Multi-variate Time-series Load Forecasting using Deep Learning

by

Saadman Sakif Arnob (170021064)
Ahmed Syed Saqalain (170021018)
Najmus Sadat Sakib (170021017)

A Thesis Submitted to the Academic Faculty in Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Islamic University of Technology (IUT)
Gazipur, Bangladesh

May 2022

Multi-variate Time-series Load Forecasting using Deep Learning

Approved by:

Dr. Ashik Ahmed

Supervisor and Professor,
Department of Electrical and Electronic Engineering,
Islamic University of Technology (IUT),
Boardbazar, Gazipur-1704.

Date: ……

Declaration of Authorship

This is to certify that the work in this thesis paper is the outcome of research carried out by the students under the supervision of Dr. Ashik Ahmed, Professor, Department of Electrical and Electronic Engineering (EEE), Islamic University of Technology (IUT).

Authors

_____

Saadman Sakif Arnob
ID-170021064

_____

Ahmed Syed Saqalain
ID-170021018

_____

Najmus Sadat Sakib
ID-170021017

Table of Contents

List of Tables

List of Figures

List of Acronyms

XGBoost- Extreme Gradient Boosting
LSTM- Long Short-Term Memory
CNN- Convolutional Neural Network
MLP- Multilayer Perceptron
ML- Machine Learning
ADF- Augmented Dickey-Fuller (ADF) test
KPSS- Kwiatkowski-Phillips-Schmidt-Shin (KPSS)
ACF- Autocorrelation Function
PACF- Partial Autocorrelation Function
GBDT- Gradient-boosted Decision Tree
MSE- Mean Squared Error
MAE- Mean Absolute Error
RMSE- Root Mean Squared Error
MAPE- Mean Absolute Percentage Error

Acknowledgments

All thanks to Almighty Allah, the Most Merciful and Gracious. The authors are grateful to their supervisor, Dr. Ashik Ahmed, Professor, Department of Electrical and Electronic Engineering, Islamic University of Technology (IUT), for his constant supervision and useful ideas. This project would not have been possible without his invaluable guidance and friendly cooperation.

Abstract

To ensure the stable and reliable operation of a power system, load forecasting is required. Accurate forecasting leads to efficient dispatch, unit commitment, and energy security. Smart power management in the generating, transmission, and distribution network, as well as the accompanying energy demand, can be realized with accurate forecasting approaches. This paper analyses the short-term load forecasting of the Bangladesh power system. Various deep neural network models- XGBoost, LSTM, Stacked LSTM, CNN, CNN-LSTM, Time Distributed MLP, and Encoder-Decoder are used to forecast the load. The load is predicted based on previous load data and various features like temperature, Weekdays, Weekends, and Peak Business Hours are taken to ensure the accuracy of the results. This study reports the advantages and disadvantages of each model.

Chapter 1

1 Introduction

Forecasting electrical load is crucial for developing power system planning and operating strategies. Unit commitment and economic dispatch are dependent on accurate demand predictions. It allows an electricity supplier in determining which generators should run at what hours and at what levels to meet the demand. Maintaining the balance between load and generation is one of the most difficult tasks for the power system. Because it is uneconomical to store electric energy. Power should be generated on-demand. As a result, in order to provide proper planning and reliable operation of the power system, the customer's load requirements must be predicted in advance. There are three methods of load forecasting in general. Short-term load forecasting predicts load for the next few hours to a few weeks, midrange load forecasting for a week to a year, and long-term load forecasting for more than a year.

1.1 Motivation

Only a few reports have been released to estimate Bangladesh Power System's future load demand. Some reports are focused on specific regions [1]. There are reports on the island of Bangladesh [2]. But there is not any proper forecasting model for the entire country that is focused on Bangladesh Power System. Although some reports forecast the entire country, these are based on daily peak demand [3]. To develop a reliable power system and reduce cost per unit, it is essential to have a solid short-term forecasting model that is focused on the Bangladesh power system.

1.2 Objectives

Our primary goal is to forecast short-term load demand using multiple data and information. Many neural network models exist for forecasting future data. Our goal is to create a load forecasting model that is both efficient and optimum.

1.3 Problem Statement

Our goal is to find a model with the least amount of error using datasets native to Bangladesh. More relevant features, such as weather, temperature, and business hours, are being added to our model. This also aids our neural network model in detecting the trend in the load data curve, reducing inaccuracy. Furthermore, we will examine the advantages and downsides of several Neural Network models for our datasets. This will aid in the smoothing of any future research approaches.

Chapter 2

2 Literature Review

As modern society progresses, the demand for power and a variety of electrical loads changes regularly. Electrical load demand is influenced by a variety of factors, including weather, industrial processes, and human activities. Researchers have tried a variety of approaches to developing a prediction system for electrical load demands over the years. The utility industry's constant advancements and structural developments lead to load forecasting systems being highly important. Kalman filtering [4], dynamic linear filtering [5], and other key STLF techniques can be found in the previous literature. Previous studies have employed nonlinear models [6], as well as other load forecasting optimization techniques [5]. Support Vector Machines (SVMs) [7], artificial neural network (ANN) [8], ANN structure optimization using genetic algorithms [9] are also of interest to researchers. Several studies employed time series forecasting models like Auto-Regressive Integrated Moving Average (ARIMA) [10], Seasonal ARIMA (SARIMA), or Auto-Regressive Fractionally Integrated Moving Average (ARFIMA) [11] to forecast the load. The variance of available information and the projected time period heavily influence the model selection. However, because the ANN technique is primarily concerned with reducing experience risk by employing the empirical risk reduction principle based on the Statistical Learning Theory, it cannot provide faultless outcomes or quantitative analysis (SLT). SVM, on the other hand, makes use of the Structural Risk Minimization (SRM) method. SRM approach lowers the generalization error rather than minimizing the empirical error [12], [13]. Furthermore, the SVM regression method, a sophisticated forecasting methodology, is based on statistical learning theory and the structural risk minimization concept. Deep learning has recently become one of the most popular technologies in several fields of research. In contrast to shallow Deep learning is the process of stacking numerous layers of neural networks and relying on stochastic optimization to solve problems. Activities involving machine learning A varied level of abstraction can be achieved by using a different number of layers to increase learning ability and task performance [14]. In the field of sequence learning, the long short-term memory (LSTM) recurrent neural network (RNN), which was first described by Hochreiter et al. [15], has gotten a lot of attention. In several fields, such as natural language translation [16], picture captioning [17-19], and speech recognition [7], effective applications based on LSTM networks have been documented. In this study, it is aimed to implement several deep learning methods. There are comparison reports of two models such as ARIMA and SVM [20]. This research aims to compare more models and discuss the advantage and disadvantages of the models.

Chapter 3

3 Methodology

Over the years, numerous methods and models have been used to forecast or predict Electrical load data. The overall method that we've used can be summarized in figure 3.1. Firstly, historical Load data and weather data were collected. Then the data is passed through the Data Pre-Processing stage. After that, the data is divided into input and test Data. Then, suitable deep neural network model is selected and the data is fit into the model. Comparing the outcomes using selected evaluation methods, the best model is selected. The model is then improved by tweaking the model parameters such as numbers of input layers, hidden Layers, output Layers, activation function, loss metrics, optimizer learning rate, number of epochs and batch size.

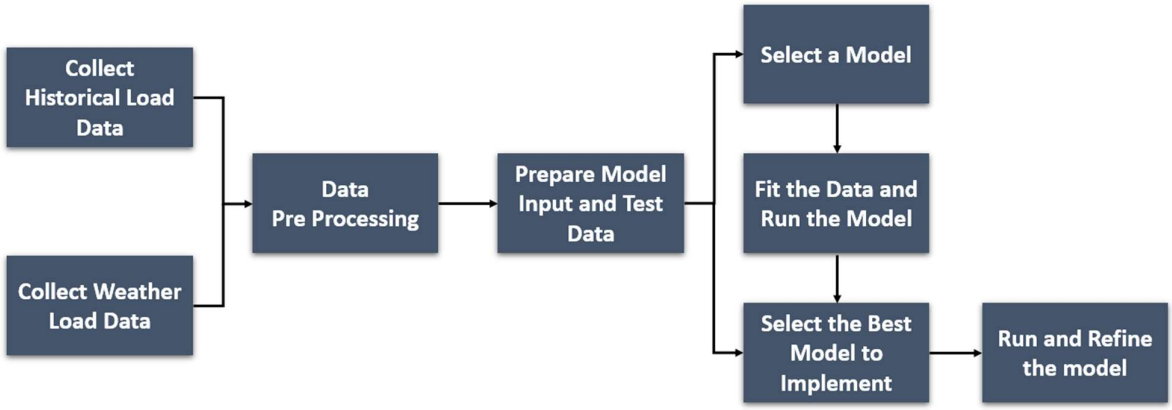

Figure 3. 2: **Basic Structure**

3.1 Data Pre-Processing

3.1.1 Data Acquisition, removing NaNs and duplicate values

Electrical Load data and weather data of every hour for the years 2017, 2018 and 2019 were used to train the models. Which equates to about 26280 individual data points. The data includes parameters such as date and time, maximum temperature, minimum temperature, total snowfall, sun hour, uv index, moon illumination, moonrise time, moonset time, sunrise time, sunset time, dewpoint, feels like temperature, heat index, windchill, wind gust, cloud cover, humidity, precipitation, pressure, temperature, visibility, wind direction, wind speed, location and load data. The weather data were collected from the https://www.worldweatheronline.com/ [21] and the electrical load data were collected DPDC.

Table 3. 1: Overview of Dataset

time	maxtempC	mintempC	sunHour	uvIndex	moon_illumination	DewPointC	FeelsLikeC	HeatIndexC	WindChillC	WindGustKmph	...	pressure	tempC	visibility	winddirDegree	windspeedKmph	loads	hour	weekday	month
2017-01-01 01:00:00+00:00	32	19	8.7	7	22	17	23	25	23	8	...	1015	20	10	293	5	4807.00	1.0	0.0	1.0
2017-01-01 02:00:00+00:00	32	19	8.7	7	22	17	23	25	23	7	...	1015	22	10	297	5	4525.00	2.0	0.0	1.0
2017-01-01 03:00:00+00:00	32	19	8.7	7	22	17	25	25	23	6	...	1014	23	10	300	5	4395.00	3.0	0.0	1.0
2017-01-01 04:00:00+00:00	32	19	8.7	7	22	16	22	23	22	8	...	1015	22	10	312	5	4211.00	4.0	0.0	1.0
2017-01-01 05:00:00+00:00	32	19	8.7	7	22	15	20	21	20	9	...	1015	20	10	324	5	4197.00	5.0	0.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2019-12-31 19:00:00+00:00	28	19	8.7	6	32	12	25	25	22	12	...	1015	22	10	312	8	8669.19	19.0	0.0	12.0
2019-12-31 20:00:00+00:00	28	19	8.7	6	32	12	22	25	22	13	...	1016	22	10	322	8	8494.27	20.0	0.0	12.0
2019-12-31 21:00:00+00:00	28	19	8.7	6	32	11	21	24	21	13	...	1017	21	10	332	8	8234.27	21.0	0.0	12.0
2019-12-31 22:00:00+00:00	28	19	8.7	6	32	11	21	24	21	12	...	1017	21	10	221	7	7521.27	22.0	0.0	12.0
2019-12-31 23:00:00+00:00	28	19	8.7	6	32	11	21	24	21	12	...	1017	21	10	111	7	6930.27	23.0	0.0	12.0

After data acquisition, the data is processed in order to be fitted into neural network models. Real world data may contain noises, missing values, duplicate values or data type which are not suitable for certain models. Data preprocessing is necessary to make the data suitable for machine learning model which helps to increase the accuracy and efficiency of the model.

3.1.2 Detecting and removing outliers

Outliers are data that significantly vary with the trend of the rest of the data. This can be caused by measurement error, human error or instrument error. Some common methods to remove outliers are:

- Boxplots
- Z-score
- Inter Quantile Range (IQR)

Boxplot is used to detect and visualize the existing outliers. After detection the Inter Quantile Range method is applied to remove the outliers in order to get correct median value of overall data. In figure 3.2 and figure 3.3, we can see the outliers in windspeed and wind gust parameters.

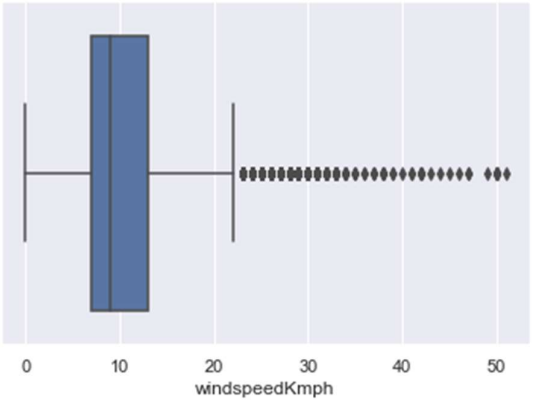

Figure 3. 3: **windspeed including outliers**

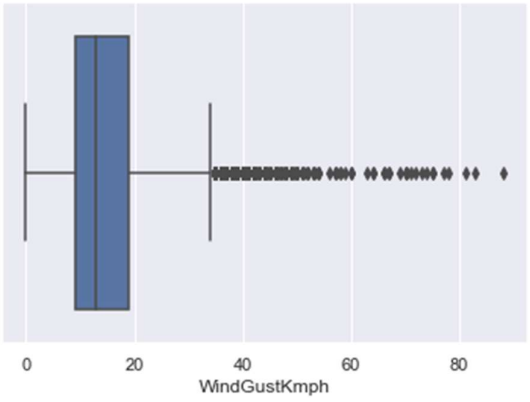

Figure 3. 4: **wind gust including outliers**

In Inter quantile range method the data which lies 1.5 times of IQR above Q3 and below Q1 are considered as outliers.

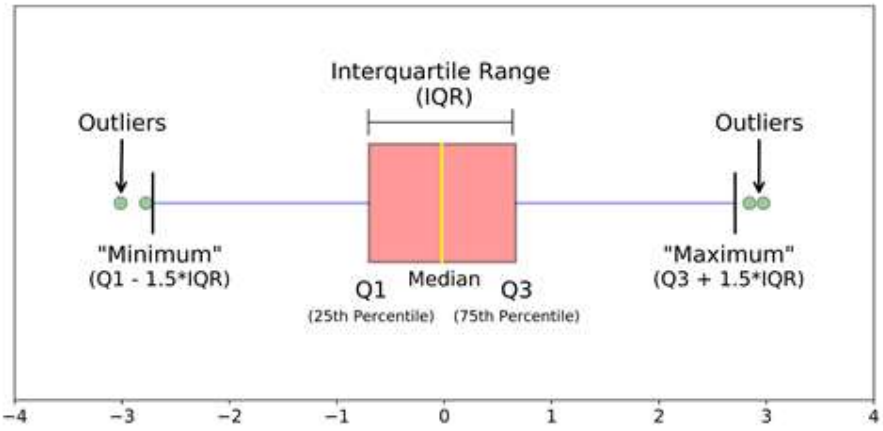

Figure 3. 5: Inter quantile range method

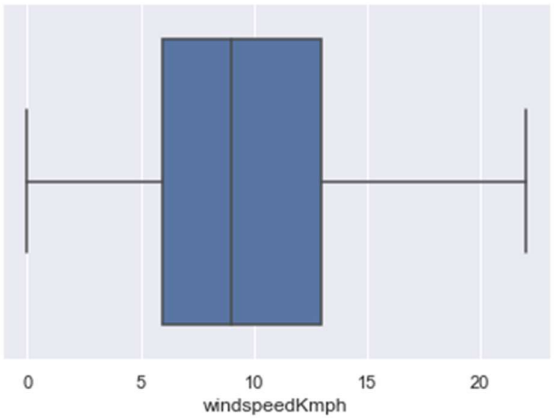

Figure 3. 6: **windspeed without outliers**

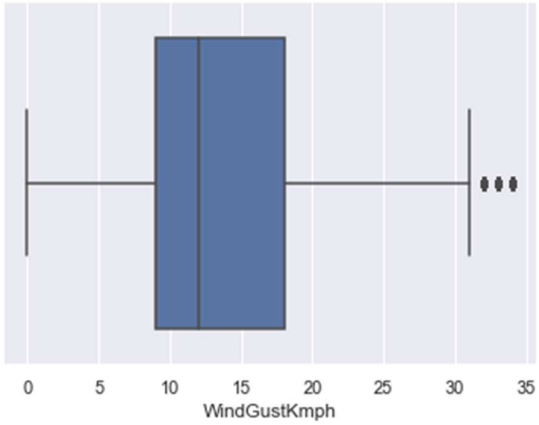

Figure 3. 7: **wind gust without outliers**

3.1.3 Data Visualization

To understand the trend and behavioral characteristics of the data that is being used, visualization is necessary.

In figure 3.7, the actual total load of first 2 weeks is shown. The electrical load follows a periodical trend oscillating in a daily manner.

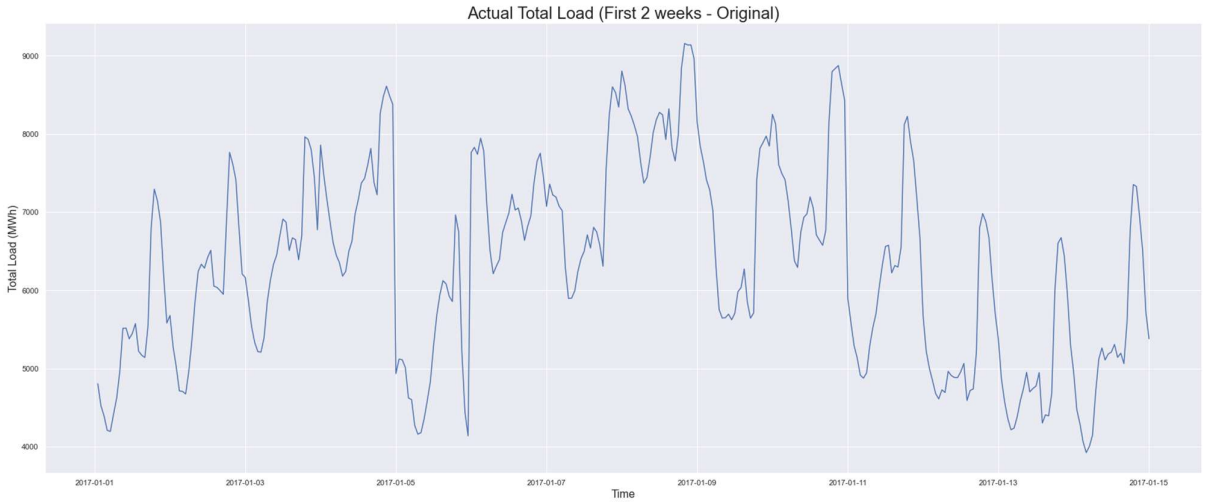

Figure 3. 8: **Actual load (First 2 weeks) vs time graph**

Here we've plotted the actual hourly loading from 27/05/2018 up to 10/06/2018, comprising two weeks' worth data. Numerous patterns and periodicities can be observed, including:

There is a weekly cyclic pattern as business days have higher loads, and weekends and especially Sundays, have lower loads. Loads peaks during the day and falls during the night. In some cases, the loads drop for a short time between 01:30PM and 04:30PM, most probably because of the lunch break between those hours.

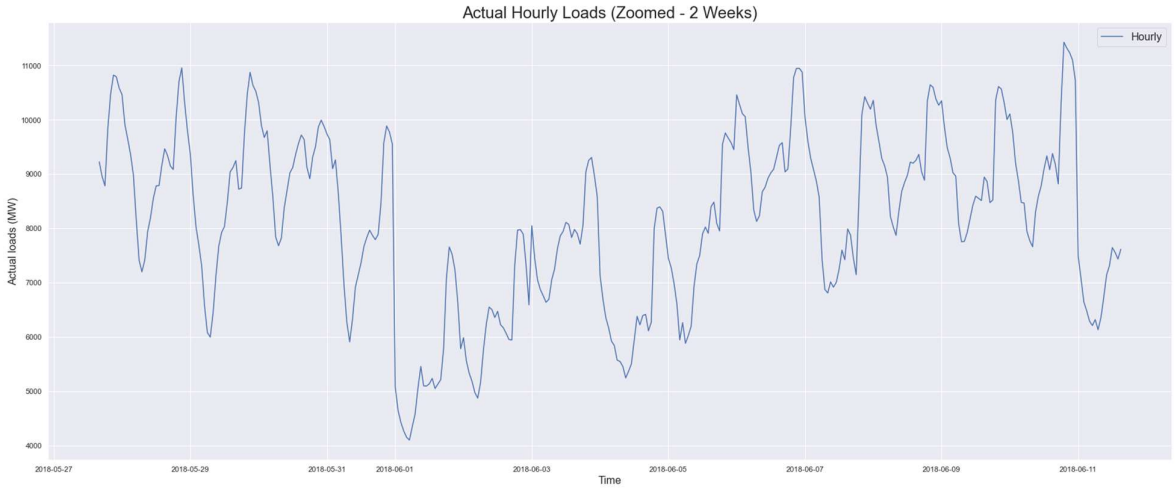

Figure 3. 9: **Actual hourly load (zoomed – 2 weeks) vs time graph**

In the figure 3.9, we can see the resampled version (monthly basis) of the actual loads and it's lagged 1-year samples. The spikes in certain month tell us that incorporating a new month feature would greatly boost our model's performance. The same spike would not be visible if the figure showed actual loads on a daily or weekly basis in its 1 year lagged timestamp.

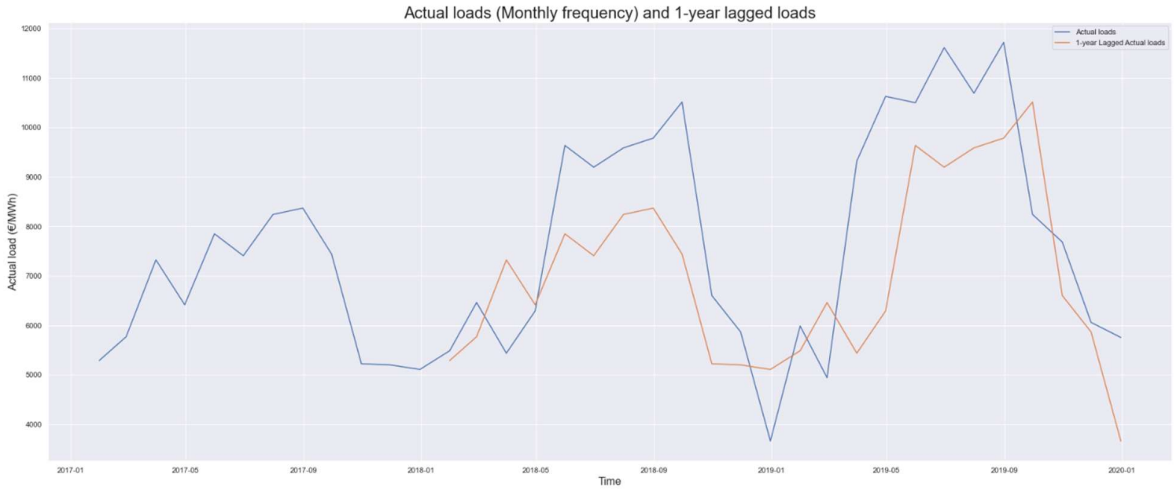

Figure 3. 10: **Actual loads and 1 – year lagged loads vs time graph**

Figure 3.10 shows the percentage change of hourly basis in the load. It shows how that load data changes compared to the previous hour.

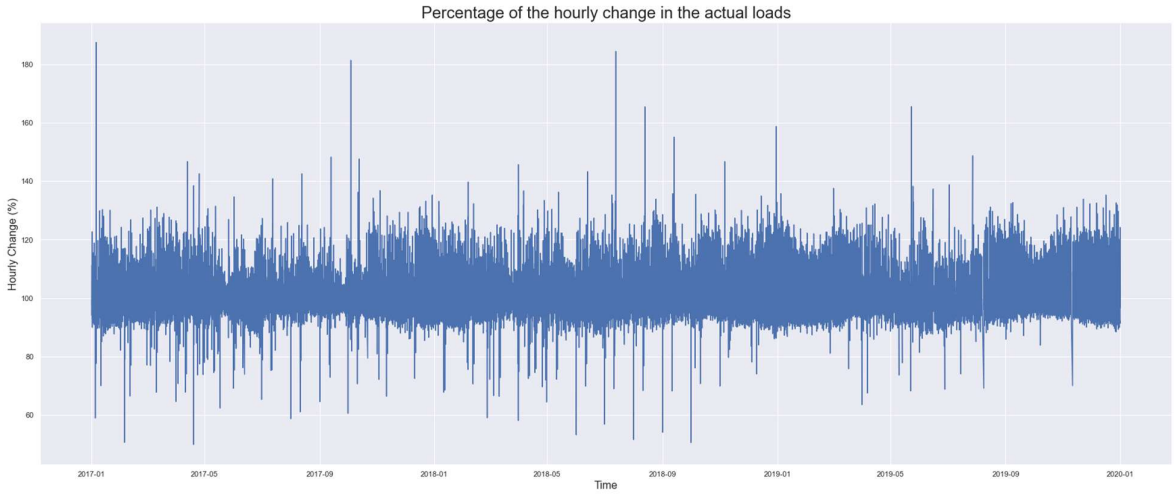

Figure 3. 11: **Percentage of hourly change in actual loads**

Figure 3.11 compares between the actual hourly load and weekly rolling mean.

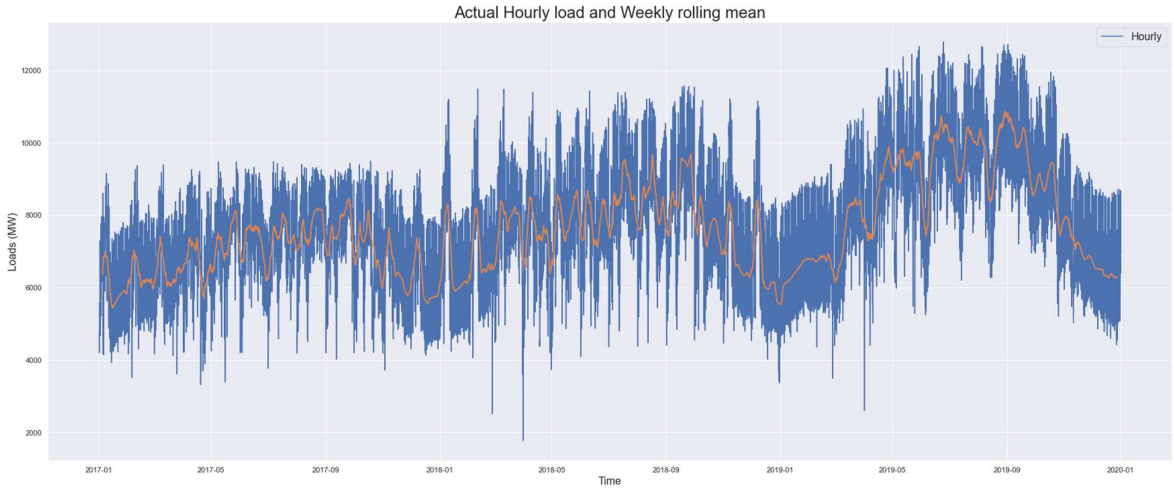

Figure 3. 12: **Actual hourly load and Weekly rolling mean**

3.1.4 Decomposition and stationarity tests

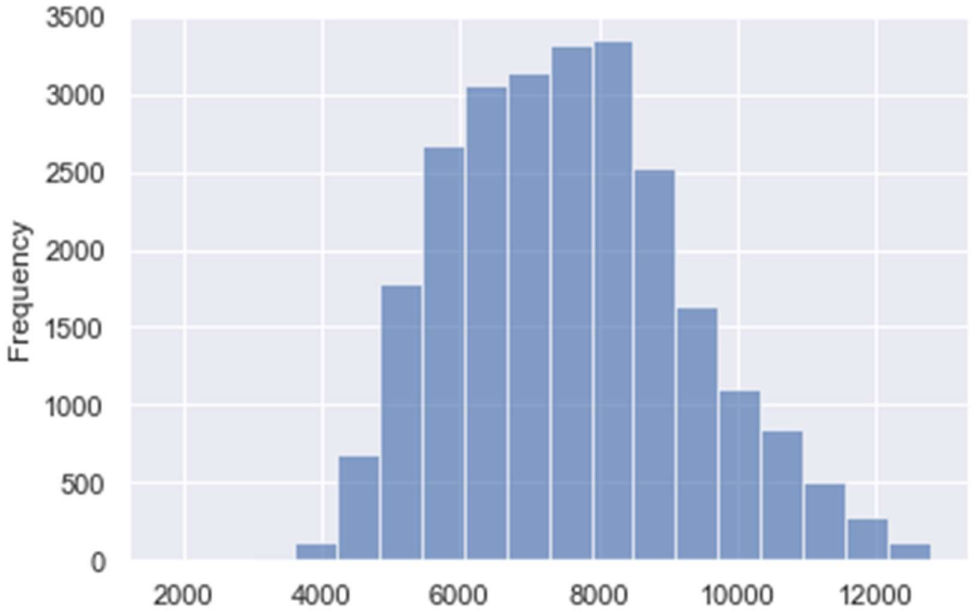

Figure 3. 13: **Load distribution over the range**

As we can see in the above figure, the actual load roughly follows a normal distribution, so standardization is feasible. We must also check if our time series needs transformation or not.

The time series of loads will be visually decomposed into its component time series, and then we will check whether they are stationary.

Time series have 3 main elements:

1. Seasonality
2. Trend
3. Noise

Figure 3. 14: **Additive decomposition**

As a general rule, most ML algorithm assume that input features and outputs have a static relationship. Having constant parameter values for inputs and outputs is called a static relationship. As a result, algorithms are most effective when their inputs and outputs are stationary. Time series forecasting does not follow this rule. Seasonality and trends are among the unique properties that can be found in distributions that change with time. Therefore, the mean and variance of the series fluctuate, leading to difficulties in modeling their behavior. Thus, it is essential to make a distribution stationary when forecasting time series.

We are performing two particular stationarity tests for our dataset. they are:

1. Augmented Dickey-Fuller (ADF) test
2. Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test

Augmented Dickey-Fuller (ADF) test: Through this test, we can find out how much the trend is dominating the time series:

- **Null Hypothesis**: For the null hypothesis to be rejected, the p-value < 0.05 because unit root is assumed. Based on this, we can say that the time series is stationary.

On our Time Series Data, this is the ADF test results:

Table 3. 2: ADF Test Results

ADF Statistic	-8.256870
p-value	0.00000
Lags used	48
Critical Value (1%)	-3.430610
Critical Value (5%)	-2.861655
Critical Value (10%)	-2.566831

Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test, a test that is not to be confused with ADF test as it does quite the opposite operations.

- **Null Hypothesis**: The test statistic must be greater than the critical values in order to reject the null hypothesis. Hence, a low p-value would be automatically generated if it is actually higher than the critical value. As a result, kpss will be greater than 5% if the p-value < 0.05.

On our Time Series Data, this is the KPSS test results:

Table 3. 3: **KPSS Test Results**

KPSS Statistic	6.453684
p-value	0.010000
Lags used	86
Critical Value (10%)	0.347000
Critical Value (5%)	0.463000
Critical Value (2.5%)	0.574000
Critical Value (1%)	0.739000

3.1.5 Autocorrelation, partial autocorrelation and Pearson correlation matrix

Autocorrelation Function (ACF)

Correlation between time series with a lagged version of itself. Observations at current time spots and observations at previous time spots have a positive correlation. It starts with a lag 0, which is a comparison of the time series with itself, resulting in a correlation of 1.

Partial Autocorrelation Function (PACF)

Each successive lagged term explains additional correlation. If we consider the two observations to be correlated with observations at other time points, the correlation between observations at two different time spots will be high. The partial autocorrelation at lag k is the autocorrelation between Xt and Xt−k that is not accounted for by lags 1 through k−1.

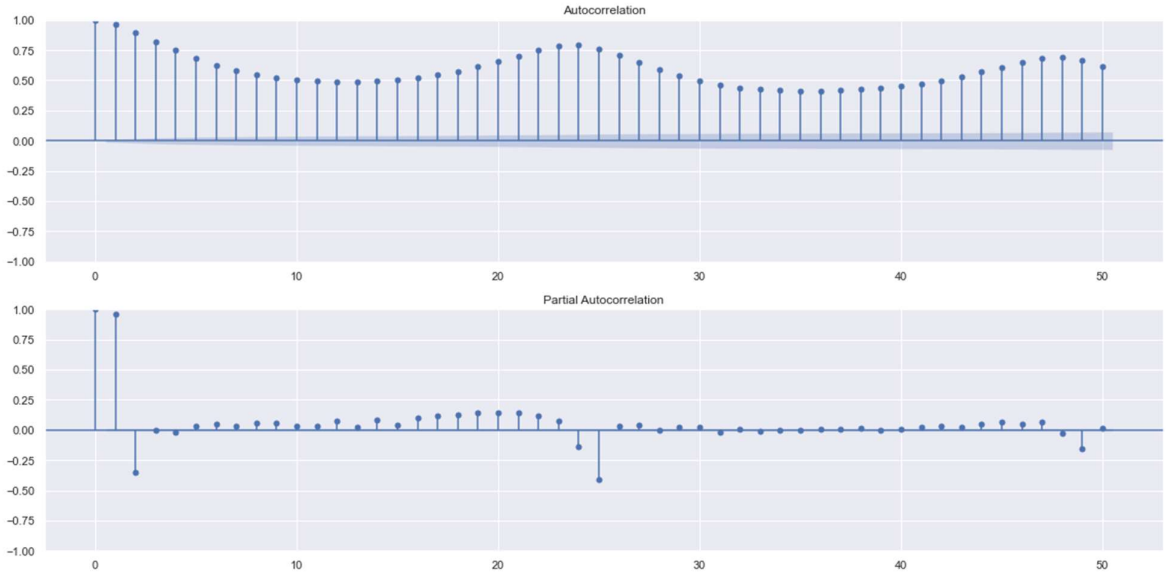

Figure 3. 15: **Autocorrelation and Partial Correlation**

Observations done at time steps t-1,2,24,25 demonstrate increasing partial autocorrelation and diminishing thereafter. For our models, we will use the first 25 values of each time series.

Correlation helps figuring out how one variable is dependent on another variable. In any data there might be numerous variables. Not all variables affect the outcome similarly. Pearson coefficient are linear correlation coefficients that capture distinct degrees of probabilistic dependence but not necessarily causation. In figure 3.15, Pearson correlation matrix is used to observe the features that are highly correlated to each other.

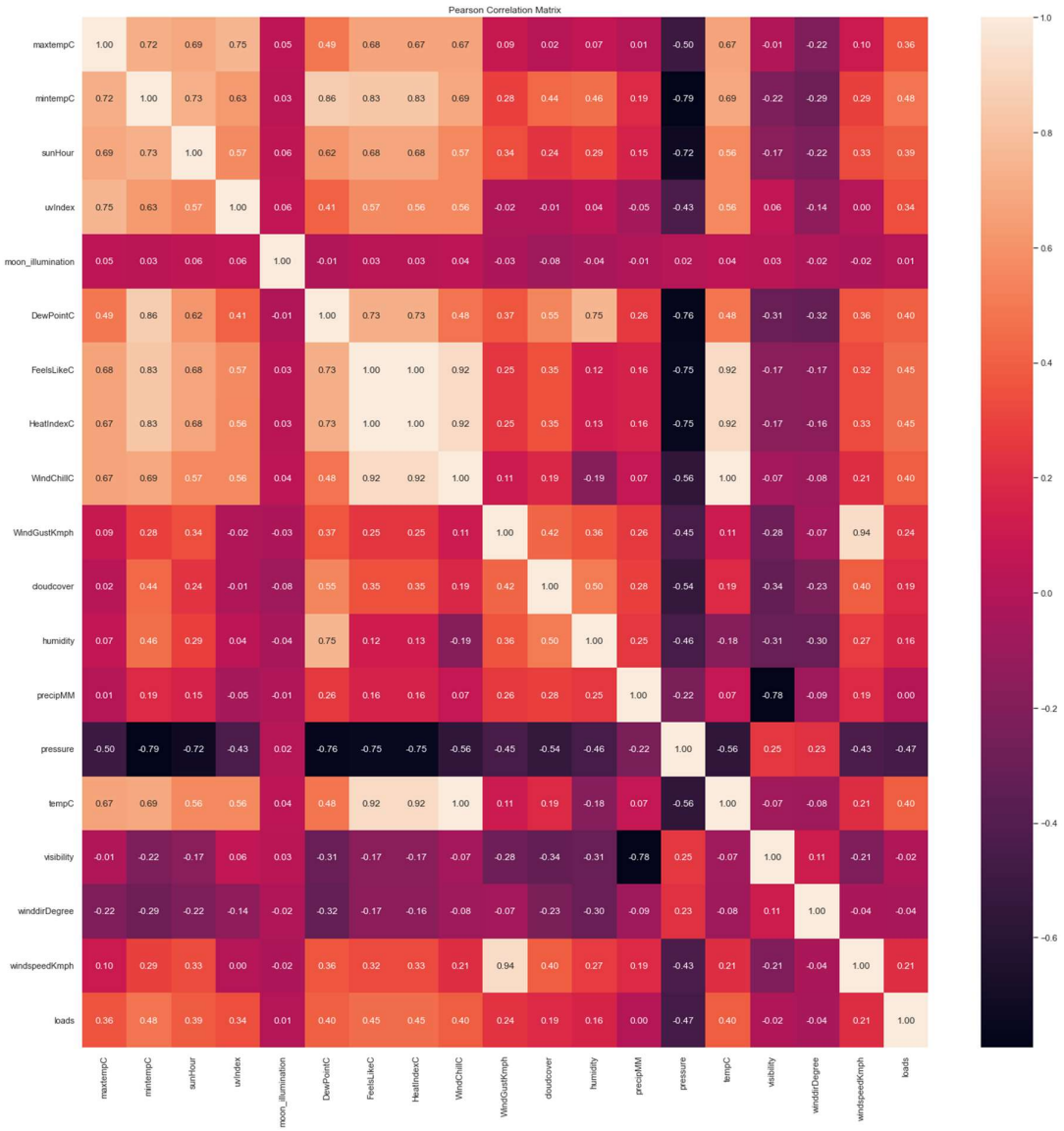

Figure 3. 16: Pearson Correlation matrix

Table 3. 4: **Pearson Correlation Coefficients**

Feature	*Highly Related Feature(s)*	*Correlation Coefficient*
maxtempC	uvIndex	0.751072
mintempC	DewPointC	0.859587
	FeelsLikeC	0.831510
	HeatIndexC	0.829710
uvIndex	maxtempC	0.751072
DewPointC	mintempC	0.859587
	humidity	0.754922
FeelsLikeC	mintempC	0.831510
	HeatIndexC	0.998397
	WindChillC	0.920138
	tempC	0.920292
HeatIndexC	mintempC	0.829710
	FeelsLikeC	0.998397
	WindChillC	0.917135
	tempC	0.917182
WindChillC	FeelsLikeC	0.920138
	HeatIndexC	0.917135
	tempC	0.998566
WindGustKmph	windspeedKmph	0.935152
humidity	DewPointC	0.754922
tempC	FeelsLikeC	0.920292
	HeatIndexC	0.917182
	WindChillC	0.998566
windspeedKmph	WindGustKmph	0.935152

3.2 Feature Engineering

3.2.1 Feature generation

Electrical load depends on weather. After taking all the necessary weather data into account, it can be seen that electrical load also depends on features like weekday, weekend, and business hour. These parameters were generated and implemented in the data along with other weather features. We will also generate an important feature - the business hours. Due to lunch break in between, not all businesses follow the 9AM-5PM working day. Hours of operation are usually Sunday-Thursday, 9:30AM-4:00PM, and then 6:00PM-10PM. 'Temp_range' is a feature which is the difference between the minimum and maximum temperature ('temp_max') and is used in dimensionality reduction and thus acquiring new insights.

3.2.2 Feature Selection

Feature selection is the process of eliminating irrelevant features from a set of data. The number of input variables should be reduced to lower the computational cost of modeling and, in some situations, to increase the model's performance.

The dataset is then divided into Training set, validation set and Test set. Distribution is as follows:

- Training set: 70%
- Validation set: 20%
- Test Set: 10%

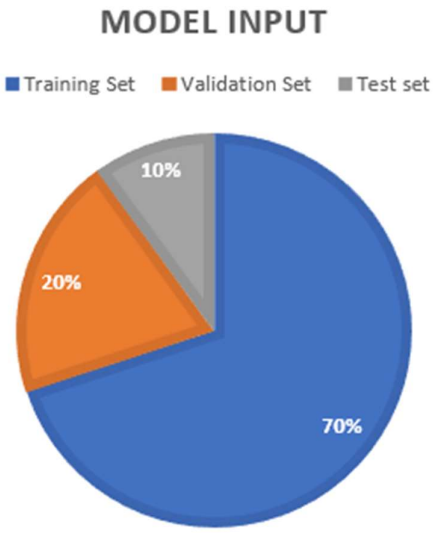

Figure 3. 17: **Model input divisions**

3.3 Implemented Models

3.3.1 LSTM

The hidden layer of the LSTM is recurrent. This layer contains special units which are called memory units. Memory cells contain information about the network's temporary state. Another part, which is called the gate, dictates the amount of information that can enter or exit through the memory blocks. Entry of input activation into the memory cell is determined by the input gate. There is also an output gate. Its job is to control the flow of cell activations. But these memory blocks cannot process continuous input streams. To resolve this issue, forget gate was added.

The forget gate resets the memory of a cell before it does its recurrent function. Furthermore, the contemporary LSTM architecture includes peephole connections between internal cells and gates within the same cell to learn accurate output timing. [22]

Let us consider an input sequence $x = (x_{1,\ldots,}x_T)$ to an output sequence $y = (y_{1,\ldots,}y_T)$. The network unit activations are calculated using the equations given below iteratively from *t = 1* to *T*.

$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}C_{t-1} + b_i)$	(3.1)
$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}C_{t-1} + b_f)$	(3.2)
$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1} + b_c)$	(3.3)
$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}C_t + b_o)$	(3.4)
$m_t = o_t \odot h(c_t)$	(3.5)
$y_t = \emptyset(W_{ym}m_t + b_y)$	(3.6)

Here, the W term dictates wight matrices. The b variable dictates bias vectors, σ is logistic sigmoid function and i, f, o are input gate, forget gate, output gate. m is the cell output activation vector. g and h denotes cell input and cell output. The network output activation function is denoted by $tanh$ and $\emptyset$ respectively.

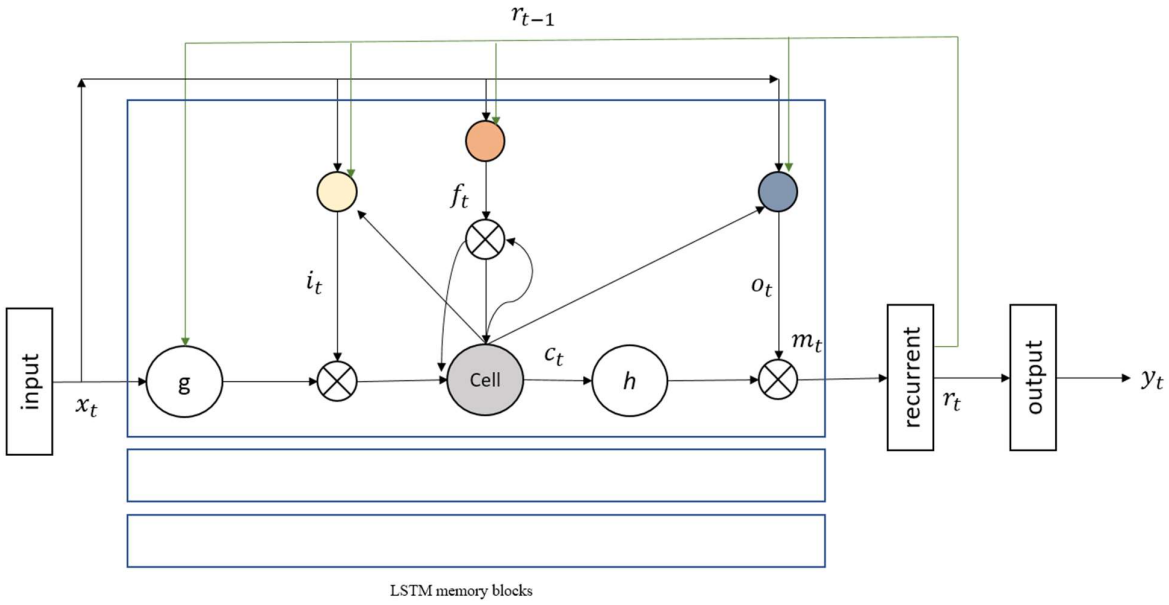

Figure 3. 18: **Structure of LSTM**

3.3.2 Stacked LSTM

The main reason for stacking LSTM is to increase the intricacy of the model. In a simple feedforward network, layers are piled on top of each other in order to generate a hierarchical presentation of input data. It is same for stacked LSTM. Between every input and out layers, several nonlinear mapping layers are used. The output of hidden layer is used as the input of the next hidden layer of the LSTM, as seen in Figure 3.18.

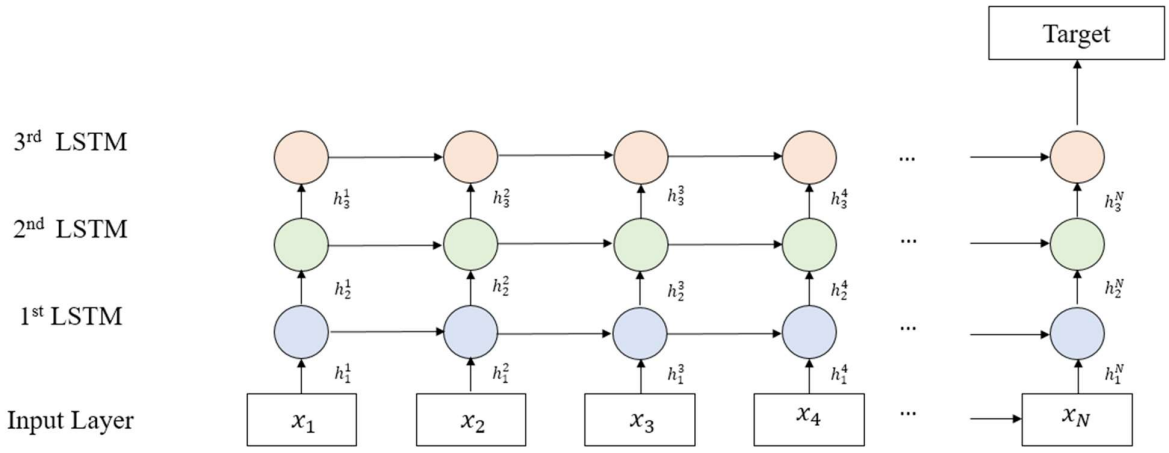

Figure 3. 19: **Stacked LSTM Structure**

3.3.3 CNN

One type of neural network is convolutional neural networks (CNNs). One or more convolutional layers are present. It's frequently employed in classification, image processing, and correlation. CNN is structured similarly to how nerve cells in our bodies communicate with connected neurons. They are distinguished from other neural networks by the convolutional process, which applies filters to every component of the previous input to extract patterns and feature maps.

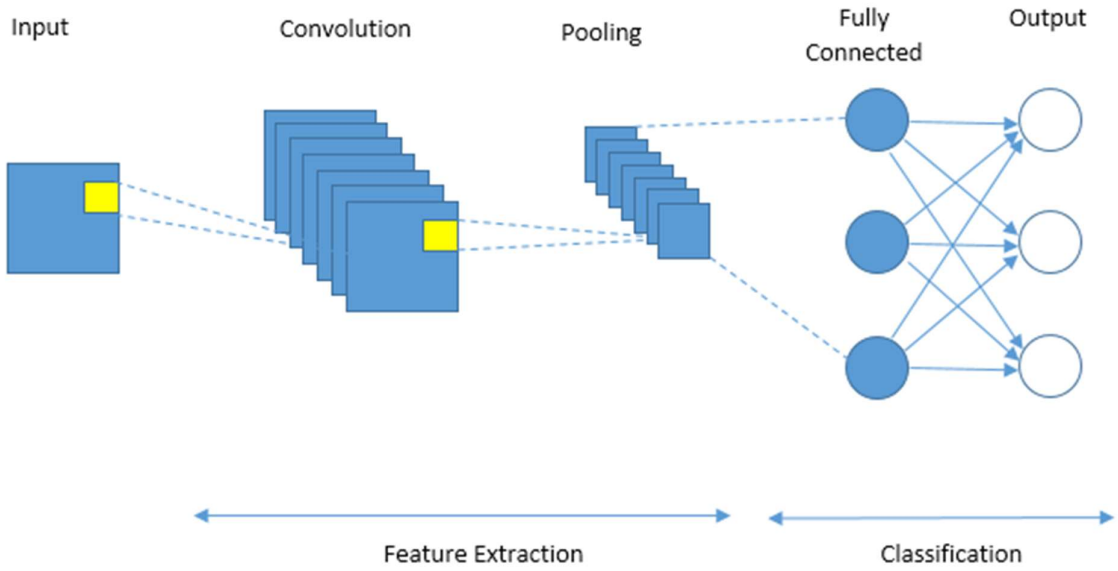

Figure 3. 20: **CNN Algorithm**

There are three layers (Input layer, hidden layer and output layer) in CNN. In hidden layer there are Convolutional layers, pooling layers and fully connected layers.

Convolutional Layers: it is the main layer of CNN. It convolves the input and contains the parameter that is needed for the training.

Pooling Layers: After the convolutional layers, there are pooling layers. It summarizes the features extracted from convolutional layer output. It reduces the parameters by reducing the dimension.

Fully Connected Layers: in this layer every neuron of every layer is connected.

3.3.4 CNN-LSTM

In a CNN-LSTM model, CNN and LSTM layers are coupled. CNN captures the features from the input data, and LSTM anticipates the sequence. CNN LSTMs were proposed to manage issues enhanced data time series prediction. From visual sequences, it may produce written descriptions.

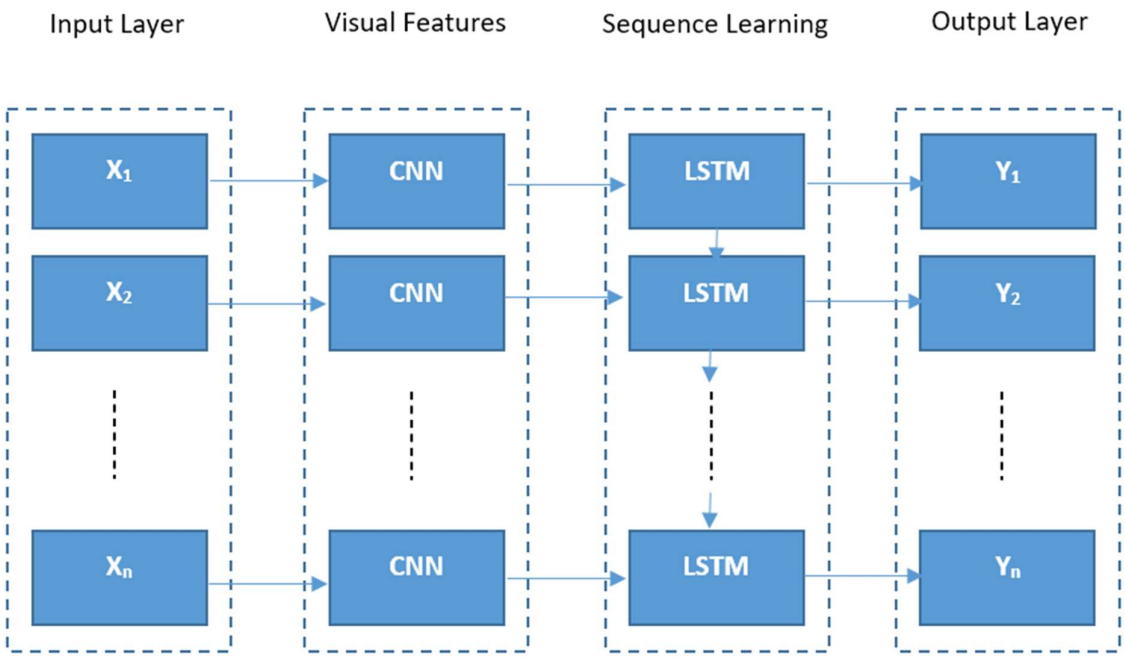

Figure 3. 21: **CNN- LSTM Algorithm**

3.3.5 MLP

The multi-layer perceptron (MLP) is one kind of feed-forward neural network. There are three layers as well. They are input layer, hidden layer and output layer. The input layers take the input signal and the output layer is for guessing and classification. There are multiple number

of hidden layers. Data goes from input layers to output layers. Back propagation is used to train the neurons. MLPs are equipped to deal with tasks that are not at all differentiable and can estimate any continuous function. MLP's most typical uses include pattern identification, recognition, prediction, and approximation.

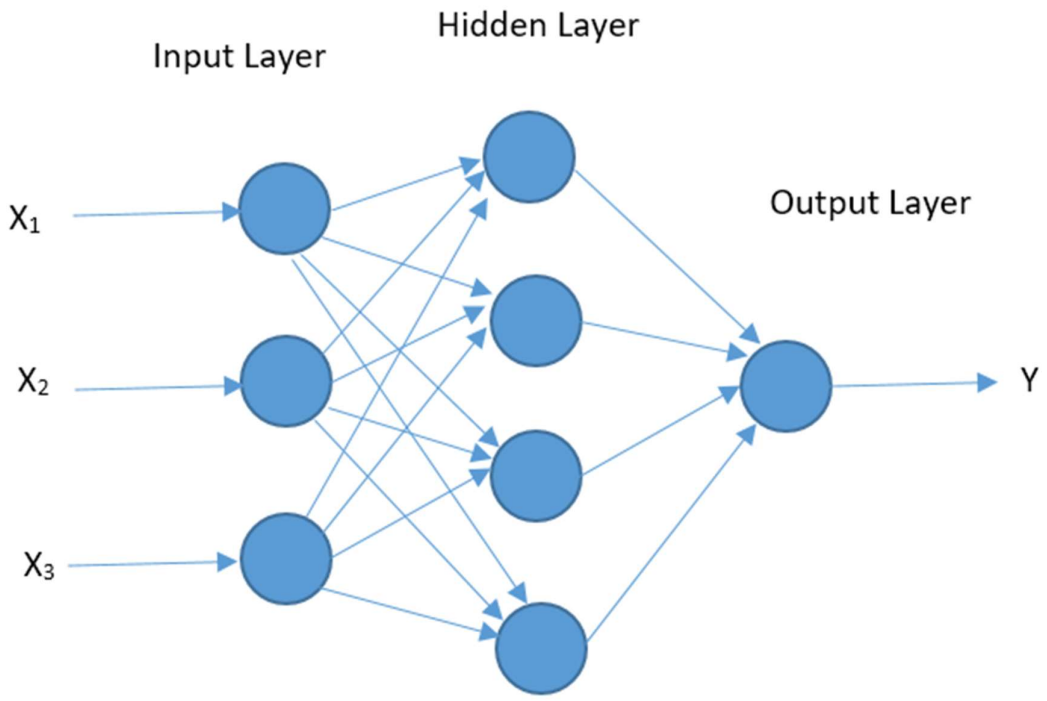

Figure 3. 22: **MLP Algorithm**

3.3.6 XGBoost

Extreme Gradient Boosting (XGBoost) is a distributed gradient-boosted decision tree (GBDT) machine learning toolkit. It can be scaled. It is a tool for regression and classification. It contains parallel boosted trees.

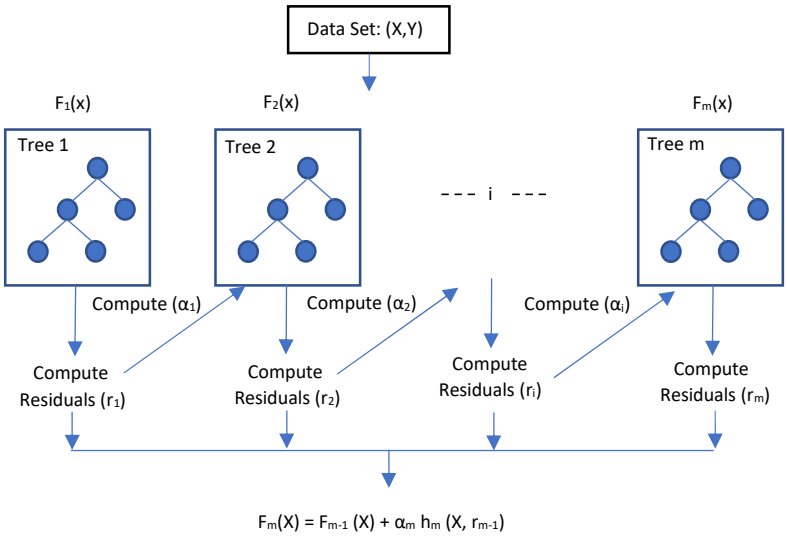

Figure 3. 23: **XGBoost Algorithm**

Where,

α_i= regularization parameters of i^{th} tree

r_i= residuals computed with i^{th} tree

h_i= function that is trained to predict residuals, r_i using X for i^{th} tree

3.3.7 Encoder- Decoder

Encoder decoder models enable a machine learning model to construct a phrase that describes an image. It takes an image as input and produces a string of text. This also applies to videos. The encoder-decoder architecture can handle variable-length sequences as inputs and outputs, making it ideal for sequence transduction problems like machine translation.

Figure 3. 24: **Encoder- Decoder Algorithm**

3.4 Evaluation Method

The performance of any forecasting model is done by comparing the forecasted value and the actual test value. Similarly in our load forecasting model we used various evaluation metrics to evaluate the performance of our deep learning models. The most commonly used metrics are: MAPE, MAE, RMSE, MSE etc. According to our study majority of the researchers use RMSE as their main evaluation metric. For our short term load forecasting case, we used RMSE, MAPE, MSE, MAE AND R^2.

MAPE: The sum of total absolute errors divided by the actual demand (for each period separately). Percentage errors are averaged.

$$MAPE = \frac{100\%}{n}\sum\left|\frac{y - \hat{y}}{y}\right| \tag{3.7}$$

MSE: A three-dimensional sum of squared differences between fitted values and observed values minus the number of parameters in the model, divided by the number of historical points.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.8}$$

RMSE: In other words, the square root of the MSE. Observed data values are on the same scale as the square root of the MSE.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{3.9}$$

R-square: According to a linear model, R-squared accounts for a certain portion of the variation of the dependent variable.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2} \tag{3.10}$$

Chapter 4

4 Result Analysis

The performance of each model is analyzed in this section. Comparing evaluation metrices of each methods help us determine which model gives the best prediction.

4.1 LSTM Method

LSTM model parameters:

- LSTM layer: 24, 100
- Dense Layer: 200
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003

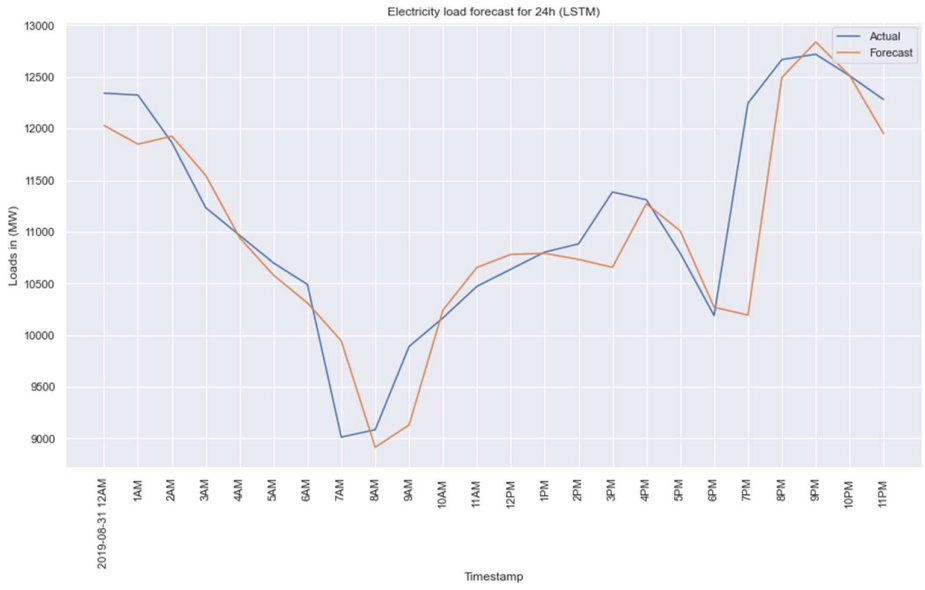

Figure 4. 1: **Actual and Predicted load for LSTM Method**

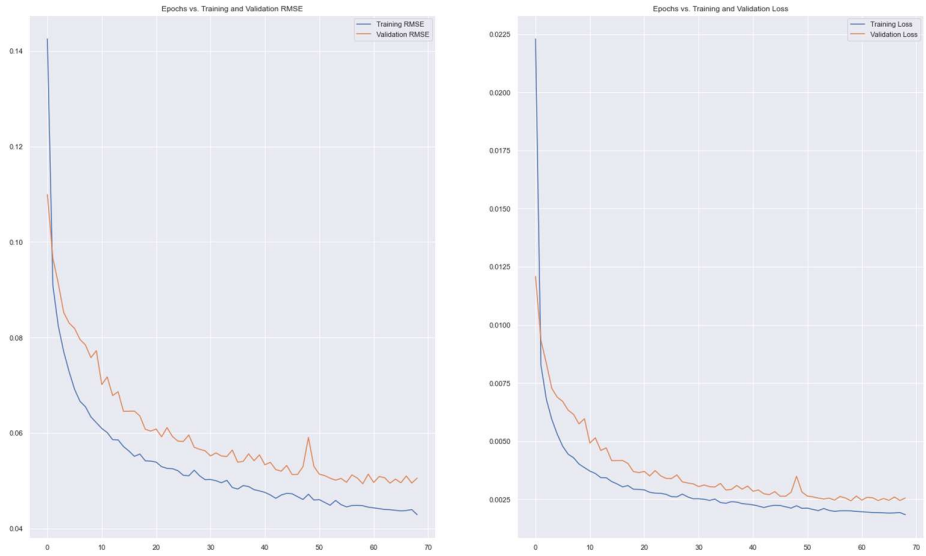

Figure 4. 2: **Epoch vs Training and Validation RMSE and loss**

Table 4. 1: **Result of evaluation metrices for LSTM Method**

Evaluation Method	***Result***
MSE	87521.655
MAE	186.556
RMSE	295.840
MAPE	2.280
R-squared	0.973

4.2 Stacked LSTM

Model parameters:

- 1st LSTM layer: 250
- 2nd LSTM layer: 150
- Dense Layer: 150
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003

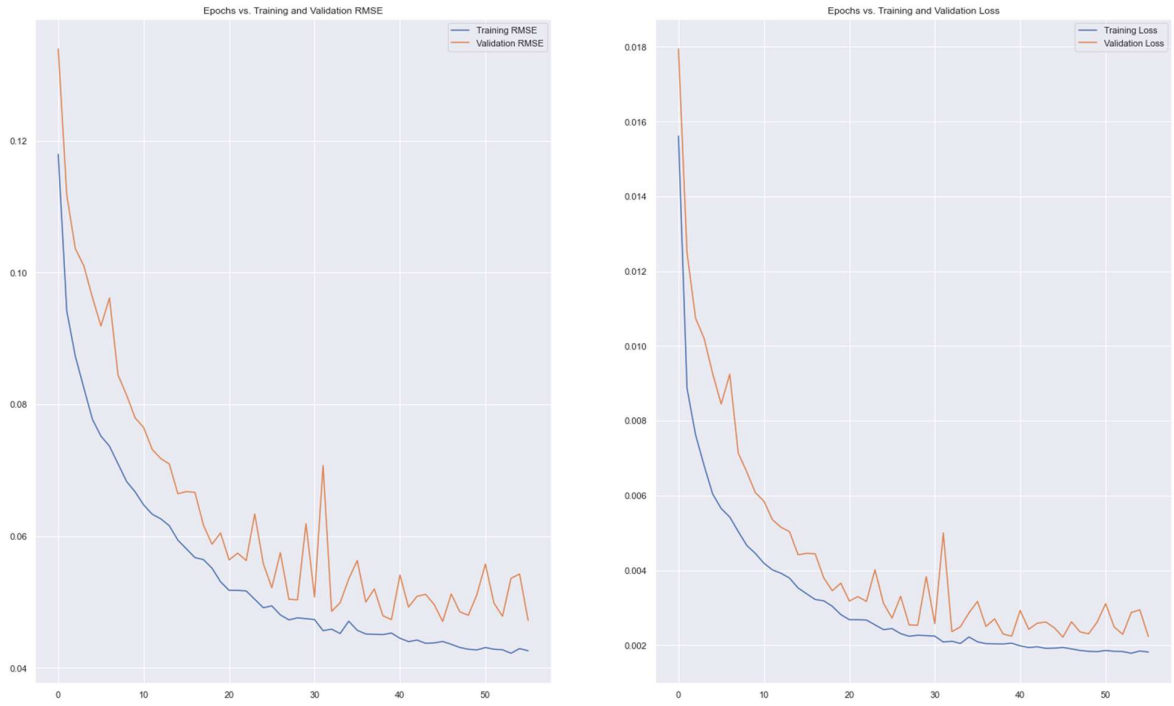

Figure 4. 3: **Epoch vs Training and Validation RMSE and loss**

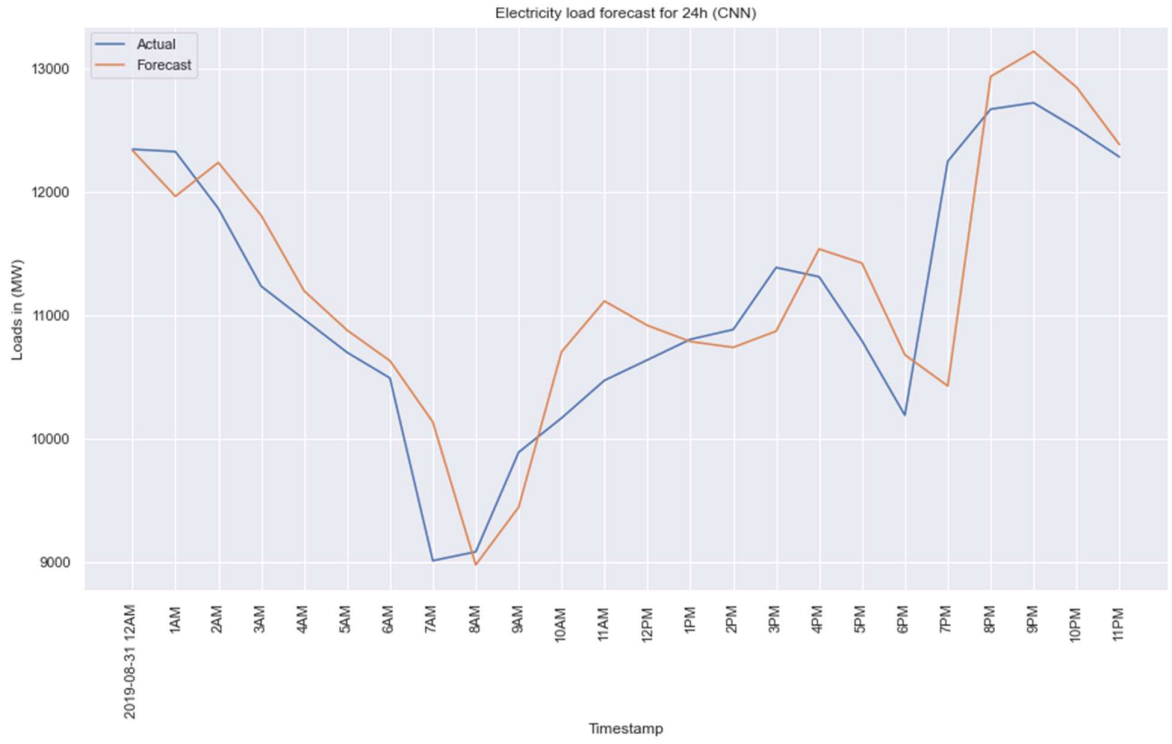

Figure 4. 4: **Actual and Predicted load for Stacked LSTM Method**

Table 4. 2: **Result of evaluation metrices for Stacked LSTM Method**

Evaluation Method	*Result*
MSE	89664.370
MAE	190.224
RMSE	299.440
MAPE	2.316
R-squared	0.972

4.3 XGBoost forecast

Figure 4. 5: **Actual and Predicted load for XGBoost forecast**

\

Table 4. 3: **Result of evaluation metrices for XGBoost forecast**

Evaluation Method	*Result*
MSE	204701.385
MAE	274.113
RMSE	452.439
MAPE	3.106
R-squared	0.937

4.4 CNN forecast

Model parameters:

- Dense Layer: 150
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003

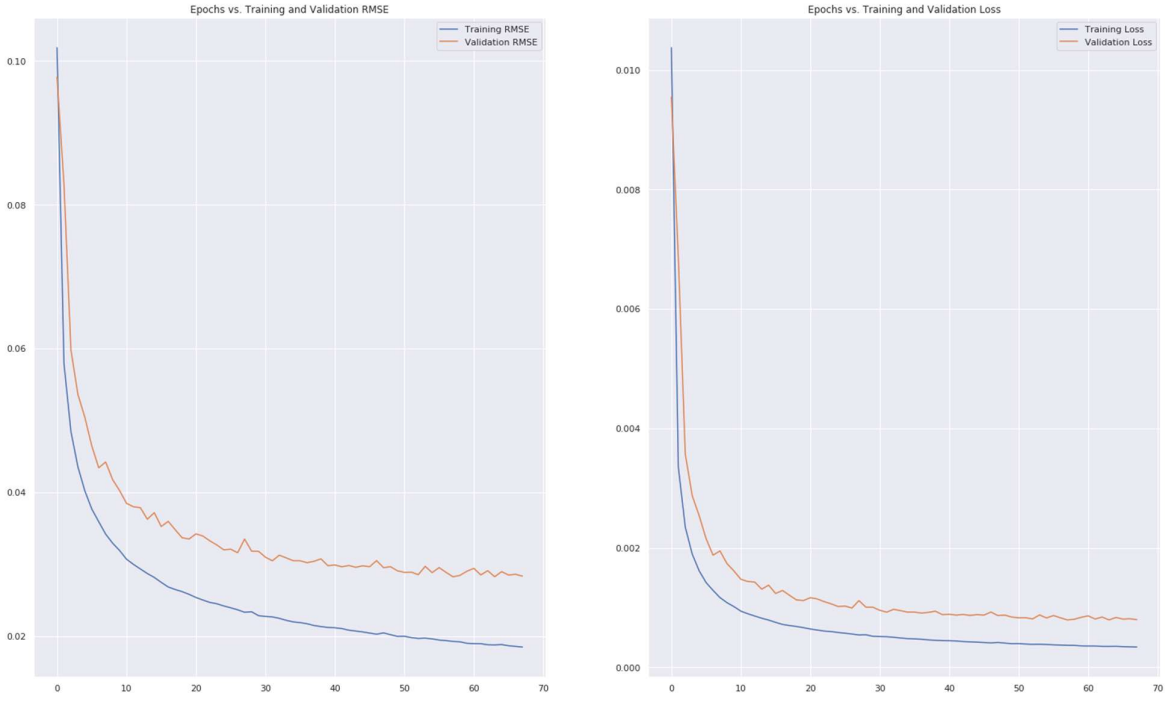

Figure 4. 6: **Epoch vs Training and Validation RMSE and loss**

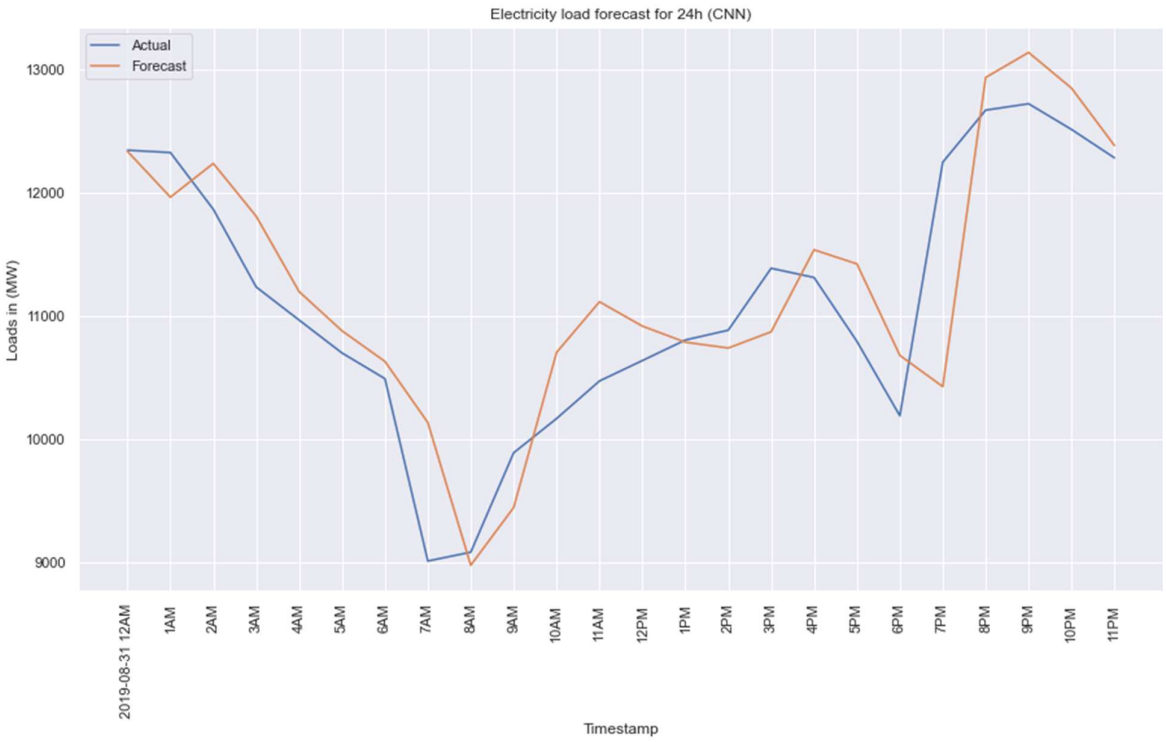

Figure 4. 7: Actual and Predicted load for CNN Method

Table 4. 4: Result of evaluation metrices for CNN Method

Evaluation Method	*Result*
MSE	95594.147
MAE	218.473
RMSE	309.183
MAPE	2.759
R-squared	0.970

4.5 CNN-LSTM forecast

Model parameters:

- LSTM layer: 100
- Dense Layer: 50
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003

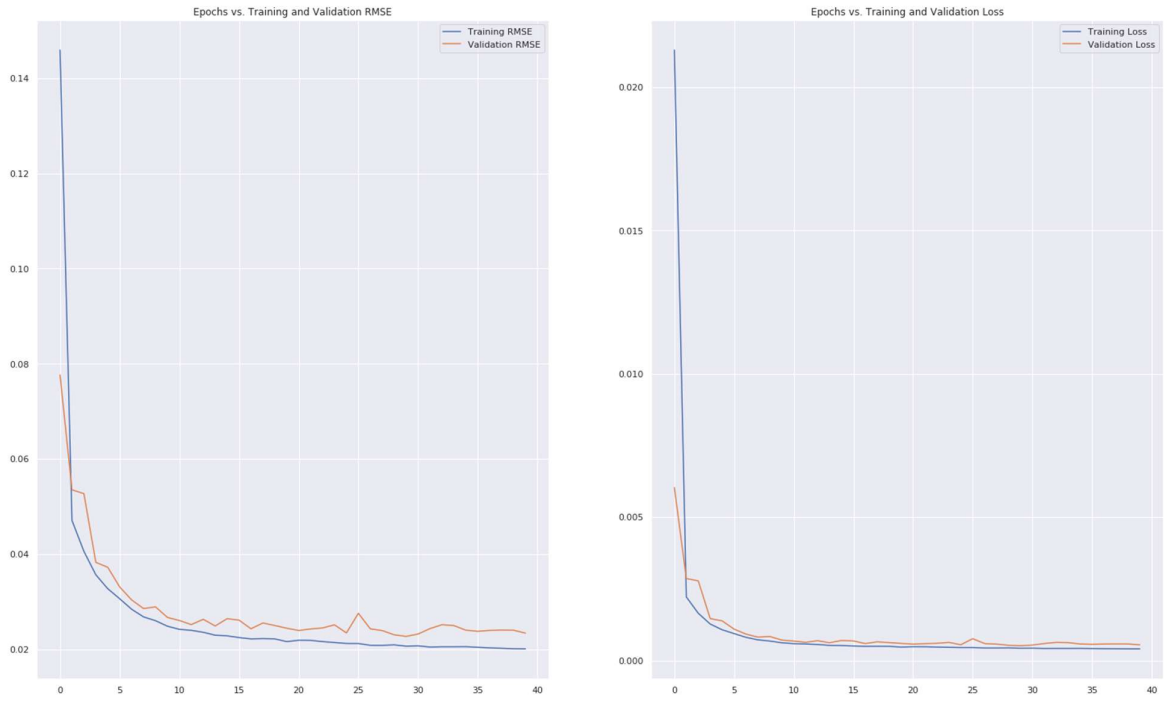

Figure 4. 8: Epoch vs Training and Validation RMSE and loss

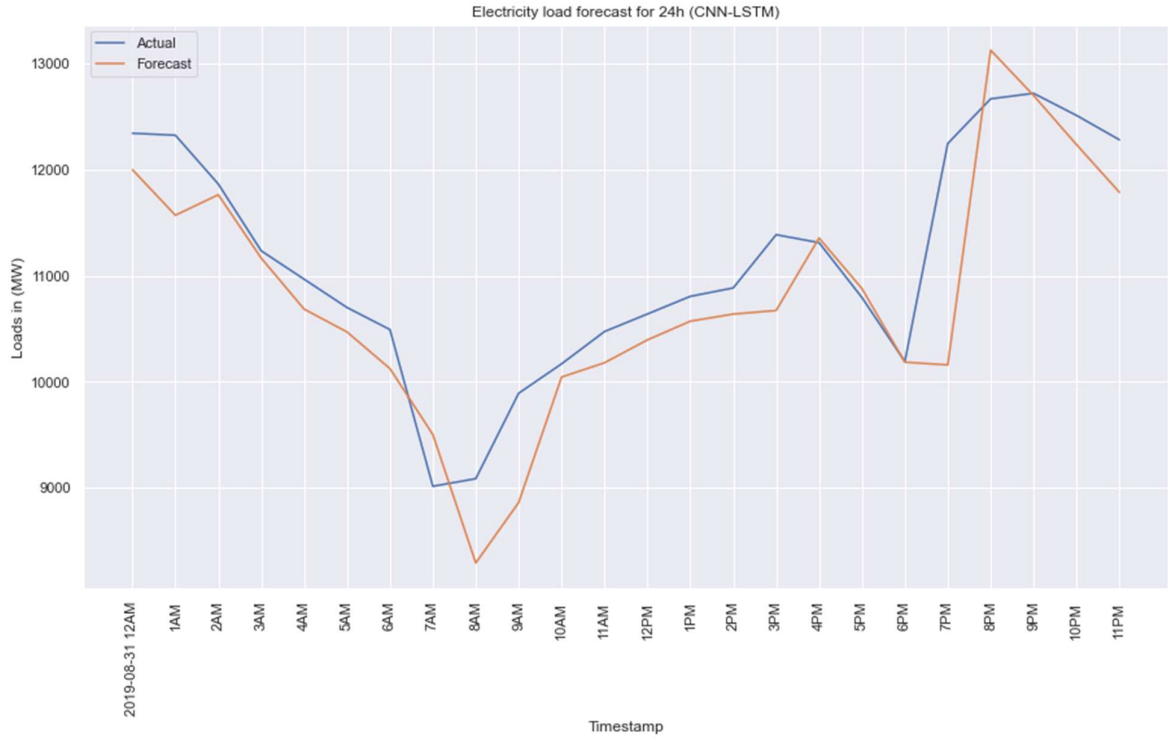

Figure 4. 9: Actual and Predicted load for CNN-LSTM Method

Table 4. 5: **Result of evaluation metrices for CNN-LSTM Method**

Evaluation Method	*Result*
MSE	81058.879
MAE	183.236
RMSE	284.708
MAPE	2.234
R-squared	0.975

4.6 MLP forecast

Model parameters:

- Time Distributed Dense layer: 200, 150, 100, 50
- Dense Layer: 150
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003Figure:

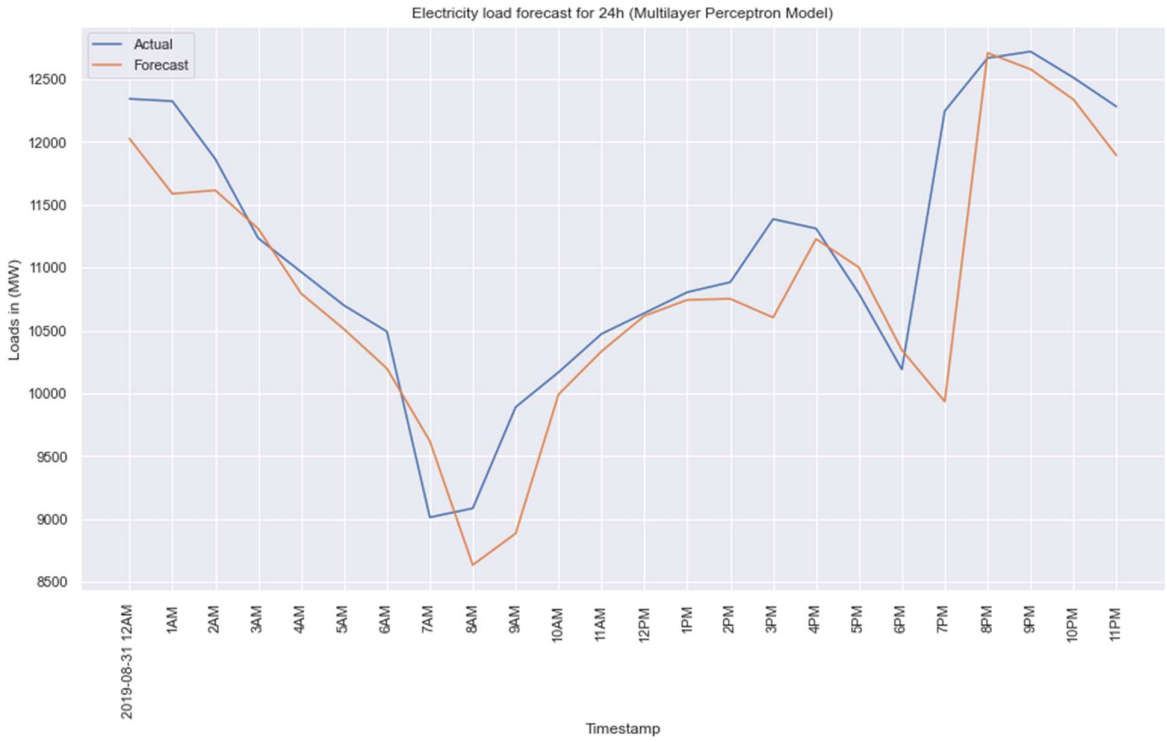

Figure 4. 10: **Actual and Predicted load for MLP forecast**

Table 4. 6: **Result of evaluation metrices for MLP forecast**

Evaluation Method	*Result*
MSE	78738.070
MAE	182.337
RMSE	280.603
MAPE	2.268
R-squared	0.975

4.7 Encoder-Decoder forecast

Model parameters:

- LSTM layer: 50, 50
- Time Distributed Layer: 50
- Dense Layer: 150
- Activation Function: relu
- Optimizer: Adam
- Learning rate: 0.0003

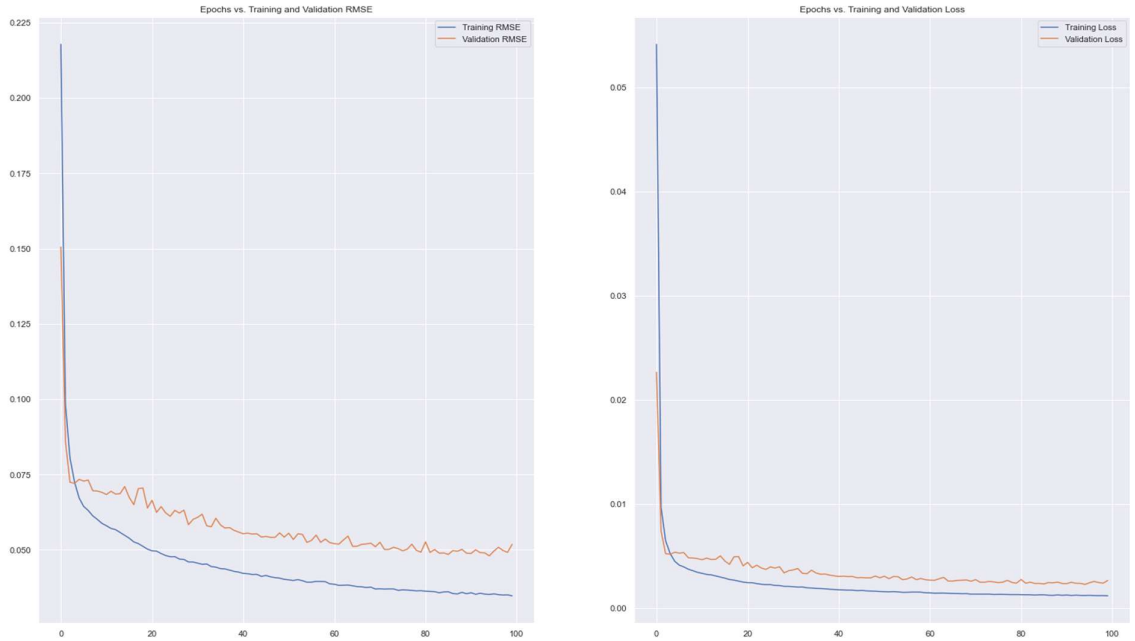

Figure 4. 11: **Epoch vs Training and Validation RMSE and loss**

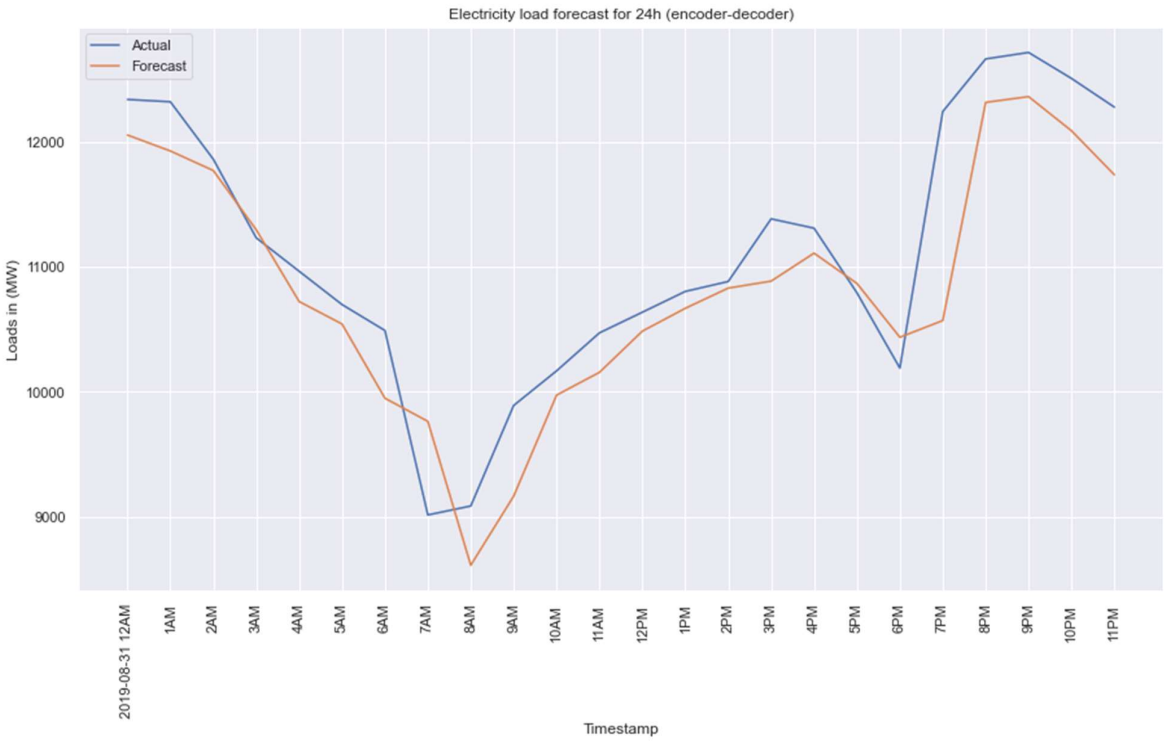

Figure 4. 12: **Actual and Predicted load for Encoder-Decoder forecast**

Table 4. 7: **Result of evaluation metrices for Encoder-Decoder forecast**

Evaluation Method	***Result***
MSE	122748.698
MAE	222.331
RMSE	350.355
MAPE	2.720
R-squared	0.962

The overall comparison of each model is shown in Table: . It can be seen that, the model with the lowest MSE is MLP. In case of MAE, MLP also performs better. If we consider RMSE values of each model, we can see that, MLP performs the best. But, in case of MAPE, the best model is CNN-LSTM. Comparing all the R-Squared vales, it can be seen that, XGBoost performs the best.

Table 4. 8: **Comparison of Models based on evaluation metrices**

Model	***Evaluation Method***				
	MSE	***MAE***	***RMSE***	***MAPE***	***R-Squared***
LSTM	87521.655	186.556	295.840	2.280	0.973
Stacked LSTM	89664.370	190.224	299.440	2.316	0.972
XGBoost	204701.385	274.113	452.439	3.106	0.937
CNN	95594.147	218.473	309.183	2.759	0.970
CNN-LSTM	81058.879	183.236	284.708	2.234	0.975
MLP	78738.070	182.337	280.603	2.268	0.975
Encoder-Decoder	122748.698	222.331	350.355	2.720	0.962

Chapter 5

5 Conclusion

As presented in the initial chapters, electric load forecasting is nowadays a vital process. For a developing country like ours and our diminishing resources, it is important to have an optimized plan of load demand and consumption and also the growing demands for electricity from consumers and from our day-to-day lives. As seen during literature review, mostly machine learning models and statistical approaches exist to predict short-term load forecasting. Deep neural networks have also been used worldwide. But on our dataset, we have incorporated multivariate time series analysis using deep learning models comparing them with ML models. Applying six of them (LSTM, Stacked-LSTM, CNN, LSTM-CNN, Time Distributed MLP and Encoder Decoder) we found out that the deep learning models outperformed the machine learning models. One interesting finding of ours is that the multilayer perceptron model got the lowest error rate compared to the other deep learning models. Overall, deep learning models dominated the machine learning models.

5.1 Future Work

As can be seen from the literature, many techniques have yet to be developed. Model ensemble by using different types of optimizers in use. The introduction of a spark for distributed computing would enhance performance replacing grid search, which is very time consuming. Only Dhaka City was used and its meteorological features were used. What if we incorporated more cities? This will make the process easier. Since we have pre-trained models for making predictions, we only need to construct new ones with corresponding calendar and meteorological values for the future date. Particularly, LSTM-CNN model giving overall better performance can be implemented on other data sets from other countries having similar condition to ours.

6 Reference:

[1] Haque, S. A., & Islam, M. A. (2021). Artificial Neural Network-Based Short-Term Load Forecasting for Mymensingh Area of Bangladesh. *International Journal of Electrical and Computer Engineering*, *15*(3), 99-103.

[2] Islam, A., Hasib, S. R., & Islam, M. S. (2013). Short term electricity demand forecasting for an isolated area using two different approaches. *Journal of Power Technologies*, *93*(4), 185-193.

[3] HASAN, T. M., ISLAM, S. S., HRIDOY, M. A. K., PARVEZ, H., SHARIF-AL-AMIN, M. D., CHOWDHURY, N. H., ... & DHAKA, B. (2020). Demand side management of electricity for controlling peak demand in bangladesh.

[4] Paparoditis, E., & Sapatinas, T. (2013). Short-term load forecasting: The similar shape functional time-series predictor. *IEEE Transactions on power systems*, *28*(4), 3818-3825.

[5] Yu, Z. (1996). A temperature match based optimization method for daily load prediction considering DLC effect. *IEEE Transactions on Power Systems*, *11*(2), 728-733.

[6] Sadownik, R., & Barbosa, E. P. (1999). Short-term forecasting of industrial electricity consumption in Brazil. *Journal of Forecasting*, *18*(3), 215-224.

[7] Al Amin, M. A., & Hoque, M. A. (2019, March). Comparison of ARIMA and SVM for Short-term Load Forecasting. In *2019 9th annual information technology, electromechanical engineering and microelectronics conference (IEMECON)* (pp. 1-6). IEEE.

[8] Grimaccia, F., Mussetta, M., & Zich, R. E. (2012, June). Advanced predictive models towards PV energy integration in smart grid. In *2012 IEEE International Conference on Fuzzy Systems* (pp. 1-6). IEEE.

[9] Bunn, D. W. (2000). Forecasting loads and prices in competitive power markets. *Proceedings of the IEEE*, *88*(2), 163-169.

[10] Hipel, K. W., & McLeod, A. I. (1994). *Time series modelling of water resources and environmental systems*. Elsevier.

[11] Kardakos, E. G., Alexiadis, M. C., Vagropoulos, S. I., Simoglou, C. K., Biskas, P. N., & Bakirtzis, A. G. (2013, September). Application of time series and artificial neural network models in short-term forecasting of PV power generation. In *2013 48th International Universities' Power Engineering Conference (UPEC)* (pp. 1-6). IEEE.

[12] Nie, H., Liu, G., Liu, X., & Wang, Y. (2012). Hybrid of ARIMA and SVMs for short-term load forecasting. *Energy Procedia*, *16*, 1455-1460.

[13] Karthika, S., Margaret, V., & Balaraman, K. (2017, April). Hybrid short term load forecasting using ARIMA-SVM. In *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)* (pp. 1-7). IEEE.

[14] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1798-1828.

[15] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

[16] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, *27*.

[17] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3156-3164).

[18] Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3128-3137).

[19] Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., & Yuille, A. (2014). Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*.

[20] Al Amin, M. A., & Hoque, M. A. (2019, March). Comparison of ARIMA and SVM for Short-term Load Forecasting. In *2019 9th annual information technology, electromechanical engineering and microelectronics conference (IEMECON)* (pp. 1-6). IEEE.

[21] World Weather Online - https://www.worldweatheronline.com/

[22] Zia, T., & Zahid, U. (2019). Long short-term memory recurrent neural network architectures for Urdu acoustic modeling. International Journal of Speech Technology, 22(1), 21-30.