Trading Hub Europe Task

John Lyons
john.lyons@stud.th-rosenheim.de
Faculty of Computer Science, Technical University of Applied Sciences
Rosenheim, Bavaria, Germany

Florian Weiss
florian.weiss@stud.th-rosenheim.de
Faculty of Computer Science, Technical University of Applied Sciences
Rosenheim, Bavaria, Germany

Maximilian Weber
maximilian.weber@stud.th-rosenheim.de
Faculty of Computer Science, Technical University of Applied Sciences
Rosenheim, Bavaria, Germany

Benedict Schwind
benedict.schwind@stud.th-rosenheim.de
Faculty of Computer Science, Technical University of Applied Sciences
Rosenheim, Bavaria, Germany

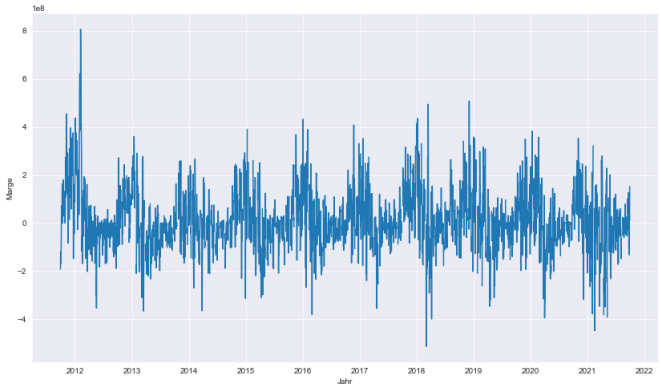

Figure 1: Time series

ABSTRACT

This article describes the elaborated solution of the Trading Hub Europe Task of the team Aleph-Omega Analytics team. First, the data analysis, the seasonality of the time series and the correlation of the data are described. In this project a special sliding window approach was used to deal with the data gaps. The two final models for the 1 day (NNAR) and 2–5 day (LSTM) forecast are explained as well as the development process. Considerations for continuing the project such as replacing Facebook Prophet with its own model or method were taken.

KEYWORDS

time series forecasting, neural networks, challenge

ACM Reference Format:
John Lyons, Florian Weiss, Maximilian Weber, and Benedict Schwind. 2022. Trading Hub Europe Task. In *Proceedings of Bavarian-French Artifical Intelligence Challange (AI-Cup '22).* ACM, New York, NY, USA, 4 pages.

AI-Cup '22, July 11, 2022, Passau, BY
2022.

1 UNDERLYING TIME SERIES OF THE DATA SET

Fig. 1 shows the trend of a time series. The underlying data, which are also used in the course of this work for the evaluation of the different models, are from "Trading Hub Europe". It is clear to see that there is a certain periodicity in the data set. Thus, it seems that there is an annual seasonality and no continuous trend. On closer analysis, there is even a weekly seasonality. This is shown by the autocorrelation function and the analysis using the "statsmodels" library in Fig. 2.

The problem implicitly involves forecasting gas consumption in industry and households. The main task of Trading Hub Europe is to balance the gas budget in the gas network. Surplus gas has to be fed out and missing gas has to be fed in due to the physical capacity of the network.

The data is provided in a 24-hour interval and the main attributes of the dataset are:

- margin
- SystemBuy
- SystemSell
- NK balance
- BK balance

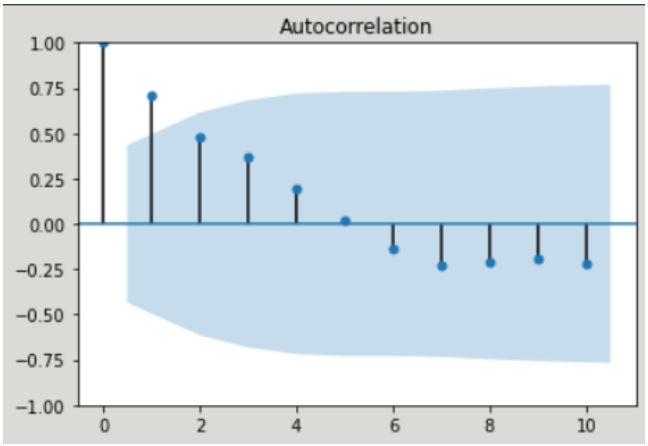

Figure 2: Autocorrelation function

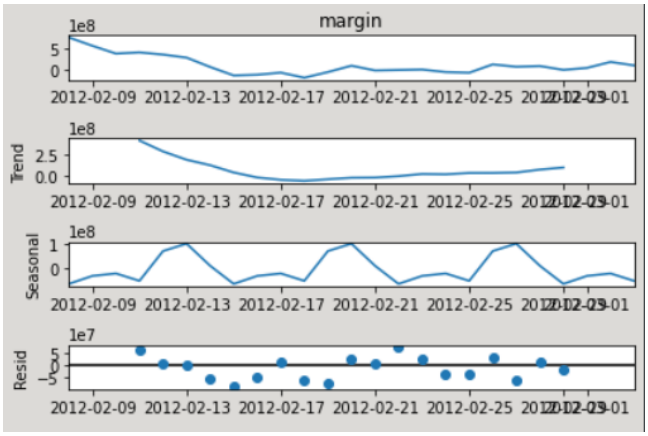

Figure 3: Trend and seasonality in the time series

Here, *margin* is directly composed of *SystemBuy* and *SystemSell*. *SystemBuy* is the amount of gas bought in total at the end of the day and *SystemSell* is the amount of gas sold at the end of the day. Both of these attributes can be non-zero at any point in time (i.e., the end of the day). This is the case because balancing sometimes has to be done several times a day, and therefore it may happen that both bought and sold was done in one day.

If it were known in advance how much gas would ultimately have to be traded at the end of the day, it would be possible to make clever purchases at the most favorable times on the previous days and thus save money.

The difficulty of predicting these data can be justified even without exploratory data analysis. This is because the prediction of the *margin* attribute is based on other exogenous attributes, such as *NK balance* and *BK balance*, and these attributes are already based on external predictions, which can no longer be influenced. Thus, *BK balance* is the difference between the actual consumption of large households (industry, etc.) and the external prediction. Therefore, the model (if it takes into account the exogenous variables) must implicitly make a prediction for an estimate, which proves difficult.

2 DATA PREPARATION FOR NEURAL NETWORKS

In contrast to the aforementioned classical methods, the data set requires fewer adjustments when it is prepared for a neural model. The reason for this is that the high number of parameters and the non-linear activation functions can model very complex time series.

Unlike, for example, ARIMA or XGB, (a well-known machine learning technique for classification and regression, but which can also be applied to time series), multiple data points including all attributes can easily be passed to the model.

For this purpose the so-called "Sliding Window" method is used. From a sorted set M with N elements an input length $input\_width$ and an output length $output\_width$ is defined. Fig. 4 shows the dataset as tiles as an example.

Pieces of length $n = input\_width + output\_width$ are then cut out, always starting at the beginning of the sorted set and selecting the next initial element based on the step size $stride$. Fig.4 shows how two training data points are produced from the six training data. Here $N = 6$, $stride = 1$, $input\_width = 3$, $output\_width = 2$ and thus $n = 5$.

Finally, for the days to be predicted, all features are removed except for those that are to be used by the model in the prediction, such as SLP. For the label data, obviously only the target value margin remains.

Since the time series always has missing values in regular time steps, this must be handled specifically. One possibility for handling gaps is to remove the six missing days in each case. However, then the time sequence of the daily data is disturbed as soon as the resulting time series is concatenated again. Thus, let the time series $X = (x_1, x_2, x_3, ..., x_n)$ be the sequence of daily data. Now, if, for example, x_2 is removed, x_1 is immediately followed by x_3. However, since the training data contains only partially perturbed sequences (since there is a gap less sequence of 389 days to begin with), the model is largely trained with inconsistent data. This is because if the model draws a relation to the day x_{t-i} with respect to the day x_t to be predicted during training, this relation is no longer valid as soon as the day x_{t-i} is replaced by another day like x_{t-i-1}.

Using the example of the underlying time series, the lag $input\_width = 2$ is chosen for a potential model. Let the day to be predicted be a Saturday, the model takes the values from Thursday and Friday to predict the desired day. Obviously, an erroneous training data point occurs when Wednesday is padded (day $x_{t-i}, i = 2$) instead of Thursday (day x_{t-i-1}).

Another way to handle the gaps that have no value for the target variable and other important attributes is (linear) interpolation. However, the problem that arises is the systematic bias of the training data, since the filled gaps do not contain real information.

Therefore, the only option is to remove the gaps from the training data set, but not concatenate the remaining subsequences. Let $X = (x_1, x_2, x_3)$ be a time ordered sequence of the target variable. Then the removal of x_2 gives rise to the two subsequences $X_1 = (x_1)$ and $X_2 = (x_3)$. The original sequence X can be transformed into a training data point by using the variables x_1 and x_2 as past values and x_3 as the target prediction. Obviously, the lag

here is $input\_width = 2$ and thus no more training data point can be produced from the subsequences X_1 and X_2.

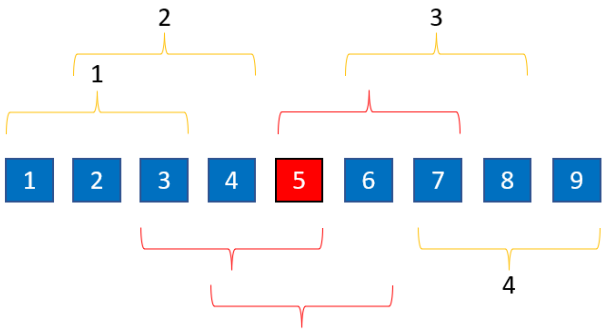

Figure 4: Amount of training data points after removal of intervening day without concatenation.

In the case of the underlying time series, 24 days are always given completely and are followed by six day gaps. Let $X = (X_1, X_2, X_3)$ be a sequence of sequences and let $X_1 = (x_0, ..., x_{24})$ be the sequence of 24 completely given days, $X_2 = (x_{25}, ..., x_{30})$ be six day gaps and $X_3 = (x_{31}, ..., x_{54})$ be another sequence of 24 completely given days. Then, removing X_2 gives rise to the implicit subsequences X_1 and X_3, each containing 24 elements. With a lag of $input\_width = 14$, a prediction period of $output\_width = 6$, a training data point must consist of exactly $input\_width + output\_width = 20$ days. To prevent a biased sequence, there must be no concatenation of the subsequences X_1 and X_3.Accordingly, training data points must be extracted individually from each X_1 and X_3 using the above sliding window procedure. Thus, with a number of $|X_1| = |X_3| = 24$ daily data, "only" $2 * [|X_1| - (input\_width + output\_width + 1] = 2 * (24 - 20 + 1) = 10$ training data points can be created. Compared to the number of training data points created by the sliding window method after interpolating the training data gaps, this is very little. Because here would be a total of

$$|X| - (input\_width + output\_width)$$

$$= |X_1| + |X_2| + |X_3| - (input\_width + output\_width) + 1$$

$$= 24 + 6 + 24 - 20 + 1 = 35$$

Training data points possible.

Thus, since interpolation as well as the removal of gaps without exception with subsequent concatenation distort the training data, only the last chosen approach remains. The only way to increase the number of training data points obtained is to reduce the number of past days as well as the number of days to be predicted.

Finally, the processed data points are loaded into tensors so that they can later be passed to the model for training and evaluation.

3 MODEL DESCRIPTION

At the beginning, different possible models were evaluated such as ARIMA, XGB and Facebook Prophet. These produced acceptable results and served as a basis for further action.

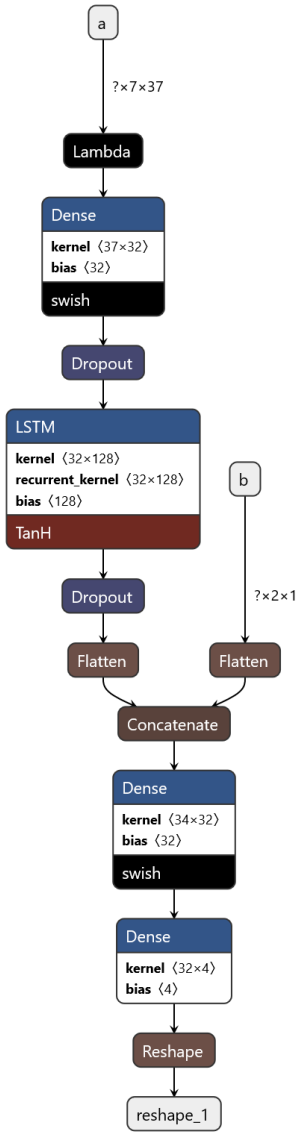

Figure 5: Model architecture LSTM with future.

In the following, experiments were performed using simple neural models such as LSTM, Convolution, and MLP [1]. Further development of the models by integrating "future" data (weather data and standard load profile for the 6-day forecast slots). From the knowledge gained, more complex models were integrated by combining dropout, LSTM, and dense layers. Fig. 6 shows the model for the first day forecast and Fig. 5 the model for the 2–5 day forecast. Testing various activation functions (swish, leakyRelu, selu) [2],

especially swish has to be emphasized [4]. Implementation of a custom loss function for an all in one model with a stronger weighting on the first day. The approach with a custom loss function yields worse results than two models for day first day and another one for days 2–5.

Grid searches for hyperparameter tuning were performed with the promising models. Due to the diversity of models and hyperparametric combinations, optimizers like like ADAM and RMSProp are not suitable (manual learning rate). Therefore the COCOB optimizer was used, because it finds the "best" learning rate independently [3]. A separate script was developed for the grid search, which ranks the different models. For improving the accuracy of the neural models the "Future Days" were predicted with ENTRY, EXIT, and RLM by Facebook Prophet [5].

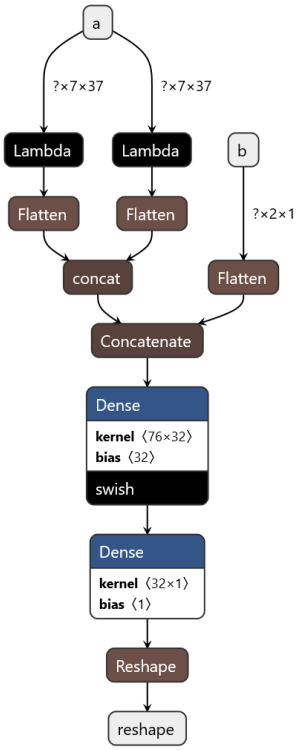

Figure 6: Model architecture NNAR with future.

4 RESULTS

The best result so far is shown in Fig. 7. The MAE for the first day is 53268428 and for days 2–5 64648796. A total score of 58958612 was achieved.

#	User	Entries	Date of Last Entry	Team Name	MAE for 1 day ahead foreasting ▲	MAE for days 2-5 ahead forecasting ▲	Overall Score ▲
1	jolyons123	4	07/09/22	Aleph-Omega Analytics	53268428.7 (1)	64648796.4 (1)	58958612.5 (1)

Figure 7: Result score

5 OUTLOOK

The next step is to integrate the full algorithm and the combination of Prophet and the neural models must be integrated into an end-to-end solution. With the goal of a simple push of a button to produce a prediction.

Some possible continuation options are listed below.

- To better include the volatility fluctuations of, for example, summer and winter to be included in the forecast.
- For longer forecasts, the use of a 2P method.
- Analysis of the residuals of the current models.
- Extraction of more auxiliary variables through architecture adjustments.
- Replacement of Prophet by another neural network or other method (Prophet was used only because it is easy and fast to implement).

ACKNOWLEDGMENTS

Special thanks to Prof. Dr. Breunig, Prof. Dr. Schmidt, Prof. Dr. Tilly and Prof. Dr. Wellisch by providing the lectures ML, DL, KS and thus preparing us for this challenge.

REFERENCES

[1] Seok-Jun Bu and Sung-Bae Cho. 2020. Time Series Forecasting with Multi-Headed Attention-Based Deep Learning for Residential Energy Consumption. *Energies* 13, 18 (2020). https://doi.org/10.3390/en13184722

[2] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. 2018. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *CoRR* abs/1811.03378 (2018). arXiv:1811.03378 http://arxiv.org/abs/1811.03378

[3] Francesco Orabona and Tatiana Tommasi. 2017. Backprop without Learning Rates Through Coin Betting. *CoRR* abs/1705.07795 (2017). arXiv:1705.07795 http://arxiv.org/abs/1705.07795

[4] Prajit Ramachandran, Barret Zoph, and Quoc Le. 2017. Swish: a Self-Gated Activation Function. (10 2017).

[5] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72, 1 (2018), 37–45.