UiT The Arctic University of Norway

Department of Electrical Engineering

Electric Load Forecasting of Residential Building

Mubashir Nasir Lodhi

Candidate # : 14

Master's thesis in Electrical Engineering…ELE-3900…May 2024

ACKNOWLEDGEMENT

I would like to express my sincerest appreciation to my Supervisors, PhD researcher Nasrin Kianpoor and Professor Bjarte Hoff for helping me out in this journey. Nasrin Kianpoor was available every time for my help and guidance. Her experience, guidance, and constructive criticism have played a very important role in the success of this thesis.

I also want to thank my Manager (Christian Murillo) for giving me all the resources and relaxation that was needed for me to complete my thesis. His support was very instrumental and helpful and made me complete my degree along with a full-time job.

Furthermore, I would present my heartful thanks to my parents and siblings who believed in my abilities and were always there for my support whenever it was needed. Also, I would like to thank my friends who guided me through the process and were available for my help.

Finally, Thanks to everyone who has contributed to my academic journey and has made all those baby steps for me to reach this post-graduate level today.

ABSTRACT

As global energy consumption continues to rise relevant energy sectors and communities must accurately forecast future electricity needs. This foresight is critical for effective planning, preserving the stability of the electricity grid, and avoiding blackouts. Accurate forecasting methodologies are critical in guiding decision-making processes correlated to resource allocation, infrastructure development, and policy formulation.

In this thesis, we used a combination of traditional methods like Linear Regression and advanced techniques such as Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Transformer Machine learning models, as well as Empirical Mode Decomposition (EMD) signal processing. Notably, the use of the Transformer method, a relatively new approach to time series forecasting, delivered particularly promising results. We observed a significant improvement in prediction accuracy after incorporating EMD analysis along with training models.

We used a variety of metrics to evaluate model performance and assess its effectiveness. The metrics used were Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Normalized Root Mean Square Error (NRMSE), and coefficient of determination (R2). Using this diverse set of metrics, we aimed to gain a comprehensive understanding of each model's predictive capabilities.

This comprehensive methodology enabled us to evaluate strengths and shortcomings in several forecasting approaches and make informed decisions about their feasibility for practical application. We wanted to ensure the reliability and robustness of our forecasting models by conducting thorough evaluation utilizing several criteria, allowing for more accurate and informed decision-making in energy management strategies in residential buildings and beyond.

Table of Contents

LIST OF FIGURES

List of Tables

ABBREVIATIONS

LSTM	Long short-term memory
ML	Machine Learning
SVM	Support vector machines
RES	Renewable Energy Sources
PV	photovoltaic
HEMS	Home Energy management System
EMD	Empirical mode decomposition
ESU	Energy storing unit
IMF	Intrinsic Mode Function
GAM	Generalized Additive Model
CNN	convolution Neural Network
RNN	Recurrent Neural Network
NIDS	Network Intrusion Detection System
AI	Artificial Intelligence

MARL	Multi agent reinforcement learning
NAR	Nonlinear Autoregressive
ANN	Artificial Neural Network
NRMSE	Normalized mean squared error
MLP	Multi-Layer Perceptron
DNP	Deep Neural Prophet
GPU	Graphics Processing unit
R2	Coefficient of Determination
NLP	Natural Language processing
NN	Neural network
ARIMA	Autoregressive integrated moving average
MAE	Mean Absolute Error
RMSE	Root mean square error
ACF	Auto Correlation Function
DLT	Day light Time
DSO	Distribution system operator

Chapter 1

1. INTRODUCTION

This Chapter will cover the background and motivation for working on this topic. The whole problem statement is explained, together with a discussion of the thesis's key objectives and scope.

1.1. Background and Motivation

As the world transitions to smarter, more energy-efficient houses, electric load forecasting in residential buildings becomes more essential. These houses are not only comfortable, but they are also environmentally friendly as they comply with EU's ambitious 2050 climate targets. These goals require a paradigm shift in how buildings consume and regulate energy and have to account for sustainability and minimum impact on the environment [1].

During this transition, Home Energy Management Systems (HEMS) have been playing an important role for both the energy optimization and the integration of renewable energy sources. Such systems would not be able to operate without the proper estimations of power demand which are both accurate and precise. Unbiased forecasting of loads is vital for a number of reasons such as energy saving, grid stability, infrastructure development, and policy factors [3].

It becomes even more important for the cases of renewable energy integration where intermittence is the main problem for the sources like solar and wind power.Researchers have tried many forecasting methods. Initially focus was on physics based models but later on the concept was expanded to accommodate data driven techniques with the introduction of big data

and the evolution of machine learning. The use of AI and ML was especially successful in deep learning based on RNNs and LSTMs models that were able to identify complex patterns in electricity usage [11].

More advanced AI strategies have recently evolved, such as Transformer models that use self-attention mechanisms. These methods, when combined with signal processing techniques like EMD, have the potential to significantly enhance forecast accuracy. This thesis is driven by the need to explore these sophisticated methodologies in the context of household energy consumption, with the goal of creating models that can give more accurate forecasts.

1.2. Problem Statement

The fundamental issue addressed in this thesis is the inadequate prediction capability of existing traditional load forecasting techniques, which limits the practicality of Home Energy Management Systems (HEMS). As EU's 2050 climate goals are being met with the transition of residential buildings towards smart houses, the precise models with high reliability of forecasting become necessary for strategic plans, the support of grid stability, and the prevention of power outages.

Traditional forecasting models depend on the basic statistical methods and fail to predict the nonlinear patterns of user behavior and weather etc. causing deviations in electrical consumption [4]. Sophisticated artifical intelligence technologies, like deep learning algorithms have not been widely adopted yet in the energy sector.

Moreover implementation of these AI techniques, including RNNs, LSTMs, and Transformer models has their own challenges. These include availabilty of data huge processing resources and knowledge of advanced machine learning. Empirical Mode Decomposition (EMD), an effective approach for analyzing non-stationary and non-linear time series data, is also not fully utilized in the field of electric load forecasting.

The challenge is thus the need for better forecasting accuracy using complex methodologies, and the practical use of these approaches in a way that is feasible and financially viable for residential HEMS. This thesis aims to create and test an effective forecasting model that makes use of modern AI and signal processing techniques, as well as to compare its performance to traditional models.

1.3. Research Objectives

This thesis covers the following objectives.

- Literature review of Load forecasting techniques.
- Load forecasting of a residential building by different methods and compare the Results including LSTM.
- Validating and testing the forecast results with real measured data.
- Analyzing the performance of trained network in terms of error between measured data and forecast result with different metrics.

1.4. Thesis Limitations

Here are some limitations of this thesis.

- Only data of one house is used for this analysis. The results may change, and model may not capture diverse energy usage behaviors and pattern when implementing it on multiple households.
- The algorithms were trained on a very limited dataset; hence, there is significant potential for improved performance as data availability increases, allowing the algorithms to identify more complete patterns.

1.5. Outline of thesis

This thesis is divided into five major chapters with each chapter as a unit for detailed evaluation of electric load forecasting using advanced machine learning techniques.

Below is a detailed outline of each chapter:

- **Introduction:** This first chapter of the research is aimed at providing an overview of the importance of precise electric load forecasting in residential settings and in the process, it sets the stage for the research. It outlines the problem statement, specifies the objectives of the study, and describes the importance of the research. The chapter ends by giving a quick outline of the thesis structure so that readers can easily follow the next chapters.
- **Literature review:** The literature review chapter provides a systematic literature review covering the previous research of electric load forecasting. It includes a series of methods from classical statistical models to modern machine learning techniques. The chapter also comprises the strengths and weaknesses of existing models and it sets the gaps in the existing literature. Therefore, there is the need for the proposed research.
- **Methodology:** This chapter gives detailed information on the research methodology applied in the thesis. It starts by data collection and preparation that includes how the datasets were sourced and processed. It describes the data pre-processing steps, exploratory data analysis, and feature engineering methods used for modelling. The following section sheds light on the development and utilization of different forecasting models such as linear regression, RNNs, LSTMs, and Transformers. It also includes a novel use of the Empirical Mode Decomposition (EMD) method. The optimization methodology with Bayesian optimization is also discussed.

- **Results and Discussion:** This chapter will demonstrate the outcomes of the different models and will present in-depth analysis of the models. It contains a thorough analysis of the models' performance, where it is revealed which of them are more accurate, reliable, and efficient in forecasting electric load. Moreover, the discussion focuses on the impact of this approach on the models' potential to cope with non-linear and non-stationary data that is affected by external factors such as weather conditions and time changes.
- **Conclusion:** The last chapter includes the main findings of the research. It discusses the extent to which the study achieved its objectives and the contribution of the research to the field of load forecasting based on electric power. It appraises the practical values of the research results for stakeholders which include utility firms and energy policymakers. Moreover, the chapter shows the shortcomings of the present study and the way to conduct future research, together with the ideas for the improvement of the load forecasting models.

This structure provides a coherent progression from introduction of the topic to the presentation of methodologies and models, through to the results, ending with the conclusions and recommendations that are based on solid grounds. Each chapter is a continuation of the previous one, which makes the whole thesis a coherent and comprehensive work that not only provides new knowledge about the subject but also contributes to the field of electric load forecasting.

Chapter 2

2. LITERATURE REVIEW

2.1 Introduction

The ability of forecasting electrical load with high precision is one of the key factors for the reliable operation and the management of power systems. Electric load forecasting is an important tool for all types of operations and planning decisions, ranging from daily load management to the long-term investment plans. With the energy systems getting very complex, mainly with the introduction of renewable energy sources and smart grids, the need for accurate forecasting models also increases.

In this chapter we will go through the transformation of the load forecasting methods in electric power systems, illustrating how the study has come from simple statistical approaches to sophisticated machine learning and deep learning techniques. Some of the modelling approaches, influence of data pre-processing, how external features are integrated, and latest methodological innovations, will be the key points that will be covered in the literature. Discussion set forth here is to merely characterize the existing issues that the research community is currently facing and to highlight which areas need more breakthroughs and innovations.

The structure of the literature review here is aimed to have a methodical approach to the analysis of electric load forecasting. At first, a historical overview is provided which explains how the methods of forecasting have evolved through the years. Then followed by an in-depth look at the modern techniques, including a discussion of traditional statistical models, as well as the latest machine learning algorithms. Further, the review goes into the details of data processing and the way external factors like temperature and wind affect the forecasting precision. The following sections will be devoted to the combination of various applications

methodologies that include hybrid models and EMD which are used for complex and nonlinear data. Finally, the literature considers the existing issues within this field that needs to be addressed.

This thesis will use literature review to build a basis for the research, anchor it in the existing academic and industrial efforts, and declare how it expounds the research area of the electric load forecasting. The brainstorming process not only pinpoints the voids of the existing research but also provides a prelude for the following chapters that unfold the sureties of the methodology, results, and impacts on the innovative solutions that have been discovered in this study.

2.2 Historical Review

The electrical power system's load forecasting technique is quite ancient, going back to when utilities started realizing the need to determine future demand in order to be able to manage the supply optimally. The history of data science has undergone several major stages, each marked with improvements in the statistical and computer science tools. This section is dedicated to the historical evolution of the electric load forecasting starting from early days and up to the present times.

2.2.1 Early Developments

Traditional models that were used in the load forecasting were based on the trend analysis and simple statistics which allows to predict consumption patterns during a certain period.

Some of the development of early times include:

- **Trend Analysis (1940s-1950s):** At the start, models employed historical load data in projecting future demand through observation of the trend. This model of research was mainly based on qualitative tools using human expertise in the manual analysis.

- **Moving Average and Exponential Smoothing (1950s - 1960s):** Smoothing historical data with simple moving average and exponential smoothing were presented to estimate actual loads and predict the future loads. Brown (1959) and Holt (1960) published their first foundational concepts that are the basis of these studies [10].
- **Regression Models (1960s-1970s):** Regression models became more popular and complex because of their ability to explain the relationship between electricity demand and exogenous variables (temperature, day, week) and economic indicators [15].

In the beginning, load forecasting rested on simple statistical methods including, moving averages and exponential smoothing, to make projections of the existing consumption patterns for the future [16]. With electricity systems getting complex demand forecasting gained more attention. The advent of digital computers in the mid-20th century brought a new epoch of forecasting, allowing for the use of more versatile mathematical models and algorithms [20]. Time series analysis, regression analysis, and econometric models emerged as imperative tools in the arsenal of load forecasting systems which ensured increased precision and dependability.

2.2.2 Time Series Analysis

In 1970´s time series analyis was introduced which was capable of detecting long term dependencies in data more effectively.

- **AutoRegressive Integrated moving average (ARIMA)**

  The ARIMA model popularized by Box & Jenkins among others in their seminal 1970s research is one of the foundational models in the time series forecasting field. The model involves combining three essential components lagging autoregressive, differencing and moving average to deal with non-stationary data that appear as trends [19].

Key Features: ARIMA models are known for their flexibility and capacity to model a large collection of time series data. They have shown really good results in short-term load forecasting where immediate trend and cycle information is paramount.

- **Seasonal ARIMA (SARIMA)**

  Besides ARIMA model, the Seasonal ARIMA (SARIMA) further manages the seasonal fluctuations where changes can be observed in electric load patterns.

  Key Features: SARIMA can capture seasonal effects more effectively and this makes it perfectly suitable for time series analysis, as for forecasting residential load, where daily or weekly patterns are involved [12].

- **Exponential Smoothing State Models (ETS)**

  ETS (Exponential Smoothing State Space Models) is a more advanced version of the earlier techniques developed through exponential smoothing [18]. These models are constructed to encompass errors, trends, and seasonality factors as a unified state space formulation.

  Key Features: This model has shown really great results where the load data exhibits complicated seasonal patterns that evolve over time.

2.2.3 Statistical models

In the late 20th century as the capabilities of computing power reached to advanced levels and data analysis become more complex, there was a shift toward developing a sophisticated statistical model in electric load forecasting. Such advancement helped researchers to analyse more complex relationships in the data allowing them to achieve better results. Some of the developments include:

- **Multiple Linear Regression (MLR):** Multiple Linear Regression (MLR) is advanced version of the simple linear regression model that can analyse multiple independent

variables. Thus, we will have the ability to analyse the influence of these factors on the independent variable in a comprehensive way.

- **Generalized Additive Models (GAM):** The Generalized Additive Models (GAM) have a flexible framework that helps to define the effect of each predictor on the outcome by using smoothing functions that can be non-linear [2].

  Key Features: GAMs prove to be very beneficial in cases where it is hard to assume a linear relationship between predictors and the response variable or the relationship is too complex to be correctly handled by a linear model. This capability enables researchers to deal with the non linear temperature effect, time indices, and other predictors of power demand for load forecasting.

2.3 Introduction to Machine Learning

Machine learning has become a power tool that has transformed multiple disciplines, as through this technology the ways of automating tasks, making decisions, or drawing benefits from data have been radically evolved. As a form of artificial intelligence (AI) machine learning is based on the development of algorithms and statistical models that enable computers to learn from data and make predictions or decisions based on data automatically without the explicit intervention of a programmer in machine learning tasks. In essence, the machine learning algorithms which are iterative, get trained from data, detecting patterns and making decisions making the automation of complex processes possible [7]. In this section, these concepts are introduced, thus setting the stage for grasping the modern modelling techniques used in electric load forecasting.

There are three main categories of Machine Learning models:

- **Supervised Learning:**

  Supervised learning the algorithm is trained to work on a labeled dataset, in which each data component is associated with a target output [22]. The purpose is to find a mapping

from input parameters to certain output parameters using a training set. The tasks that are carried out in supervised learning prominently include classification and regression. Figure 1 shows the example of supervised machine learning.

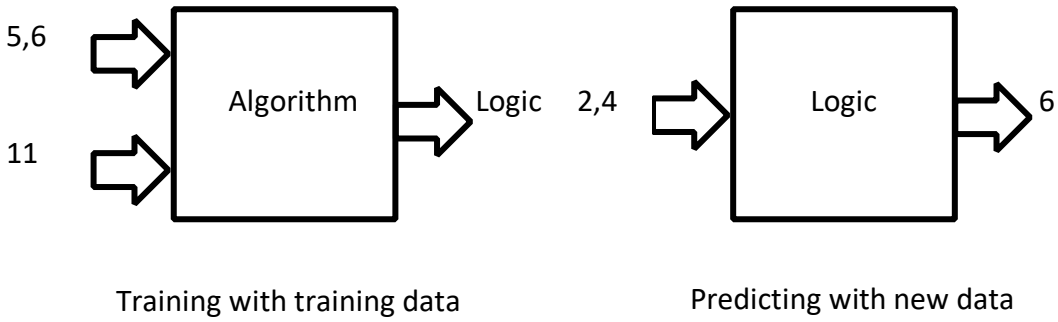

Figure 1: Supervised Machine Learning

- **Unsupervised Learning:**

  Unsupervised learning uses unlabeled data to train machines.Unlabeled data means that there is no fixed output varaible.The model learn from the data,discovers the pattern and features in data and return the output [21]. Figure 2 shows the example of unsupervised learning [51].

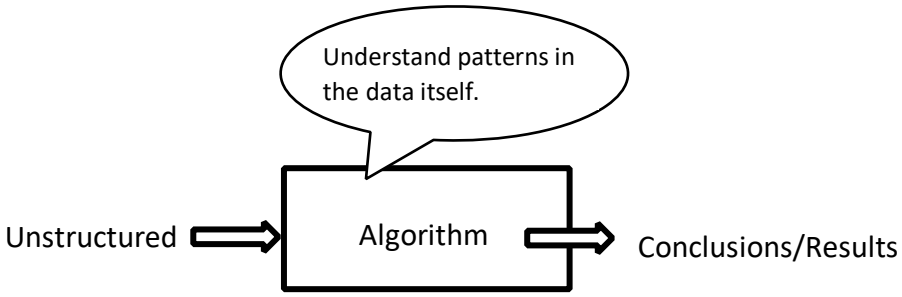

Figure 2: Unsupervised Machine Learning [51]

- **Reinforcement Learning:**

  Reinforcement learning uses an agent and an enviroment to produce actions and rewards.Agent has start and end state but there might be a different paths for reaching the end state. In this learning technique there is no predefined target variable.For

example to train a machine that can identify the shape of object like square,triangle or circle [5].

2.3.1 Advanced Machine learning Methods

With the advancement of computational resources, machine learning methodologies have also been evolved with time. This section will highlight capabilities of modern machine learning models and compare modern techniques with traditional statistical approaches.

Some of the machine learning models used in load forecasting are:

- **Support Vector Machines (SVM):** SVMs are type of supervised machine learning models that are used for classification and regression problems. In load forecasting, SVMs are efficient in this respect because they can model non-linear relations among inputs and outputs quite well [6]. The capability of SVMs to correctly separate data points by a hyperplane enables them to deliver high accuracy forecasts and is more evident when volatile and unstable load patterns are involved.
- **Decision Trees:** Decision trees splits data into branches to form the structure of the tree, and decisions are made based on features. Decision Trees are the most straightforward type as they are simple to understand and visualize. For load forecasting, decision trees emphasize on the cornerstones of the influencing factors like time of day, weather conditions, and customer behaviour on load demands [8].
- **Ensemble Methods:** One of the approaches being used is Random Forests and Gradient Boosting Machines which make use of many learning algorithms, thus enabling them to surpass the performance that can be got from any of the learning algorithms. Ensemble methods are praised for being more precise and robust, and they can parallelly reduce bias and variance. Therefore, this makes in achieving reliable load forecasts [9].

2.3.2 Deep learning approaches

Deep learning methods consist of multiple layers of neural networks, have been at the forefront of recent advancements in predictive analytics. Conventional techniques like linear regression models have not been really good in predictive analytics. So that's why there is a need for deep neural networks. Different types of neural network are:

2.3.2.1 Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNNs) are a type of neural network that is specifically tailored to handle sequential data. These networks therefore perfectly fit time series forecasts including load forecasting. Unlike traditional feed forward neural network, it does not process each input independently but keep the internal connection and retain the information from previous input to capture long term dependencies and temporal patterns in sequential data. RNN has a recurrent connection it retains the data across each time step. The output of network is fed back as an input at each time step which enables RNN to learn from sequential data, making it more efficient to handle complex temporal patterns.

Figure 3 shows the architecture of RNN. Where ′X′ is the input layer and ′h′ is the hidden layer, ′Y′ represents the output layer of the RNN network. ′A′, ′B′, ′C′ are networking variables that were used to enhance output of the network [13].

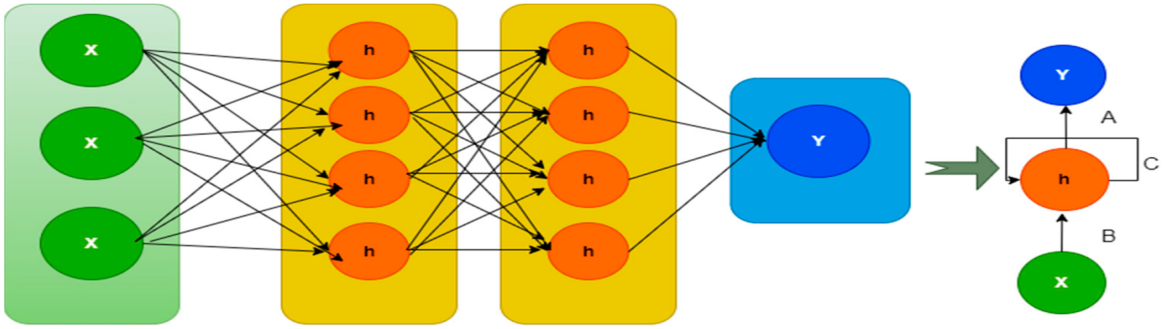

Figure 3: RNN Architecture [13]

Eq 1 shows the calculation of hidden layer where (h_t) is hidden state at time t, (x_t) represents the input at time step t, (w_{hh}) and (w_{xh}) are weight matrix for the hidden state and input. (b_h) is the bias vector for hidden state.

Equation 1:

$$h_t = \tanh\left(w_{hh}h_{t-1} + w_{xh}x_t + b_h\right)$$

Output of RNN is computed based on hidden state (h_t) and corresponding weight matrices and biases as shown in Eq 2

Equation 2:

$$y_t = w_{hy}h_t + b_y$$

Despite of their effectiveness to process sequential data RNN have some limitations. One of the major problems with RNN is vanishing gradient which can affect its performance for data set having long term dependencies [13].

2.3.2.2 Long Short-Term Memory (LSTM)

LSTM is a type of recurrent neural network used to overcome vanishing gradient problem of RNN and to capture long term dependencies.LSTM has a different memory cells and gate that helps it to retain or forget the data depending upon its relevancy and importance.This gated mechanism makes LSTM more suitable for applications having long term dependencies such as time series forecasting, language processing and speech recognition [23].

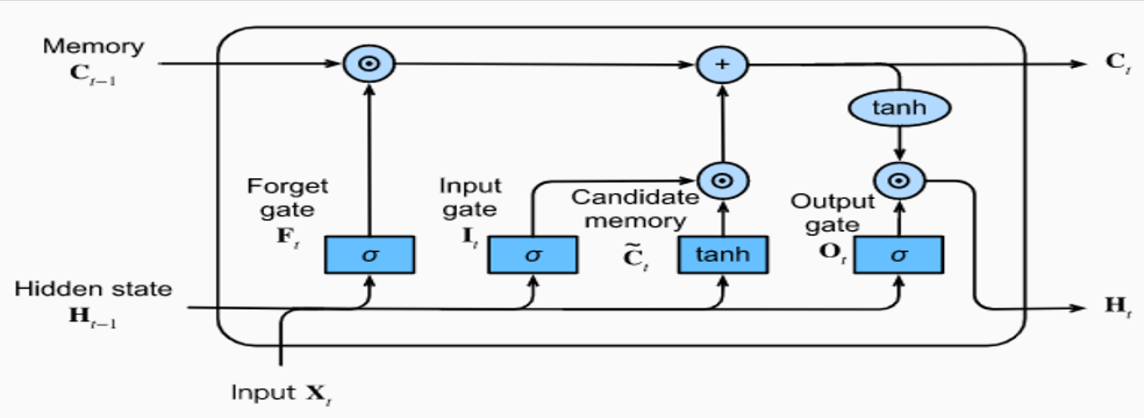

Figure 4: LSTM Architecture[17]

LSTM architecture comprises of several gates and memory states:

- **Input Gate (i_t):** This gate identifies that how much new information should be incorporated into the cell state (c_t) at time step t.

  Equation 3:

$$\boldsymbol{i_t = \sigma(w_{Xi}x_t + w_{hi}h_{t-1} + w_{ci}c_{t-1} + b_i}$$

- **Forget gate (f_t):** it decides which information from previous cell should be retained or forgotten.

  Equation 4:

$$\boldsymbol{f_t = \sigma(w_{Xf}x_t + w_{hf}h_{t-1} + w_{cf}c_{t-1} + b_f}$$

- **Output gate(O_t):** it controls how much current cell state ($\boldsymbol{c_t}$) should be added in output at time step t.

  Equation 5:

$$\boldsymbol{o_t = \sigma(w_{Xo}x_t + w_{ho}h_{t-1} + w_{co}c_t + b_o}$$

- **Cell State update(c_t):** it takes information from candidate cell state ($C\sim t$) and input gate (i_t) to update cell ($\boldsymbol{c_t}$)**.** Equation for candidate cell is given as:

  Equation 6:

$$C\sim t = \boldsymbol{\tan h\,(w_{xc}x_t + w_{ho}h_{t-1} + b_c}$$

  Equation 7:

$$Ct = \boldsymbol{f_t \odot c_{t-1} + i_t \odot C\sim t}$$

- **Hidden State update(h_t):** Finally we determine the output of LSTM at time step t.Equation is given as:

  Equation 8:

$$ht = \boldsymbol{o_t \odot tanh(c_t)}$$

Where σ represents sigmoid activation function and $\odot$ represents the element wise multiplication, and W and b are the weight matrices and bias vectors.

In Load forecasting LSTM models outperformed RNNs and other forecasting techniques in terms of accuracy. Through the memory cells and the gating mechanisms, LSTM models can effectively capture both the fluctuations in the short-term as well as in the long-term trends in electricity consumption data, leading to more credible and reliable forecasts [17].

2.3.2.3 Transformer Machine Learning

Transformer architectures are a very recent breakthrough in the field of natural language processing (NLP) and have also demonstrated importance in time series forecasting applications, such as load forecasting. Contrary to RNNs and LSTMs which process sequence data sequentially, transformers use a self-attention mechanism that enables them to pay attention to any part of the input sequence at one time [24].

The network employs an encoder and decoder architecture with positional encoding.It is pretty much like RNN but the difference is that input sequence can be passed in parallel which is not possible in RNN and LSTM.Transformers rely on self attention mechanism to remember things they don't have any reccurents like RNN and LSTM,hence faster then both of them as data doesn't need to be processed sequentially and can be processed in parallel way.

Figure 5 shows the basic architecture of transformer models used in NLP.it consist of encoders and decoders with layer of self-attention mechanism, add & norm block and feed forward network.it uses positional encoding to keep track of sequence as input and output sequence is processed with embeddings. Model predicts the text or other sequence data after linear and soft max transformation.

Transformers are mostly used in Natural Language processing (NLP) and has shown a very good results as compared to RNN and LSTM.Now researchers have started to use this advanced machine learning model in other applictions as well like time series forecasting.it is believed

self attention mechanism can be very powerful for time series forecasting as its self attention mechanism on entire data set and can provide better results than LSTM and RNN [24].

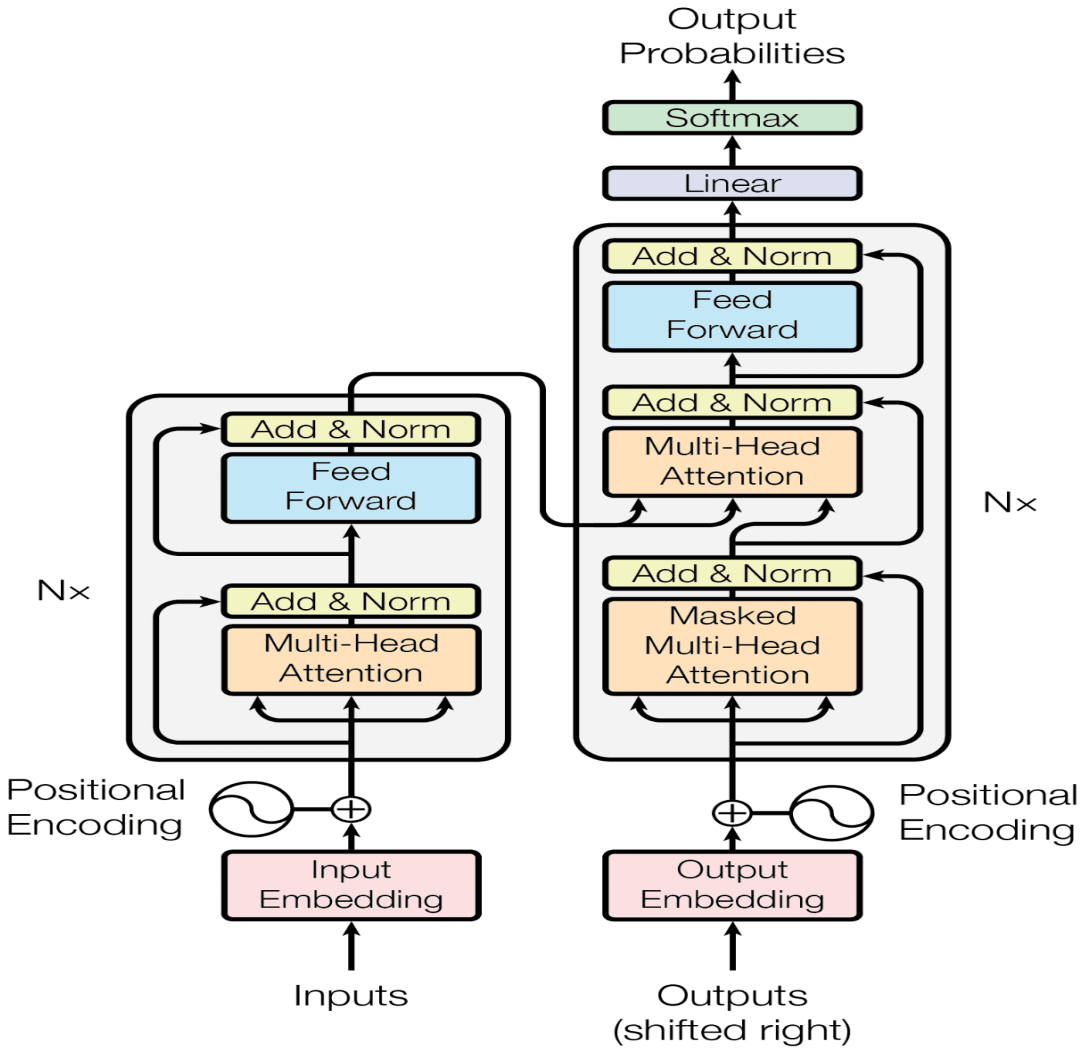

Figure 5: Architecture of Transformers

2.4 Hybrid Models

The hybrid forecasting models have recently obtained popularity because of their features which include blending different forecasting methods to enhance precision and reliability. Such models usually combine other complementary techniques, like utilizing statistical models, machine learning algorithms, and physical models, to capitalize advantages of each and minimize their weaknesses. Li et al. (2020) suggested an ensemble forecasting framework that

integrates the ARIMA model and machine learning techniques, namely random forests and gradient boosting machines, to predict real-time electricity demand in smart grid environments.

2.5 Technological Advancements in Load forecasting

Technological innovations have been key factors that have shaped the field of electric load forecasting. Smart meters, sensor networks, and advanced communication technologies are converged for real-time metering of electricity consumption at a fine level of granularity. These growing amounts of data now allow us to improve forecasting and energy management models through better optimization. Internet of Things (IoT) has propelled the creation of smart devices and sensors, drastically increasing the sources of data available for load forecasting. The emergence of cutting-edge statistics tools like big data analytics and machine learning has allowed utility companies to not only identify but also react instantly and appropriately to critical information hidden in large data sets [28].

2.5.1 Smart Meters: Enhancing Granularity and Real-time Data Collection:

Smart meter which is a great milestone in energy monitoring and management, allows utility providers to collect granular, real-time data on electricity consumption for each household. Smart meters differ from traditional analog meters in providing users with visibility into consumption patterns with granular insights such as time-of-use, peak demand periods, and load profile .Smart meters in residential premises have several impacts on demand forecasting. On the other hand, it allows the utilities to view real-time and precise data on electricity demand which leads to better prediction and load balancing of the electric power. Smart meters provide a basis where utilities reward consumers for shifting their energy usage from peak hours to off-peak periods which in turn leads to an overall reduction of demand with better grid reliability [26].

2.5.2 Internet of Things (IoT) Devices: Expanding Data Sources and Connectivity:

IoT devices are now very popular in residential areas that allows us to have to more data available for load forecasting. Home automation systems give the useful information about the energy usage, user behavior, and environment conditions . Analysing data from IoT devices and smart meters will allow us to create more accurate load forecasting models. A smart thermostat is a good illustration of this because it has the information on the heating and cooling patterns. Other devices such as smart appliances can offer data on usage patterns that are related to a particular device such as refrigerators, washing machines, and electric vehicles [22].

2.5.3 ELHUB and Data Aggregation Platforms:

Aggregating energy data is a way to make the collection and the usage of energy data easy for the energy stakeholders in the residential sector. ELHUB is the data hub of the Norwegian power industry that enables data exchange between the participants of the market, like, utilities, grid operators, retailers and consumers [28]. Consolidating data from various sources like smart meters, IoT devices as well as market transactions by ELHUB gives utilities an option to access the whole wide range of datasets for load forecasting and energy management. Besides, it promotes collaboration and knowledge exchange among the players in the market which in turn leads to more innovations and efficient operation of the market (Elhub, 2021).

2.6 Importance of external factors on load forecasting

Introduction of external factors into electric load forecasting models greatly improves the precision by consideration of the factors not contained in the historical load data. External factors like weather conditions, economic indicators, and social behaviour have been incorporated into models, based on different research studies, and has shown great results [29].

Weather Sensitivity

Weather is one of the most important parameters in load forecasting because of its direct effect on heating, cooling, and energy consumption in general [29]. Different studies have shown the significance of weather variables:

- **Temperature:** Temperature fluctuations are directly or indirectly linked to energy consumption, where the worst cases are seen in regions with severe climatic conditions. Those models that use temperature data along with power consumption tend to have high accuracy because they can adjust the seasonal energy demands more effectively [30].
- **Humidity and Wind Speed:** Inclusion of humidity and wind speed can enhance the efficacy of load forecasts, especially for coastal regions where wind cooling has a noticeable impact on the air temperature [4].
- **Solar Irradiance:** Solar power forecasting study shows that the role of solar irradiance is becoming more essential in the load modelling methods for the countries switching to more renewable energy [31].

Economic and Social Factors: User behaviour and social factors also impact demand of electricity. Load forecasting models incorporating these factors can better predict variations due to social related activities.

- **Economic Indicators:** Indicators like GDP growth rates and industrial production indices are critical factors that one needs to be taken into account for long term electricity demand forecasts as they are very important in a developing region that is growing rapidly [27].
- **Public Holidays and Events:** Integration of the calendar variables like public holidays and special days shows a positive correlation with the forecasting quality. Electric

demand largely differs during holidays and events because of fluctuations in commercial and industrial activity [32].

- **Demographic Factors:** Population growth and urbanization in electricity demand cannot be underestimated. Demographic changes influence long-term electricity demand forecasts, calling for models that are sophisticated enough to include these factors in forecasting.

2.7 Signal Processing techniques

The signal processing tchniques are very useful tool in the analysis of the data for the load forecasting models. In this section,we will go through the practical methods of signal processing including EMD, Hilbert transform, and FFT, and their ability and effectiveness to be applied in the pre-processing of the electricity consumption data.

2.7.1 Traditional Techniques:

- **Fast Fourier transform (FFT)**: It is a coardinate transformation which projects your data to orthognal basis system with sines and cosines.So it covers all frequencies that could be contained in your data and assigns weight depending upon the importance of each frequency for the system that is represented by your data.This weight are amplitudes of your sines & cosines but it is valid for only linear and stationary data [35]. But since our real world data is non linear and non stationary so we can only obtain approximate result using fourier transform for this kind of data.
- **Wavelet transform:** It is extension of fourier transform and it additionaly contains temporal information.So instead of only telling which frequencies are important to your system.it also tell out at which instances they occur.This method is also mostly used on linear and stationary data.

2.7.2 Empirical Mode Decomposition

Introduction to EMD:

Empirical Mode Decomposition (EMD) is a signal processing technique that dynamically analyses non linear and non stationary data series through its data-based methodology. EMD represents a signal into a set of Intrinsic Mode Functions (IMFs) that are functions with equal number of zero-crossings and extrema and symmetrical envelopes defined by local maxima and minima [34].

Decomposition usually include coordinate transformation means that you take your data and project onto new basis system which allows you to decompose it in prescribed manner. EMD provides intrinsic mode functions (IMFS) and a residual. The sum of all the modes and the remaining residual forms the original signal as shown in Eq 9.

Equation 9:

$$f(t) = \sum_{0}^{n} IMF \ + Res$$

Methodology of EMD:

IMFs produced by EMD are obtained from an iterative process called sifting. Each IMFs are obtained through:

- **Identifying Extrema**: So, you mark all the local maxima and minima, since finding extrema of this signal is first step of algorithm. Main assumption of EMD is the characteristic's time scale of our signal. It is defined by the time lapse between the signal extrema. So, if you want to extract model representation of the signal that contains specific characteristics, temporal features of data. We must rely on these extrema.
- **Envelope Construction:** Fit maxima and minima to individual envelope using cubic spline function or any other interpolation method.

Equation 10:

$$Maxima = E_{up}(t)$$

Equation 11:

$$Minima = E_{low}(t)$$

- **Envelope mean calculation:** Determine mean of upper and lower envelope. Equation for mean calculation of envelope is given as:

Equation 12:

$$Emean(t) = (E_{up}(t) + E_{low}(t))/2$$

- **Detail Extraction (Residue):** The subtraction of the mean envelope from the original data is now done to extract the detail layer having characteristics of an IMF. Equation for detail extraction is given as:

Equation 13:

$$Res(t) = f(t) - E_{mean}(t)$$

If the extracted layer fits a particular condition (having symmetric zero crossing and extrema), it will count as an IMF. The Subtraction of an IMF from residue, serves as the input for the following iteration. The process continues until the residue contains no more harmonic content [34].

2.7.3 EMD comparison with traditional techniques

When compared with Fourier Transform (FT) and Wavelet Transform (WT), the Role of Empirical Mode Decomposition (EMD) stands out significantly. FT and WT follow the basis functions that can be either sines, cosines, or wavelet based. They also require the data to be periodically stationary or linear. On contrary, the EMD works on a totally different principle. EMD involves the assumption less approach and is not limited to the use of specific basis function which makes it more suitable for analysis of signals that show irregular dynamics. Additionally, EMD has gained a lot of interest because of its effectiveness and applicability. The method brought up the best results, with particular attention to the non-linear and non-

stationary data issues, which the traditional approaches tend to fail. The integration of EMD with a deep learning is a combination of EMD decomposition ability with the robust pattern recognition and predictive models deep learning features. Such a synergy is very powerful and can be leveraged in a variety of applications as diverse as biomedical engineering and forecasts of financial time series [36].

Liu et al. (2021) discussed integration of EMD with deep neural networks as a pre-processing step in mechanical fault diagnosis application area. EMD can be used to improve the accuracy of neural networks for the detection and classification of fault patterns making it very powerful diagnostic tool [30].

EMD was partnered with CNNs to perform analysis of EEG signals in the early detection of neurological diseases. They found that the combination detects the inherent dynamics of EEG data which allows more precision in the diagnosis of diseases like epilepsy [37].

2.8 Challenges and Limitation

Even though the forecasting of electric load is now done with the help of various advanced techniques but still it is facing many restrictions and challenges. The first issue is the fact that the demand for power is ever-changing and thus cannot be predicted. which is a result of the change in consumers' behavior, weather variability and economic fluctuations. To be precise, for the forecasters, modeling and uncertain mitigation are the difficult. The enormous problems which are faced due to data availability and quality, particularly in developing countries and markets with inadequate infrastructure are the biggest challenges that affect the data-driven decision. The wrong forecasting models and the bad decisions which might be caused from the unreliable data .Resolving this data gaps and improving the quality of data are the principal research objectives in load forecasting.

2.9 Conclusion

Although many advancements have been made, this area of research is still confronted by problems including reliability, data quality, and the agility of a system to practically adapt in the rapidly changing energy market. The review has also identified some of the research areas which have not yet been explored much, for example the utilization of latest technologies like transformers and hybrid models combining EMD with deep learning methods.

Transformer has shown really good results in Natural language processing (NLP) because of its ability to process data in parallel way that was a limitation in traditional recurrent neural networks. On the other side, it is still a lot of mixed assumptions about their efficacy in field of time series analysis.

This literature emphasizes that there is still lot of research needs to be done to overcome the current limitations and explore other opportunities in load forecasting. The use of new methodologies like transformers and deep learning methods combined with pre-processing techniques like EMD, is bound to open doors.

Chapter 3

3. Methodology

This chapter covers the methodologies used in this thesis to create a predictive model for load forecasting of a Residential building. It covers different machine learning models used including Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN), Linear regression & Transformers. Additionally, Bayesian optimization was utilized for hyper parameter tuning to enhance performance of models. Empirical Mode Decomposition (EMD) was used in parallel to train the model to see how it will impact the results and accuracy of model. This chapter provides a detailed discussion on each of these tools used throughout the modelling process.

3.1 Flowchart

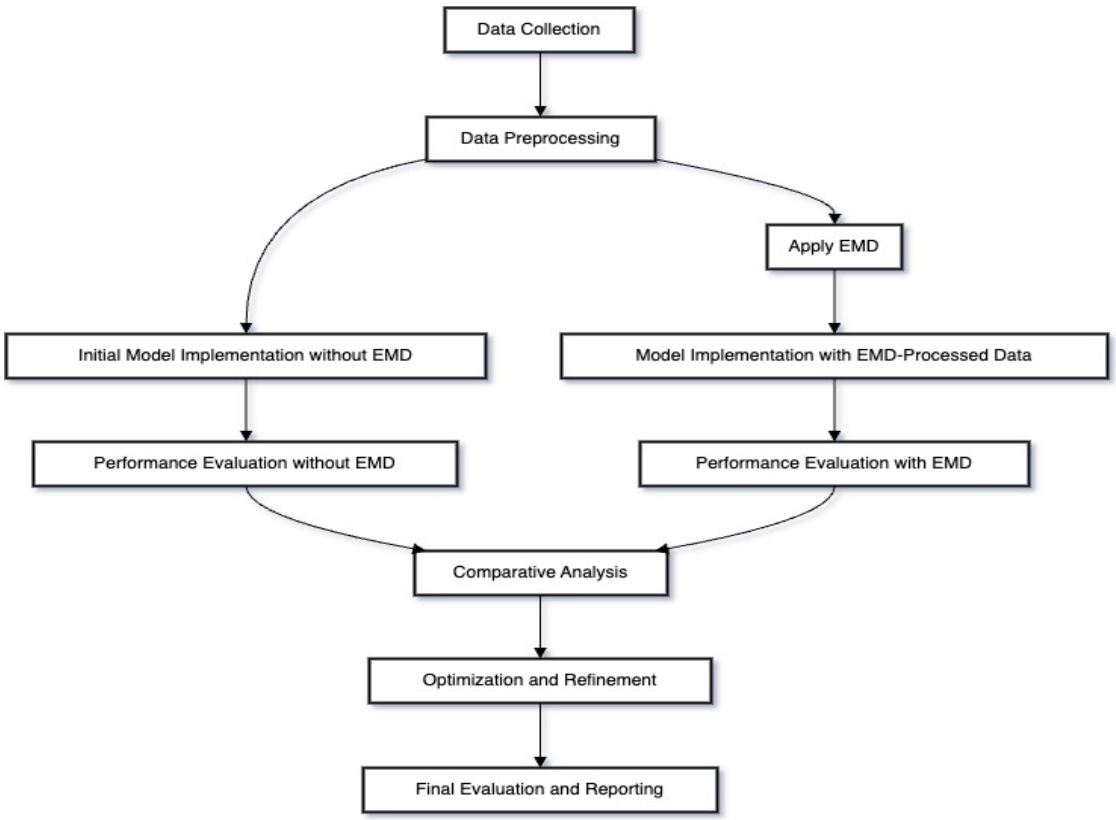

Figure 6: Methadology Flowchart

3.2 Experimental Setup

For the development of this model, I initially used MacBook Air M2 with 10 core GPU and 8gb of Ram. While training transformer model with increased volume of data it was struggling and was taking longer time, indicating that it was not suitable for this task. So, then I used HP Elite book Core (TM) i7 @ 3,5 GHZ with 32gb RAM and 4 GB of NVIDIA graphics card. This setup proved to be more capable than MacBook but still took longer time with large number of epochs.

3.3 Data Collection & Analysis

To Develop an effective forecasting model, it is very important to collect and analyse the data. In this case study, data has been collected from the smart meter of one of the residential houses in Narvik, Northern Norway. This Data set includes information regarding power consumption (measured in KWH) along with the additional variables such as time of the day, temperature, and wind speed [46]. This section will cover everything related to data preparation, pre-processing, and visualization. Data accuracy is very important, as any inaccuracy can impact model's predictions capabilities. Through data handling procedures, we aim to maintain the highest standard of data quality, setting a strong foundation for our forecasting model.

3.3.1 Power Consumption Data

Power consumption data is taken from energy meter data that was organized and stored on Elhub [40]. All the Norwegian households are required to have smart meters and Elhub is obliged to gather this data from smart metering devices. This data that I have used belongs to a detached house in Narvik [46]. This House is equipped with the following main devices:

- Mitsubishi Outlander PHEV with a 12KWH Lithium-ION battery, equipped with KW in house one-directional charger station.

- 200 Litre electric Hot water tank with 2 KW resistive heating element
- 1.5 KW Portable Electric Radiator
- Four floor electric heating cables
- Necessary Kitchen & Laundry appliances

Dataset includes hourly load consumption from April 1,2019 to March 31, 2020, along with information about temperature and wind at each hour having total 8785 data points. Figure 7 shows all the points and distribution of data. Hourly fluctuations are shown by grey bars, whereas daily and monthly trends are shown in blue and red lines. Impact of seasonal changes is easily visible from graph; high power consumptions and higher average consumptions can be seen during the winter period because of extensive use of heating appliances to combat the harsh cold. In Norway during winter times, residential energy demands surge as households employ heating systems to maintain comfort in severe cold, resulting in pronounced spikes seen in the graph. This seasonal impact stabilizes as winter transitions into spring or summer seasons and use of electric heating diminishes, and downtrend in red line on the graph can be seen.

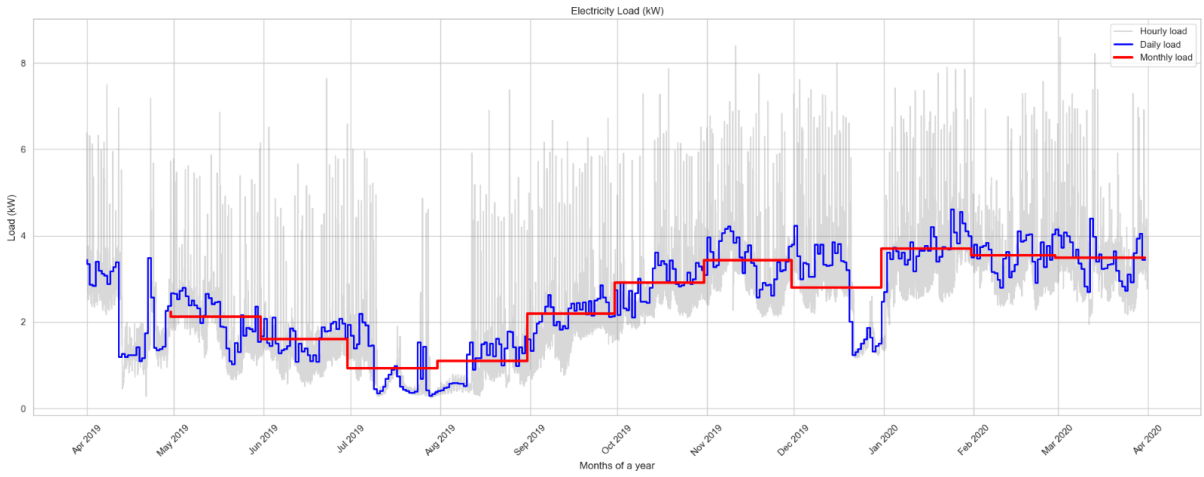

Figure 7: Residential Load Data Overview

3.3.2 Data Preprocessing

The preliminary process of pre-processing of the dataset took place at the beginning. This cleaning up, enhancement and partitioning of data guaranteed the quality and suitability for subsequent modelling.

Data Cleaning:

The first step in preprocessing framework is to clean and eliminate errors . Key tasks performed during this stage included:

- **Missing Values Handling:** Missing values scan was done on all variables in an effort to identify and work on any values that are missing. For missing values imputation techniques were used, among which mean imputation or interpolation were major approaches utilized to improve the data integrity of the dataset.
- **Outlier Detection and Treatment:** Outliers in the data was identified using the statistical methods or the knowledge of a specific area for faulty point detection. Such anomalous points were examined whether they should be corrected or omited.
- **Feauture Engineering:** Intensive feature engineering has to be done purposely to widen the set of existing data variables for the purpose of more detailed knowledge of the dataset.These features are really helpul in detecting the hidden patterns in the data set.Hence making model more efficient and reliable.
- **Lagged Variables:** For understanding the nature of the dataset, lags were introduced in the model to detect previous dependent historical patterns and temporal trends. By involving lagged versions of important variables, for instance, lagged power consumption or lagged environment features, we wanted to discover hidden connections as well as potential periodic tendencies among the data.
- **Interaction Terms:** To investigate the complex interconnection of variables, interaction terms were created by combining two or more variables. These interactions

modeled the underlying dependencies among power consumption, wind speed and temperature.

3.3.3 Data Splitting

To facilitate robust model training, validation, and evaluation, the dataset was partitioned into distinct subsets, each serving a specific purpose in the modelling framework:

- **Training Data (70%):** The biggest chunk of the data budget has been allocated to training the model. This accounts for 70% of the data. It was the part of the data that acted as the building blocks or the foundation upon which machine learning algorithms were trained to spot the patterns and relationships that existed in those data.
- **Validation Data (15%):** In a slightly smaller subset of data, which stands for 15%, the model was evaluated. We tested and refined the model using this dataset which helped us in tuning of hyper parameters and layered architecture to offer best performance and wide generalization capabilities.
- **Test Data (15%):** Lastly, the remaining 15% of the data set was held out for the final model evaluation. It was as an independent evaluation that this set was tested which provided information on the actual predictive ability of the trained models, revealing its real behavior and generalization.

Our goal was to create these datasets as a data partitioning method to strike a balance between model complexity, predictive accuracy and robustness.

3.3.4 Exploratory Data Analysis (EDA)

Data exploration is the foundation of the data research process that makes it possible for the researchers to understand patterns and trends, as well as relationships within the data set. The summary statistics and visualizations was our starting point that led us to extract essential insights about the data set.

Summary Statistics

The starting point of our EDA odyssey was the computation of summarizing statistics for all variables in the data set. The measures of central tendency, dispersion, and distribution indicators enabled us to get an overview of the nature of the data set being analyzed.

- **Mean:** A measure of central tendency reflecting the average value of the target variable for the entire dataset.
- **Median:** A strong index of central tendency, this value is the middle value for the variable when it is arranged in ascending order, providing insights into the data sets central tendency unaffected by outliers.
- **Standard Deviation:** The feature of divergence, which characterizes the degree of dispersion or spread within each variable.
- **Minimum and Maximum Values**: The extreme values of each variable delimited by the boundaries gives the overview of possible range of all values,offering insights into the data sets overall span.

Table 1 shows the statistical summary of load data, summarizing the average consumption, variability and range including the key quartile markers.

	Mean (KWH)	STD (KWH)	Min (KWH)	25 %	50%	75 %	Max (KWH)
Energy Consumption (KWH)	2.499459	1.4069	0.25	1.38325	2.4955	3.3	8.591

Table 1: Summary Statistics

These summary contains information about the central tendency, dispersion and coverage of the dataset.

3.3.5 Data Visualizations

The numerical summaries were supported by a vivid visualization that allowed the structure and the internal relationships within the dataset to be perceived more clearly. We used various

tools – histograms, line plots and correlation matrices to understand how different objectives were interconnected and interdependent.

- **Histograms:** Histograms gave a snapshot picture of the spread of each variable, revealing the dimension and the skew of each, as well as showing the modality. The frequency distribution of electrical load, temperature and wind speed was graphically presented. We discovered the hidden trends and patterns that explain these features.
- **Line Plots:** The changing line plot gave insight into the temporal variation of underlying characteristics across the study duration. The plot of power consumption, temperature and wind speed against time, reflects distinct trends, seasonality and periodic patterns contained in the dataset.
- **Correlation Matrices:** Correlation matrices provided us with an amazing way of quantifying the bivariate relationships among the variables. By measuring the level and direction of linear relationships through the correlation matrix, hidden patterns and interdependencies were revealed, which later guided the feature selection as well as model building efforts.

Impressive amounts of information from summary statistics and visualization were used in our journey. We managed to build a vivid picture of facts which is to be used for next step of the analysis.

3.4 Model Development

In this thesis, traditional machine learning techniques as well as the recent advances in machine learning models are combined which allow for the creation of predictive models that prove to be more dependable and accurate. Goal is to improve the accuracy of electric load forecasting by using a combination of LSTM, RNN, and transformer models together with the Empirical Mode Decomposition (EMD), to see how the use of EMD can affect the efficiency of the existing deep learning models.

3.4.1 Linear Regression

Linear regression is a machine learning based algorithm that is used to solve supervised machine learning problems. Linear regression provides us a proper framework to observe the different variables and data points and identify the relationship between these variables. It shows that what will be the impact on dependent variable if there is a change in independent variable. Equation for linear regression is given as:

Equation 14

$$Y = B_0 + B_1X_1 + B_2X_2 + \cdots + B_nX_n + \in$$

Where Y is independent variable and X_i is independent variable.

We used linear regression as a starting point for this study to first try the basic algorithm and then move toward the more advanced version. Despite their widespread use in forecasting, linear regression models possess limitations that can compromise their predictability. One such drawback is the assumption of linear relationships among explanatory variables and load demand, which may not always hold true in practice. Nonlinear deviations in data can affect the accuracy of predictions, particularly in handling complex patterns and interactions, especially with non-stationary or highly dynamic load patterns.

3.4.2 Long Short-Term Memory (LSTM)

LSTM is a type of recurrent neural network used to overcome vanishing gradient problem of RNN and to capture long term dependencies.LSTM has a different memory cells and gate that helps it to retain or forget the data depending upon its relevancy and importance.This gated mechanism makes LSTM more suitable for applications having long term dependencies such as time series forecasting, language processing and speech recognition.

The model training process, for the LSTM models, was very comprehensive, as it entailed the search for optimal hyperparameters and model architectures, which were mainly informed by robust experiments and validation.

Implementation Details:

- **Model Configuration:** For the LSTM model, a single layered architecture of 100 neurons was developed. To reduce the probability of overfitting, the dropout of 20% was applied after the LSTM layer and a dense output layer was added to predict the electric load.
- **Regularisation:** To reduce overfitting, we used L2 regularisation with a lambda value of 0.001 on the LSTM layer.
- **Learning and optimization:** Adam optimizer was used to train the model with a learning rate of 0.001, resulting in a reduced MSE loss function.
- **Training:** The model was trained over 200 epochs with a batch size of 100, using early stopping to discontinue training when the validation loss stopped decreasing.
- **Validation:** As a part of training, validation data of 20% was taken to observe and avoid the over-fitting effectively. Such an approach enables us to evaluate the model's efficacy on the part of the dataset that is not used during the training phase. It guarantees that the performance of the model is stable and valid and is not only localized to the measured data.

3.4.3 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) model was implemented with a dynamic architecture that was specifically adjusted for this time series forecasting application, and it is especially good at capturing the temporal dependencies of electricity load data. RNNs inherit the advantage of dealing sequential prediction problems because they retain previous data elements in their

memory, which stems from the retrospective nature of forecasting where historical information serves as the basis for predicting future events.

Implementation Details:

- **Configuration:** Single-layered RNN model with 200 neurons and 10% dropout was utilised to prevent overfitting, followed by a dense layer to forecast output.
- **Regularisation:** To reduce overfitting, we used L2 regularisation with a lambda value of 0.001 on the RNN layer.
- **Learning and optimization:** Adam optimizer was used to train the model with a learning rate of 0.0001, resulting in a reduced MSE loss function.
- **Training:** The model was trained over 200 epochs with a batch size of 60, using early stopping to discontinue training when the validation loss stopped decreasing.
- **Validation:** To prevent overfitting 20% validation split was employed during training.

The model was assessed in both short-term (1 week) and medium-term (1 month) horizons. Such a two-phase evaluation approach helped in understanding the effectiveness of the model, when it was used to forecast for different forecasting periods.

3.4.4 Transformer Machine Learning

In 2017 transformers artificial neural networks were introduced.The network employs an encoder and decoder architecture with positional encoding.It is pretty much like RNN but the difference is that input sequence can be passed in parallel which is not possible in RNN and LSTM.Transformers rely on self attention mechanism to remember things they don’t have any reccurents like RNN and LSTM,hence faster then both of them as data doesn’t need to be processed sequentially and can be processed in parallel way.

Implementation Details:

- **Configuration:** The Transformer architecture was designed with a great deal of focus given to sequential data handling. It uses multiple transformer blocks with 100 heads

in the multi-head attention layers, and a dropout rate of 20%. The model takes a feed-forward network structure and an hidden layer dimensionality of 4 and a dropout of 40% to protect from overfitting.

- **Optimizer and Regularization:**The training process utilized the Adam optimizer at 0.0001 learning rate. The lr of 0.0001 together with L2 regularization (lambda = 0.001) was utilized to control model complexity.
- **Training Process:** Model utilized 200 epochs for training with batch sizes set to 100, including an early stop mechanism that relies on validation loss to help optimize performance with high efficiency.
- **Validation:**As a part of training, validation data of 20% was taken to observe and avoid the over-fitting effectively. Such an approach enables us to evaluate the model's efficacy on the part of the dataset that is not used during the training phase. It guarantees that the performance of the model is stable and valid and is not only localized to the measured data.

3.5 Parameter Tuning and Regularization

Parameter tuning and regularization are key components during model development and training which allow model performance to be improved effectively and reduce cases of overfitting. The techniques for hyper-parameter tuning comprise grid search, random search, and Bayesian optimization under which the hyper-parameter space is systematically searched to identify the most efficient model configuration for the specified problem. Regularization techniques, which include tools such as dropout, L2 regularization, and early stopping, are used in preventing overfitting.

3.5.1 Parameters tuning using Bayesian Optimization

Bayesian optimization has become the most popular technique in the process of hyperparameter tuning used in artificial intelligence and data science applications. Bayesian optimization is superior to the two other types of search methods including the grid search and random search.

In order to identify the optimal hyperparameters for our predictive models, we implemented Bayesian Optimization. This technique optimizes the hyperparameters by building a probabilistic model that maps hyperparameters to the probability of a score on the objective function, which in our case is the minimization of the validation loss. The main hyperparameters involved in the model optimization via Bayesian Optimization include the learning rate, batch size and number of epochs.

In the optimization process there was a first exploratory stage with quite a lot of random trials to have the chance to create a variety of possible solutions at the beginning and afterwards a set of steps that are used to fine-tune the parameter after having gained insights from previous outcomes. This technique is more computationally efficient than the exhaustive grids searching, as it focuses all the evaluations on the hyperparameters which are more likely to result in the improvements hence, it boosts the model results and efficiency.

3.6 Model Training and Validation

Training phase followed by a rigorous process to make sure that models can learn correctly from the historical data, emphasizing the generalization capabilities to do well on unseen data. Proper strategy was followed to train and validate the models with the created datasets.

Training Process:

- **Data Splitting:** The datasets were then split into training and testing sections so the models could be tested critically. Large portion of data set was used for training the machines to have an accurate learning set.
- **Batch Processing:** Training was done in batches for the purpose of thoughtful consumption of memory and faster convergence of the model. Batch size was chosen through Bayesian optimization to strike a balance between the computational cost and the network efficiency.
- **Epochs:** Each model was trained iteratively with multiple epochs to create conditions that enable the models to both learn from the overall dataset and be exposed to the data more than once which is a very important feature since deep learning models greatly benefit from repeated exposure to data.
- **Early Stopping:** To avoid overfitting, early stopping was applied. It evaluates the model's performance on the validation set and stop the training when the performance no longer increases. This prevents the models from overlearning the training data.

3.7 Application of Empirical mode decomposition (EMD)

The pre-processing stage was synchronized with the Empirical Mode Decomposition (EMD) to enhance the forecasting accuracy of the models. EMD decomposes a signal into a series of Intrinsic Mode Functions (IMFs). This helps to identify hidden patterns in the electric load data which are not very easy to detect in raw data.

The application process started with the process of breaking down the original load data into several IMFs. Each IMF represents a simple oscillatory mode embedded in the signal, through which the models can detect and use more refined features of the load patterns. These IMFs were then integrated as the second feature beside the traditional inputs like temperature and

wind speed, thus enriching the input data of the models and providing a more detailed ground for learning complex load dynamics.

The use of IMFs was the element that contributed the most to the enhancement of the model's performance. It was the reason why the predictive models were more responsive to the small load variations; thus, the forecasts were more accurate, especially in the short term where the past data is essential. Moreover, the technique reduced the usual errors of forecasting that are normally related to MSE and MAE. Models were able to overcome the variability of the electric load data which was because of sudden change in the consumer behaviour or the weather events.

Thus, the use of EMD as a pre-processing technique was vital in the improvement of the forecasting models' sensitivity and accuracy, hence, confirming its effectiveness in the handling of the complexities of time series data in electric load forecasting.

3.8 Model Evaluation using Metrics

The performance evaluation of the forecasting models was one of the significant aspects of the process which was meant to check the efficiency of the implemented models. A complete methodology was applied, using different indicators to evaluate the accuracy and stability of the models in forecasting the electric load.

Key Metrics Employed:

- **Mean Absolute Error (MAE):** This metric is a measure of the overall size of the errors in a set of predictions, without considering their direction. It is of great value for grasping the average error magnitude at an abstract level.
- **Mean Squared Error (MSE):** MSE is mean of the squares of the errors, that is the average squared difference between the estimated values and the actual ones. The

metric is useful since it disproportionately penalizes larger errors over the smaller ones, therefore, the model performance is evaluated more strictly.

- **Root Mean Squared Error (RMSE):** RMSE is the square root of the mean of the squared errors. It is similar to MSE in terms of evaluation but is especially useful because it is in the same units as the response variable, hence, making it more interpretable in the context of the data.
- **R-squared (R^2):** This percentage reveals the degree of fit and thus a measure of how probable the prediction of the unseen samples is, compared to the mean of the actual data.

3.9 Comparison of Results

We did analysis of all the implemented predictive models, i. e. , Linear Regression, RNN, LSTM, and Transformers, that were checked with and without the EMD. Models were evaluated using different evaluation metrics to measure their accuracy. The comparison was made to show the effect of the application of the modern neural architectures and EMD. With the help of the visual aids such as the error distribution plots and the predictive accuracy graphs, the models effectiveness in different situations was demonstrated.

Chapter 4

4. Results and Discussion

4.1 Overview

In this chapter we will discuss all the results that were obtained after implementation of each ML model.After that we will discuss the results that were obtained after the implementation of EMD along with the ML models, showing their impact on each model and hence covering the key objective of our thesis.

4.2 Models Perfomance Evaluation

4.2.1 Linear Regression Results

Linear regression model was taken as a baseline to check how traditional models can perform on complex data.Linear regressions didn`t performed well in our case because of non stationary and non linear nature of residential load data.It showed higher errors in prediction as compare to Advanced ML models.

As shown in figure 8 and figure 9, Linear regression model was able to capture general trend but fail to capture rapid fluctuations.This result shows the limitation of linear regression to be used for non linear and non staionary data hence indicating the need for use of more advaced machine learning model.

Linear Regression model results were not very satisfactory and was used as a comparison for other models implemented in this work.

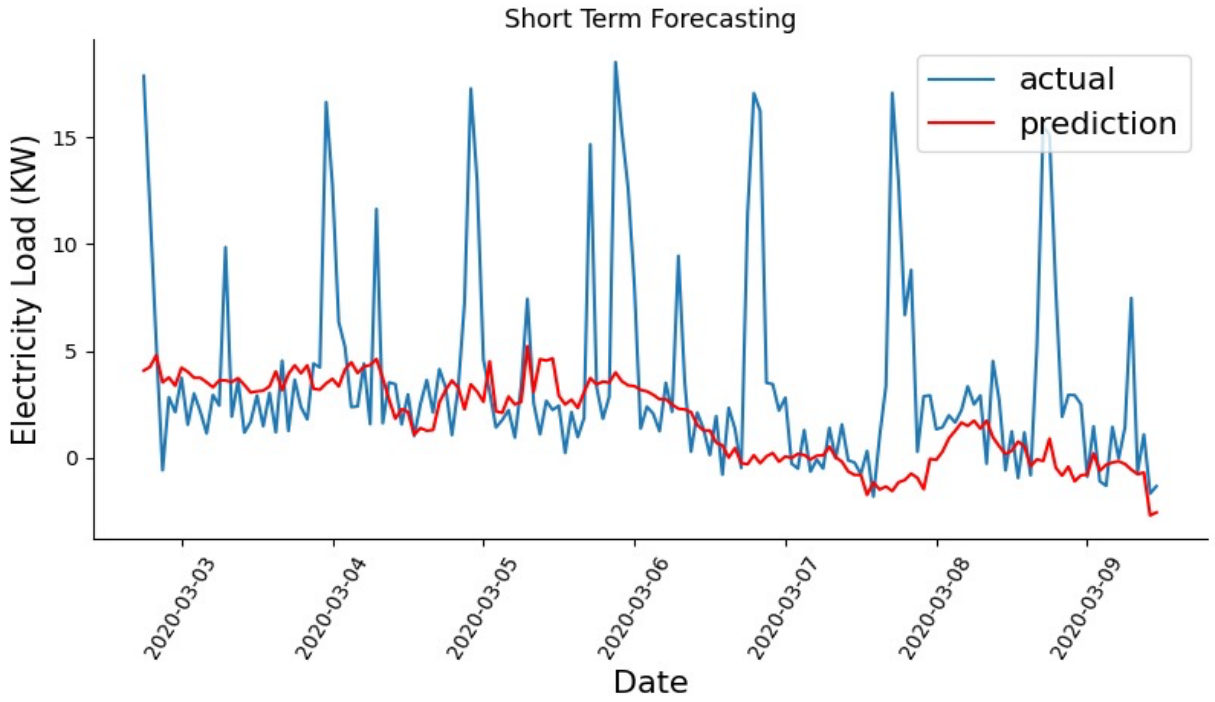

Figure 8: Predicted Vs Actual load by Linear Regression(1 Week)

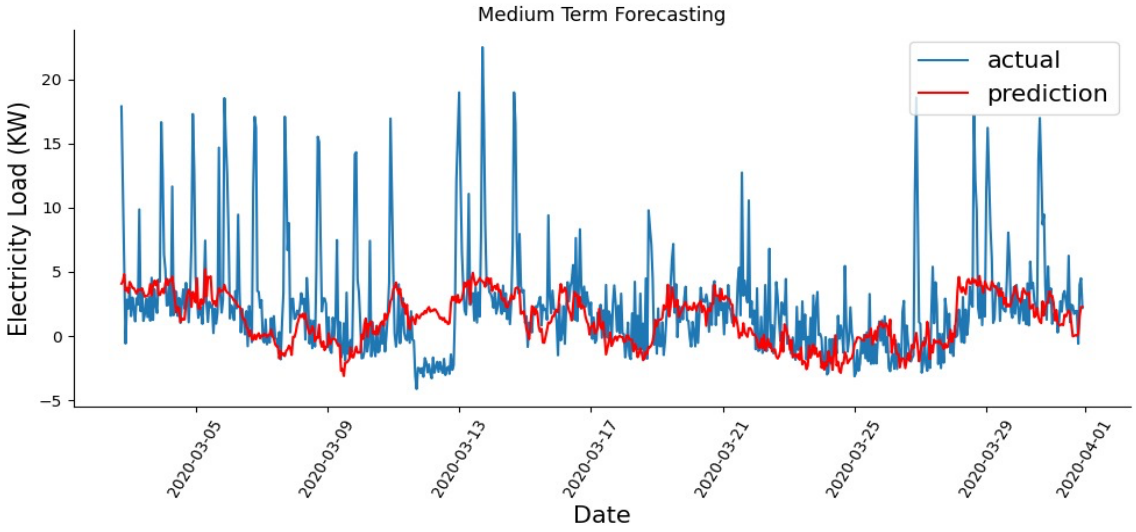

Figure 9: Predicted Vs Actual load by Linear Regression (1 Month)

4.2.2 Recurrent Neural Network(RNN) Results

This section shows the RNN model performance comparison for both the short term and medium term forecasting, and then it shows the model loss and error analysis for each of them.

- **Training and Validation Loss:** RNN model loss is shown in figure 10 over 150 epochs (with early stopping). The training and validation curve tends to converge smoothly indication good performance with minimal overfitting.

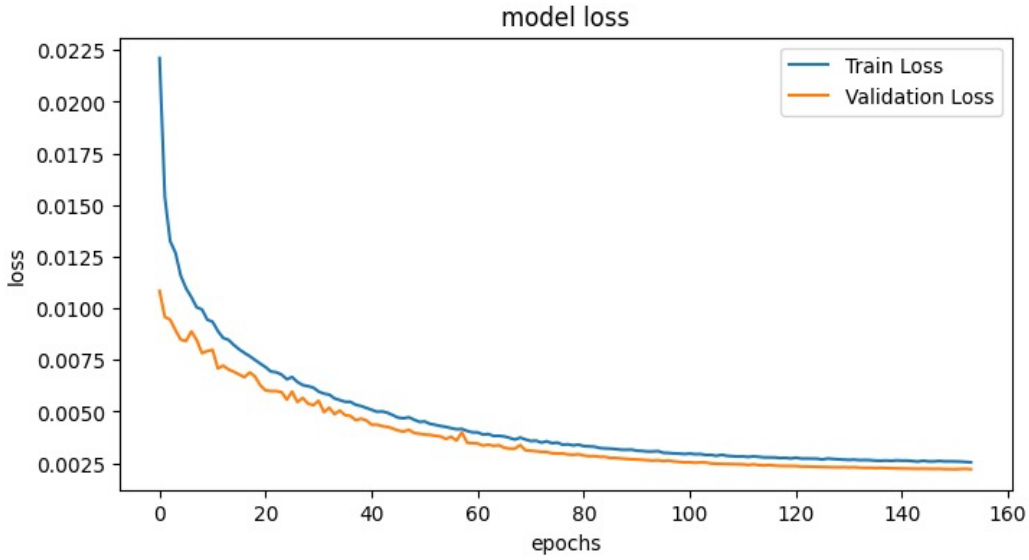

Figure 10 RNN Model Loss

- **Actual vs Predicted Load:** RNN model was capable of capturing the overall trend and seasonal fluctuation, but it shows a little deviation in predicting the peak loads. Result for 1 week prediction is shown in figure 11 and figure 12 shows the results for 1 month horizon.

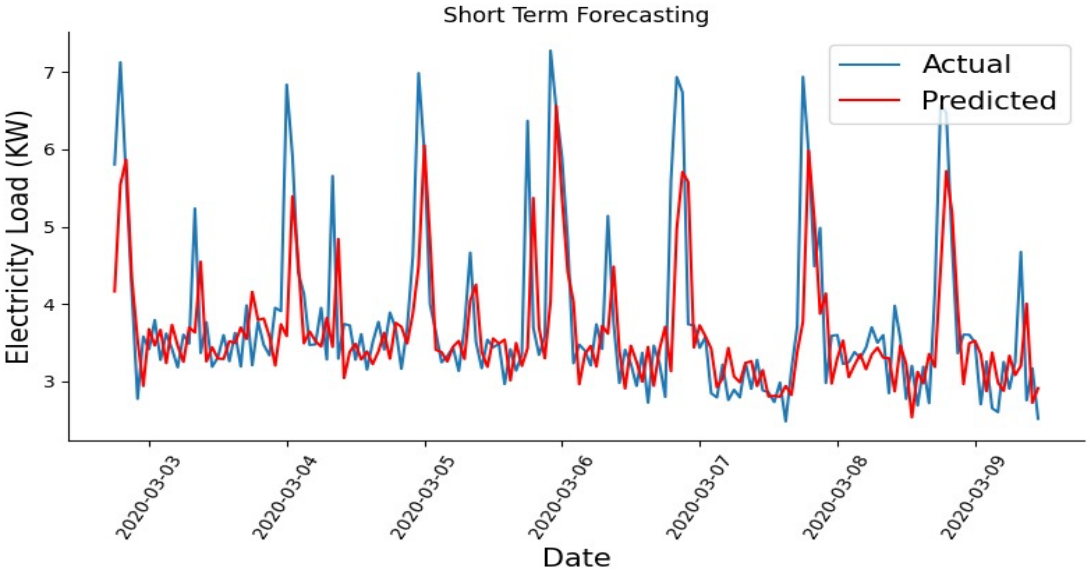

Figure 11 Actual Vs Predicted Results by RNN (1 Week Horizon)

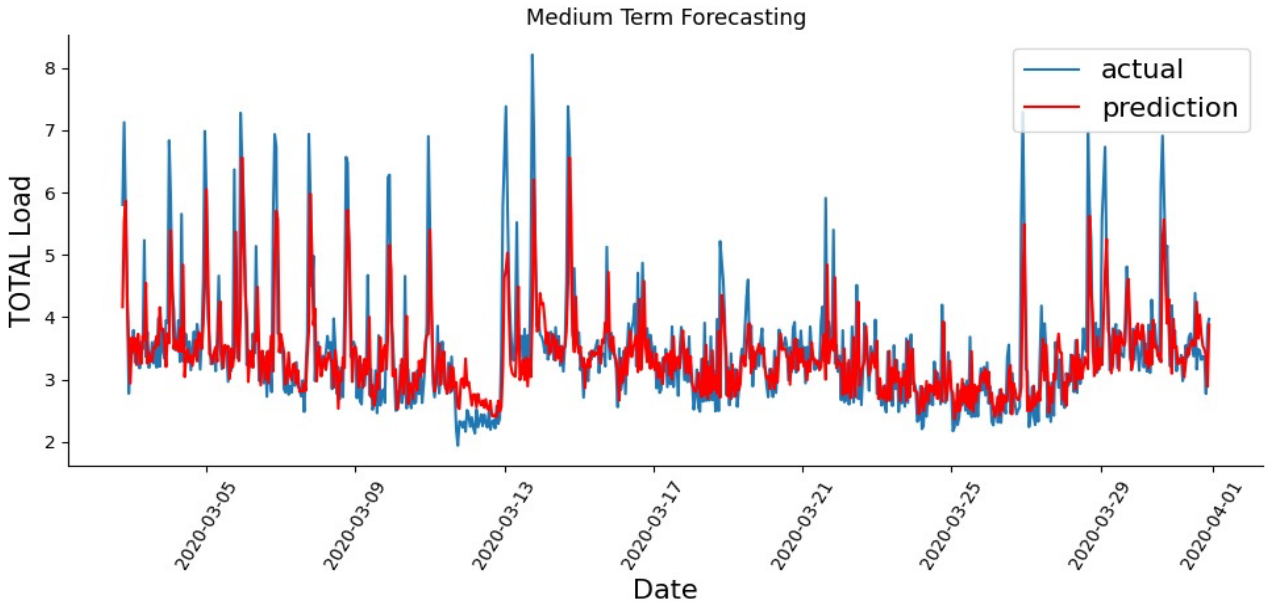

Figure 12: Actual Vs Predicted Load Results by RNN(1 Month horizon)

- **Interpretation of RNN Results:**

  RNN was used for both short term and medium-term forecasting application to check the capability of algorithm to perform with different time horizons. In short term forecast RNN showed a moderate performance with MAE of 0.573 and RMSE of 0.855 and having an R2 score of 0.368. While in medium term forecast models' performance was improved slightly with MAE of 0.521 and RMSE of 0.753 having R2 score of 0.382. The performance was reasonable but not very perfect fit.

Discussion

RNN model showed a very reasonable performance as compared to traditional linear regression model, but it can be further improved maybe with LSTM or transformers. Also, we are going to implement EMD on RNN to see if it can make any huge difference in term of prediction, which is main aim of our thesis.

4.2.2 LSTM Results

LSTM model was implemented in this thesis to take advantage from their ability to capture long term dependencies in load data. This section shows the RNN model performance comparison for both the short term and medium term forecasting, and then it shows the model loss and error metrics for each of them.

- **Training and Validation Loss**: LSTM model loss is shown in figure 6 over 160 epochs (with early stopping). The training and validation curve tends to converge smoothly indication good performance with minimal overfitting.

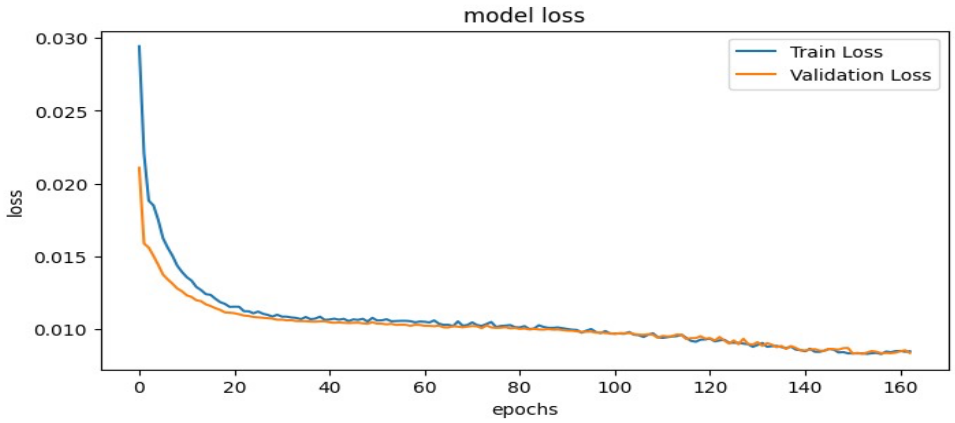

Figure 13: LSTM Model Loss

- **Actual vs Predicted Load:** LSTM model was able to capture the overall trend and seasonal fluctuation and was more stable in predicting the peak loads but still was not able to capture very rapid fluctuations in load data. Result for 1 week prediction is shown in figure 14 and figure 15 shows the results for 1 month horizon.

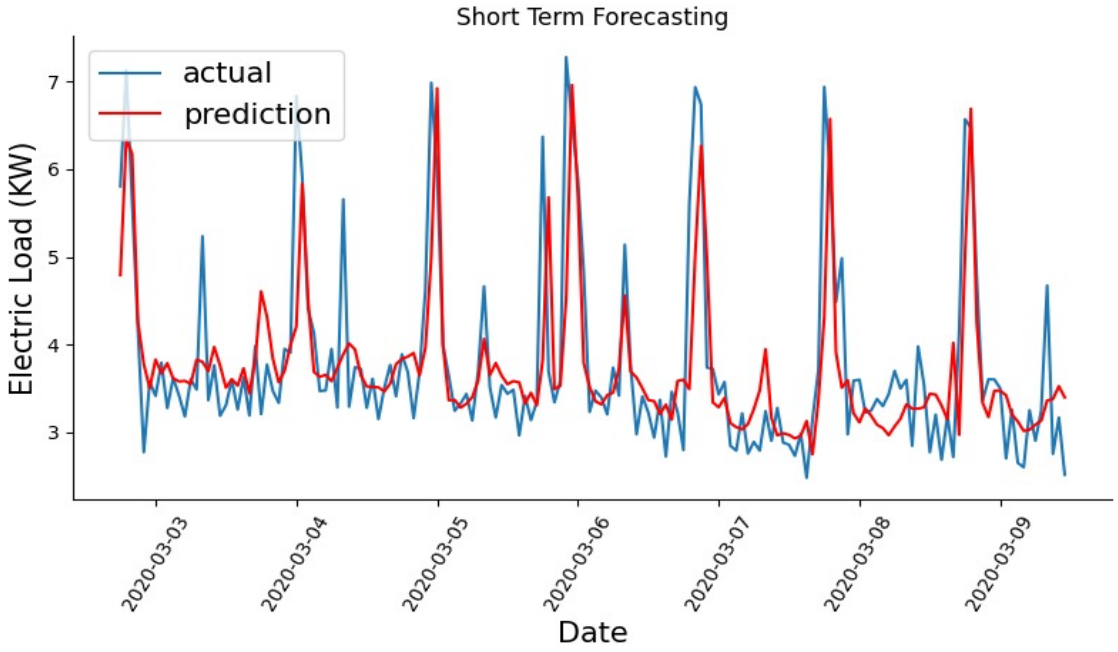

Figure 14: Actual vs Predicted Load By LSTM (1 week Horizon)

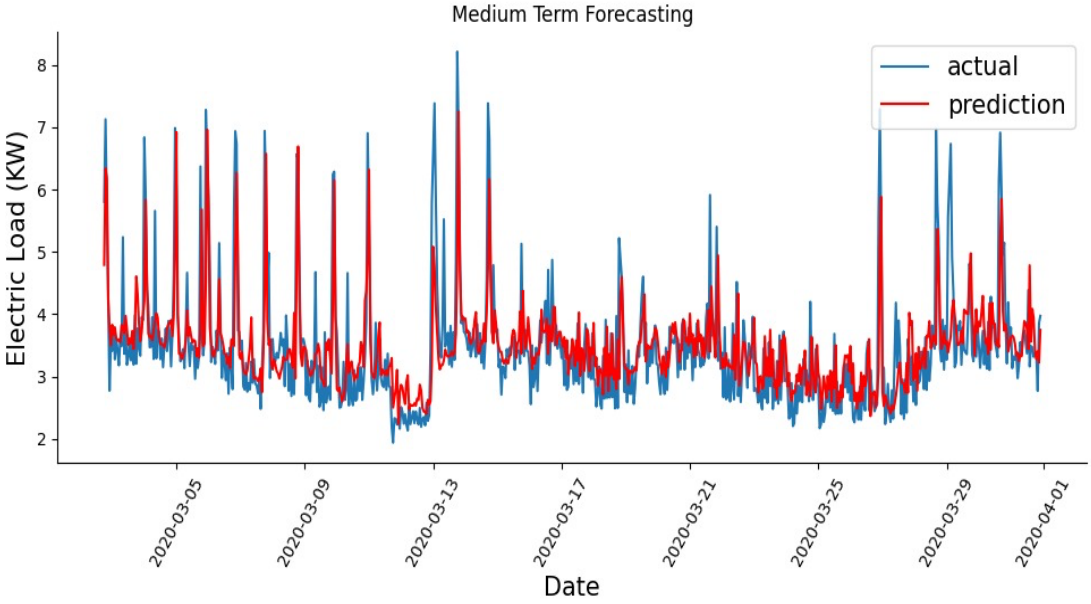

Figure 15: Actual Vs Predicted Load by LSTM (1 Month Horizon)

- **Interpretation of LSTM Results:**

  LSTM was also used for both short term and medium-term forecasting application to check the capability of algorithm to perform with different time horizons. In short term forecast LSTM achieved MAE of 0.521 and RMSE of 0.794 and having an R2 score of

0.454. While in medium term forecast performance was improved slightly with MAE of 0.507 and RMSE of 0.732 showing that model was preforming very well and was more stable over longer periods.

Discussion

LSTM model performance was slightly better than RNN due to gating mechanism in LSTM but still there still room for improvement. We will see if the newer transformer algorithm can perform better than LSTM. Also, in the end we will implement EMD on LSTM to see it impact on efficiency of LSTM.

4.2.3 Transformer Results

Transformers were launched in 2017 and is quite new in field of ML. It is mainly used in natural language processing and right now it is centre of attraction for all researcher to see if they can do a remarkable job in other applications as well. There is very limited research on this one and it is a debate among researchers if it is well suited for Time series forecasting or not. We used this ML model out of curiosity to check whether they can perform well on time series or not.

In this section we will see how the response of transformer in prediction of residential load forecasting and will have a discussion on the results that were obtained after the implementation.

- **Training and Validation Loss**: Transformer model loss is shown in figure 16 over 160 epochs (with early stopping). The training and validation curve tends to converge smoothly indication good performance with minimal overfitting.

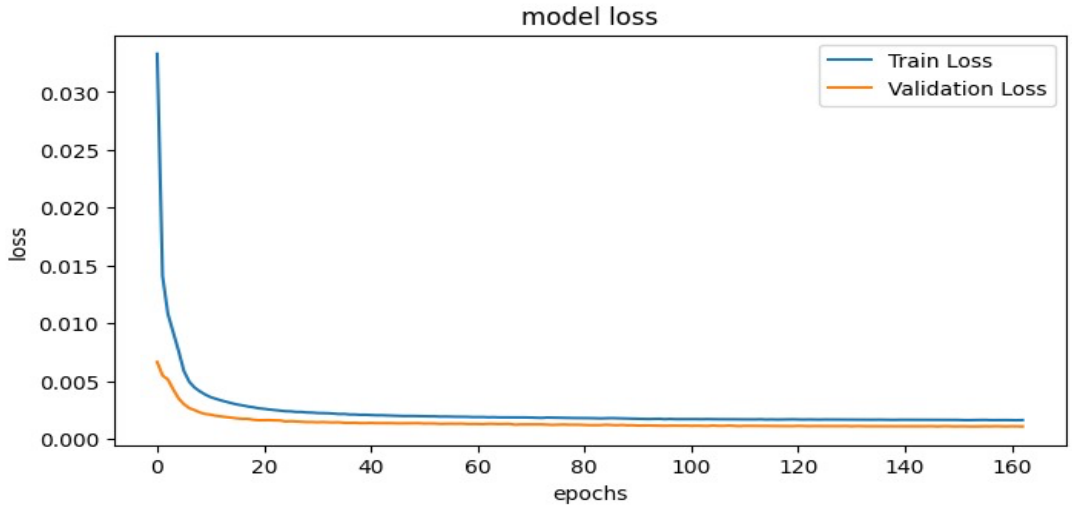

Figure 16: Transformer Model loss

- **Actual vs Predicted Load:** Transformer model was able to capture the overall trend and seasonal fluctuation and was more stable in predicting the peak loads but still was not able to capture very rapid fluctuations in load data. Result for 1 week prediction is shown in figure 17 and figure 18 shows the results for 1 month horizon.

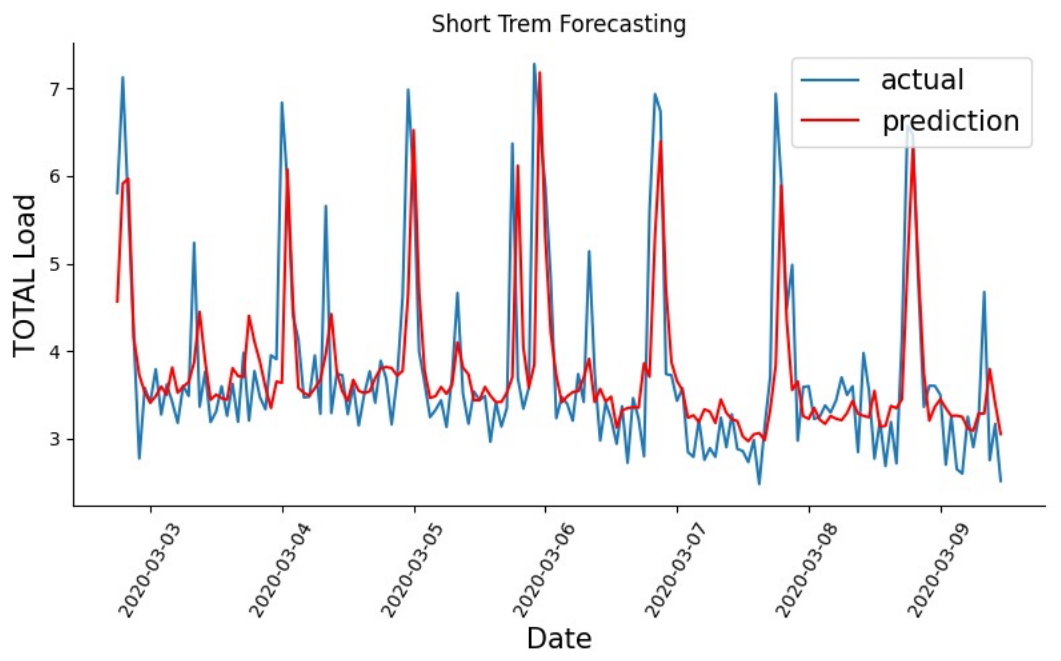

Figure 17: Actual vs Predicted Load by Transformers (1 Week Horizon)

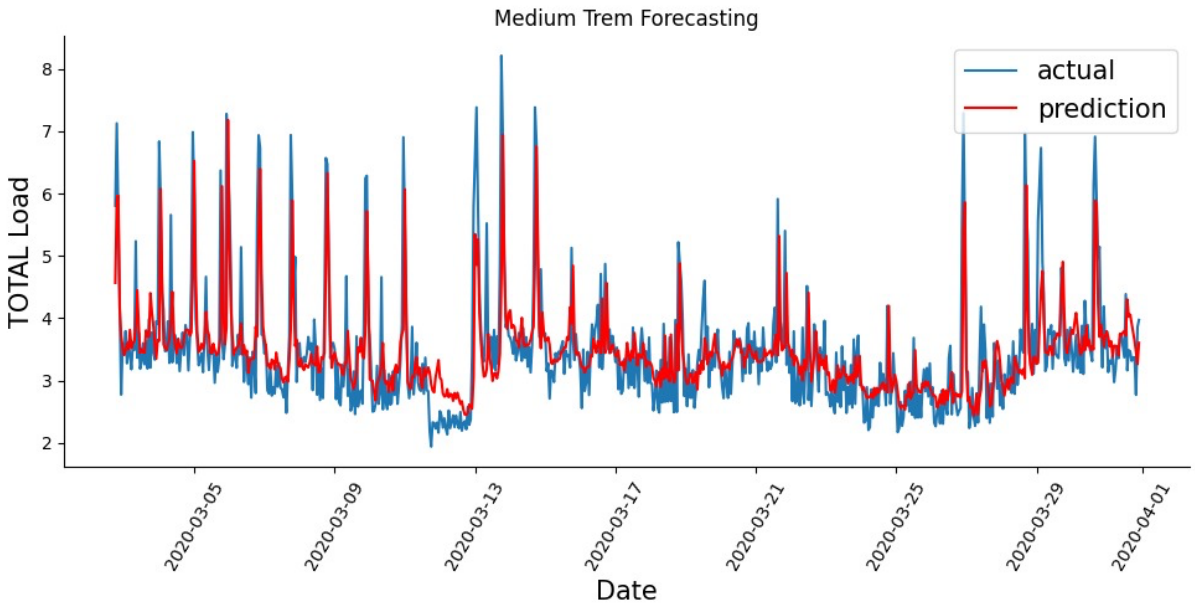

Figure 18 Actual vs Predicted Load by Transformer (1 Month Horizon)

- **Interpretation of Transformer Results:**

  Transformer was also used for both short-term and medium-term forecasting application to check the capability of algorithm to perform with different time horizons. In short term forecast Transformer achieved MAE of 0.483 and RMSE of 0.774 and having an R2 score of 0.482. While in medium term forecast performance was improved slightly with MAE of 0.456 and RMSE of 0.690 showing that model was preforming very well and was more stable over longer periods.

Comparison of transformers with RNN, LSTM:

Model	Forecast Horizon	Test MAE	Test RMSE	Test NRMSE	Test R^2 Score
RNN	Short-Term	0.5732	0.8552	0.2257	0.3687
RNN	Medium-Term	0.5214	0.7532	0.2180	0.3824
LSTM	Short-Term	0.5216	0.7948	0.2098	0.4548
LSTM	Medium-Term	0.5078	0.7321	0.2119	0.4166
Transformer	Short-Term	0.4840	0.7746	0.2045	0.4821
Transformer	Medium-Term	0.4562	0.6901	0.1997	0.4816

Table 2: Comparison of all the Models

Discussion:

Implementation of transformer was successfully done in this thesis, hence achieving another objective of using this novel technique. Performance of transformer was slightly better than RNN and LSTM but still there was not a very major difference as all of them were struggling with rapid fluctuations in load data. We will implement EMD to see if that can improve the accuracy of these models to get better results on sudden fluctuations due to user behaviour or seasonal changes.

4.6 Implementation of EMD with Advanced ML models

So, after implementation of all the models and seeing the results that we obtained. After that we implemented EMD on data to decompose it into several IMFS to extract the hidden temporal patterns that were affecting the predictions. After this we trained our models using the EMD IMFS and there was a huge improvement in the results.

Implementation of Empirical Mode Decomposition (EMD) for various forecasting models was mostly proved to increase their accuracy. We applied EMD to several models: Linear Regression (LR), RNN, LSTM, and Transformer. In each model, EMD was the most useful feature which led to the reduction of the error and increase of accuracy. It indicates that EMD makes complicated data simpler and generate more refined patterns that the models can easily learn and predict. These improvements were seen both in the short term and medium term forecasts showing the EMD ability to make our predictions more accurate across all models tested.

Figure 19 shows the IMFS that were decomposed from the data to extract the meaningful patterns. It shows 12 IMFS that were extracted through EMD iterative process. IMF no 12 is a residual it means that after this it is not possible to decompose it further, so EMD process is stopped. If we sum up all the IMFs with the residual, we can get the original signal back.

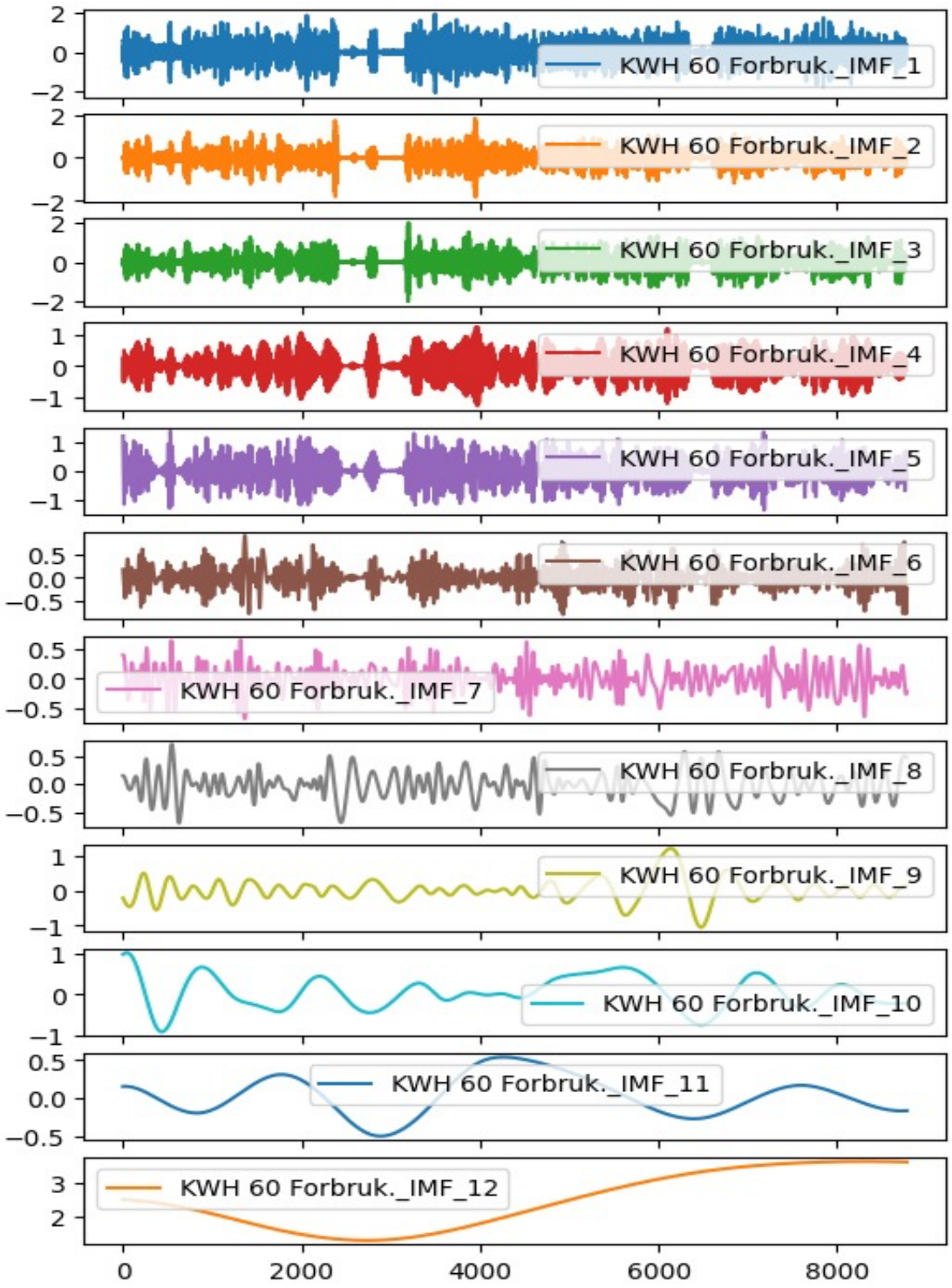

Figure 19: IMFS Decompostion By EMD

4.6.1 RNN Results with EMD

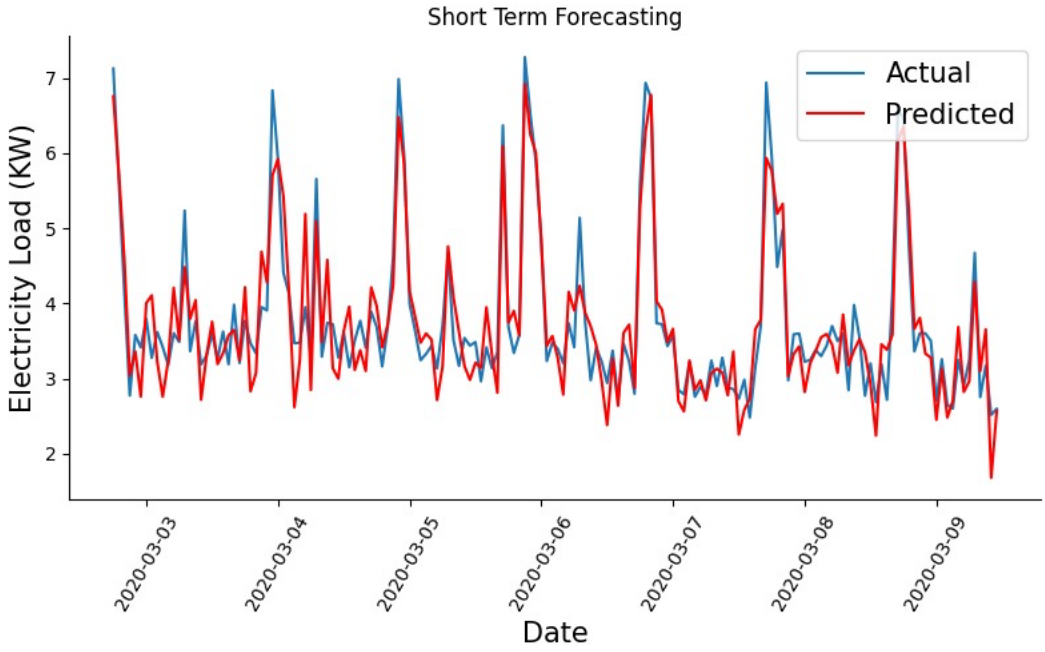

Figure 20: Actual vs Predicted Load Result by RNN using EMD (1 Week Horizon)

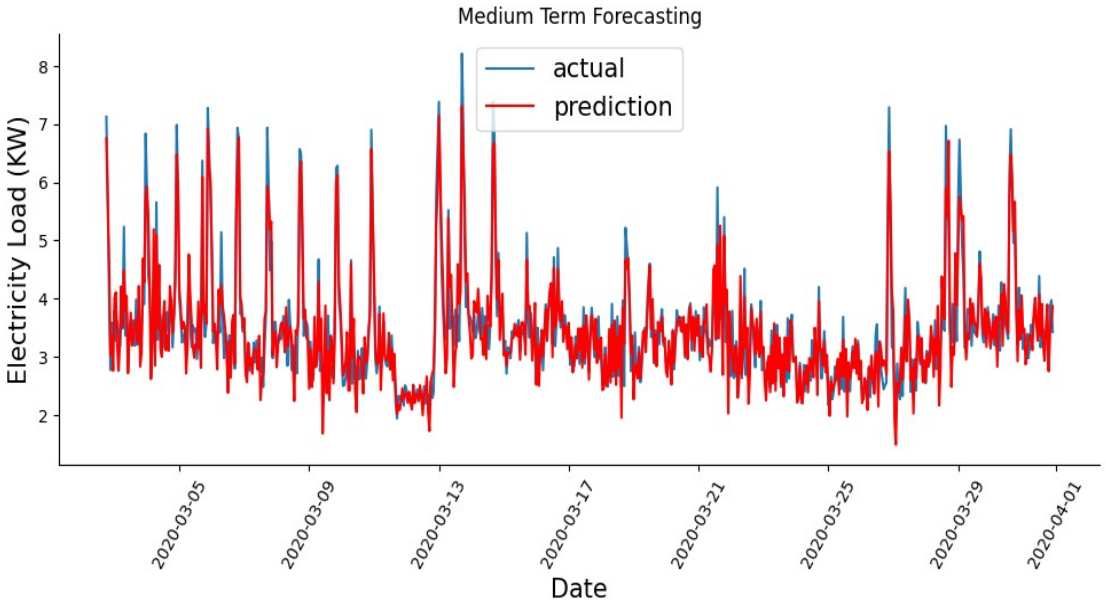

Figure 21: Actual vs Predicted Load Result by RNN using EMD (1 Month Horizon)

Figure 20 and 21 shows result obtained after the implementation of EMD. It is clearly visible now model was able to capture most of the sudden fluctuations. There was a bit of over or

under prediction in short term forecast as we had very limited data to train the model. We will see the effect of EMD on error metrics in comparison section having all the model's performance numbers.

4.6.2 LSTM Results with EMD

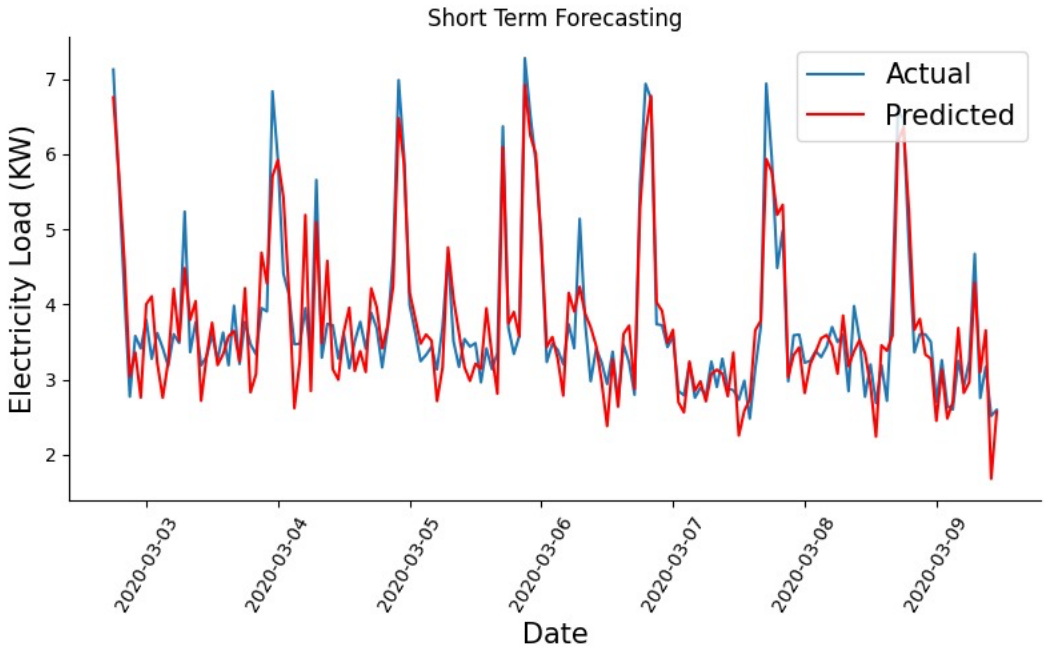

Figure 21:Actual vs Predicted Load by LSTM using EMD

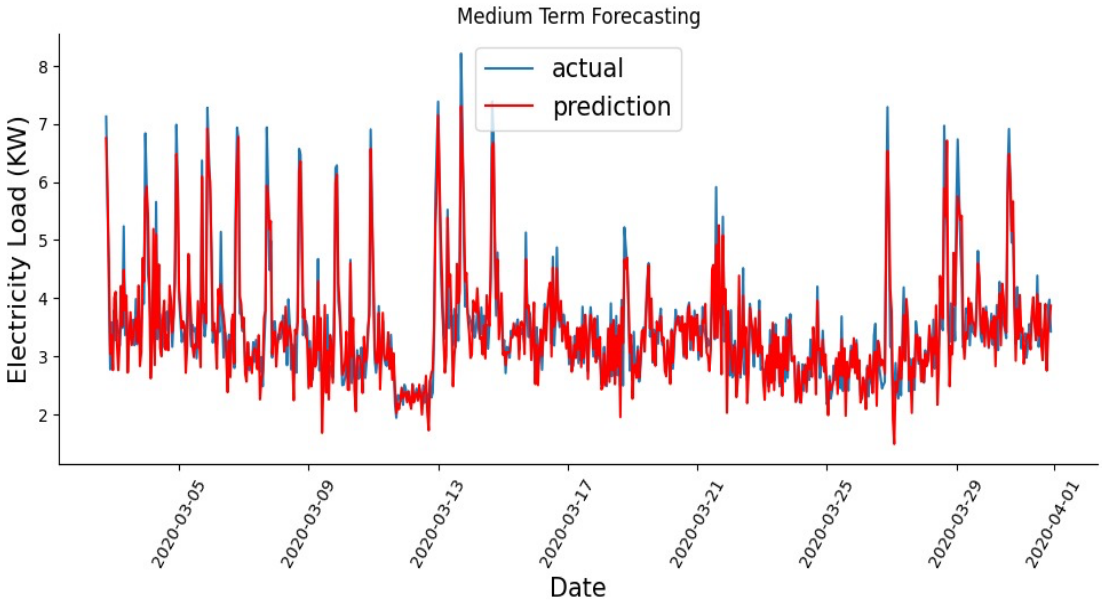

Figure 22:Actual vs Predicted Load by lstm using EMD (1 Month Horizon)

Figure 21 and 22 shows the result for LSTM implemented with EMD and it is clearly visible through graphs that they are preforming good with the rapid fluctuations due to user behaviour or seasonal change. LSTM performed better for both short term and medium-term forecasts. Performance of LSTM and RNN was almost the same with little bit of reduced error in LSTM.

4.6.3 Transformer Results with EMD:

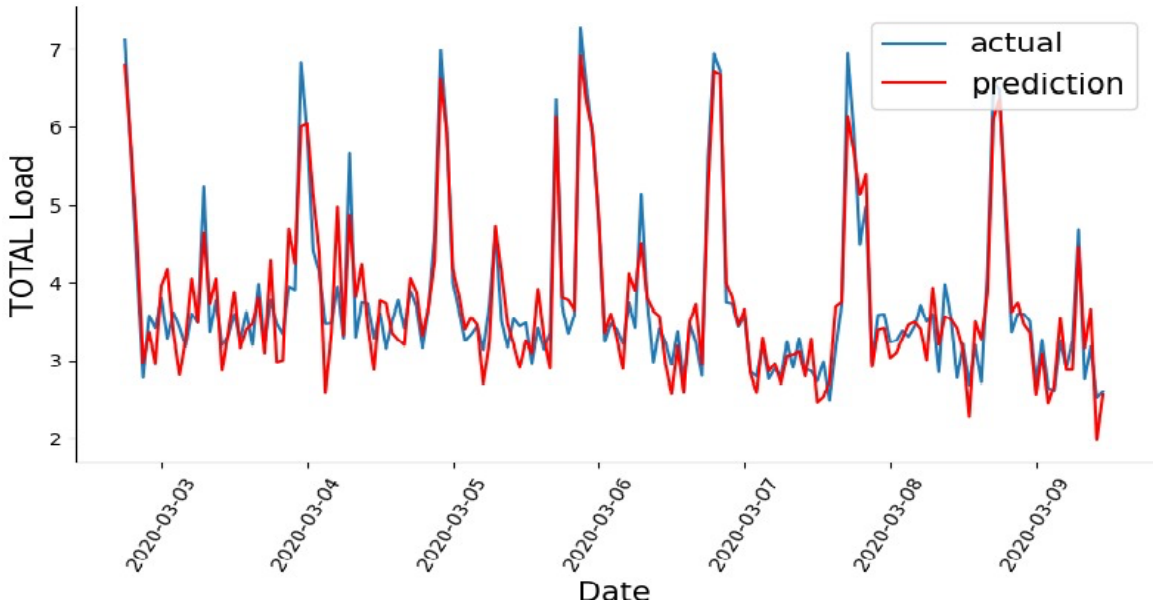

Figure 23: Actual vs Predicted load by Transformer (1 Week Horizon)

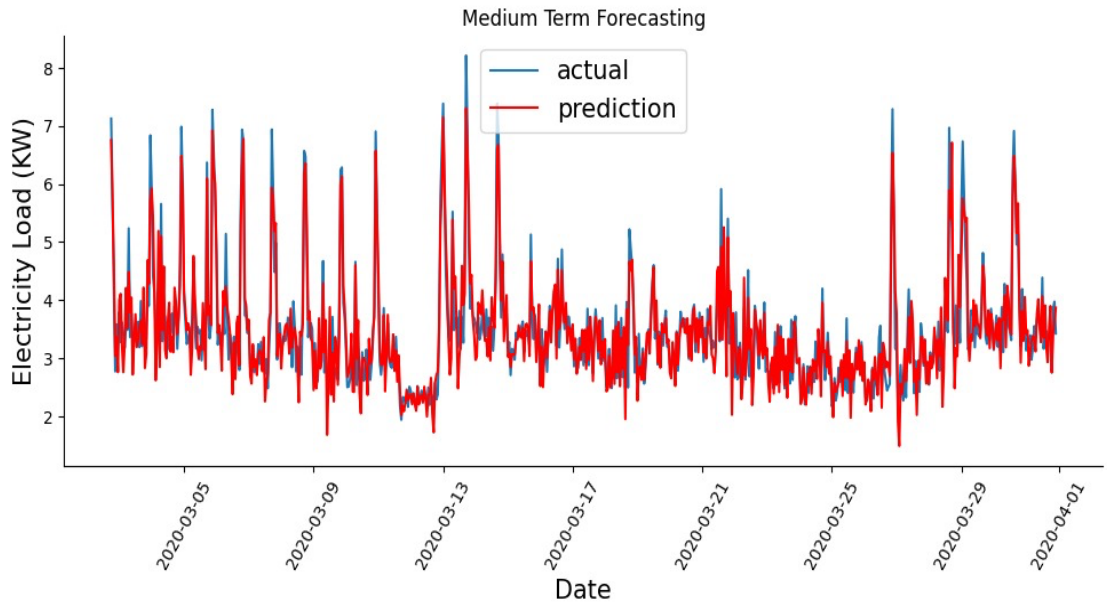

Figure 24: Actual vs Predicted Load By Transformers using EMD (1 Month Horizon)

Figure 23 and 24 shows the result that was obtained using the transformer models in combination with EMD pre-processing technique. Transformer performance was close to LSTM but had a better performance numbers and outperformed both LSTM and RNN.

4.7 Comprehensive Performance Comparison of Forecasting models

Model	EMD Applied	Forecast Horizon	Test MAE	Test RMSE	Test NRMSE	Test R^2 Score
RNN	No	Short-Term	0.5732	0.8552	0.2257	0.3687
RNN	No	Medium-Term	0.5214	0.7532	0.2180	0.3824
RNN with EMD	Yes	Short-Term	0.3321	0.4144	0.1100	0.8496
RNN with EMD	Yes	Medium-Term	0.2704	0.3562	0.1032	0.8607
LSTM	No	Short-Term	0.5216	0.7948	0.2098	0.4548
LSTM	No	Medium-Term	0.5078	0.7321	0.2119	0.4166
LSTM with EMD	Yes	Short-Term	0.3246	0.4053	0.1075	0.8562
LSTM with EMD	Yes	Medium-Term	0.2644	0.3481	0.1009	0.8670
Transformer	No	Short-Term	0.4840	0.7746	0.2045	0.4821
Transformer	No	Medium-Term	0.4562	0.6901	0.1997	0.4816
Transformer with EMD	Yes	Short-Term	0.2842	0.3525	0.0935	0.8912
Transformer with EMD	Yes	Medium-Term	0.2419	0.3224	0.0934	0.8859

Table 3: Comparison all the Implemented models

Key Insights:

- **Impact of EMD:** It is clearly visible from the results that implementation of EMD has shown a really good impact on perfomance of all the models by reducing the errors.
- **Model Efficiency:** All the models has shown improved perfomance but Transformer along with EMD outperformed all other models achieving lowest MAE of 0.241 and best R2 score of 0.885.

4.8 Findings and discussion

This thesis is build upon the earlier research done by Nasrin Kianpoor using same data set.In that research traditional neural networks like multilayer perceptron (MLP) and radial basis function neural networks were utlized for electric load forecasting [46]. MLP model achieved RMSE of 0.87 and RBF-NN model had an RMSE of 1.01 on test data [46].

In this theis we applied more advanced machine learning models along with novel pre-processing techniques to see what results can be achieved, by doing this hybrid combination.Findings from this thesis demonstrated a significiant improvement in results.Our tranformer model combined with EMD showed really great results achieving RMSE of 0.322 and MAE of 0.241.

In short advanced neural network with the combination of pre-processing techniques has shown a great imporvement in capabilites of predictiive models to predict non linear patterns more efficeintly.This advancement not only collaborates but significantly extends the previous findings using same data set to achieve better results.

Chapter 5

5. Conclusion and Future Work

In this thesis we implemented several advanced machine learning models including LSTM, RNN & Transformers and analyzed their perfomance on residential load data from one of the house in Narvik,Norway. We also used combination of Empirical Mode Decomposition (EMD) with these models to see the impact of this novel technique on ML model perfomance. Foundation for this thesis is based on research presented by Nasrin Kianpoor ,which used similar data set but was restricted to MLP and RBF-NN models.

Combination of EMD with ML models enhanced the perfomace of all the models that were tested. Transformers with EMD achieved the best results among all other models indicating lowest RMSE & MAE,hence providing highly accurate and reliable forecasts.This study not only highlighted the effectivness of EMD but also highlighted the capabilities of transformers to capture the complex and hidden patterns in non stationary data using it multihead attention mechanism.

These findings can be really helpful for energy management and planning.This enhanced forecasting accuracy can help us to optimize energy distribution and to manage the peak load demands more efficiently leading to cost saving and sustainability in energy systems.

5.1 Future Work

Despite of lot of enhancement and innovation in the field of electric load forecasting there are still some areas that needs attention.Some of them are:

- **Integration of Additional data sources:** This area is still unexplored as lot of researches are done on very limited data set and we cannot jutify those results based on that limited data provided to ML model.Hence we can look for some ways to combine

diverse data having different demography, economic indicators and seasonal patterns. This will enable us to grasp the concealed complex patterns in the data and thus, would be beneficial to construct a very reliable predictive model.

- **Real Time Forecasting Implementation:** Implementation of these model in real time world would give us the better evaluation about their practical use & reliability [51].
- **Cross Regional Studies:** Implementing the current developed model in different region would help us to understand the generalizability and scalability of existing model [30].We will have a better overview that how existing models perform when implemented in different demography having different seasonal patterns and social indicators.

REFERENCES

[1] Spencer, B., Alfandi, O., & Al-Obeidat, F. (2019). Forecasting temperature in a smart home with segmented linear regression. *Procedia Computer Science*, 155, 511-518. https://doi.org/10.1016/j.procs.2019.08.071

[2] Hastie, T.J. & Tibshirani, R.J.. (2017). Generalized Additive Models. 10.1201/9780203753781.

[3] Ahmad, Tanveer & Zhang, Dongdong. (2020). Novel Deep Regression and Stump Tree-Based Ensemble Models for Real-Time Load Demand Planning and Management. IEEE Access. 8. 48030-48048. 10.1109/ACCESS.2020.2978937

[4] Habbak, Hany & Mahmoud, Mohamed & Metwally, Khaled & Fouda, Mostafa & Ibrahem, Mohamed. (2023). Load Forecasting Techniques and Their Applications in Smart Grids. Energies. 16. 1480. 10.3390/en16031480.

[5] Wang, Tianfu & Fan, Qilin & Wang, Chao & Yang, Long & Ding, Leilei & Yuan, Nicholas & Xiong, Hui. (2024). FlagVNE: A Flexible and Generalizable Reinforcement Learning Framework for Network Resource Allocation.

[6] Evgeniou, Theodoros & Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7_12.

[7] Hastie, Trevor & Tibshirani, Robert & Friedman, Jerome. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics).

[8] Rokach, Lior & Maimon, Oded. (2005). Decision Trees. 10.1007/0-387-25465-X_9.

[9] Khan, Asad Ali & Fontenot, Hannah & Alamaniotis, Miltiadis & Prakash, Vishnu & Dong, Bing. (2020). Ensemble Method for Short-Term Load Forecasting Using Lstm, Svr, and Fnn and Taking into Account Seasonal Dependency (OR-20-C052).

[10] Ostertagova, Eva & Ostertag, Oskar. (2012). Forecasting Using Simple Exponential Smoothing Method. Acta Electrotechnica et Informatica. 12. 62–66. 10.2478/v10198-012-0034-2.

[11] Abumohsen, M., Owda, A. Y., & Owda, M. (2023). Electrical load forecasting using LSTM, GRU, and RNN algorithms. Energies, 16, 2283. https://doi.org/10.3390/en16052283

[12] Yin, Chunlin & Liu, Kaihua & Zhang, Qiangjian & Hu, Kai & Yang, Zheng & Yang, Li & Zhao, Na. (2023). SARIMA-Based Medium- and Long-Term Load Forecasting. Strategic Planning for Energy and the Environment. 10.13052/spee1048-5236.4222.

[13] Abumohsen, M., Owda, A. Y., & Owda, M. (2023). Electrical load forecasting using LSTM, GRU, and RNN algorithms. Energies, 16(5), 2283. https://doi.org/10.3390/en16052283

[14] Qiu, Xueheng & Ren, Ye & Suganthan, Ponnuthurai & Amaratunga, Gehan. (2017). Empirical Mode Decomposition based Ensemble Deep Learning for Load Demand Time Series Forecasting. Applied Soft Computing. 54. 10.1016/j.asoc.2017.01.015.

[15] Madhukumar, Mithun & Sebastian, Albino & Liang, Xiaodong & Jamil, Mohsin & Shabbir, Md. Nasmus Sakib Khan. (2022). Regression Model-Based Short-Term Load Forecasting for University Campus Load. IEEE Access. PP. 1-1. 10.1109/ACCESS.2022.3144206.

[16] Hong, Tao & Fan, Shu. (2016). Probabilistic electric load forecasting: A tutorial review. International Journal of Forecasting. 32. 10.1016/j.ijforecast.2015.11.011.

[17] Hu, Z., Lee, R. K.-W., Aggarwal, C. C., & Zhang, A. (2022). Text style transfer: a review and experimental evaluation. SIGKDD Explor. Newsl., 24(1). URL: https://doi.org/10.1145/3544903.3544906

[18] Hyndman, Rob & Koehler, Anne & Snyder, Ralph & Grose, Simone. (2002). A State Space Framework for Automatic Forecasting Using Exponential Smoothing Methods. International Journal of Forecasting. 18. 439-454. 10.1016/S0169-2070(01)00110-8.

[19] Box, George. (2013). Box and Jenkins: Time Series Analysis, Forecasting and Control. 10.1057/9781137291264_6.

[20] Zhang, Peter. (2003). Zhang, G.P.: Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model. Neurocomputing 50, 159-175. Neurocomputing. 50. 159-175. 10.1016/S0925-2312(01)00702-0.

[21] Naeem, Samreen & Ali, Aqib & Anam, Sania & Ahmed, Munawar. (2023). An Unsupervised Machine Learning Algorithms: Comprehensive Review. IJCDS Journal. 13. 911-921. 10.12785/ijcds/130172.

[22] Nasteski, Vladimir. (2017). An overview of the supervised machine learning methods. HORIZONS.B. 4. 51-62. 10.20544/HORIZONS.B.04.1.17.P05.

[23] Muzaffar, Shahzad & Afshari, Afshin. (2019). Short-Term Load Forecasts Using LSTM Networks. Energy Procedia. 158. 2922-2927. 10.1016/j.egypro.2019.01.952.

[24] Wankmüller, Sandra. (2021). Neural Transfer Learning with Transformers for Social Science Text Analysis.

[25] T, Thoyyibah & Haryono, Wasis & Zailani, Achmad & Djaksana, Yan & Rosmawarni, Neny & Arianti, Nunik. (2023). Transformers in Machine Learning: Literature Review. Jurnal Penelitian Pendidikan IPA. 9. 604-610. 10.29303/jppipa.v9i9.5040.

[26] Raza, Muhammad & Rind, Yousaf & Javed, Isma & Zubair, Muhammad & Mehmood, Muhammad Qasim & Massoud, Yehia. (2023). Smart Meters for Smart Energy: A Review of Business Intelligence Applications. IEEE Access. PP. 1-1. 10.1109/ACCESS.2023.3326724.

[27] Hor, Ching-Lai & Watson, Simon & Majithia, Shanti. (2006). Daily load forecasting and maximum demand estimation using ARMA and GARCH. 1 - 6. 10.1109/PMAPS.2006.360237.

[28] Pawar, Prakash & TarunKumar, Mudige & K, Panduranga. (2019). An IoT based Intelligent Smart Energy Management System with Accurate Forecasting and Load Strategy for Renewable Generation. Measurement. 152. 107187. 10.1016/j.measurement.2019.107187.

[29] Khalid, Samina & Shehryar, Tehmina & Nasreen, Shamila. (2014). A survey of feature selection and feature extraction techniques in machine learning. Proceedings of 2014 Science and Information Conference, SAI 2014. 372-378. 10.1109/SAI.2014.6918213.

[30] Quinn, Andrew & Lopes dos Santos, Vítor & Dupret, David & Nobre, Anna & Woolrich, Mark. (2021). EMD: Empirical Mode Decomposition and Hilbert-Huang Spectral Analyses in Python. Journal of Open Source Software. 6. 2977. 10.21105/joss.02977.

[31] Mellit, Adel & Kalogirou, Soteris & Hontoria, L. & Shaari, Sulaiman. (2009). Artificial intelligence techniques for sizing photovoltaic systems: A review. Renewable and Sustainable Energy Reviews. 13. 10.1016/j.rser.2008.01.006.

[32] Li, Guang & Lawarree, J. & Liu, Chen-Ching. (2010). State-of-the-Art of Electricity Price Forecasting in a Grid Environment. 10.1007/978-3-642-12686-4_6.

[34] Zeiler, Angela & Faltermeier, Rupert & Keck, Ingo & Tomé, Ana & Puntonet, Carlos & Lang, Elmar. (2010). Empirical Mode Decomposition - an introduction. Proceedings of the International Joint Conference on Neural Networks. 1-8. 10.1109/IJCNN.2010.5596829.

[35] Zhang, X.-Y & Guan, L. & Xie, J.-B. (2004). Short-term load forecasting based on artificial neural network and modeling with spectrum analysis. 28. 49-52.

[36] Qiu, Xueheng & Ren, Ye & Suganthan, Ponnuthurai & Amaratunga, Gehan. (2017). Empirical Mode Decomposition based Ensemble Deep Learning for Load Demand Time Series Forecasting. Applied Soft Computing. 54. 10.1016/j.asoc.2017.01.015.

[37] Ahsan, Md Manjurul & Siddique, Zahed. (2021). Machine learning based disease diagnosis: A comprehensive review.

[38] Yasin, Awais & Singh, Vishwa & Ahsan, Muhammad & Awan, Mazhar & Mubashar, Rida. (2022). Efficient residential load forecasting using deep learning approach. International Journal of Computer Applications in Technology. 68. 205. 10.1504/IJCAT.2022.10049745.

[39] Sarvepalli, Sarat Kumar. (2015). Deep Learning in Neural Networks: The science behind an Artificial Brain. 10.13140/RG.2.2.22512.71682.

[40] Deng, Li. 'Deep Learning: Methods And Applications'. Foundations and Trends in Signal Processing 7.3-4 (2013): 197-387. Web.

[41] Lipton, Zachary. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning.

[42] O'Shea, Keiron & Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

[43] Yuan, F.-N & Zhang, L. & Shi, J.-T & Xia, X. & Li, G.. (2019). Theories and Applications of Auto-Encoder Neural Networks: A Literature Survey. Jisuanji Xuebao/Chinese Journal of Computers. 42. 203-230. 10.11897/SP.J.1016.2019.00203.

[44] Fang, Jingying & Zhang, Xiangyun. (2019). Research on Power System Relay Protection Method Based on Machine Learning Algorithm. E3S Web of Conferences. 136. 02012. 10.1051/e3sconf/201913602012.

[45] Panda, Mrutyunjaya & Abraham, Ajith & Das, Swagatam & Patra, Manas. (2011). Network intrusion detection system: A machine learning approach. Intelligent Decision Technologies (IDT) Journal, IOS Press. 5. 347-356. 10.3233/IDT-2011-0117.

[46] Kianpoor, Nasrin & Hoff, Bjarte & Ostrem, Trond. (2021). Load modeling from smart meter data using neural network methods. 611-616. 10.1109/ICIT46573.2021.9453662.

[47] Wang X, Huang T, Zhu K, Zhao X. LSTM-Based Broad Learning System for Remaining Useful Life Prediction. Mathematics. 2022; 10(12):2066.

[48] Shohan MJA, Faruque MO, Foo SY. Forecasting of Electric Load Using a Hybrid LSTM-Neural Prophet Model. Energies. 2022; 15(6):2158.

[49] Rundo,. (2019). Deep LSTM with Reinforcement Learning Layer for Financial Trend Prediction in FX High Frequency Trading Systems. Applied Sciences. 9. 4460. 10.3390/app9204460.

[50] Alla, Sri & Athota, Kavitha. (2022). Brain Tumor Detection Using Transfer Learning in Deep Learning. Indian Journal Of Science And Technology. 15. 2093-2102. 10.17485/IJST/v15i40.1307.

[51] Aziz, M. (2023). Time Series Forecasting of Reactive Power Support from Smart Converters in SDN Using Machine Learning. Master’s thesis, Electrical Engineering Department

APPENDIX 1: EMD CODE

This code shown in appendix is just a overview it also has several interconnection with other parts in codes.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import matplotlib.ticker as tkr
from sklearn import preprocessing
%matplotlib inline
import math
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM,SimpleRNN
from keras.layers import Dropout
from keras.layers import *
from keras.optimizers import Adam
from keras.regularizers import l2
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from keras.callbacks import EarlyStopping

from PyEMD import EMD

emd = EMD()

column_name = 'KWH 60 Forbruk.'
column_to_decompose = df[column_name]

emd_results = emd(column_to_decompose.values)

for i, component in enumerate(emd_results):
    df[f'{column_name}_IMF_{i+1}'] = component


df.head()

def get_decompose_emd(forecasted,train_size,imf_columns=imf_columns,col_ind=1):
    t_df = df.iloc[train_size+25:]
    act = t_df[imf_columns].sum(axis=1)
    t_df[imf_columns[col_ind]] = forecasted
    pred = t_df[imf_columns].sum(axis=1)
    return act,pred
```

APPENDIX 2: RNN Code

```
def get_model(input_shape):
    model = Sequential()
    model.add(SimpleRNN(200, input_shape=input_shape))
    model.add(Dropout(0.1))
    model.add(Dense(1))
    return model
def fit_with(lr,batch_size,epoch):
    columns = ['KWH 60 Forbruk.','Temperature','is_holiday']
    X_train, X_test, Y_train, Y_test, scaler,train_size = get_data(df,columns)
    input_shape = (X_train.shape[1], X_train.shape[2])
    # Create the model using a specified hyperparameters.
    model = get_model(input_shape)

    # Train the model for a specified number of epochs.
    optimizer = Adam(learning_rate=lr)
    model.compile(loss='mean_squared_error',
                  optimizer=optimizer,
                  metrics=['mse'])

    # Train the model with the train dataset.
    model.fit(X_train, Y_train, epochs=int(epoch),
steps_per_epoch=5,batch_size=int(batch_size), verbose=2)

    # Evaluate the model with the eval dataset.
    score = model.evaluate(X_test, steps=10, verbose=0)
    print('Test loss:', score[0])
    print('Test accuracy:', score[1])

    # Return the accuracy.

    return score[1]

from functools import partial

verbose = 1

fit_with_partial = partial(fit_with)

def train_rnn(df,columns,lr=0.0001,batch_size=60,epochs=200,emd=False):
  X_train, X_test, Y_train, Y_test, scaler,train_size = get_data(df,columns)
  features_count = len(columns)
  l2_lambda = 0.001
  model = Sequential()
  model.add(SimpleRNN(200, input_shape=(X_train.shape[1], X_train.shape[2]),
kernel_regularizer=l2(l2_lambda)))
  model.add(Dropout(0.1))
  model.add(Dense(1))
  optimizer = Adam(learning_rate=lr)
  model.compile(loss='mean_squared_error',
```

```
                optimizer=optimizer,)

  callbacks = [keras.callbacks.EarlyStopping(patience=4,restore_best_weights=True)]
  history = model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_size,
validation_split=0.2,verbose=1, shuffle=False,callbacks=callbacks)

  plt.figure(figsize=(8,4))
  plt.plot(history.history['loss'], label='Train Loss')
  plt.plot(history.history['val_loss'], label='Validation Loss')
  plt.title('model loss')
plt.ylabel('loss')
  plt.xlabel('epochs')
  plt.legend(loc='upper right')
  plt.show()

  test_predict = model.predict(X_test)
  train_predict = model.predict(X_train)

  test_predict = scaler.inverse_transform(np.repeat(test_predict[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]
  Y_test = scaler.inverse_transform(np.repeat(Y_test[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]

  if emd:
    Y_test, test_predict = get_decompose_emd(test_predict,train_size)

  print('Test Mean Absolute Error:', mean_absolute_error(Y_test, test_predict))
  rmse = np.sqrt(mean_squared_error(Y_test, test_predict))
  print('Test Root Mean Squared Error:',rmse)
  nrmse = rmse / np.mean(Y_test)
  print(f"Test Normalized Root Mean Squared Error (NRMSE): {nrmse}")
  rscore = r2_score(Y_test,test_predict)
  print("Test R2 Score: {}".format(rscore))

  idx = 162 # For how many hours
  aa=[x for x in range(idx)]
  plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.title("Short Trem Forecasting")
  plt.legend(fontsize=15)
  idx = 700 # For
  aa=[x for x in range(idx)]
  plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
```

```
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
  plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.legend(fontsize=15)
  plt.title("Medium Trem Forecasting")
  plt.show()
train_rnn(df,['KWH 60 Forbruk.','Temperature','is_holiday'])


train_rnn(df,[imf_columns[1],'Temperature','is_holiday'],emd=True)
```

APPENDIX 3: LSTM CODE

```
def get_model(input_shape):
    model = Sequential()
    model.add(LSTM(100, input_shape=input_shape))
    model.add(Dropout(0.2))
    model.add(Dense(1))
    return model
def fit_with(lr,batch_size,epoch):
    columns = ['KWH 60 Forbruk.','Temperature','Wind']
    X_train, X_test, Y_train, Y_test, scaler,train_size = get_data(df,columns)
    input_shape = (X_train.shape[1], X_train.shape[2])
    # Create the model using a specified hyperparameters.
    model = get_model(input_shape)

    # Train the model for a specified number of epochs.
    optimizer = Adam(learning_rate=lr)
    model.compile(loss='mean_squared_error',
                  optimizer=optimizer,
                  metrics=['mse'])

    # Train the model with the train dataset.
    model.fit(X_train, Y_train, epochs=int(epoch),
steps_per_epoch=10,batch_size=int(batch_size), verbose=2)

    # Evaluate the model with the eval dataset.
    score = model.evaluate(X_test, steps=10, verbose=0)
    print('Test loss:', score[0])
    print('Test accuracy:', score[1])

    # Return the accuracy.

    return score[1]

from functools import partial

verbose = 1
```

```
fit_with_partial = partial(fit_with) def
train_lstm(df,columns,lr=0.001,batch_size=100,epochs=200,emd=False):
  X_train, X_test, Y_train, Y_test, scaler,train_size = get_data(df,columns)
  features_count = len(columns)
  l2_lambda = 0.001
  model = Sequential()
  model.add(LSTM(100, input_shape=(X_train.shape[1], X_train.shape[2]),
kernel_regularizer=l2(l2_lambda)))
  model.add(Dropout(0.2))
  model.add(Dense(1))
  optimizer = Adam(learning_rate=lr)
  model.compile(loss='mean_squared_error',
                optimizer=optimizer,)

  callbacks =
[keras.callbacks.EarlyStopping(patience=10,restore_best_weights=True)]

  history = model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_size,
validation_split=0.2,verbose=1, shuffle=False,callbacks=callbacks)

  plt.figure(figsize=(8,4))
  plt.plot(history.history['loss'], label='Train Loss')
  plt.plot(history.history['val_loss'], label='Validation Loss')
  plt.title('model loss')
  plt.ylabel('loss')
  plt.xlabel('epochs')
  plt.legend(loc='upper right')
  plt.show()

  test_predict = model.predict(X_test)
  train_predict = model.predict(X_train)
test_predict = scaler.inverse_transform(np.repeat(test_predict[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]
  Y_test = scaler.inverse_transform(np.repeat(Y_test[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]
  if emd:
    Y_test, test_predict = get_decompose_emd(test_predict,train_size)

  print('Test Mean Absolute Error:', mean_absolute_error(Y_test, test_predict))
  rmse = np.sqrt(mean_squared_error(Y_test, test_predict))
  print('Test Root Mean Squared Error:',rmse)
  nrmse = rmse / np.mean(Y_test)
  print(f"Test Normalized Root Mean Squared Error (NRMSE): {nrmse}")
  rscore = r2_score(Y_test,test_predict)
  print("Test R2 Score: {}".format(rscore))

  idx = 162 # For how many hours
  aa=[x for x in range(idx)]
  plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
```

```
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
  plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.title("Short Trem Forecasting")
  plt.legend(fontsize=15)
  idx = 700 # For
  aa=[x for x in range(idx)]
plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
  plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.legend(fontsize=15)
  plt.title("Medium Trem Forecasting")
  plt.show()

train_lstm(df,['KWH 60 Forbruk.','Temperature','Wind'])
train_lstm(df,[imf_columns[1],'Temperature','Wind'],emd=True)
```

APPENDIX 3: Transformer CODE

```
from tensorflow import keras
from tensorflow.keras import layers

def transformer_encoder(inputs, head_size, num_heads, ff_dim, dropout=0):
    # Normalization and Attention
    x = layers.LayerNormalization(epsilon=1e-6)(inputs)
    x = layers.MultiHeadAttention(
        key_dim=head_size, num_heads=num_heads, dropout=dropout
    )(x, x)
    x = layers.Dropout(dropout)(x)
    res = x + inputs

    # Feed Forward Part
    x = layers.LayerNormalization(epsilon=1e-6)(res)
    x = layers.Conv1D(filters=ff_dim, kernel_size=1, activation="relu")(res)
    x = layers.Dropout(dropout)(x)
    x = layers.Conv1D(filters=inputs.shape[-1], kernel_size=1)(x)
    return x + res
def build_model(
    input_shape,
    head_size,
    num_heads,
    ff_dim,
```

```
    num_transformer_blocks,
    mlp_units,
    dropout=0.2,
    mlp_dropout=0.1,
):
    inputs = keras.Input(shape=input_shape)
    x = inputs
    for _ in range(num_transformer_blocks):
        x = transformer_encoder(x, head_size, num_heads, ff_dim, dropout)

    x = layers.GlobalAveragePooling1D(data_format="channels_first")(x)
    for dim in mlp_units:
        x = layers.Dense(dim, activation="relu")(x)
        x = layers.Dropout(mlp_dropout)(x)
    outputs = layers.Dense(1)(x)
    return keras.Model(inputs, outputs)
def train_transformer(df,columns,lr=0.0001,batch_size=100,epochs=200,head_size=100,
      num_heads=4,
      ff_dim=4,
      num_transformer_blocks=2,
      mlp_units=[100],
      mlp_dropout=0.4,
      dropout=0.2,emd=False):



  X_train, X_test, Y_train, Y_test, scaler,train_size = get_data(df,columns)
  features_count = len(columns)

  input_shape = X_train.shape[1:]

  model = build_model(
      input_shape,
      head_size=head_size,
      num_heads=num_heads,
      ff_dim=ff_dim,
      num_transformer_blocks=num_transformer_blocks,
      mlp_units=mlp_units,
      mlp_dropout=mlp_dropout,
      dropout=dropout,
  )
  model.compile(loss="mean_squared_error",
optimizer=keras.optimizers.Adam(learning_rate=lr))

  callbacks =
[keras.callbacks.EarlyStopping(patience=10,restore_best_weights=True)]

  history = model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_size,
validation_data=(X_test,Y_test),verbose=1, shuffle=False,callbacks=callbacks)

  plt.figure(figsize=(8,4))
```

```
  plt.plot(history.history['loss'], label='Train Loss')
  plt.plot(history.history['val_loss'], label='Validation Loss')
  plt.title('model loss')
  plt.ylabel('loss')
  plt.xlabel('epochs')
  plt.legend(loc='upper right')
  plt.show()

  test_predict = model.predict(X_test)
  # train_predict = model.predict(X_train)

  test_predict = scaler.inverse_transform(np.repeat(test_predict[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]
  Y_test = scaler.inverse_transform(np.repeat(Y_test[:, np.newaxis],
features_count, axis=1).reshape(-1,features_count))[:,0]
  print(test_predict.shape,Y_test.shape)
  if emd:
    Y_test, test_predict = get_decompose_emd(test_predict,train_size)

  print(test_predict.shape,Y_test.shape)
  print('Test Mean Absolute Error:', mean_absolute_error(Y_test, test_predict))
  rmse = np.sqrt(mean_squared_error(Y_test, test_predict))
  print('Test Root Mean Squared Error:',rmse)
  nrmse = rmse / np.mean(Y_test)
  print(f"Test Normalized Root Mean Squared Error (NRMSE): {nrmse}")
  rscore = r2_score(Y_test,test_predict)
  print("Test R2 Score: {}".format(rscore))

  idx = 162 # For how many hours
  aa=[x for x in range(idx)]
  plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
  plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.title("Short Trem Forecasting")
  plt.legend(fontsize=15)
  idx = 700 # For
  aa=[x for x in range(idx)]
  plt.figure(figsize=(8,4))
  plt.plot(aa, Y_test[:idx], label="actual")
  plt.plot(aa, test_predict[:idx],color='r', label="prediction")
  plt.tight_layout()
  sns.despine(top=True)
  plt.subplots_adjust(left=0.07)
  plt.ylabel('TOTAL Load', size=15)
  plt.xlabel('Time step(hours)', size=15)
  plt.legend(fontsize=15)
```

```
  plt.title("Medium Trem Forecasting")
  plt.show()


train_transformer(df,['KWH 60 Forbruk.','Temperature','Wind'])
train_transformer(df,[imf_columns[1],'Temperature','Wind'],emd=True)
```