Weather Forecasting using Deep Learning
Homework 5
ECSE 552

Hung-Yang Chang (260899468)
Alexander Fernandes (260960205)

Due 2 April 2021

Contents

1. *Introduction*

For this homework assignment we are performing time-series forecasting using deep learning methods for regression analysis. The goal can be summarized as using previous time steps to predict current and future time steps for specific values. Here we first outline the goal for our time series forecasting problem and the data set we are using. Then we go into more detail in the methods used to process the data and three architectures used to perform the forecasting. After we present and compare the results between the three models.

1.1 Time Series Forecasting

We are using the weather time-series data set recorded by the Max Planck Institute for Biogeochemistry from 2009 to 2016 to predict the following weather attributes:

- p (mbar), atmospheric pressure
- T (degC), air temperature
- rh (%), relative humidity
- wv(m/s), wind velocity

Using the time-series data available to us the goal is to make predictions using past time steps to predict current and future time steps. Figure 1.1 gives a visual of a basic overview of time series prediction of the current time step t using the previous 4 time steps.

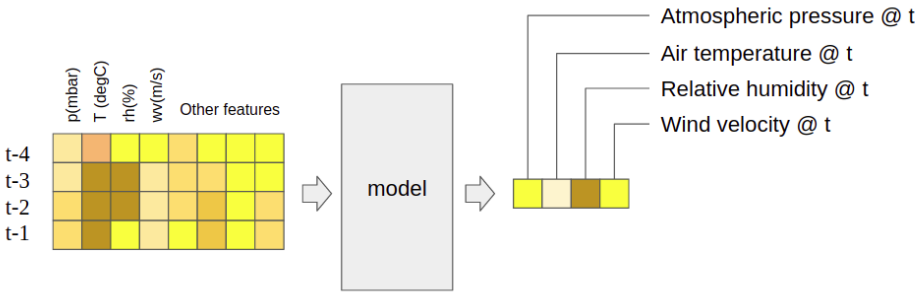

Figure 1.1: Problem overview to predict current time step using previous four time steps.

1.2 Data set

The time-series data we are using is sampled hourly. The description of the features for the Max Planck Institute for Biogeochemistry time series data is described in Table 1.1.

Table 1.1: Max Planck Institute for Biogeochemistry time series data set feature description.

Label	**Description**
p (mbar)	The pascal SI derived unit of pressure used to quantify internal pressure. Meteorological reports typically state atmospheric pressure in millibars.
T (degC)	Temperature in Celsius
Tpot (K)	Temperature in Kelvin
Tdew (degC)	Temperature in Celsius relative to humidity. Dew Point is a measure of the absolute amount of water in the air, the DP is the temperature at which the air cannot hold all the moisture in it and water condenses.
rh (%)	Relative Humidity is a measure of how saturated the air is with water vapor, the %RH determines the amount of water contained within collection objects.
VPmax (mbar)	Saturation vapor pressure
VPact (mbar)	Vapor pressure
VPdef (mbar)	Vapor pressure deficit
sh (g/kg)	Specific humidity
H2OC (mmol/mol)	Water vapor concentration
rho (g/m**3)	Airtight
wv (m/s)	Wind speed
max. wv (m/s)	Maximum wind speed
wd (deg)	Wind direction in degrees

- Location: Weather Station, Max Planck Institute for Biogeochemistry in Jena, Germany Time-frame: Jan 1, 2009 - December 31, 2016
- Description: This sequence data represents weather timeseries recorded at the Weather Station at the Max Planck Institute for Biogeochemistry in Jena, Germany. The dataset includes date-time and 14 climate measurements/features
- (e.g., atmospheric pressure, temperature and humidity), which were originally recorded every 10 minutes over 9 years. For this project the data was reduced to be every hour given in the assignment.

The number of time series data points given is 56072 (1 Jan 2009 to 24 May 2015) for training data and 14019 (24 May 2015 to 31 Dec 2016) for test data. Approximately (1:4) test to train ratio.

2. *Methods*

2.1 Data Setup

2.1.1 Data Preprocessing

To get the general behavior of given data, describe function is utilized to print out mean, std, max, min, and other data information in training and testing function. It's noteworthy that "wv (m/s)" and "max. wv (m/s)" have minimal value -9999. It's assumed that wind velocity should be greater than 0, so all values less than 0 in "wv (m/s)" and "max. wv (m/s)" is replaced with 0. Also, all data is in chronological order, even if the "Date time" column is removed, other information still stays in the same order to represent time-series data as long as data is not shuffled. After cleaning up data, normalization is performed in both training and testing data to increase the performance when feeding into models.

2.1.2 Feature Selection

Discrete Fourier Transform

With the FFT we will be able to observe any underlying periodic nature from the feature. Many features correspond to 1 year or 1 day periodicity. We may be interested in using features that have high 1-day or 24-hour periodicity. In Figure 2.1 we show the fourier transform of the four weather attributes to be predicted. We can see that p (mbar) has the lowest 1/day peak, mostly composed of low frequencies.

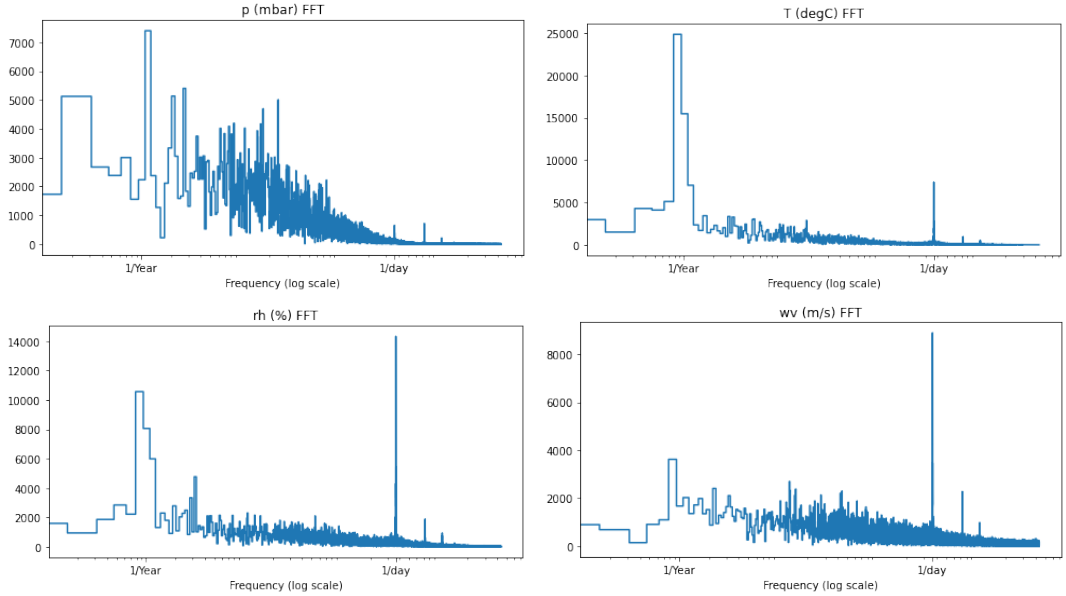

Figure 2.1: Discrete Fourier Transform of the time series data.

List of features with a distinct 1 day frequency peaks:

1. T (degC)
2. Tpot (K)
3. rh (%)
4. VPmax (mbar)
5. VPdef (mbar)
6. rho (g/m**3)
7. wv (m/s)
8. wd (m/s)

Autocorrelation

The autocorrelation of a time series describes the correlation between the lag between time points. It is important to remove high autocorrelation to make the process more stationary (constant mean and variance).

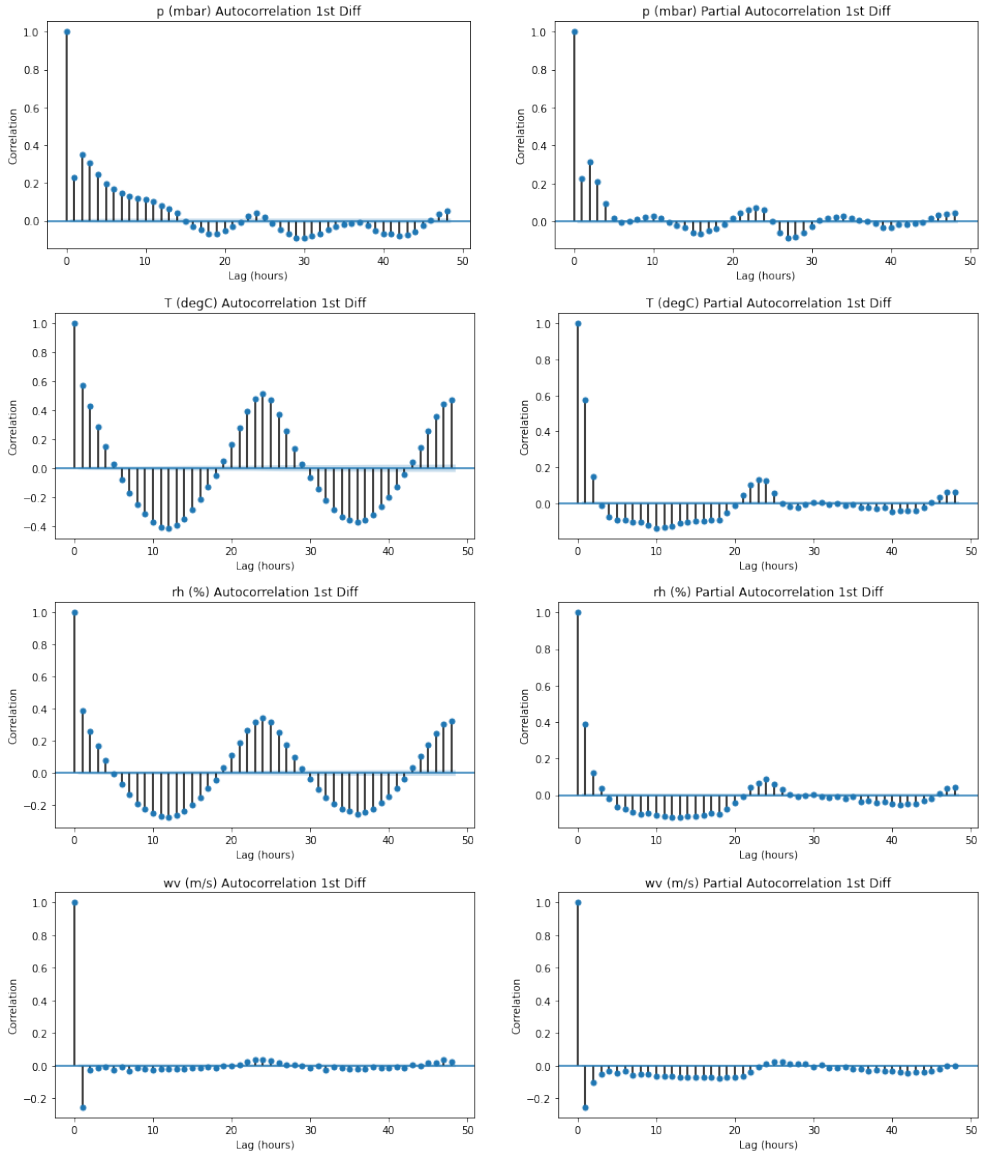

Figure 2.2: Autocorrelation and Partial Autocorrelation.

This can be fixed by taking the first difference to make the mean zero throughout all time. data diff = data[t] - data[t-1]. This first order difference is like a high pass filter to remove DC component. It might be good to then use only features with predictable periodicity autocorrelation. The auto correlation of white noise is an impulse function at zero. Basically means any feature similar to white noise is like using unpredictable random data as a feature.

Figure 2.2 shows the autocorrelation and partial autocorrelation of the first difference of the four weather attributes to be predicted.

We see T and rh have a relatively periodic autocorrelation and similar partial autocorrelation. Where as wind velocity seems to have very little autocorrelation for other time lags with somewhat more partial correlation at different time lags. Pressure seems to mostly be correlated with shorter time lags.

Features that do not resemble white noise after taking first difference are:

1. p (mbar)
2. T (degC)
3. Tpot (K)
4. rh (%)
5. VPmax (mbar)
6. VPdef (mbar)
7. rho (g/m**3)

Pearson Correlation

Figure 2.3 shows the pearson crosscorrelation across the first difference of all features. The Pearson correlation measures how two continuous signals co-vary over time and indicate the linear relationship as a number between 0 (not correlated) to 1 (perfectly correlated).

Features correlated with the four weather attributes we are interested in predicted are as follows.

p(mbar):

- rho(g/m**3) = 0.34 (highest correlated feature)

T (degC):

- Tpot (K) = 1 because it's just temp in Kelvin
- rh (%) = 0.83
- VPmax (mbar) = 0.92
- VPdef (mbar) = 0.82
- rho (g/m**3) = 0.99

rh (%):

- T (degC) = 0.83
- Tpot (K) = 0.82
- VPmax (mbar) = 0.77
- VPdef (mbar) = 0.86
- rho (g/m**3) = 0.78

wv (m/s):

- max. wv (m/s) = 0.88

From this we take all of these features listed and use them to train the model. Effectively, all other features have been discarded and are not being used.

Final set of features used: [p (mbar), T (degC), rh (%), VPmax (mbar), VPdef (mbar), rho (g/m**3), wv (m/s), max. wv (m/s)]

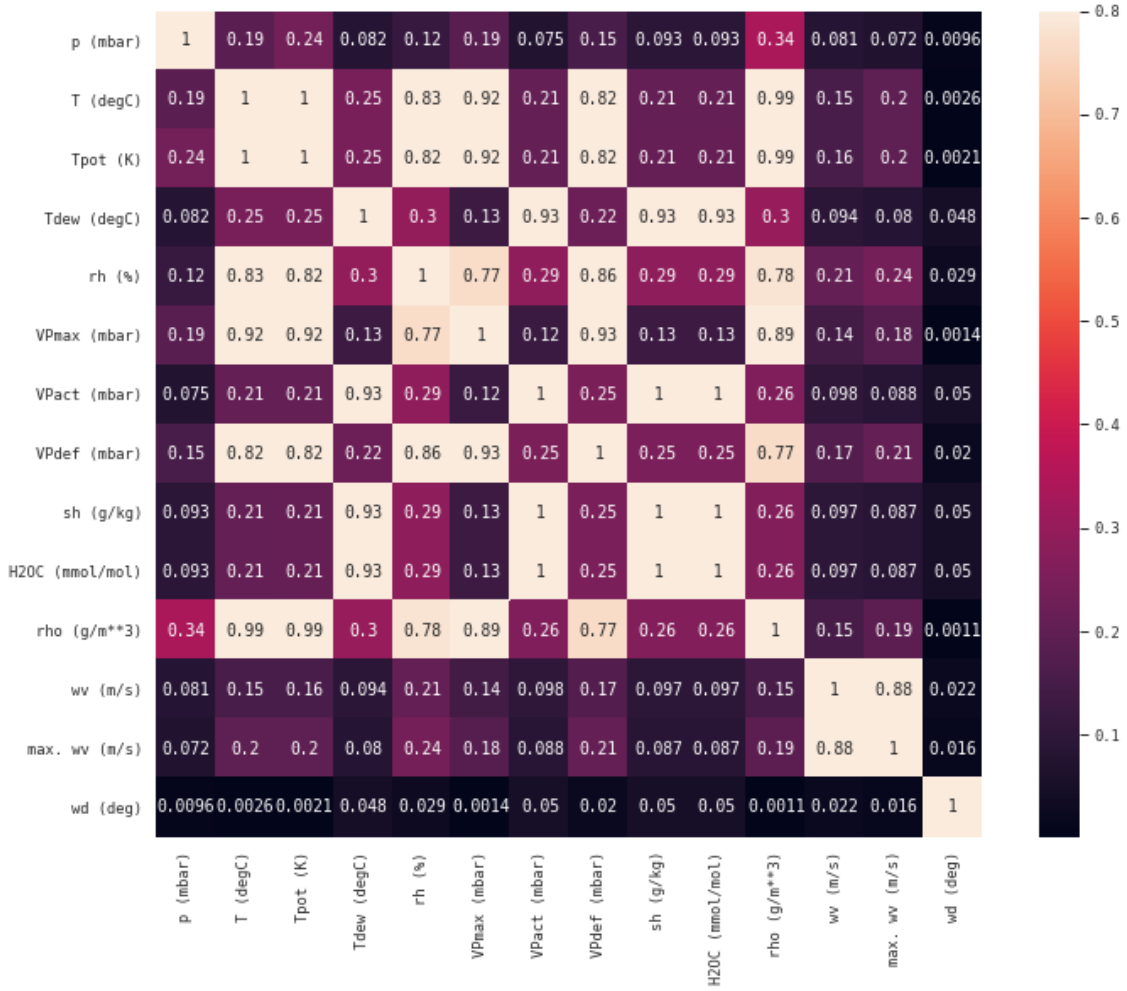

Figure 2.3: Pearson Correlation Matrix of all features.

2.1.3 Sliding Window

Sliding Window is built to generate the feature and label based on given sequence length k, *Feature columns*, and *label columns*. $k \in [4, 8]$ as requirement stated, $k = 4$ in chosen in our setup but it could be changed as a variable. *Feature columns* is chosen with pearson cross correlation according to section 2.1.2, and *label columns* is chosen with [p (mbar), T (degC), rh (%), wv (m/s)] according the requirement. Feature element take i to i+k attribute in *Feature columns* and transfer to tensor to get Feature. Similarly, label element take i to i+k attribute in *Label columns* and transfer to tensor to get Label. Afterwards, feature are label are fed into Dataloader to create a sample that reads small batches of data for models to utilize.
Note that size of feature is (batch size, sequence length, # of feature columns) and size of label is (batch size, # of label columns)

2.2 Models

All of these three models are follow the following setup unless stated otherwise:

- Number of epochs: 10
- Number of layers: 3
- batch size: 32
- k (sequence length) : 4
- # of feature columns: 8 (['VPmax (mbar)', 'rho (g/m**3)', 'p (mbar)', 'T (degC)', 'wv (m/s)', 'VPdef (mbar)', 'max. wv (m/s)', 'rh (%)'])
- # of label columns: 4 (['p (mbar)', 'T (degC)', 'rh (%)', 'wv (m/s)'])
- Start learning rate: 1e-4
- Learning rate scheduler: Step learning rate with gamma =0.5 and step size=3
- Loss criterion: Mean square error (MSE) loss function
- Optimizer: Adam optimizer
- Activation function in the hidden layer (for Baseline 1 and our model): ReLU

2.2.1 Baseline I: 3 layer dense NNs

The structure of 3 layer fully connected NNs is shown in table 2.1. It's noteworthy that in last layer of NNs is without activation function.

Table 2.1: 3 layer dense NN

Layer Size	**Activation Function**
(# of feature columns = 8 , 50)	ReLU
(50, 25)	ReLU
(25, # of label columns = 4)	None

It is expected that the size of the output of this model would be (batch size, sequence length, label columns), which is (32,4,4). However, the size of label is (batch size, # of label columns), which is (32,4). Because previous 4 (= sequence length) data is used to predict only next one data, only the last element in second dimension of output is selected and squeezed to match the size of label.

2.2.2 Baseline II: RNNs (LSTM)

The structure of baseline II is shown in table 2.2. This model consisted of two layer LSTM and 1 layer Fully connected NNs (FCNN).

Table 2.2: RNN LSTM

Layer Size	**NNs Type**
(input size = # of feature columns = 8 hidden size = 128 num layers=2, batch first=True)	LSTM
(128, # of label columns = 4)	FCNN

Similarly to baseline I, only the last element in second dimension of output is selected and squeezed to match the size of label.

2.2.3 Our model: CNNs

The structure of our model is shown in table 2.3. Our model consisted of two layer 1D CNN and 1 layer FCNN.

Table 2.3: Our model

Layer Size	**NNs Type**	**Activation Function**
(# of feature columns = 8, 50)	1D Conv with kernel = 3, padding = 1	ReLU
(50, 25)	1D Conv with kernel = 3, padding = 1	ReLU
(25, # of label columns = 4)	FCNN	None

We selected the same layer size with baseline I. (i.e. 8-50-25-4). The difference is that 1D convolution layer with kernel =3 with padding =1 is used instead. The reason choosing kernel =3 with padding =1 is to implement the same convolution to make the size of the output doesn't shrink. As figure 2.4 shown, the total size of our model is similar to baseline I, and both are much smaller than LSTM. With smaller parameters, our model can still achieve as low MSE as LSTM does. (MSE comparisons are show in the chapter 3.

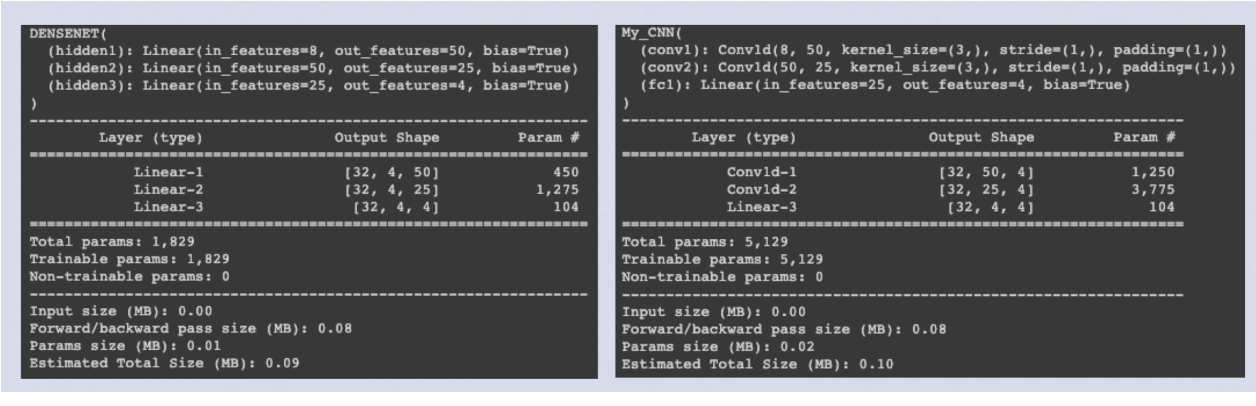

Figure 2.4: Parameter Comparison: Baseline model I (left) vs Our model (right)

Similarly to baseline I and II, only the last element in the second dimension of output is selected and squeezed to match the size of the label.

2.3 Training procedure

Each model follows the same training procedure to be evaluated and compared. The steps are as follows.

2.3.1 Step 1: Prepare Data

Data is pre-processed by the method mentioned in 2.1. Afterward, pre-processed and selected data (i.e. features are labels) are fed into Dataloader to create a sample that reads small batches of data for models to utilize.

2.3.2 Step 2: Training function

Training function is defined to be called for all three models. The training function will take the train feature as input and compare it with the label. The loss would be propagated backward to minimize the MSE loss as the provided code is shown.

2.3.3 Step 3: Testing function

Similarly to the training function, the testing function is defined to be called for all three models. It compares the testing feature and output prediction. MSE loss between prediction and label would be recorded for evaluating the model.

3. *Results*

3.1 Train and Test Loss

Measure of the mean squared error (MSE) is used for comparison.

Table 3.1: Train loss

Epoch	**Dense NN Loss**	**LSTM Loss**	**Our Model Loss**
1	0.60149	0.37105	0.44411
2	0.18535	0.14770	0.12749
3	0.12335	0.12798	0.11052
4	0.11794	0.12162	0.10694
5	0.11658	0.11889	0.10563
6	0.11577	0.11747	0.10475
7	0.11526	0.11598	0.10411
8	0.11499	0.11545	0.10371
9	0.11479	0.11495	0.10342
10	0.11461	0.11410	0.10318

Table 3.2: Test loss

Epoch	**Dense NN Loss**	**LSTM Loss**	**Our Model Loss**
1	0.27593	0.15146	0.15140
2	0.12832	0.11553	0.10921
3	0.11376	0.10722	0.10421
4	0.11109	0.10231	0.10122
5	0.11010	0.10128	0.10057
6	0.10947	0.10074	0.10002
7	0.10876	0.09805	0.09832
8	0.10855	0.09802	0.09813
9	0.10836	0.09778	0.09793
10	0.10836	0.09678	0.09720

We see that all three models converge to approximately the same test and training loss by the 10th Epoch.

3.2 Mean Square Error Prediction

Measure of the mean squared error (MSE) prediction of the four weather attributes and overall four attributes. We see that each of the squared error plots are similar across all three models.

3.2.1 Total MSE

The total mean square error is calculated by taking the mean of the square error across all time points. The overall MSE metric is the average MSE across the four weather attributes for each of the three models.

Table 3.3: Total MSE over predicted current time points

Weather Attribute	Dense NN MSE	LSTM MSE	Our Model MSE
p (mbar)	0.4075	0.7223	0.4643
T (degC)	1.0940	1.0192	0.8678
rh (%)	20.1131	14.6931	15.3853
wv (m/s)	0.8026	0.7336	0.7452
Overall MSE	5.6043	4.2921	4.1179

We see that our model had the best overall MSE. But when looking at rh (%) which has the highest MSE across all models, the LSTM was able to keep the rh (%) lowest. This is important to note because rh (%) is an outlier with high MSE so reducing the MSE of it could be beneficial to performance. In summary, our model performed best overall but the LSTM maintained the lowest maximum MSE.

3.2.2 Squared Error Plots

The squared error is calculated and shown in figure 3.1, figure 3.2, figure 3.3, figure 3.4 for each time series prediction for the four attributes.

Figure 3.1: p (mbar) Square Error Plot

Figure 3.2: T (degC) Squared Error Plot

Figure 3.3: rh (%) Squared Error Plot

Figure 3.4: wv (m/s) Squared Error Plot

By observing the square error across the full time prediction interval, we see that across all three models, the occurrence of the peaks in square error appear in the same place across all models. This is important to not because this shows that all three models behave relatively the same with only the height of the peaks being different. In summary, the location of large errors is the same throughout all models.

3.3 Error Propagation Curves

The error propagation curves are calculated and shown in table 3.4 and figure 3.5 as the MSE over the next k (=4) times series points using previous data points.
By measuring the total MSE across the error propagation, we see that the Dense NN has the best MSE over predicting multiple time series points.

Table 3.4: Total MSE over predicted current time points

Model	**Error Propagation MSE**
Dense NN	0.0397
LSTM	0.0644
Our Model	0.0517

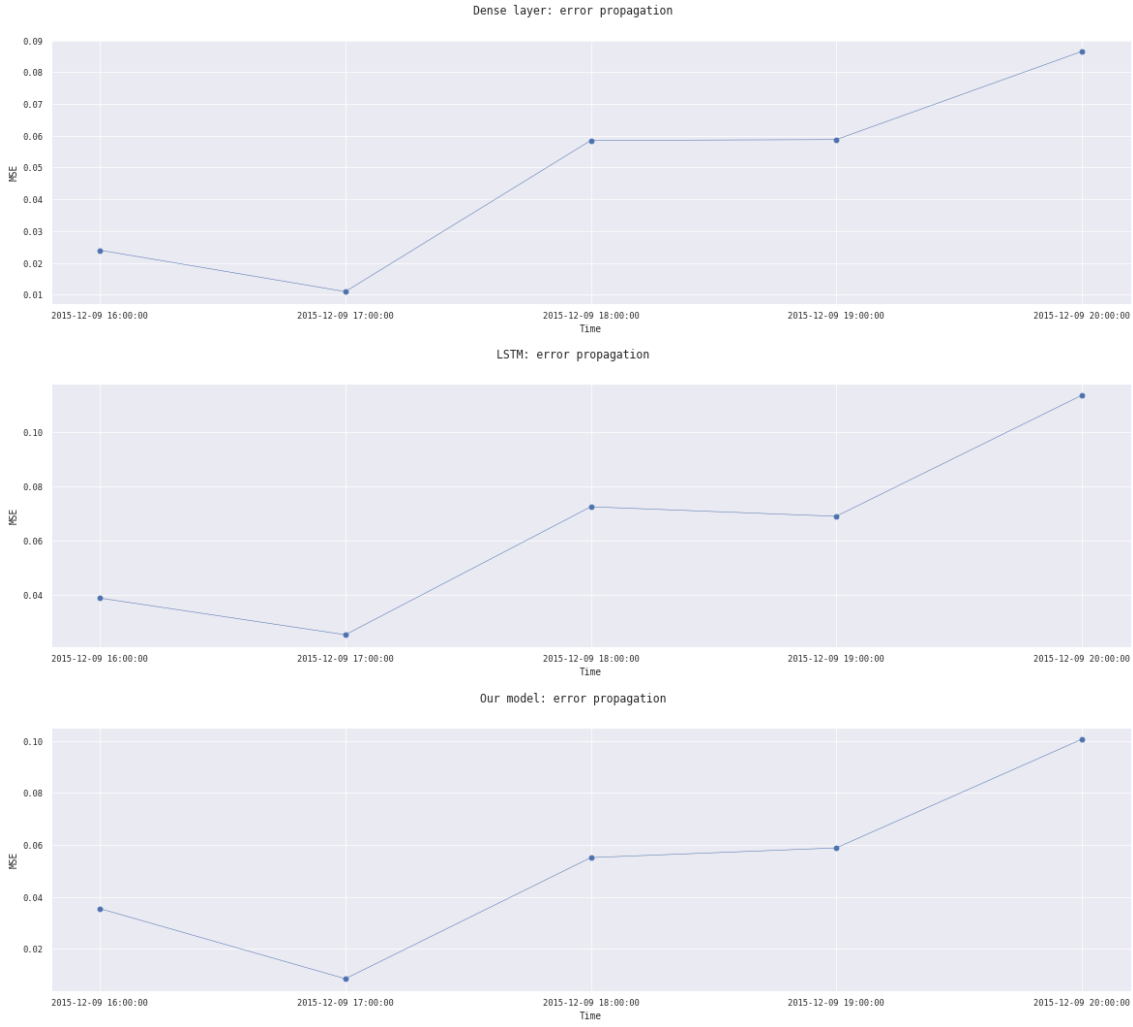

Figure 3.5: Error Propagation Curves

4. *Conclusion*

In conclusion, three models were evaulated using the same training procedure on time series data. In terms of overall prediction all three models performed relatively similar and were able to perform accurate predictions when observed across all time points. What differentiates the three models is the amount of error that occurs for difficult prediction time points. Out of the three models our CNN based model had the best overall MSE performance when predicting a single time step ahead. The dense NN had the best MSE performance when predicting for 5 future time series steps. The LSTM had the lowest maximum MSE across all weather attributes being predicted.