Kubra Iqbal
CSC 578 Final Projects
Kubra Iqbal
Kaggle Username: kubraiqbal
Leaderboard Number: 19

Summary of the model that worked the best:

LSTM/CudnLSTM Layer

1. Nodes: 60
2. Epochs: 30
3. Optimizer: Adam
4. Loss Function: MAE
5. Batch Size:20
6. Dropout : None

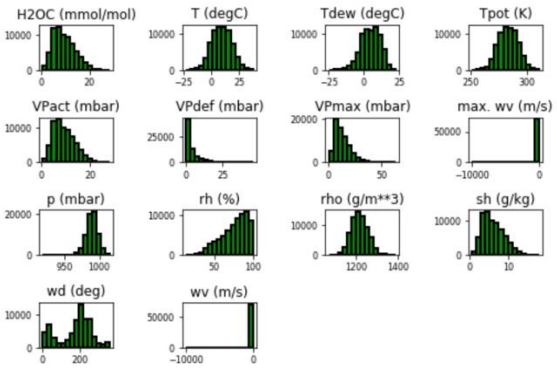
Dense Layer(s):

1. Size: 1 dense layer, size 1
2. Activation Function: default

Data Preparation:

For Data Preparation – it took a long time to understand and how to prepare this data for the model. The toughest part was restructuring the data, understanding the normalization part.
Below is how the data looked when I started the project:

```
In [0]: climatedata.hist(bins=15, color='green', edgecolor='black', linewidth=2.0,
                         xlabelsize=8, ylabelsize=8, grid=False)
        plt.tight_layout(rect=(0, 0, 1.2, 1.2))
```



Summary:
For my final project, I tried 21 models with different combinations to check which model would perform the best for me. I personally got the best result with what could be called a “base model” with some minor changes in it. I did multiple changes in this same model that worked the best for me to see what would happen if I would change the batch size, increase or decrease the number of epochs or change the number of nodes. I also tried different Optimizers in different models but turned out that Adam did

the best for me so I kept it Adam for most of the models I ran. The best score that I got to increase was 1.69 approximately. Even though I worked on different models and different experiments, this is the best I could come up.
For some future work, I would come up with different models –make changes to different lines to see how it turns out, and see if the model performs better. Even though I tried 21 models, it was very interesting to see how some models didn't change the results at all or changed them drastically positively and negatively.

Below is the Description of the Hyper Parameter Tuning that I implemented:

LSTM NODES :
For LSTM Nodes – my goal was to try a range of values between 1 and 512 but I noticed that increasing the values a little or decreasing them was not really helping with my model. I ran various models with various transitions, but then I noticed the values ranging between 40-60 were doing the best for my model. I remained constant with this range for most of the last models I ran, and 60 was that performed the best for me. The model that I picked and gave me best score had 60.

CONV1D :
For CONV1D, I was able to try different models. I added differed sizes, different activation functions. I tried adding Relu which gave good results – but after doing some research I figured out I can't really use Relu and had to move forward without using that model/results.
As I was adding more filter sizes and different activation results – I noticed that it was not making my results any better. It was just making it more complicated and the results were not turning out to be that great so in my final model I ended up removing it.

Dropout:
I did not experiment around with the Dropout a lot – I started adding that when I was first building models but my error was increasing significantly which I thought was very surprising. I tried the dropout rates ranging from 0.25 to 0.5 but this was not helping the model. Adding a separate dropout layer seemed to have a very similar effect as well. So moving forward, when my models started improving and looking better – I removed my drop out layer. And hence no dropout layer was used in my Final model.

Dense Layers:
For Dense Layers I tried a few different combinations, but nothing really worked that significantly to keep that change in my model. I proceeded with the dense layer of size 1 in most of my models and my final model.

Learning Rate:
As I was increasing the number of epochs, I tried different values within the Adam Optimizer for the Learning Rate. I noticed that it was not helping that much, so I let it to be on default settings.

Epochs:
I tried epochs ranging from 10 to 60. The best one that worked for me was 30.

The diagrams below show the code for my model that I picked and the Graph:

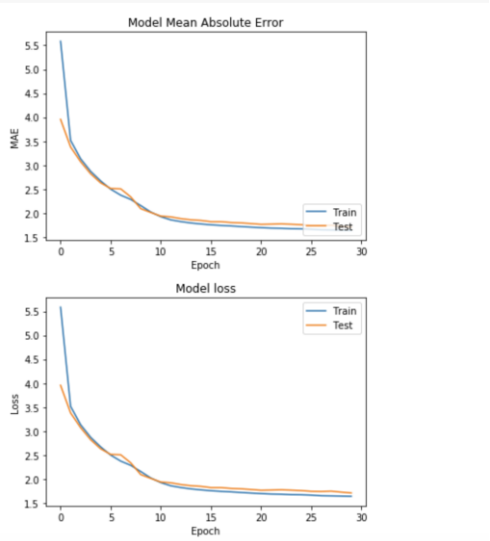
```
model_24 = Sequential()
model_24.add(CuDNNLSTM(20, input_shape=(48,4), kernel_regularizer= regularizers.l2(1e-4)))
model_24.add(Dense(1, activation='linear'))
model_24.compile(optimizer='adam', loss='mean_absolute_error', metrics=['mae'])

model_24.summary()

model_24_results = model_24.fit(reshaped_normalized_x_train, y_train,
          batch_size=45,
          epochs=30,
          validation_data=(reshaped_normalized_x_validation, y_validation),verbose=2, shuffle=False)
```

```
Model: "sequential_48"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
cu_dnnlstm_41 (CuDNNLSTM)    (None, 20)                2080
_________________________________________________________________
dense_44 (Dense)             (None, 1)                 21
=================================================================
Total params: 2,101
Trainable params: 2,101
Non-trainable params: 0
```

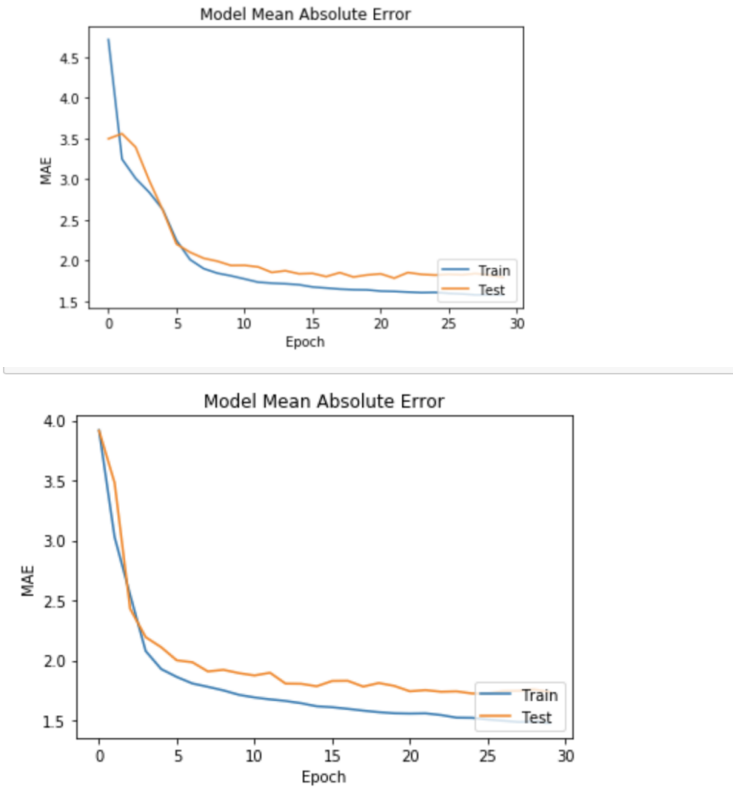
The picture above is a visualization of the model I thought did the best for me. The explanation above shows everything that I did and how many changes I made on each model. Even though I haven't showed 21 models In my write up, the explanation just shows what changes I made briefly as I proceeded with it.



The graph shows that there is no over fitting and is the graph of the model that I picked as my best model.

Below are some graphs from other two models:

Below are the graphs for the other two models that did well for me, but comparing them to the graph above of the model that I had picked – that model did better. There was less over fitting and the results also turned out to be improved.



Over All Experience with the project:

Since I did 21 models – each model performed differently. I didn't submit 21 models to my Kaggle Submission because not all of them performed at their greatest ability – but the models started doing better around my 10th submission. The score at that time was 1.7 approximately – with that model I tried various different ways to make it better – and the best I could bring it to be 1.69 approximately. The Notebook attached with my final project shows each and everything in detail about how I made it to the base model and explaining everything.

This table shows my three best models – how each of them scored and what changed.

Model 1	Epoch 30/30 - 5s - loss: 1.6419 - mean_absolute_error: 1.6388 - val_loss: 1.7112 - val_mean_absolute_error: 1.7081
Model 2	Epoch 30/30 - 10s - loss: 1.4903 - mean_absolute_error: 1.4785 - val_loss: 1.7482 - val_mean_absolute_error: 1.7365
Model 3	Epoch 30/30 - 7s - loss: 1.5823 - mean_absolute_error: 1.5773 - val_loss: 1.7989 - val_mean_absolute_error: 1.7939

The highlighted model is the model that was my final model that gave me the best results. Comparing it to the other two models below, they also gave improved results and were close to what I had for the first model.

Number of Submissions and the scores that I got:

Submission	Score
1	3.13999
2	3.17008
3	3.10446
4	3.10445
5	2.89254
6	2.68840
7	3.38529
8	3.22162
9	2.76939
10	1.79242
11	1.81386
12	1.69232

The Table above shows my journey of how I did this project and my submissions. I started with a very high number and couldn't really understand where I was going wrong, I did figure out some errors in my code and by the last submission I was able to build a model that had a good result.

Reflection:

The Final Project was very interesting, even though it was similar to Homework 6 in terms of the Kaggle Submissions – I had an easier time with that homework building the model. In this project, it was a little tricky with understanding first about what was happening, understanding how to normalize X and making changes with the data, such as removing rows. Etc.

Once I got the Base model running and after a few submissions when I felt confident in what I was doing, I was able to bring it up to 1.692 approximately.

If I had a little more time – I would have tried even more different models to see if they would help. One important thing that I took from this project was, that I read a lot of articles, looking at what were the others were doing with their models, articles explain everything in so much more detail that I understood what was supposed to be done in a very precise way.

I have mentioned this in my previous reflections as well, Kaggle was something new to try and I think it was a great way to learn and experiment something new.

On the whole – this class and project taught me a lot. Even though it has been challenging at some points and I felt over whelmed, every week I came out of it more prepared and was learning new material.