A REPORT ON

IMPLEMENTATION OF A DEEP LEARNING BASED MODEL FOR VERY SHORT-TERM ELECTRICITY LOAD FORECASTING IN THE EMERGING BUSINESS SCENARIO

Manav Kaushik	2016B3A30472P	Msc. Economics + B.E. EEE
Aviral Sethi	2016B3A70532P	Msc. Economics + B.E. CS
Mukul Jain	2016A7PS0046G	B.E. Computer Science
Aditya Pramod Nahata	2016B3A30502P	Msc. Economics + B.E. EEE

Prepared in Partial Fulfillment of
Practice School -1 Course No:
BITS C221 / BITS C231 / BITS C241 / BITS GC221 / BITS GC 231

AT

ADANI POWER TRAINING AND RESEARCH INSTITUTE ,
AHMEDABAD

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE , PILANI
(JULY , 2018)

Acknowledgement

Our team wishes to express deepest gratitude to our guide and Head of Adani Power Training & Research Institute (APTRI), Mr. Sandeep Dixit for his consistent motivation and guidance throughout the duration of Practice School 1 which helped us to achieve the set target and complete the project within the stipulated time.

Moreover, we would like to thank our mentor at Adani Power Training & Research Institute (APTRI), Mr. Kaushik Purohit for providing us with all the required facilities and sources throughout our project.

Last, but not the least, we wish to express our gratitude towards our faculty mentor, Mr. Prateek Kala and co-instructor, Mr. Yash Parik for their regular guidance and advices on the project.

Abstract

Electricity Load Forecasting is a highly important task in the business of electricity distribution as error in it can cause the distributors heavy penalties and loses. Our project addresses the same problem and thus tries to minimize this error using some of the most advanced and precise technologies and methods. Our project is to **Implement a Very Short-Term Electricity Load Forecasting Model** using the techniques of **Machine Learning: Artificial Neural Networks**.

The model has been prepared by 4 students of BITS Pilani under the supervision of Mr. Sandeep Dixit, Head of Adani Power Training & Research Institute.

This report describes the whole working of the model, economic significance of the results achieved and techniques implemented to build and optimize the model.

The model gives highly accurate predictions for short-term electricity load which can be of high significance for firms involved in the electricity distribution and firms like Adani Power which is about to enter the distribution sector.

The model uses the concept of **Artificial Neural Networks (ANN)** which is one of the most advanced techniques in Machine Learning. This ANN based model combined with several optimization techniques helped us to successfully bring down the theoretical error in electricity load forecasting to less than **2%.**

We trained the model using data available for the country: BELGIUM. The data was for the past 2.5 years and several parameters were precisely chosen to make good predictions. The weather data was collected from a website named: **Dark Sky.** Moreover, the actual load data was gathered from the official website of **European Network of Transmission System Operators for Electricity (ENTSOE)**.

The model was trained in such a way so as to get predictions for every 15 minutes interval making it even more valuable and economically significant.

Table Of Contents

Introduction

The 21st century has seen the emergence of a new form of asset i.e Data . With the huge amount data that is generated today and the development of highly efficient and compact computer systems it is now possible to utilize this data and generate results that were only a topic of imagination for the 20th century with a high accuracy.

The following report gives a detailed description of the processes that were involved in the making of a Data Science model to predict the electricity load for a region for a very short interval i.e for every 15-mins interval using the techniques of machine learning(Artificial neural networks).

Why Python?

We built our complete model using the python programming language for the following reason:

Python comes with a huge amount of inbuilt libraries. Many of the libraries are for Artificial Intelligence and Machine Learning. Some of the libraries are Numpy , Pandas , Scikit-learn (for data mining, data analysis and machine learning) , etc. The list keeps going and never ends. You can find some libraries here.

What makes Python favourite for everyone is its powerful and easy implementation. For other languages, students and researchers need to get to know the language before getting into ML or AI with that language. **This is not the case with python**. Even a programmer with vert basic knowledge can easily handle python. Apart from that, the time someone spends on writing and debugging code in python is way less when compared to C, C++ or Java. This is exactly the students of AI and ML wants. **They don't want to spend time on debugging the code for syntax errors, they want to spend more time on their algorithms and heuristics related to AI and ML**.

Not just the libraries but their tutorials, handling of interfaces are easily available online. People build their own libraries and upload them on GitHub or elsewhere to be used by others. All these features make Python suitable for them.

Forecasting Methods

Over the past several years a number of forecasting methods have been developed. For medium and long term forecasting two of the methods end-use and econometric approach are used. For short term load forecasting various method which include the similar day approach, various regression models, time series,, artificial neural networks, fuzzy logic, etc. are widely used to make predictions. The implementation of the most appropriate statistical tools can lead to highly accurate load forecasting methods.

Medium- and Long-Term Load Forecasting Methods

The end-use models, econometric models, and their combinations are the most often used methods for medium- and long-term load forecasting.

End-use models

End-use models are based on predicting the consumer electricity load demand by using direct and user information such as number of appliances, age of appliances, size of household, age of households .However, it is sensitive to the amount and quality of end-use data. End-use forecast requires less historical data but more information about customers and their equipment.

Econometric models

This kind of forecasting model combines statistics with economics. Using this method we try to estimate the relationships between consumption which is considered as a dependent variable and the factors affecting it. The relationship is established using the method of least squares. The problem faced by this model is the over weighing of outlier data points as a least square approach is followed. Moreover the model suffers from the problem of multicollinearity as this model tries to establish a linear relationship between the dependent variables and the factor affecting it ignoring the interrelationship among the parameter themselves.

Short-term load forecasting methods

For short term load forecasting a large variety of statistical methods and artificial intelligence networks have been used over the years.

Similar-day approach

In this approach the main basis is to search for in the historical load data for a day with similar characteristic as the day for which load is to be forecasted. The load of such a similar day is then considered as the forecast. By similar characteristics we mean similar temperature and weather condition, same day of week and so on. Sometimes even a linear combination of multiple similar day is used as the forecast.

Regression methods

One of the most widely used techniques in statistical analysis is regression models. In regression a model is established between the load forecast and other factors on which the load is dependent on. The problem arises when we have to specify different non-linear model to determine the best fit model.

Time series

The methods of forecasting using time series are all based on an assumption that the data has an internal structure such as autocorrelation, seasonal variation or a trend in them. The models explore such a structure and try to predict value based on the structure. It has been the most widely used technique for any form of forecasting for decades. In particular, ARMA (autoregressive moving average), ARIMA (autoregressive integrated moving average), ARMAX (autoregressive moving average with exogenous variables), and ARIMAX (autoregressive integrated moving average with exogenous variables) are the most often used classical time series methods.

Fuzzy logic

Fuzzy logic is more or less a generalization of the Boolean logic used in digital circuit design. The Boolean system takes values as 0 or 1. In fuzzy logic, the input is assigned with certain qualitative ranges. For example, the input may be classified as “low”, “medium” and “high”. Fuzzy logic enables the user to (logically) deduce outputs from such fuzzy inputs. While using fuzzy logics you don’t need to have a mathematical model mapping inputs to outputs, neither do you need to have precise (or even noise free) inputs. After the processing of fuzzy inputs, a defuzzification may be used to produce the desired outputs.

Support vector machines

Support Vector Machines (SVMs) are a bit more recent techniques for problems involving classification and regression. Support Vector Machines does a nonlinear mapping (with some specific functions called kernels) of the data into a higher dimensional space which is also called feature space after which simple linear functions are used to create linear decision boundaries in this new feature space. The problem here is to choose kernel unlike Artificial Neural Networks, where one has to decide the architecture of the network .

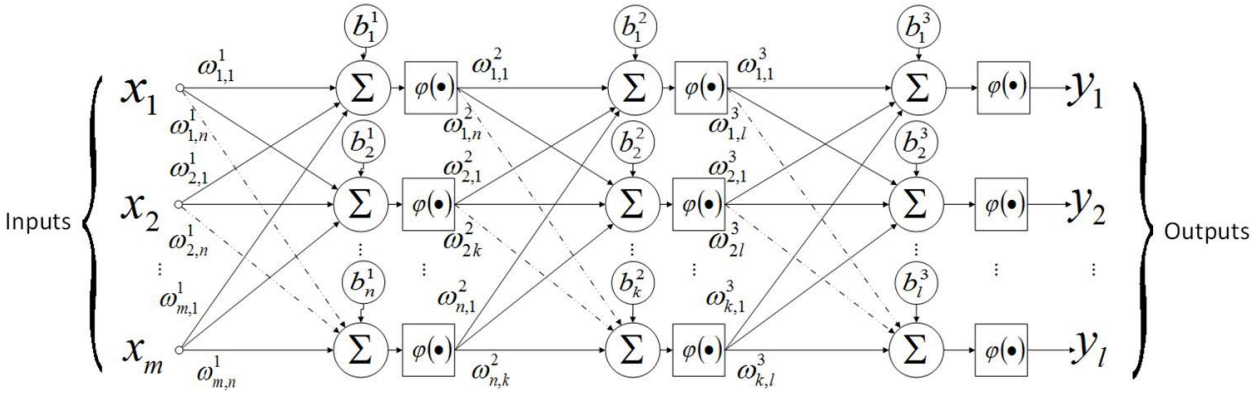
Neural networks

The us of neural nets for load forecasting has been widely studied since the 1990s. Neural networks are basically electrical circuits which have a capability of non-linear curve fitting.the output which is generated by a neural network is a linear or a non linear combination of its input.The most widely used neural network architecture for load forecasting is back propagation. Back Propagating involves continuous function valuation and supervised learning.

The Model

Architecture of Neural Net :

- ANNs consist of artificial neurons. Each neuron has a node denoted by circles and connections to other neurons denoted by lines.
- It consists of input layer which receives input signals, some hidden layers for processing and output layer which gives output.
- Each neuron in a layer is connected to every neuron in its adjacent previous and next layers.
- Each neuron has two processes taking place in it:
    - Algebraic addition of inputs from previous layer multiplied by their corresponding weights.
    - Operation of the activation function on the computed output from the previous step.



Data Acquisition and Data Pre-processing

We selected the country Belgium to train our model. Reasons:

- Belgium is a small country and with lesser population.

- Weather conditions could be assumed uniform throughout the region.
- Availability of EV sales data.
- Availability of Electric Load Data for every 15 minutes interval.

We gathered the Electric Load Data from a European website named **ENTSOE** for a period of 2.5 years from 1 January 2016 to 31 May 2018. The weather data for the same duration was collected from **Dark Sky API.** This site provided the stats for Belgium in json format. We coded the requesting link that called the API for over 1000 times with a regular time update. The acquired data then was converted to Excel sheets from where it was imported as Python data frames using the library: Pandas.

We considered following factors which could have an impact on load:

- **Weather:** Cloud Cover(1) (to capture household solar panel effect), Apparent Temperature(1), Humidity(1), Wind speed(1).
- **Time:** 15 min interval(4), Day(7), Hour(24), Month(12), Year(4).
- **Others:** EV(1), Holidays(1), Population(1).

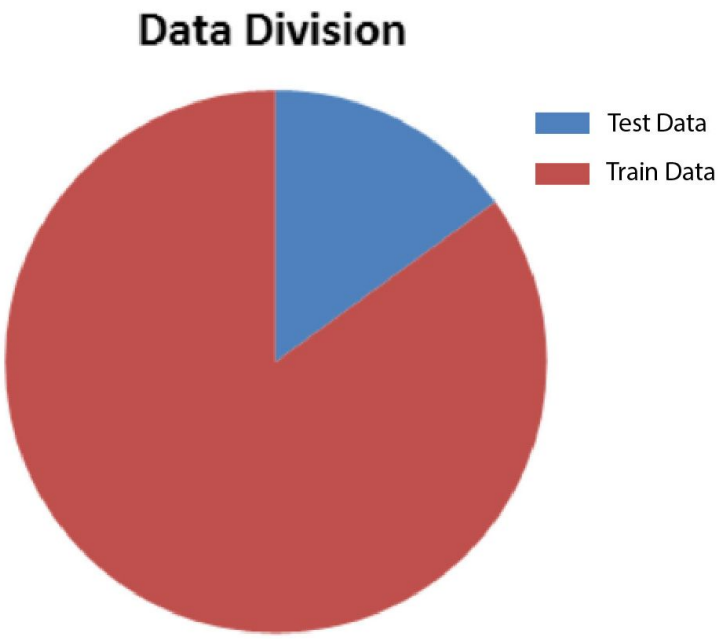
(Note: The numbers in the brackets shows the number of input neurons consumed by a particular factor).

This makes it a model with 58 input neurons.

Division of Data: Train + Test

We took 2.5 years of data in our model having total of over 880 days with each day having 96 data points. Hence, we had over 85000 data points.

Generally, for most of the neural net models, the ideal ratio is about 70% for training and 30% for testing. But, due to large number of data points in our model, we used about 70% as our training data ,20% as development data and 10% as testing data.



The data was divided randomly into three classes Training Data , Development Data and Testing Data respectively with the help of train-test split function present in the scikit-learn library of python.

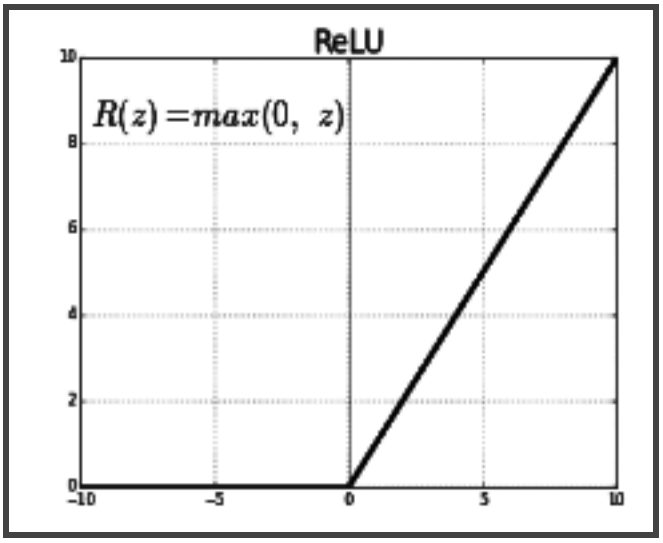
Learning Processes

Inputs

After going through a lot of papers and other research works, we finally narrowed down our inputs to: Weather, Time, Electric Vehicles (on road), Holidays and Population. Moreover, the input layer has 58 neurons description of which has been provided above. To compensate any loses/ add ons, a bias term had been allocated to each neuron which accounted for any further error.

Activations

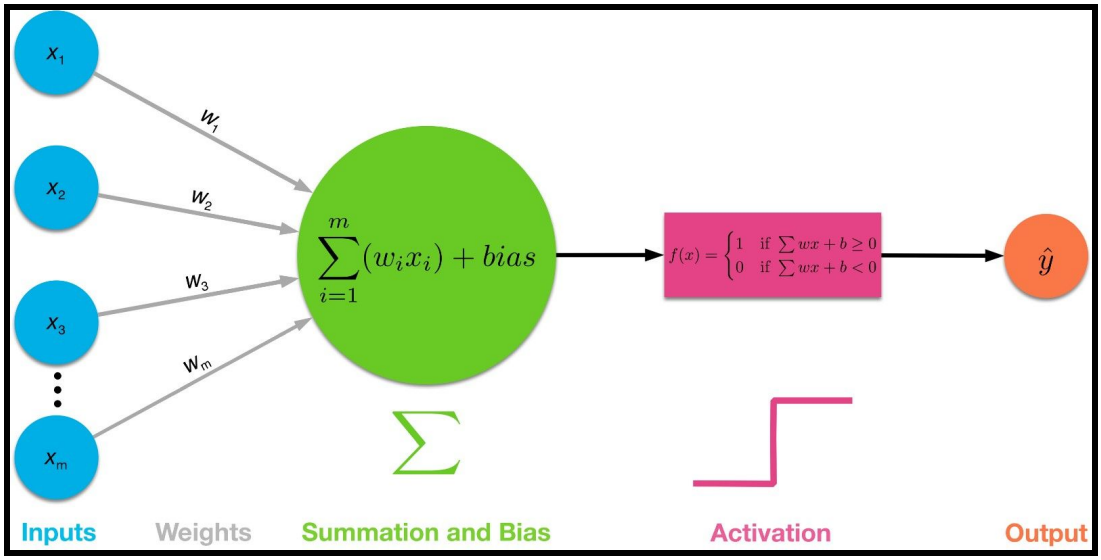
It defines the output of a node for a given set of inputs in a neural network. Its purpose is to bring non-linear properties to the neural network.
Example, sigmoid function, hyperbolic tangent function, rectifier linear unit(ReLU) function, etc.
We used ReLU activation function in our model.



We chose ReLU as our activation function as it suites to most of the problems and other activation functions like sigmoid and hyperbolic tan have very small portion of x-axis where they have a derivative other than zero thus making it computationally weak for the backward propagation step where derivative is of high importance. But ReLU solved this problem given it's derivative is 0 $\forall$ x<0 and 1 $\forall$ x>0.

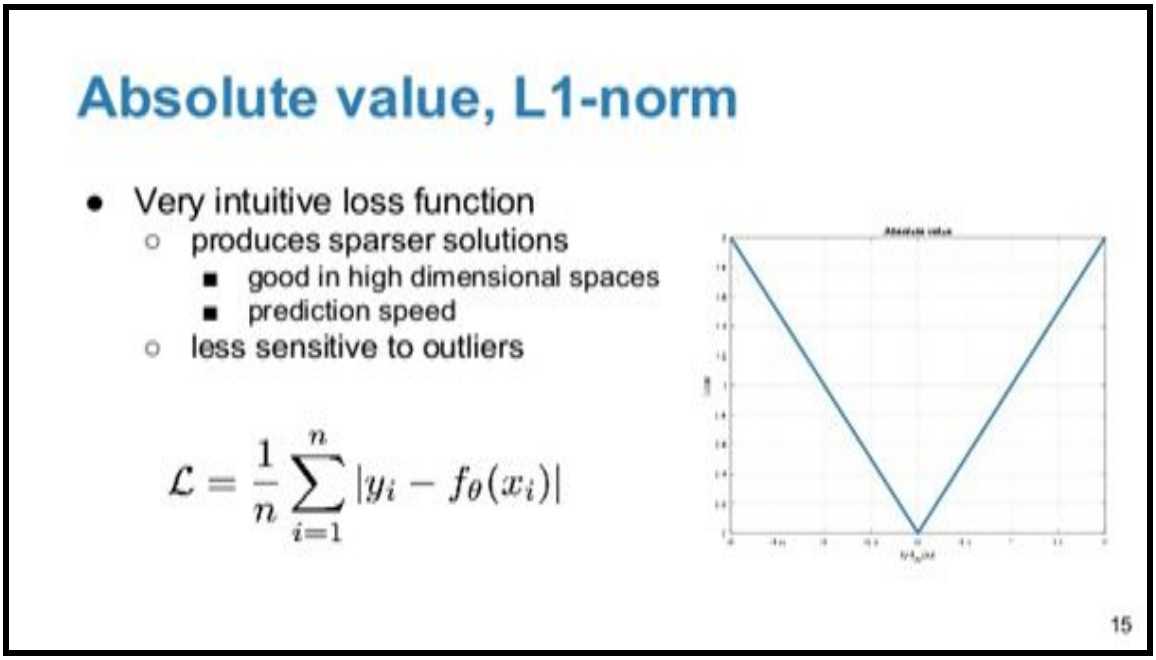
Forward Propagation

The learning of neural nets consists of two steps – Forward and Backward Propagation. Random values of weights and biases are initialized and then used to calculate the output with the help of given inputs. The signals travel from input layer to output layers. Thus, it is named forward propagation.
The algorithm can be seen in the figure below.



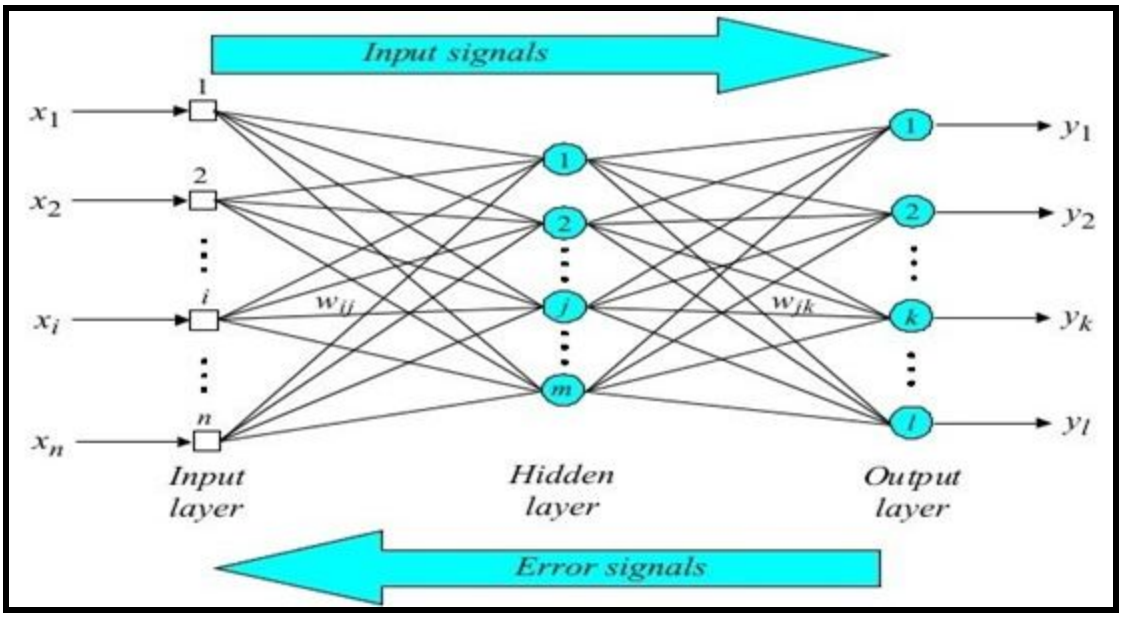
Cost Function

The cost function we used is mean absolute error function.
The properties of cost function can be seen in figure below.



Backward Propagation

Backward Propagation is used to calculate a gradient descent of cost function. This algorithm is used to minimize the cost function by updating weights and biases which are done by signals sent by output neurons to hidden layers and then to input layers, that is backwards. Thus, it is named Backward Propagation.



Updating Parameters

On the completion of each epoch, ours parameters (i.e. weights and biases) get updated. Updation takes place with specific learning rate and in the direction of decreasing cost / loss function which is calculated by obtaining the derivatives of the cost function w.r.t to weights and biases.

Optimizations & Improvements

Optimization is a highly important part of any machine learning model as it helps to refine the predictions and makes the model more stable towards changes. Unfortunately, several times this part is ignored by people thus leading to inefficiencies.

Some of the notable benefits of incorporating optimization techniques are:

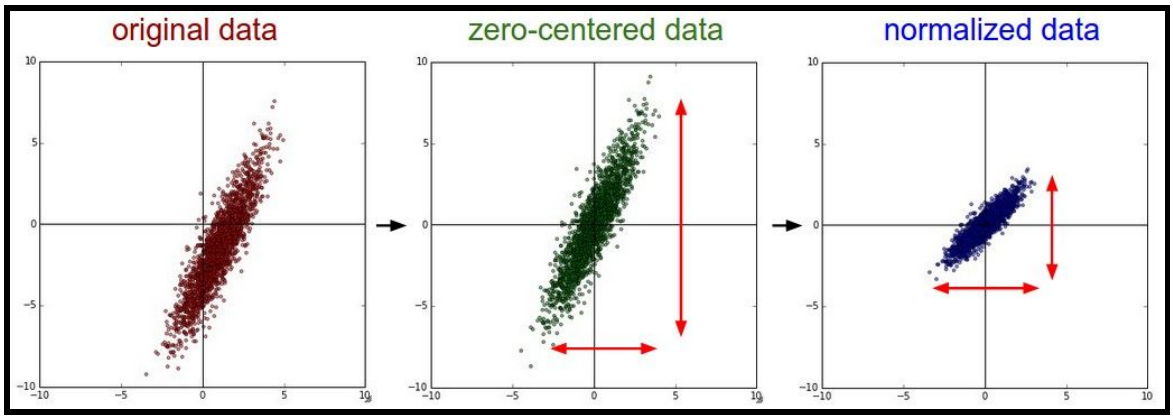
- Increase Efficiency
- Increase Accuracy
- Increase Adaptability
- Increase Learning Ability

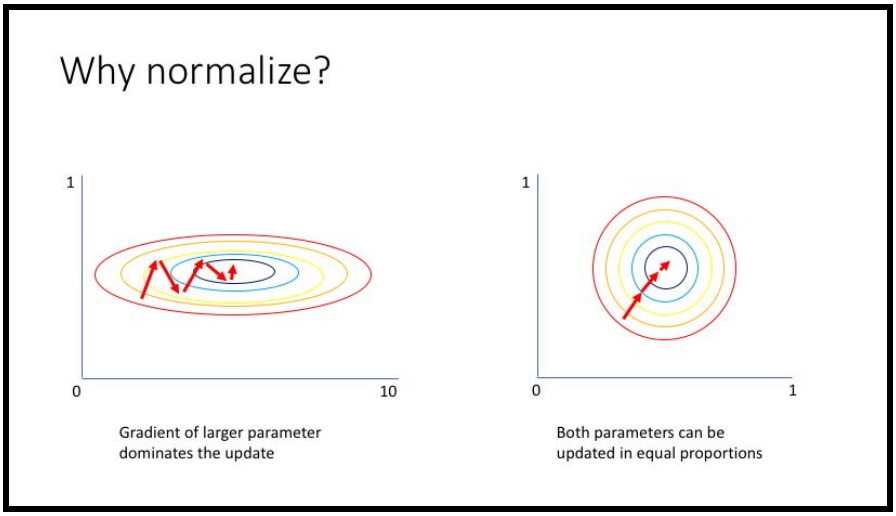
We have used several such techniques in our model to improve its practical performance. These techniques are briefly discussed below.

Normalization:

It is a technique that standardizes the input data for the machine to be fed with so that it can process it more easily. This is done by tuning mean to be 0 and variance to be 1.

Moreover, this is a good practice to minimize the effect of outlier data points.





The benefits of normalization are:

- Better Efficiency
- Better Adaptability

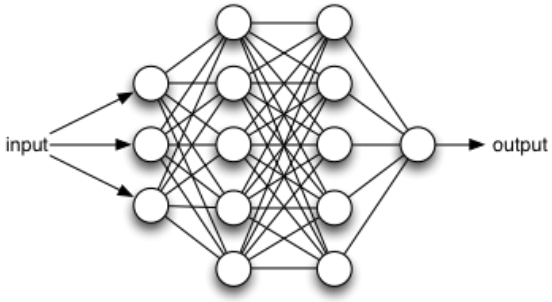
Dropout Regularization:

Regularization is process through which the variance of the output is minimized so as to avoid the problem of overfitting.
There are majorly two regularization techniques namely:

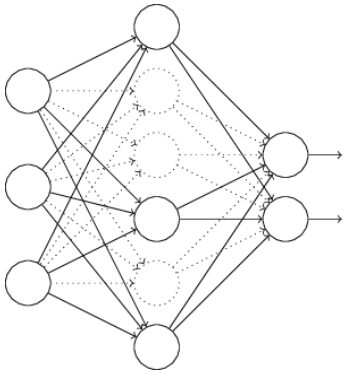
- L2 Regularization
- Dropout Regularization

We chose Dropout Regularization for our model as it gave us better results in terms of time and accuracy.

Dropout involves dropping out random neurons from each hidden layer (but not from input and output layer) of the model during each iteration. This forces other neurons to make better predictions even in the absence of a few neurons from the layer making them independent of each other.



Before Dropout



After Dropout

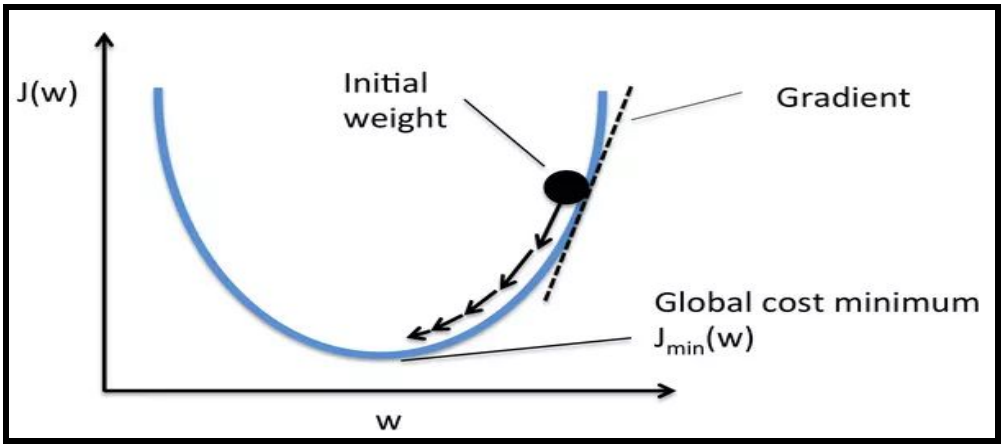
The benefits this regularization technique are:

- Better Learning Ability
- Better Adaptability

Learning Rate Decay:

This is technique which aims at minimizing the time taken by gradient descent to reach the optimum point. Initially, the learning rate is kept on higher side and then gradually decreased with the increment in the number of iterations. As the gradient descent reaches near the optimum point it becomes very small so as to not it over.

In our model the initial rate was kept at 0.01 and then decreased in a inverse relation with the number of epochs. It should be noted that the learning rate and it's decay rate can be changed as and when required.



The benefits of using decaying learning rate are:

- Better Efficiency
- Better Accuracy

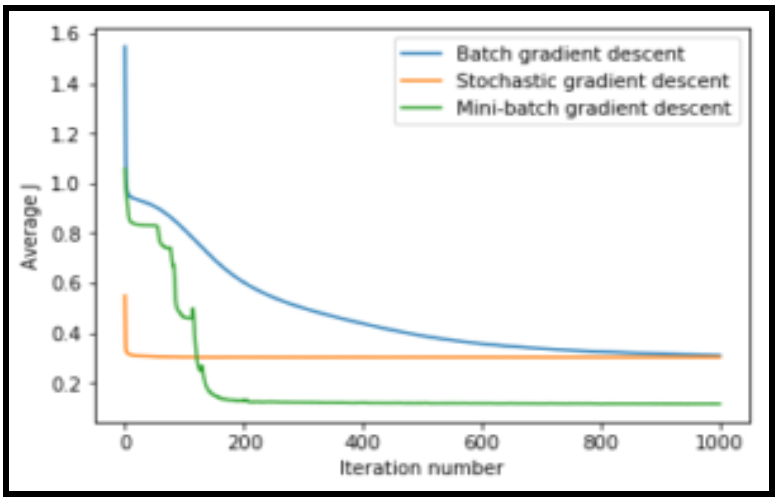
Mini Batch Gradient Descent:

It has been observed that feeding the machine with the whole dataset all at once leads to larger computation time and somewhat poor prediction. Thus, we used mini batch gradient descent to resolve this problem.

This method allows the machine to pick up data in small batches and train on them individually rather than on the whole dataset at once giving it the opportunity to learn several times in a single epoch.

Not only this, this method enables the machine to learn even from small datasets improving its learning ability on different scales of datasets.

We kept the mini batch size equal to 256 (though sizes like: 128, 512, 1024 also work fine).



The benefits of this method are:

- Efficiency
- Accuracy
- Learning Ability

ADAM Optimization:

ADAM Optimization has been the backbone of the whole optimization process.

Based on the principle of exponentially moving averages, ADAM is a combination of 2 methods:

- **Gradient Descent with Momentum:**

Maintains a per-parameter learning rate which results in better performance on problems with sparse gradients

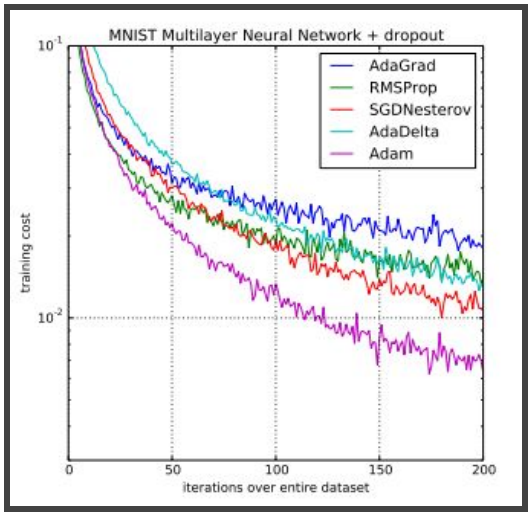
- **Root Mean Square Propagation:**

Maintains a per-parameter learning rates which are adapted based on the average of recent magnitudes of the gradients for the weights thus giving more importance to more recent data and happenings

A rough mathematical algorithm for this optimization is given below:

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

A rough estimate for the performance of the model with ADAM and with other techniques is also given below:



The benefits of using ADAM Optimization are:

- Better Accuracy
- Better Learning Ability
- Better Adaptability
- Better Efficiency

Hyperparameter Tuning:

Majorly, the hyperparameters of our model are:

- Learning rate (Decaying, Initial Value: 0.01)
- Number of epochs (Value: 50)
- Number of hidden layers (Value: 1)
- Number neurons in each hidden layer (Value: 210)

While a decaying learning rate was adopted, other hyperparameters were finalized by trail and testing the model again and again and reaching an optimum conclusion (values for each hyperparameter).

Output Of The Model

After the successful completion of the model training for various combinations of number of hidden layers, number of neurons in each hidden layer and other hyper-parameters and implementation of various optimization techniques the following result was achieved :

- Our model was successful in surpassing the required error of **5%** or less with a minimal theoretical error of **1.67%** This is the average error that we achieved on our test data which was a random set of **5000** data points between 1st January 2016 and 31st may 2018.

- Our model was successful in predicting for a future date with an error in the band of 0-5% with error going as low as **0.01%** on some times of the day.

- The model was successful in predicting the electricity load for any future date starting from 1st June 2018. The model accuracy decreased as we went too far away from the last trained date so in order to keep achieving the same accuracy training up to at least 15 days prior to the requested date should be done.

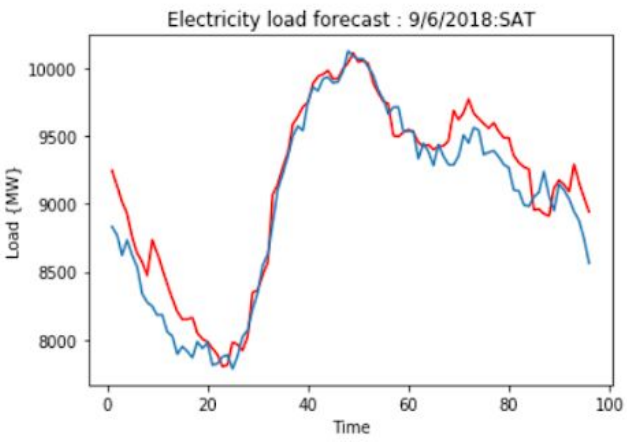
The current model requires the user to feed in the data for the region he/she needs the prediction in the 58 point format as explained above in the data acquisition part.The model will train on the fed data and give the forecast for that region.

Day - Wise Error and Mean Error

Shown below are the various load curves for a few selected days for the month of June 2018. The curve in red is the load curve based on our prediction and the curve in blue is the actual load curve for the date stated in the title. The X-axis is divided into 96 data points to represent 96 time intervals of 15-min for the whole day i.e (24*4) starting from 00:00 hours i.e midnight.

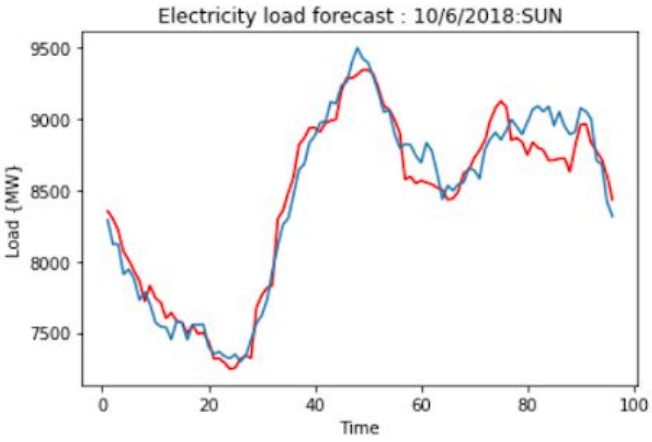
9/6/2018 : SAT

Our Prediction in Red



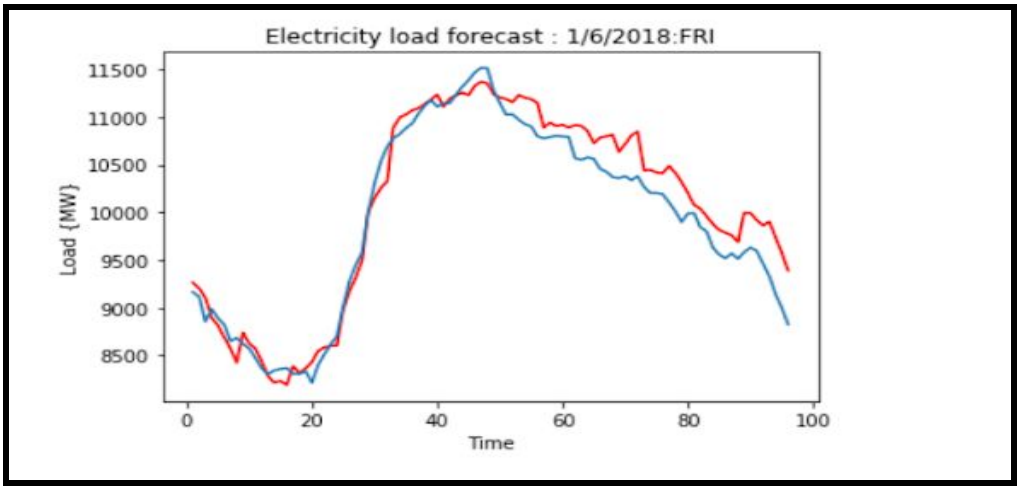
10/6/2018 : SUN

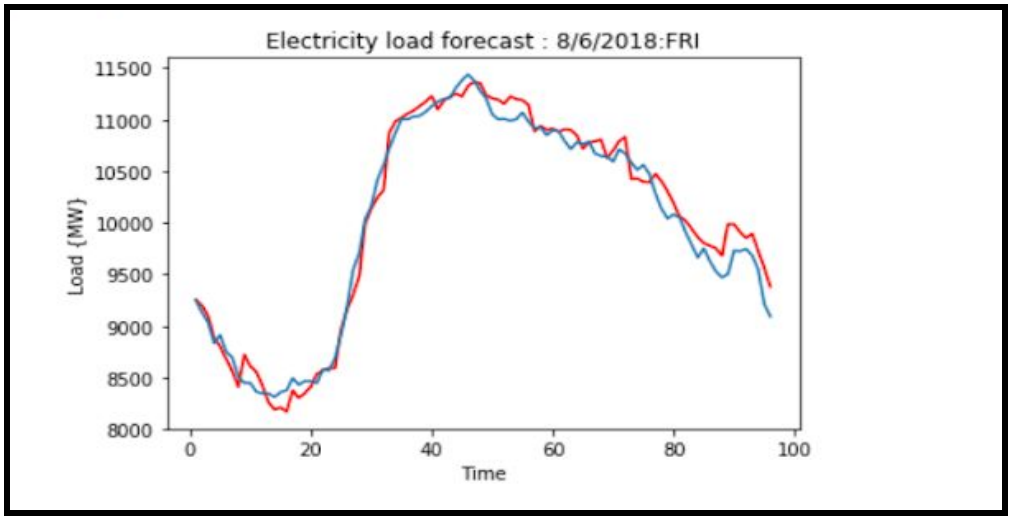
Actual Value in Blue

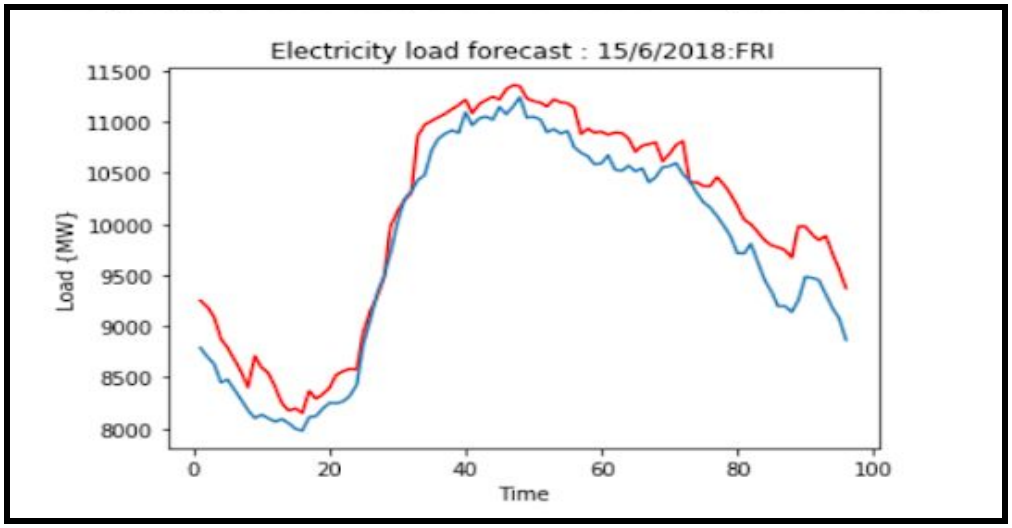


The next three graphs are for 1st , 8th and 15th June respectively with an average error curve for Friday drawn at the end of the curves. After analysing the graph it was found that the minimum error was found at around the **41th mark i.e. 11:15 - 11:30 A.M.** in the morning and the maximum error was found out to be around the **88nd mark i.e. 22:00 - 22:15 P.M** on the night of the respected day . This is the general trend for fridays in the month of June.

1st,8th,15th June 2018 (Friday) - Electricity Load Curves





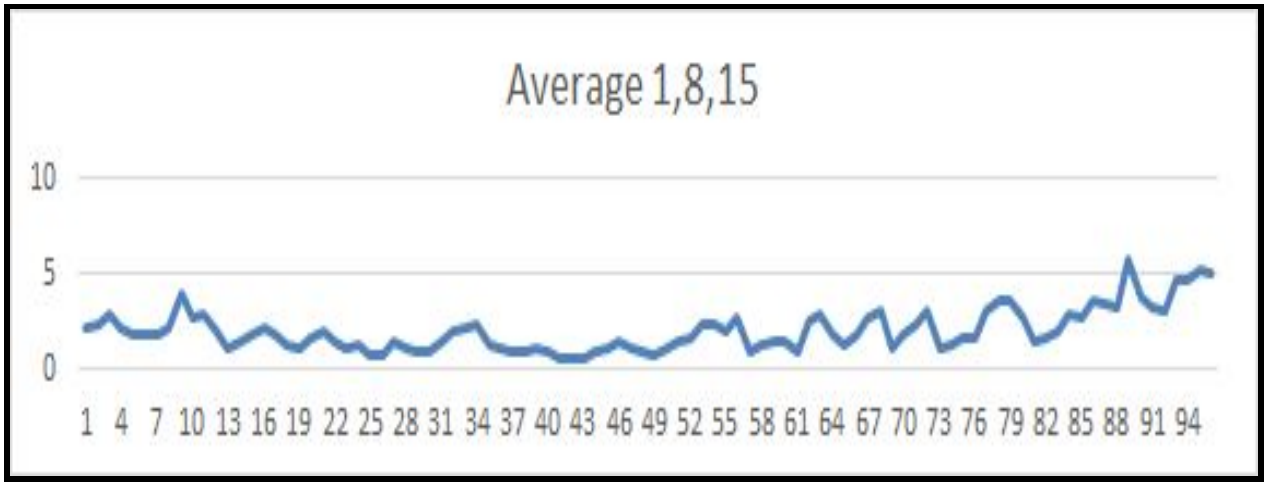


The average error was found out to be **2.06%** for the Fridays that came in the month of June 2018.

Also it was found that there was a steep rise during the 20-40 time intervals irrespective of the date and hence it was the portion of the curve where we achieved the best accuracy owing to the fact that machine trained itself quite well for this part because of its regular repetitive nature.

Table 1 below shows the various averages for fridays of June 2018. It can be clearly seen from the table that the more nearer the requested date to the last trained date is , the better is the prediction. The following table was made when the model was trained upto 31st May 2018.

Graph shows the average error for the first three fridays in the month of June :



The following table shows the error for the various days and the overall average:

TABLE - 1

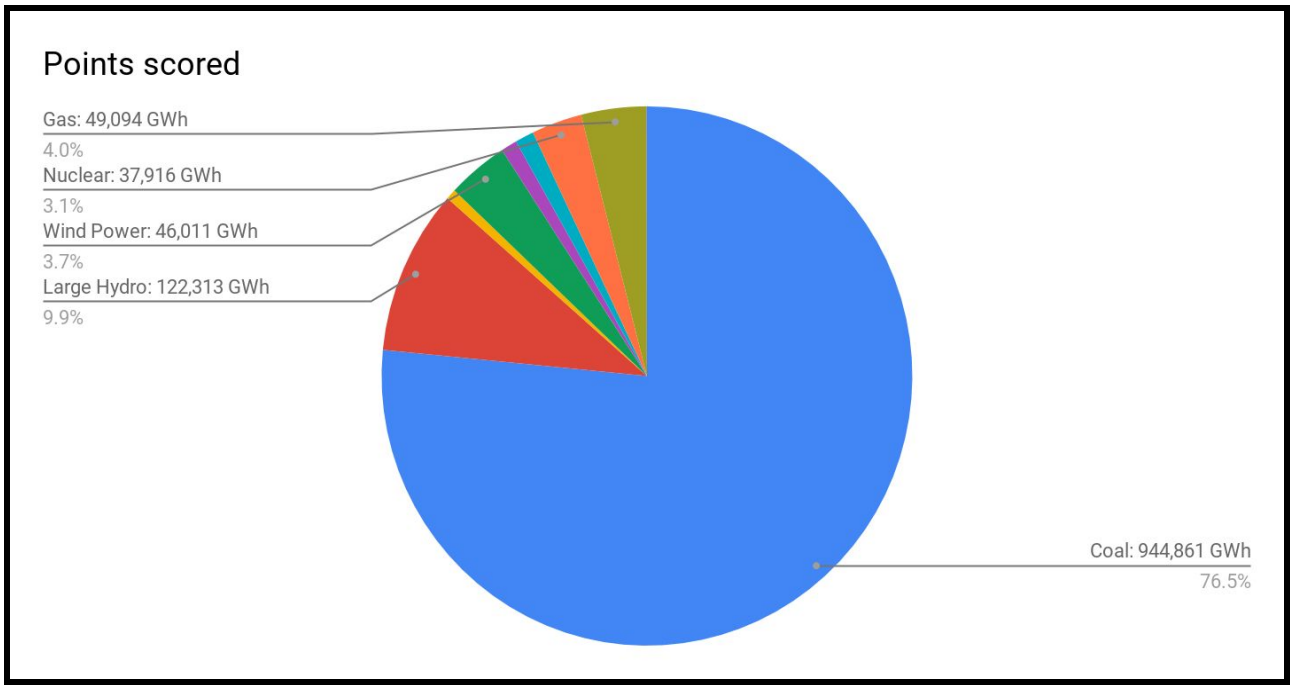
Description of Value Field	Average Daily %age Error
Average error on 1st June	1.99060022 %
Average error on 8th June	1.235388451 %
Average error on 15th June	2.958054162 %
Average of Average error	2.061347611 %
Max of Overall Average Error	5.71 %
Min of Overall Average Error.	0.54 %

Economic Significance

As on 30 June 2018 the National Grid capacity of India's electricity sector is around **344.00 GW**. Renewable power plants contribute to around **33.23%** of total installed capacity in India. During the fiscal year 2016-17, the gross generation of electricity in India by utilities was **1,236.39 TWh** and the total electricity generation (utilities and non utilities) in the country was **1,433.4 TWh**. The gross electricity consumption was **1,122 kWh** per capita in the year 2016-17. India is the world's third largest producer and third largest consumer of electricity. Electric energy consumption in agriculture was recorded highest (**17.89%**) in 2015-16 among all countries.

During the fiscal year 2016-17, the energy availability was **1,135.334 billion KWh** with a short fall of requirement by **7.595 billion KWh (-0.7%**) against the **1.1%** surplus anticipated (error of **1.8%**). The peak load met was 156,934 MW with a short fall of requirement by **2,608 MW (-1.6%)** against the **2.6%** surplus anticipated(error of **3.2%**).

Electricity generation by source in India in FY 2016-17



An **Electric Distribution Company** is responsible to transmit and deliver the electricity to a customer's home or business along the poles and wires in a given service area. It is of extreme importance or a Electric Distribution Company to forecast electricity load that would be required by the service area at a given time on a given day. The importance of forecasting is for several technical as well as economic reasons.

The electricity network has several generators (e.g. power stations, or perhaps a solar panel on your roof).If all these feed into the same network, they must be in phase i.e operating at same frequency or instead of providing power they would be taking it. The problem is, frequency can be difficult to control – if the exact amount of electricity being used i.e. in demand by the consumers is not matched by generation i.e. by suppliers who generate electricity and supply through a distributor company, it can affect the frequency of the electricity on the grid. For example, if there's more demand for electricity than there is supply, frequency will fall. If there is too much supply, frequency will rise. To make matters more delicate, there's a very slim margin of error.

If the there is a huge change in grid frequency due to unmatched demand and supply of electricity then there is a high chance of the electricity grid falling out resulting in a blackout to entire grid region as it creates a chain reaction resulting in fall out of all connecting grids. Thus to maintain frequency of grid it is important for not only distributors but also for generators to have accurate electricity load forecast

From the economic point of view for a distributor, electricity procurement cost differs widely from long term procurement and short term procurement. Long Term Procurement include Power purchase agreement (upto 25 years), OTC Licensed traders (3 months-3 years), Weekly contracts on energy exchanges. Short Term Procurement includes Intraday, Day ahead Market, Day ahead contract, Daily contract and Unscheduled Interchange i.e. Real time procurement.

The table below shows percentage of each procurement method used by distributor in total electricity procurement to meet demand.

About **4-5%** of procurement is short term. The reason for short term procurement especially Unscheduled Interchanges in accuracy of predicting accurate load. Average cost of energy procurement in long term to distributor is **2.80 ₹/KWh**. On the other hand average cost of energy procurement in short term is about **5.90 ₹/KWh**. Thus if load prediction accuracy is improved a distributor save **3.10 ₹/KWh** in procuring load in long term instead of short term.

So if the prediction accuracy is improved by even 0.1%, considering nation electricity demand for fiscal year 2016-2017,the annual load prediction error is reduced by **1.135334 billion KWh**. And procuring this energy in long term can help distributors save **3.10 x 1.135334** which is about **₹ 3.52 billion** **(**approx.**)**. This number is bit vague and inaccurate as a lot of assumptions and approximations have been taken to come to this figure but this figure definitely is suggestive of how much our nation can save by improving forecasting method and accuracy.

Conclusions & Recommendations

- The project assigned to us (to develop a model for short term load forecasting) was completed well in stipulated time i.e. within 6 weeks with commendable results and accuracy.

- Though the model is highly efficient, scope for improvement always exits with use of better techniques and more time investment. For instance, the training time can be reduced further if a more advanced processor is used and GPU support is also integrated for the processing.

- With increment in data, in terms of both quantity and quality, better performance can be achieved given the nature of technique (deep learning) that we have used to develop this model.

- In further expansion of the project one can work on building a user interface/ frontend for the model.

- Finally, it may be concluded that this model is ready for industrial usage and that too with pretty good accuracy.

References

- Prevalent & Past Methods:
  - Ciarán Lier: “Applying Machine Learning Techniques to Short Term Load Forecasting”
  - Ping-Feng Paia & Wei-Chiang Hong: “Support vector machines with simulated annealing algorithms in electricity load forecasting”
  - http://almozg.narod.ru/bible/lf.pdf

- Weather Data :
  - https://darksky.net/dev/docs
  - https://github.com/bitpixdigital/forecastiopy3

- Load Data For Belgium :
  - Entsoe- API

- National Holidays Data:
  - https://www.officeholidays.com/countries/belgium/index.php

- Population Data :
  - https://data.worldbank.org/country/Belgium

- EV Data
  - https://www.statista.com/statistics/698294/number-of-electric-passenger-cars-in-use-in-belgium/

- Economic Analysis :
  - https://powermin.nic.in/en/content/power-sector-glance-all-india
  - https://qz.com/1237203/india-is-now-the-worlds-third-largest-electricity-producer/
  - https://www.iexindia.com/marketdata/tradedetails.aspx

- Optimizations:
  - Diederik Kingma & Jimmy Ba: “Adam: A Method for Stochastic Optimization”
  - http://sebastianruder.com/optimizing-gradient-descent/index.html
  - https://medium.com/@nishantnikhil/adam-optimizer-notes-ddac4fd7218
  - deeplearning.ai

Glossary

- **Epoch** : A term that is often used in the context of **machine learning**. An **epoch** is one complete presentation of the data set to be learned to a **learning machine**.**Learning** machines like feedforward neural nets that use iterative algorithms often need many **epochs** during their **learning** phase.

- **Activation Function** : In artificial neural networks, the **activation function** of a node defines the output of that node given an input or set of inputs. A standard computer chip circuit can be seen as a digital network of **activation functions** that can be "ON" (1) or "OFF" (0), depending on input.

- **Cost Function** : A **cost function** is a measure of "how good" a **neural network** did with respect to it's given training sample and the expected output. It also may depend on variables such as weights and biases. A **cost function** is a single value, not a vector, because it rates how good the **neural network** did as a whole.

- **Hyperparameters** : In machine learning, a **hyperparameter** is a parameter whose value is set **before** the learning process begins. By contrast, the values of other parameters are derived via training.

- **Gradient Descent** : **Gradient descent** is a first-order iterative optimization algorithm for finding the minimum of a function.

- **Regularization** : **Regularization** is a technique used in an attempt to solve the overfitting problem in statistical models.

    - **Overfitting** : An **overfitted** model is a statistical model that contains more parameters than can be justified by the data. The essence of **overfitting** is to have unknowingly extracted some of the residual variation (i.e. the noise) as if that variation represented underlying model structure.

- **Learning Rate** : The **learning rate** is how quickly a network abandons old beliefs for new ones.