Electricity Consumption Prediction on Comparison between State and Stateless LSTM

Gonsoo Moon
Post Master Student of Computer Science
University of Texas at Dallas,
gonsoomoon@gmail.com

Abstract

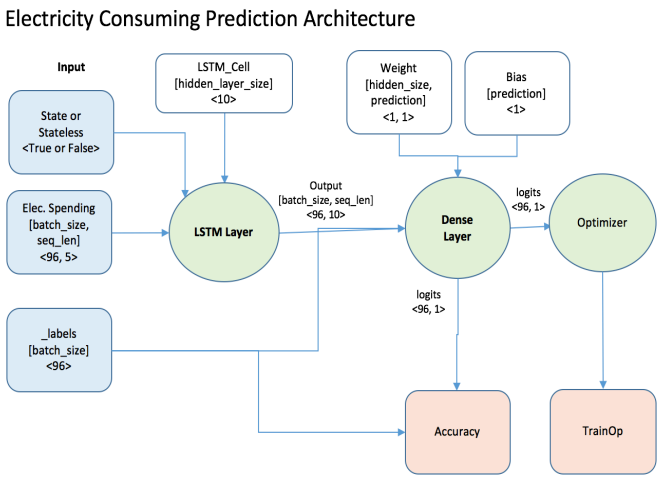
With the problem, electricity consumption prediction, how differently state and stateless LSTM (Long-Short Term Memory) are based on several sequence lengths.

1. Introduction

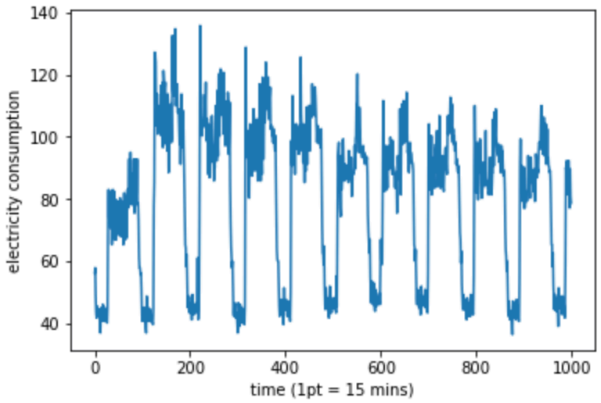
In most time series data, a stateless LSTM is mostly superior to a state one depending on a variation of data. With electricity consumption data over 4 years in U.S. this paper shows how different they are. Using [2] the electricity consumption data set that is taken at 15 minute intervals over a four-year period from 2011 to 2014, a simple LSTM network is experimented mainly for comparison of state and stateless LSTM

2. Network Architecture

This network simply consists of an input, LSTM with 10 hidden nodes and Dense layer with one hidden node, containing a parameter such as being state or stateless.



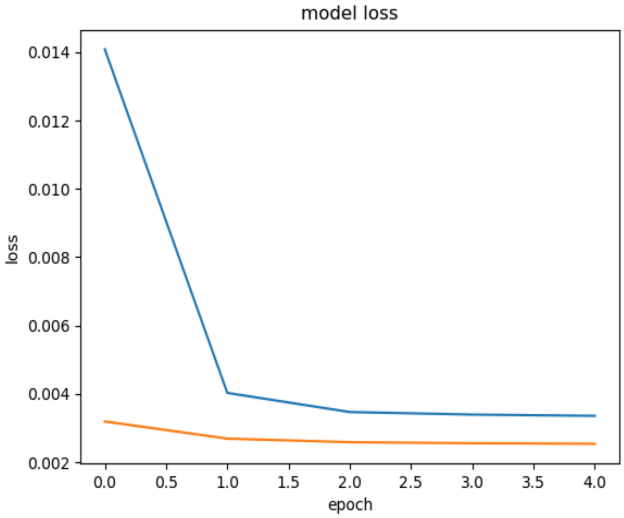
For the input, the total number of an instance is 140, 256. The number of an input sequence consisting of 5 time steps is 140,251. Y value as a label is made from the next value after one input sequence. For example, if there is a list, [56, 67, 87, 101, 89, 98, 99], with a parameter of 5 time steps, X as a feature and Y are as follows: [56, 67, 87, 101, 89] | [98], [67, 87, 101, 89, 98] | [99]. As a result, 139,554, 99.5%, as a train data set and 697, 0.5%, as a test data set were made. The Fig.2 shows an example of 1000 points at interval of 15 minutes about 10 days for electricity consumption in kw (kilo watt) in order to see what shape of the data set.



<Fig.2, 1000 samples from input data >

3. Experiment & Result

The experiment is conducted, basically making a model when two losses of train and validation data were converged shown in the Fig.3 at which the blue line means a train loss and the orange line is a validation loss. Relatively, because this data set is one dimension, 4 or 5 epochs were enough to meet the convergence. The <Fig.3> shows 96 batch size, 5 sequence lengths, stateless LSTM with 10 hidden nodes on 111,643 train data and 27,911 validation set which is 8:2 ratio of the total data set. In this training strategy, I trained the model.



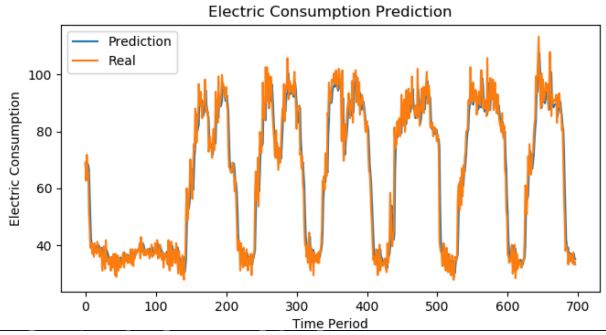
<Fig.3, loss of train and validation>

State	TimeStep	num_epochs	batch_size	MSE	Diff_Mean
Stateless	5	5	96	0.00186597	4.66721624
Statefull	5	5	96	0.00189948	4.74038315
Stateless	10	5	96	0.00193099	4.9130755
Statefull	10	5	96	0.00201959	4.99530081
Stateless	20	5	96	0.001967784	4.93182854
Statefull	20	5	96	0.00191333	4.91587665
Stateless	96	5	96	0.00220175	5.38776538
Statefull	96	5	96	0.00220125	5.37307081

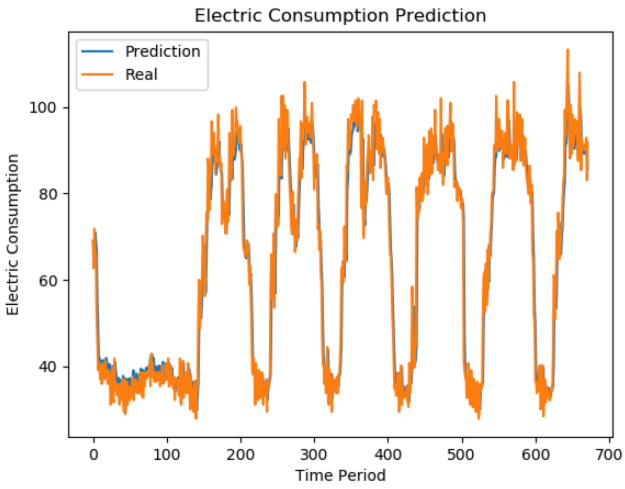
<Fig.4, the result table with a different parameter set>

The best result is 4.667 of the Diff_Mean (mean of difference between real value and prediction on the test data set) on being stateless with the parameters: The TimeStep is 5, the num_eopochs is 5 and the batch_size is 96 (about one day). As the time steps increases, the Diff_Mean also increases, in this data set, 5 to 10 of time steps is a proper level. On focus of stateless and state LSTM, being stateless is more robust in this data.

For Diff_Mean, 4.667, and 4.740 with being stateless and state on 5 of time steps are shown in the Fig.5 and Fig.6. They are very similar on visibility of two charts but from 0 to 100 points, there is slightly different. A series of prediction point in blue are slightly above over real values in the Fig.6.



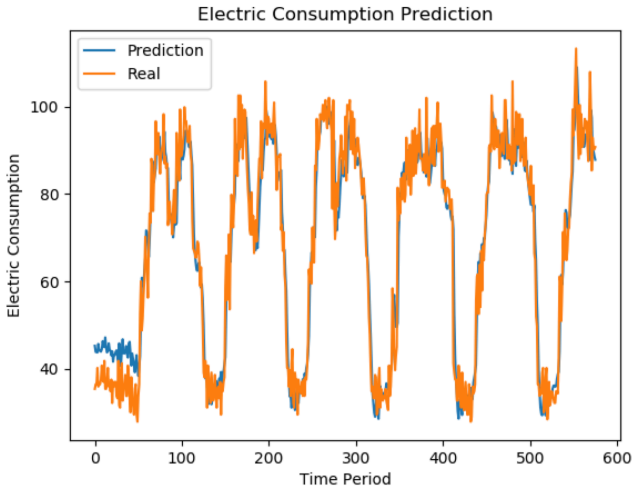
<Fig.5, Diff_Mean, 4.667 of real and predicting values with stateless and 5 of time steps>



<Fig.6, Diff_Mean, 4.740 of real and predicting values with state and 5 of time steps>

To show visually difference of Diff_Mean, 5.373, the Fig.7 shows more gap between real and prediction values

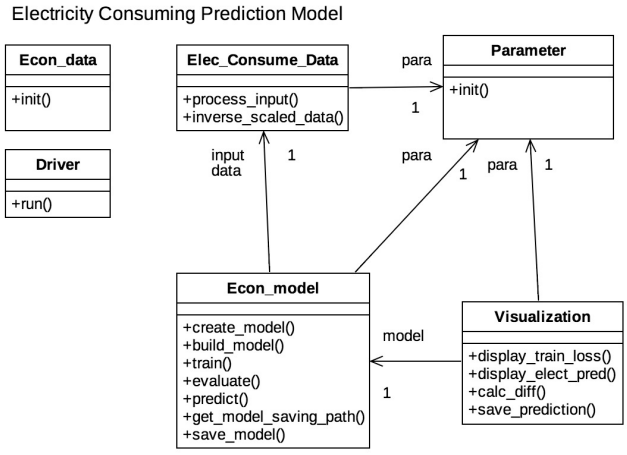
.



<Fig.7, Diff_Mean, 5.373 of real and predicting values with state and 5 of time steps>

The Fig.8 shows a class diagram of this model from an implementation perspective. The Econ_data class extracts one specific user from all users over a period of 2001 to 2004. Elec_Consumer_Data makes a train and test sequence depending on a time steps parameter. The parameter has all hyper-parameters and options such as being state and stateless. The Econ_model contains all core functions like

train(), evaluate(), predict() and save__model(). The visualization consists of displaying charts and saving them. Finally, the driver puts all together with an entry point named a run function.



<Fig.8, class diagram of the prediction model>

5. Conclusion

This paper is for judging whether which one of being state or stateless LSTM is effective on the electricity consumption prediction problem. With the experiment in this paper, being state LSTM is more robust. For most time series data, the fact has been proven. Also, this problem on the data set are the same as the usual cases.

6. References

[1] Antonio Gulli, Sujit Pal (2017). Deep Learning with Keras. Packt Publishing Ltd

[2] UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014