Forecasting Building Energy Demand Using Neural Network

Introduction

Electricity is an integral part of economic and social development for any country. It touches lives of everyday people, affects industrial production and supports all the basic infrastructure of the country. Currently it is very expensive to store electricity in large amounts, thus it most of the electricity that is produced, needs to put to immediate use. To limit the waste of electricity is becomes very important to estimate the load demand beforehand. Load forecasting also helps to improve revenue for power companies and maintain grid stability.

There are various factors that affect the load demand of a building, ranging from customer behavior to historical load demand and load profile and special holidays among others in one day load forecasting. Temperature, humidity and other weather features also play an important role determining the electricity load of a given building. This paper attempts using these above-mentioned features and Neural Network architecture to account for the non-linearity present in these features to forecast hourly loads for a Scaffie Building present in Carnegie Mellon University, Pittsburgh. First a simple linear regression method is used to determine a baseline accuracy for forecasting model. Then a simple Deep Neural Network is implemented using the same features. Finally, recurrent neural network model, accompanied with some convoluted neural network layers was implemented. This architecture is based on the “Load Forecasting via Deep Neural Networks” paper published in 2017 [1].

Data Cleaning and Processing

The original load data of the building consists pf building load readings from November 2013 to November 2014, every 10 minutes. Since the aim of the project is to predict hourly loads, the temporal resolution of the original dataset is changed to one hour. The weather file is extracted from Underground Weather website [2]. It is obtained using web scraping module beautiful soup in python. contains hourly temperature, humidity, wind speed and weather condition values. Originally weather condition was a categorical data, suggesting type of weather, like snow, sunny, thunderstorm, cloudy, and so on. This weather file is then merged with load data file to create the whole dataset. Some of the features to be used for forecasting include day of the week and hour of the day. Thus, using the timestamp of a given data, one hot vectors for weekdays and hours is created. The aim of the project is to use load demand data of the past week to predict load demand of given hour. For example, if load demand on July 3, 5:00 pm needs to be predicted, then load data from June 27, 5:00pm to July 2, 5:00 pm will be used as input. Thus, for each hourly prediction a total of 168 hours of past load demand (24×7) is to be used. Thus, the dataset is rearranged such that to each corresponding reading, there are 168 columns, each column having load demand of some hour in the past week of that particular reading. Finally, the dataset is split in to training set (70%), validation set (20%) and test set (10%).

Feature Analysis

Following are the features used in forecasting:

- Load Demand of the given Hour
- T-1 to T-168: Columns containing hourly Load Demand of past week
- Hour 0 to Hour 23: One hot vector representing hour of the day. 12:00 am is given as hour 0.
- Weekday 0 to Weekday 6: One hot vector for day of the week. Monday is given as Weekday 0.
- Temperature: Gives temperature in ^{0}C for given hour
- Wind Speed: Gives wind speed in m/s
- Humidity: Relative Humidity (%) of the given hour
- Weather Condition: Categorical Feature, converted to integers, as mentioned above.

Below are shown behavior of some of the features throughout the year. Figure 1 shows the total daily load for a given day of the week. Load is drastically reduced during the weekend, which is understandable for a university building. Figure 2 shows average hourly load for a given hour of the day. Load is higher between 10:00 am to 6:00 pm, which are the usual class hours, while the load reduces during the evening and is least at 5:00 pm, which can be explained by typical operation of college building. Figure 3 shows total monthly load for a given month in a year. Load is higher during months of June, July and august, which makes sense as there is high cooling energy demand during this month. The unusual dip in consumption during November can be explained by missing data for this month. Figure 4 shows temperature in F throughout the year.

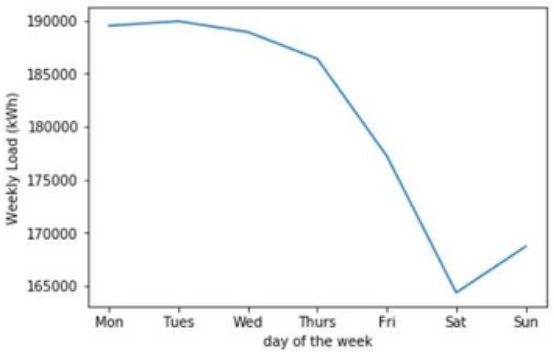

Figure 1 Average daily Load for given Weekday

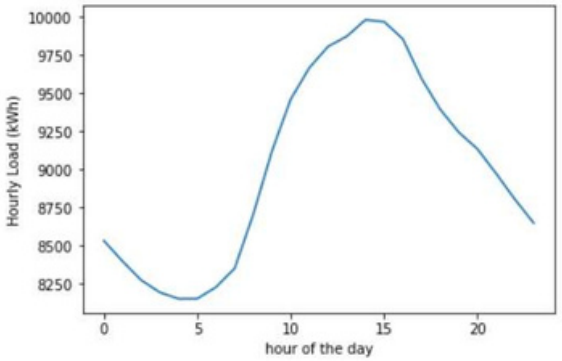

Figure 2 Average Hourly Load for given Hour of the Day

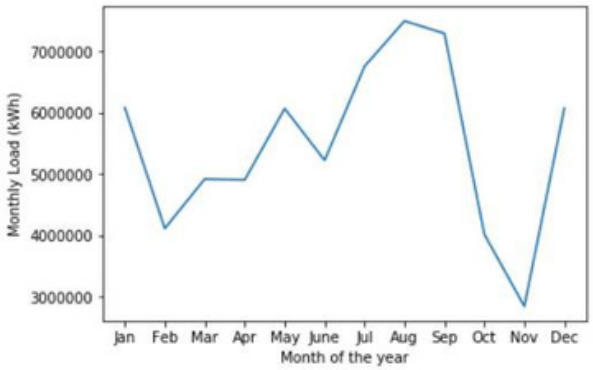

Figure 3 Total monthly Load for a given Month in a Year

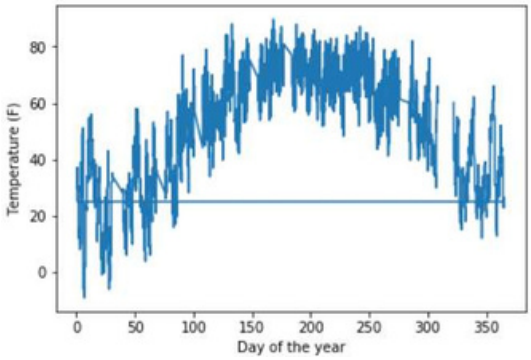

Figure 4 Temperature throughout the Year

Linear Regression Model

To implement a simple regression model on the given data set, python's sci-kit learn module is used. To compare all models, loss functions are optimized based on the mean square error. To compare their performance, mean square error (MSE) and accuracy score (R2) is used. For linear regression model, accuracy score is defined as:

$$R^2 = 1 - \frac{RSS}{TSS}$$

Where, RSS is the Residual sum of squares, i.e. sum of all mean square errors, while TSS is the total variance of Y. R2 value helps in visualizing the effectiveness of the regression better, as seen in the plot below. This same accuracy score is further used for simple neural network and C-RNN models. Following results were obtained:

Table 1 Accuracy Results for Linear Regression Model

	Training Set	Validation Set	Test Set
MSE	0.35	0.40	0.52
R2	0.87	0.86	0.81

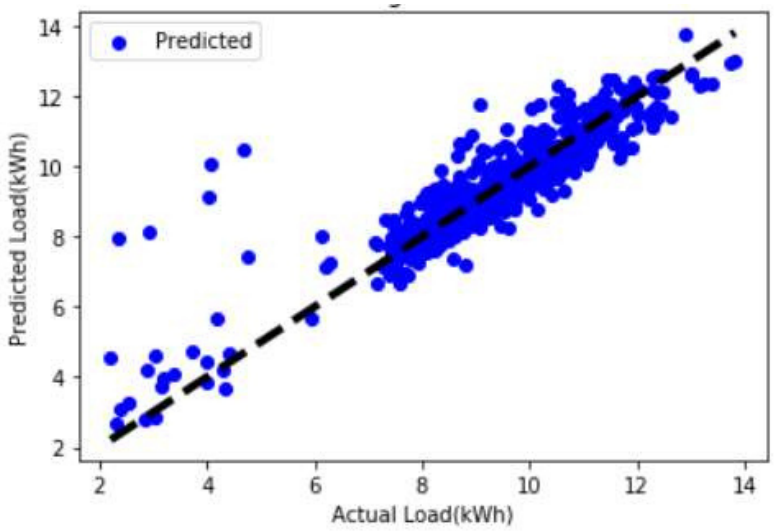

Figure 5 Accuracy Score Visualization for Linear Regression

The performance of the Linear Regression Model is used as the Baseline for the following to models to compare with.

Simple Neural Network

This neural network model employs 4 layers, with first 3 layers having 512 hidden units, while the last layer has only one activation output. For all the layers a rectified linear unit (ReLu) activation is used, except layer 3 for which sigmoid activation is used for output activation. Some other parameters used here are:

- Optimizer: Adam Optimizer
- Learning Rate: 0.001
- Batch Size: 64
- Number of epochs: 200
- Parameters W (weights) and b (bias) are initialized randomly

Following results were obtained:

Table 2 Accuracy Result for Simple Neural Network

	Training Set	Validation Set	Test Set
MSE	0.21	0.26	0.39
R2	0.96	0.96	0.95

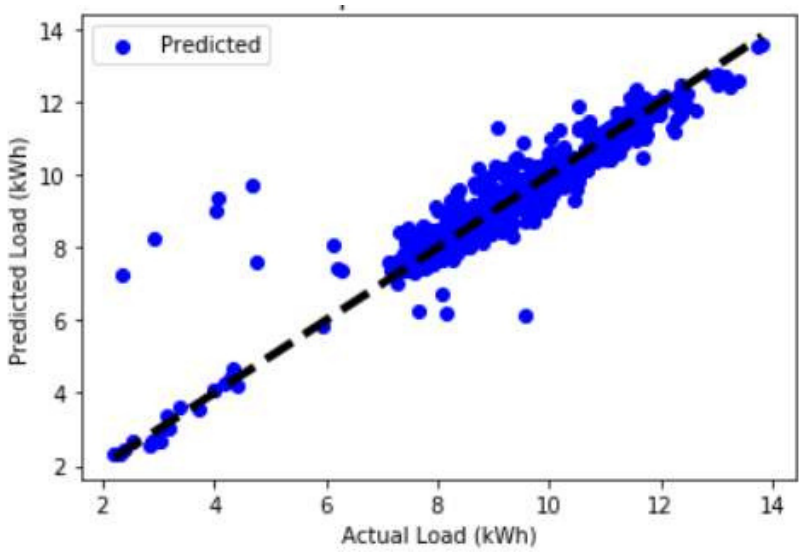

Figure 6 Accuracy Score Visualization for Simple Neural Network

As seen in the figure above the tightness of predicted values is much better for the Neural Network compared to the linear regression model.

It must be mentioned here that the MSE and accuracy score for the Neural Network would depend on many factors like number of layers, number of epochs, number of hidden units, and many more. A grid search can be performed to identify optimal parameters to maximize the accuracy of this model.

C-RNN

This model employs a combination of convolution and recurrent neural network (LSTM), to estimatethe building load. The architecture of this model is based on the model given in "Load Forecasting via Deep Neural Networks" paper published in 2017 [1]. The model is explained in the followingfigure:

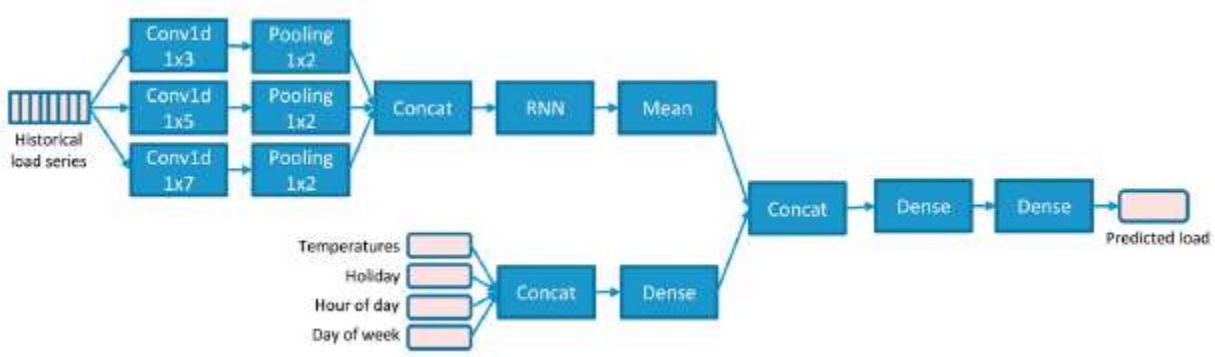

Figure 7 C-RNN Model Architecture

Keras module along with Tensorflow is used to model this architecture in python. Convolution and Recurrent layers are used only for the historical data (T-1 to T-168 columns) as only these columns are temporally dependent. Other features are concatenated separately and then added to historical data after it's passed through LSTM layer. It is expected that this model gives a higher accuracy than simple neural network. Some parameters used here are:

- Optimizer: Adam Optimizer
- Learning Rate: 0.0001
- Momentum: 0.8
- Batch Size: 20
- Number of epochs: 20
- Parameters W (weights) and b (bias) are initialized randomly

Following results were obtained:

Table 3 Accuracy results for C-RNN Model

	Training Set	Validation Set	Test Set
MSE	0.48	0.52	0.61
R2	0.83	0.82	0.77

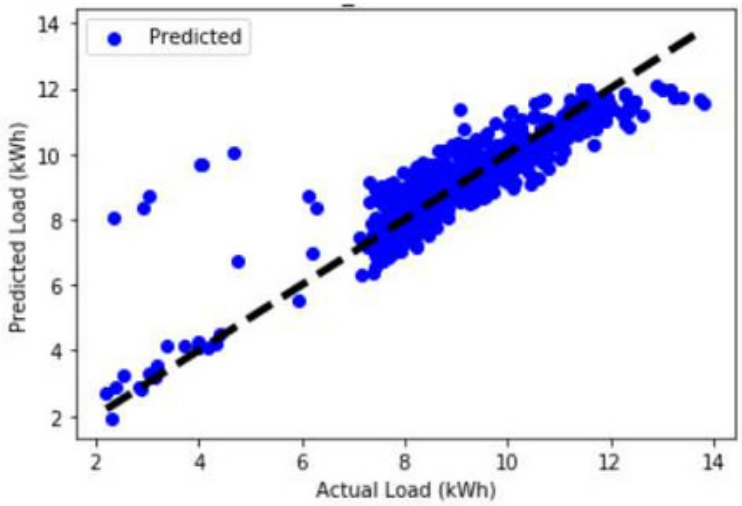

Figure 8 Accuracy Score Visualization for C-RNN Model

The C-RNN model drastically underperforms, and has MSE and accuracy score, a little worse than the linear regression problem. This may be attributed the fact that there are very few data points (5103) in the training set, which is very less for such a deep Neural Network. Vanishing gradient can also be a reason for lower accuracy as is the case many times when Convolution Neural Network is used. This issue can be resolved by adding more data or reducing the number of layers in the network. Also like the Simple Neural Network, the MSE and accuracy score for the Neural Network would depend on many factors like number of layers, number of epochs, number of hidden units, and many more. A grid search can be performed to identify optimal parameters to maximize the accuracy of this model.

Result Comparison

The Simple Neural Network outperforms the linear Regression model by 14% (based on MSE and 15% based on Accuracy Score. By altering various parameters like number of epochs, learning rate, etc., the performance of Simple Neural Network can be further increased. The C-RNN model performs a little worst than Linear Regression Model, which can be attributed to the smaller size of the given data.

References

[1] W. He, "Load Forecasting via Deep Neural Networks," Procedia Computer Science, vol. 122, pp. 308-314, 2017.

[2] Wunderground, "Pittsburgh International, PA," Wunderground, 28 November 2018. [Online]. Available: https://www.wunderground.com/history/daily/KPIT/date/1997-1-1. [Accessed 28 November 2018].