CSE-523 Project Report

Time Series Prediction for Power Generation from Wind.

Guide: Prof. Zhenhua Liu

- Shashwat Arghode 111563937

Introduction

Wind speed/power has received increasing attention around the earth due to its renewable nature as well as environmental friendliness. With the global installed wind power capacity rapidly increasing, the wind industry is growing into a large-scale business. Reliable short-term wind speed forecasts play a practical and crucial role in wind energy conversion systems, such as the dynamic control of wind turbines and power system scheduling. A precise forecast needs to overcome problems of variable energy production caused by fluctuating weather conditions. Power generated by wind is highly dependent on the wind speed. Though it is highly non-linear, wind speed follows a certain pattern over a certain period of time. We exploit this time series pattern to gain useful information and use it for power prediction.

Learning Model Details

Now we know that wind energy prediction is a very interesting problem, let's look into why it is a difficult one. Accurate and reliable wind speed forecasts are a significant challenge due to its stochastic nature with high rates of change, highly nonlinear behavior with no typical patterns, and dependency on elevation, terrain, atmospheric pressure, and temperature, which results in large uncertainties of wind speeds. This makes it difficult for any machine learning model to figure out a pattern and give an accurate prediction. We made an effort to interpret this problem as time series forecasting problem as wind follows a particular pattern for a certain period like a day, month or year. Long Short-Term Memory (LSTM) machine learning model, which is best known for time series data prediction is used to learn these patterns in wind and make a prediction about power. We will look into the model and implementation details ahead. This prediction problem was divided into two categories:

1. **Estimation**: Given weather conditions like temperature, wind speed, pressure etc. determining the energy power prediction.
2. **Prediction**: Without knowing any details about the weather conditions predicting the power generation using the pattern which it has followed in a certain period of time.

Long Short Term Memory Machine Learning Model

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feed forward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

The expression long short-term refers to the fact that LSTM is a model for the short-term memory which can last for a long period of time. An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events. LSTMs were developed to deal with the exploding and vanishing gradient problem when training traditional RNNs. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods in numerous applications.

This just a introduction about LSTM, for more architectural and mathematical details you can read from the links provided in references.

Note:

Look Back/Lag is a common term used in LSTM which signifies the number of steps (apart from the pattern learned) an LSTM model will use to predict the next result.

In all the plots which are shown below. X-axis represents the hours for which we are predicting and Y-axis represents the power generated by the system.

Blue: True Power Generated **Orange: Predicted Power Generated**

Data for LSTM Experiments

Historical wind energy data is taken from NREL to do this analysis. 6 years of wind power generation data is used in this experiment. The data after pre-processing have details about timestamp, air temperature (C), pressure (atm), wind direction (deg), wind speed (m/s) and Power generated by the system (kW). We have hourly data for about 6 years.

Out[10]:

DateTime	Air temperature | ('C)	Pressure | (atm)	Wind speed | (m/s)	Wind direction | (deg)	Power generated by system | (kW)
2007-01-01 00:00:00	10.926	0.979103	9.014	229	33688.1
2007-01-01 01:00:00	9.919	0.979566	9.428	232	37261.9
2007-01-01 02:00:00	8.567	0.979937	8.700	236	30502.9
2007-01-01 03:00:00	7.877	0.980053	8.481	247	28419.2
2007-01-01 04:00:00	7.259	0.979867	8.383	256	27370.3

Data Snapshot

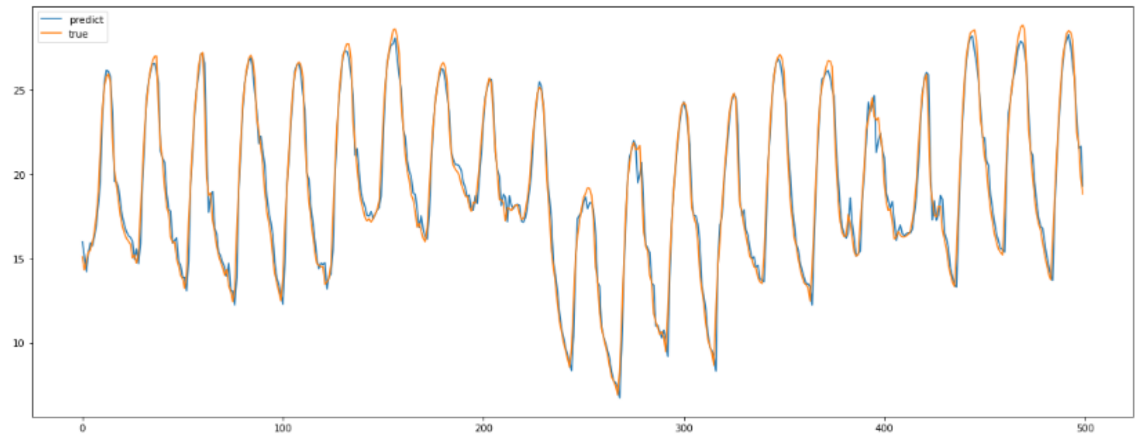
All the features are used for Estimation model and only time series features i.e. DateTime and power generated by the system are used for prediction experiments.

Estimation

Estimation is all about predicting wind power generation given the current wind direction. Current wind and temperature conditions are given, this makes this problem a bit easy for a model like LSTM which looks at the current state of the weather and the previous trend which the weather is following to predict the power generated by the system. Multivariate time series forecasting with LSTM with Keras library is used for this. After forming the baseline model and initial experimentations we found that 8 look backs is a very good number which provides significant results in the prediction. Estimations models are useful if we get the weather information about the present day or the future publically using machine learning with certain accuracy. Then this model can be used to be the perfect estimation of power generated by the system.

Experiment 1:

Six years hourly data was divided into 70-30 train test batch for this experiment. That means 4 years of data was used to predict 2 years of wind power generation. Good result with root mean square error(RMSE) 1.242 and Variance 0.984 was observed for this experiment.

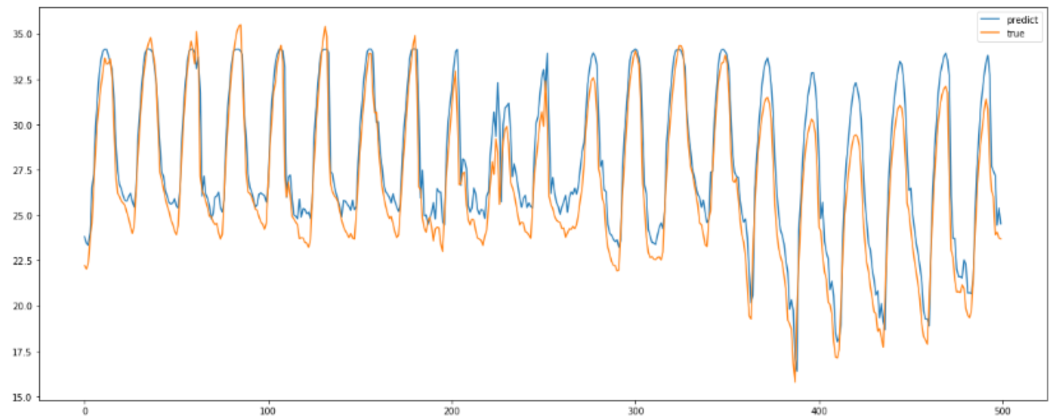



```
In [11]: print("Mean squared error: %.3f" % mean_squared_error(testY, yhat))

         Mean squared error: 1.242

In [12]: print("Root mean squared error: %.3f" % sqrt(mean_squared_error(testY, yhat)))

         Root mean squared error: 1.115

In [13]: from sklearn.metrics import mean_squared_error, r2_score
         print('Variance : %.3f' % r2_score(testY, yhat))

         Variance : 0.984
```

Code Link: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp1-Estimation-8%20look%20backs%2070-30%20split.ipynb

Experiment 2:

Six years hourly data was divided into 60-40 train test batch for this experiment. That means 3 years of data was used to predict 3 years of wind power generation. Good result with RMSE of 1.667 and Variance 0.969 was observed for this experiment.




```
In [11]: print("Mean squared error: %.3f" % mean_squared_error(testY, yhat))

Mean squared error: 2.778

In [12]: print("Root mean squared error: %.3f" % sqrt(mean_squared_error(testY, yhat)))

Root mean squared error: 1.667

In [13]: from sklearn.metrics import mean_squared_error, r2_score
         print('Variance : %.3f' % r2_score(testY, yhat))

Variance : 0.969
```

Code Link: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp2-Estimation-8%20look%20backs%2060-40%20split.ipynb

Prediction

Let's look into the pure time series analysis. In Prediction part, we are predicting the power generated by the system without any knowledge of the future weather. This is important because predicting the future weather is also a different prediction problem machine learning with its different set of challenges. We are not going to have any knowledge what wind speed is going to be or what temperature or pressure is going to be in the future. So, we try to predict the power only by analyzing pattern in the past data using LSTM. Data to this model will be Date time and Power generated by the system in the supervised form as required by the LSTM. LSTM will analyze the prior data and try to get useful knowledge about the patterns in previous data. And using that knowledge it is going to predict the results. Walk forward validation is used for predicting future values and evaluating results.

Data to LSTM for prediction:

	Power generated by system | (kW)
DateTime	
2007-01-01 00:00:00	33688.1
2007-01-01 01:00:00	37261.9
2007-01-01 02:00:00	30502.9
2007-01-01 03:00:00	28419.2
2007-01-01 04:00:00	27370.3

After forming the baseline LSTM model, we performed many different experiments to get the perfect look back and the neurons which are required by the LSTM. After getting the look back, neuron number and certain other parameters right for the model we performed a few experiments and predictions. The result of them as follows.

Experiment 3:

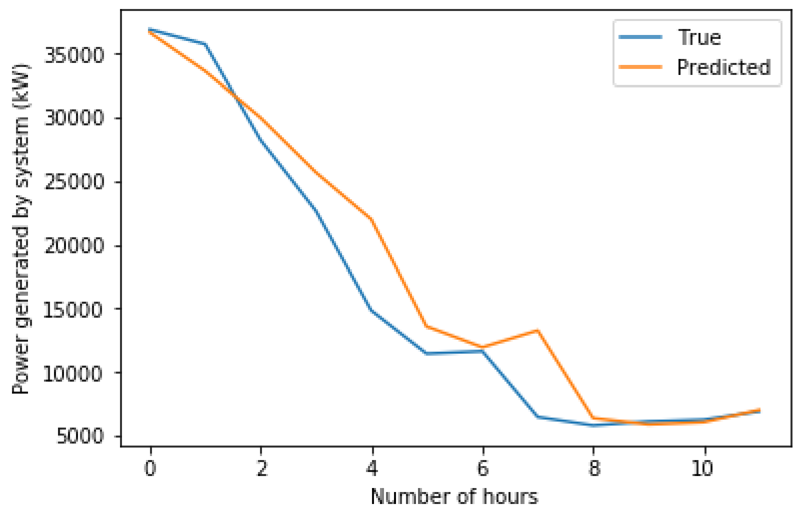
12 hours data was predicted with the same model configurations. And a Mean percent error of 18.28% is observed.

LSTM model configuration:

Input Batch Size : 1
Epochs : 7
No. of Neurons : 10
Look Backs/lag : 24

```
Hour=1, Predicted=36645.864606, Expected=36876.500000
Hour=2, Predicted=33615.890842, Expected=35723.600000
Hour=3, Predicted=29971.099147, Expected=28221.500000
Hour=4, Predicted=25673.883808, Expected=22650.000000
Hour=5, Predicted=21989.834303, Expected=14845.100000
Hour=6, Predicted=13579.051097, Expected=11449.700000
Hour=7, Predicted=11946.287910, Expected=11637.200000
Hour=8, Predicted=13268.918765, Expected=6465.350000
Hour=9, Predicted=6379.478074, Expected=5802.110000
Hour=10, Predicted=5876.449804, Expected=6110.570000
Hour=11, Predicted=6053.338795, Expected=6251.340000
Hour=12, Predicted=7015.175104, Expected=6899.170000
```

```
In [19]: # Evaluate performance
expectations = np.array(expectations)
predictions = np.array(predictions)
print("Mean Percent Error: ",(np.mean(np.abs((expectations - p

Mean Percent Error:  18.276662461837244
```



Code Link: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp3-Prediction-Batch%201-12%20hour.ipynb

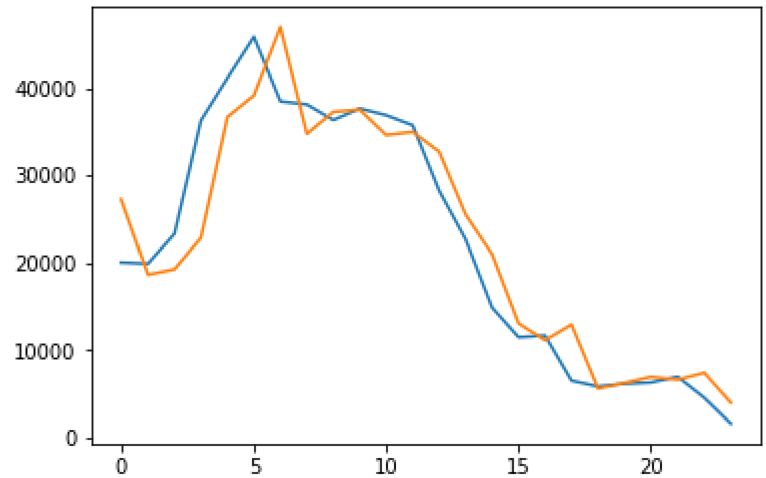
Experiment 4:

24 hours data was predicted and a Mean percentage error of 24.53% was observed. This is brilliant for a day ahead prediction and from the predicted value plot is seen that LSTM is able to find the pattern is a day's data and the same pattern which was in expected data is predicted.

```
Hour=1, Predicted=27243.064542, Expected=20005.800000
Hour=2, Predicted=18602.173972, Expected=19870.600000
Hour=3, Predicted=19226.017874, Expected=23296.700000
Hour=4, Predicted=22843.009441, Expected=36275.400000
Hour=5, Predicted=36677.630417, Expected=41119.700000
Hour=6, Predicted=39130.904941, Expected=45831.200000
Hour=7, Predicted=46951.271863, Expected=38451.600000
Hour=8, Predicted=34744.402930, Expected=38107.700000
Hour=9, Predicted=37258.460807, Expected=36325.100000
Hour=10, Predicted=37514.469822, Expected=37641.300000
Hour=11, Predicted=34596.531165, Expected=36876.500000
Hour=12, Predicted=34966.788134, Expected=35723.600000
Hour=13, Predicted=32700.244708, Expected=28221.500000
Hour=14, Predicted=25493.625228, Expected=22650.000000
Hour=15, Predicted=20947.633680, Expected=14845.100000
Hour=16, Predicted=13027.415709, Expected=11449.700000
Hour=17, Predicted=11098.948812, Expected=11637.200000
Hour=18, Predicted=12893.757226, Expected=6465.350000
Hour=19, Predicted=5563.589926, Expected=5802.110000
Hour=20, Predicted=6200.452970, Expected=6110.570000
Hour=21, Predicted=6890.973331, Expected=6251.340000
Hour=22, Predicted=6583.691803, Expected=6899.170000
Hour=23, Predicted=7362.150686, Expected=4514.490000
Hour=24, Predicted=3953.142474, Expected=1561.250000
```

```
n [40]: expectations = np.array(expectations)
predictions = np.array(predictions)
print("Mean Percent Error: ", (np.mean(np.abs((expectations - predictions) / expectations))*100))
```

```
Mean Percent Error:  24.527352319784743
```



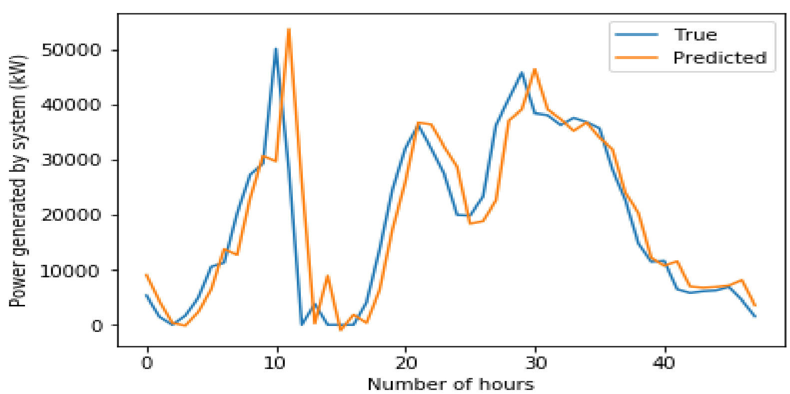
Code Link: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp4-Prediction-Batch%201-24%20hour.ipynb

Experiment 5:

48 hours of data was predicted and a Mean percentage error of 39.47% was observed. This was great for a prediction for wind power generation which is dependent on highly non-linear parameter as Wind Speed.

```
In [22]: expectations = np.array(expectations)
         predictions = np.array(predictions)
         print("Mean Percent Error: ", (np.mean(np.abs((expectations

         Mean Percent Error:  39.47723149683532
```

```
In [20]: # line plot of observed vs predicted
         pyplot.plot(raw_values[-predict_values_exp:], label="True")
         pyplot.plot(predictions, label="Predicted")
         pyplot.legend(loc='upper right')
         pyplot.xlabel("Number of hours")
         pyplot.ylabel("Power generated by system (kW)")
         pyplot.show()
```



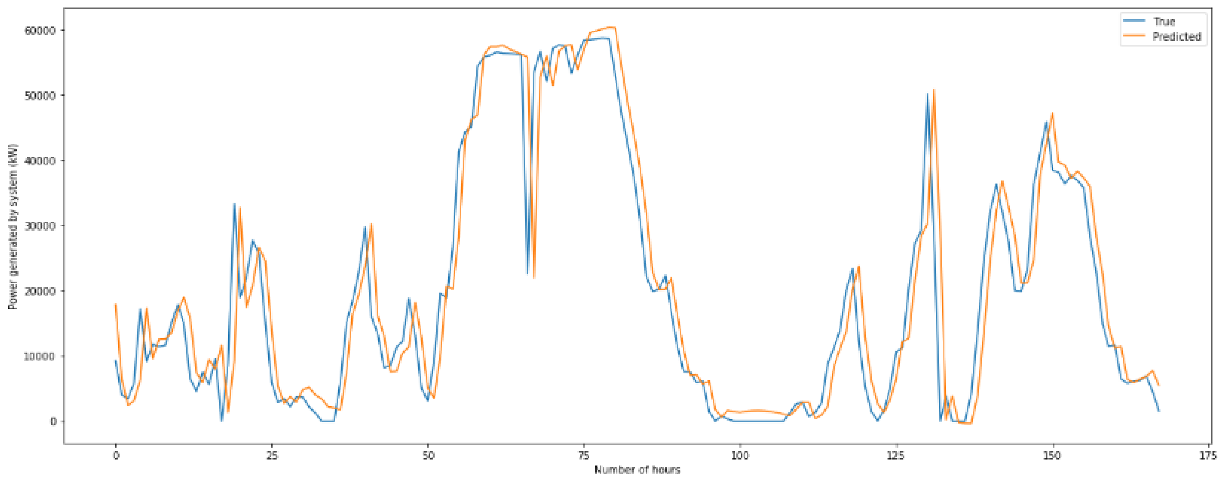
Code Link: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp5-Prediction-Batch%201-48%20hour.ipynb

Experiment 6:

1 weeks data was predicted, the error observed was 44.74%

```
Mean Absolute Percent Error:  44.742541420265795
```

```
# line plot of observed vs predicted
pyplot.figure(figsize=(20,8))
pyplot.plot(expectations_plot, label="True")
pyplot.plot(predictions_plot, label="Predicted")
pyplot.legend(loc='upper right')
pyplot.xlabel("Number of hours")
pyplot.ylabel("Power generated by system (kW)")
pyplot.show()
```

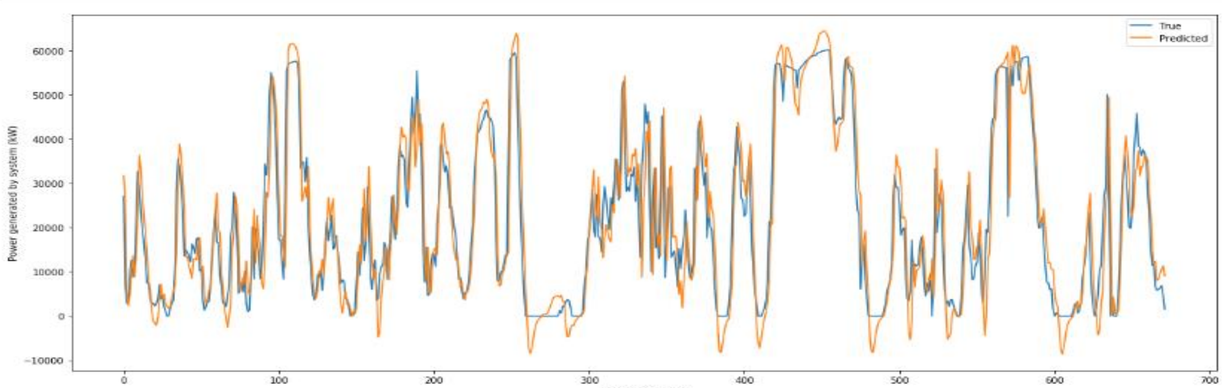


Experiment 7:

1 month's data was predicted with mean error of 52.88%.

```
Mean Absolute Percent Error:  52.88248839155405
```

```
# line plot of observed vs predicted
pyplot.figure(figsize=(20,8))
pyplot.plot(expectations_plot, label="True")
pyplot.plot(predictions_plot, label="Predicted")
pyplot.legend(loc='upper right')
pyplot.xlabel("Number of hours")
pyplot.ylabel("Power generated by system (kW)")
pyplot.show()
```



In all the above experiments in the line graph we can see that the LSTM is able to learn the pattern in the data very well and therefore giving some excellent results for a moderately long time such as month's prediction.

Then LSTM model was optimized like increasing the input batch size to control weight updates in the model, adding extra hidden layers like Batch Normalization and Dense layer with different activation functions like relu and tanh. Experiment number 8, 9 and 10 in the GitHub repo depicts the results of these experiments.

The results were quite the same for the above experiments when they were repeated with optimized model. But it was useful for the experiments which we are going to do next.

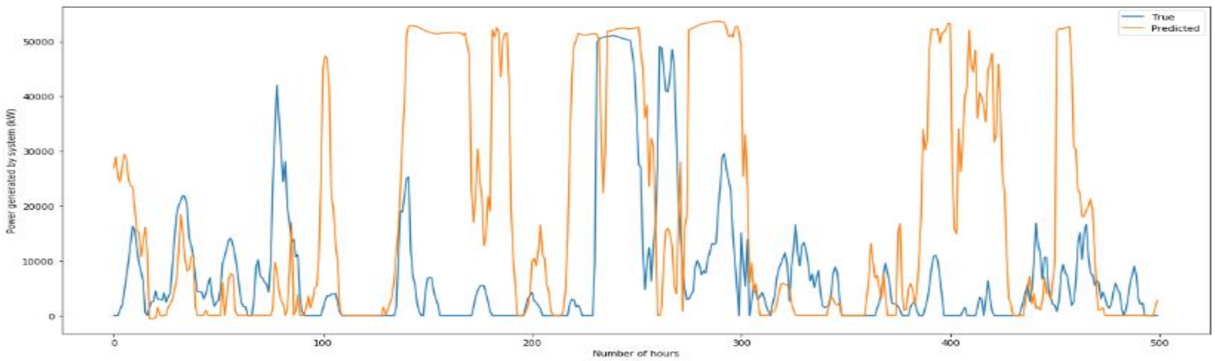
Look Ahead Prediction

Now, we wanted to predict the wind energy generated by the system ahead of time. We wanted to know if we start from today and predict energy for X days ahead how much accuracy the model gives. This is different from what we did in previous section, as we are now predicting the data of X hour starting from this hour and then evaluating results. In previous section, we were prediction for the next hour and evaluating results and then rolling on to the next hour and repeating the same. Here we are predicting X hours ahead in one go and then rolling on to the next. 1 year ahead data was predicted with different look ahead (like a sliding window). Results are as follows.

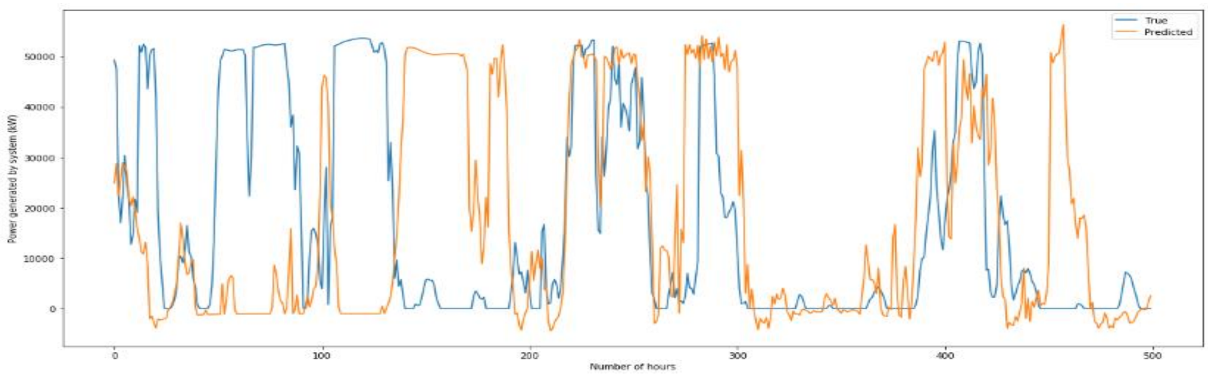
Results of Experiments for Washington(WA) as follows:

Experiment No.	Prediction duration	Result (Mean Absolute Percent Error)
11	24 hours	99.74
12	2 days	117.59
13	1 week	127.70
14	1 month	124.45

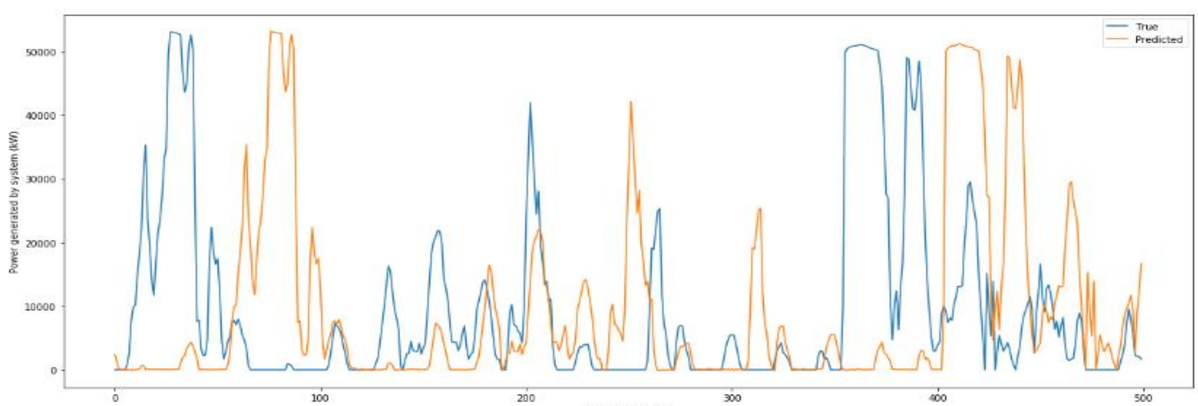
Plots for Prediction:



1 Month ahead Prediction



1 Week ahead Prediction



2 Days ahead Prediction

Code Links:

24 Hours: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp11-WA-Prediction%20Wind%20Approach%201%20Batch%201%2024%20hours%20ahead.ipynb

48 Hours: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp12-WA-Prediction%20Wind%20Approach%201%20Batch%201%2048%20hours%20ahead.ipynb

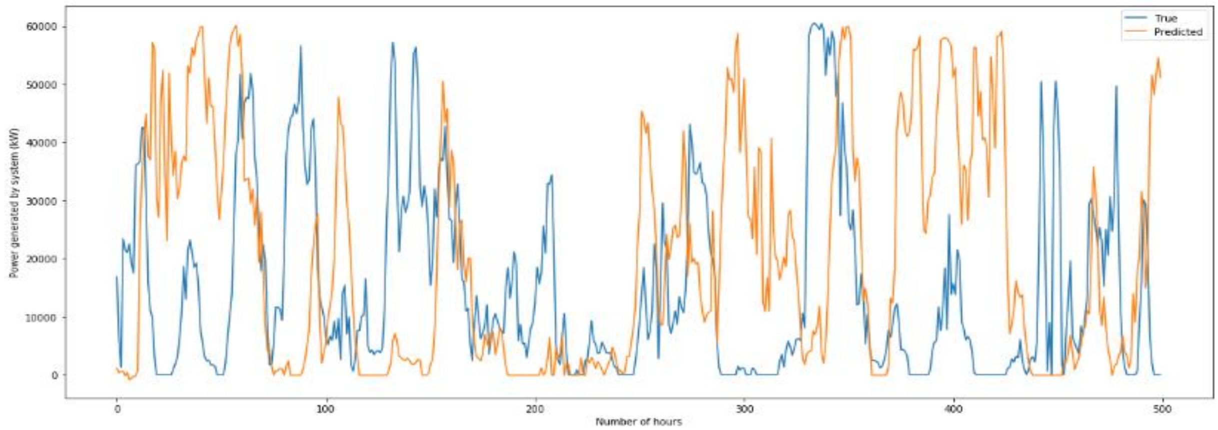
1 Week: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp13-WA-Prediction%20Wind%20Approach%201%20Batch%201%20week%20ahead.ipynb

1 Month: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp14-WA-Prediction%20Wind%20Approach%201%20Batch%201%20month%20ahead.ipynb

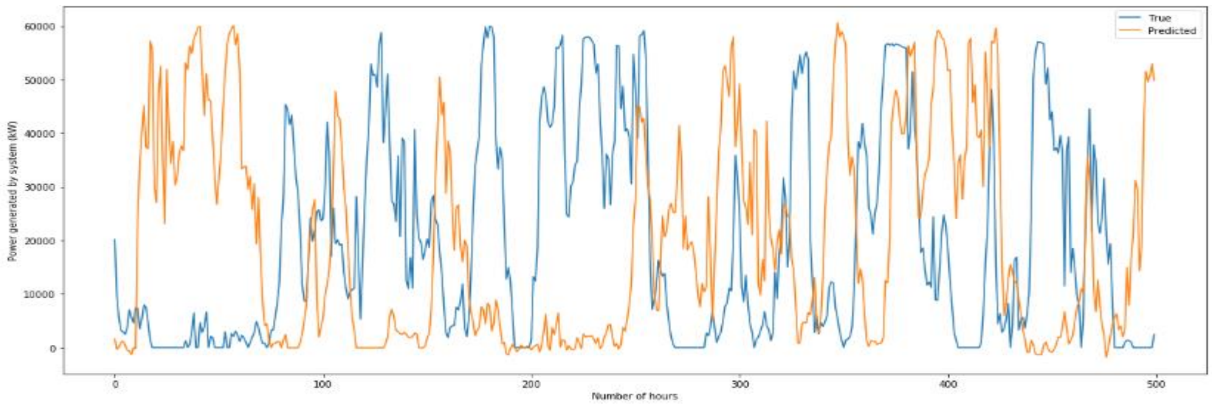
Results of Experiments for Alabama (AL) as follows:

Experiment No.	Prediction duration	Result (Mean Absolute Percent Error)
15	24 hours	97.77
16	2 days	106.72
17	1 week	117.70
18	1 month	114.90

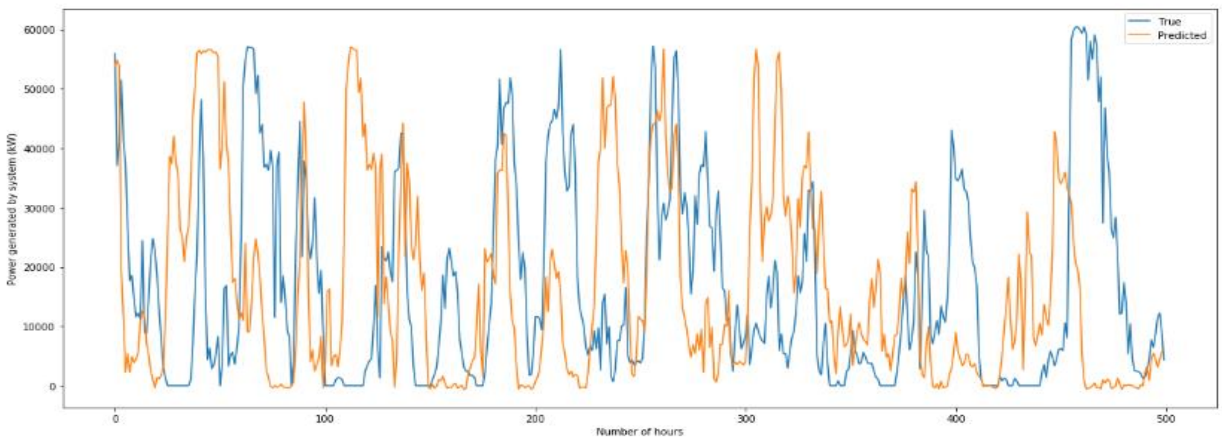
Plots for Prediction:



1 Month ahead Prediction



1 Week ahead Prediction



2 Days ahead Prediction

Code Links:

24 Hours: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp15-AL-Prediction%20Wind%20Approach%201%20Batch%201%2024%20hours%20ahead.ipynb

48 Hours: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp16-AL-Prediction%20Wind%20Approach%201%20Batch%201%202%20days%20ahead.ipynb

1 Week: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp17-AL-Prediction%20Wind%20Approach%201%20Batch%201%201%20week%20ahead.ipynb

1 Month: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM/blob/master/Exp18-AL-Prediction%20Wind%20Approach%201%20Batch%201%201%20month%20ahead.ipynb

The result were pretty same compared to SVM and AR methods which was done to predict the wind energy generated.

GitHub repo: https://github.com/ShashwatArghode/Wind-Energy-Prediction-using-LSTM

Conclusion

We observed that LSTM was able to get the pattern in the long term data i.e. seasonal and annual changes in the power generated by the system. And its performance is same as that of SVM and AR methods. But, we also observed that if the wind speed is less than 4 m/s the power generated by the system is zero. LSTM was not able to learn this pattern as this is not the part which it can understand in time series analysis. So, if a hybrid new model is created which can work as the combination of Decision Tree/Random Forest and LSTM we can improve upon these results as well.

References

1. https://en.wikipedia.org/wiki/Long_short-term_memory

2. https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

3. https://www.hindawi.com/journals/mpe/2015/939305/

4. https://machinelearningmastery.com/multivariate-time-series-forecasting-lstms-keras/

5. https://www.kaggle.com/jphoon/bitcoin-time-series-prediction-with-lstm

6. https://machinelearningmastery.com/time-series-forecasting-long-short-term-memory-network-python/

7. https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

8. https://machinelearningmastery.com/use-different-batch-sizes-training-predicting-python-keras/

9. https://machinelearningmastery.com/multi-step-time-series-forecasting-long-short-term-memory-networks-python/

10. https://machinelearningmastery.com/feature-selection-time-series-forecasting-python/

11. https://machinelearningmastery.com/time-series-forecasting-long-short-term-memory-network-python/

12. https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

13. https://github.com/tgjeon/TensorFlow-Tutorials-for-Time-Series

14. http://dacatay.com/data-science/part-5-time-series-prediction-prophet-python/

15. https://machinelearningmastery.com/diagnose-overfitting-underfitting-lstm-models/