Recurrent neural networks for time series forecast

Yutong He

1 Objective

This project applies recurrent neural networks (RNNs) to conduct time series analysis and produce forecast over different time spans. We will explore data features, develop RNN models with different network parameters, and present a comparison between the resulting forecast. The analysis uses a Kaggle dataset on Electric Power Consumption in Tetouan, Morocco, for the entire year of 2017. See this Jupyter Notebook for more details on the model development processes.
Time-series forecast could potentially provide significant values to many sectors, such as the energy industry, climate science, and stock market, where fluctuating variables such as electricity demand, weather patterns, and stock prices are difficult to predict.

1.1 Sub-tasks

Some anticipated tasks are as follows:

1. Data cleaning and exploratory analysis. This could entail dealing with missing entries or entries with invalid values, if any, and encoding non-numeric values to numeric ones, if necessary. Since this project concerns time-series forecast, we will likely reduce the overall data features down to one containing time variable and one indicating a target variable, such as energy demand.

2. We will visually inspect the correlations between different input features to gain a first-step intuitive understanding of the dataset, especially between the target/forecast variable and the time component.

3. RNN is often used for sequential data. We will build long short-term memory (LSTM) models, a type of RNN, to learn the time-series and produce forecast. Since this is a deep learning project, the framework will involve multiple layers with a large number of parameters specifying the network structure. This will in turn require trial and error to output a desirable forecast.

4. If possible, we will attempt to produce forecast covering different time ranges, from the short-term (days) to medium-long term (up to a year). This could be possible given the cadence of 10 minutes in the time-series dataset, which we will describe in section 2.

2 The time-series dataset

2.1 Data description

The initial dataset contains 52416 time stamps (rows) and 9 features (columns). All features, except datetime, are floats. These features reflect environmental variables such as temperature and humidity, as well as the energy consumption in three different zones in Tetouan. The time stamps starts from midnight 1/1/2017 till the end of that year, in intervals of 10 minutes. The header and the first 5 rows are shown in table 1.

Datetime	Temperature	Humidity	Wind Speed	General Diffuse Flows	Diffuse Flows	Zone1	Zone2	Zone3
1/1/2017 0:00	6.56	73.80	0.08	0.05	0.12	34055.70	16128.88	20240.96
1/1/2017 0:10	6.41	74.50	0.08	0.07	0.09	29814.68	19375.08	20131.08
1/1/2017 0:20	6.31	74.50	0.08	0.06	0.10	29128.10	19006.69	19668.43
1/1/2017 0:30	6.12	75.00	0.08	0.09	0.10	28228.86	18361.09	18899.28
1/1/2017 0:40	5.92	75.70	0.08	0.05	0.09	27335.70	17872.34	18442.41

Table 1: Dataset header. The last 3 columns are power consumptions in 3 different zones.

2.2 Feature exploration

We explore the dataset and perform the following tasks.

1. Usually a dataset should be checked for anomalies such as missing or invalid values. The dataset imported for this project contains no such anomalies.

2. We extracted some basic statistical information about the dataset, such as the means and value ranges for all columns, shown in table 2.

3. We visualize the pair-wise feature correlation via figure 1, and note that many variables are weakly correlated (bluer color gradients). Some features are more strongly correlated (redder gradients), such as energy consumptions in zones 1, 2, and 3. This makes sense as the zones are located in the same city and should therefore share similar energy demands.

	Datetime	Temperature	Humidity	Wind Speed	General Diffuse Flows	Diffuse Flows	Zone1	Zone2	Zone3
count					52416				
mean	182.00	18.81	68.26	1.96	182.70	75.03	32344.97	21042.51	17835.41
std	105.08	5.82	15.55	2.35	264.40	124.21	7130.56	5201.47	6622.17
min	0.00	3.25	11.34	0.05	0.00	0.01	13895.70	8560.08	5935.17
25%	91.00	14.41	58.31	0.08	0.06	0.12	26310.67	16980.77	13129.33
50%	182.00	18.78	69.86	0.09	5.04	4.46	32265.92	20823.17	16415.12
75%	272.99	22.89	81.40	4.92	319.60	101.00	37309.02	24713.72	21624.10
max	363.99	40.01	94.80	6.48	1163.00	936.00	52204.40	37408.86	47598.33

Table 2: Statistical description of the dataset. The last 3 columns are power consumptions in 3 different zones.

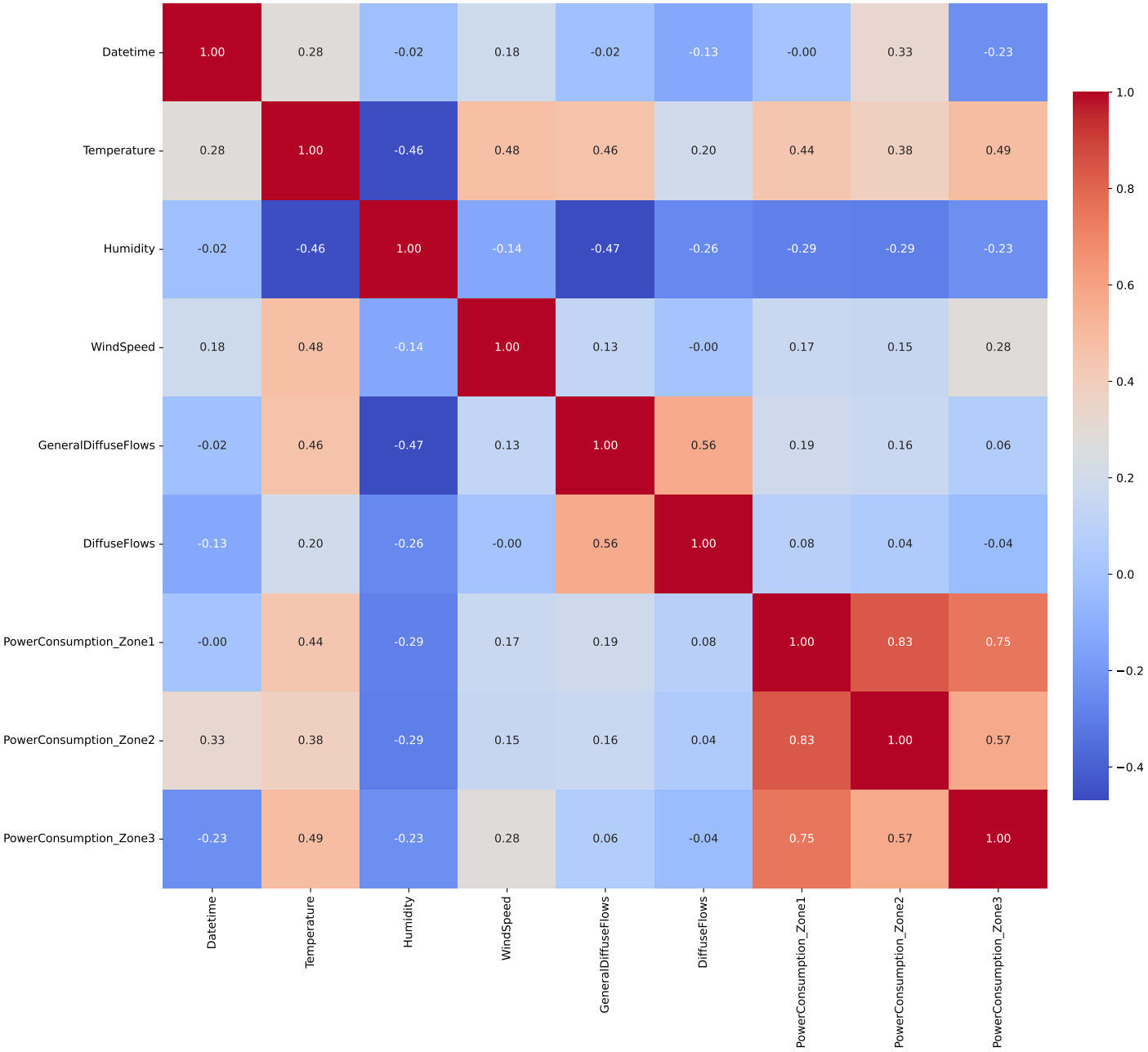

Figure 1: Heatmap showing the dataset features and their pair-wise correlations, where the color gradient indicates correlation strength.

2.3 Reduced data: focus on the energy consumption features

For this project, we focus on analyzing the time-series of energy consumption and leave the effects of environmental variables to a future study. To prepare for RNN models later, we extract the energy consumption from the last three columns of the dataset. Figure 2 illustrates the fluctuations of energy consumption throughout the year (three upper panels) and over the course of two days (three lower panels). We note that the fluctuations follow a similar pattern that repeats over the course of one day but the repetitive patterns are less visible on larger time scales. Without loss of generality, we choose to analyze zone 2 throughout. In section 3 we will test whether RNN-based forecasting models could capture these characteristics on different time scales.

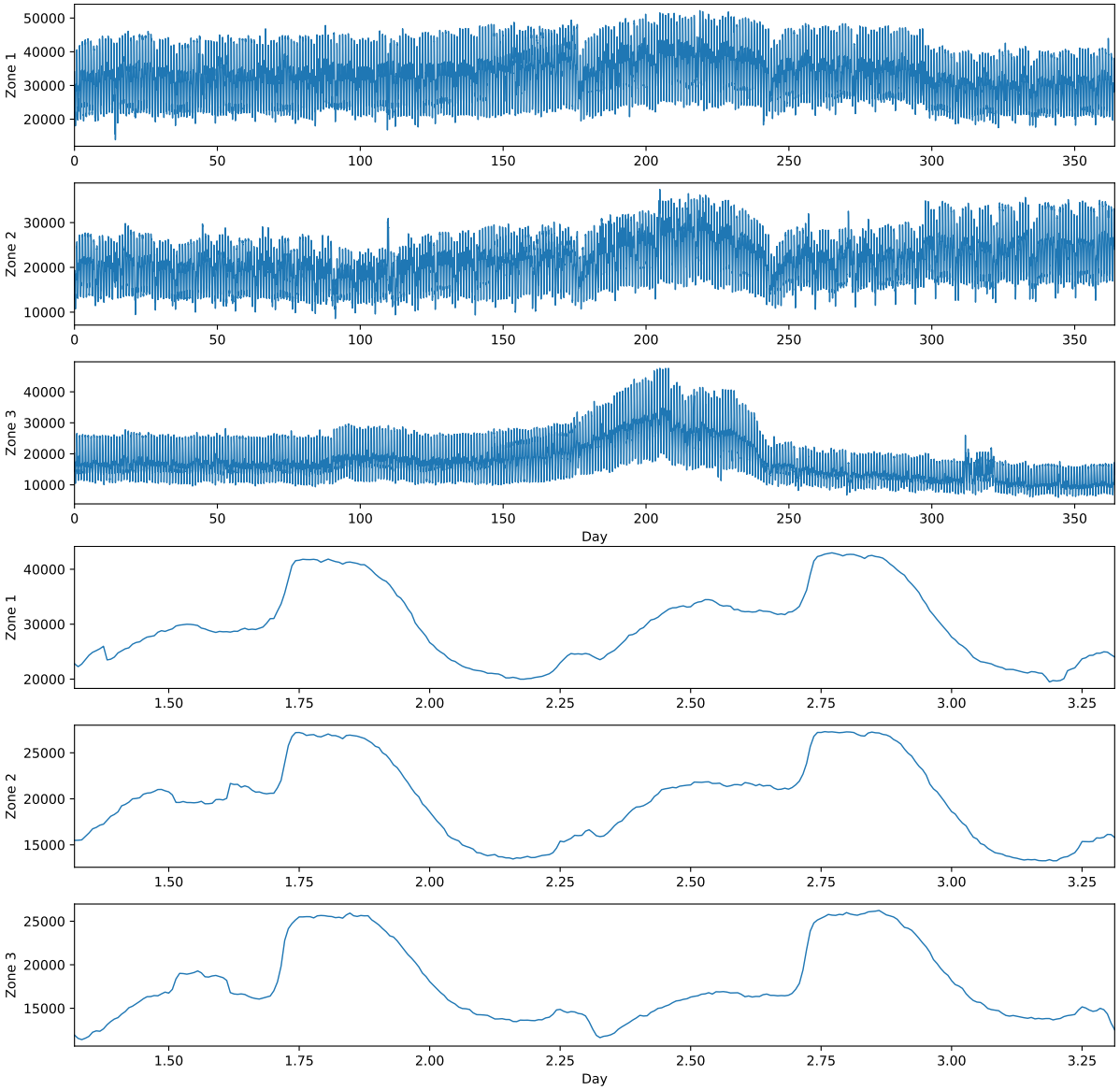

Figure 2: Time series of electric power consumption in three different zones over the course of one year (three upper panels) and one day (three lower panels).

3 Long short-term memory (LSTM) models

All models used in this project are based on the LSTM approach. LSTM is a type of RNN that contains a forget gate, controlling what past information gets to the next stage, in additional to adding new information at the current unit. It provides a way for the network to retain short-term memory and incorporate it in the weights adjustment process. We construct a LSTM model for each investigated time span: one day 3.1, two days 3.2, one week 7, one month 3.4, one year 3.5. The LSTM models for these different scenarios share a similar architecture but vary in the specific parameters. See the Jupyter Notebook for detailed model development.

3.1 Model 1: Training and forecast for one-day time series

After tuning the model parameters by hand, we settle down to the values summarized in figure 3. The resulting model output and performance are presented in figure 4.

Layer (type)	Output Shape	Param #
lstm_192 (LSTM)	(None, 1, 50)	10,400
dropout_63 (Dropout)	(None, 1, 50)	0
dense_257 (Dense)	(None, 1, 50)	2,550
dropout_64 (Dropout)	(None, 1, 50)	0
lstm_193 (LSTM)	(None, 1, 50)	20,200
dense_258 (Dense)	(None, 1, 10)	510
dropout_65 (Dropout)	(None, 1, 10)	0
lstm_194 (LSTM)	(None, 50)	12,200
dropout_66 (Dropout)	(None, 50)	0
dense_259 (Dense)	(None, 50)	2,550
dense_260 (Dense)	(None, 50)	2,550
dense_261 (Dense)	(None, 1)	51

Figure 3: Summary of the LSTM model used for one-day forecast, totaling 51011 parameters.

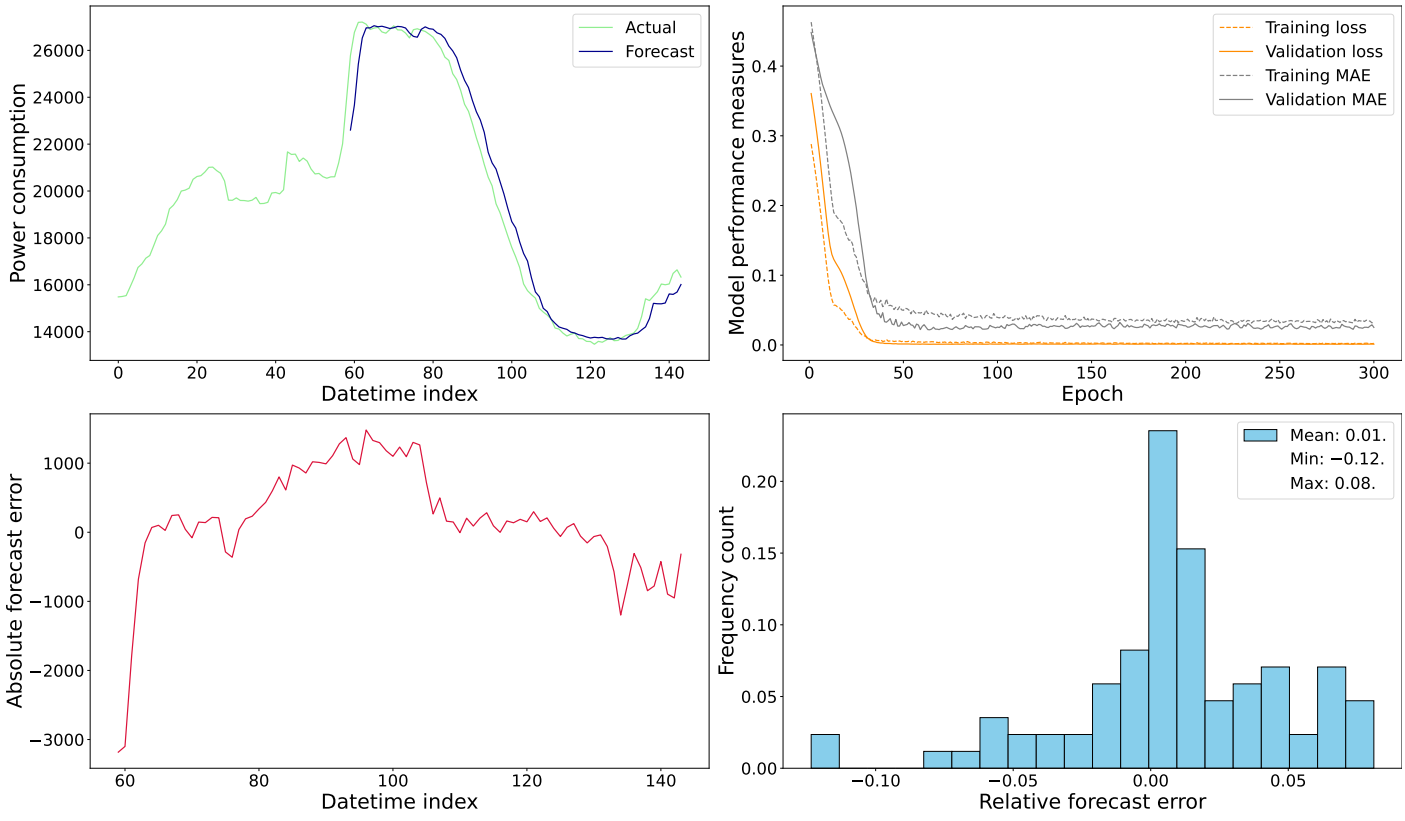

Figure 4: Summary of model output for the time period of one day. Upper left: actual (green) and predicted (blue) time series. Upper right: loss and MAE over the training epochs. Lower left: absolute forecast error over time. Lower right: frequency distribution of relative forecast error.

In figure 4, the upper left panel shows that the forecast power consumption roughly matches the trajectory of their actual values although the same pattern has not occurred before in the training set covering the period of one day. The model performance, measured in loss and mean absolute error (MAE), indicated in the upper

right panel of figure 4, significantly improves over the first ~ 50 epochs. It saturates afterwards but does not exhibit any overfitting tendencies. This is perhaps due to the inclusion of 4 dropouts with a rate of 0.1 each in the framework (see figure 3). Then the lower left panel of figure 4 shows the absolute error, i.e., forecast minus actual consumption, which varies between -3000 and 1000. To put the absolute errors into perspective, the lower right panel of figure 4 presents the distribution of the relative error, i.e., as a fraction of the actual value, and its distribution over different bins. The distribution has a mean of 0.01 and spans between -0.12 and 0.08. This aligns with the by-eye estimation from the lower left panel, as most of the error fluctuations occur around zero.

3.2 Model 2: Training and forecast for two-day time series

The reason to include a two-day scenario is that a repetitive pattern would appear in the dataset and it would be interesting to explore whether this could impact the learning outcome in any ways. We use the model parameters summarized in figure 5, and the learning results are presented in figure 6 with the same content structure as figure 4.

Layer (type)	Output Shape	Param #
lstm_219 (LSTM)	(None, 2, 50)	10,400
dropout_99 (Dropout)	(None, 2, 50)	0
dense_302 (Dense)	(None, 2, 50)	2,550
dropout_100 (Dropout)	(None, 2, 50)	0
lstm_220 (LSTM)	(None, 2, 50)	20,200
dense_303 (Dense)	(None, 2, 10)	510
dropout_101 (Dropout)	(None, 2, 10)	0
lstm_221 (LSTM)	(None, 50)	12,200
dropout_102 (Dropout)	(None, 50)	0
dense_304 (Dense)	(None, 50)	2,550
dense_305 (Dense)	(None, 50)	2,550
dense_306 (Dense)	(None, 1)	51

Figure 5: Summary of the LSTM model used for two-day forecast, totaling 51011 parameters.

Note that we choose to keep the model framework similar to that of the one-day scenario, i.e., same number of LSTM and dense layers, dropout rate, learning rate, and the total number of model parameters. The fitting process also goes through the same 300 epochs as the one-day scenario. We change the look back steps to be 2 instead of 1 from before. The resulting model performance also looks similar. However, there are subtle differences between the two cases. For example, the learning improvement takes a different trajectory as can be seen by the larger bulge in the upper right panel of figure 5, which reflects that the two-day learning model improves at a slightly later epoch. The model also does not saturate the learning process until ~ 80 epochs as opposed to ~ 50. Overall, the error measures demonstrate the similarity between the two models as can be seen by comparing the relative error distribution in the lower right panels of figures 6 and 4.

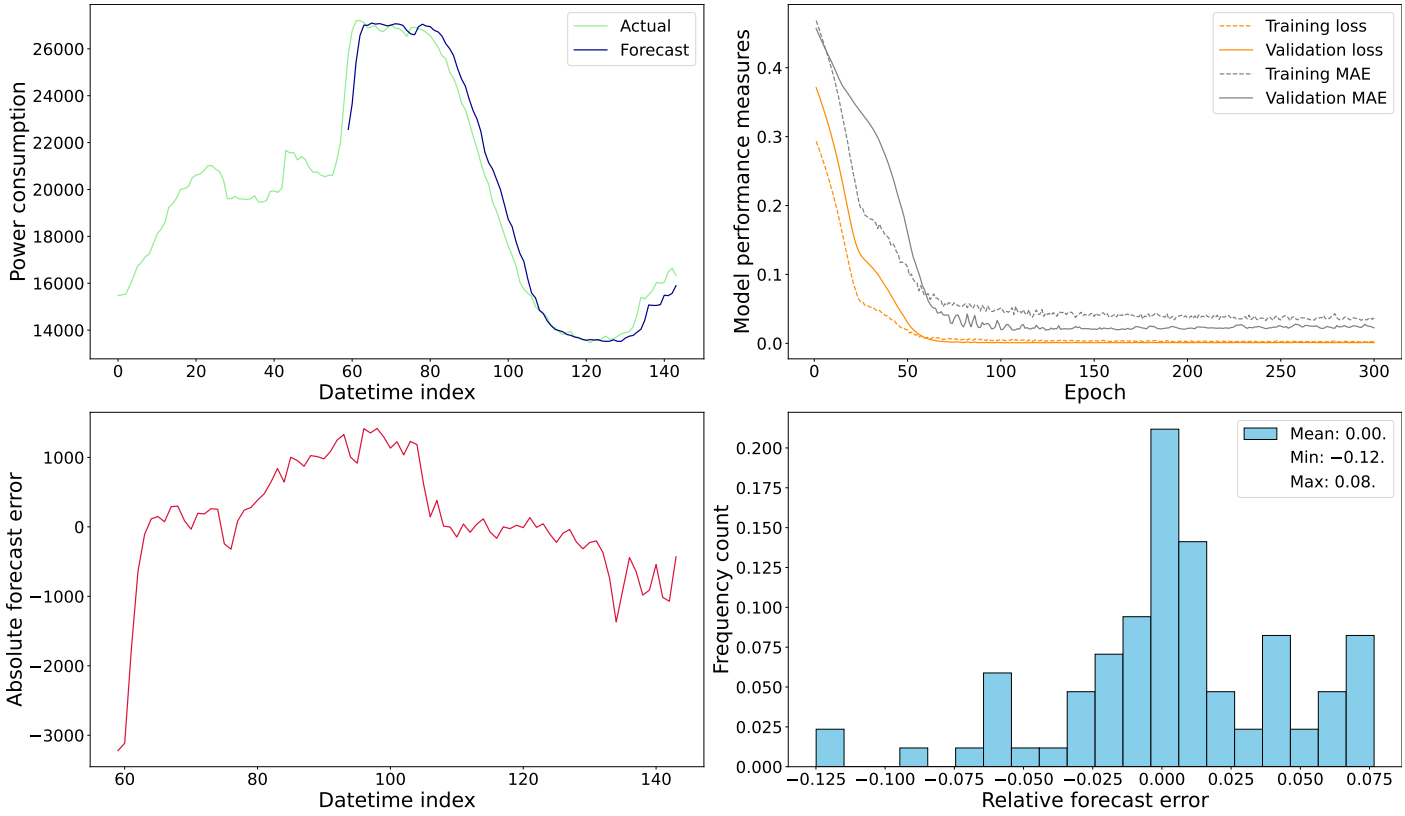

Figure 6: Summary of model output for the time period of two days. Upper left: actual (green) and predicted (blue) time series. Upper right: loss and MAE over the training epochs. Lower left: absolute forecast error over time. Lower right: frequency distribution of relative forecast error.

3.3 Model 3: Training and forecast for one-week time series

Next we investigate the medium coverage, over one week, with the model summarized in figure 7. Note that compared to the short-term models, the one-week model has reduced number of layers and no dropout during the training process. But it also has more computing units per LSTM layer, resulting in an overall increased amount of parameters, to 79911.

Layer (type)	Output Shape	Param #
lstm_311 (LSTM)	(None, 2, 60)	14,880
dense_420 (Dense)	(None, 2, 50)	3,050
lstm_312 (LSTM)	(None, 2, 60)	26,640
dense_421 (Dense)	(None, 2, 50)	3,050
lstm_313 (LSTM)	(None, 60)	26,640
dense_422 (Dense)	(None, 50)	3,050
dense_423 (Dense)	(None, 50)	2,550
dense_424 (Dense)	(None, 1)	51

Figure 7: Summary of the LSTM model used for one-week forecast, totaling 79911 parameters.

Removing dropouts is motivated by observing the lack of overfitting in the previous scenarios. The resulting learning progress, as shown in the upper right panel of figure 8, confirms that dropouts are indeed not necessary. Interestingly, the learning “bulge” still persists at early epochs but this feature in the one-week scenario

also shows a slight overfitting, which quickly goes away as the model performance improves continuously until it saturates before ~ 50 epochs. The overall forecast matches the cyclic pattern of the power consumption time series but also predicts a slightly downward trend across multiple cycles, shown in upper left panel of figure 8. The error distribution still resembles a Gaussian with zero mean but seems to be more clustered toward the mean than before, and has a slightly larger variance between -0.14 and 0.11. In other words, the forecast is now more closely matched with actual values but when it does deviate, the error fluctuates more significantly. This is also visible in lower left panel of figure 8.

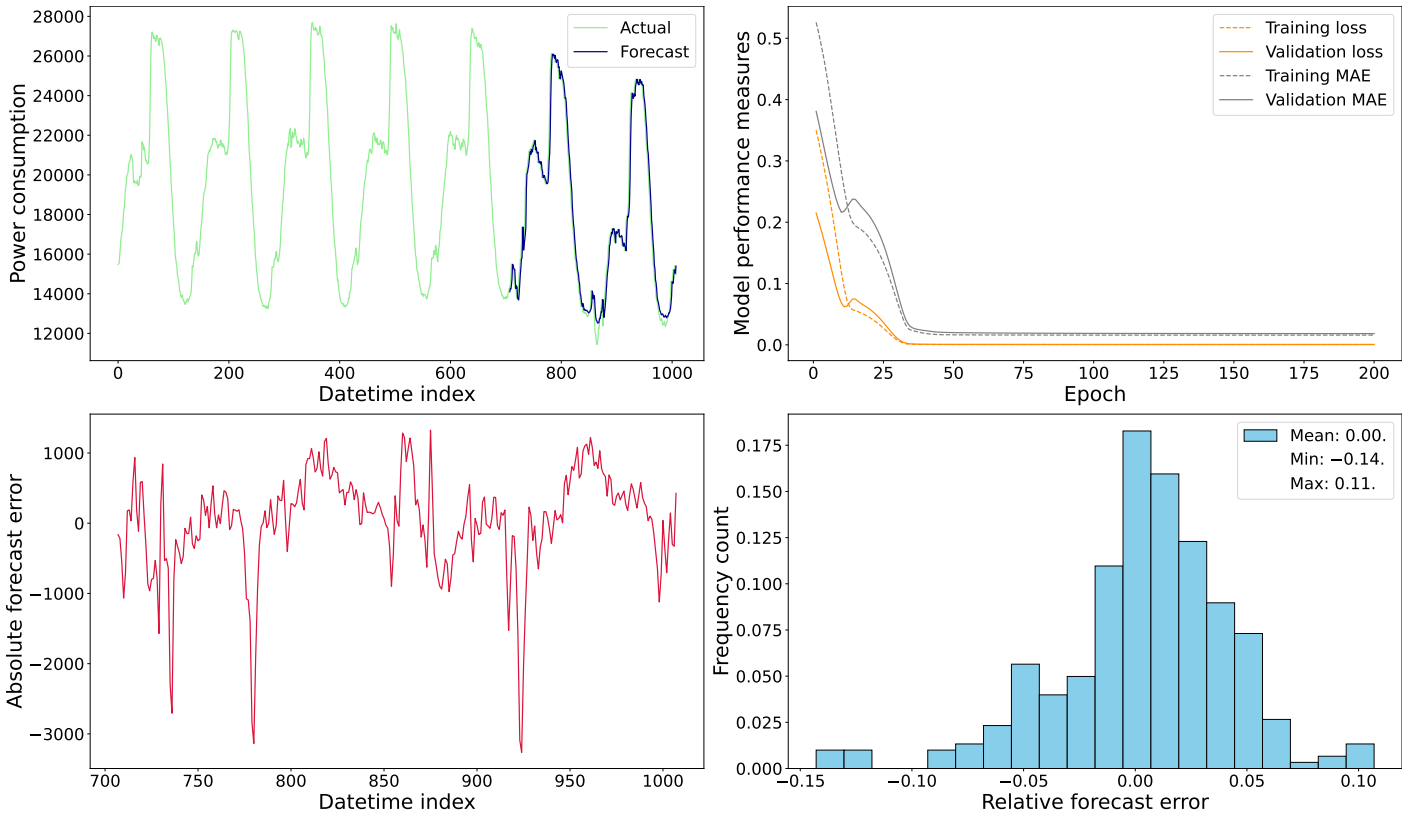

Figure 8: Summary of model output for the time period of one week. Upper left: actual (green) and predicted (blue) time series. Upper right: loss and MAE over the training epochs. Lower left: absolute forecast error over time. Lower right: frequency distribution of relative forecast error.

3.4 Model 4: Training and forecast for one-month time series

Prompted by the saturation of model performance no later than ~ 100 epochs in the scenarios investigated so far, we reduce the number of epochs to 50 for the one-month case study. We increase the look back steps to retain longer short-term memories and reintroduce dropouts to the model after trial and error. The final model parameters are shown in figure 9, with an even larger amount of trainable parameters than before.

Layer (type)	Output Shape	Param #
lstm_352 (LSTM)	(None, 30, 60)	14,880
dense_481 (Dense)	(None, 30, 50)	3,050
lstm_353 (LSTM)	(None, 30, 60)	26,640
dropout_211 (Dropout)	(None, 30, 60)	0
dense_482 (Dense)	(None, 30, 50)	3,050
lstm_354 (LSTM)	(None, 30, 60)	26,640
dense_483 (Dense)	(None, 30, 50)	3,050
lstm_355 (LSTM)	(None, 60)	26,640
dense_484 (Dense)	(None, 50)	3,050
dropout_212 (Dropout)	(None, 50)	0
dense_485 (Dense)	(None, 50)	2,550
dense_486 (Dense)	(None, 1)	51

Figure 9: Summary of the LSTM model used for one-month forecast, totaling 109601 parameters.

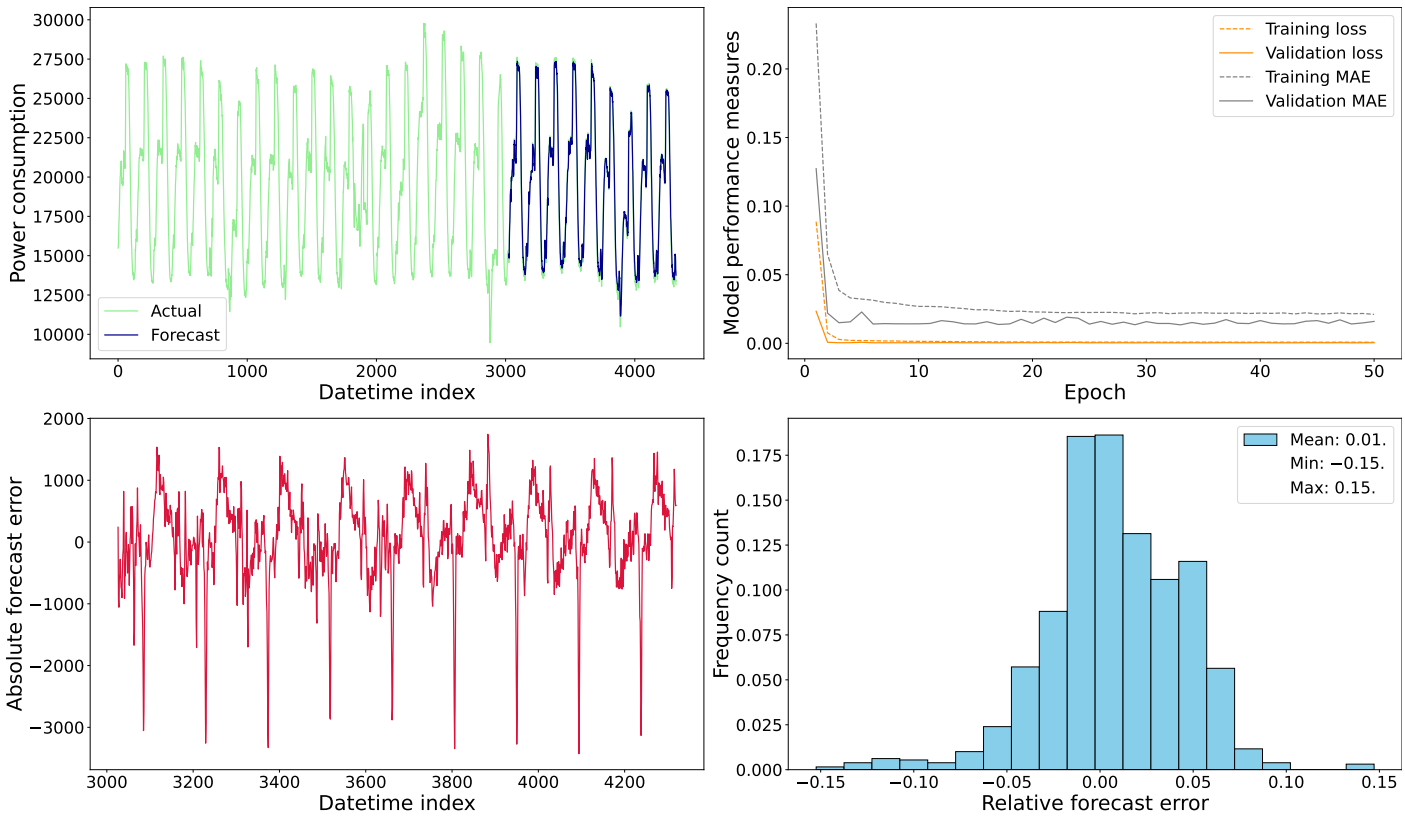

Figure 10: Summary of model output for the time period of one month. Upper left: actual (green) and predicted (blue) time series. Upper right: loss and MAE over the training epochs. Lower left: absolute forecast error over time. Lower right: frequency distribution of relative forecast error.

The resulting model output and performance are presented in figure 10. The forecast once again produces the essential characteristics of the actual time series, including fluctuation features beyond the daily cycle. The model performance saturates at around ~ 10 epochs, much quicker than the previous cases. This is

perhaps due to the fact that incorporating many daily cyclic patterns in the training and allowing the model to retain longer memories (longer look back steps) prompt the learning to be more skewed toward capturing longer-term data trend instead of the more rapid features on shorter time scales. The overall error distribution takes the observation we noted from the one-week model one step further, as the distribution now is even more centered around the zero mean and has a larger variance, between ± 0.15.

3.5 Model 5: Training and forecast for one-year time series

We tune the final model for the one-year scenario by inheriting some of the model features from before, such as setting the epochs to be 50, and reducing the look back step to capture short-term fluctuations in a long-term time series. Overall the model, shown in figure 11, resembles that of the one-month model in figure 9, with the same number of layers and total parameters.

Layer (type)	Output Shape	Param #
lstm_367 (LSTM)	(None, 1, 60)	14,880
dense_503 (Dense)	(None, 1, 50)	3,050
lstm_368 (LSTM)	(None, 1, 60)	26,640
dropout_217 (Dropout)	(None, 1, 60)	0
dense_504 (Dense)	(None, 1, 50)	3,050
lstm_369 (LSTM)	(None, 1, 60)	26,640
dense_505 (Dense)	(None, 1, 50)	3,050
lstm_370 (LSTM)	(None, 60)	26,640
dense_506 (Dense)	(None, 50)	3,050
dropout_218 (Dropout)	(None, 50)	0
dense_507 (Dense)	(None, 50)	2,550
dense_508 (Dense)	(None, 1)	51

Figure 11: Summary of the LSTM model used for one-year forecast, totaling 109601 parameters.

Figure 12 summarizes the results from the one-year model. The long-term fluctuation features are well captured in the upper left panel. The learning performance saturates very quickly before ~ 10 epochs, which is similar to what we found in the one-month model. Short-term errors are present throughout the test set and their distribution becomes a much more skewed zero-mean Gaussian (close to 50% is at/around zero) but with very long tails between -0.28 and 0.40. Going from the one-day coverage to the full one-year span, we have progressively discovered features in the LSTM models that we will now highlight in section 3.6.

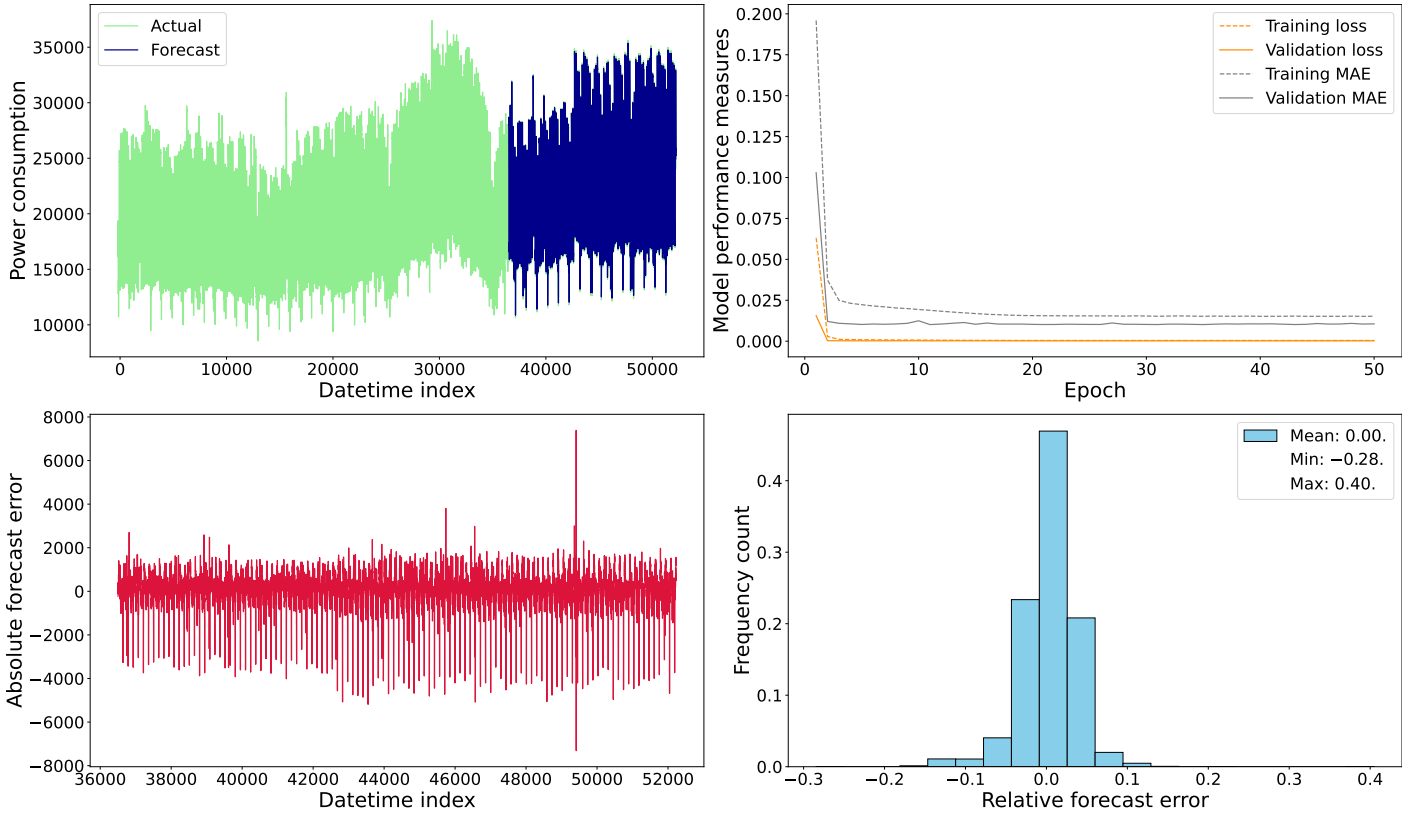

Figure 12: Summary of model output for the time period of one year. Upper left: actual (green) and predicted (blue) time series. Upper right: loss and MAE over the training epochs. Lower left: absolute forecast error over time. Lower right: frequency distribution of relative forecast error.

3.6 Choosing a RNN model for forecasting tasks

We compare the forecasting results from the scenarios presented in sections 3.1–3.5, and make the following observations regarding the choice of machine learning models for forecasting tasks.

1. LSTM models are a powerful tool for forecasting but the specific network structure and model parameters require careful tuning. Among other factors, the length of the desired forecast coverage significantly impacts the kind of LSTM model best fit for the objective. Therefore no specific LSTM model setup could be ranked as the top for forecasting energy consumption time series in general.

2. Although the optimal model parameters can only be determined upon fixing the length of the forecast length (and potentially other goals), LSTM overall produces predictions that match the actual time series values, as is evident from overlap between the two shown in the upper left panels of figures 4, 6, 7, 9, and 11. Therefore, we would conclude that LSTM framework is among the top options for a time-series forecast.

4 Insights and key findings

Over the course of the project, we summarize the following key insights and findings:

1. All LSTM models in this project produced predictions with somewhat reasonable errors. The errors all distribute around zero, follow an approximately normal distribution, and have larger variance for larger time spans.

2. The learning progress for short-term forecast, up to one week, share a similar feature of a “bulge” at early epochs, which subsequently gives away to continuous improvement until saturation at around ~ 50 to 100 epochs. The trajectory for long-term forecast, i.e., one month and one year, looks different as the performance saturates very early on at around ~ 10 epochs.

3. It is not straightforward to figure out the best network structure. For example in the one-week scenario, the model is the simplest, i.e., smallest number of layers, no dropouts, etc. It is not obvious why that configuration produced same/similar performance as more complex structures, or why it was the case for the one-week model. It is entirely possible that if all the parameters were adjusted following a careful grid search, our choice of model setups for the one-week (or any other) scenario would not be the most ideal.

4. Although LSTM has significant potential for forecasting, fully tapping into its capabilities requires finding the best network architecture and the optimal model parameters. This requires experience and intuition.

5 Next steps

A number of aspects could be improved in the future, in term of both the technical model development and the data aspects.

1. **GridSearch.** The model parameters are tuned by hand in this project to output approximately correct forecast. To better utilize the capacity of LSTM models, finding a set of best model parameters through grid search or other techniques would be needed. The parameter space to be optimized includes (i) number of units per LSTM layer, (ii) dropout rates, (iii) learning rate, (iv) the choice of optimizer, (v) look back steps, and so on.

2. **Gated recurrent unit (GRU)**. GRU models could be an alternative approach to LSTM for sequential data. It has simpler structure and might work as well as LSTM on small datasets, such as our aimed forecast covering short-term time scales such as days up to a week. So a next step task could be to try building a number of GRU models and compare their output with LSTM.

3. **Feature construction**. For a time-series analysis, the dataset with 10-min intervals is probably quite sufficient in terms of the cadence. But besides working directly with the raw feature of energy consumption as a function of time, perhaps some physical factors from the energy grid and/or the environment, or statistical measure within the dataset itself could be constructed and help guiding the development of the forecast model. Some examples of statistical features obtainable from the time series could be (i) seasonality, (ii) lag-based features, (iii) peak and trough Analysis, (iv) anomalies and outliers over longer time periods. Incorporating these features from the start might help building a more finessed model.