Head	167.76	163.22	16.22	707.43
2Heads	196.75	203.90	17.23	846.61
4Heads	191.83	201.32	17.03	852.67
8Heads	176.71	179.23	15.93	753.52

Figure 3.26: Result table from the 3EMA model with an input of 72

Model	mean	std	min	max
1Head	126.24	139.24	11.21	1102.85
2Heads	128.24	131.83	15.34	947.16
4Heads	150.04	165.97	13.44	1107.38
8Heads	135.38	143.28	17.96	1037.23

Figure 3.27: Result table from the basic model with an input of 24

Model	mean	std	min	max
1Head	180.26	185.28	15.39	1304.29
2Heads	174.25	173.91	16.43	962.63
4Heads	189.73	198.81	19.55	1176.67
8Heads	133.23	129.85	16.35	925.21

Figure 3.28: Result table from the basic model with an input of 48

Model	mean	std	min	max
1Head	195.22	204.13	12.34	1502.11
2Heads	163.99	166.44	16.62	1185.26
4Heads	174.03	180.29	18.25	1531.07
8Heads	204.51	216.94	16.99	1599.16

Figure 3.29: Result table from the basic model with an input of 72

For the sake of completeness, the remaining results are presented here in table with the most important features for the analysis. Compared to the previous dataset, there is no describable pattern in the increase in the number of heads. The difference in the mean RMSE values of the two models 3EMA and basic is hardly noticeable.

3.4.2 LSTM Vaiying the Input Sequence Length

This part shows the results of the tests with the LSTM for the BTC data. Two different model types with a different number of inputs are trained. The naming is the same as for the transformer model, i.e. basic and 3EMA. For the basic model, only the target column is used to train the model, for the 3EMA model the same columns are used as for the Transformer training. The model architecture differs slightly from that of the energy dataset with only three layers, 50 neurons and a dropout rate of one while the batch size of 32 stays the same. Since the BTC dataset is much larger, a trained model could be obtained after a smaller number of epochs. No further fine-tuning is done for the LSTM model. The model is only intended to serve as a comparison to a Transformer model.

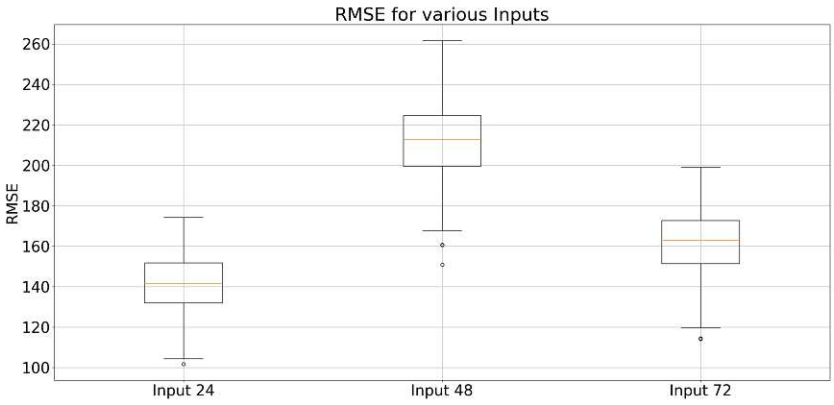


Figure 3.30: RMSE score from the basic model

This plot shows the results of the basic LSTM model. The lowest values were achieved with an input sequence length of 24 minutes. This model configuration is then used

for comparison with the SARIMA model and the Transformer model with the best performance.

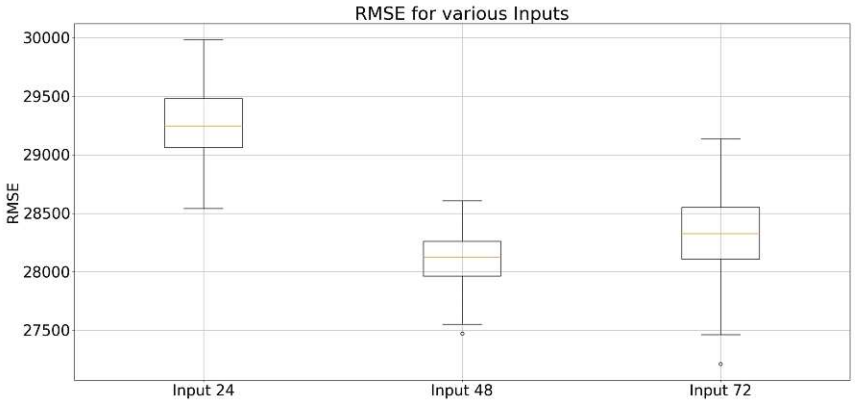


Figure 3.31: RMSE score from the 3EMA model

The 3EMA model type is not able to predict BTC prices with a reasonable good accuracy, as the result in the plot shows. The lowest RMSE values are in the range of $2,750$, which is a big difference to the result of the basic type.

3.4.3 SARIMA Varying the Input Sequence Length

For the BTC dataset, a slight change in the hyperparameters for SARIMA is required as the measurement frequency is one minute. The seasonal order is set to $(1, 1, 1, 2)$, which corresponds to a minute resolution. The rest remains the same as for the other dataset.

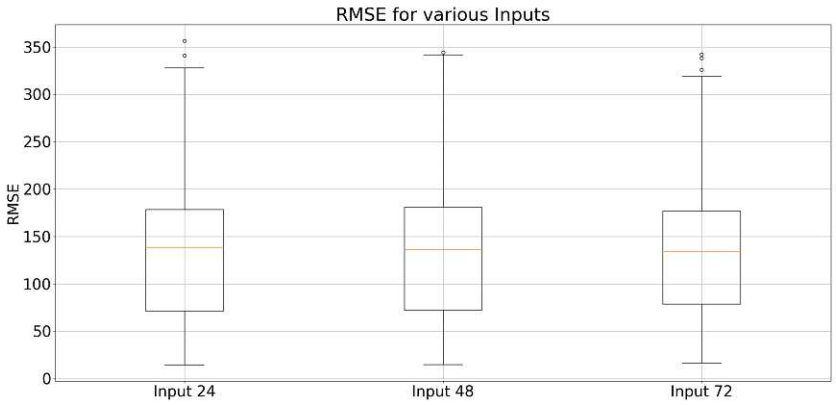


Figure 3.32: RMSE score from the basic model

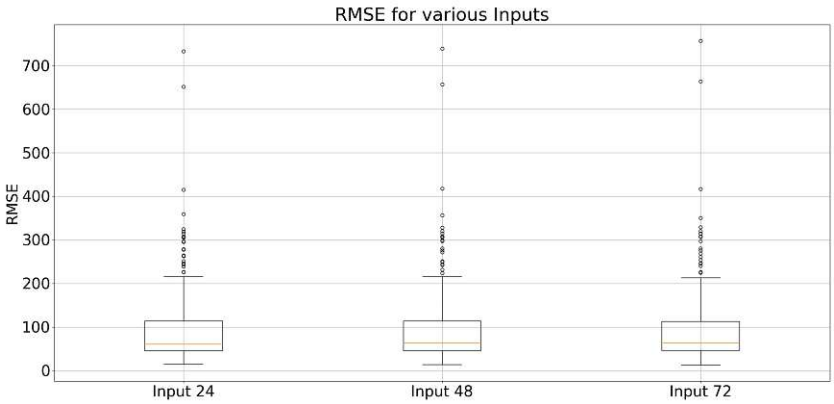


Figure 3.33: RMSE score from the 3EMA model

Both configurations of the SARIMA model can predict the closing prices well, with the 3EMA model the prediction of the price is most accurate. There is also no real difference in a different length of the model's input for both model types. The only notable difference is that there are many more outliers in the 3EMA model.

3.4.4 Transformer vs. SARIMA vs. LSTM

In this section, the 3 models with the lowest RMSE values are compared with each other and the individual predictions are analysed in more detail. The following hypothesis is formulated for this purpose: There are no statistical differences between the three models.

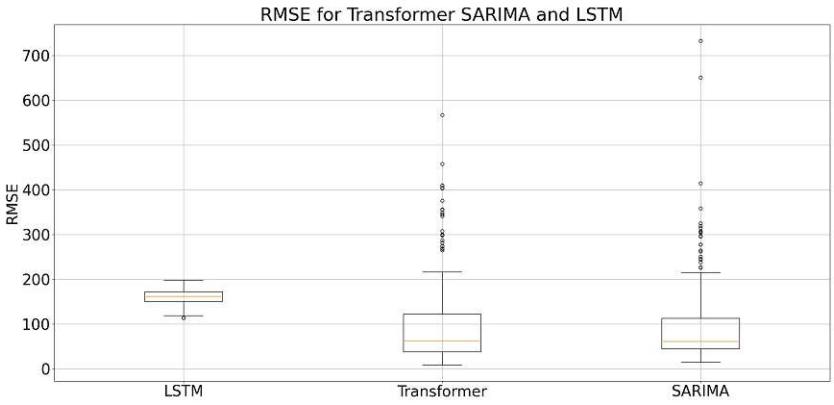


Figure 3.34: Transformer vs SARIMA vs LSTM RMSE results

The LSTM model has the smallest boxplot size, which indicates a small distance between the maximum and minimum RMSE values. This shows that the LSTM model makes more consistent predictions than the other two models, which have many outliers. The Transformer and SARIMA models have a lower median RMSE value. The size of these two boxplots is also almost identical.

Model1	Model2	meandiff	p-adj	reject
Transformer-24-1Head-3EMA	SARIMA-24-3EMA	−5.22	0.74	False
Transformer-24-1Head-3EMA	LSTM-72-3EMA	27.62	0	True
SARIMA-24-3EMA	LSTM-72-3EMA	32.83	0	True

Table 3.3: Result table Tukey HSD for 3 model comparision

This table shows the results of the Turky HSD test, which is used to check whether there is a statistical difference between the three models. In the first row, the test compares the SARIMA model with the Transformer model. Here, the difference between the two mean values is very small, so that the hypothesis is not rejected. In the other two rows, in which the LSTM model is compared, the hypothesis is rejected and a statistically relevant difference is determined by the test.

Figure 3.35: Transformer price prediction example

This plot shows an example of a Transformer model with an input of 24 minutes and one head. The blue line represents the input sequence for the transformer, while the yellow line shows the target sequence. The target sequence is exactly as long as the prediction target of 10 minutes. The red line shows the price prediction by the model, while the green line shows the actual price. The model predicts a reasonable good price development, which also corresponds to the actual values. In contrast, the minimum at the end of the forecast range was not predicted.

Figure 3.36: SARIMA price prediction example

This plot shows the prediction of the SARIMA model. The legend is the same as before, with the exception that the target sequence is missing and is not required for this model. The input length here is 24 minutes with model type 3EMA. In this example, the model forecasts a similar price trend, but this time commodity prices fall towards the end of the forecast.

Figure 3.37: LSTM price prediction example

The forecast of the LSTM model is displayed here. This is the basic model configuration with an input length of 24 minutes. The red line represents the prediction of the model. This behavior was observed in all tests where the forecast is either approximately in a straight line upwards or downwards, which means rising or falling prices.

3.5 Visualization of Attention

The purpose of this section is to analyze the attention from the decoder layer by using various techniques to find patterns and understand the attention mechanism better. For the following plots, the basis is the multi-head attention equation formulated as follows: $\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$, $head_i = \text{Attention}(Q_i, K_i, V_i)$. The trained models with the defined training weight are used to calculate the attention. The test range is the same as in the previous chapter when the prediction performance was measured. The frequency with which the time window is shifted is also the same, i.e., 24 hours for the energy dataset and 10 minutes for the BTC prices. The two different datasets are analyzed in different ways, e.g., for the building energy dataset and the BTC dataset the influence of the additional features and how evenly the attention is distributed over the input sequence. How periodic the attention is is only applied to the very strongly periodic building energy dataset. While for the BTC dataset alone, a one-, two-, four-, and eight-head analysis is performed to find patterns and analyze them.

3.5.1 Attention Reaction with Weekend Indicator

This section of the paper focuses on analyzing the influence of the additional features on the attention. The hypothesis is that attention in the area of the additional features is higher than in models without such features. First, a heatmap representing the attention is shown for the basic model configuration. Secondly, a plot is presented where the attention versus the actual input sequence is presented. This plot style is applied for the weekend model configuration, and finally, two boxplots will be created. The results will be tested by applying statistical tests to find statistically relevant patterns.

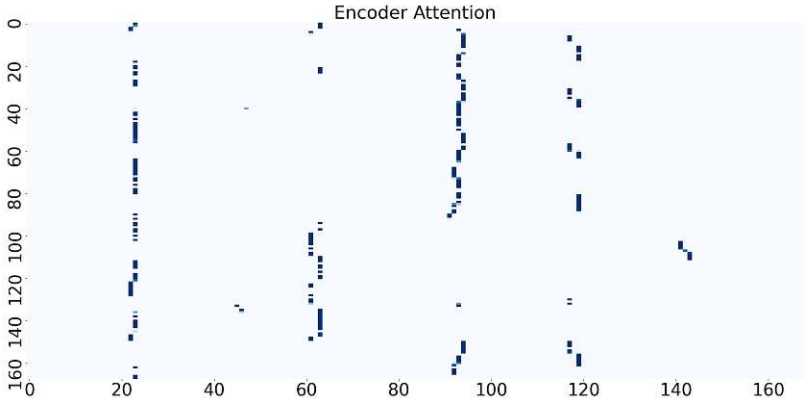


Figure 3.38: Attention matrix from an input length of 168h with a basic model configuration

This heatmap shows the attention pattern from a two-head model configuration for a specific input sequence. The heatmap's dimensions are 168 times 168, which corresponds

to the dimension of the model itself and the length of the output of the model. There are multiple lines of attention at different positions such as 20, 60, 90, and 115; this indicates a two-head model configuration. Fewer lines of attention would indicate a one-head model configuration. The first row corresponds to 168 points from an input sequence, and the blue dots represent the attention to each point. The second row is then shifted by one hour until the end of the test range is reached.

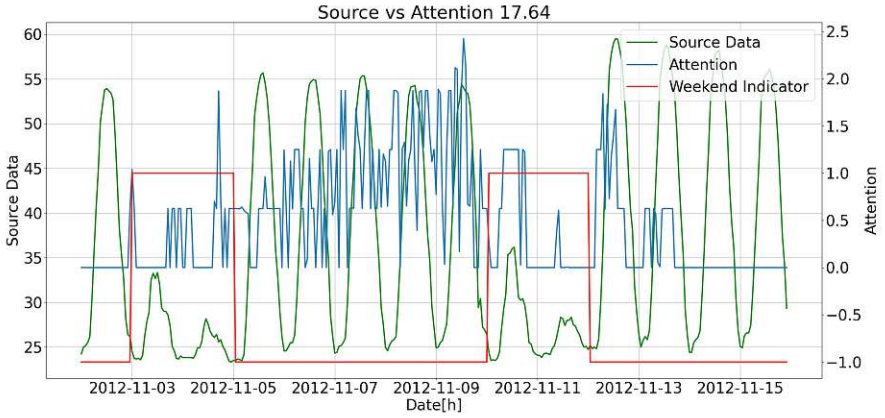


Figure 3.39: Input sequence vs. attention with an input length of 168 and a basic model configuration

This diagram shows the attention compared to the actual input sequence. The green line shows the input sequence for the transformer, the blue line shows the attention for the input sequence, and the red line shows the weekend indicator. The total range shown is 336 points, which corresponds to two weeks. The attention is plotted by first taking the first row with its attention values from the attention heatmap from earlier. Then the values from the second row are taken and shifted one point to the right. These two lines are then added together, whereby the length is obtained, and the total range after this step is then 169. This process is then repeated 167 times for the entire next week and the attention is measured in the areas where the red line has a value of 1. The header provides the percentage of the total attention relative to the attention within the additional feature's domain. In this plot, 17.64% of the attention falls within the weekend range. This relatively low score suggests that, for this particular input sequence, the attention transformer is not focusing on the weekend periods much.

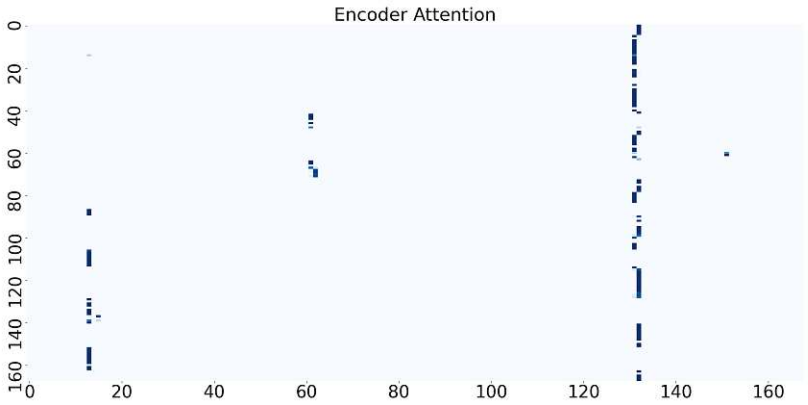


Figure 3.40: Attention matrix from an input length of 168h with a weekend model configuration

This heatmap shows the attention visualization from a one-head model configuration. In this figure, there is only one major attention line noted at the point on the x-axis of about 135. This indicates a one-head model configuration with fewer attention lines, and the conjecture is strengthened by this observation.

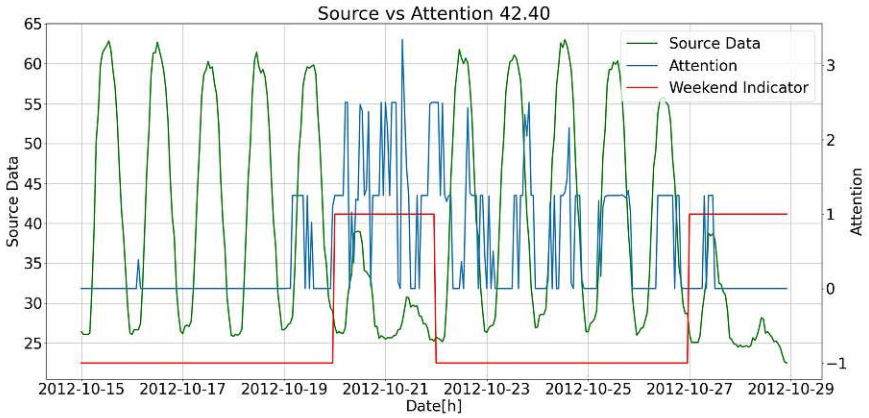


Figure 3.41: Input sequence vs. attention with an input length of 168 and a weekend model configuration

This graph shows the feature attention from a one-head weekend model configuration. For this model configuration, the attention during the weekend increases to 42.40%. This example reinforces the influence of the additional feature on attention for this specific input sequence. Furthermore, the model's training with only one attention head reinforces the significance of this attention-related feature.

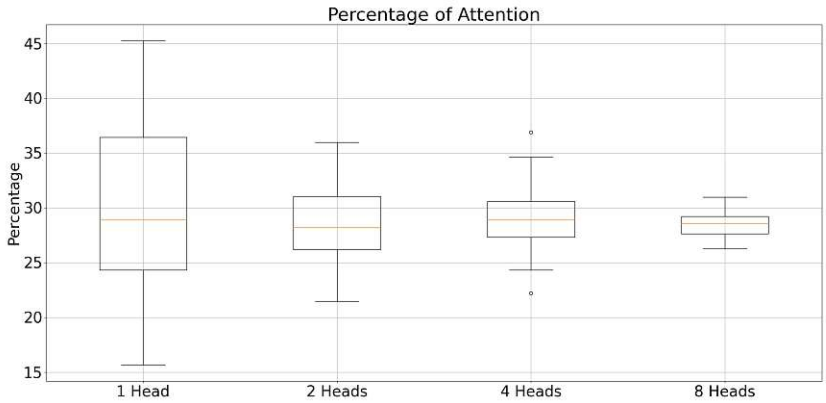


Figure 3.42: Attention percentage from the basic model

This graph shows the accumulated attention percentages of the basic model over four different head combinations. The median value for all numbers of heads is about 28%. Notably, as the number of heads increases, the size of the boxplot decreases, while the median value stays at the same value. For the single-head configuration, the maximum attention percentage reaches 45%, and the minimum drops to 15%, representing the widest range among all other head configurations.

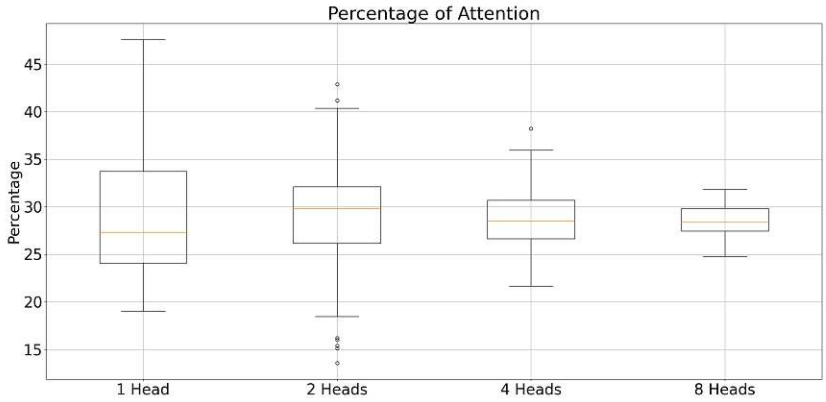


Figure 3.43: Attention percentage from the weekend model

This plot presents the result for the weekend model configuration, with a median range of about 30% across all number of heads. The size of the boxplot has the same tendency as in the results from earlier. In general, the differences between the two model configurations shown are not notable. In total, the hypothesis is rejected because the median value does not decrease with an increase in the number of heads.

3.5.2 Attention Uniformly Analysis

The purpose of this section is to analyze how evenly the attention is distributed over the input sequence. Based on the observation from the section before, where the number of attention lines increases with the number of heads, it may be possible that attention is more evenly distributed with a higher number of heads. As shown in figures 4.42 and 4.43 in the last section, the eight-head configuration reaches the smallest boxplot size for both weekend and basic types. A hypothesis is formulated as follows: Attention is more uniformly distributed with a higher number of heads compared to a lower number of heads. To test this, the standard deviation of attention is calculated after each prediction. A value of 0 indicates a perfectly uniform distribution, and a higher value indicates a less uniform distribution.

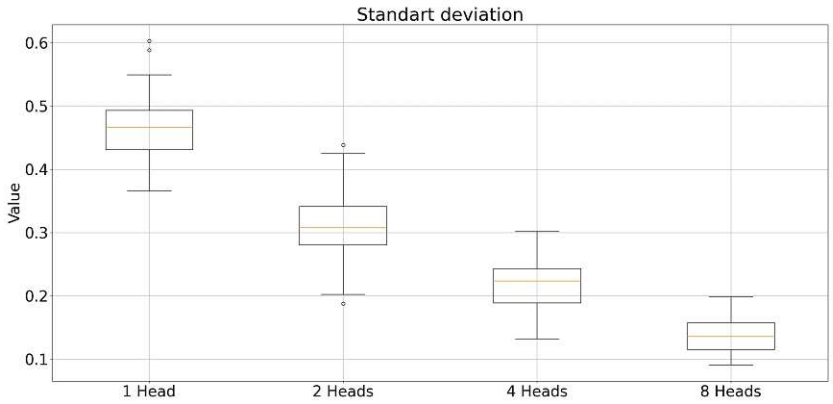


Figure 3.44: Uniformly distributed plot for the weekend model with an input of 168

This graph shows the standard deviation values for each different number of head combinations. The median value tends to decrease as the number of heads increases. This behavior supports the hypothesis. The median from the eight-head configuration is approximately four times smaller than the one-head configuration. These results were achieved with the weekend model, and the basic model configuration has the same behavior with no notable differences. Therefore, the hypothesis is correct for both types of models.

3.5.3 Periodicity Analysis

In this section, the periodicity of the attention will be analyzed. In the previous section and in the data analysis, a tendency for periodicity in the data was found, and the question now is how this affects the attention. Since attention is more evenly distributed with increasing number of heads, which was found in the previous section of this chapter, the following hypothesis can be formulated: The periodicity of the attention is higher for a lower number of heads, and vice versa.

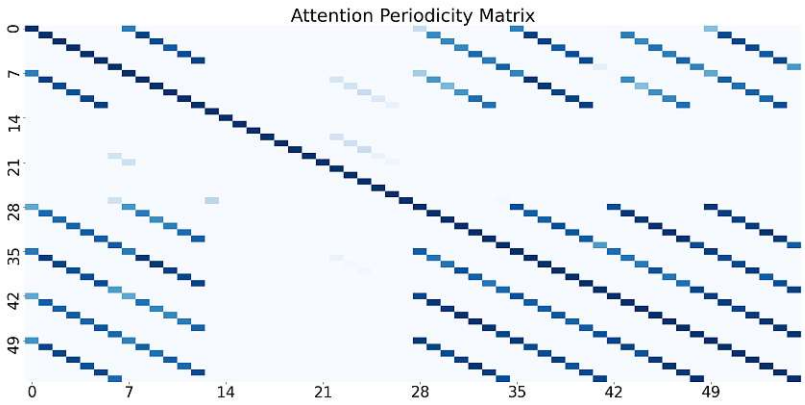


Figure 3.45: Attention periodicity heatmap from the weekend model with an input of 168h

This heatmap shows how the attention pattern according to an input sequence is periodic. Each point in the heatmap represents the R^2 score, calculated by comparing the attention of one prediction with another. On the diagonal, all values are one because two identical attention patterns were compared. Besides the diagonal, there are other points with an R^2 score larger than 0, such as in the top left corner of the heatmap. For example, with combinations of points like (0,7), (1,8), (2,9), and so on. This behavior gives the idea that the input sequence has a weekly periodicity and the attention as well. Additionally, it's noteworthy that there is a corridor where the periodicity abruptly breaks. This phenomenon is explained by the non-periodic nature of certain parts of the input sequence. In the validation data, holidays were not filtered out to see how the model reacted to those days. The heatmap was generated by a model with only one head, and there are points beside the diagonal that significantly exceed a value of 1.

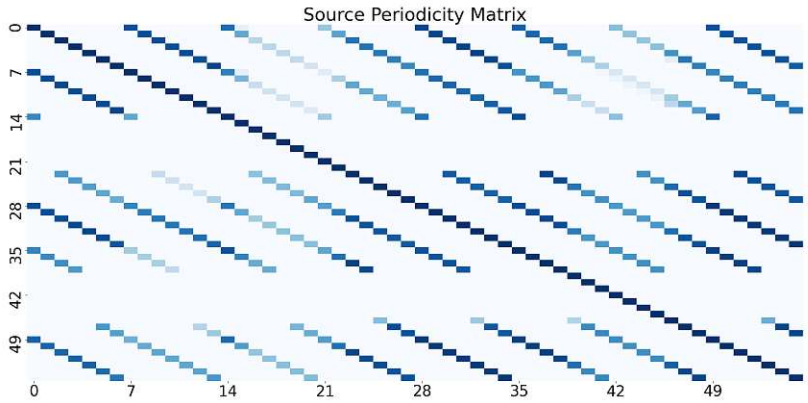


Figure 3.46: Input sequence periodicity heatmap with an input of 168h

This heatmap represents the periodicity of the input sequence for the Transformer. There is a clear corridor from the 15th to the 21st point and from the 38th to the 46th point, where the periodicity is broken. The remaining holidays are in this area, which is why attention is also interrupted.

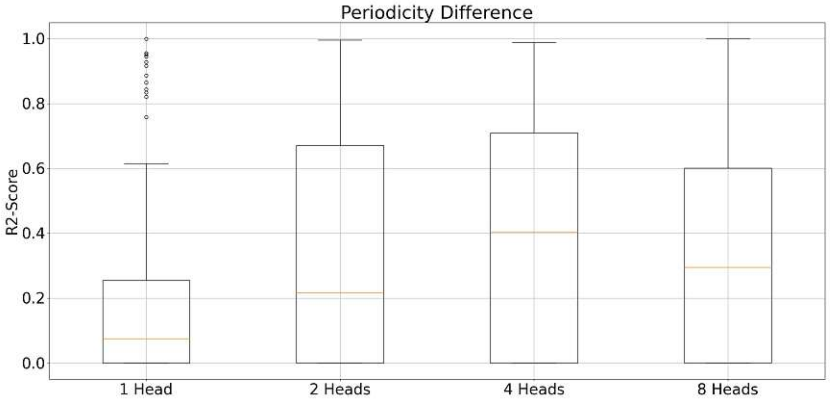


Figure 3.47: Periodicity difference from the weekend model

A mask is generated to measure how strongly the attention is periodic. The mask has a value of one for every point in the postulated weekly pattern. Then the difference is calculated using the value in the mask and the previously shown attention matrix. Furthermore, only the data points that align with the assumed weekly pattern are considered for the calculation, while the data points where the weekly pattern is disrupted are excluded. This graph plots the difference to show which model configuration has the best attention periodicity. The lowest values were achieved with the one-head model configuration. For the other head configurations, the values are overall higher. These

results were created with the “Weekend” model type.

Model1	Model2	meandiff	p-adj	reject
Periodic-1Head	Periodic-2Head	0.18	0	True
Periodic-1Head	Periodic-4Head	0.22	0	True
Periodic-1Head	Periodic-8Head	0.17	0	True
Periodic-2Head	Periodic-4Head	0.04	0.66	False
Periodic-2Head	Periodic-8Head	−0.01	0.99	False
Periodic-4Head	Periodic-8Head	−0.05	0.50	False

Table 3.4: Results from the Weekend Model

This table shows the results from the Tukey HSD Test, it tests which different model configurations have a statistically relevant difference. All combinations with the one-head configuration reject the hypothesis with a confidence level of 95%. The differences from the other model configurations have no statistical differences.

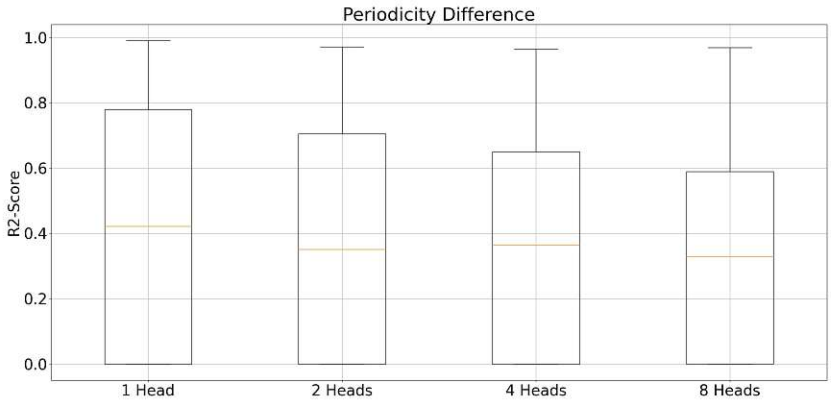


Figure 3.48: Periodicity difference from the basic model

This plot shows the results from the basic model configuration. Interestingly, the lack of the additional feature ends in a lack of periodicity of the attention. Overall, the periodicity score stays at a median of 0.4 for all head configurations, indicating the influence of additional features in the attention periodicity.

Model1	Model2	meandiff	p-adj	reject
Periodic-1Head	Periodic-2Heads	−0.04	0.76	False
Periodic-1Head	Periodic-4Heads	−0.05	0.06	False
Periodic-1Head	Periodic-8Heads	−0.07	0.39	False
Periodic-2Head	Periodic-4Heads	0.01	0.99	False
Periodic-2Head	Periodic-8Heads	−0.02	0.95	False
Periodic-4Head	Periodic-8Heads	−0.01	0.99	False

Table 3.5: Results from the basic Model

For the basic model configuration there are no statistically relevant differences with the confidence of 95%.

3.5.4 One Head Analysis

In this section, the one-head configuration of the transformer is analyzed based on the BTC dataset. The hypothesis is formulated as follows: The attention is concentrated in the area of the largest price change of a certain input sequence. First, every different input sequence will be analyzed of its price change behavior in the attention range by calculating the slope of the price change. If the slope is larger than 1.1 then this is defined as a rising price. If the slope is less than 1.1 then it is defined as a falling price, and otherwise, the price change is defined as neutral. This was done for all different input lengths and for the two model types.

Model	rising	falling
Basic-24	64	175
Basic-48	192	47
Basic-72	106	133
3EMA-24	96	143
3EMA-48	166	73
3EMA-72	131	108

Table 3.6: Number of price changes from the two models

This table shows the number of rising i.e. falling number of price changes from the input sequence in the attention range. Both types achieve approximately the same number of falling and rising price changes for the input sequence length of 72. More precisely, the values differ only by about 10%. For the other rows, the difference is much higher. Notable is that there are no neutral price changes.

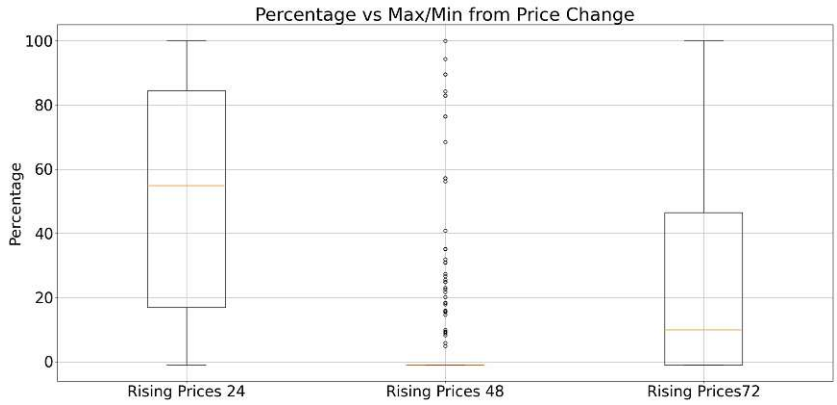


Figure 3.49: Attention percentage of rising prices

This plot shows the percentage of price change relative to its maximum price change in the range of the attention for rising price change for the 3EMA model. 100% means that the attention range is in the highest positive price change of this specific input sequence, where 0% means the smallest price change. The median for the lowest input sequence length is larger than 57% whereas the mean value is 51%. For this input sequence length, the Transformer concentrates slightly higher price changes. For the 48 input lengths, the majority are outliers, and the tendency is toward 0%. For the last input length, the median is about 12% while the boxplot ranges over the whole range from 0 to 100 with no outliers.

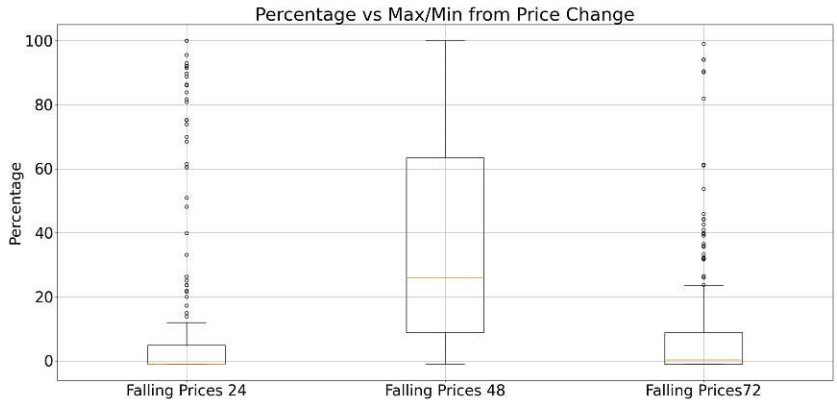


Figure 3.50: Attention percentage of falling prices

In comparison, this plot shows the results of the falling price changes. Instead of the maximum, the minimum is now taken as a reference to calculate the percentage. The two input sequence lengths of 24 and 72 have the same tendency as the 48 input from

the plot from before, with a large number of outliers. Interestingly, the highest median from 23% is achieved by the 48 input, where the number of points is also the highest from this model configuration.

Model	mean	std
Basic-24	31.12	34.87
Basic-48	20.37	29.97
Basic-72	22.41	29.24

Figure 3.51: Rising prices summary

Model	mean	std
Basic-24	18.81	32.49
Basic-48	16.25	26.31
Basic-72	13.58	25.99

Figure 3.52: Falling prices summary

The table on the left represents a short summary of the rising prices from the basic model type. There are no major mean differences over the range of different input lengths, like for the other model type. This behavior is also noticed for the falling price table on the right side. In total, for the basic model configuration, the Transformer concentrates a bit more on the relative positive price change.

3.5.5 Two Head Analysis

This section analyzes the two-head model configuration by forming the following hypotheses: if the attention is more likely to occur at the beginning and end of an input sequence, then a significant price change occurs. A non-parametric test is used to test the results of their correlation. Attention is measured in the first and last thirds of the attention area, and then the percentage of total attention is calculated. The sum of these two results is then formed. This is done for each test.

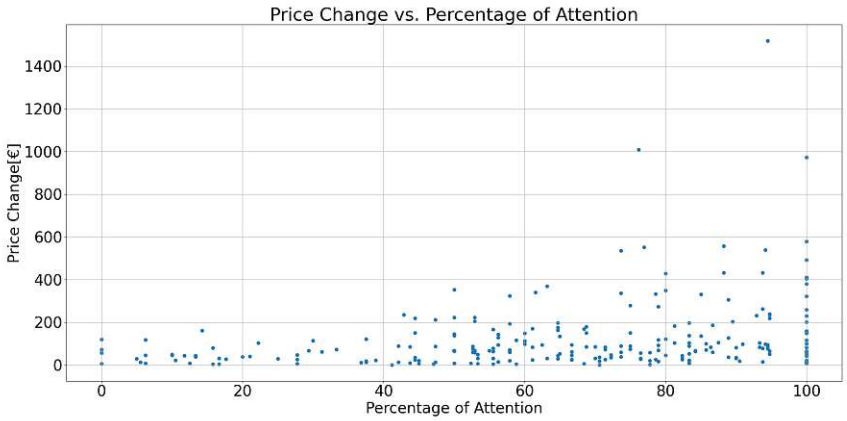


Figure 3.53: 2 head scatter plot

This scatter plot represents the results of the basic model type with an input length of 24. For this input sequence, the correlation is the largest among the tests where a small positive correlation was found.

Model	p-adj	correlation
Basic-24-2Heads	0	0.31
Basic-48-2Heads	0	0.23
Basic-72-2Heads	0.03	0.14

Table 3.7: Results from the basic Model

This table shows the results of the non-parametric Spearman's rank-order correlation test, which measures the correlation of those statistically relevant. For the basic model type, all three different input lengths have a p-value less than 0.05, indicating that the results have a statistical relevance. In total, eleven values are positive, so a positive correlation is measured. The largest correlation is achieved with the lowest input of 24, which was also plotted before.

Model	p-adj	correlation
3EMA-24-2Heads	0.023	0.147
3EMA-48-2Heads	0.325	0.064
3EMA-72-2Heads	0.598	0.034

Table 3.8: Results from the 3EMA Model

This table shows the results of the 3EMA model type. Only the correlation with the entry of 24 has statistical relevance with a 95% confidence level. Compared to the basic model type, the correlation is significantly lower.

3.6 Trading Simulator

The last section of the result chapter will focus on a simulation of how the trained models could be used as a tool to trade on the BTC market. The rules for the simulation were taken from a real trading platform, namely Binance. For the simulation, three different trading classes were defined as follows: the first class is Long, which is a decision from the model that the price will increase. The second class is Short, where the model predicts a fall in prices where also a potential profit is possible, and the last class is Neutral, where the model predicts that the price will not change in a defined range. Every trading decision will cost fees, which in the case of Binance are 0.075%. Additional, leverage can be applied to increase the profit from a trade. For the tests, a leverage of 10 was chosen due to the small time resolution and relatively small price changes. For the tests, two Transformer based models will be analysed i.e. the basic and 3EMA model configurations with an input of 24 and the basic configuration of the SARIMA model. The LSTM model will not be analysed due to its poor performance shown in the last section. First two examples of trading decisions, especially for a right and wrong decision in the case of a long and short case, will be shown and explained. Then a heatmap of the entire trading range is displayed, and its classification results are analysed. To end this section, the

evolution of a virtual wallet will be shown, and the price changes of each training step will be analysed in terms of their normal distribution.

3.6.1 Trading Analyse

The model trades as follows: first, the model gets the input and predicts the output. A linear regression is applied on the prediction to smooth the output from the model. Then based on the regression line, the model makes a decision from the three classes by comparing the last price of the input sequence and the last price of the regression line.

- Long: The start price should be smaller than then predicted end-price.
- Short: The start price should be larger than the predicted end-price.
- Neutral: The difference between the start and end price is in the given range of 10€.

Now the rules for a sell condition were defined. For a long prediction if any true price gets below the start-price −€10 then the model sells the position and the trade prediction would be classified as not successfully. On the other hand, if any true price gets larger than the predicted end-price +€10 then the predicted decision is a success. A short prediction counts as a success if any true price gets below the predicted end-price of −10€ otherwise if any true price gets above the start-price of +€10 then the short position counts a not successful prediction. For the case that any true price not crosses the defined ranges, the model sells the position at any price at the end of the 10-minute forecast range. This action could lead to a positive or negative profit.

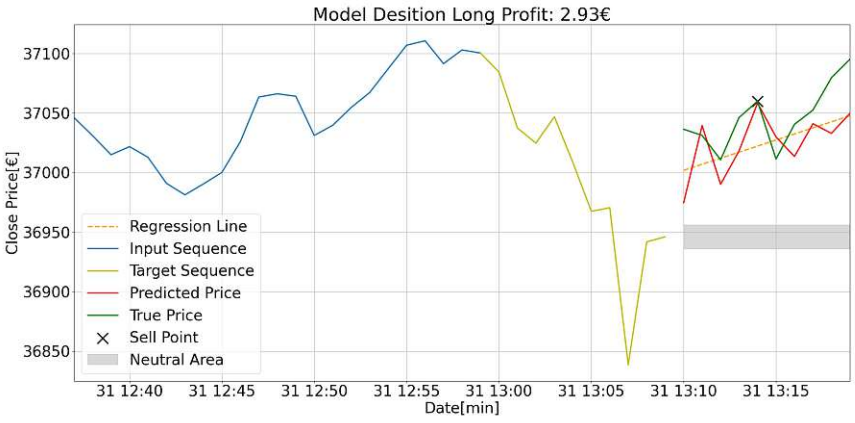


Figure 3.54: Right long decision from the basic Transformer model

The blue line represents the input for the model, the target sequence is illustrated by the yellow line while the red line shows the predicted values, and the green line shows

the true price evolution. The orange dotted line indicates the linear regression from the prediction and is the base for the trading decision made by the model. The gray area shows where the model predicts that the price will stay, which is in this case ±€10. Additionally, a black cross indicates where the model sells the position after a buy. In this plot, a profit €2.93 was made by a successful predicted long position considering fees.

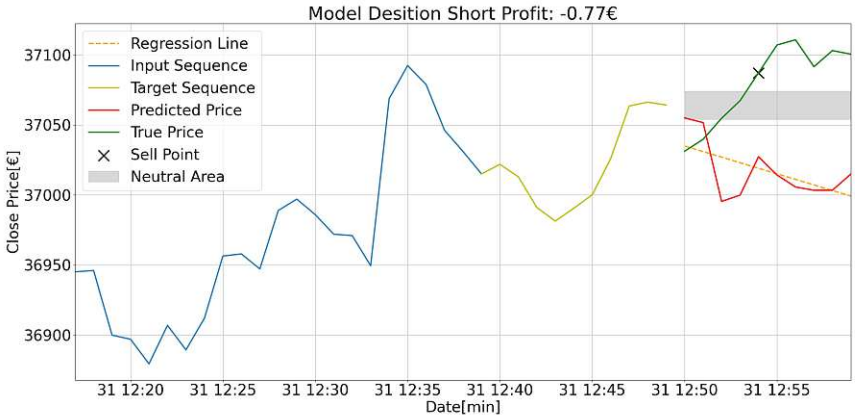


Figure 3.55: Wrong short decision from the basic Transformer model

This plot shows a incorrect short decision from the same model configuration as above. In this case, a loss of 0.77 was achieved with a wrong prediction of falling prices. It is worth mentioning that the model does not sell the position immediately. Instead, the true prices rise above the neutral range, and so the short prediction counts as unsuccessful.

3.6.2 Trading Classification

This section concentrates on the classification analysis from the trading simulation. The range for the tests is now one whole month, which gathers results from Monday to Friday from 08:00 to 16:00, summing up to 960 trading steps.

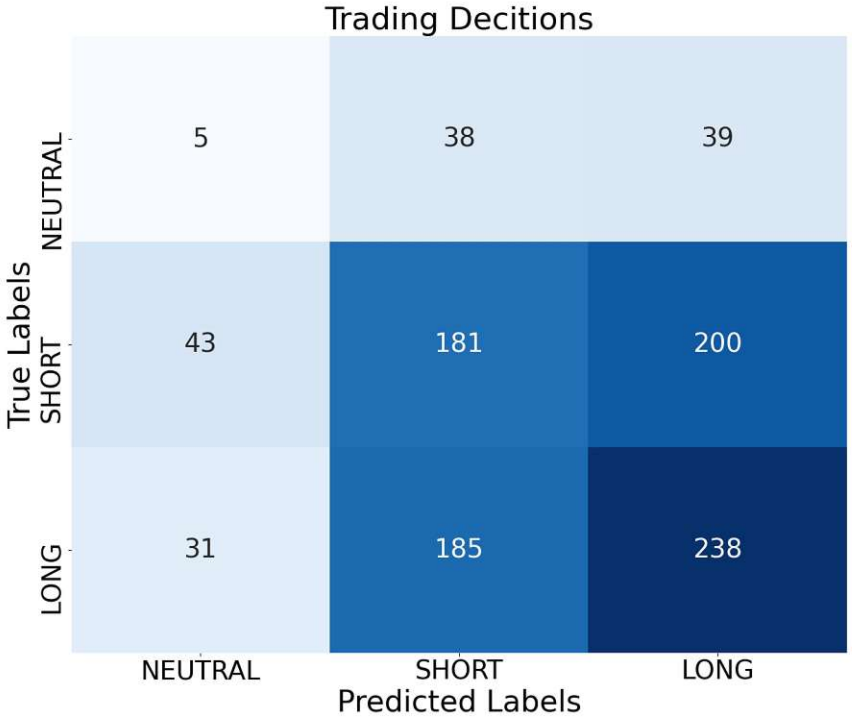


Figure 3.56: Basic trading strategy

This heatmap represents the classification results from the Transformer basic model configuration. On the x-axis, the predicted labels are plotted, and on the y-axis, the true values. Unfortunately, the model's performance in terms of classification is very poor. The worst misclassification occurs when the model predicts a short position instead of a long position, and vice versa.

Class	precision	recall	F1-score
Neutral	0.10	0.12	0.11
Short	0.45	0.42	0.44
Long	0.51	0.52	0.52

Table 3.9: Classification metric for the basic Transformer model

This table shows a small summary of the most important metrics from a classification task. The long class achieved the highest F1-score of 0.52, which is overall a poor performance. The short class gets 0.44 in case of the F1-score and the neutral only 0.11. In total, an accuracy of 44% was achieved by this model configuration.

Class	precision	recall	F1-score
Neutral	0.11	0.18	0.14
Short	0.46	0.24	0.32
Long	0.50	0.66	0.56

Table 3.10: Classification metric for the 3EMA Tansformer model

This table shows the results from the classification task from the 3EMA model configuration. Overall approximately the same results with a slightly higher F1-score for the long class compared to the previous model. An accuracy of 43% was calculated for this model type.

Class	precision	recall	F1-score
Neutral	0.09	0.22	0.13
Short	0.49	0.43	0.46
Long	0.52	0.44	0.47

Table 3.11: Classification metric for the basic SARIMA model

The classification task results from the SARIMA model are shown here. This model achieved overall the worst results among the three tested model types, with a total accuracy of 42%.

3.6.3 Virtual Wallet Evolution

In the last part the model trades with a virtual wallet and the outcome will be analyzed by the amount of money the wallet has have at the end of the month. Additionally, a plot is presented that shows the amount of profit each trade produces, which could be positive or negative. Every model starts with €1000 and every trade budget is fixed to €100. To ensure fair conditions for every model and to get a reasonable good analysis of the profit distribution later.

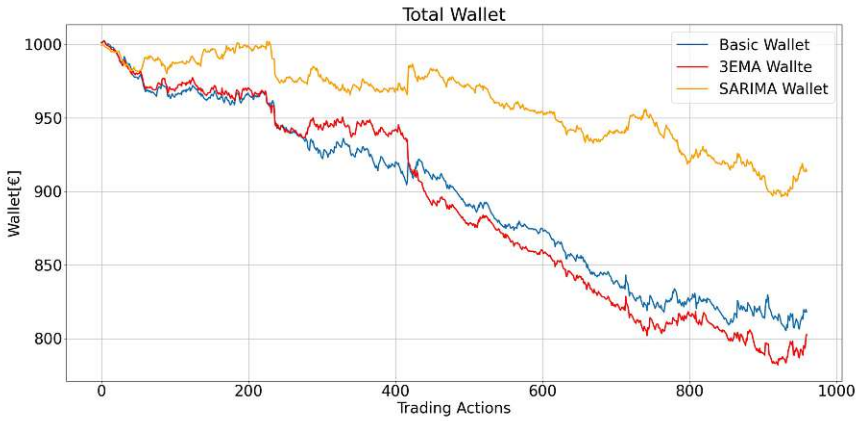


Figure 3.57: Total wallet evolution over one month

This plot represents the wallet evolution from the three tested models from before. Unfortunately, all three models have a negative wallet evolution over the given range, indicating the poor classification performance shown in the last section. The two Transformer based models have the same tendency, where the basic model configuration has more money left in the wallet than the from the 3EMA model. The SARIMA model has a less negative trend and the simulation ends with a higher wallet value than the other models. In total the two Transformer based models end the simulation with a loss of around €200 where the loss of the basic model is significantly less than the loss from the 3EMA model. The loss from the SARIMA model is around €80.

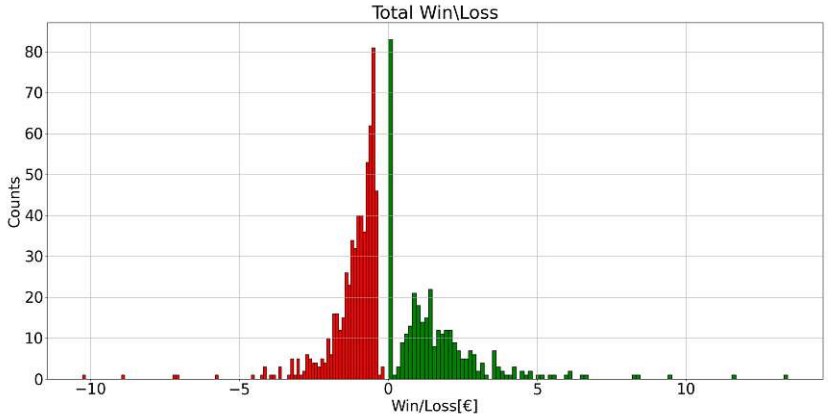


Figure 3.58: Profit distribution from basic model configuration

This plot represents a profit histogram from the basic model configuration. Overall, the number of negative profit counts is higher, indicated by the much denser negative side of the plot. All three models follow a Student's t-Distribution with a 95% confidence and

a value of risk calculation could be performed to indicate the largest possible loss for a given time-frame.

model	1 day	5 days	15 days	25 days
Basic-24-1Head	−22.73	−51.07	−88.59	−114.42
3EMA-24-1Head	−21.62	−48.61	−84.34	−108.94
SARIMA-24	−19.14	−42.91	−74.39	−96.06

Table 3.12: Variation of risk table for multiple days

This table shows the value of risk for four different time ranges. The SARIMA model has overall the lowest value, while the basic Transformer model the highest. This is interesting due to the slightly better performance of the basic model configuration from the wallet evolution.

CHAPTER 4

Disscusion

The following chapter will conclude and summarize all the important findings from the result chapter.We will summarize the key points from the preceding chapter, including energy prediction, close price prediction, visualization of attention, and finally, the trading simulator.

4.1 Energy prediction

In the energy prediction part the main idea was to have a data set based on energy values from a building which was closer described in the previous chapter. The different models would be trained with this data and the prediction power of the different models like Transformer, LSTM and the SARIMA model were analyzed.
By varying the number of heads for the Transformer model and applying the RMSA score for the prediction different results for the different number of heads were measured. Especially for the trained Transformer models two model types with different numbers of inputs were trained. The first type of model is fed by an indicator when the weekend starts and the other type the model is trained only by the variation of the different input length.
For an input sequence length of 24 and 48 hours the results shows no significant periodicity. Generally the performance improved with an increase of the numbers of heads, but the benefits plateaued or even deteriorated at higher head configurations. The best performance was achieved with a four head configuration with an 48-hour input length. With increasing the number of heads it suggests a possible overfitting the model or a lack of efficiency in dding more attention heads to the model configuration. On the other side in particular for 96 and 168 hours input a clearer periodicity was discovered. For certain input length the Transformer model may exhibit inherent periodic tendencies in performance as the number of heads varies. For the longest input sequence length of 168

hours the most distinct periodicity across different head configurations was observed. This could lead to an interaction between the attention mechanism and the temporal structure of the input data, where certain head configurations better capture the underlying patterns. In total the number of heads depending on the input sequence length may depend on learning complex patterns in the data.
For a better comparison with the LSTM and SARIMA model two types of models were designed with the same input parameters to form the weekend and the basic model. The tests show that the SARIMA outperforms the LSTM model the Transformer based models were close behind in the RMSE-performance to the SARIMA models. Only one model was picked from each model type and tested its performance closer. The Tukey HSD test shows that the SARIMA and the Transformer models have statistical similar results. After a visualization inspection from the different outputs the Transformer model creates a more pointed output whereas the LSTM model produces a smoother output.

4.2 Close Price prediction

The idea of the second part was to test the same models as shown before with a clearer non-periodic data set which contains close price BTC values in a 10-minute resolution. For the Transformer model there are two different types provided, one which gets close price data defined with the input sequence length and the other filled with multiple columns, such as close price data and indicator from the 3-EMA strategy. In contrast to the other data set there is basically no difference between the different head-configurations and different model types. In terms of the RMSE score the Transformer and the SARIMA model achieve the lowest ones. For the most time the LSTM-model creates a linear like output which does not reflect any future price patterns from the validation data set. The Tukey HSD test further confirmed that while the LSTM model exhibited statistically significant differences in performance compared to the other models, no significant difference was found between the SARIMA and Transformer models, indicating their comparable effectiveness in this context.

4.3 Attention Visualisation

In particular in relation to adding additional features like the weekend indicator for energy data and buy or sell signals for BTC prices the analysis and visualization of the attention mechanism provided an important insight. Furthermore, the attention was also analyzed on their distribution, on their periodicity and finally a closer inspection how the attention reacts to different attention-head configuration. During the weekend period the model type “Weekend” shows a slightly higher attention value percent of 42.40% than the basic model type of 17.64%. Overall the impact of the additional features are not that high as in the hypotheses proposed. On the other hand these features does not significantly change the attention or shift them. They have some influence on it but not that dramatically as the hypotheses predicted. Another point that was pointed out and explained was how the attention is distributed over a variation of the number of

attention heads. It has been shown that while the head number increases the attention itself becomes more uniformly distributed. This was done by calculating the standard deviation of the attention itself and comparing it with the results from different head counts. A more spiky attention was exhibited by models with a single head configuration. So a larger number of heads leads to an even and widespread attention distribution. For the energy data set a separate analysis was done focused on how the attention is periodic for a clear periodic input sequence in the model. The study showed a strong connection between the periodicity in the data and the attention itself. This was supported by the calculation of the R^2 score and compared it with the periodicity form the input data.

Especially for the BTC data set a separate analysis was done to check how the attention reacts to significant price changes. This was done for the single-head configuration, where the hypothesis said, that in these areas the attention should be more concentrated. The results show that the attention focuses more on the region where there is a positive price change especially for shorter input ranges. This is not true for a negative price change. A separate analysis was done for two-head model configuration,which focuses on the attention at the beginning and the end of the input sequence. The hypothesis was that if the attention is concentrated in these areas then there should be a significant price change. The results show with help from a non-parametric Spearman's rank-order correlation test a small positive correlation for the following input sequence of 24.

4.4 Trading Simulator

Guided by the rules of the Binance platform a trading simulator was designed to test the predicting performance from the Basic Transformer, 3EMA Transformer, and SARIMA model. The results also gives an insight how these models can perform in a real situation, by reading market trends and making profitable trades. At the end of the simulation of all three models produced a negative profit indicating that the models were not robust and could not read and learn the non-linearity in the Bitcoin data. A heatmap shows with the additional classification scores a dramatic missclassification between a long and short position and vice versa. With a given leverage, this is a significant downfall to produce positive profit. The total loss for the trading simulation is for the two Transformer based models €200 and for the SARIMA model €80.

4.5 Future Research

A future extension to the Transformer model could be to train it with dynamic numbers of attention heads. With this new technique a meta learning framework could create where the model learns to choose multiple input characteristics to achieve a reliably good performing output. This approach using a meta-learning method was tested for a general model [7]. They introduced a dynamic attention which could also increase the performance of the prediction for financial data.

To increase the performance of the trading simulation additional input data could be provided such as newsletters form all over the world. The investigation of other positions with the same correlation of the BTC data could also improve trading performance. By combining the Transformer with a SARIMA to create a hybrid model this approach could forecast the non-linear behavior of finance data.

CHAPTER 5

Conclusion

This thesis showed that the Transformer architecture is also capable of predicting energy values and BTC data prices, and is not only limited to NLP tasks or translation or speech translation. It also investigated various Transformers, LSTM, and SARIMA models for predicting energy consumption and close prices, and their effectiveness in a trading simulation. Additionally, this thesis gives an insight into the attention mechanic from a Transformer based on visual inspection for different approaches.
By varying the number of heads the SARIMA model showed the lowest RMSE score overall, the Transformer was close to the performance followed by the LSTM model for the energy consumption data set. It loses some performance compared to the others due to lesser dynamic adaptability. The performance for the Transformer model is closer to the SARIMA model for the BTC close price data set. The LSTM model could not handle the non-linear finance data and produces a non-optimal output with no movement. A deeper analysis shows that adding additional features such as weekend indicator only has only a minimal impact on the attention for the Transformer model. On the other hand by increasing the number of attention-heads, the distribution of the attention itself becomes more evenly distributed. If there exists any strong periodicity pattern in the source data itself the attention pattern mirrors this behavior too. This behavior also has a minimal impact on the number of heads. The trading simulator shows that all three models such as Basic Transformer, 3EMA Transformer and SARIMA model performed poorly in terms of generating profit after a trading period. Starting with 1000€ for each model, the simulation ends with a negative profit generation of 200€ for the two Transformer based models and 80€ for the SARIMA model. This finding opens the way for future research to improve the performance in financial market trends.

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] Brendan Artley. Time series forecasting w/ ARIMA, SARIMA. 2023. Accessed: 2023-02-09. https://www.kaggle.com/code/brendanartley/time-series-forecasting-w-arima-sarima.

[2] Kirolos Atef. Stock prediction using Twitter sentiment analysis. Accessed: 2023-02-09. https://www.kaggle.com/code/kirolosataallah/stock-prediction-using-twitter-sentiment-analysis.

[3] Jason Brownlee. Data preparation for variable length input sequences. In *Machine Learning Mastery*, 2023. Accessed: 2024-06-04. https://machinelearningmastery.com/data-preparation-variable-length-input-sequences\-sequence-prediction.

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *arXiv preprint arXiv:1406.1078*, 2014. https://arxiv.org/abs/1406.1078.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *arXiv preprint arXiv:1810.04805*, 2019. https://arxiv.org/abs/1810.04805.

[6] Amir Razghandi et al. Short-term load forecasting in smart homes: A sequence-to-sequence learning approach. In *arXiv preprint arXiv:2106.15348*, 2021. https://arxiv.org/abs/2106.15348.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017. https://arxiv.org/abs/1703.03400.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT Press, 2016. Accessed: 2024-05-31. https://www.deeplearningbook.org/.

[9] Jake Grigsby, Zhe Wang, Nam Nguyen, and Yanjun Qi. Long-range transformers for dynamic spatiotemporal forecasting. In *arXiv preprint arXiv:2109.12218*, 2023. https://arxiv.org/abs/2109.12218.

[10] Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, and Wangyou Zhang. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, page 449–456, 2019. https://arxiv.org/abs/1909.06317.

[11] Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, and Wangyou Zhang. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, page 449–456, 2019. https://arxiv.org/abs/1909.06317.

[12] Prasoon Kottarathil. BTC-USD historical data. 2023. Accessed: 2023-07-13. https://www.kaggle.com/datasets/prasoonkottarathil/btcinusd.

[13] Y. Liu, Z. Zhang, Y. Wang, and et al. Trading with the momentum transformer: An intelligent and interpretable architecture. In *arXiv Preprint*, 2021. https://arxiv.org/abs/2112.08534.

[14] Y. Liu, Z. Zhang, Y. Wang, and et al. Wftnet: Exploiting global and local periodicity in long-term time series forecasting. In *arXiv Preprint*, volume 2309.11319, 2023. https://arxiv.org/abs/2309.11319.

[15] Kiran Madhusudhanan, Johannes Burchert, Nghia Duong-Trung, Stefan Born, and Lars Schmidt-Thieme. Yformer: U-net inspired transformer architecture for far horizon time series forecasting. In *arXiv preprint arXiv:2110.08255*, 2022. https://arxiv.org/abs/2110.08255.

[16] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *arXiv preprint arXiv:1704.02934*, 2017. https://arxiv.org/abs/1704.02934.

[17] Clayton Miller. Building data genome project v1. 2023. Accessed: 2023-02-22. https://www.kaggle.com/datasets/claytonmiller/building-data-genome-project-v1.

[18] Mary Phuong and Marcus Hutter. Formal algorithms for transformers. In *arXiv preprint arXiv:2207.09238*, 2022. https://arxiv.org/abs/2207.09238.

[19] Jimeng Shi, Mahek Jain, and Giri Narasimhan. Time series forecasting (TSF) using various deep learning models. In *arXiv preprint arXiv:2204.11115*, 2022. https://arxiv.org/abs/2204.11115.

[20] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler. Efficient transformers: A survey. In *Journal of Machine Learning Research*, volume 21, pages 1–45, 2020. https://arxiv.org/abs/2009.06732.

[21] Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *arXiv preprint arXiv:1905.05950*, 2019. https://arxiv.org/abs/1905.05950.

[22] Pankaj Valuence. Time series forecasting LSTM, FBProphet, Transformer. 2023. Accessed: 2023-02-10. https://www.kaggle.com/code/pankajvaluence/time-series-forecasting-lstm-fbprophet-transformer.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, page 6000–6010, 2017. https://arxiv.org/abs/1706.03762.

[24] Jesse Vig. Visualizing attention in transformer-based language representation models. 2019. https://arxiv.org/abs/1904.02679.

[25] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. 2020. https://arxiv.org/abs/2001.08317.

[26] Haohan Zhang, Zhiwei Li, Jingrui Wang, et al. Deep Hawkes process for high-frequency market making. In *arXiv Preprint*, volume 2109.15110, 2021. https://arxiv.org/abs/2109.15110.