Predicting Temperature based on recent weather history using Time Series Forecasting.

Saurabh Jain

Viterbi School of Engineering, University of Southern California,

Los Angeles, California 90089, USA

(Dated: April 30, 2021)

Abstract

Making predictions about the future is called extrapolation in the classical statistical handling of time series data. More modern fields focus on the topic and refer to it as time series forecasting. Forecasting involves taking models fit on historical data and using them to predict future observations. Descriptive models can borrow for the future (i.e. to smooth or remove noise), they only seek to best describe the data. The primary task of this assignment is to predict temperature based on recent weather history using time series forecasting.

Various models were used after creating baseline models, that exploit the seasonality of the temperature. Multi-step models were primarily computed, such as: Simple Linear Model, Dense Layer Model, Convolutional Neural Network Model, Recurrent Neural Network Model (LSTM), Advanced Autoregressive Recurrent Neural Network Model (LSTM).

From all the models that were run, Advanced Autoregressive Recurrent Neural Network Model (LSTM) was the best performing model when predicting temperatures using all the features available in the data set. Recurrent Neural Network Model (LSTM) was a better performing model when predicting temperature using just the temperature feature of the data set.

I. INTRODUCTION

The primary task of this assignment is to predict temperature based on recent weather history using time series forecasting. Various tasks were run and all tasks included various models as well. The tasks included:

1. Predicting 24 hours of temperature using the past 24 hours, with all features.
2. Predicting 24 hours of temperature using the past 24 hours, with only temperature.
3. Predicting 24 hours of temperature using the past 24 hours, with just temperature, humidity, and pressure.
4. Predicting 24 hours of temperature using the past 24 hours, with just weather, wind speed and wind direction.
5. Predicting 24 hours of temperature using the past 9 days, with all features.
6. Predicting 24 hours of temperature using the past 9 days, with only temperature.
7. Predicting 24 hours of temperature using the past 10 days, with all features.
8. Predicting 24 hours of temperature using the past 10 days, with only temperature.

II. DATA EXPLORATION

The data used was an hourly compilation of 5 years of Los Angeles' weather data that composed of date & time, temperature, pressure, humidity, weather, wind direction and wind speed. Date & Time were in the format of yyyy-mm-dd hh:mm:ss. Weather column 25 different distinctions which were grouped, as described in the data pre-processing section:

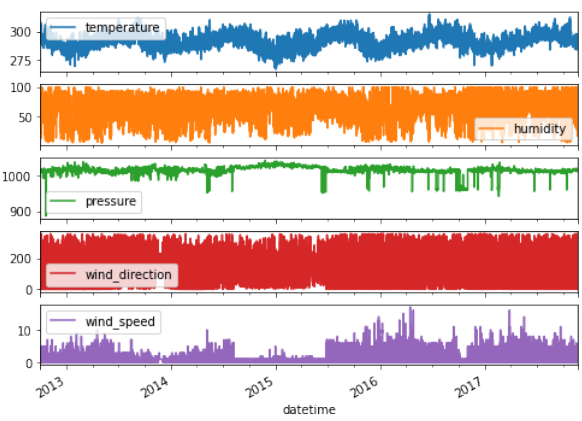
There consisted two files of data, one of them being ps6_trainvalid.csv, which was used to for training and validation purposes and the other one being ps6_test.csv, which was used to test the same models to check their mean absolute score.

III. DATA PREPROCESSING

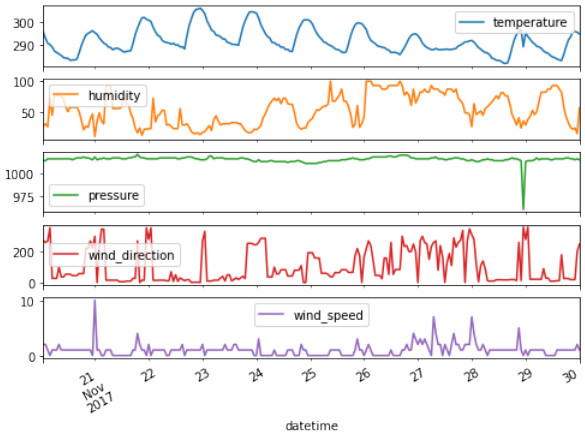
The very first data point of the training/validation data set did not contain any value for the features namely, temperature, humidity, pressure, weather, wind_direction, and

wind_speed. Therefore, the first step was to remove that row. Secondly, missing data points were interpolated using the linear method. temperature, humidity and pressure had 2, 151, and 251 missing values respectively which were interpolated. Next, the 25 weather types were grouped into 6 different categories:

1. Clear:
    a) sky is clear
2. Fog
    a) haze b) mist c) fog
3. Clouds
    a) broken clouds b) overcast clouds c) scattered clouds
    d) few clouds
4. Smoke/Dust
    a) smoke b) dust
5. Rain
    a) light rain b) moderate rain c) heavy intensity rain
    d) light intensity drizzle e) very heavy rain f) shower rain
    g) drizzle h) proximity shower rain i) light intensity shower rain
    j) squalls
6. Thunderstorms
    a) proximity thunderstorm b) thunderstorm c) thunderstorm with light rain
    d) thunderstorm with rain e) thunderstorm with heavy rain



Train/Validation plots



Test plots

One hot encoding was done to the weather column so that the categorical variable can be processed to incorporate this into the model.

Next, the entire data was plotted to check the general visualized trend and training/validation & test dataset can be visualized in the above figure. Wind speed and wind direction was combined to have modified columns as Wx and Wy, which signified the X and Y projection of the wind. The last column of the data, wind_direction, gives the wind direction in units of degrees. 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 velocity columns to a wind vector.

Similarly the Date Time column is very useful, but not in this string form. Start by converting it to seconds. Similar to the wind direction the time in seconds is not a useful model input. A simple approach to convert it to a usable signal is to use sin and cos to convert the time to clear "Time of day" and "Time of year" signals

Next, the data was normalized by subtracting the mean and dividing by the standard deviation of each feature.

IV. MODEL SELECTION

The models will make a set of predictions based on a window of consecutive samples from the data. The main features of the input windows are:

- The width (number of time steps) of the input and label windows

- The time offset between them.
- Which features are used as inputs, labels, or both.

A variety of models (including Linear, DNN, CNN and RNN models) are built, and uses them for multi-time-step predictions. Data Windowing is implemented so that it can be reused for all of those models.

Data windows are generated as follows:

1. Predicting 24 hours of temperature using the past 24 hours, with all features.
2. Predicting 24 hours of temperature using the past 24 hours, with only temperature.
3. Predicting 24 hours of temperature using the past 24 hours, with just temperature, humidity, and pressure.
4. Predicting 24 hours of temperature using the past 24 hours, with just weather, wind speed and wind direction.
5. Predicting 24 hours of temperature using the past 9 days, with all features.
6. Predicting 24 hours of temperature using the past 9 days, with only temperature.
7. Predicting 24 hours of temperature using the past 10 days, with all features. [1]
8. Predicting 24 hours of temperature using the past 10 days, with only temperature. [2]

The following steps were conducted before any model could be computed:

1. Indexes and offsets: By creating the WindowGenerator class, the __init__ method includes all the necessary logic for the input and label indices. It also takes the train, validation, and test dataframes as input. These are converted to tf.data.Datasets of windows later.
2. Split: Given a list consecutive inputs, the split_window method converts them to a window of inputs and a window of labels.

[1] Some parts of train-valid data had to be moved test because test data had exactly 10 days worth of data, which is why 9 days of data was computed to keep the test set consistent.

[2] Some parts of train-valid data had to be moved test because test data had exactly 10 days worth of data.

3. Plot: Plotting a window to visualize the 24 hours of data as seen in fig. 1a

4. Create tf.data.Datasets: Finally, make_dataset method takes a time series DataFrame and convert it to a tf.data.Dataset of (input_window, label_window) pairs using the preprocessing.timeseries_dataset_from_array function.

V. MODEL EVALUATION

The following models were computed for various windows that have been described above. Predicting 24 hours of temperature using the past 24 hours, with all features will be described and the results will be compared to other windows.

1. Multi-Step Last Baseline: A simple baseline for this task is to repeat the last input time step for the required number of output time-steps. This can be visualized in Fig. 1b

2. Repeat Baseline: This is another simple approach to repeat the previous day, assuming tomorrow will be similar as visualized in Fig. 1c

3. Simple Linear Model: A simple linear model based on the last input time step does better than baseline. The model will predict OUTPUT_STEPS time steps, from a single input time step with a linear projection. It can only capture a low-dimensional slice of the behavior, likely based mainly on the time of day and time of year. This can be visualized in Fig. 1d

4. Dense Neural Network: Adding a layers.Dense between the input and output gives the linear model more power, but is still only based on a single input timestep. This can be visualized in Fig. 1e

5. Convolutional Neural Network: A convolutional model makes predictions based on a fixed-width history, which leads to better performance than the dense model since it can see how things are changing over time. A convolutional model sees how things change over time. This can be visualized in Fig. 1f

6. Recurrent Neural Network (LSTM): A recurrent model learns to use a long history of inputs, if it's relevant to the predictions the model is making. Here the model

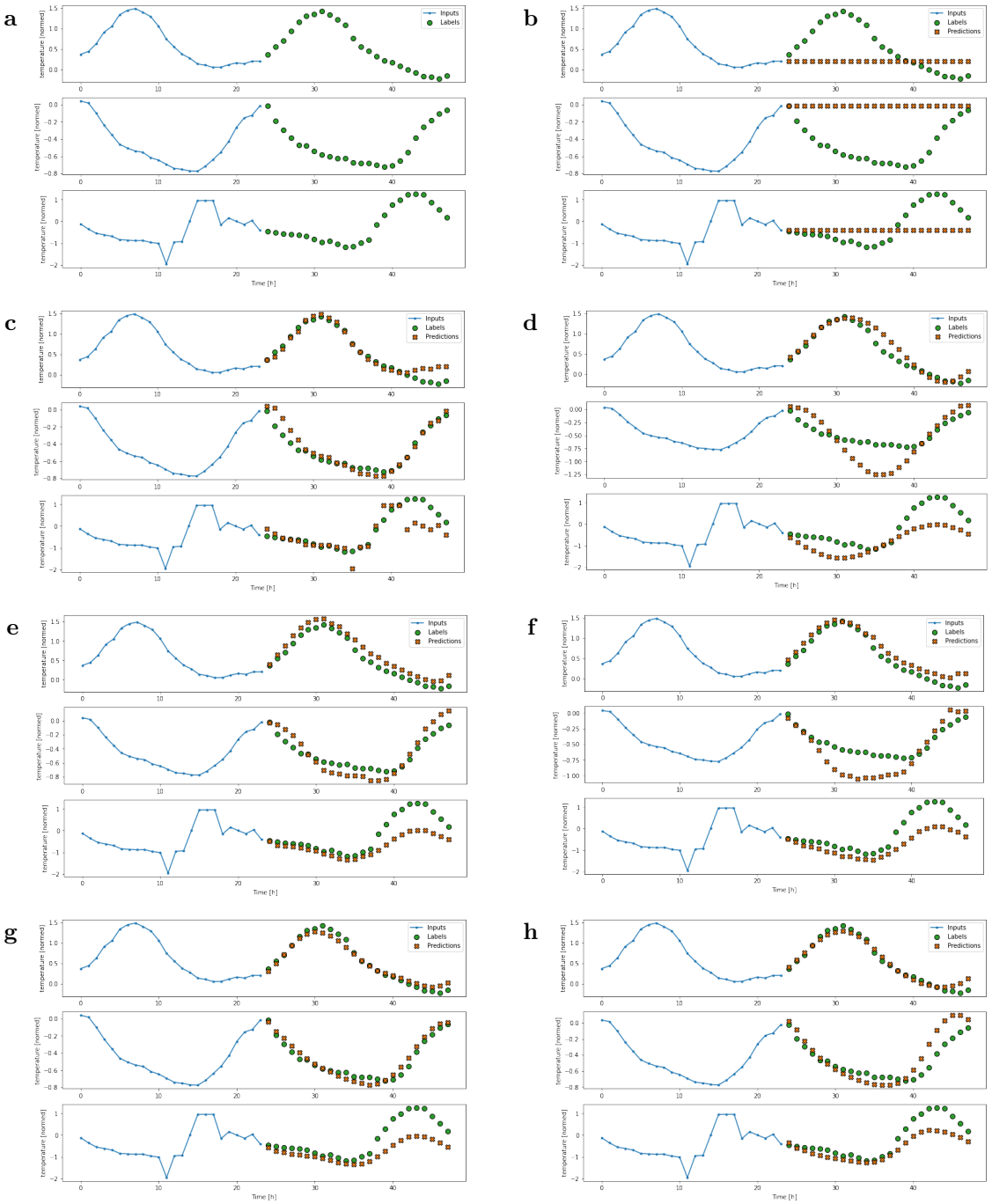


FIG. 1: Various models' visualizations predicting 24 hours of temperature from 24 hours of window.

accumulates internal state for 24h, before making a single prediction for the next 24h. This can be visualized in Fig. 1g

7. Advanced: Autoregressive Recurrent Neural Network Model (LSTM): The model has the same basic form as the single-step LSTM models: An LSTM followed by a lay-

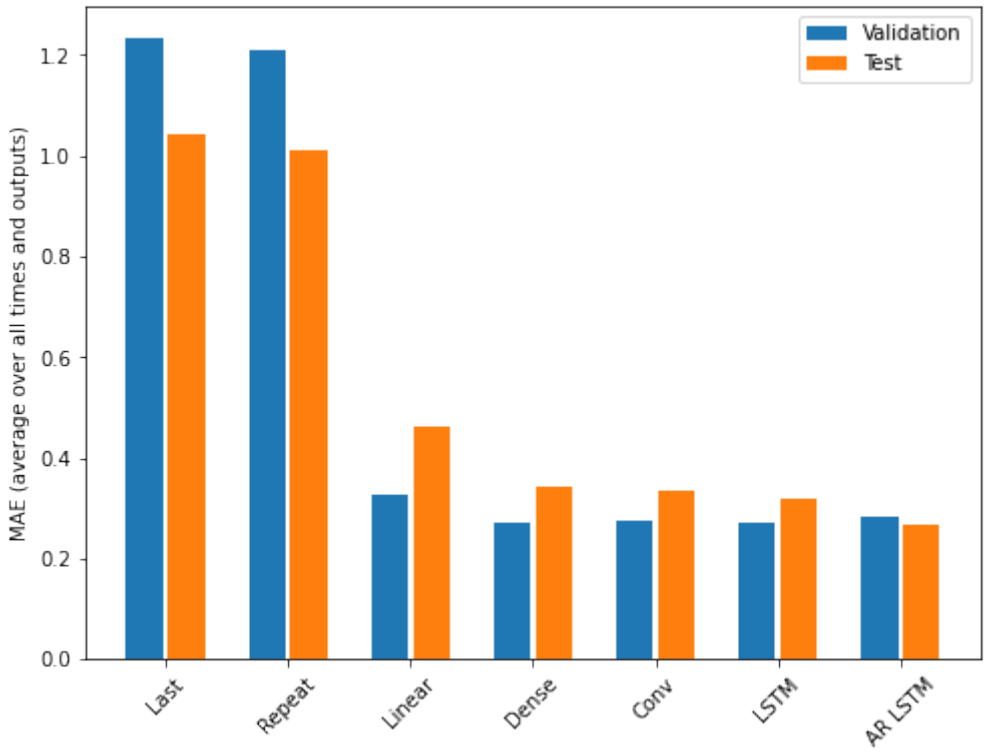


FIG. 2: Mean Absolute Error for predicting 24 hours using 24 hour window

ers.Dense that converts the LSTM outputs to model predictions. A layers.LSTM is a layers.LSTMCell wrapped in the higher level layers.RNN that manages the state and sequence results. In this case the model has to manually manage the inputs for each step so it uses layers.LSTMCell directly for the lower level, single time step interface. This can be visualized in Fig. 1h

The results on the test set are as follows, which is the mean absolute error for various models are visualized in the following figure 2.

- Last Baseline: 1.0446
- Repeat Baseline: 1.0121
- Simple Linear : 0.4625
- Dense Neural Network: 0.3431
- Convolutional Neural Network : 0.3358
- LSTM RNN: 0.3192

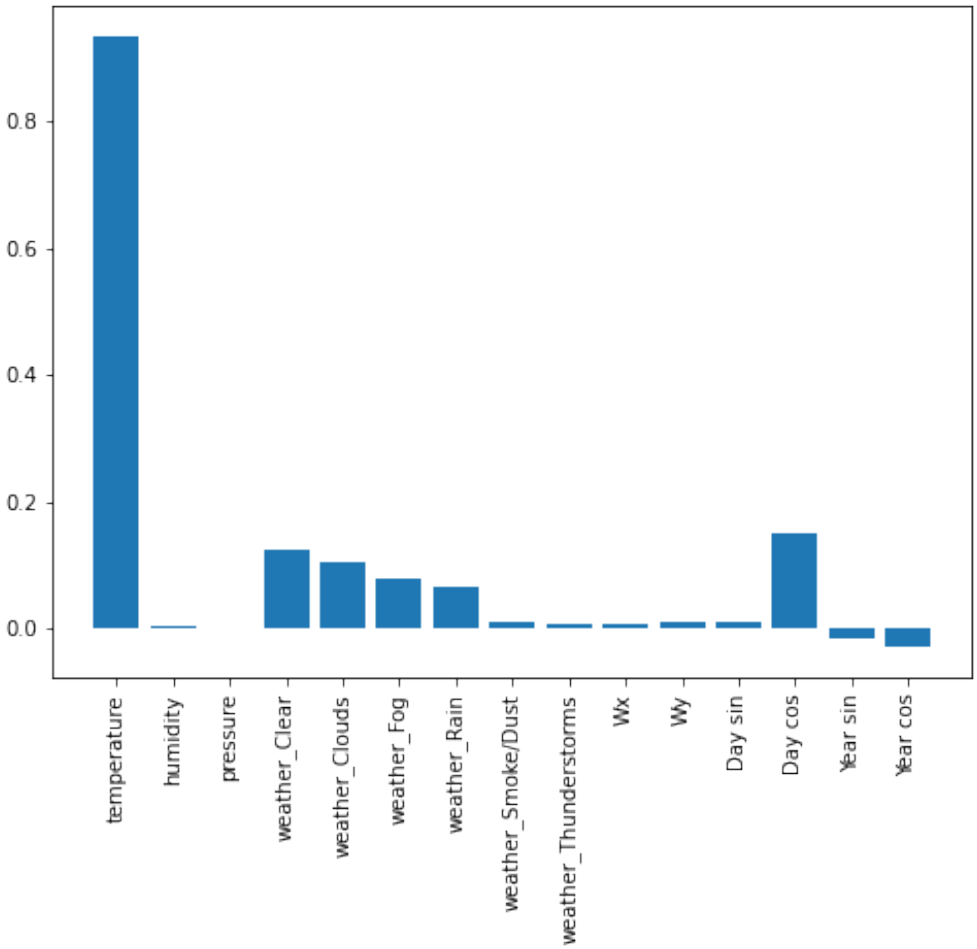


FIG. 3: Feature Importance

- AR LSTM RNN: 0.2679

The results above concludes that the Advanced: Autoregressive Recurrent Neural Network Model (LSTM) works best for predicting 24 hours of temperature. The gains achieved going from a dense model to convolutional and recurrent models are only a few percent, and the autoregressive model performed clearly best. So these more complex approaches are worth while on this assignment.

VI. FEATURE IMPORTANCE

Feature importance was calculated using the single step model and temperature is the best feature and quite important to predict next day's temperature. The next best was the day and the fact that the weather was clear or not. This can be visualized in Fig. 3

All these models were run with just temperature by dropping other features from the data set. These models were also run, once with just temperature, humidity, pressure and another time with temperature, weather, wind direction, wind speed. The visualization of the data will be displayed in the next section along with the mean absolute error values.

VII. INTERPRETATION

Three figures are described as follows:

1. Figure 4: LSTM model visualizations of various windows predicting 24 hours of temperature.

2. Figure 5: AR LSTM model visualizations of various windows predicting 24 hours of temperature.

3. Figure 6: Mean Absolute Error Bar Plots of various windows predicting 24 hours of temperature.

Each figure below has the following references:

a) Predicting 24 hours of temperature using the past 24 hours, with all features. Fig. 4a, 5a, 6a.

b) Predicting 24 hours of temperature using the past 24 hours, with only temperature. Fig. 4b, 5b, 6b.

c) Predicting 24 hours of temperature using the past 24 hours, with just temperature, humidity, and pressure.Fig. 4c, 5c, 6c.

d) Predicting 24 hours of temperature using the past 24 hours, with just weather, wind speed and wind direction. Fig. 4d, 5d, 6d.

e) Predicting 24 hours of temperature using the past 9 days, with all features. Fig. 4e, 5e, 6e.

f) Predicting 24 hours of temperature using the past 9 days, with only temperature. Fig. 4f, 5f, 6f.

g) Predicting 24 hours of temperature using the past 10 days, with all features. Fig. 4g, 5g, 6g.

h) Predicting 24 hours of temperature using the past 10 days, with only temperature. Fig. 4h, 5h, 6h.

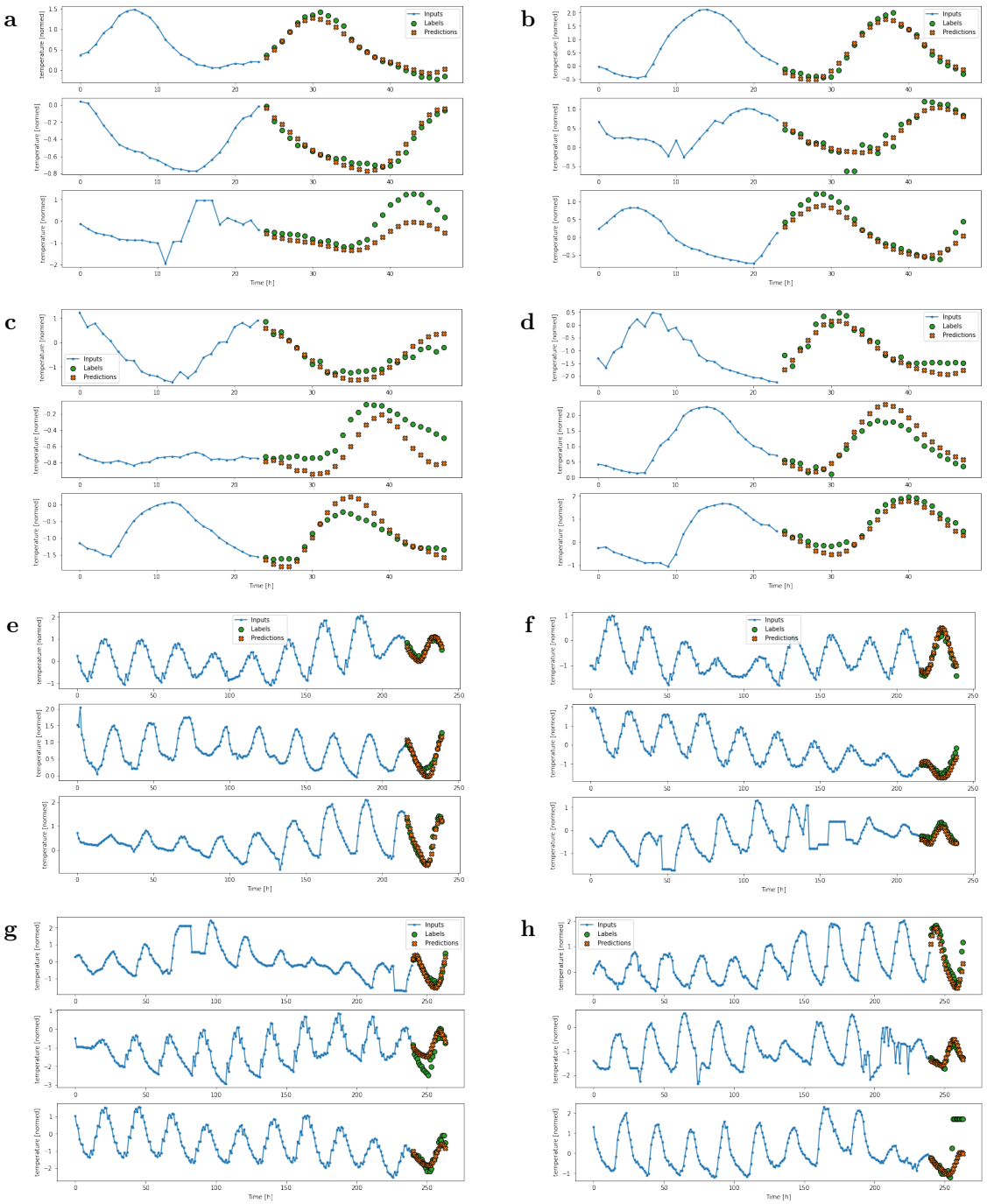


FIG. 4: LSTM model visualizations of various windows predicting 24 hours of temperature.

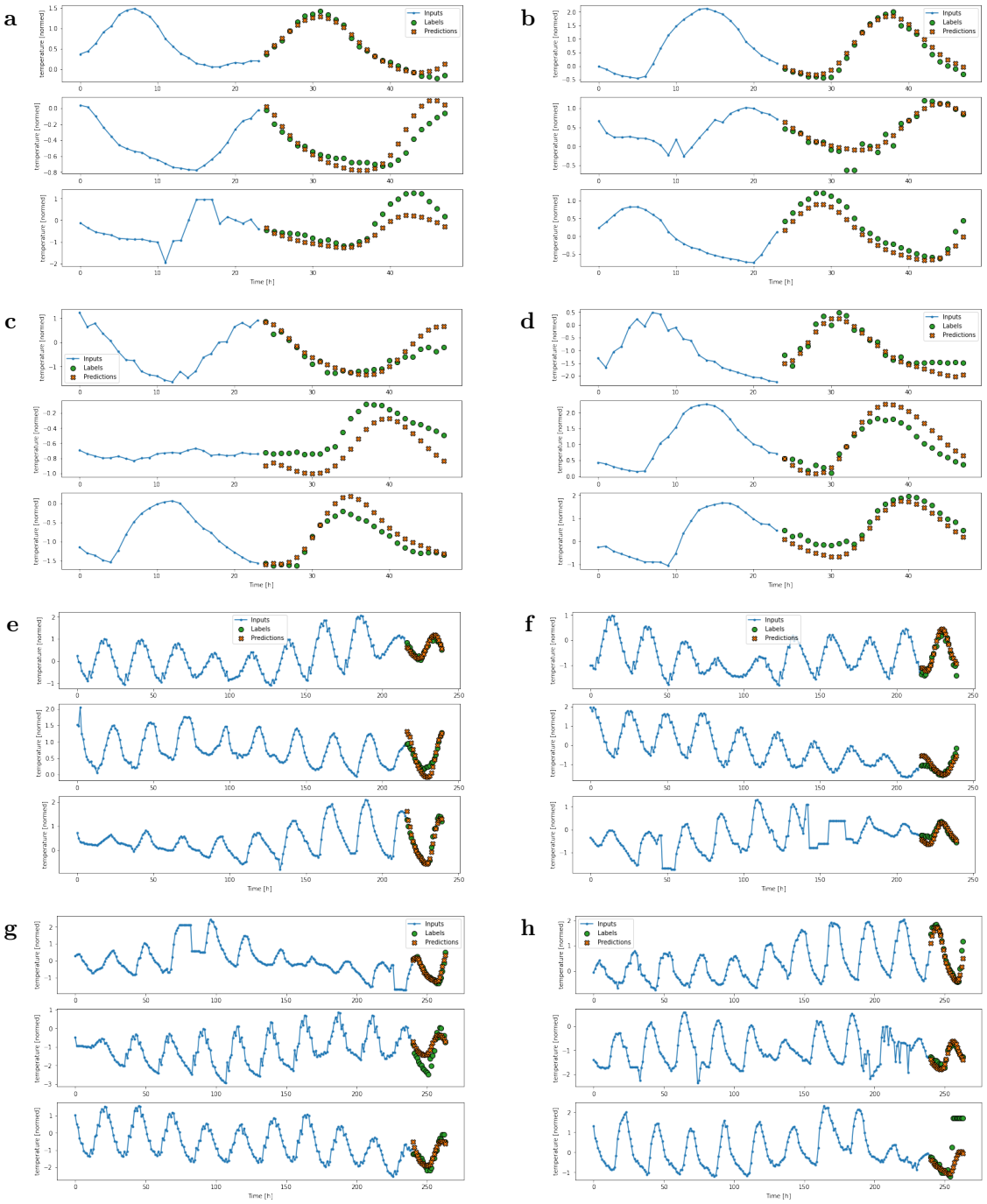


FIG. 5: AR LSTM model visualizations of various windows predicting 24 hours of temperature.

It can be inferred from the MAE (Mean Absolute Error) Bar plots that the Advanced: Autoregressive Recurrent Neural Network Model (LSTM) and the RNN (LSTM) Model work best for predicting 24 hours of temperature. The gains achieved going from a dense model to convolutional and recurrent models are only a few percent, and the autoregressive

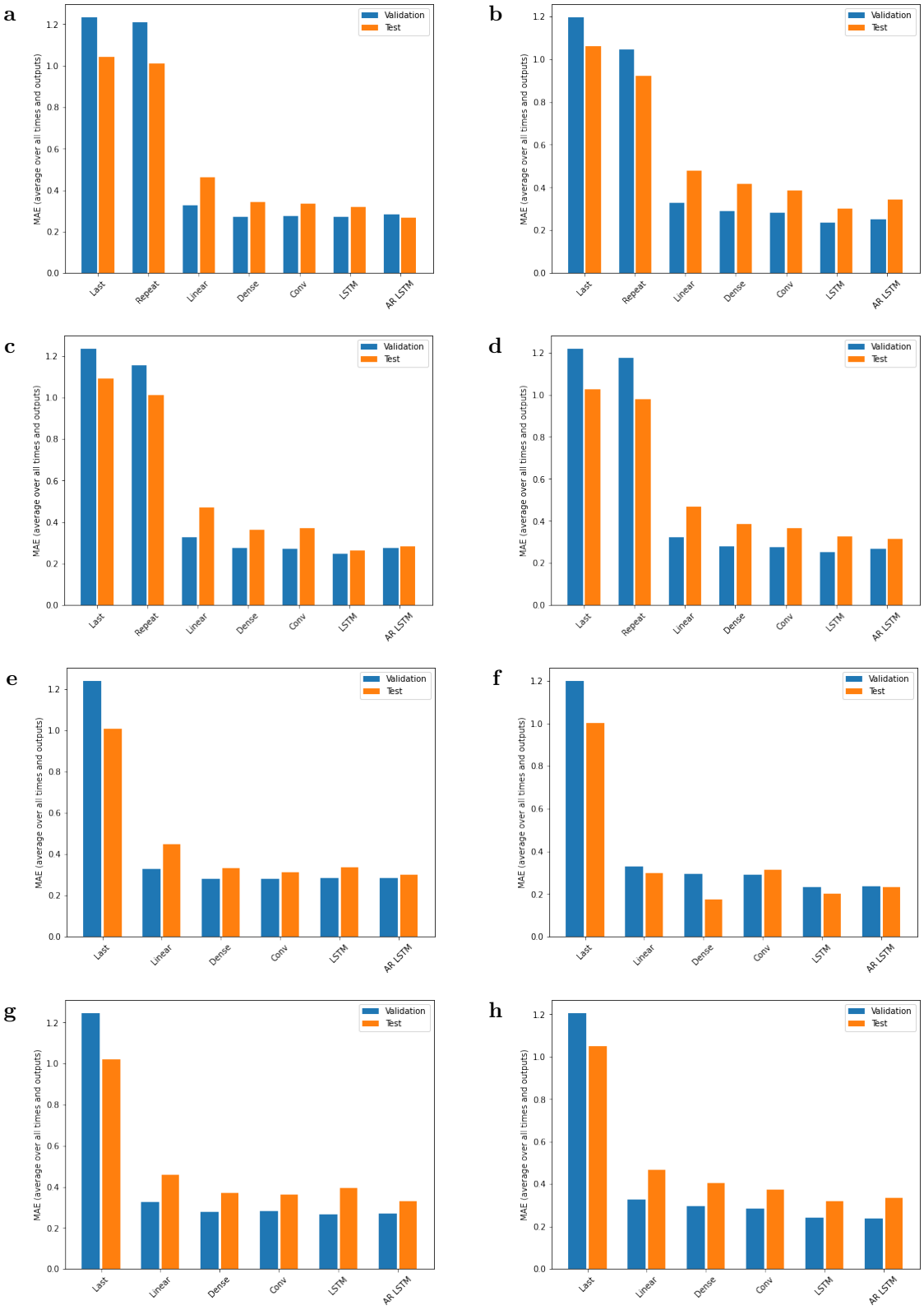


FIG. 6: Mean Absolute Error Bar Plots of various windows predicting 24 hours of temperature.

Name	a	b	c	d	e	f	g	h
Last Baseline	1.0446	1.0623	1.0933	1.0252	1.0049	1.0024	1.0198	1.0482
Repeat Baseline	1.0121	0.9221	1.0128	0.9771	-	-	-	-
Linear	0.4625	0.4802	0.4698	0.4695	0.4497	0.2989	0.4589	0.4677
Dense	0.3431	0.4176	0.3623	0.3837	0.3317	0.1746	0.3690	0.4030
CNN	0.3358	0.3876	0.3708	0.3648	0.3138	0.3130	0.3604	0.3755
LSTM	0.3192	0.3025	0.2616	0.3268	0.3367	0.2005	0.3937	0.3203
AR LSTM	0.2679	0.3436	0.2826	0.3148	0.2997	0.2340	0.3310	0.3329

TABLE I: MAE(Mean Absolute Error of different models with various windows

model performed clearly best. So these more complex approaches are worth while on this assignment. The Baseline Models are not the greatest at predicting for time series forecasting since there is seasonality as well.

It can also be computed that for the models which contained just the temperature, LSTM models were better than AR LSTM Model. This is interesting because AR LSTM creates additional layers and LSTM does not and therefore it provides better results.

The Mean Absolute Error values of each model are listed in Table I:

VIII. CONCLUSIONS

Various models were used after creating baseline models, that exploit the seasonality of the temperature. Multi-step models were primarily computed, such as: Simple Linear Model, Dense Layer Model, Convolutional Neural Network Model, Recurrent Neural Network Model (LSTM),Advanced Autoregressive Recurrent Neural Network Model (LSTM).From all the models that were run, Advanced Autoregressive Recurrent Neural Network Model(LSTM) was the best performing model when predicting temperatures using all the features available in the data set. Recurrent Neural Network Model (LSTM) was a better performing model when predicting temperature using just the temperature feature of the data set.

It can be concluded from the MAE (Mean Absolute Error) scores that the Advanced: Autoregressive Recurrent Neural Network Model (LSTM) and the RNN (LSTM) Model work best for predicting 24 hours of temperature. The gains achieved going from a dense model to convolutional and recurrent models are only a few percent, and the autoregressive model performed clearly best. So these more complex approaches are worth while on this

assignment. The Baseline Models are not the greatest at predicting for time series forecasting since there is seasonality as well.

DATA AVAILABILITY

Data is available at blackboard namely:

- ps6_trainvalid.csv
- ps6_test.csv

CODE AVAILABILITY

Code is available at `https://github.com/USC-DSCI-552-Spring2021/dsci552-spring2021-32416d-git`

- 24-24.ipynb
- 24-24-just-temp.ipynb
- 24-24-Temp-pres-humi.ipynb
- 24-24-weather-wd-ws.ipynb
- 216-24.ipynb
- 216-24-just-temp.ipynb
- 240-24.ipynb
- 240-24-just-temp.ipynb

ACKNOWLEDGMENTS

I would like to thank professor Kristina Lerman, professor Keith Burghardt, the TAs and the graders of the course DSCI 552 for all their efforts towards this assignment.

Appendix A: References

1. "Time Series Forecasting — TensorFlow Core." TensorFlow, 2020, https://www.tensorflow.org/tutorials/structured_data/time_series

2. "4.2. Permutation Feature Importance — Scikit-Learn 0.24.2 Documentation." Scikit, 2020, https://www.scikit-learn.org/stable/modules/permutation_importance.html.