IM1102-232433M - Deep Neural Engingeering assignment 2

Modifying the attention mechanism of transformers for time series forecasting

Arne Lescrauwaet - Joachim Verschelde - Alexander Van Hecke

April 25, 2024

Introduction

This report details the steps taken by Arne Lescrauwaet (852617312), Joachim Verschelde (852594432) and Alexander Van Hecke (852631385) for the second assignment of the 2023 Deep Neural Engineering course organised by the Open University (1).

For this assignment, we explore different attention mechanisms for time-series-based transformers (2). The attention mechanism allows a transformer model to selectively focus on relevant parts of the input data, capturing long-range dependencies and relationships between items. This is particularly important for time series data that contains recurring patterns, such as hourly traffic counts or hourly power consumption. By capturing the "local context" of recurring patterns, we can use this information to forecast future events. Specifically, when encountering an event similar to a past event, we consider the outcome of that past event in our prediction.

Multiple kinds of attention mechanisms exist for transformer-based time-series forecasting. Convolutional self-attention is introduced in (3), which aims to capture the local context of input events, but does this using a symmetric convolution, thereby taking both input data leading to a particular event and the outcome of that event into account. A dual-stage attention mechanism is used for a Recurrent Neural Network (RNN) architecture in (4), using an input attention mechanism in an encoder step, and a temporal attention mechanism in a decoder step.

Even though transformers were originally designed in the field of Natural Language Processing (NLP), a lot of work has been done to use transformers with time series data. An overview of the different approaches can be found in (5). Tashreef et al. introduced a transformer-based approach using the time2vec encoding mechanism (6). The authors of this paper use transformer models to predict stock prices, and claim these models can be used both for short and long term predictions. Finally the effectiveness of using a transformers-based approach to forecast time series data is tested in (7).

The original transformer architecture introduces a quadratic time and space complexity $O(L^2)$. Much work has been done to overcome these bottlenecks. A novel LogSparse transformer architecture is introduced in (3), which reduces the memory cost to $O(L(\log L)^2)$. The informer model (8) even achieves a time complexity of $O(L\log L)$. In this report we will soley focus on attention mechanisms in the context of time series forecasting, ignoring the space and time complexity bottleneck of the transformer algorithm.

Goal

In this paper, we focus on time-series forecasting using transformer-based approaches. We aim to compare different attention mechanism and determine which mechanism best captures the outcome of past events. We formulate a first research question:

> **RQ 1 : When comparing the different attention mechanisms, which mechanism best predicts future values using root mean square error (RMSE) as metric?**

The Elia dataset we use (see the dataset description section for more info) contains both measurements and predictions for the next day and the next week. We formulate a second research question:

> **RQ 2 : Is the RMSE of a transformer model better than the Elia prediction model?**

To start off, we will look at the characteristics of the dataset used and discuss pre-processing steps. Then, we will consider several attention mechanisms, discuss design and implementation details and finally evaluate the performance of these attention mechanisms on the dataset.

Data analysis

Dataset description

We use data from Elia (9), which operates the electricity transmission network in Belgium. In particular, we use the solar power forecast dataset, which contains measurements of solar power in megawatt (MW) grouped per 15 minutes starting from February 2013 to February 2024. The measured value is the running average of the amount of power during these 15 minutes. Next-day and next-week predictions are available as well. The layout of the dataset is fully described here (10). We recap the most important points in Table 1.

Table 1: Features captured per 15 minutes in (10)

feature	description	range
DateTime	Date and time per 15 minutes	[00:00 - 24:00]
Measurement	Measured solar power production in MW	[0.0 - 6000.0]
Next-day prediction	Next-day solar power forecast in MW	[0.0 - 6000.0]
Next-week prediction	Next-week solar power forecast in MW	[0.0 - 6000.0]

Data general properties

The target column is not normally distributed but highly regular and characterized by a distinct day-night recurring pattern, where a local minimum occurs at night and a local maximum occurs during the day. This is illustrated in Figure 11.

Furthermore there are obvious differences in solar power generation between summer months and winter months, but the general pattern remains the same, as illustrated in Figure 12.

Data pre-processing

The Elia data (10) is very fine grained and contains 96 measurements per day, resulting in around 2880 measurements per month. To keep the dataset reasonably small we added the possibility to aggregate this dataset. Possible choices are **(i)** no aggregation, **(ii)** hourly aggregation, **(iii)** 4-hourly aggregation, and finally **(iv)** aggregation per day. Aggregation is done by averaging the values in the selected timeframe.

All data starting from February 2013 was taken into account, in order to maximize the possibility of finding interesting patterns in the data. The sequence length L has to be chosen carefully in basic transformer architectures because of the quadratic time complexity. With too few measurements, it will be difficult to spot similar events in the past. On the other hand too many measurements would make it infeasable to train and evaluate the model in a reasonable amount of time due to the transformers quadratic nature. A sequence length of 24 in combination with hourly aggregation would result in every sequence containing the information of one full day.

Outlier analysis

A visual analysis of outliers yielded no abnormal or obviously erroneous values. This outcome was expected and thus no values were discarded.

Methodology and Implementation

Research methodology

Given a basic transformer architecture, we implemented a number of attention mechanisms. Because of our limited computational resources, it was not possible to perform an automated hyperparameter

sweep such as GridSearch or RandomizedSearch. Therefore we manually evaluated the different models by varying some hyperparameters (see Table 3). A fixed set of hyperparameter and aggregation values was then used for the rest of the experiments.

Data up to but not including 2020 was used for training. The period from 2020 up to but not including 2021 was used for validation, and finally the test-set contained all data starting from 2021. All input data was scaled using a `MinMaxScaler` to normalize the values to $[0, 1]$. Models were first trained on the training dataset and then validated on the validation dataset for a maximum of 100 epochs. An early stop was forced if the average validation error of the running epoch exceeded the minimum average validation error 5 consecutive times, as this indicates the validation error was no longer decreasing. Each model was then tested on the test set and all losses (training losses, validation losses and test losses) were kept for later analysis. In all cases, RMSE was used as the loss metric.

Design elaboration

We decided to implement and evaluate the following attention mechanisms (Table 2) :

- regular self-attention (AM-1). This is the mechanism described in the original transformer paper (2).
- convoluted self-attention as described in (3) (AM-2). This mechanism generalizes the regular self-attention mechanism and uses a 1D convolution to transform the Query (Q) and Key (K) values before using them in the transformer architecture.
- right padded convoluted self-attention (AM-3). This is a variation of the mechanism described in (3). Whereas (3) uses a symmetric convolution, here we use a convolution that focuses on the right hand side to transform Q and K values before using them in the transformer architecture. Padding to the right is done to prevent looking at future values. The intuition behind this mechanism is that it focuses more on the outcome of past events than regular convoluted self-attention.
- Fast fourier transform based self-attention (AM-4). The input sequence can be seen as a periodic function. Our novel approach will first transform the periodic function from the time domain into the frequency domain using a FFT (Fast fourier transform). The intuition behind this is that during the self-attention 2 vectors will be similar if they can be decomposed into similar sine and cosine functions, which would mean their shape is similar.

Table 2: Attention mechanisms

attention mechanism	abbreviation
regular self-attention	AM-1
convoluted self-attention	AM-2
right padded convoluted self-attention	AM-3
fast-fourier-transform self-attention	AM-4

Transformer models have several tunable hyperparameters. We first experimented with variations of number of layers, number of heads, levels of forward expansion, convolution kernel sizes and aggregation levels of input data for the **base transformer model**, as detailed in Table 3. This yielded very small differences in average validation loss (see Table 8). Given our limited computational resources, we decided to fix the number of layers to 2, the number of heads to 4, the forward expansion to 256 and the aggregation to "1 day". For the models using a convolution (AM-2 and AM-3), we experimented with kernel sizes of [3, 6, 9]. In all scenarios, RMSE was used as the measure to optimize for.

Table 3: Hyperparameters

attention mechanism	hyperparameters
AM-1	layers [2, 4, 6], heads [4, 8], forward expansion [256, 512], aggregation [1day, 4 hours]
AM-2	layers [2], heads [4], forward expansion [256], aggregation [1day], kernel size [3, 6, 9]
AM-3	layers [2], heads [4], forward expansion [256], aggregation [1day], kernel size [3, 6, 9]
AM-4	layers [2], heads [4], forward expansion [256], aggregation [1day]

The implemented early stopping mechanism was activated at different epochs for the different attention mechanisms, see Table 4.

Table 4: Number of epochs trained

attention mechanism	number of epochs
AM-1	13
AM-2	10
AM-3	13
AM-4	29

Feature embedding was done using a combination of both positional encoding and a more specific temporal encoding, taking into account hour of the day, day of the week, day of the month and month of the year of the data. This temporal encoding could be learned by the network in the same way as the informer model (8) . The temporal encoding was added to the input vector and served as global context for the transformer model.

Implementation

All code and data is available in a github repository (11). All deep learning models were implemented using the pytorch python package, visualisation was done using matplotlib and seaborn. We recap the most important files here :

- `building_blocks.py` contains all pytorch modules and models, the different attention mechanism, input embedding and other support code to execute different scenarios.
- `datasets.py` contains the code to load and aggregate the Elia data into one pytorch dataloader.
- `figures.ipynb` is a jupyter notebook that contains code to generate figures.
- `result-statistics.ipynb` is a jupyter notebook that contains code to check statistical validity and generate figures for RQ1.
- `elia_predictions_vs_timeseries_transformer.ipynb` is a jupyter notebook that contains code to check statistical validity and generate figures for RQ2.
- `stats.ipynb` is a jupyter notebook that contains code generate tables about statistics.
- `scenario-runner.ipynb` is a jupyter notebook focusing on the execution of scenarios. In this notebook, model and scenario parameters are created, models are instantiated, trained, validated and finally tested against the test set. All results (losses and model weights) are saved to disk for later processing (statistics and figure generation).

Evaluation and Results

Evaluation

Evaluation of RQ 1

As discussed in the design elaboration, we evaluated different values of hyperparameters to get a sense of which hyperparameters would work well enough without taking too much in terms of computational resources. For all attention mechanisms, we settled on 2 layers, 4 attention heads and a forward expansion of 256 (see Figures 13, 16, 17, 18, 19, 21, 20, 14, 15 for validation losses). Using these (minimal) settings yielded sufficient results and limited compute time. For the models using convolutions, we saw a difference between AM-2 where a kernel size of 9 seemed to yield the best results, and AM-3 where a kernel size of 3 seemed to yield the best results. This is summarized in Table 5.

Table 5: Used hyperparameters

attention mechanism	labels	heads	forward expansion	kernel size
AM-1	2	4	256	N/A
AM-2	2	4	256	9
AM-3	2	4	256	3

attention mechanism	labels	heads	forward expansion	kernel size
AM-4	2	4	256	N/A

In order to evaluate the first research question, we formulated the following H_0 hypothesis :

> **H_0 : The performances of the four attention mechanisms (AM-1, AM-2, AM-3, AM-4) are equal.**

To validate whether the performances of the attention mechanisms were equal, we ran a one-way ANOVA test. If the p-value of this test was below $\alpha = 0.05$, we could reject H_0 and accept the alternative hypothesis, that at least one of the attention mechanisms has a performance that differs from the other attention mechanisms. If the one-way ANOVA test showed a difference in performance between the groups, we wanted to use a post hoc Tuckey HSD test to compare the mutual differences between the groups.

Evaluation of RQ 2

Given the good results of AM-4 for RQ1, we wanted to investigate whether AM-4 would be as good as the Elia model. In order to evaluate the second research question, we formulated the following H_0 hypothesis :

> **H_0 : The performance of the fourier attention mechanism (AM-4) is equal to the performance of the Elia model.**

If the p-value was below $\alpha = 0.05$, we could reject H_0 and accept the alternative hypothesis, that there is indeed a difference between the performance of the transformer based predictions and the Elia predictions. Note that Elia provides no details on their prediction model, so this is in fact a comparison between the Elia prediction model and a transformer based prediction model.

Results

Results of RQ 1

One-way ANOVA assumptions (normality and homogeneity of variances) were checked on the test losses. First, normality checks were done both graphically (Figure 1) and using a shapiro test (Table 6). Both the graphical test and the shapiro test indicate the fourier data is not normally distributed. However, the graphical test shows it is close to being normally distributed, and one-way ANOVA should be robust to small deviations from normality.

Table 6: One-way ANOVA normality check

attention mechanism	test value	p-value
AM-1	0.9473	0.0938
AM-2	0.9885	0.9686
AM-3	0.9879	0.9604
AM-4	0.9270	0.0228

The homogeneity of variances was checked using a Levene's test. This showed variances were not equal (test value = 9.9881, p-value = 5.4035e−6). Therefore we concluded a one-way ANOVA test could be used, yielding an F-stat value of 310.7705 and a p-value of 2.4661e−38, indicating the H_0 hypothesis should be rejected and we can conclude that at least one of the attention mechanisms has a performance that differs from the other attention mechanisms.

To investigate the differences between the attention mechanisms, we conducted a post hoc Tuckey HSD test, see Table 7. This test shows the biggest differences in mean RMSE for AM-4 compared to the other attention mechanisms. Note that the Tuckey test in the `statsmodels` python package reports the mean difference of group2 - group1, so a negative value indicates a lower mean RMSE for group 2 compared to group1. AM-1 and AM-3 seem to differ very little in their performance.

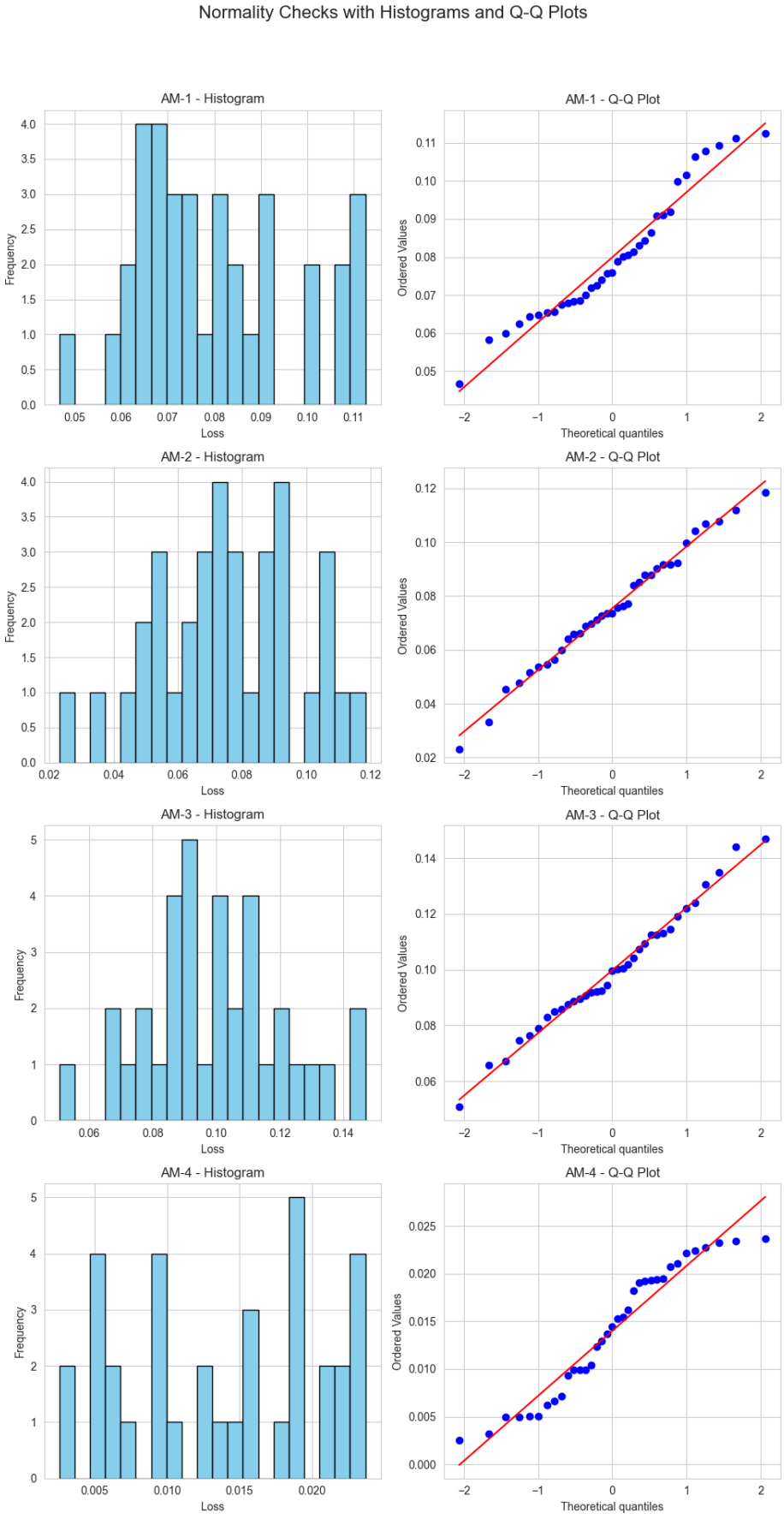

Figure 1: One-way ANOVA normality check

Table 7: Tuckey test

group1	group2	meandiff	p-adj	lower	upper	reject
AM-1	AM-2	-0.0164	0.0036	-0.0286	-0.0042	True
AM-1	AM-3	0.0079	0.332	-0.0043	0.0202	False
AM-1	AM-4	-0.0778	0.0	-0.09	-0.0656	True
AM-2	AM-3	0.0243	0.0	0.0121	0.0366	True
AM-2	AM-4	-0.0614	0.0	-0.0737	-0.0492	True
AM-3	AM-4	-0.0858	0.0	-0.098	-0.0735	True

As can be seen in Figure 2 and Figure 3 the fourier attention mechanism showed the best performance on the test and validation dataset.

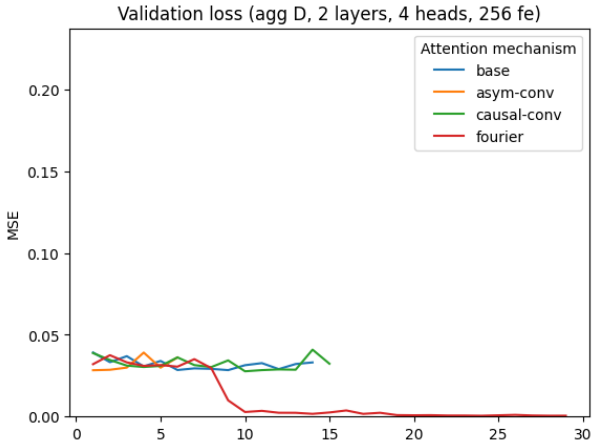

Figure 2: Validation loss comparison

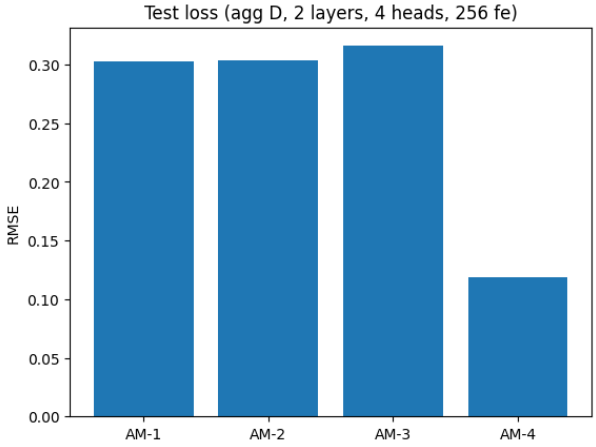

Figure 3: Test loss comparison

To get an indication about where attention is focused, we visualised the attention focus for a random input sequence for each of the different attention mechanisms. Figure 4 illustrates this for AM-1, Figure 5 illustrates this for AM-2, Figure 6 illustrates this for AM-3 and Figure 7 illustrates this for AM-4.

Results of RQ 2

We compared 150 forecasted values of the fourier model to 150 forecasted values of the Elia model. A shapiro test was done to check normality for both fourier and Elia residuals. Neither seemed to be normally distributed, indicating a non-parametric test should be used. A Wilcoxon signed-rank yielded a test value

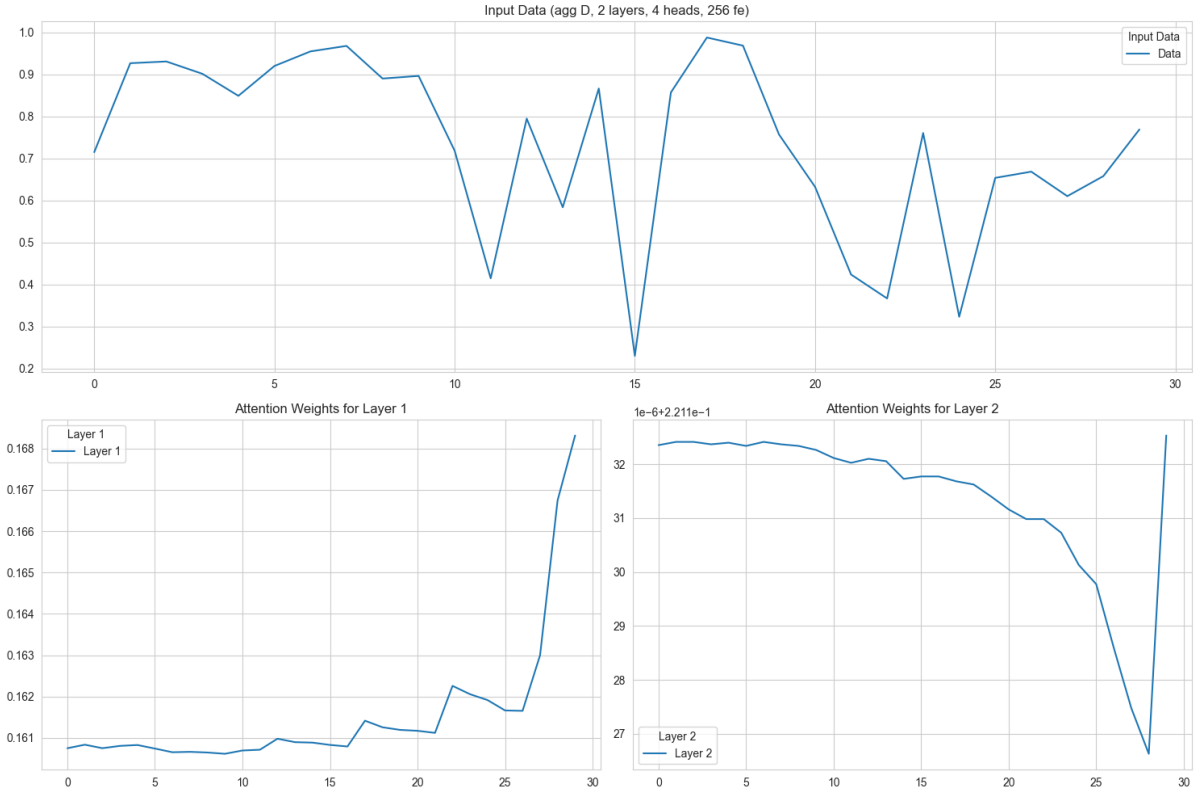

Figure 4: Attention visualisation for AM-1

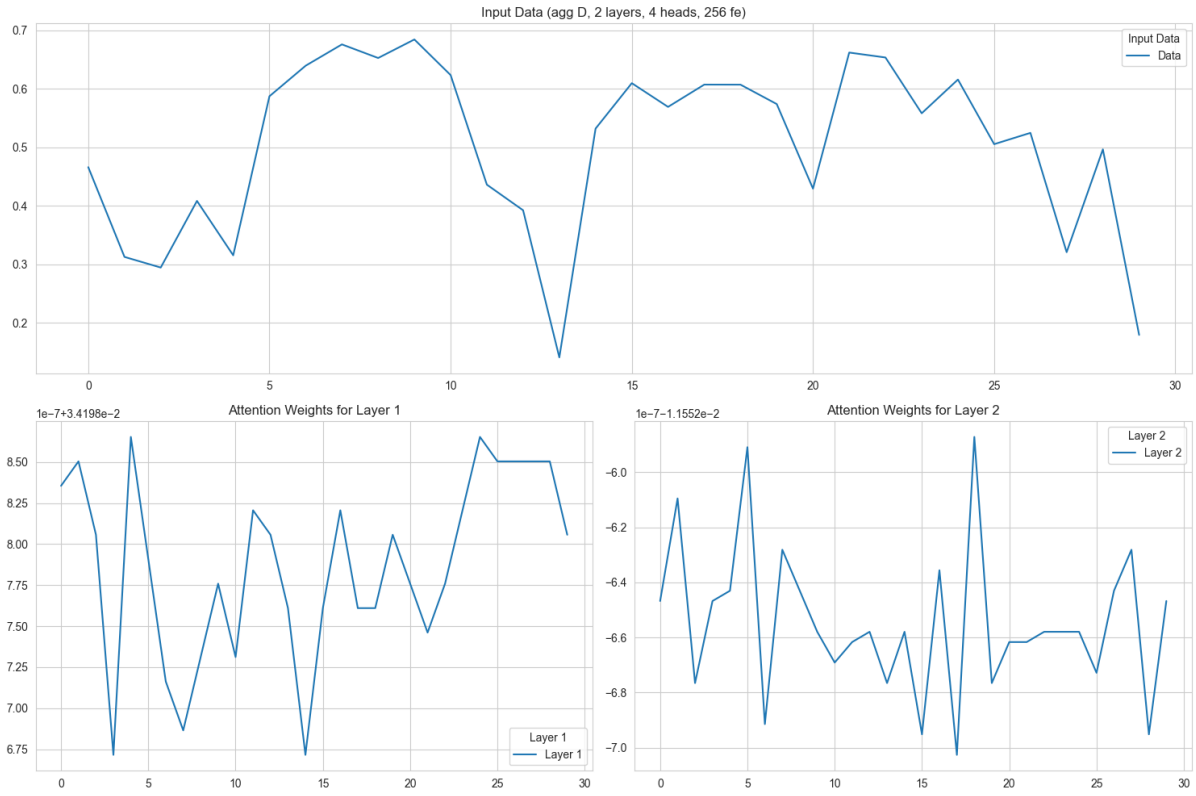

Figure 5: Attention visualisation for AM-2

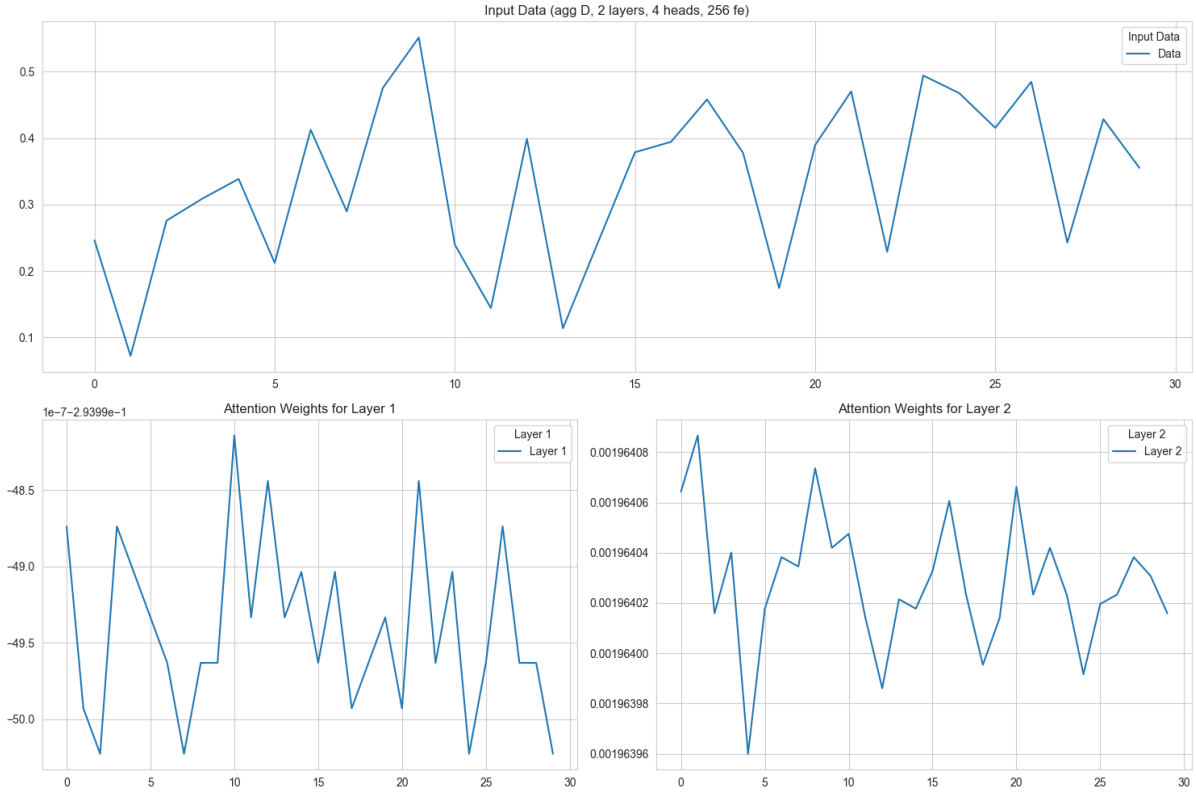

Figure 6: Attention visualisation for AM-3

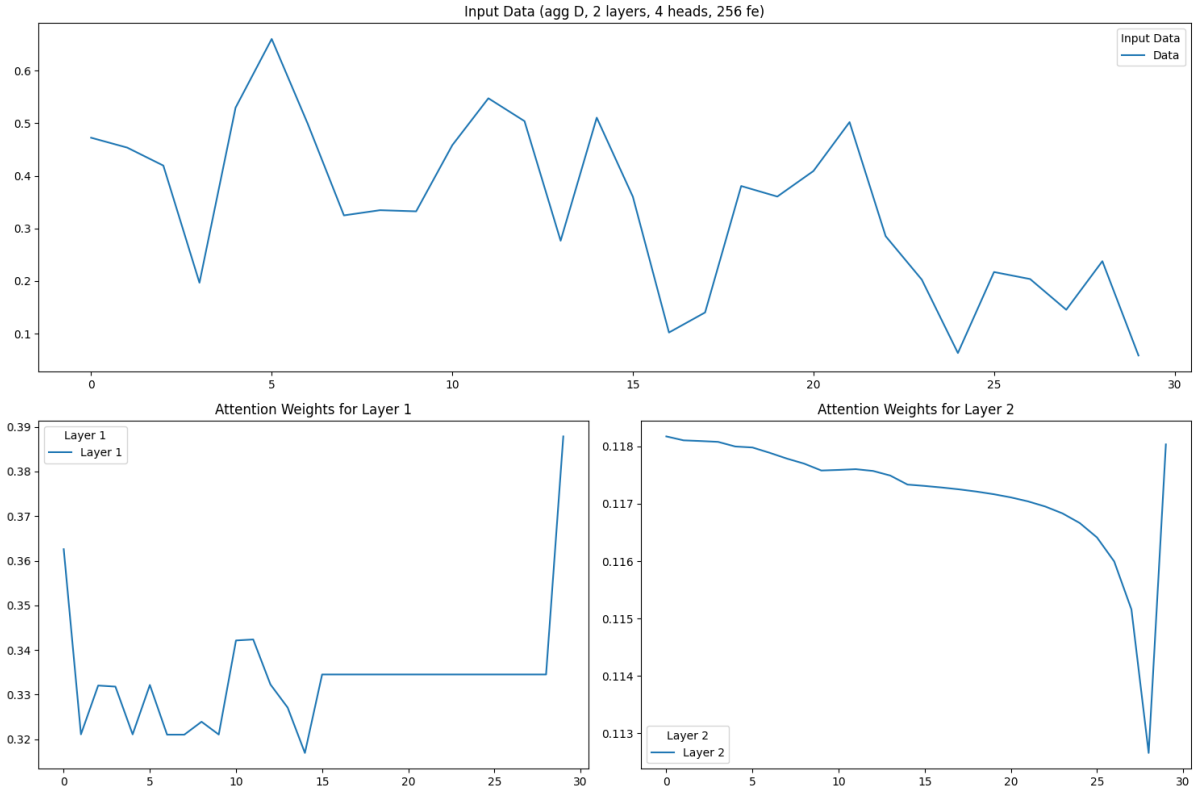

Figure 7: Attention visualisation for AM-4

of 2990699 and a p-value < 0.0001, indicating the H_0 hypothesis should be rejected and we can conclude that the performance of the fourier model is not equal to the performance of the Elia model. This is obvious when looking at Figure 8. The Elia model shows predicted values very close to actual values, whereas the fourier model shows larger gaps between actuals and predictions. A Cohen's d value of 0.7080 confirms the effect size is large.

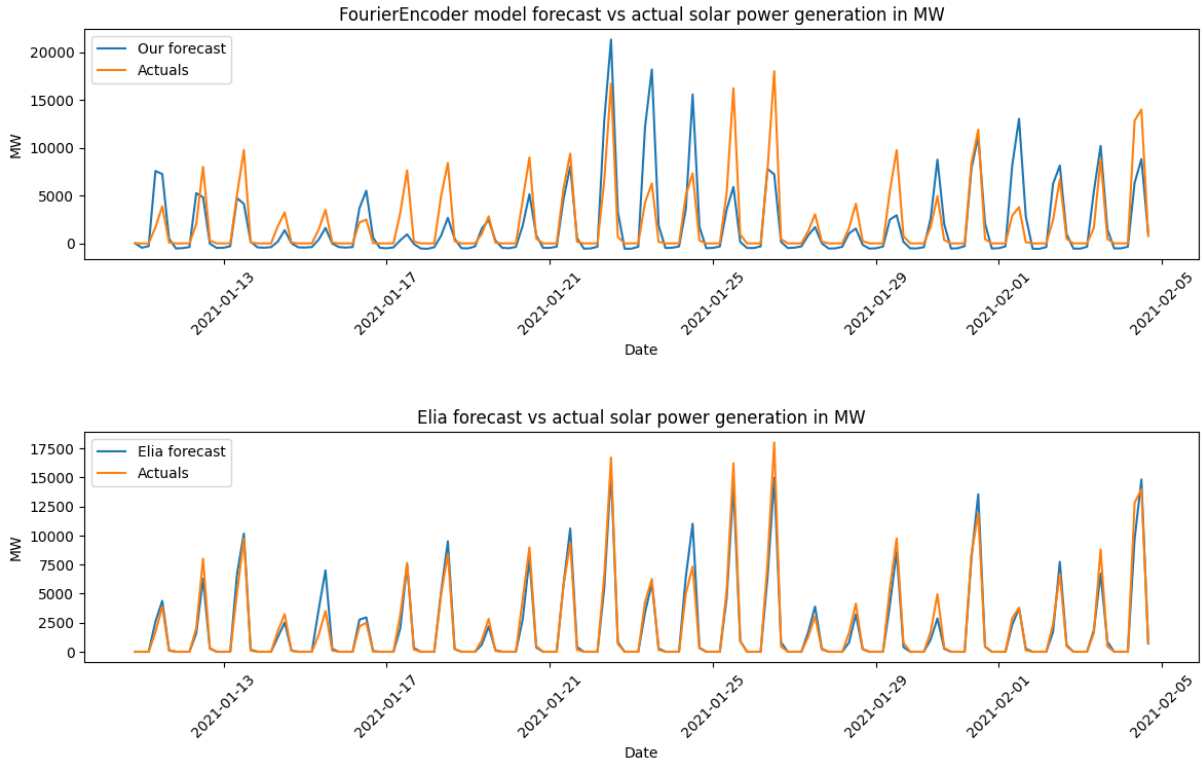

Figure 8: Performance of AM-4 vs performance of Elia

Plotting (partial) autocorrelation functions for both fourier (Figure 9) and Elia (Figure 10) residuals supports the same conclusion : the Elia model has better performance on this particular dataset.

Conclusions and Discussion

In this paper, we analysed the role of attention mechanisms in transformer models for forecasting values of timeseries data. Several attention mechanisms were evaluated, **(i)** regular self-attention, **(ii)** convoluted self-attention, **(iii)** right padded convoluted self-attention and **(iv)** fast-fourier-transform self-attention. Input data of Elia, the Belgian electricity transmission network operator, was aggregated to daily values and our models generated a next-day forecast. First, we did a mutual comparison of the different attention mechanisms. Second, we compared the next-day forecast of our transformer models with the (proprietary) next-day forecast of Elia.

In evaluating the different attention mechanisms, we implemented a modular and composable base transformer architecture. This allowed us to only vary the input encodings for each attention mechanism, leaving the rest of the architecture unchanged. The fourier based input encoding clearly yields the best results in our test setup, outperforming causal convolution 5 times.

Comparing the forecast of the transformer based models to the forecast of Elia did not yield good results. The predictive model of Elia (no details are published about this model) is clearly better than our transformer based approach. We can only speculate about the cause of this, but we must mention **(i)** our limited computational resources and number of epochs trained, **(ii)** our fixed set of hyperparameters due to these constraints and **(iii)** the maturity of the Elia model.

We see several opportunities for future work. First, we have only evaluated the different attention mechanism on one type of dataset (solar power measurements). This used dataset contained highly

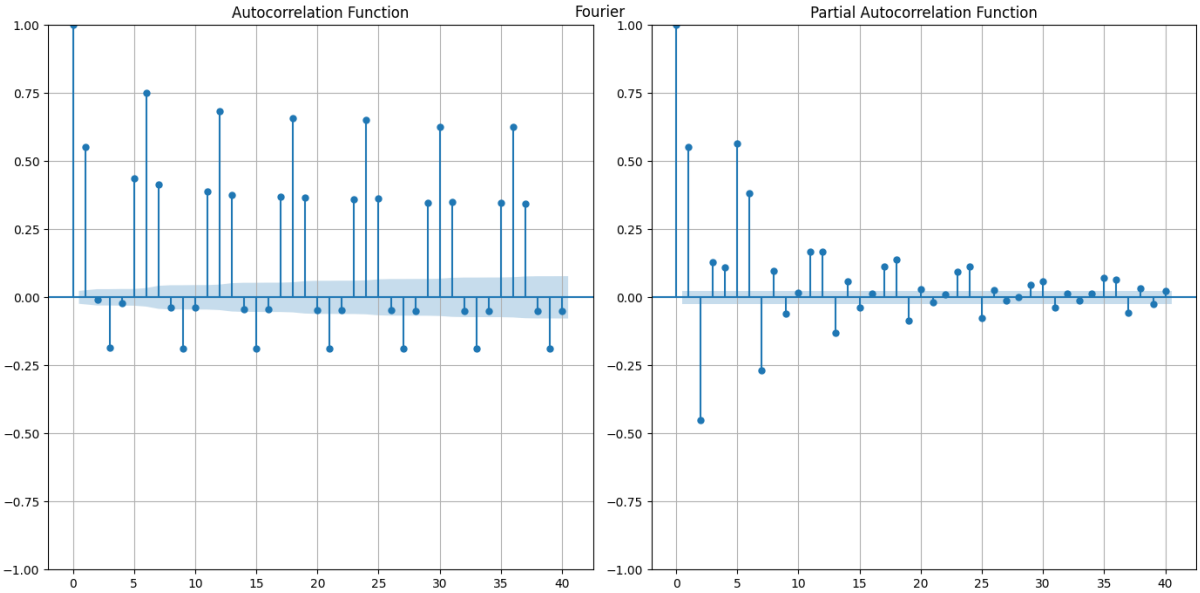

Figure 9: (P)ACF of AM-4

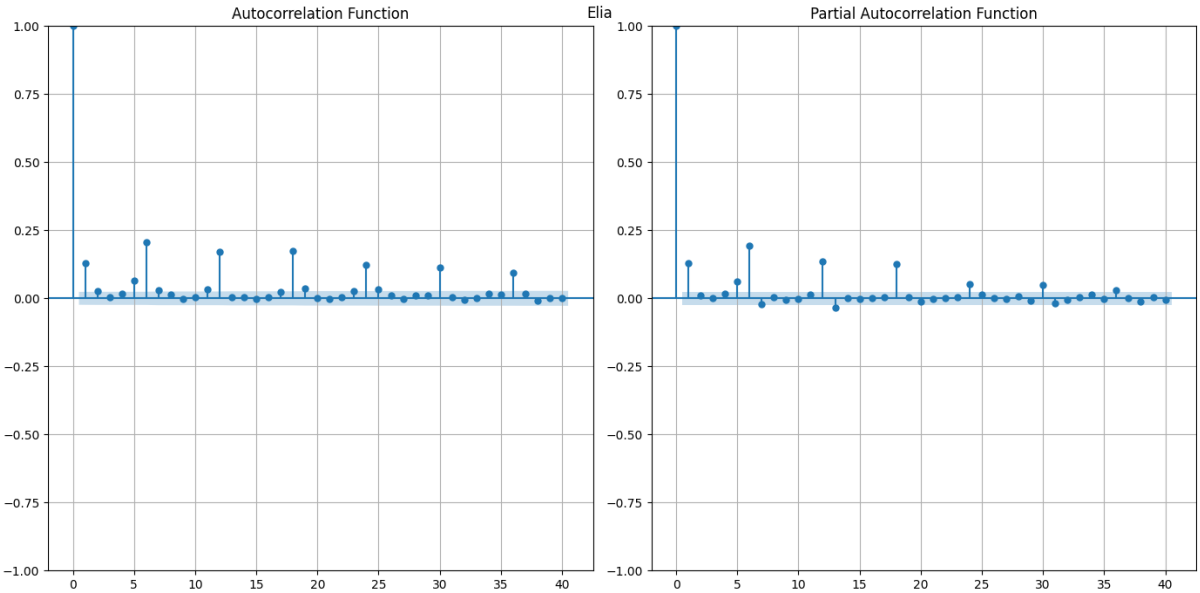

Figure 10: (P)ACF of Elia

regular, cyclical data. It would be interesting to see whether the same results can be obtained on other types of (less regular) timeseries data like stock market prizes. Second, given that **(i)** the fourier input encoding seemed to be very efficient in our tests and **(ii)** this input encoding can be used in established transformer architectures, it would be interesting to see if there are benefits in using fourier input encoding in established transformer architectures. For example we could use a renowned Timeseries Transformer model sourced from Huggingface and test it's performance on a prediction task with vs without performing a FFT on the input embeddings.

References

1. Startpagina - IM1102-232433M - Deep Neural Engineering [Internet]. [cited 2024 Mar 21]. Available from: https://brightspace.ou.nl/d2l/home/8636
2. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All you Need. In: Advances in Neural Information Processing Systems [Internet]. Curran Associates, Inc.; 2017 [cited 2024 Mar 6]. Available from: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
3. Li S, Jin X, Xuan Y, Zhou X, Chen W, Wang YX, et al. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. In: Advances in Neural Information Processing Systems [Internet]. Curran Associates, Inc.; 2019 [cited 2024 Mar 10]. Available from: https://proceedings.neurips.cc/paper/2019/hash/6775a0635c302542da2c32aa19d86be0-Abstract.html
4. Qin Y, Song D, Chen H, Cheng W, Jiang G, Cottrell GW. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence [Internet]. Melbourne, Australia: International Joint Conferences on Artificial Intelligence Organization; 2017 [cited 2024 Mar 6]. p. 2627–33. Available from: https://www.ijcai.org/proceedings/2017/366
5. Wen Q, Zhou T, Zhang C, Chen W, Ma Z, Yan J, et al. Transformers in Time Series: A Survey [Internet]. arXiv; 2023 [cited 2024 Mar 7]. Available from: http://arxiv.org/abs/2202.07125
6. Muhammad T, Aftab AB, Ibrahim M, Ahsan MdM, Muhu MM, Khan SI, et al. Transformer-Based Deep Learning Model for Stock Price Prediction: A Case Study on Bangladesh Stock Market. International Journal of Computational Intelligence and Applications [Internet]. 2023 Sep [cited 2024 Mar 16];22(03):2350013. Available from: https://www.worldscientific.com/doi/full/10.1142/S146902682350013X
7. Cholakov R, Kolev T. Transformers predicting the future. Applying attention in next-frame and time series forecasting [Internet]. arXiv; 2021 [cited 2024 Mar 16]. Available from: http://arxiv.org/abs/2108.08224
8. Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, et al. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting [Internet]. arXiv; 2021 [cited 2024 Mar 7]. Available from: http://arxiv.org/abs/2012.07436
9. Elia: Belgian's Electricity System Operator [Internet]. Elia. [cited 2024 Mar 30]. Available from: https://www.elia.be/en/
10. Solar-PV power generation data [Internet]. [cited 2024 Mar 23]. Available from: https://www.elia.be/en/grid-data/power-generation/solar-pv-power-generation-data
11. Hecke AV, Lescrauwaet A, Verschelde J. Avhou/dne [Internet]. 2024. Available from: https://github.com/avhou/dne

Appendix A : Data general properties

Appendix B : Average validation losses (base transformer)

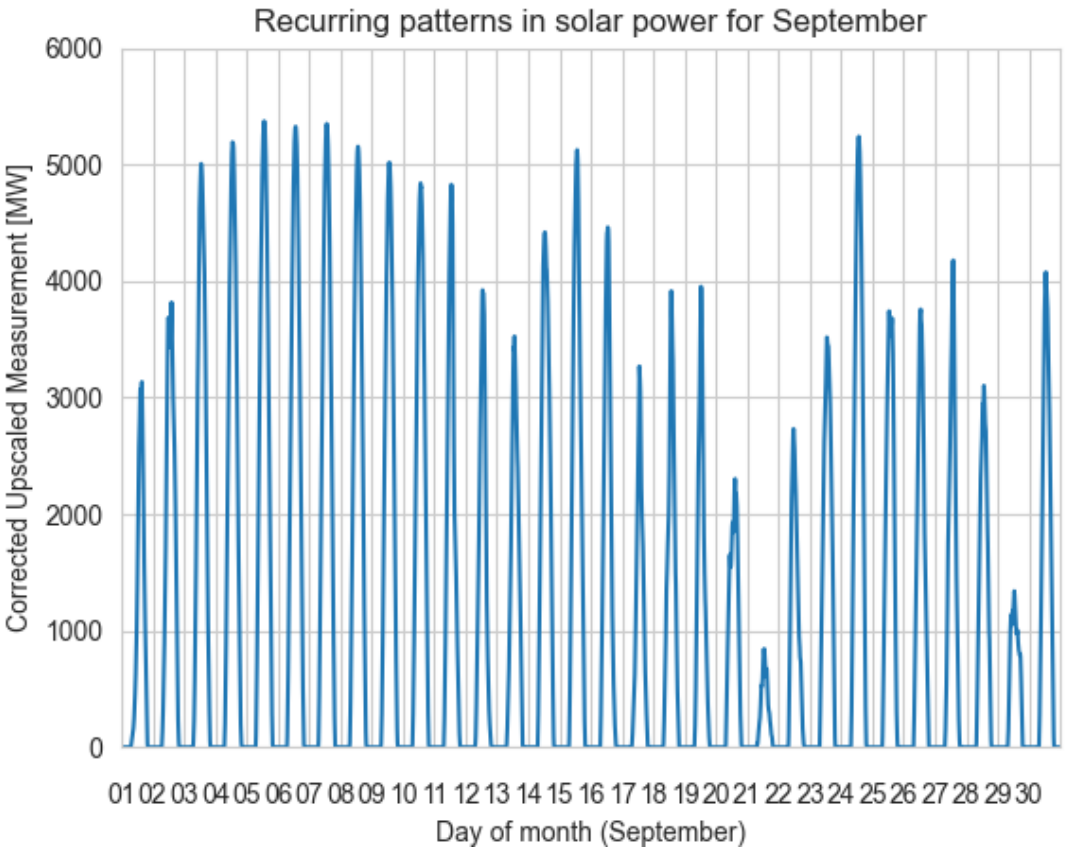

Figure 11: Typical recurrent patterns, here for September 2023

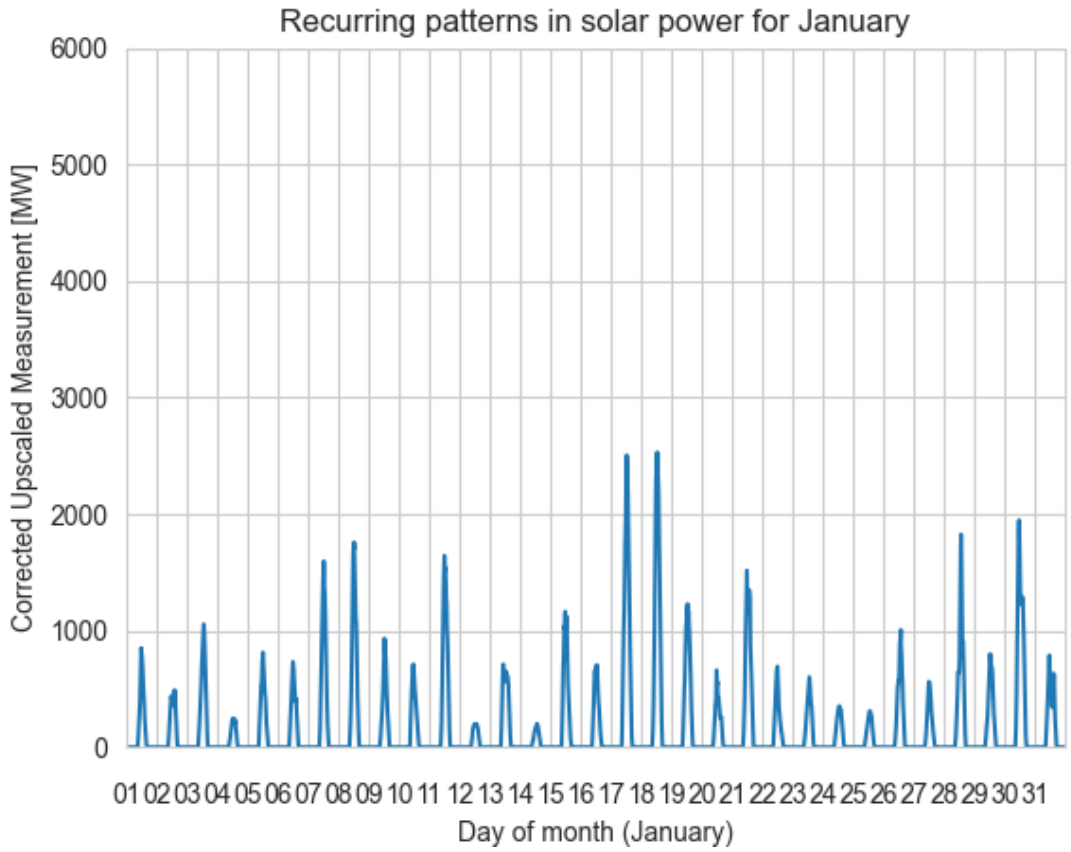

Figure 12: Typical recurrent patterns, here for January 2023

Table 8: Average validation losses for base transformer hyperparameter variations

layers	heads	forward expansion	average	stddev
2	4	256	0.0336	0.0036
2	8	256	0.0316	0.0023
4	4	256	0.0324	0.0032
4	8	256	0.0350	0.0055
6	4	256	0.0332	0.0031
6	4	512	0.0353	0.0046
6	8	256	0.0339	0.0035
6	8	512	0.0336	0.0047
2	4	256	0.0059	0.0087

Appendix C : Hyperparameter experimentation validation losses

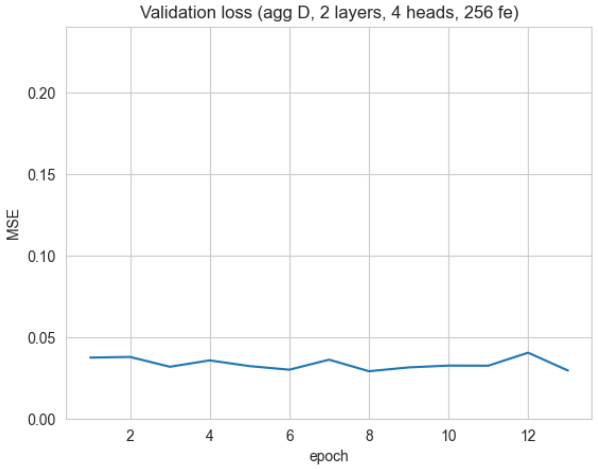

Figure 13: Validation loss AM-1, 1 day aggregation, 2 layers, 4 heads, 256 forward expansion

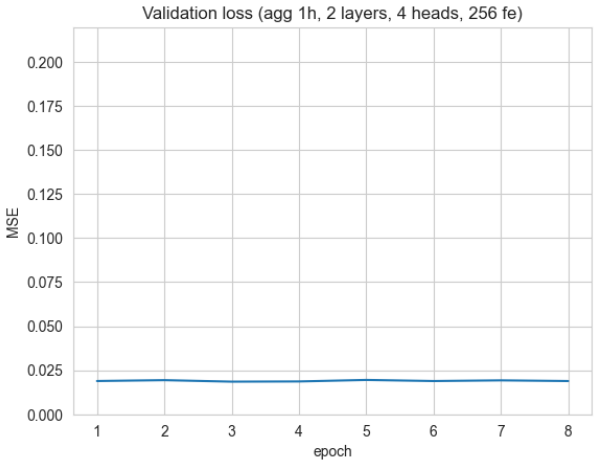

Figure 14: Validation loss AM-1, 1 hour aggregation, 2 layers, 4 heads, 256 forward expansion

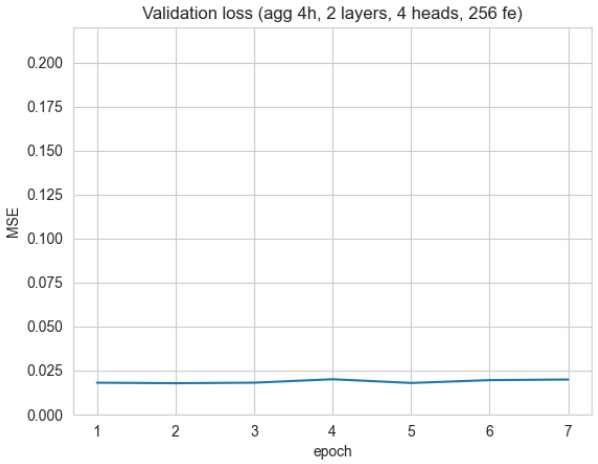

Figure 15: Validation loss AM-1, 4 hour aggregation, 2 layers, 4 heads, 256 forward expansion

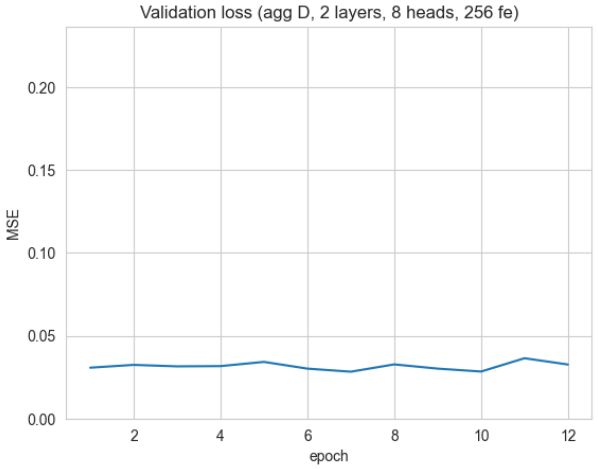

Figure 16: Validation loss AM-1, 1 day aggregation, 2 layers, 8 heads, 256 forward expansion

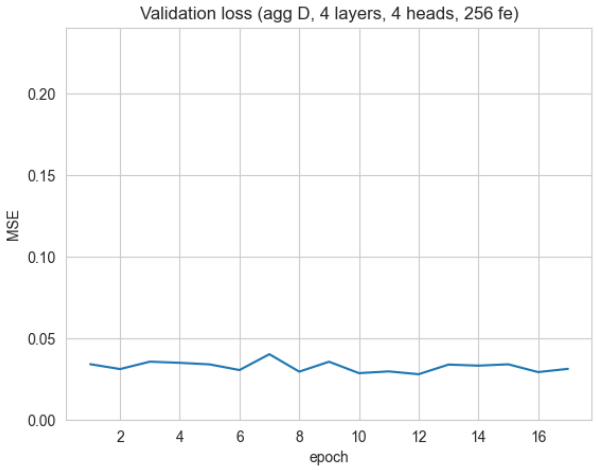

Figure 17: Validation loss AM-1, 1 day aggregation, 4 layers, 4 heads, 256 forward expansion

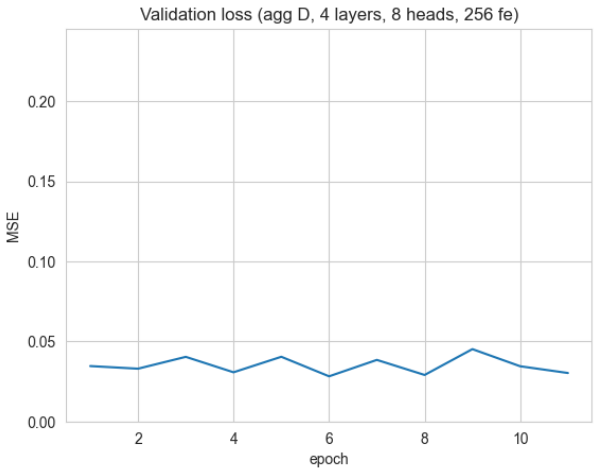

Figure 18: Validation loss AM-1, 1 day aggregation, 4 layers, 8 heads, 256 forward expansion

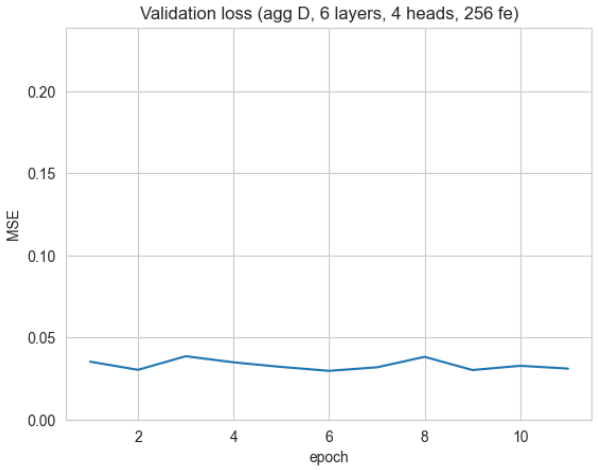

Figure 19: Validation loss AM-1, 1 day aggregation, 6 layers, 4 heads, 256 forward expansion

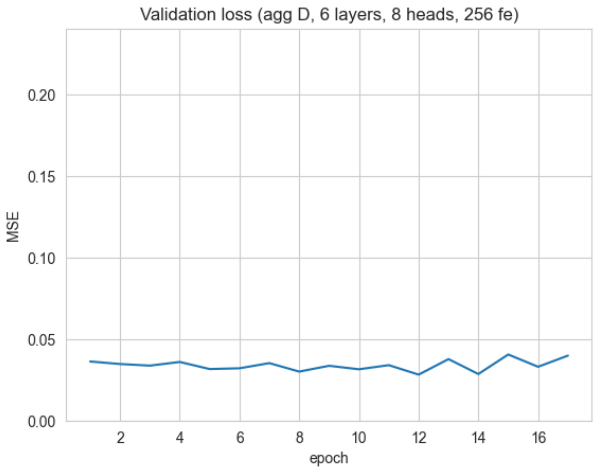

Figure 20: Validation loss AM-1, 1 day aggregation, 6 layers, 8 heads, 256 forward expansion

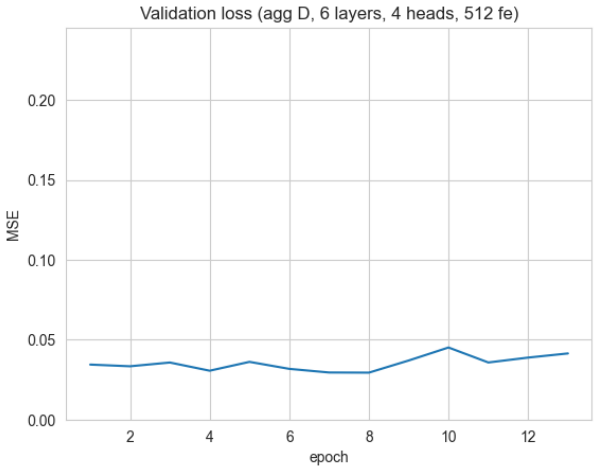

Figure 21: Validation loss AM-1, 1 day aggregation, 6 layers, 4 heads, 512 forward expansion