Michal Chowaniak

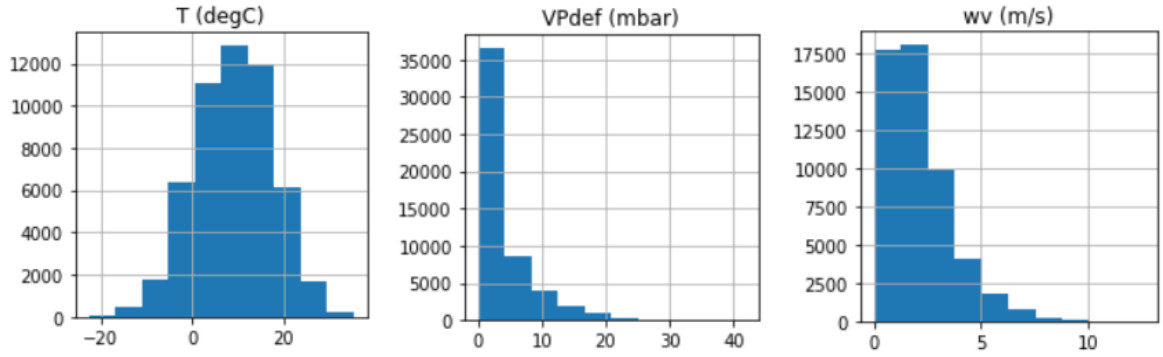
CSC 578_901_1030

Kaggle username: Michal C

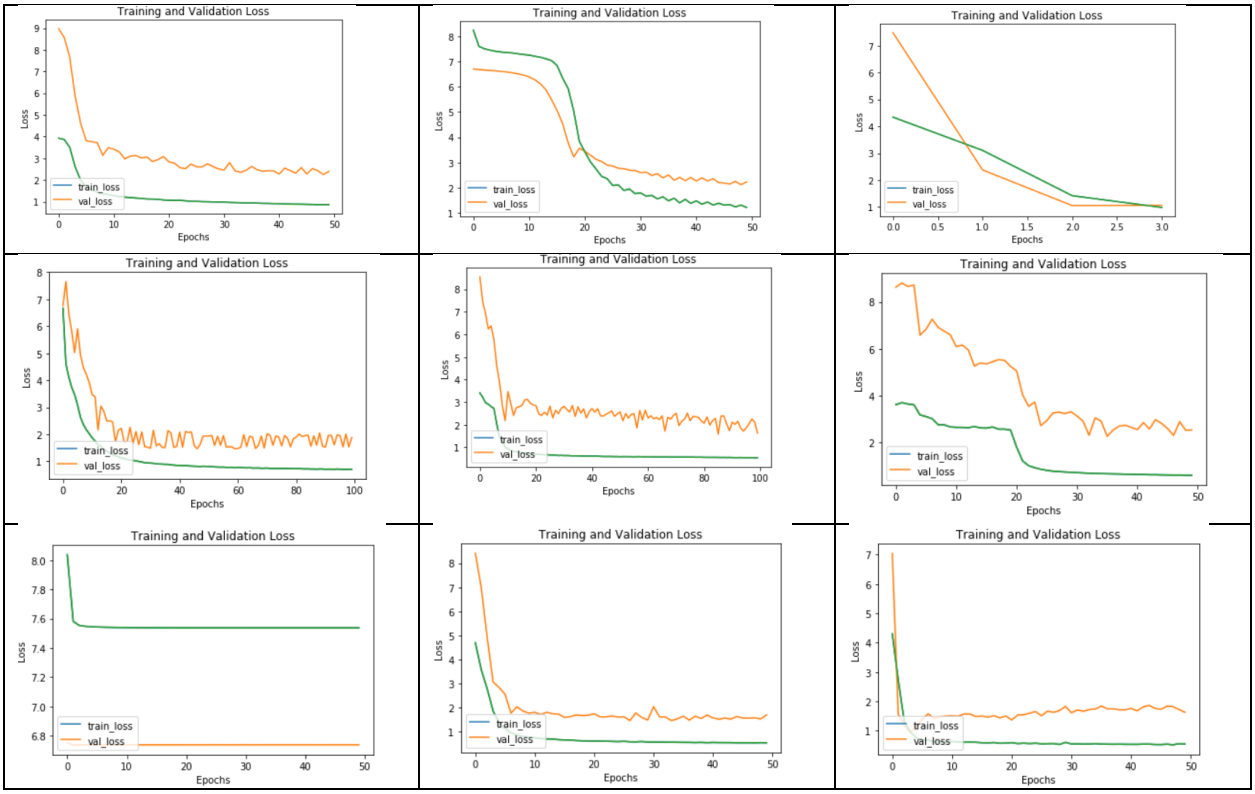
Public and Private Ranking: 31

Final Project

Climate training dataset included 52566 hourly observations, which consisted of 14 following measurements: atmospheric pressure (p (mbar)), temperature (T (degC)), potential temperature (Tpot (K)), dew point temperature (Tdew (degC)), relative humidity (rh (%)), saturation water vapor pressure (VPmax (mbar)), actual water vapor pressure (VPact (mbar)), water vapor pressure deficit (VPdef (mbar)), specific humidity (sh (g/kg)), water vapor concentration (H2OC (mmol/mol)), air density (rho (g/m**3)), wind velocity (wv (m/s)), maximum wind velocity (max. wv (m/s)), wind direction (wd (deg)). At first, I trained on full train dataset, and later I decided to use the validation set, which was half of the training dataset. Test data set included 17447 observations; the target variable was not separated. The minimum temperature for training was -22.76 and maximum was 35.48 degrees of Celsius. There were no missing values. Temperature distribution seemed to be normal, but some other variables like water vapor pressure deficit, wind velocity were highly skewed. As expected, the temperature looked seasonal, which suggested I the analysis could be performed on differences, not actual values, however, because of the test dataset format I decided to work with values.



The goal of this project was to predict the temperature at the 25th hour, by reading a window of the past 24 hours at the time. In the beginning, while developing my code, I run several models on full train dataset, without validation set and I achieved a test score of 1.3. Next, I created a smaller training set and validation set. I spent most of the time developing models using various LSTM layers and test and validation, but after dozens of various models, I could not get test score any better than models run on the full training set. I run the same models using various optimizers, like adam, rmsprop, adadelta. Out those 3, adadelta performed the best and was the quickest. I experiment with bath size, models run with larger batches run much faster, but were less accurate. The other problem was that some models run with low batch size did not perform much better. Training results I could get down to 0.5-0.6 range, but validation results were usually much higher. Using drop out layers did not help much. I run several more complicated models, which took a long time to train. I noticed that simpler models performed better, I run several models with single LSTM layer and various numbers of units from several hundred to a few, it seems that larger number of units, the better the model performed, but again overfitting was a problem, even after adding drop out and recurrent drop out arguments. Below are examples of a few model results.



Facing an uphill battle, I decided to go back to the beginning, and I started experimenting with different types of recurring layers using the full training set. Finally, GRU, Gated Recurrent Unit, produced my best model with a test score of 0.88. I run this model dozens of times with different settings like various number of epochs (between 20, 30, 40, 50 and 60), various number of units (32, 64, 128, 240), different optimizers (RMSprop, Adam, Adadelta), different types of normalizations (Min/Max and Z-score),

various batch sizes (6, 12,24, 64, 128, 240, 336). I even run it on smaller training and validation set. I used 2 normalized test sets, one provided and another one with outliers of values -9999 replaced with 0. I was able to improve this model but only minimally from 0.88 to 0.84. Please see below the final results

Training set	Testing set	Normalization	Epochs	Optimizer	Batch size
52566	17447	Min/Max	30	RSMprop lr=0.001, rho=0.9	24
Metrics	Layer	Units	Dropout	Recurrent dropout	Training/test loss
mae	GRU 1 layer/ Dense(1)	32	0.1	0.5	1.1077/**0.84**

Here is a train (1.33) and validation loss (2.24) results of my best model, but run on training and validation data, the performance is worse compared to the same model run on full training data set (1.1).

Although the results were far from my expectations and compared to the time and effort, I put into this project, I really enjoyed it. I could not help myself to do this one more experiment, to check one more idea. I tried to speed up the model development process by running several notebook instances in Amazon SageMaker, (thanks to $100 credit offered to students) to be able to train a few models at the same time, but this strategy did not help me to find that one good model, I could be proud of. The main problem in my project was that even when I was developing models with low test and validation absolute mean squared error, which was the lowest around 0.9-1.0, the Kaggle score was close to 3.0. I spent most of my time developing models using LSTM train and validation dataset, but as I started with no validation set, I had much better results. In the end, I went back and run some of my models on the full train data set. It looked like by creating a validation data set I put myself in disadvantage by having not long enough training set. In the end, I experimented with simple GRC networks, which finally produced a decent score, but it was almost twice higher than the best score in Kaggle competition.

https://drive.google.com/open?id=1J1B59G-03wKu3ww8usvRALrpUWrFB7YU