Conclusions
Results & Discussion
Data & Model
Introduction
Thank You!

It's Getting Hot In Here

Alicja Kałucka
Amit Singh
Bence Takács
Daniel Dalsgaard
Fabiano Tracanna

Goal

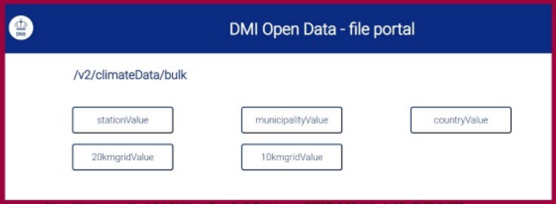
Trying to come up with a geomodel of the weather stations and trying to predict data for the future

Motivation

- It would be nice to compare it with a data that is already there and check how good the model is
- Maybe we can predict the data for the future and see some threats for the future climate

Data

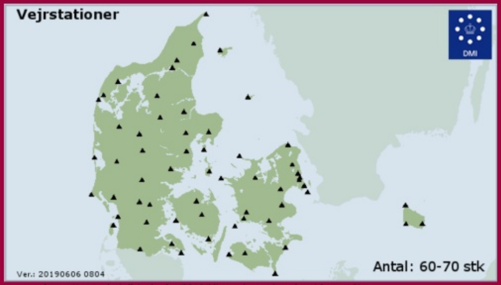
DMI - stationvalue


DMI Open Data - file portal

/v2/climateData/bulk

https://dmigw.govcloud.dk/v2/climateData/bulk/?api-key=ee8335f9-fa08-4bbe-ba15-a0bf5b0e89fb

55 different stations all over Denmark



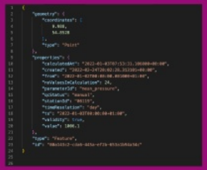
https://www.dmi.dk/nyheder/2019/klimatologen-har-det-sidste-ord/

Data preprocessing

Bornholm

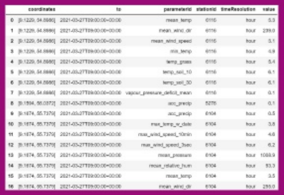
Methods

- The raw data we got



```
{
    "geometry": {
        "coordinates": [
            9.908,
            54.8928
        ],
        "type": "Point"
    },
    "properties": {
        "calculatedAt": "2022-01-03T07:53:31.306000+00:00",
        "created": "2022-03-24T20:02:28.352301+00:00",
        "from": "2022-01-02T00:00:00.000000+01:00",
        "noValuesInCalculation": 24,
        "parameterId": "mean_pressure",
        "qcStatus": "manual",
        "stationId": "06119",
        "timeResolution": "day",
        "to": "2022-01-03T00:00:00+01:00",
        "validity": true,
        "value": 1006.3
    },
    "type": "Feature",
    "id": "[illegible]"
}
```

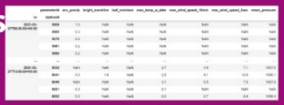
- CSV file with 90 million entries



	coordinates	to	parameterId	stationId	timeResolution	value
0	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_temp	6116	hour	5.3
1	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_dir	6116	hour	239.0
2	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_speed	6116	hour	3.1
3	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	min_temp	6116	hour	4.9
4	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_grass	6116	hour	5.4
5	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_10	6116	hour	6.1
6	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_30	6116	hour	6.1
7	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	vapour_pressure_deficit_mean	6116	hour	0.1
8	[9.1594, 56.0372]	2021-03-27T09:00:00+00:00	acc_precip	5276	hour	0.1
9	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	acc_precip	6104	hour	0.5
10	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_temp_w_date	6104	hour	3.8
11	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_10min	6104	hour	4.6
12	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_3sec	6104	hour	6.2
13	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_pressure	6104	hour	1008.9
14	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_relative_hum	6104	hour	83.3
15	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_temp	6104	hour	3.5
16	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_wind_dir	6104	hour	255.0

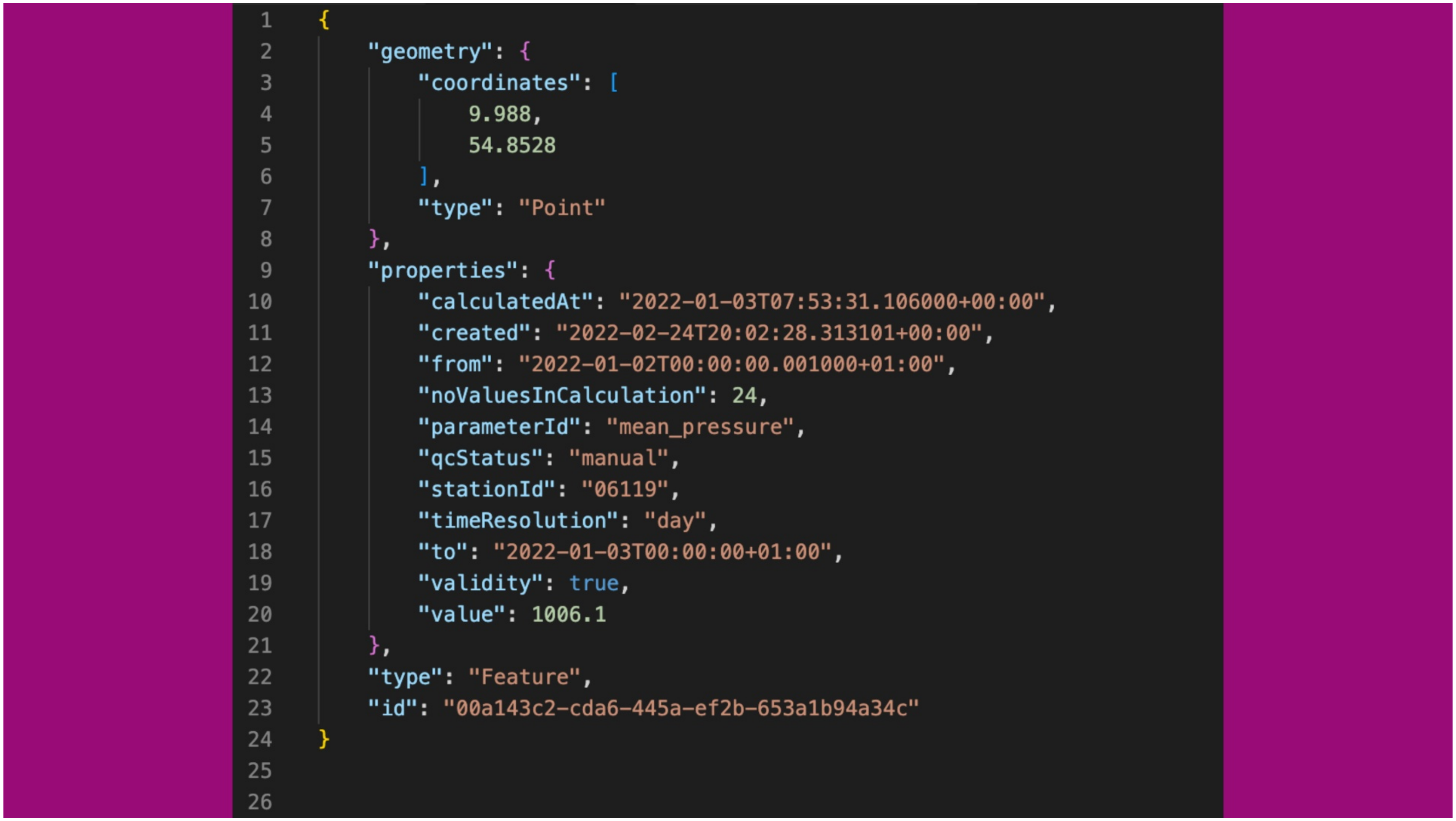
- We pivoted it

- Use only Synop stations

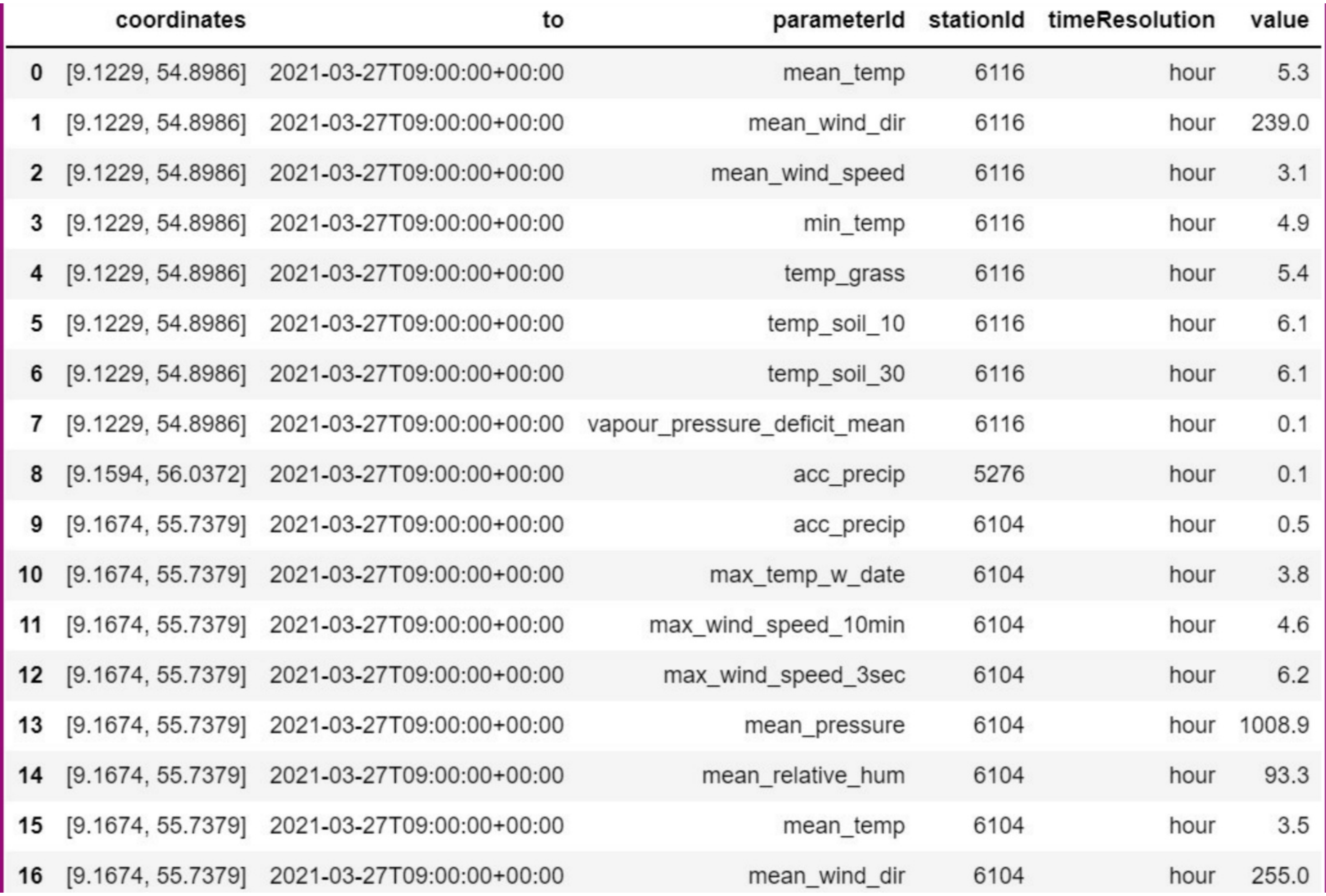


	parameterId	acc_precip	bright_sunshine	leaf_moisture	max_temp_w_date	max_wind_speed_10min	max_wind_speed_3sec	mean_pressure
to	stationId							
2021-03-27T09:00:00+00:00	5009	1.3	NaN	NaN	NaN	NaN	NaN	NaN
	5005	0.4	NaN	NaN	NaN	NaN	NaN	NaN
	5070	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	5081	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	5089	0.2	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...
2021-03-27T10:00:00+00:00	6032	NaN	NaN	NaN	2.7	4.9	7.1	1007.0
	6041	0.3	1.6	NaN	2.8	9.1	12.8	1006.1
	6049	NaN	NaN	NaN	3.1	5.5	7.5	1007.9
	6051	0.3	NaN	NaN	0.1	NaN	NaN	NaN
	6052	0.3	NaN	NaN	0.3	0.7	6.6	1008.4

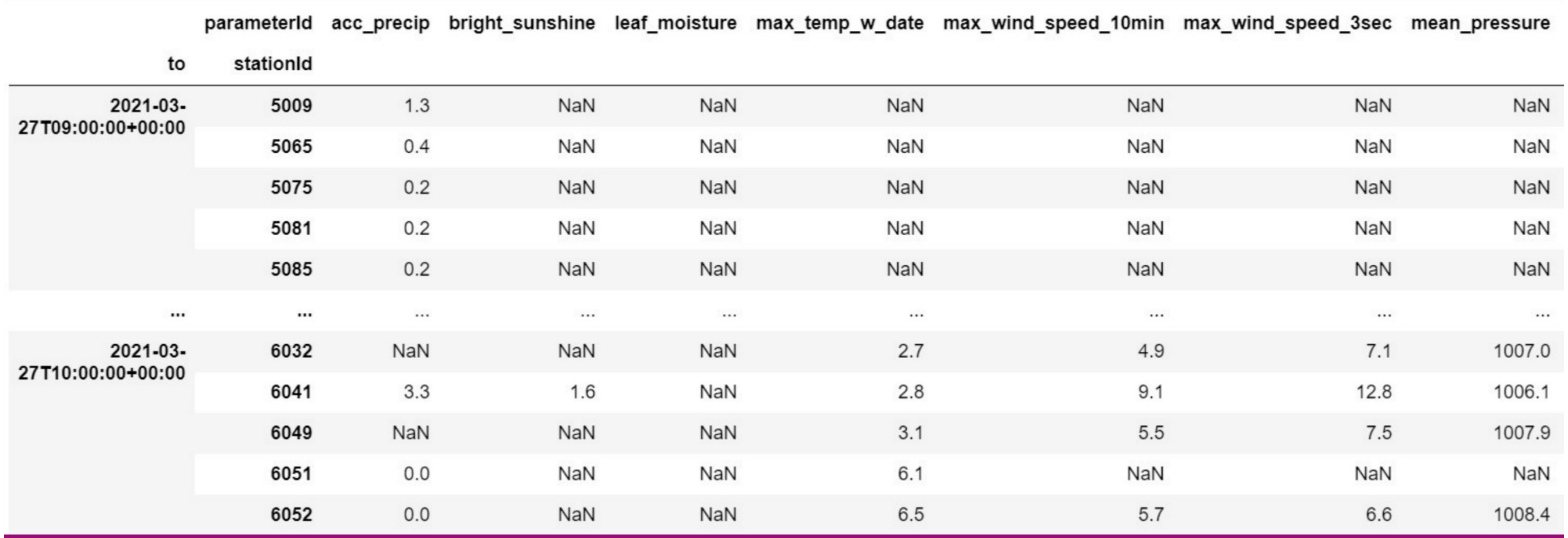
- Remove ALL NaN's
  4.5 million entries



```json
{
    "geometry": {
        "coordinates": [
            9.988,
            54.8528
        ],
        "type": "Point"
    },
    "properties": {
        "calculatedAt": "2022-01-03T07:53:31.106000+00:00",
        "created": "2022-02-24T20:02:28.313101+00:00",
        "from": "2022-01-02T00:00:00.001000+01:00",
        "noValuesInCalculation": 24,
        "parameterId": "mean_pressure",
        "qcStatus": "manual",
        "stationId": "06119",
        "timeResolution": "day",
        "to": "2022-01-03T00:00:00+01:00",
        "validity": true,
        "value": 1006.1
    },
    "type": "Feature",
    "id": "00a143c2-cda6-445a-ef2b-653a1b94a34c"
}
```

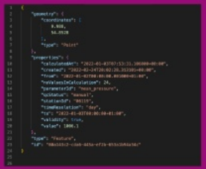


	coordinates	to	parameterId	stationId	timeResolution	value
0	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_temp	6116	hour	5.3
1	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_dir	6116	hour	239.0
2	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_speed	6116	hour	3.1
3	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	min_temp	6116	hour	4.9
4	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_grass	6116	hour	5.4
5	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_10	6116	hour	6.1
6	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_30	6116	hour	6.1
7	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	vapour_pressure_deficit_mean	6116	hour	0.1
8	[9.1594, 56.0372]	2021-03-27T09:00:00+00:00	acc_precip	5276	hour	0.1
9	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	acc_precip	6104	hour	0.5
10	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_temp_w_date	6104	hour	3.8
11	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_10min	6104	hour	4.6
12	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_3sec	6104	hour	6.2
13	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_pressure	6104	hour	1008.9
14	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_relative_hum	6104	hour	93.3
15	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_temp	6104	hour	3.5
16	[9.1674, 55.7379]	2021-03-27T09:00:00+00:00	mean_wind_dir	6104	hour	255.0



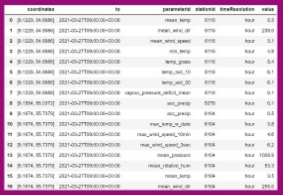
	parameterId	acc_precip	bright_sunshine	leaf_moisture	max_temp_w_date	max_wind_speed_10min	max_wind_speed_3sec	mean_pressure
to	**stationId**							
2021-03-27T09:00:00+00:00	**5009**	1.3	NaN	NaN	NaN	NaN	NaN	NaN
	5065	0.4	NaN	NaN	NaN	NaN	NaN	NaN
	5075	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	5081	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	5085	0.2	NaN	NaN	NaN	NaN	NaN	NaN
...	**...**	...	...	...	...	...	...	...
2021-03-27T10:00:00+00:00	**6032**	NaN	NaN	NaN	2.7	4.9	7.1	1007.0
	6041	3.3	1.6	NaN	2.8	9.1	12.8	1006.1
	6049	NaN	NaN	NaN	3.1	5.5	7.5	1007.9
	6051	0.0	NaN	NaN	6.1	NaN	NaN	NaN
	6052	0.0	NaN	NaN	6.5	5.7	6.6	1008.4

- The raw data we got



```
{
    "geometry": {
        "coordinates": [
            [illegible],
            [illegible]
        ],
        "type": "Point"
    },
    "properties": {
        "calculatedAt": "2022-01-03T07:53:31.[illegible]+00:00",
        "created": "2022-02-24T20:02:28.[illegible]+00:00",
        "from": "2022-01-03T00:00:00.[illegible]+01:00",
        "noValuesInCalculation": 24,
        "parameterId": "mean_pressure",
        "qcStatus": "manual",
        "stationId": "[illegible]",
        "timeResolution": "day",
        "to": "2022-01-03T00:00:00+01:00",
        "validity": true,
        "value": 1008.1
    },
    "type": "Feature",
    "id": "[illegible]"
}
```

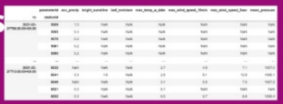
- CSV file with 90 million entries



	coordinates	to	parameterId	stationId	timeResolution	value
0	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_temp	6116	hour	5.3
1	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_dir	6116	hour	239.0
2	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	mean_wind_speed	6116	hour	3.1
3	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	min_temp	6116	hour	4.9
4	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_grass	6116	hour	5.4
5	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_10	6116	hour	6.1
6	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	temp_soil_30	6116	hour	6.1
7	[9.1229, 54.8986]	2021-03-27T09:00:00+00:00	vapour_pressure_deficit_mean	6116	hour	0.1
8	[9.1594, 56.0372]	2021-03-27T09:00:00+00:00	acc_precip	5276	hour	0.1
9	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	acc_precip	6104	hour	0.5
10	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	max_temp_w_date	6104	hour	3.8
11	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_10min	6104	hour	4.6
12	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	max_wind_speed_3sec	6104	hour	6.2
13	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	mean_pressure	6104	hour	1008.9
14	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	mean_relative_hum	6104	hour	93.3
15	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	mean_temp	6104	hour	3.5
16	[9.1874, 55.7379]	2021-03-27T09:00:00+00:00	mean_wind_dir	6104	hour	295.0

- We pivoted it

- Use only Synop stations



	parameterId	acc_precip	bright_sunshine	leaf_moisture	max_temp_w_date	max_wind_speed_10min	max_wind_speed_3sec	mean_pressure
to	stationId							
2021-03-27T08:00:00+00:00	[illegible]	1.3	NaN	NaN	NaN	NaN	NaN	NaN
	[illegible]	[illegible]	NaN	NaN	NaN	NaN	NaN	NaN
	[illegible]	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	[illegible]	0.2	NaN	NaN	NaN	NaN	NaN	NaN
	[illegible]	0.2	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...
2021-03-27T10:00:00+00:00	[illegible]	NaN	NaN	NaN	2.7	4.8	7.1	1007.0
	[illegible]	3.3	1.6	NaN	2.6	8.1	12.8	1006.1
	[illegible]	NaN	NaN	NaN	3.1	5.5	7.5	1007.3
	[illegible]	0.3	NaN	NaN	6.1	NaN	NaN	NaN
	[illegible]	0.3	NaN	NaN	6.0	3.7	6.6	1008.4

- Remove ALL NaN's
  4.5 million entries

Bornholm is well-known island for its beautiful landscapes, but it's also known for quite a temperate climate (long, warm-ish summers and wild winters)

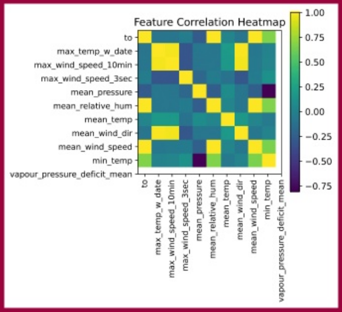
mer/bornholm-denmark/

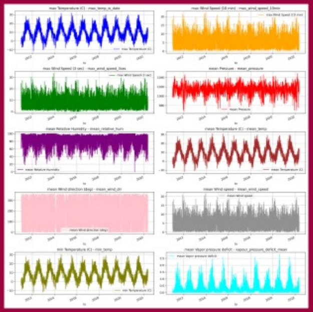
https://d3qvqlc701gzhm.clou

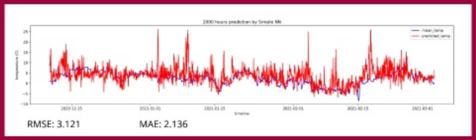
In order to see whether it's possible to predict next timestamps, we tried to use:

- We tried to use a simple NN just to check
- GNN
- GRU
- LSTM

Features description





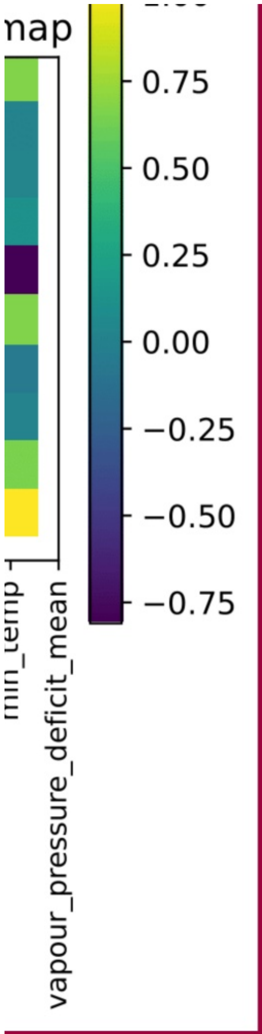


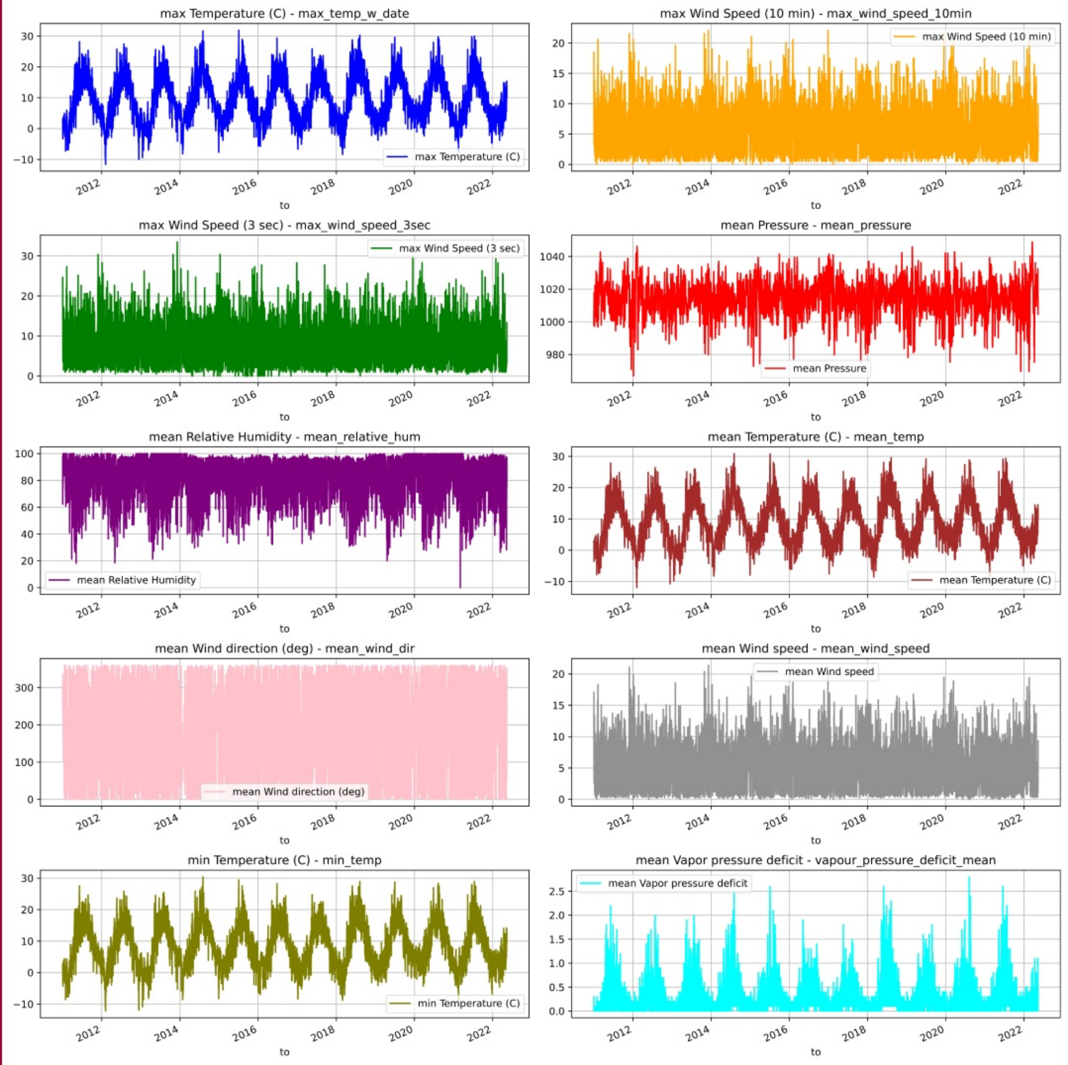
LSTM

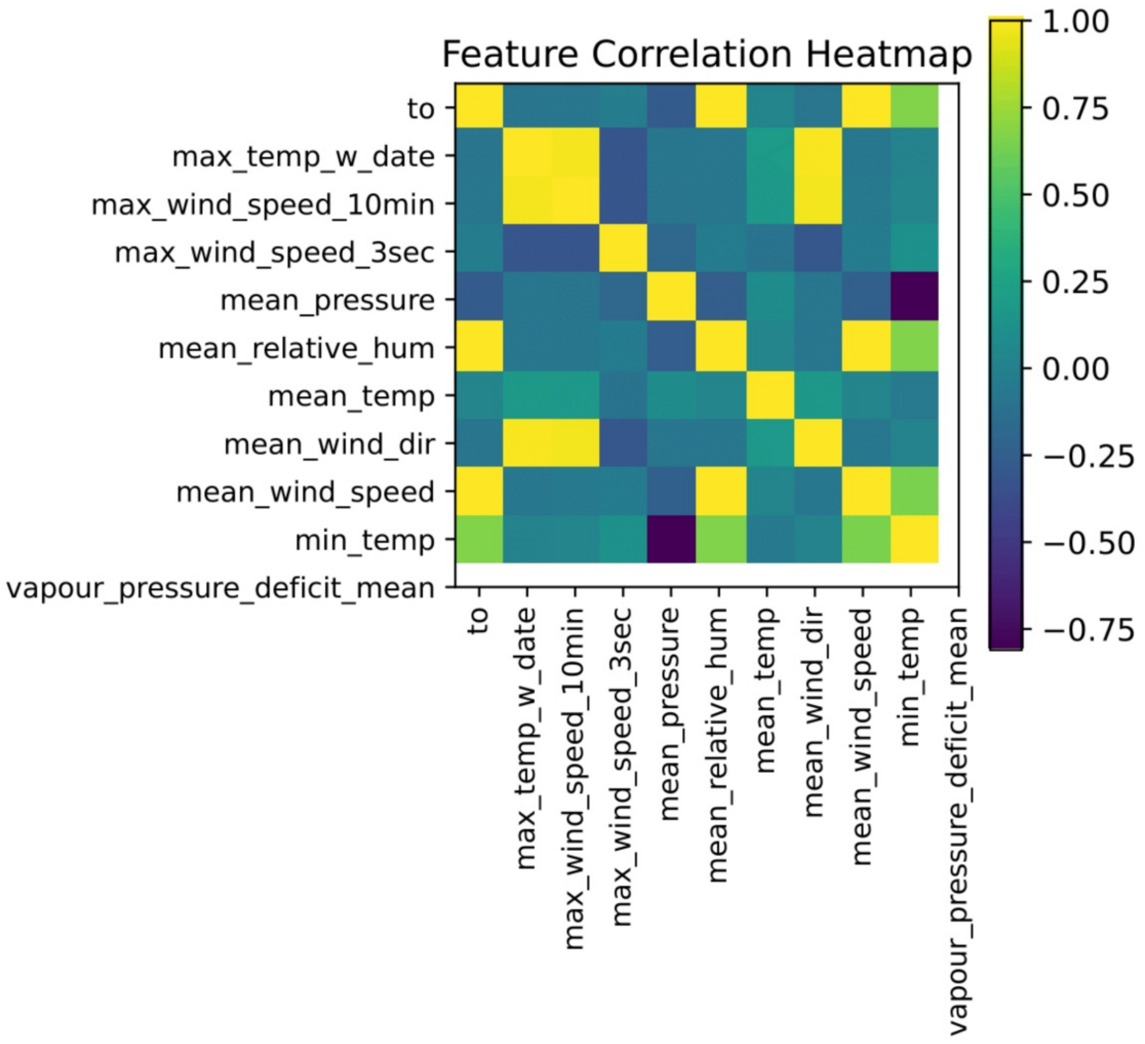
LSTM vol. 2

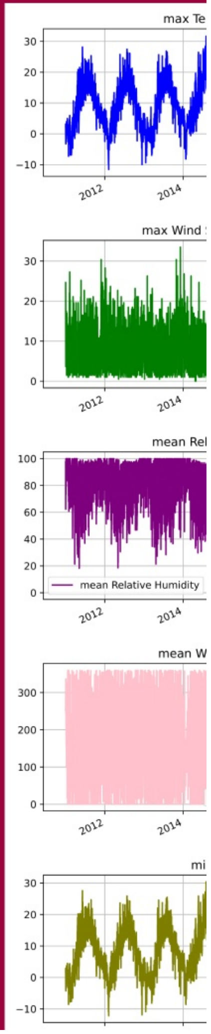
GRU

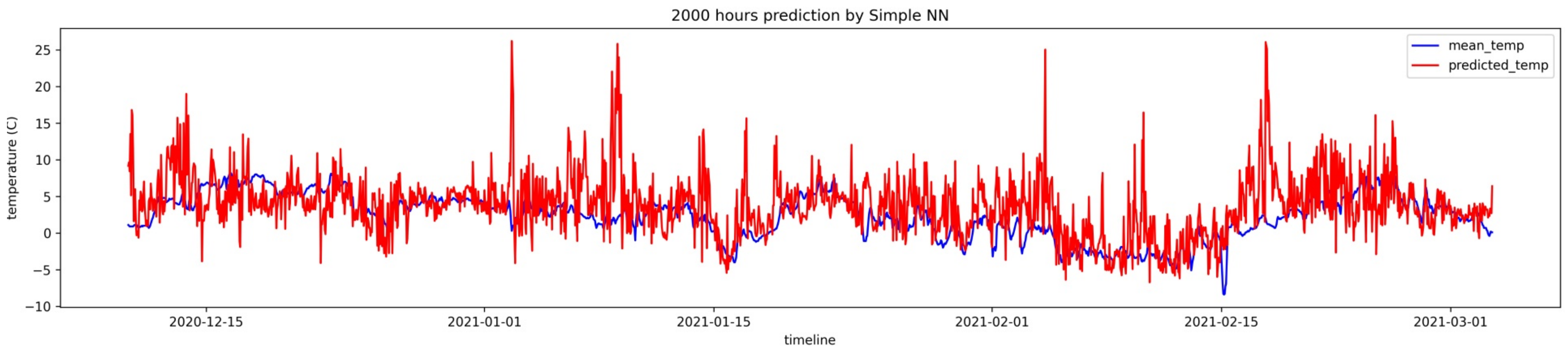
GNN


nap
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
min_temp
vapour_pressure_deficit_mean


max Temperature (C) - max_temp_w_date
max Wind Speed (10 min) - max_wind_speed_10min
max Wind Speed (3 sec) - max_wind_speed_3sec
mean Pressure - mean_pressure
mean Relative Humidity - mean_relative_hum
mean Temperature (C) - mean_temp
mean Wind direction (deg) - mean_wind_dir
mean Wind speed - mean_wind_speed
min Temperature (C) - min_temp
mean Vapor pressure deficit - vapour_pressure_deficit_mean
to

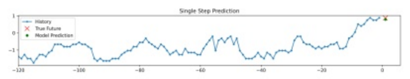

Feature Correlation Heatmap
to
max_temp_w_date
max_wind_speed_10min
max_wind_speed_3sec
mean_pressure
mean_relative_hum
mean_temp
mean_wind_dir
mean_wind_speed
min_temp
vapour_pressure_deficit_mean
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

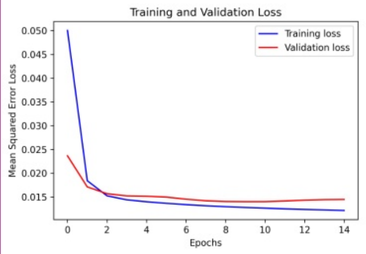

max Te
max Wind
mean Rel
mean Relative Humidity
mean W
mi
2012
2014



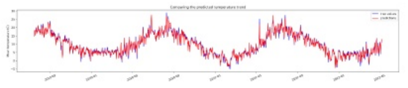
RMSE: 3.121 MAE: 2.136

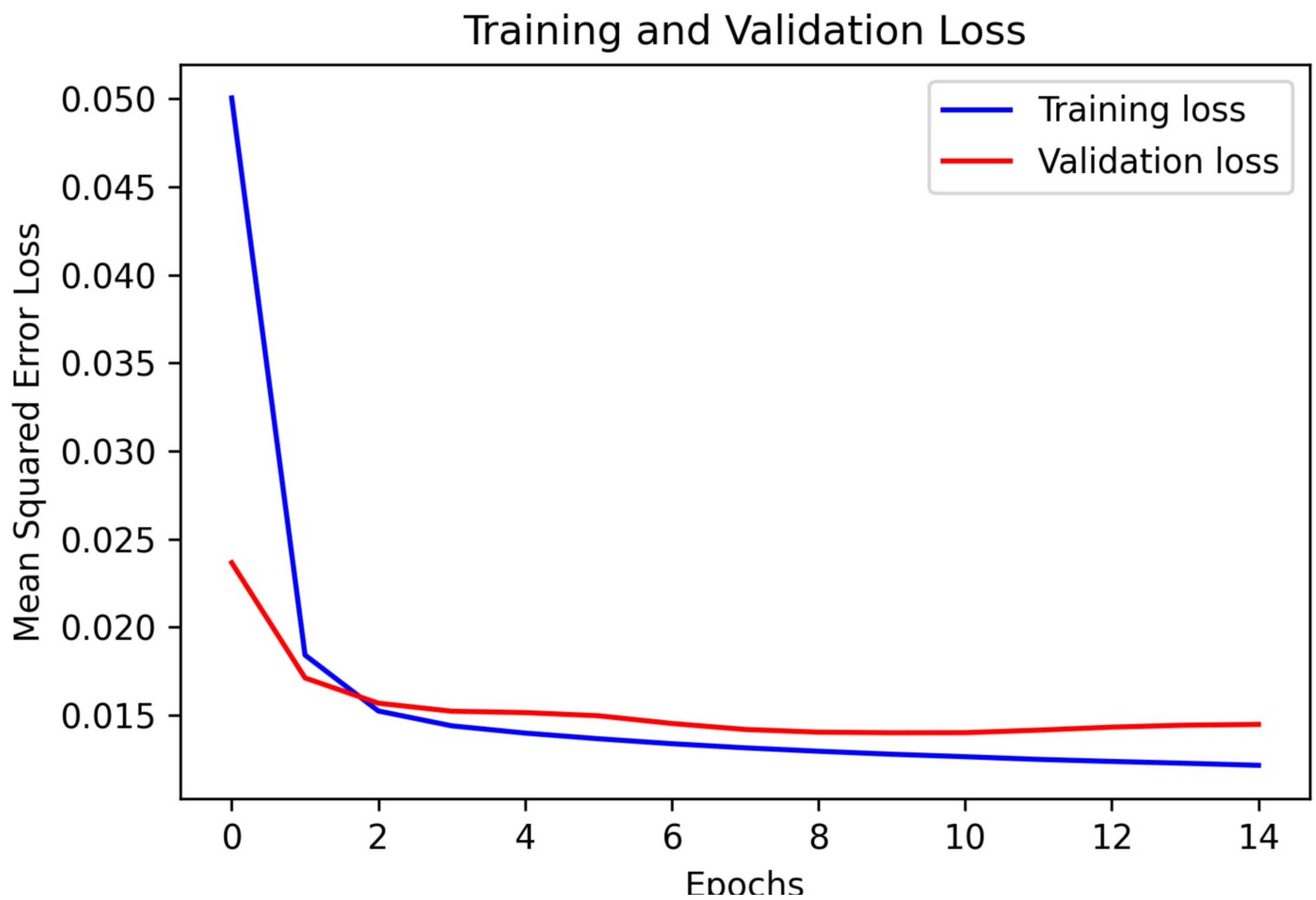
LSTM



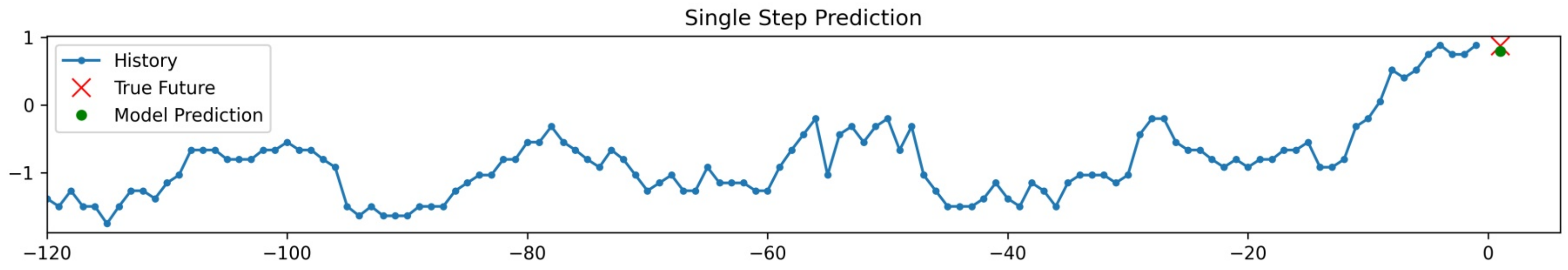


LSTM (1 LSTM layer with 32 units, activation="tanh", recurrent_activation="sigmoid")
7 features (without feature engg.) & 13 features (with feature engg.)
HP:
learn. rate: 0.001
batch size: 20
epochs: 30
sequence length: 120
Loss fun. : MSE
RMSE (test): 0.7935 deg C (with F engg.) & 1.0191 deg C (without F engg.)
MAE (test): 0.5756 deg C (with F engg.) & 0.7248 deg C (without F engg.)



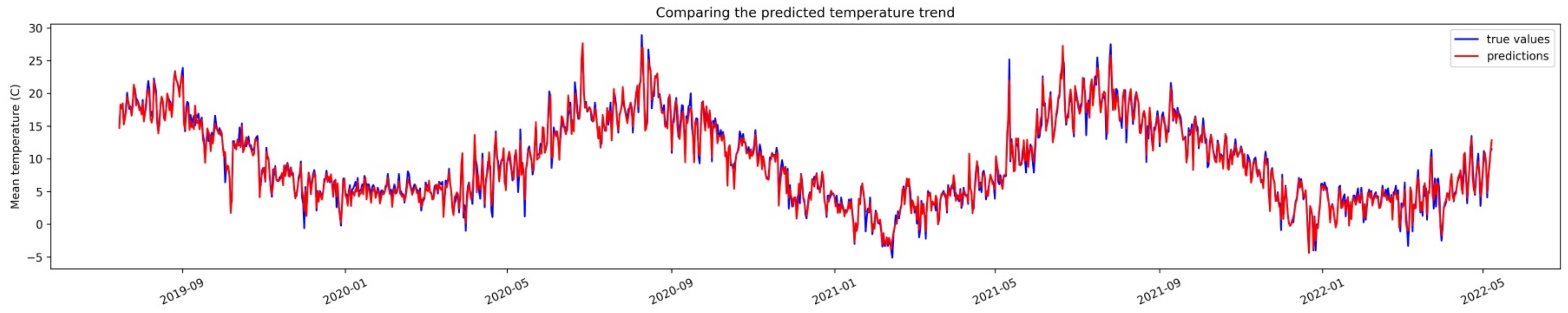


LSTM
recur
7 feat
HP:
learn
batch
epoch
seque
Loss
RMSE
MAE



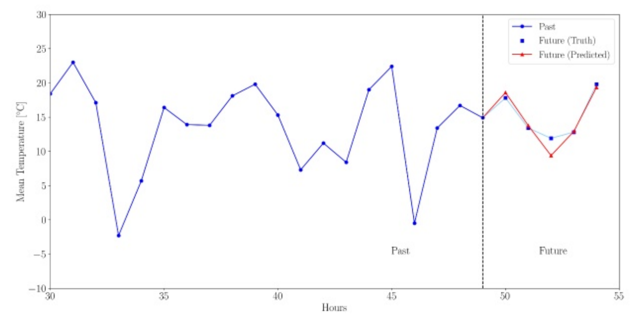
LSTM (1 LSTM layer with 32 units, activation="tanh", recurrent_activation="sigmoid")
7 features (without feature engg.) & 13 features (with feature engg.)

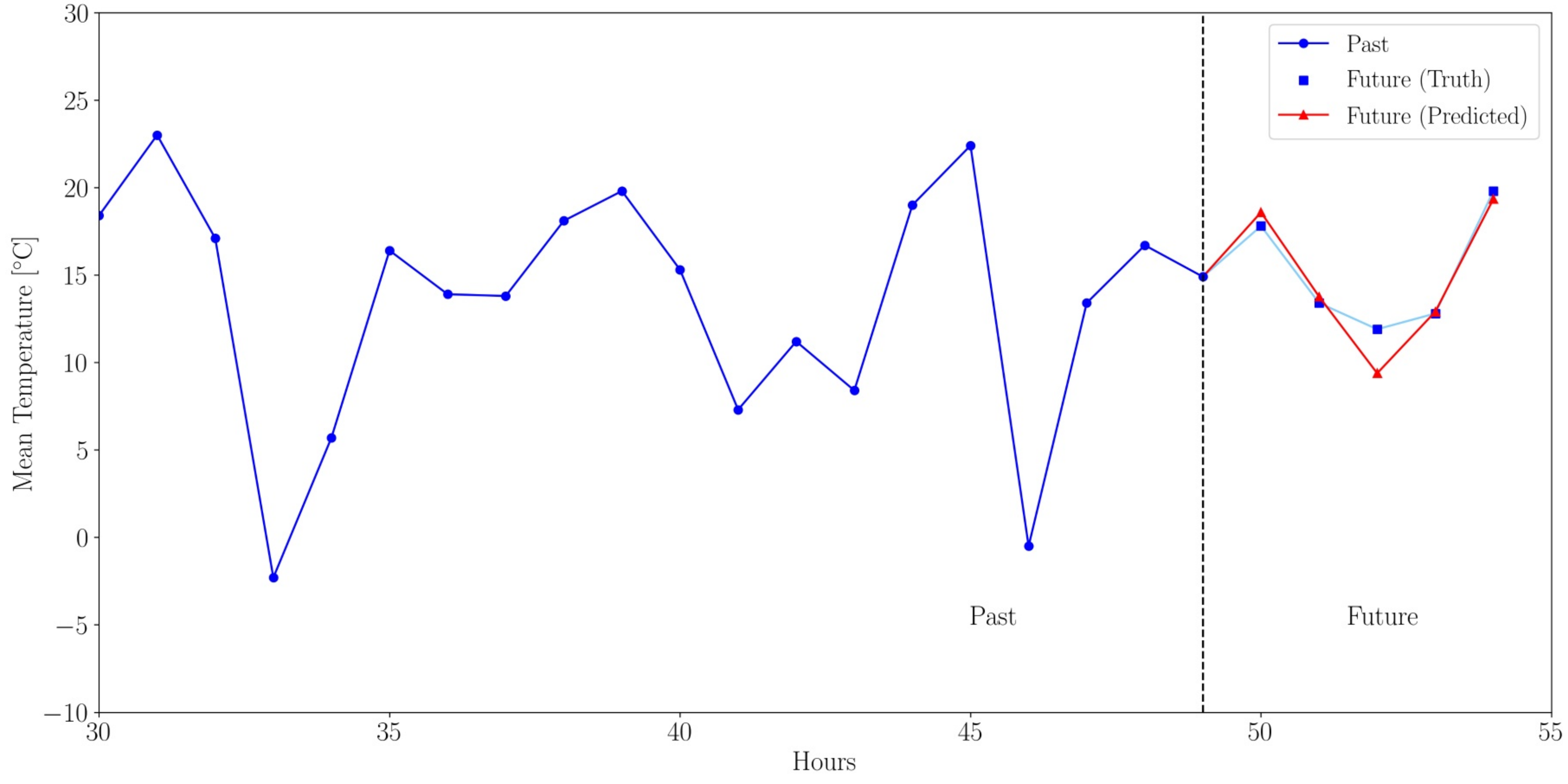
RMSE (test): 0.7935 deg C (with F engg.) & 1.0191 deg C (without F engg.)
MAE (test): 0.5756 deg C (with F engg.) & 0.7248 deg C (without F engg.)



LSTM vol. 2

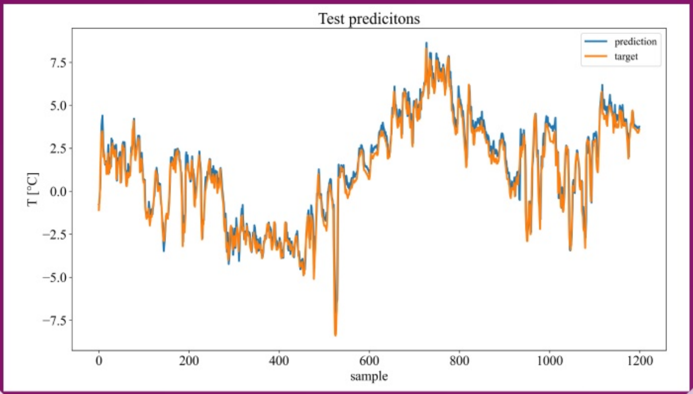
- Sequence length: 50
- Forecast horizon: 5
- RMSE = 0.09

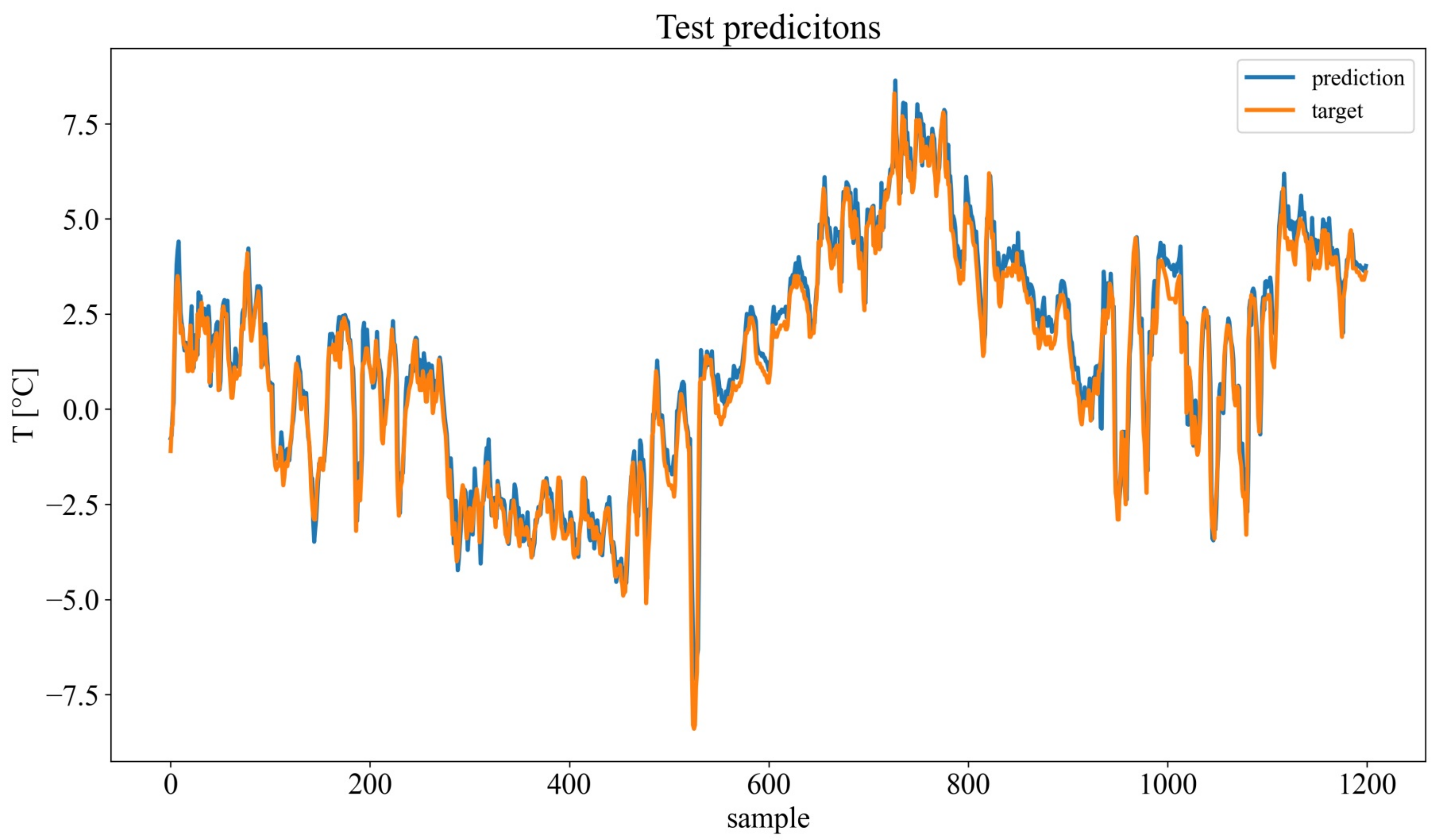



Past
Future (Truth)
Future (Predicted)
Past
Future
Mean Temperature [°C]
Hours
30
25
20
15
10
5
0
−5
−10
30
35
40
45
50
55

GRU

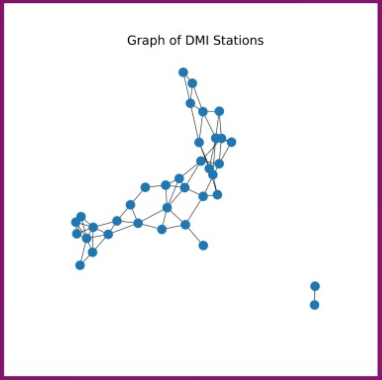
- Sequence length: 96 (4 days)
- Epochs: 15
- Learn rate: scheduler
- Dropout: 0.2
- RMSE = 0.62 °C




Test predicitons
prediction
target
T [°C]
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
0
200
400
600
800
1000
1200
sample

GNN

It runs, but
- Computationally expensive
- Takes 100's of hours of GPU compute



Conclusions

- We can predict temperature with an acceptable error for everyday people
- Extend predictions over a few days
- GNN looks promising but takes more work
- Combine more (all) stations to make a global model

It's Getting Hot In Here

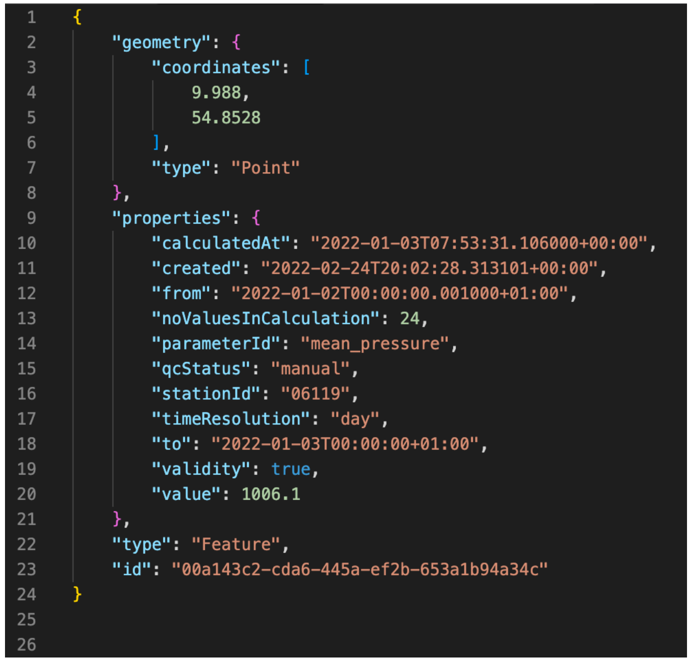
Alicja Kałucka
Amit Singh
Bence Takács
Daniel Dalsgaard
Fabiano Tracanna

Appendix

Alicja, Amit, Bence, Daniel, and Fabiano

Data Preprocessing

- Downloading raw data from DMI
  - Contains ALL weather stations and all their measurements for ~10 years
  - Stations in Greenland as well as Denmark
- JSON dictionary format for each measurement
  - One measurement is one feature per station per time
- RAM intensive to ingest and process



```json
{
    "geometry": {
        "coordinates": [
            9.988,
            54.8528
        ],
        "type": "Point"
    },
    "properties": {
        "calculatedAt": "2022-01-03T07:53:31.106000+00:00",
        "created": "2022-02-24T20:02:28.313101+00:00",
        "from": "2022-01-02T00:00:00.001000+01:00",
        "noValuesInCalculation": 24,
        "parameterId": "mean_pressure",
        "qcStatus": "manual",
        "stationId": "06119",
        "timeResolution": "day",
        "to": "2022-01-03T00:00:00+01:00",
        "validity": true,
        "value": 1006.1
    },
    "type": "Feature",
    "id": "00a143c2-cda6-445a-ef2b-653a1b94a34c"
}
```

Data Preprocessing - continued

- Steps:
  - Ingest data
  - Save relevant datapoints in pandas dataframe
    - This includes feautere name, feature value, coordinate, station id, time resolution
    - Reduces file size about 50% by not saving the rest of the JSON dictionary
    - Save dataframe as single csv
  - Split data by time resolution and keep working with hourly data
  - Pivot dataframe so columns are feature names and indices are timestep AND stationID
    - Many missing values.
    - Drop features with more than 10% missing values (not all stations measure everything)
    - Drop ALL NaN’s to have some data to work with

Data Preprocessing – continued 2

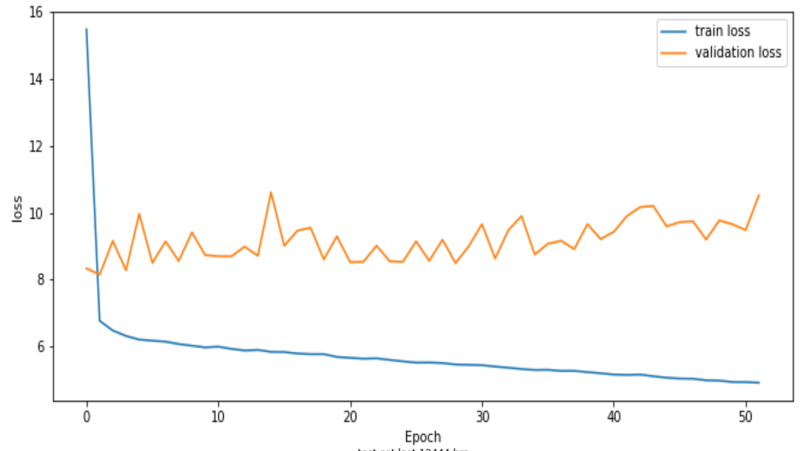
- Split dataset into stations only for Bornholm and pick just station 6190
  - Models don't support the previous way of storing all data, so we had to stick with just one station (see slide on GNN for more info on all stations)
- Save station 6190 in one file, and save the rest with NaN's removed in 10 files to reduce the RAM requirements to work with it.
- The process can be found in data_processing.ipynb files 1-5
  - Split in 5 amongst other reasons for RAM considerations
- In the end we have a total of 36 stations that have the data for all features we use for all timesteps.

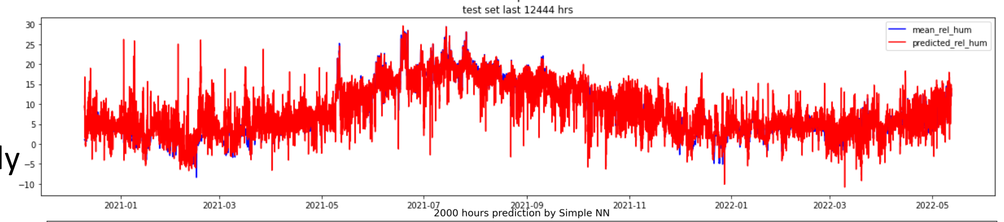
Alicja’s Deep Neural Network

- Note: This code had some algorithmic errors.
- Using TensorFlow Keras and Sk-learn
- Data scaled with Sklearn’s Standard scaler.
- Taking all the 16 input features from data file and predicting one output.
- Loss function was MAE.
- Model description (Sequential model):
  - 6 hidden layers (units : 100,200,200,200,100,20 respectively from layer 1 to 6)
  - 1 output layer
  - activation function – ReLu (for hidden layers)
- Hyper-parameters:
  - optimizer – Adam
  - epochs – 10
  - batch_size – 100

Amit’s Deep Neural Network

- Note: This regular NN ignores timeseries order and deals it as a simple regression model.
- Using TensorFlow Keras and Sk-learn
- Data scaled with Sklearn’s Standard scaler.
- Taking 7 input features from data file and predicting one output.
- Loss function was MSE.
- Model description (Sequential model):
  - 4 hidden layers (units : 256,256,128,128 respectively from layer 1 to 4)
  - activation function – ReLu (for hidden layers)
  - 1 output layer
- Hyper-parameters:
  - optimizer – Adam (with learning_rate = 0.001)
  - epochs – 500 (Early stopping callback with patience=50)
  - batch_size – 50





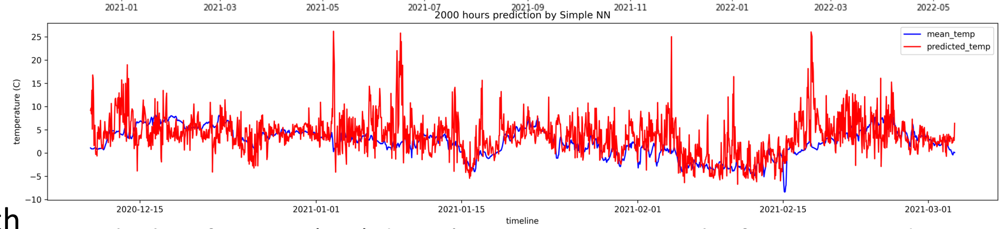


Fig: The loss function (top) doesn’t seem converging; the first 2000 prediction (bottom) looks very noisy and bad with RMSE- 3.121 °C ; MAE: 2.136 °C.

Amit’s LSTM

- Using TensorFlow Keras and Pandas
- Normalisation - Standard Scaling the whole data w.r.t training data.
- Features used :
  - 7 without feature engineering
  - 7 + 6 (self-made) with feature engineering (the additional 6 features were made to preserve the daily and yearly periodicity of the temperature data in our model)
  - Ignored the feature “wind direction” because its values were angles.

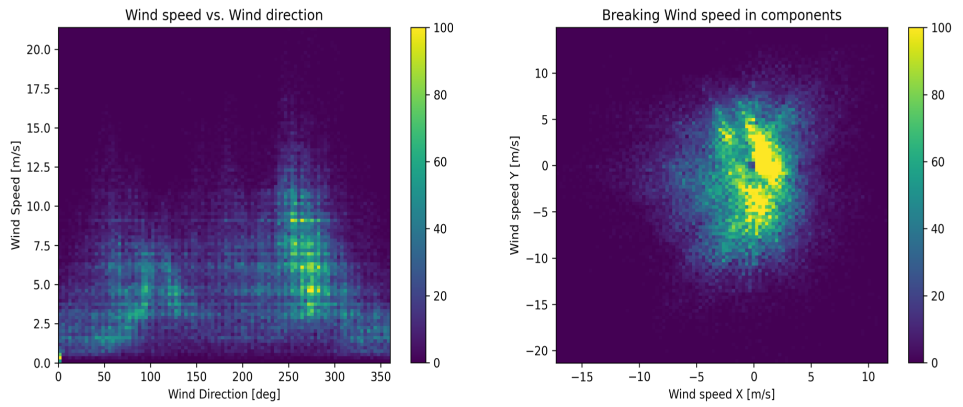
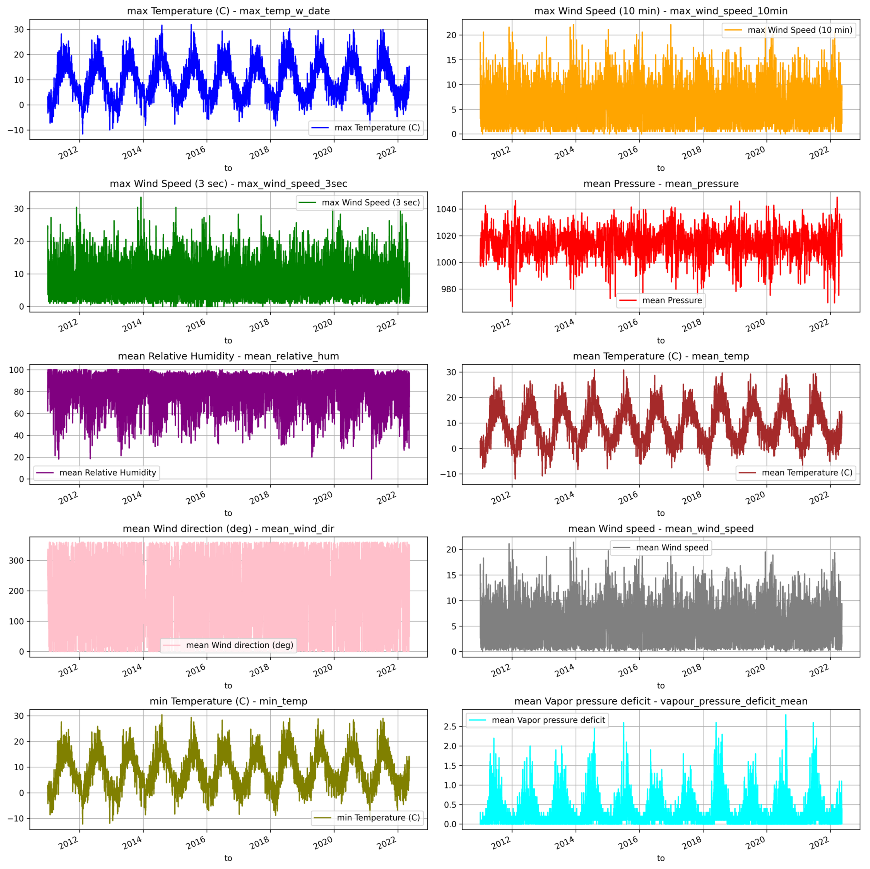


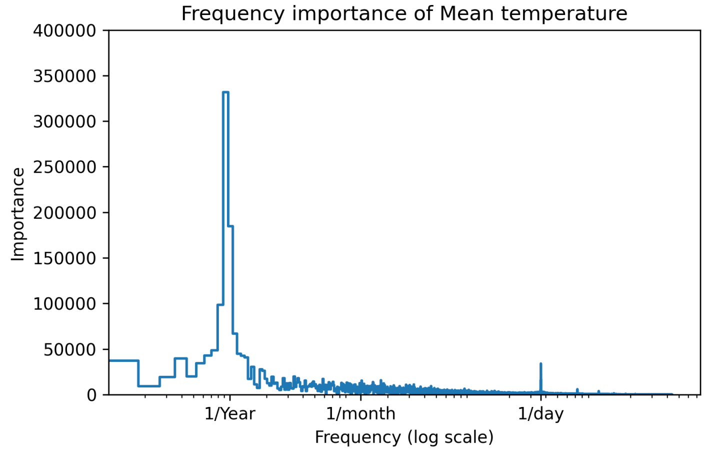
Fig: wind speed visualization before and after vectorizing it.

Angles do not make good model inputs: 360° and 0° should be close to each other and wrap around smoothly. Direction shouldn't matter if the wind is not blowing. But this will be easier for the model to interpret if you convert the wind direction and speed columns to a wind vector. The distribution of wind vectors is much simpler for the model to correctly interpret. (For good model always keep data in same coordinate system)

Amit’s LSTM - continued

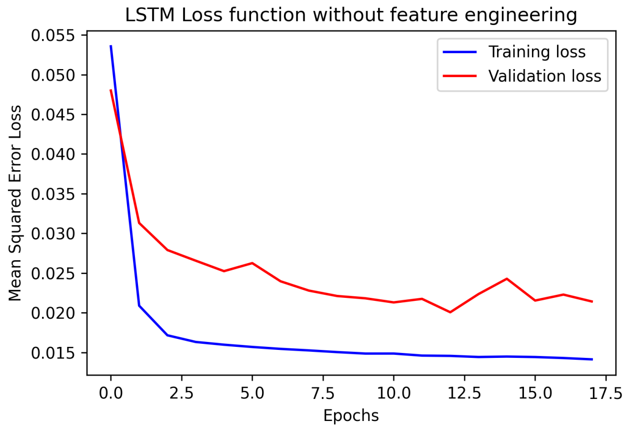


- **Being weather data, it has clear daily and yearly periodicity. There are many ways we could deal to preserve periodicity feature of our data.**
- We can get usable signals by using sine and cosine transforms to clear "Time of day/year" signals.
- This gives the model access to the most important frequency features. In this case we knew which frequencies were important.
- If we don't have that information, we can determine which frequencies are important by extracting features with Fast Fourier Transform. To check the assumptions, here is the fft of the temperature over time. Note the obvious peaks at frequencies near 1/year and 1/day.



Amit’s LSTM – continued 2

- Hyper-parameters :
  - batch_size = 20
  - epochs = 30 (with early stopping callback)
  - optimizer = Adam
  - learning_rate = 0.001
  - sequence_length = 120
  - sampling_rate = 1
  - Loss function = MSE
- Model description :
  - Input layer : (batch_size, sequence_length, features)
    - shape (20, 120, 13) with feature engg.
    - shape (20, 120, 7) without feature engg.
  - LSTM layer with 32 units
    - shape (20, 32)
    - activation function - tanh
    - recurrent activation function - sigmoid
  - Output layer with shape (20, 1)



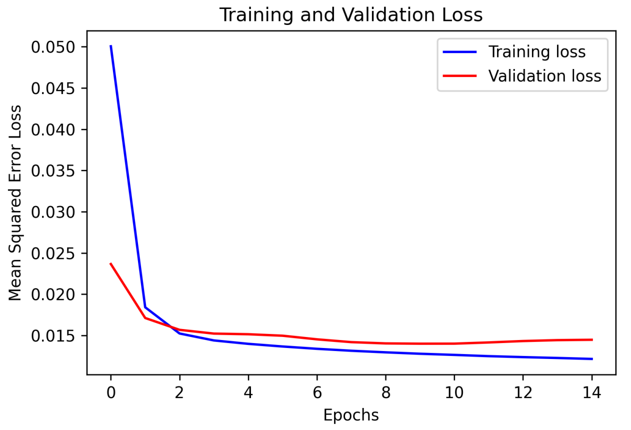
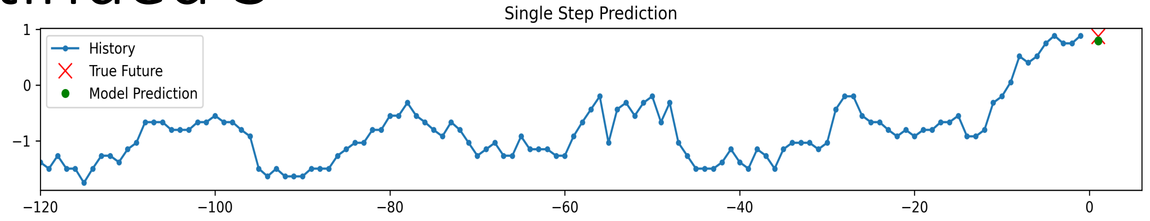
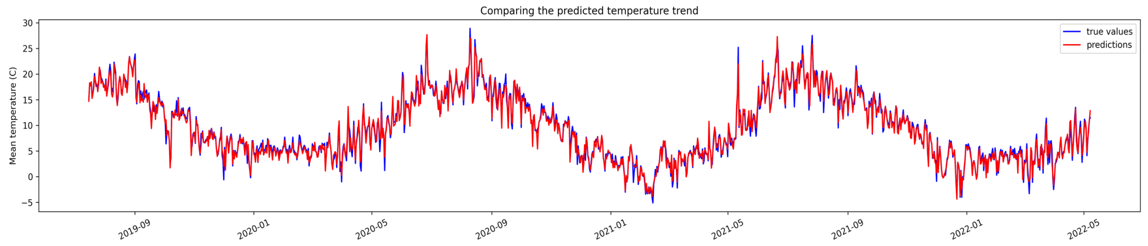


Fig: loss function without feature engg. (above) vs. with feature engg. (below) which converges faster than above.

Amit’s LSTM – continued 3

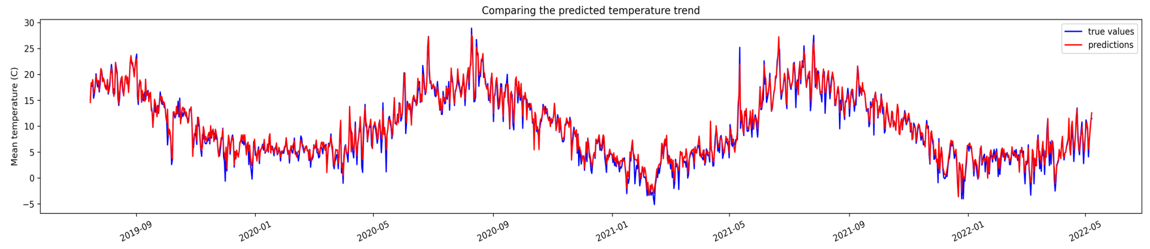
Single time-step prediction for 1 hr (forecast horizon) in future by taking past 120 hrs (sequence length).





Results with feature engineering model. (RMSE: 0.794 °C and MAE: 0.576 °C) Plotted all the 1233 single time-step predictions together to compare the general trend of temperature.

Results without feature engineering model. (RMSE: 1.019 °C and MAE: 0.725 °C) Plotted all the 1233 single time-step predictions together to compare the general trend of temperature. (looks kind of worse than above plotted results as you can see more blue part around minimum)



Bence’s LSTM

One Feature LSTM (OFLSTM):

- Model built using PyTorch.
- Model structure is an LSTM layer of hidden size 8 with one input and a final linear layer with one output.
- Data scaled with sklearn's MinMaxScaler
- Data shaped into a special format for RNN use. Basically many sequences of data which roll by one value for each sequence. For example: data = [0, 1, 2, 3, 4, 5]. Sequences are [0, 1, 2], [1, 2, 3], [2, 3, 4] for a sequence length of 3.
- Model was trained and optimized with Adam optimizer. Loss function was RMSE = 0.84
- As this model only took one feature in, we used the mean temperature and predicted the mean temperature. Only one timestep ahead could be predicted

Bence's LSTM – continued

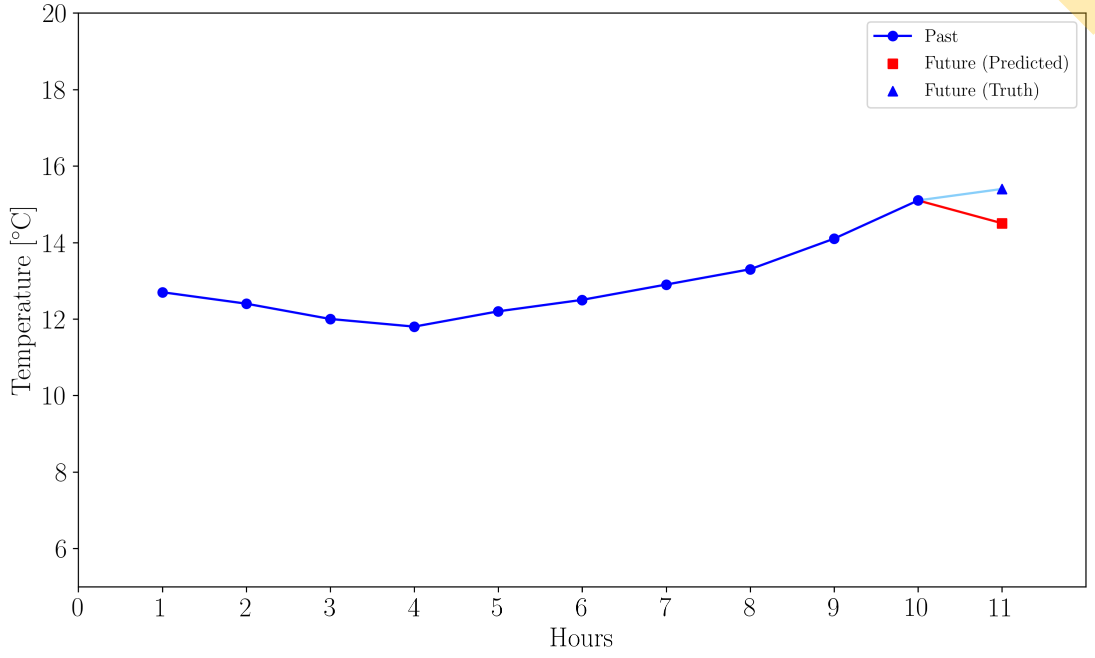
Hyperparameters of OFLSTM:

- Learning rate = 5e-3
- N_epochs = 1000
- Sequence Length = 10
- Forecast Horizon = 1 (number of future timesteps to predict)

Results:

Predicting one hour ahead with 10 hours of data from June 8, 2022.

Predicted value was 5.8% off from truth value, although the gradient change should also be considered.



Bence’s LSTM – continued 2

Multi-feature LSTM (MFLSTM):

- Model built using Keras TensorFlow
- Model structure is a 10-unit LSTM with 0.01 dropout and a final linear activation layer
- Data scaled with sklearn's MinMaxScaler
- Data shaped into a special format for RNN use, same as for OFLSTM
- Model was trained and optimized with Adam optimizer. Loss function was RMSE = 0.09
- This model took in all input features and could predict on one feature. Can predict multiple timesteps into the future

Bence's LSTM – continued 3

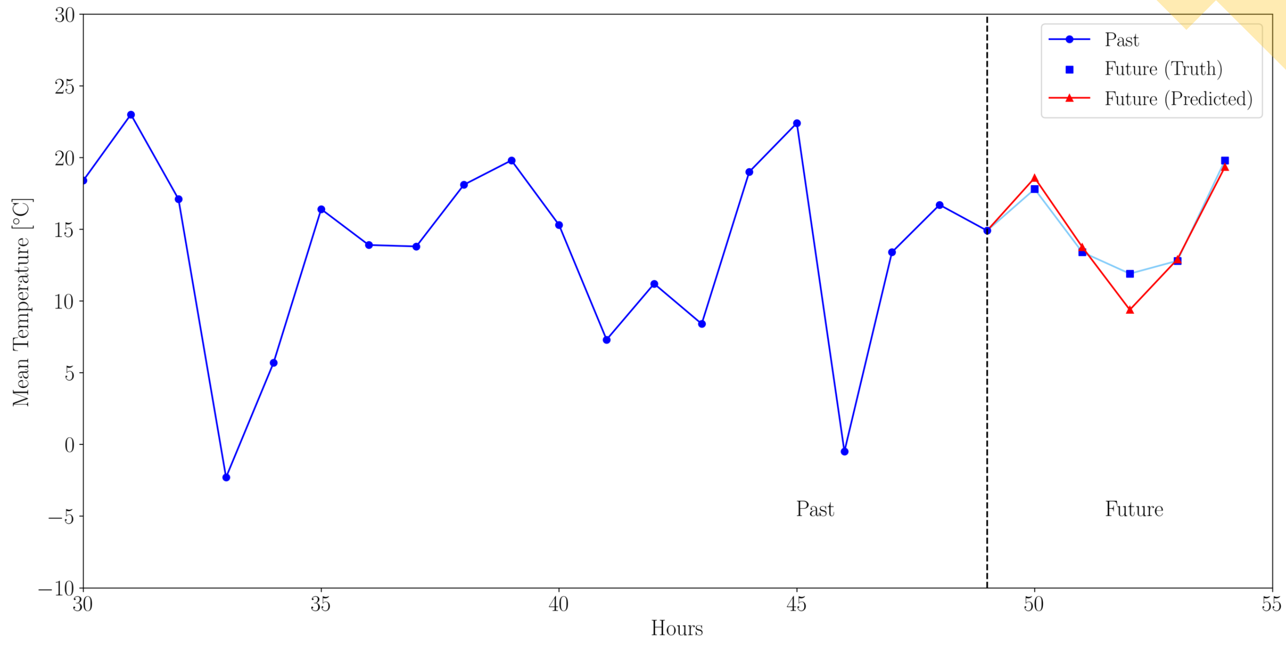
Hyperparameters of MFLSTM:

- Learning rate = 1e-3
- N_epochs = 15
- Batch size = 50
- Sequence Length = 50
- Forecast Horizon = 5

Results:

Predicting 5 hours ahead with 50 hours of data

Predicted values were, on average, 6.3% off from truth value (less than 3% excluding hour 52), and generally agreed with gradient changes



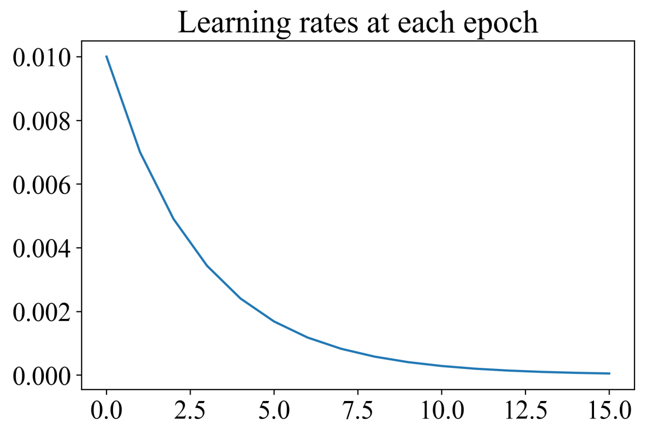
GRU

- Implemented in Pytorch (cpu)
- Data scaling: MinMaxScaler (from scikit-learn)
  - This is justified because after cleaning the data no strong outliers were present.
  - Tried StandardScaler too, did not change appreciably.
- Loss function: MSE
- Model structure:

```
GRUNet(
  (gru):  GRU(input_size=8, hidden_size=256, num_layers=2, batch_first=True, dropout=0.2)
  (fc):   Linear(in_features=256, out_features=1, bias=True)
  (relu): ReLU()
)
```

GRU -- continued

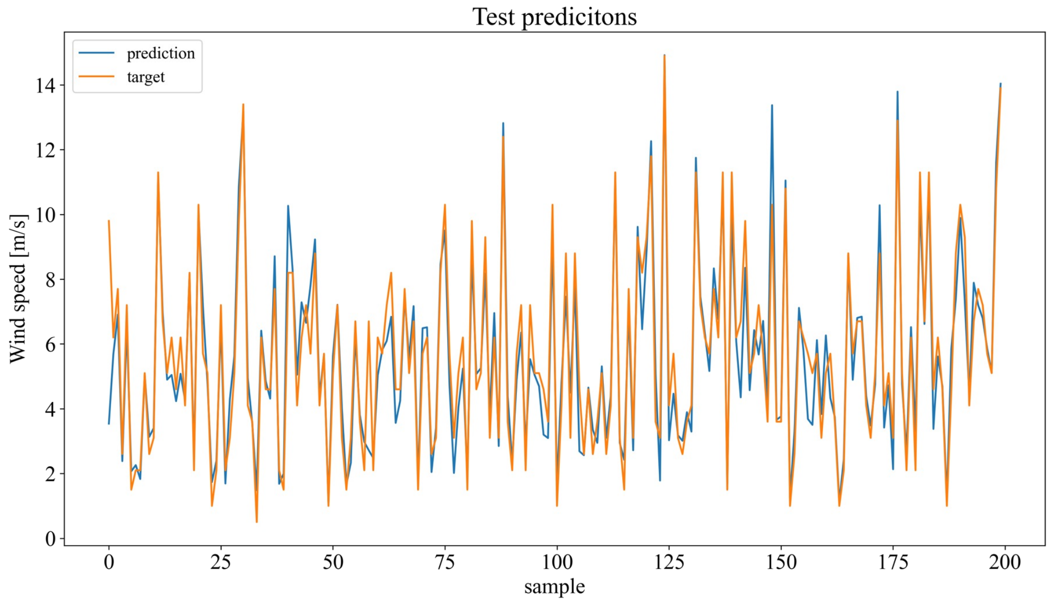
- Hyperparameters:
  - Sequence length: 96 (i.e. 4 days)
  - Forecast horizon: 1
  - Batch size: 512
  - Learning rate: Exponential Scheduler (gamma=0.7)
  - Epochs: 15
  - Dropout rate: 0.2
- HP optimization: manual
  - Tried a few values of
    - Epochs
    - learning rates (both constants and gamma value in scheduler)
    - Sequence length
    - Batch size (smaller gave worse results)
- Overfitting prevention
  - Visually inspected validation loss history
  - Dropout layer after each GRU layer



Training time: 98 min on Intel Core i5-1135G7

GRU: wind speed

- Wind speed as target
- Same structure as final GRU
- Same HP as final GRU, except for:
  - Epochs: 10
- RMSE (test): `0.88`



GNN

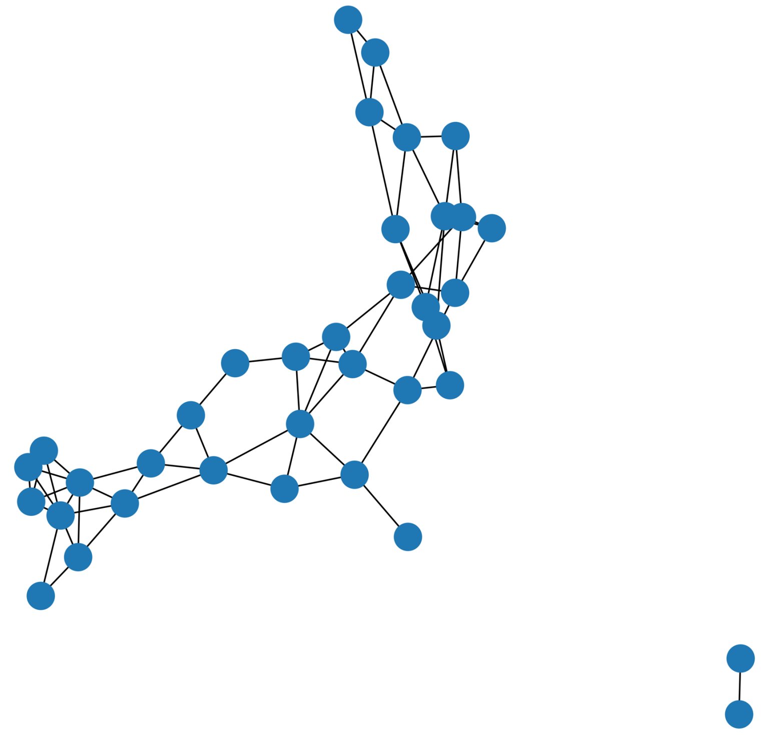
- Using pytorch geometric
  - We had MANY issues installing this, and in the end it ONLY worked on Google Colab with the help of Rasmus Ørsøe (TA for the course in 2021).

- GNN’s ingest data as graph, where in this case each station is considered a node in the graph
  - This is rather hard to structure and read in the data properly with a “real” dataset instead of example datasets.
  - Through correspondence with Rasmus we made this work after a while of trying

GNN – continued

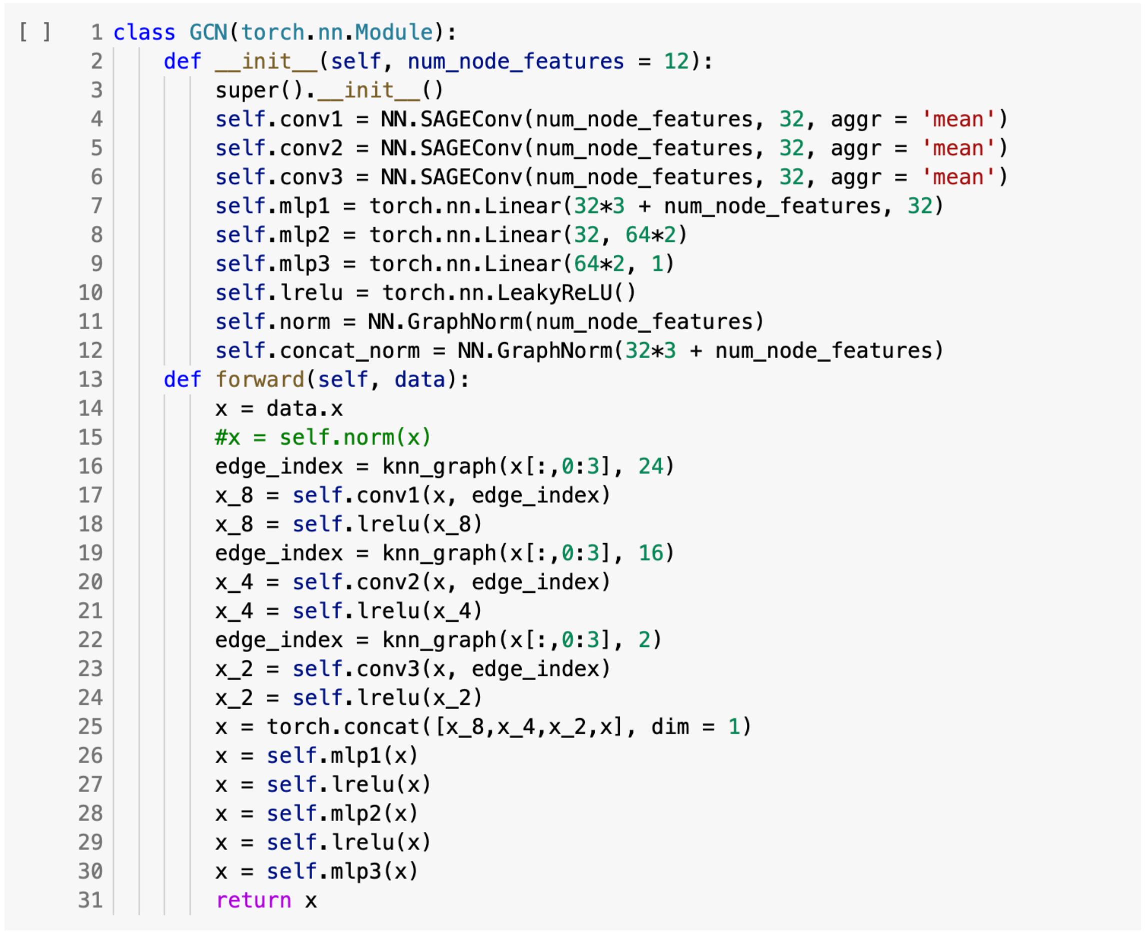
- Sample graph of 36 DMI stations
  - Edges selected based on K-Nearest Neighbours in coordinate space
- 2 groups of stations
  - The small one is presumably Bornholm, and the larger one is the rest of Denmark

Graph of DMI Stations



GNN – continued 2

- Structure of the GNN ➡
- Similar to a generalised CNN
  - Needs “conv” layers, altough these are GNN specific
  - Here we use SAGEConv, which is one of the standards
  - 3 linear layers after GNN layers
- PyTorch requires forward function and other standard “Boilerplate” code.



```python
class GCN(torch.nn.Module):
    def __init__(self, num_node_features = 12):
        super().__init__()
        self.conv1 = NN.SAGEConv(num_node_features, 32, aggr = 'mean')
        self.conv2 = NN.SAGEConv(num_node_features, 32, aggr = 'mean')
        self.conv3 = NN.SAGEConv(num_node_features, 32, aggr = 'mean')
        self.mlp1 = torch.nn.Linear(32*3 + num_node_features, 32)
        self.mlp2 = torch.nn.Linear(32, 64*2)
        self.mlp3 = torch.nn.Linear(64*2, 1)
        self.lrelu = torch.nn.LeakyReLU()
        self.norm = NN.GraphNorm(num_node_features)
        self.concat_norm = NN.GraphNorm(32*3 + num_node_features)
    def forward(self, data):
        x = data.x
        #x = self.norm(x)
        edge_index = knn_graph(x[:,0:3], 24)
        x_8 = self.conv1(x, edge_index)
        x_8 = self.lrelu(x_8)
        edge_index = knn_graph(x[:,0:3], 16)
        x_4 = self.conv2(x, edge_index)
        x_4 = self.lrelu(x_4)
        edge_index = knn_graph(x[:,0:3], 2)
        x_2 = self.conv3(x, edge_index)
        x_2 = self.lrelu(x_2)
        x = torch.concat([x_8,x_4,x_2,x], dim = 1)
        x = self.mlp1(x)
        x = self.lrelu(x)
        x = self.mlp2(x)
        x = self.lrelu(x)
        x = self.mlp3(x)
        return x
```

GNN - continued 3

- We made the model run, BUT each epoch takes many hours to run, even on a datacenter GPU and a largely reduced dataset
  - Google Colab provides Nvidia Tesla T4 GPU with 16GB VRAM
  - 1% of the dataset takes ~2 hours per epoch, and a few epochs gave no usable results
    - More epochs or more data would exceed the available time, both clock-time and GPU compute time
- GNN’s look really promising for this kind of data but will require more time to work out small bugs along with access to more GPU compute (either more GPU’s or more time for one GPU)
  - Can become very expensive