Assignment 2

Forecasting Electricity Demand of Residential Building with LSTM

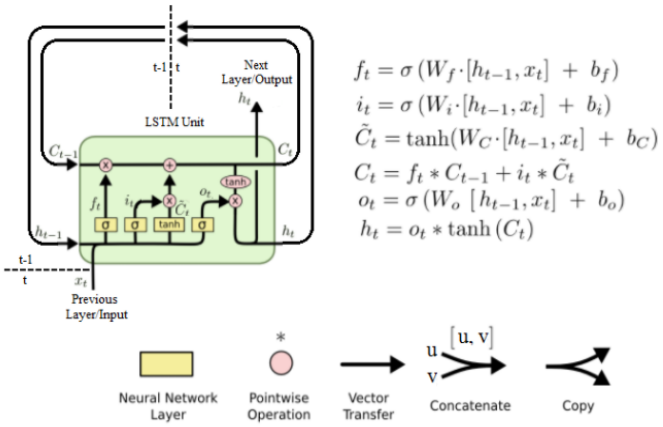
Keshav Sharma (251189343)
Department of Electrical and Computer Engineering
Western University
London, Canada
kshar26@uwo.ca

I. Introduction

Problem is to forecast current hour electricity consumption of a house by using previous hour values for which we will using Long-Term Short Memory Deep Neural Network (LSTM). Above models support multiple variables. For comparing and checking performance we will use RMSE and R2 performance metrices to compare models. *The Dataset contains 8 years of hourly electricity demand for a residential building from 2012 to 2018 with one hour time step. The dataset contains multiple variables like air pressure, temperature, humidity, wind speed, solar radiation, and cloud cover at predetermine location. There is total 70080 recordings. The dataset is provided by IEEE [1]*

A. Long-Term Short Memroy Neural Network

LSTM is a deep learning technique which is feedback neural network. This algorithm is very good for learning dependencies in sequence prediction problems. LSTM use recurrent Neural network



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Fig. 1 LSTM Network [4]

LSTM cell has 3 parts knows as gates, first is forget gate, second is input gate and third is output gate. LSTM have hidden states which represents previous timestamp H(t-1) and current time stamp H(t). and cell states of previous timestamp C(t-1) and current time stamp C(t). Hidden state is known as short term memory and cell state called long-term memory. In LSTM network first step decide whether to keep data from previous timestamp or forget it using equation

$$f_t = \sigma\left(x_t * U_f + H_{t-1} * W_f\right)$$

Fig. 2 LSTM forget gate equation [4]

Where,

- Xt: input to the current timestamp.
- Uf: weight associated with the input
- Ht-1: The hidden state of the previous timestamp
- Wf: It is the weight matrix associated with hidden stat

Later sigmoid function applied over it which will make function result between 1 and 0. Then ft is multiplies by cell state of previous timestamp, if it gives 0 then forget everything and if it gives 1 forget nothing.

Input gate is used to quantify importance of new information coming from input using equation

$$i_t = \sigma\left(x_t * U_i + H_{t-1} * W_i\right)$$

Fig. 3 LSTM input gate equation [4]

Where,

- Xt: input to the current timestamp.
- Ui: weight matrix of the input
- Ht-1: The hidden state of the previous timestamp
- Wi: It is the weight matrix of input associated with hidden state

Now, sigmoid function is applied to get result between 0 and 1. After that output gate use equation

$$o_t = \sigma(x_t * U_o + H_{t-1} * W_o)$$

Fig. 4 LSTM output gate equation [4]

The it put result in sigmoid function to get result between 0 and 1.

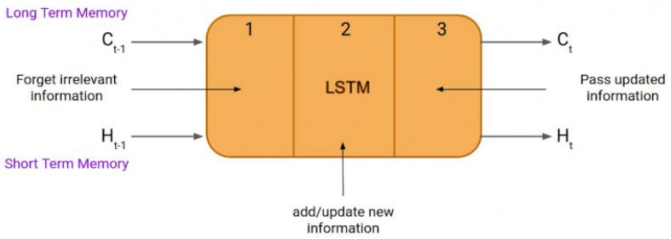


Fig. 5 LSTM cell[4]

These gates allowed LSTM to retain memory for many time steps and protect cell from irrelevant events which make LSTM very good for predicting sequential data like timeseries, text etc. that's why we choose LSTM for this assignment.

II. Dataset

This dataset contains the electricity demand of a residential building which contains 7 variables air pressure, air temperature, relative humidity, windspeed, solar irradiation, total cloud cover, electricity demand values and heat demand values. Which contains 70080 recorded at the difference of one hour from year 2012 to year 2018. There are some missing values in data for each model we must prepare data for them and split in train and test set.

```
NUmber of Rows and Columns
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70080 entries, 0 to 70079
Data columns (total 9 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Time                            70080 non-null  object
 1   air_pressure[mmHg]              69934 non-null  float64
 2   air_temperature[degree_celcius] 69903 non-null  float64
 3   relative_humidity[%]            69903 non-null  float64
 4   wind_speed[M/S]                 69125 non-null  float64
 5   solar_irridiation[W/m²]         70080 non-null  int64
 6   total_cloud_cover[from_ten]     69837 non-null  object
 7   electricity_demand_values[kw]   70073 non-null  float64
 8   heat_demand_values[kw]          70073 non-null  float64
dtypes: float64(6), int64(1), object(2)
```

Fig. 6 dataset Information

III. Data Prepration and cleaning

This dataset has many discrepancies we cannot straight away use this data as it is. We must perform some data preparation methods to prepare data for LSTM.

1. Replacing whitespace from columns names to make it easier to understand by computer.

```
Data columns (total 9 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Time                            70080 non-null  object
 1   air_pressure[mmHg]              69934 non-null  float64
 2   air_temperature[degree_celcius] 69903 non-null  float64
 3   relative_humidity[%]            69903 non-null  float64
 4   wind_speed[M/S]                 69125 non-null  float64
 5   solar_irridiation[W/m²]         70080 non-null  int64
 6   total_cloud_cover[from_ten]     69837 non-null  object
 7   electricity_demand_values[kw]   70073 non-null  float64
 8   heat_demand_values[kw]          70073 non-null  float64
dtypes: float64(6), int64(1), object(2)
```

Fig. 7 dataset Information

2. Change total_cloud_cover data to numerical data by changing no cloud value to 0 and fraction value to decimal value
3. Fore forecasting problem we have to change Time column timeseries of datatype datetime64[ns] from object and make it index of series.
4. replace NaN values and missin values by using interpolate function which will replace them by values between 25 percentile and 75 percentile.

min

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 70080 entries, 2010-12-01 00:00:00 to 2018-11-28 23:00:00
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   air_pressure            70080 non-null  float64
 1   air_temperature         70080 non-null  float64
 2   relative_humidity       70080 non-null  float64
 3   wind_speed              70080 non-null  float64
 4   solar_irridiation       70080 non-null  int64
 5   total_cloud_cover       70080 non-null  float64
 6   electricity_demand_values 70080 non-null  float64
 7   heat_demand_values      70080 non-null  float64
dtypes: float64(7), int64(1)
memory usage: 4.8 MB
```

Fig. 8 dataset Information after removing NaN

5. Next we checked if series are stationry or not using ADF[5] test, This test tell weather time series is stationary or not stationary and we need stationary time series which does not have any trend and seasonality in it. After testing every series adf result tell that all series are staionry.

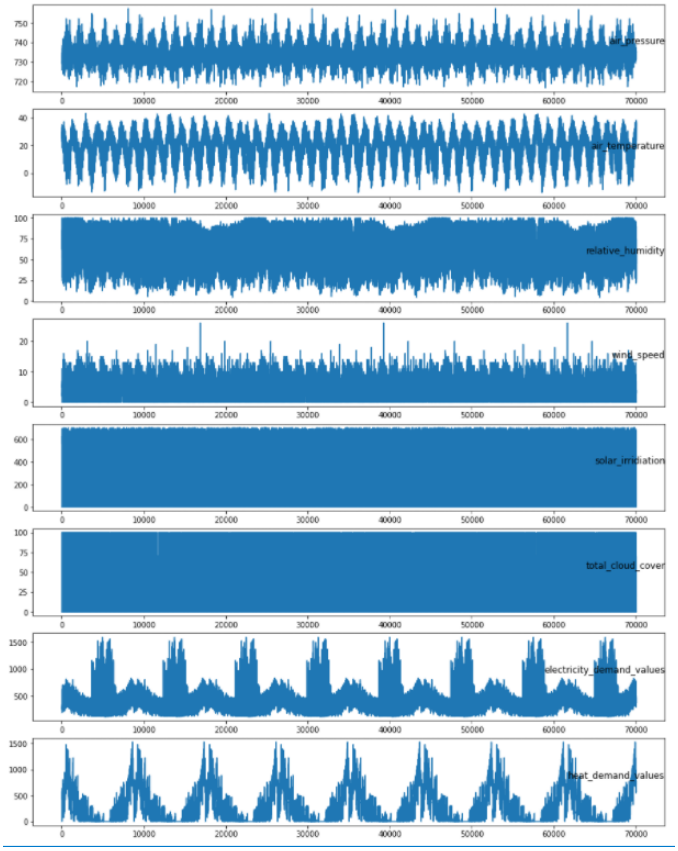


Fig. 9 stationary series

Now data is stationary we can use it in models for forecating.

6. For predicting current hour electricity consumption of house we need engineer lag feature of previous hour for all varibales. Before making lag features firstly scale down the data to increase speed of model.for scaling we are using MinMax scalar which will scale down the data between 0 and 1.

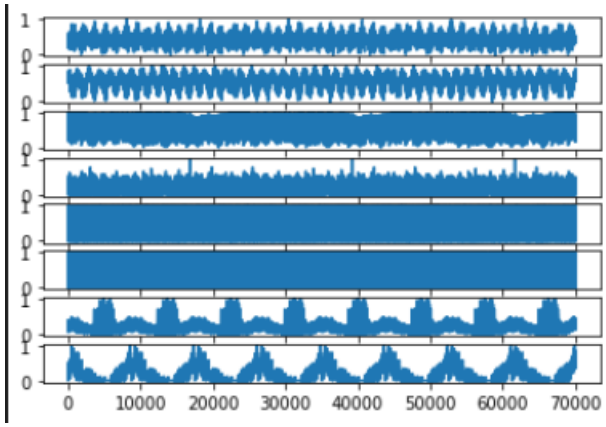


Fig. 10 scaled data

After scaling, we make lag features by subtracting current value with previous value. We will use lag feature of as input and current electricity value scaled(var7(t)) as target.

var1(t-1)	var2(t-1)	var3(t-1)	var4(t-1)	var5(t-1)	var6(t-1)	var7(t-1)	var8(t-1)	var7(t)
0.321951	0.682842	0.843750	0.192308	0.0	0.0	0.119342	0.283121	0.099477
0.314634	0.731369	0.760417	0.269231	0.0	0.0	0.099477	0.309930	0.090764
0.302441	0.826690	0.604167	0.269231	0.0	0.2	0.090764	0.316040	0.097982
0.368292	0.807626	0.604167	0.076923	0.0	0.5	0.097982	0.357005	0.098184
0.392683	0.644714	0.958333	0.115385	0.0	0.2	0.098184	0.360017	0.111240

Fig. 11 lag features

7. Now we split training and testing data to increase the speed of training of this model, we will fit the model on the first 2 years, then evaluate it on the remaining 6 years of data

IV. Methodology and Evaluation

Methodology:

For this forecasting problem we are using LSTM deep neural network for its benefits over sequential data. In our model we will use evaluate 3 models' model_prev from previous assignment 1, model 1 without any tuning and model 2 which is same model after tuning model 1. All models are using same timeseries. We will evaluate all 3 models on root mean square (RMSE) and R2 scores and plot actual vs prediction plot. We are predicting current electricity demand from previous hour.

Evaluation:

LSTM model_prev from previous assignment

In this model we add 4 LSTM layers followed by dropout layers. Unit size is 50, Dropout layers were added to prevent overfitting. Then compile with mean squared error with 25 epochs. As LSTM is time consuming, we only run for 25 epochs.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 6, 50)             10400
_________________________________________________________________
dropout (Dropout)            (None, 6, 50)             0
_________________________________________________________________
lstm_1 (LSTM)                (None, 6, 50)             20200
_________________________________________________________________
dropout_1 (Dropout)          (None, 6, 50)             0
_________________________________________________________________
lstm_2 (LSTM)                (None, 6, 50)             20200
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 50)             0
_________________________________________________________________
lstm_3 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_3 (Dropout)          (None, 50)                0
_________________________________________________________________
dense (Dense)                (None, 1)                 51
=================================================================
Total params: 71,051
Trainable params: 71,051
Non-trainable params: 0
```

Fig. 12 LSTM Model summary

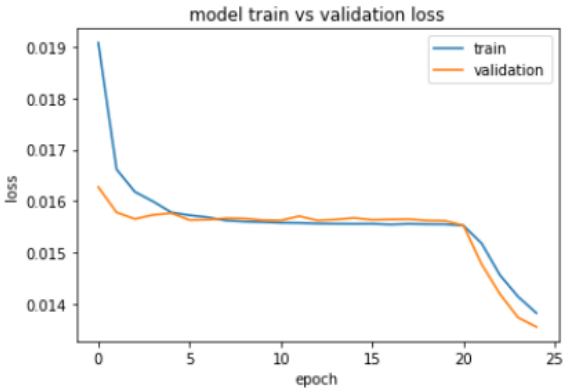


Fig. 13 LSTM model_prev training and validation loss

We got 56.3761 RMSE and 0.154 R2 for this model. We can see model us underfit at starting and ion the end.

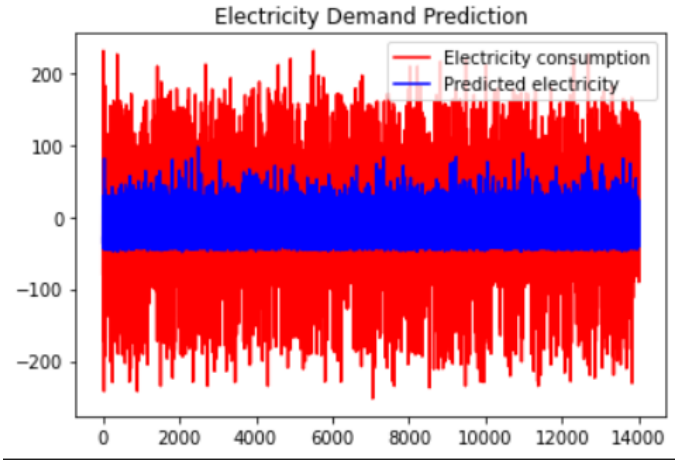


Fig. 14 LSTM Model_prev actual vs prediction Plot

We can see this model is not predicting very well and took too much time to give results.

LSTM model 1

This model contains 50 neurons as first hidden layer and 1 neuron at output layer. Input shape is 1 time step with 8 features. Here mse is used as loss function and adam version of gradient descent as optimizer.

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 50)                11800
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 51
=================================================================
Total params: 11,851
Trainable params: 11,851
Non-trainable params: 0
_________________________________________________________________
```

Fig. 15 LSTM Model 1 Summary

We fit this model with batch_size of 10 and 50 epochs also validation set is 33%. And plot training and validation loss below. And we can see that model 1 is overfitting.

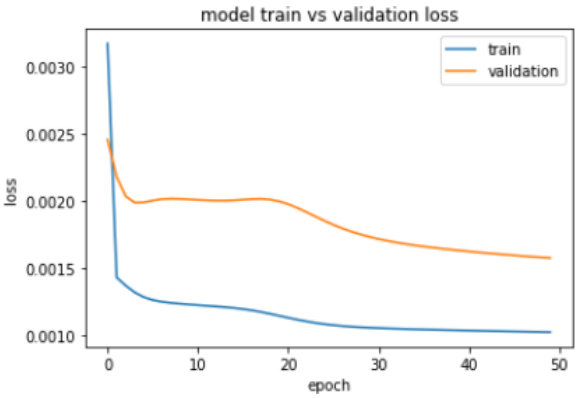


Fig. 16 LSTM Model 1 training and validation loss

We got RMSE value as 1.877 and R2 value 0.9197 for this model.

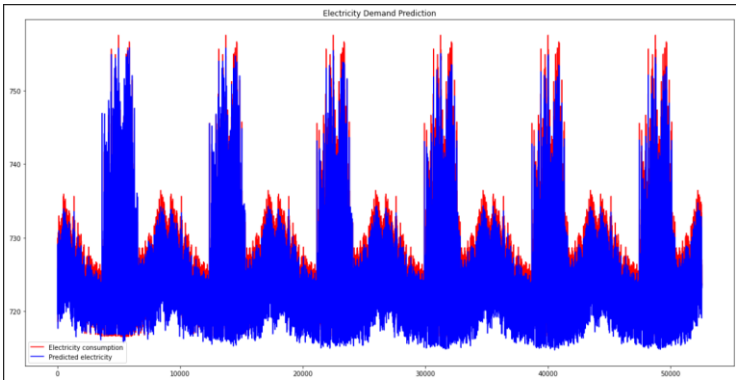


Fig. 17 LSTM Model 1 actual vs predicted plot

LSTM model 2

Now we will try to tune the previous model by adding 2 more hidden LSTM layers followed by leaky Relu layer so that if hidden layer output is less than 0, the next input does not become 0 which is dying Relu problem [6] and after that we put dropout layer to overcome overfitting.

This model contains 128 neurons as first and second hidden layer, 64 in third hidden layer and 1 neuron at output layer. Input shape is 1 time step with 8 features. Here mse is used as loss function and adam version of gradient descent as optimizer.

```
Model: "sequential_14"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_42 (LSTM)               (None, 1, 128)            70144
_________________________________________________________________
leaky_re_lu_20 (LeakyReLU)   (None, 1, 128)            0
_________________________________________________________________
dropout_40 (Dropout)         (None, 1, 128)            0
_________________________________________________________________
lstm_43 (LSTM)               (None, 1, 128)            131584
_________________________________________________________________
leaky_re_lu_21 (LeakyReLU)   (None, 1, 128)            0
_________________________________________________________________
dropout_41 (Dropout)         (None, 1, 128)            0
_________________________________________________________________
lstm_44 (LSTM)               (None, 64)                49408
_________________________________________________________________
dropout_42 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 65
=================================================================
Total params: 251,201
Trainable params: 251,201
Non-trainable params: 0
_________________________________________________________________
```

Fig. 18 LSTM Model 2 summary

We fit this model with batch_size increased to 75 to get more predictions at new input and 50 epochs also validation set is 33%. And plot training and validation loss below.

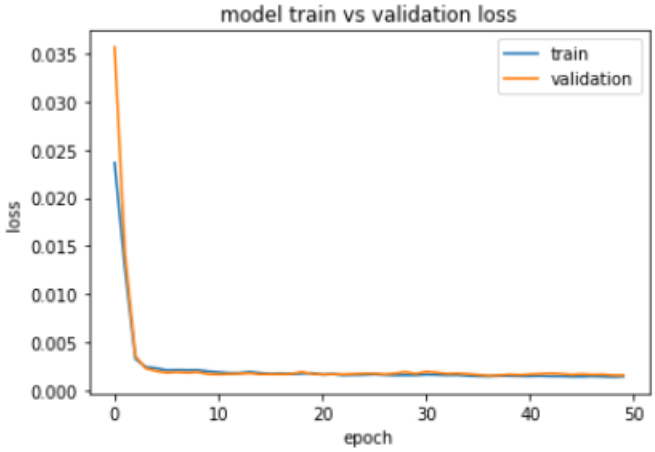


Fig. 19 LSTM Model 2 training and validation loss

We can see that this model is better than previous models. The RMSE value for this model is 1.798 and R2 value for this model is 0.9263. both are better that previous model.

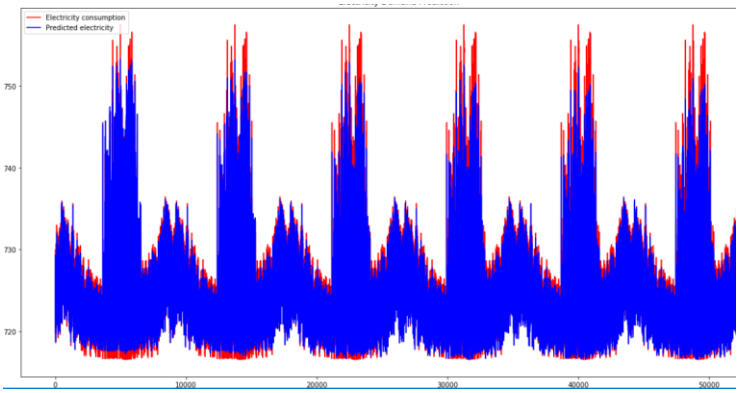


Fig. 20 LSTM Model actual vs prediction plot

REFERENCES

[1] https://ieee-dataport.org/open-access/8-years-hourly-heat-and-electricity-demand-residential-building#files

[2] https://www.machinelearningplus.com/time-series/vector-autoregression-examples-python/

[3] https://en.wikipedia.org/wiki/Linear_regression

[4] https://machinelearningmastery.com/deep-learning-for-time-series-forecasting/

[5] https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test

[6] https://datascience.stackexchange.com/questions/5706/what-is-the-dying-relu-problem-in-neural-networks