LI HUANG CSC578-701 CSC578 Final Project Summary
Kaggler user name **Cathy Huang** ranking **15**
Link to video:
https://www.dropbox.com/s/l0puwc57747nfyu/CSC578%20Final%20Project%20-%20Li%20Hua
ng.mov?dl=0

Summary of submitted competition model

- My final selected model is based on reduced dataset of Selected_Features, and then combine CNN, RNN. I have tried various other combination of architectures and parameters, like GRU, stacking layers, bidirectional RNN, dropout, recurrent dropout, learning rate, combination of dropout and recurrent dropout, but this one outperform all of them. And it is also clear and not too complicated so that it also runs faster than many other models.
- Conv1D(64,3) + MaxPooling1D(2) + LSTM (50, recurrent_dropout=0.2) + Dense (1)
  The convolutional layer consist of 64 filters of size 3*3, using padded methods and relu activation. I have tried different filter sizes and tried to add dropout, batch normalization, adding more convolutional layers, but it seems a simple layer CNN outperform other configurations.
  The pooling layer I used her is maxpooling, I have also tried average pooling and globalaverage pooling, the result is not as good as maxpooling.

  The units in LSTM used is 50, other number seems does not produce better result. The dropout I used here is recurrent_dropout.even though I have tried dropout, and combination of dropout and recurrent_dropout, and different values of recurrent_dropout, the results were not as good. I have also tried stacking the LSTM layers, adding batch normalization, and use GRU instead of LSTM, but the results favor the simple layer of LSTM.
  The final layer is Dense layer, I have tried to stacking dense layer and add dropout, but they did not produce better result.

Hyper-parameter (and architecture) tweaking

- Part 1. Architecture Experiment

After Data Importation & Preprocessing like scaling and feature selection, 24-lag Supervised Data Transformation & Train-Test Split, I wanted to try a set of basic models on original dataset to see which one performs better and worth further investigation. I could use a subset of features in the beginning as for my final model, but it would risk information loss in the beginning and could make it hard to find out which model performs better.

Model_name	Configuration	Test MAE loss
model	LSTM(50)+Dense+Epoch(15)+Batch_S(500)	0.736
model_0	LSTM(50,dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	0.863
model_1	LSTM(50,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	0.786
model_2	LSTM(50,recurrent_dropout=0.2)+LSTM(50,recurrent_dropout=0.2)+LSTM(50,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	2.810
model_3	Bidirectional(LSTM(50,recurrent_dropout=0.2))+Bidirectional(LSTM(50,recurrent_dropout=0.2))+Bidir ectional(LSTM(50,recurrent_dropout=0.2))+Dense+Epoch(15)+Batch_S(500)	1.726
model_4	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	0.604
model_5	Conv1d(64,3)+Maxpooling1d(2)+GRU(50,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	2.093
model_6	Conv1d(64,3)+Batchnormalization+Maxpooling1d(2)+LSTM(50,recurrent_dropout=0.2)+Dense+Epoch (15)+Batch_S(500)	2.035

Table1. Experiment results of Base models

As displayed in Table 1, the basic models including LSTM model, dropout/recurrent_dropout, Stacking recurrent layers & Recurrent dropout, Bidirectional RNNs & Recurrent dropout, LSTM & CNNs Combination, LSTM with Batch Normalization. I used 15 epochs for simple testing, as I started with 30 epochs, but it takes too long to test out so many models with different parameters combinations, so I decrease the epochs to 15 to produce an initial result.

- Part 2. Parameter Experiment on selected base model

Based on the base models results from Table1, I chose model_4 as the model to investigate further and perform parameter experiments in Table 2. I have also tried the second best model, which is the base model to perform further investigation, but the performance is very similar as model_4, not better, so I focused on model_4 for the experiments.

Model_name	Configuration	Test MAE loss
model_4_1	Conv1d(64,3)+Maxpooling1d(2)+Dropout(0.25)+LSTM(50,recurrent_dropout=0.2)+Dense+Epoch(15)+ Batch_S(500)	1.745
model_4_2	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50,dropout=0.2,recurrent_dropout=0.2)+Dense+Epoch(15)+Ba tch_S(500)	1.480
model_4_3	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50,recurrent_dropout=0.2)+LSTM(50,recurrent_dropout=0.2) +LSTM(50,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	2.659
model_4_4	Conv1d(128,3)+Maxpooling1d(2)+LSTM(100,recurrent_dropout=0.2))+Dense+Epoch(15)+Batch_S(500)	1.085
model_4_5	Conv1d(64,3)+Maxpooling1d(2)+Conv1d(64,3)+Maxpooling1d(2)+LSTM(100,recurrent_dropout=0.2)) +Dense+Epoch(15)+Batch_S(500)	1.131
model_4_6	Conv1d(32,3)+Maxpooling1d(2)+LSTM(100,recurrent_dropout=0.2)+Dense+Epoch(15)+Batch_S(500)	1.063
model_4_7	Conv1d(64,3)+Maxpooling1d(2)+LSTM(100,recurrent_dropout=0.4)+Dense+Epoch(15)+Batch_S(500)	1.604
model_4_8	Conv1d(64,3)+Maxpooling1d(2)+LSTM(100)+Dense+Epoch(15)+Batch_S(500)	0.920
model_4_9	Conv1d(64,3)+GlobalAveragepooling1d(2)+LSTM(50)+Dense+Epoch(15)+Batch_S(500)	0.821
model_4_10	lr(0.015)+Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(15)+Batch_S(500)	2.700
model_4_11	Conv1d(64,3)+Averagepooling1d(2)+LSTM(50)+Dense+Epoch(15)+Batch_S(500)	1.428

Table2. More Experiments on Base models

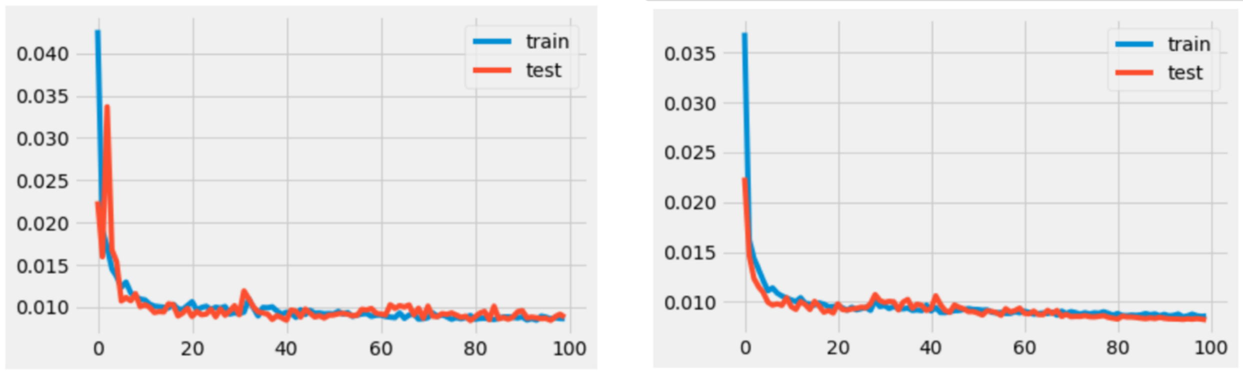
- Part 3. Final Model fine_tuning and feature selection Experiment

Based on the best model selected so far, model_4, I tried to firstly tune epochs and batch_sizes, I have also tried batch-sized in 80,300,400, but none of them perform as good as 200. And when using the original data, increasing of epoch size from 50 to 100 did not increase the performance; while using selected features, the result is improved. The selected features are based on the plots and correlation table of the 14 features, and there are four features that seems not highly correlated with our target feature, 'p (mbar)','wv (m/s)', 'max. wv (m/s)','wd (deg)', with correlation less than 0.1. That is why I tried to fit my best model using a reduced dataset with 10 selected variables, and the result is the best I have achieved so far.

Model_name	Configuration	Test MAE loss
model_f_1	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(50)+Batch_S(100)	0.536
model_f_2	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(15)+Batch_S(200)	0.639
model_f_3	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(50)	0.499
model_f_4	Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(100)	0.526
model_f_5	Selected_features+Conv1d(64,3)+Maxpooling1d(2)+LSTM(50)+Dense+Epoch(100)	0.492

Table3. Parameter fine_tuning on the selected model

- Part 4. Plot of the best two models



Graph1: Plots of best model using original data(left) and selected features(right)

As from the above two plots, the Plots of best model using all feature and selected features below show that the selected 10 features have created a more smoothed dataset for time series prediction and thus achieved better result than the model using all features. As we can see from the plots, the loss plots for training datasets are very similar; while for test dataset, there is huge fluctuation in the plot using all features, and the loss plot using selected features are much more stable over the 100 epochs performance. Overall, both models showed a downward trend of loss toward 0 as the epoch increases.