A Comparative study of various Machine Learning approaches in predicting EV charge demand at charging station

A Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Electrical Engineering

by
Chirag Srivastava (20202030)
Shreshtha Gupta (20206050)
Sandeep Oraon (20202072)

to the
ELECTRICAL ENGINEERING DEPARTMENT
MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD PRAYAGRAJ
December, 2023

UNDERTAKING

I declare that the work presented in this report titled "*A Comparative study of various Machine Learning approaches in predicting EV charge demand at charging station*", submitted to the Electrical Engineering Department, Motilal Nehru National Institute of Technology Allahabad, for the award of the ***Bachelor of Technology*** degree in ***Electrical Engineering***, is my original work. I have not plagiarized or submitted the same work for the award of any other degree. In case this undertaking is found incorrect, I accept that my degree may be unconditionally withdrawn.

December, 2023
Allahabad

Chirag Srivastava (20202030)
Shreshtha Gupta (20206050)
Sandeep Oraon (20202072)

CERTIFICATE

Certified that the work contained in the report titled "*A Comparative study of various Machine Learning approaches in predicting EV charge demand at charging station*", by *Chirag Srivastava (20202030), Shreshtha Gupta (20206050), Sandeep Oraon (20202072)*, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

(Dr. Navneet Kumar Singh)
Electrical Engineering Dept.
M.N.N.I.T, Allahabad

December, 2023

Preface

In this report, we will delve into the application of machine learning techniques to predict electric vehicle (EV) charging demand at charging stations. This endeavor involves training various machine learning models on historical charging patterns data and evaluating their performance on a reserved test set. Subsequently, we will compare the performance of these models to identify the most effective approach for EV charge demand prediction.

The overarching goal of this project is to develop a robust and accurate model for predicting EV charging demand. This model can empower charging station operators to optimize resource management and ensure adequate electricity supply to meet customer demands.

The significance of this project lies in its potential to enhance the reliability and convenience of EVs for consumers. Additionally, by accurately predicting EV charging demand, we can contribute to alleviating the strain on the power grid and improving its overall efficiency.

We are deeply indebted to Dr. Navneet Kumar Singh for his invaluable guidance and support throughout this research endeavor. His unwavering mentorship enabled us to grasp the fundamental concepts and progress to the stages of coding, testing, analysis, and comprehensive understanding of the project. We are truly grateful for his contributions to our learning and success.

Acknowledgements

It is a great honour for me to recognize the giants that I stand upon. My project supervisor, Dr. Navneet Kumar Singh, has provided me with invaluable guidance and support throughout this project, for which I am truly grateful. He was always a source of knowledge and support, and I am appreciative of his mentoring.
I also want to express my gratitude to Dr. R.S. Verma, our director, for giving us this chance to follow our research interests. I appreciate his leadership and vision.
In closing, I want to express my gratitude to each and every panellist for their thoughtfulness and time spent reviewing our project. I value their insightful criticism and observations.

Contents

Chapter 1

Introduction

The rapid adoption of electric vehicles (EVs) is transforming the transportation landscape, presenting both challenges and opportunities for the energy sector. One of the key challenges is managing the increasing demand for electricity at EV charging stations. To address this challenge, effective prediction of EV charging demand is essential. Accurate predictions enable charging station operators to optimize resource management, ensure adequate electricity supply, and enhance the overall customer experience.

In this report, we explore the application of machine learning techniques to predict EV charging demand at charging stations. We train and evaluate various machine learning models on a dataset of historical charging patterns and compare their performance to identify the most effective approach for EV charge demand prediction. The developed models can be utilized by charging station operators to make informed decisions regarding resource allocation, pricing strategies, and infrastructure planning.

1.1 Motivation

Electric vehicles (EVs) are becoming more and more popular due to government incentives and environmental concerns. Nonetheless, the current power grid is strained by the growing number of EVs on the road. Accurately estimating the amount of electricity needed to charge EVs at charging stations is crucial to ensuring that the grid can handle the increasing demand for electricity from EVs. There are several strong reasons to develop a reliable and accurate model for forecasting EV charging demand:

- **Improved Resource Management**: By accurately predicting EV charging demand, charging station operators can optimize their electricity consumption and infrastructure utilization. This translates to reduced operational costs, increased profitability, and a more sustainable approach to resource management.
- **Enhanced Customer Experience**: Accurate demand prediction enables charging station operators to maintain adequate charging capacity and reduce wait times for EV users. This enhances customer satisfaction, encourages repeat business, and promotes the adoption of EVs.
- **Grid Stability**:Predicting EV charging demand helps utilities better manage the load on the power grid and maintain stability during peak demand periods. This reduces the risk of strain on the grid, minimizes the need for expensive infrastructure upgrades, and contributes to a more resilient and efficient power system.
- **Sustainable Transportation**:Efficient EV charging infrastructure is essential for promoting the widespread adoption of EVs, which in turn contributes to a more sustainable transportation system. Accurate demand prediction plays a crucial role in ensuring that EV charging infrastructure can meet the growing demand, leading to reduced greenhouse gas emissions and improved air quality.

1.2 Objective

In the realm of analysis, prediction stands as a crucial concept, encompassing the act of making statements about future events or data based on available information and analysis. It involves utilizing past patterns, trends, and relationships to form informed judgments about what lies ahead. Predictions can range from simple guesses to sophisticated models, each tailored to the specific domain and context. This project aims to develop robust and accurate models for predicting EV charging demand at charging stations. By evaluating the performance of various machine learning approaches, including Transformers, N-BEATS, LSTM, and SVM models, we will identify the most effective method for prediction EV charge demand. Additionally, we will gain insights into the factors influencing EV charging patterns. By successfully predicting EV charging demand, we can contribute to a more efficient and sustainable transportation system.

Chapter 2

Literature Survey

The rapid adoption of electric vehicles (EVs) has brought about a surge in demand for charging infrastructure. To ensure the adequacy of charging infrastructure to meet this demand, accurate prediction of EV charge demands is crucial.

The majority of the methodologies for predicting the consumption of electrical energy are categorised into two, i.e., conventional statistical methods and ML methods. Classic statistical techniques rely on assumptions; however, they have the advantageous capability of interpreting a factor from estimated values. EV studies commonly use regression modelling [2], ARIMA modelling [4], and the Bayesian inference method [3]. Machine learning methods typically offer superior prediction accuracy, where the application of random forests [5][12] and support vector machines (SVM) [6][7][12] has been reported. In addition, a number of deep learning technologies have been investigated, such as long short-term memory (LSTM) [10][11], recurrent neural networks (RNN) [9][10], convolutional neural networks (CNN) [9], and artificial neural networks (ANN) [8].

The process of creating load prediction modes and the choice of input features become crucial because of the variety of loads and the intricacy of the influencing factors. A study[12] that employed SVR and Random forest techniques along with ensemble learning algorithms for studying EV charging behaviours found that, including factors such as climate, traffic and vehicular information produced significantly better results compared to previous studies for predicting the charging

scheduling at stations, but for the prediction of energy consumption the proposed models do not perform as expected. In 2006, Hinton et al. first proposed the deep learning concept [13]. Compared to traditional methods, deep learning techniques exhibit superior nonlinear learning capabilities, robustness, and generalization, especially when dealing with extensive data sets. Deep learning techniques could be used to train a model that could be used in complex, large-scale scenarios where only a few hyperparameter adjustments would be necessary to produce the desired results. The goal of long short-term memory, which was initially presented by Hochreiter et al., was to address the gradient vanish issue with the original RNN. In [1][11], the authors made use of LSTM and ANN-based techniques for EV load prediction based on various factors, including weather and days of the week[1]. The results of LSTM outperformed both RNN and ANN models on MAE, RMSE and R2 performance metrics for forecasting super-short-term EV charging load.The sequential processing of the input data, however, has a number of drawbacks, particularly when working with datasets that have lengthy dependencies. Particularly in cases where the training data is sparse, the models are prone to overfitting.

Transformer-based solutions have been developed to address the long-term dependencies problem in time series prediction [15]. Self-attention or the scaled dot-product operation is the primary learning mechanism of the Transformer model, which was put forth by A. Vaswani et al. in 2017 [14]. Examining the outcomes, it is clear that the Transformer model can be successfully applied to EV charging record datasets due to its strong generalization abilities. Furthermore, the Transformer model exhibits a higher capacity to adapt to modifications in the attributes of EV records in comparison withthe LSTM and RNN models.

In summary, the existing literature applies various conventional and deep learning algorithms and forecasting methodologies for prediction using available datasets. At present, attention mechanism-based neural network models, especially Transformer models, are rarely used for EV charging demand prediction. This paper fills this gap and further extends the existing method with time series forecasting using the N-Beats network[16]. N-BEATS is based on the principle of basis expansion. It can learn problem-specific functions for basis expansion, or it can be constrained to

have interpretable outputs. The paper will present a comparative study of different deep-learning approaches for electric vehicle charging load forecasting on the basis of various performance metrics to determine more accurate and optimised results for load prediction.

Chapter 3

Background

EV charging load prediction poses unique challenges due to the dynamic and stochastic nature of EV usage patterns. Factors such as weather conditions, traffic patterns, and driver behavior significantly influence EV charging demand, making prediction complex and unpredictable.

3.1 Time Series Prediction

Time series prediction is a crucial field in data science and machine learning that deals with prediction future values based on past observations. Time series data consists of a sequence of data points recorded or measured over successive points in time. This type of data is ubiquitous, ranging from financial markets and weather patterns to stock prices and industrial processes.

The significance of time series prediction lies in its ability to uncover hidden patterns, trends, and dependencies within temporal data. By analyzing historical information, predictive models can be built to anticipate future values, enabling informed decision-making and proactive planning. Applications of time series prediction span across various industries, including finance, healthcare, energy, and manufacturing.

3.2 Supervised Machine Learning

The main objective in machine learning (ML) is to develop a learning framework that can learn from experience, i.e., the training dataset, without explicit programming. Primarily, ML algorithms are classified as either supervised learning or unsupervised learning. Unsupervised machine learning (ML) aims to group similar data points together without labeling the training set. In supervised learning, on the other hand, the models are trained using a labeled dataset containing the desired output, also known as the target variable—that is, the variable that needs to be predicted. By optimizing a particular objective function, the representation between the input and target variables is learned iteratively. Because the target variables in this work—the length of the session and the amount of energy consumed—are both labeled, supervised learning will be applied. Furthermore, rather than using classification models, which deal with categorical target values, we will use regression models since both target variables are continuous values. The most commonly used regression models for this are XGBoost, SVM, RF, and deep ANN.

3.2.1 Support Vector Machine

Support Vector Machines (SVMs) are a versatile machine learning algorithm that can be employed for both classification and regression tasks. In the realm of time series prediction, SVRs can be utilized to predict future values of a time series based on historical data. Two primary methodologies exist for employing SVMs in time series prediction: Support vector regression (SVR) and support vector classification (SVC). SVR is a regression algorithm suitable for prediction continuous values, while SVC is a classification algorithm designed to predict binary values.

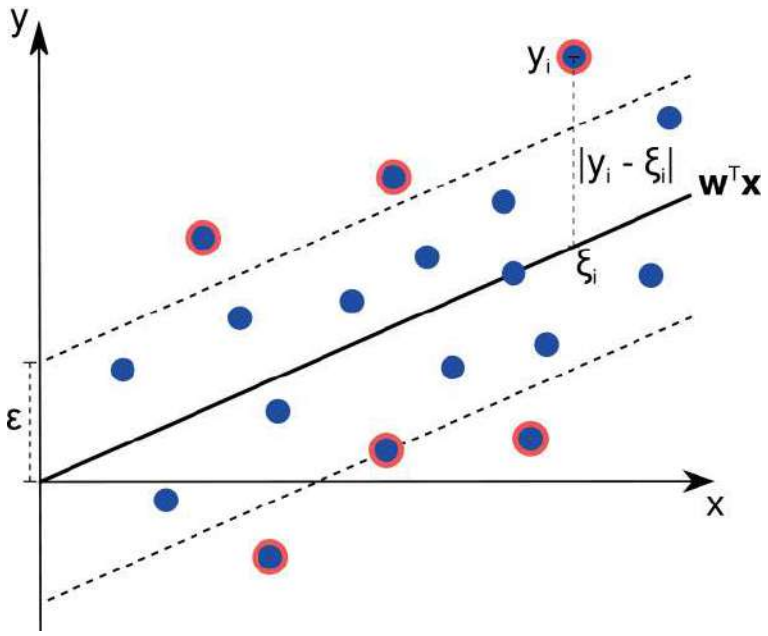


Fig.1 SVR

$$f(x) = w * x + b$$

$$\min_{\omega,b} \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}(\xi + \xi_i^*) + b$$

Where:

- (w) : *the weight vector, it defines the coefficients associated with each feature*
- (b) : *bias term*
- (ϵ) : *error margin in plane*
- (ξ) : *deviations beyond the error margin*
- (C) : *balances the trade − off between minimizing errors and having a smooth hyperplane*

SVRs offer several advantages for time series prediction, including their ability to capture nonlinear relationships in the data, their robustness to outliers, and their generalization capabilities. SVRs are computationally expensive to train, for large datasets, and require careful parameter selection.

3.2.2 Random Forest

Strong and adaptable, random forest is a machine learning technique that has attracted a lot of interest because of its outstanding results across a wide range of applications.Compared to conventional decision trees, it achieves higher predicted accuracy by utilizing the power of ensemble learning. The foundation of random forest is the idea of bagging, also known as bootstrap aggregating, which is building several decision trees out of bootstrap samples of the training set. Bagging is the process of using random sampling with replacement to divide the training data into smaller sections. To provide diversity among the decision trees, each one is trained using a different bootstrap sample.

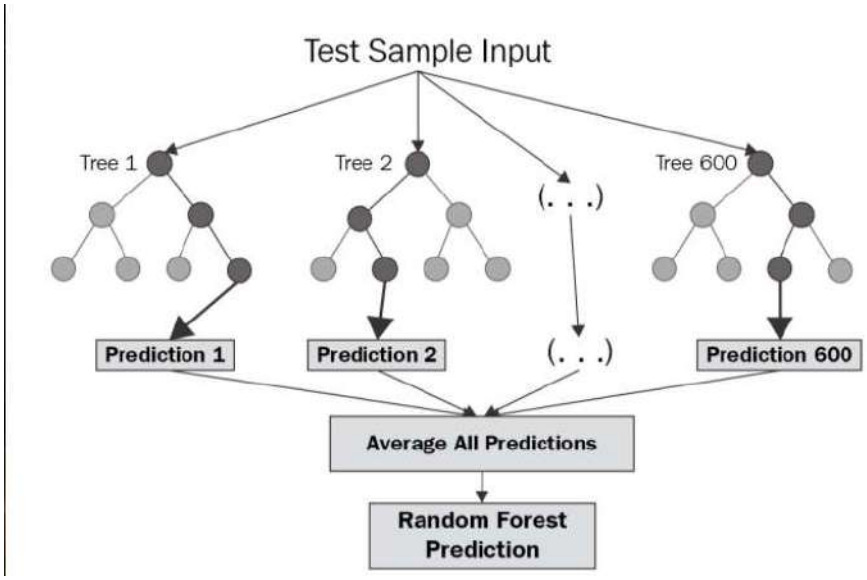


Fig.2 Random forest regressor

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2 (p_i)$$

$$GiniImpurity = 1 - \sum_{i=1}^{C} (p_i)^2$$

Where:

(p_i)	: *proportion of data points in the node belonging to class i*
$(GiniImpurity)$	: *probability of misclassifying a data point assigned to that node*
$(Entropy)$	: *average amount of information required to classify a data point*

Random forests boost the power of decision trees by adding randomness. Instead of blindly considering all features at each split, they randomly pick a smaller subset. This clever trick helps avoid overfitting and improves the forest's ability to generalize to unseen data.

3.2.3 XG Boost

Regression tasks suit XGBoost's potent ensemble learning algorithm well. To create a reliable and accurate model, it builds a series of weak learners, usually decision trees, and aggregates their predictions. Now let's examine each of XGBoost's main features in relation to regression.A regularized objective function, which is the product of a regularization term and the loss function, is minimized by XGBoost.The power of XGBoost resides in its capacity to manage intricate relationships in data, deal with missing values automatically, and stop overfitting through regularization. Its ability to produce reliable and accurate regression models is demonstrated by its popularity in machine learning competitions.

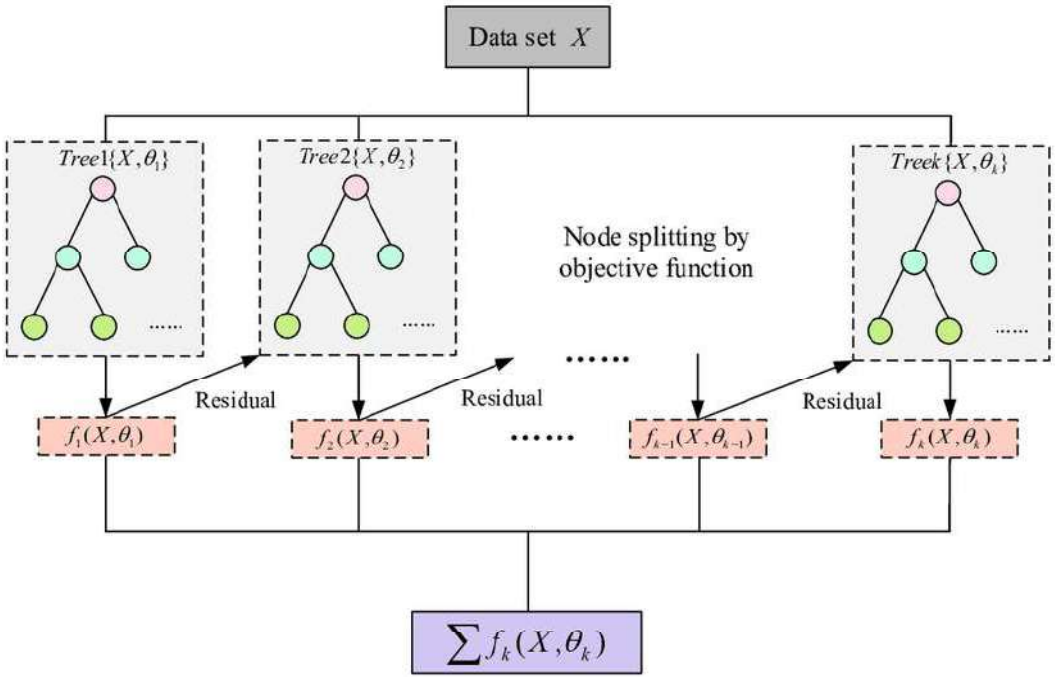


Fig.3 XG Boost Regressor

$$Obj(\theta) = \sum_{i=1}^{n} L\left(\dot{y}_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega\left(f_k\right)$$

$$\Omega\left(f_k\right) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2$$

Where:

(y_i)	: *true value*
$(\hat{y}_i)$	: *predicted value by model*
$(\Omega\left(f_k\right))$	: *Regularizing term*
(T)	: *number of trees in leaves in tree*
(γ, λ)	: *regularizing parameters*

3.3 Deep Learning

Artificial neural networks are used in deep learning, a kind of machine learning, to extract knowledge from data. Inspired by the architecture and operation of the human brain, neural networks are capable of extracting intricate patterns from data. Conversely, deep learning models don't require human assistance and may extract features straight from the data. This explains why deep learning models have demonstrated such remarkable performance across an array of tasks.

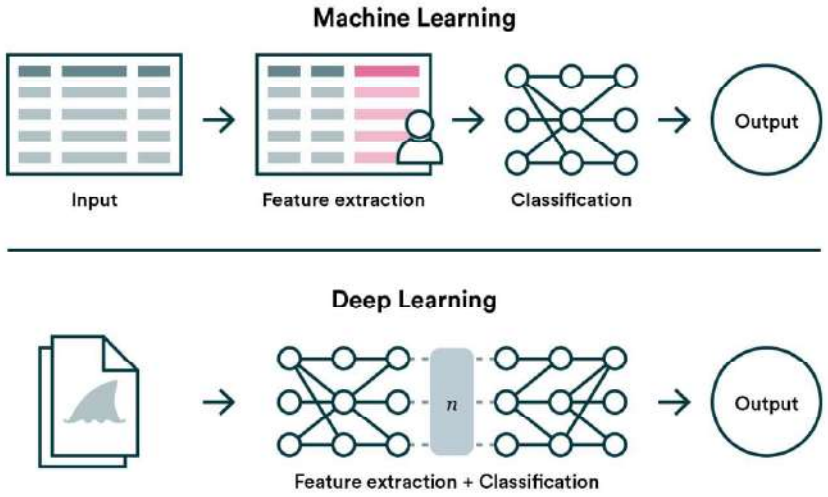


Fig.4 Classical ML vs Deep Learning

Data representations that are hierarchical can be learned by deep learning models. This implies that deep learning models may discover the relationships between various features and how to utilize them to predict outcomes. One of the reasons deep learning models have been so successful in difficult tasks like image classification and natural language processing is their capacity to build hierarchical representations of input.

3.3.1 Long Short-Term Memory (LSTM) networks

Recurrent neural network (RNN) architectures with long short-term memory (LSTM) networks are particularly well-suited for time series prediction because they are made to handle sequential data. Long-term dependencies in time series data are efficiently captured by LSTMs, which solves the vanishing gradient issue that traditional RNNs have. This allows long or irregular time intervals between events to be accommodated in the learning of patterns and prediction by LSTM models based on historical data.

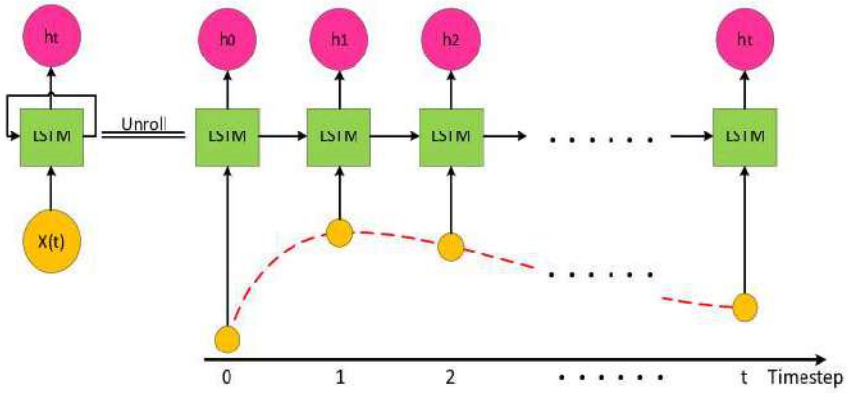


Fig 5. LSTM architecture for time series data

The three essential parts that let LSTM networks efficiently extract long-term dependencies from time series data are the forget gate, input gate, and output gate. The vanishing gradient issue that besets conventional recurrent neural networks is solved by the cooperative action of these gates, which enable selective memory and processing of information over time. To make sure that only pertinent information is carried over, the forget gate decides which data from the previous cell state should be kept. By incorporating important updates from the current input, the input gate determines what new data should be added to the cell state.

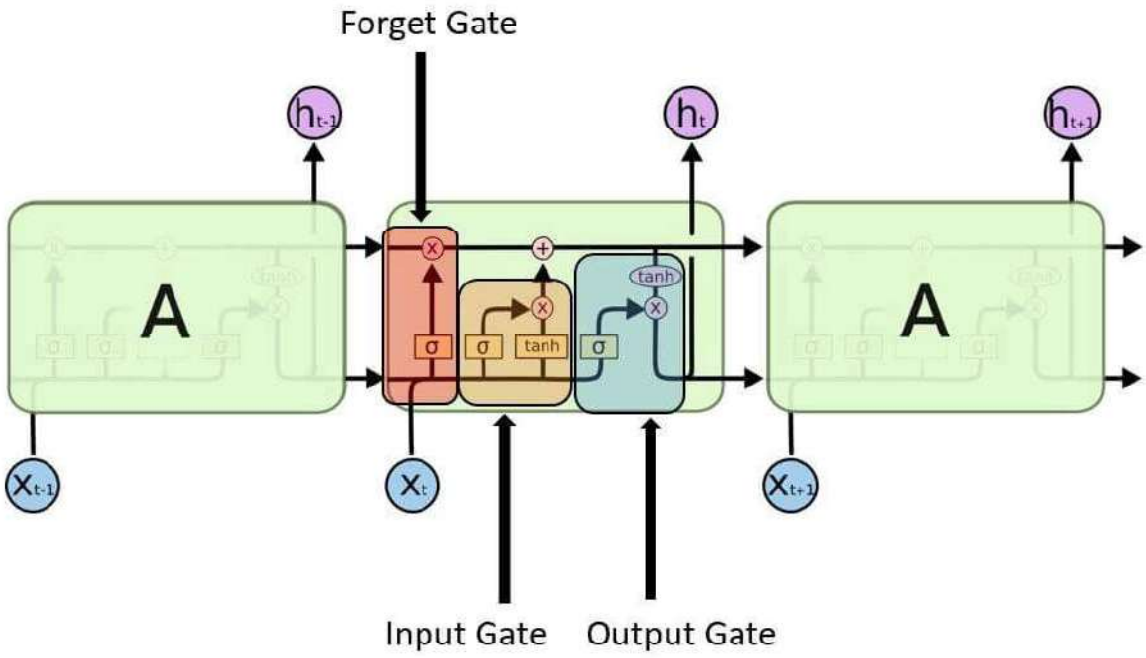


Fig 6. Gates in a LSTM unit cell

The output gate regulates which information from the updated cell state is exposed to the next layer of the network, ensuring that only relevant information is propagated forward.This intricate interplay of gates enables LSTM networks to maintain a continuous flow of information through the network, effectively capturing long-term dependencies and making them a powerful tool for time series prediction. Despite their effectiveness in capturing long-term dependencies in time series data, LSTM models also have some drawbacks that limit their applicability. Their complex architecture can make them computationally expensive to train, especially for large datasets. Additionally, the intricate interplay of gates and cell states can contribute to the difficulty in interpreting LSTM models, hindering the understanding of how they arrive at their predictions.

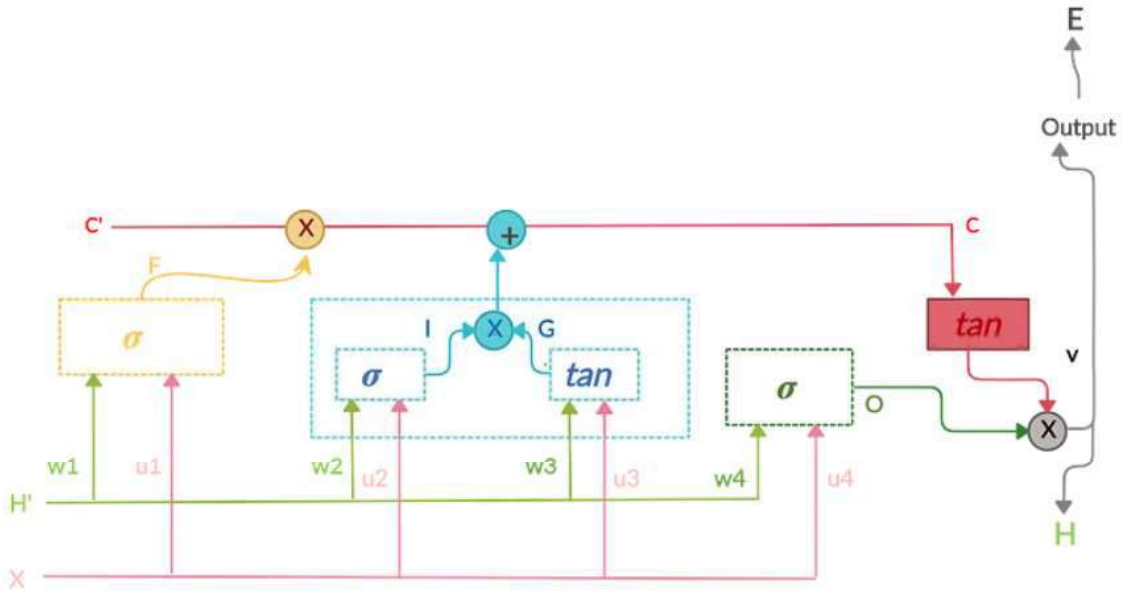


Fig.7 Internal structure of a unit LSTM cell

$$F = \sigma\left(u_1 X + \omega_1 H'\right)$$

$$I = \sigma\left(u_2 X + \omega_2 H'\right)$$

$$G = \tan\left(u_3 X + \omega_3 H'\right)$$

$$C = C'F + G$$

$$O = \sigma\left(u_4 X + \omega_4 H'\right)$$

$$Output = v.O.\tan\left(C\right)$$

Furthermore, LSTM models are prone to overfitting, where they learn the training data too well and fail to generalize to unseen data, requiring careful regularization techniques to prevent this issue.Finally, LSTM models are sensitive to hyperparameter tuning, making them challenging to optimize effectively.

3.3.2 N-Beats

N-BEATS (Neural Basis Expansion Analysis Time Series) is a versatile neural network architecture tailored for time series prediction, renowned for its flexibility, interpretability, and scalability. Comprising stacks, each containing trend and seasonality subnetworks, N-BEATS accommodates various time series patterns without presuming any specific structure. Its interpretability is a distinctive feature, enabling users to discern the contributions of individual components. Training involves the representation of time series data as input windows, and the model is optimized using gradient-based techniques.

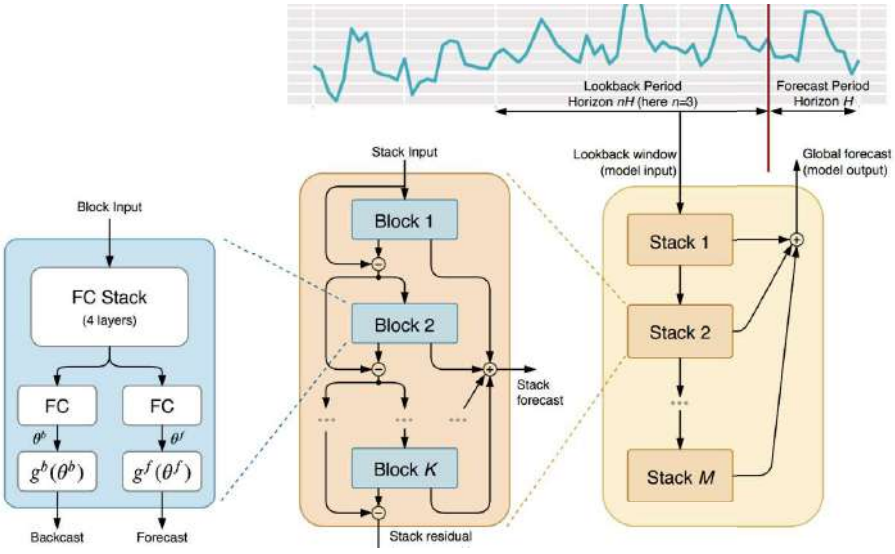


Fig.8 N-Beats Model for time series prediction

N-BEATS excels in both univariate and multivariate time series prediction, exhibiting competitive performance and scalability, making it an influential choice in competitions and practical applications. Its modular design and interpretative framework make it well-suited for diverse prediction tasks, contributing to its widespread adoption in the time series prediction domain. N-BEATS, or Neural Basis Expansion Analysis for Time Series, is a deep learning architecture specifically designed for time series prediction. It addresses the limitations of traditional deep learning models by employing a novel basis expansion approach, enabling it to capture both

long-term and short-term patterns in time series data.

The N-BEATS architecture consists of three primary components: stacks, blocks, and expansion layers. Stacks are arranged in a hierarchical manner, with each stack responsible for capturing different temporal frequencies in the time series data. Blocks, the fundamental processing units of the network, consist of fully connected layers followed by residual connections, allowing for efficient information flow and preventing vanishing gradients. Expansion layers transform the input sequence into a higher-dimensional representation, enabling the model to capture more complex patterns in the data.

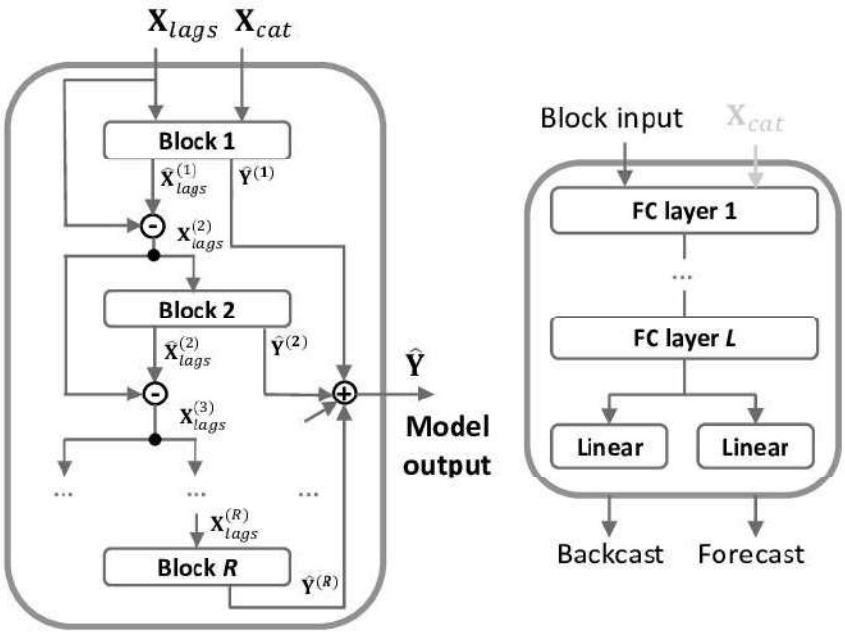


Fig.9 Architecture of N-Beats Model

The working mechanism of N-BEATS involves sequentially processing the input time series data through the stacked blocks. Each stack extracts relevant information and refines the forecast, with the outputs of different blocks combined through residual connections. Basis expansion layers enhance the model's ability to capture complex nonlinear relationships in the data. The final output of N-BEATS is a forecast of the future values of the time series, generated by combining the outputs of all stacks.

3.3.3 Transformer

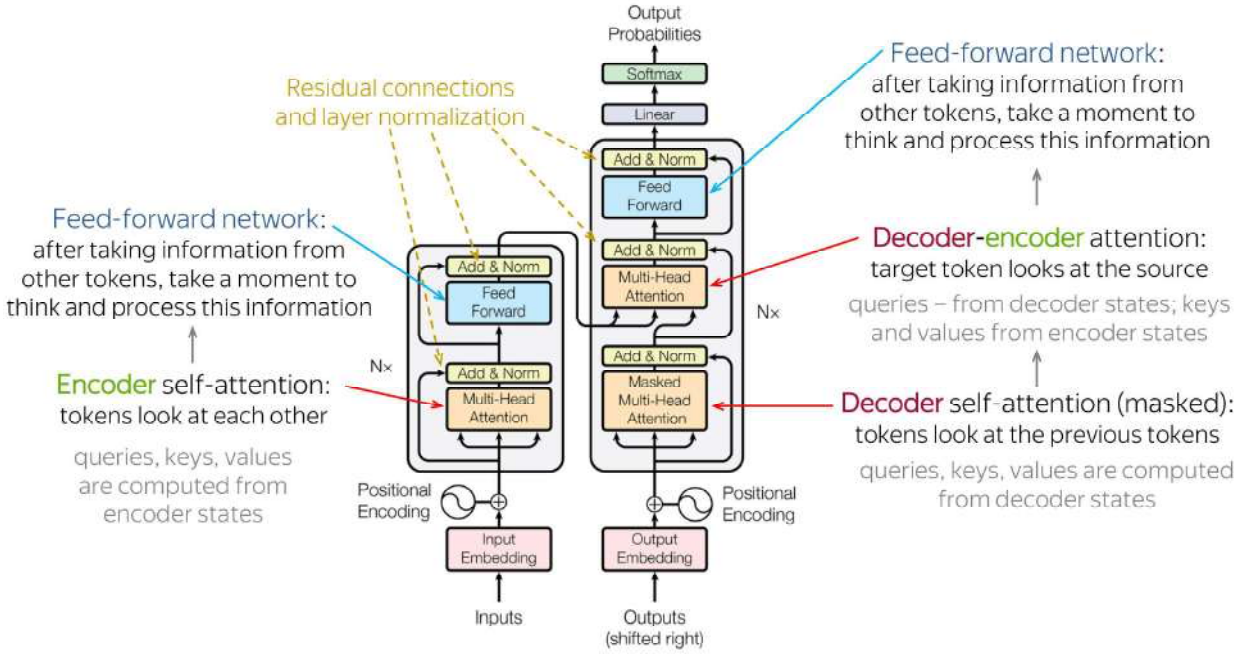


Fig.10 Transformer Architecture

With ease managing long-range dependencies, the transformer neural network is a novel architecture designed to solve sequence-to-sequence tasks. It was initially suggested in the paper "Attention Is All You Need"[14] and is currently considered a cutting-edge method in NLP.It learns long-range dependencies in sequential data by utilizing a self-attention mechanism. The network can learn the relationships between various sequence segments, even if they are separated by a large number of time steps, thanks to the self-attention mechanism. An encoder and a decoder are the two primary parts of the transformer model. An input sequence of tokens is mapped into a sequence of continuous representations by the encoder. After receiving the encoder output, the decoder creates an output token sequence.

Encoder

The encoder is composed of a stack of 6 encoder layers, each encoder layer is composed of 2 sublayers: multi-head self attention mechanism and feed-forward network: Multi-head self attention means that each position of the input sequence is attended to by all other positions of the sequence. This means that the model is able to observe long-range dependencies within the input sequence. Feed-forward network means that a simple neural network is applied to the output of a multichannel ReLU activation function.

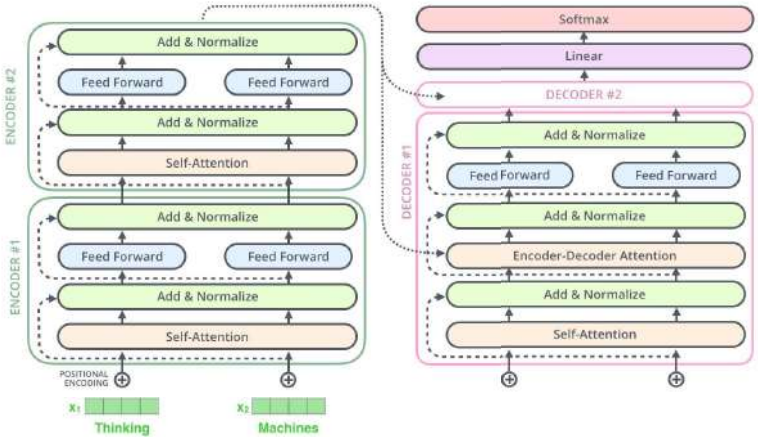


Fig.11 Encoder-Decoder block in Transformer

Decoder

Six decoder layers are stacked to form the decoder. Three sublayers make up each decoder layer: a feed-forward network, a multi-head encoder-decoder attention mechanism, and a masked self-attention mechanism. Every place in the output sequence can attend to every preceding position in the output sequence thanks to the masked self-attention technique. By doing this, the model is kept from paying attention to spots in the output sequence that come after, which would go against the task's autoregressive character. Every location in the output sequence can attend to every position in the input sequence thanks to the multi-head encoder-decoder attention system. This enables the decoder to create the output sequence while incorporating data from the input sequence.

Positional Encoding

The transformer model does not use recurrent connections, which can suffer from the vanishing gradient problem. Instead, the transformer model uses positional encoding to add information about the relative positions of the tokens in the input and output sequences.

$$\mathrm{p}_{k,2i} = \sin\left(\frac{k}{10000^{2i/d}}\right) \qquad \mathrm{p}_{k,2i+1} = \cos\left(\frac{k}{10000^{2i/d}}\right)$$

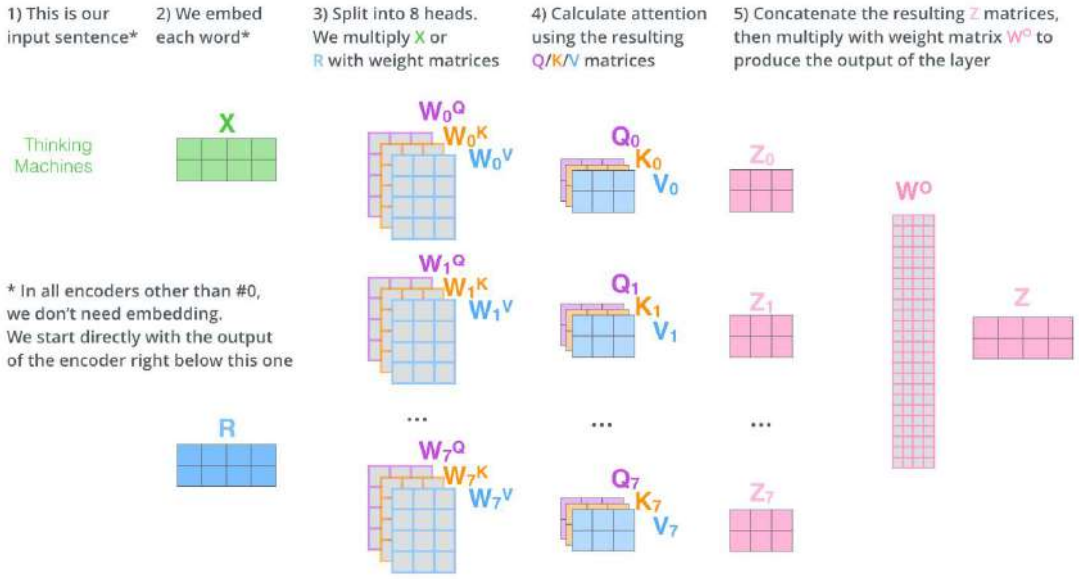


Fig.12 Calculation of Multihead Attention in Transformer

$$Q^{(h)}(\mathbf{x}_i) = W_{h,q}^T \mathbf{x}_i, \quad K^{(h)}(\mathbf{x}_i) = W_{h,k}^T \mathbf{x}_i, V^{(h)}(\mathbf{x}_i) = W_{h,v}^T \mathbf{x}_i, \quad W_{h,q}, W_{h,k}, W_{h,v} \in \mathbb{R}^{d\times k},$$

$$\alpha_{i,j}^{(h)} = softmax_j\left(\frac{\left\langle Q^{(h)}(\mathbf{x}_i), K^{(h)}(\mathbf{x}_j)\right\rangle}{\sqrt{k}}\right),$$

$$\mathbf{u}_i' = \sum_{h=1}^{H} W_{c,h}^T \sum_{j=1}^{n} \alpha_{i,j}^{(h)} V^{(h)}(\mathbf{x}_j), \qquad W_{c,h} \in \mathbb{R}^{k\times d},$$

$$\mathbf{u}_i = LayerNorm(\mathbf{x}_i + \mathbf{u}_i'; \gamma_1, \beta_1), \qquad \gamma_1, \beta_1 \in \mathbb{R}^d,$$

$$\mathbf{z}_i' = W_2^T ReLU\left(W_1^T \mathbf{u}_i\right), \qquad W_1 \in \mathbb{R}^{d\times m}, W_2 \in \mathbb{R}^{m\times d},$$

$$\mathbf{z}_i = LayerNorm(\mathbf{u}_i + \mathbf{z}_i'; \gamma_2, \beta_2), \qquad \gamma_2, \beta_2 \in \mathbb{R}^d.$$

Multihead Attention

Multi-head attention is a core component of the transformer architecture, enabling it to effectively capture long-range dependencies and relationships within sequences. It involves projecting input queries, keys, and values into multiple parallel heads, each learning a distinct representation of the input.

The calculation of multi-head attention begins with linear projections of the input queries, keys, and values into separate heads. This creates a unique set of query, key, and value vectors for each head. Next, for each head, the scaled dot-product of the query and key vectors is computed, resulting in attention scores that measure the similarity between each query-key pair. These scores are then normalized using a softmax function to obtain attention weights, representing the relative importance of each key for a given query.

Utilizing the attention weights, a weighted sum of the value vectors is calculated, generating a context vector for each head. These context vectors from all heads are concatenated and subsequently projected linearly to obtain the final multi-head attention output. This refined output captures a more comprehensive representation of the input sequence, allowing the model to better understand and process complex relationships within the data.

Transformers have emerged as powerful tools for time series prediction, effectively capturing long-term dependencies in time series data. Unlike traditional recurrent neural networks (RNNs) that struggle with long-term dependencies, transformers employ the self-attention mechanism, enabling them to consider relationships between distant data points. By treating each time step as a token, transformers can process the entire time series simultaneously, overcoming the vanishing gradient problem that plagues RNNs. This enables transformers to identify and learn from long-range patterns in time series data, leading to more accurate forecasts.

3.4 Evaluation Metrics

- *Mean Absolute Error(MAE)*

Time series prediction frequently uses mean absolute error (MAE) as an error metric. The difference between the expected and actual values is measured on an average absolute basis. Error metrics that are easy to compute and understand include mean average error (MAE). Its insensitivity to outliers makes it a good option for assessing the effectiveness of time series prediction models.

$$MAE = \frac{1}{n}\sum \mathbf{abs}(y - \hat{y})$$

- *Mean Squared Error(MSE)*

With a slight variation in mean absolute error, mean squared error (MSE) is a widely used and straightforward metric. The calculation of the mean squared error involves calculating the squared difference between the actual and predicted values.It shows the squared difference between the values that are actual and those that are predicted. To the advantage of MSE, we perform squared to prevent the cancellation of negative terms.

$$MSE = \frac{1}{n}\sum (y - \hat{y})^2$$

- *Root Mean Squared Error(RMSE)*

RMSE is the square root of MSE.It is also a good measure of the overall error of the model. It is less sensitive to outliers than MSE, but it is more difficult to interpret

$$RMSE = \sqrt{\frac{1}{n}\sum (y - \hat{y})^2}$$

$$Where:$$

$(\bar{y})$: "$mean\ of\ the\ values\ of\ the\ outputs$"

$(\hat{y})$: "$predicted\ value\ of\ the\ output$"

(n) : "$total\ number\ of\ data\ points$"

Chapter 4

Methodology

Data collection, preprocessing, feature engineering, model selection, training, and evaluation are the usual steps in the machine learning workflow.The process of collecting data entails obtaining information from many sources, including sensors, databases, and the internet. Cleaning and preparing the data for analysis, including addressing missing values, outliers, and inconsistencies, is known as data preprocessing. Retrieving pertinent features from the data is the process of feature engineering.

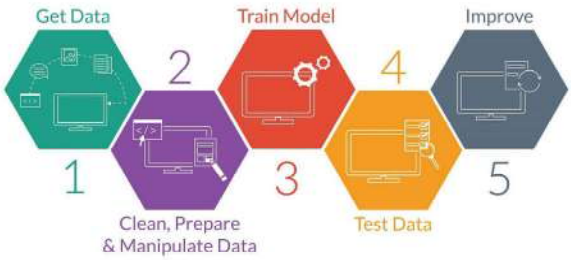


Fig.13 Machine learning workflow

Selecting a suitable machine learning algorithm for a given problem is known as model selection. The kind of data, the intended result, and the available computational power all influence the algorithm selection. In order to minimize error on the training data, model training entails training the machine learning algorithm.To make sure the machine learning algorithm generalizes well to new data, model evaluation entails assessing the algorithm's performance on testing data, a set of data that was not used for training.

4.1 Dataset Description

The ACN-Data dataset is a publicly available repository of over 30,000 EV charging sessions meticulously compiled from two workplace charging sites in California. This invaluable resource is continuously enriched with newly recorded sessions, ensuring its relevance and timeliness. The dataset's custodian, PowerFlex, maintains an extensive network of over 60 charging sites across the country, and the ACN-Data team envisions expanding the dataset to encompass these additional locations, broadening its scope and applicability.ACN-Data was collected from two Adaptive Charging Networks located in California. This work will make use of the ACN [17] dataset, one of the few publicly available datasets for non-residential EV charging. The dataset includes charging logs from JPL and Caltech, two university campus stations. Employees are the only ones who can access the JPL station, in contrast to the public that can visit the Caltech station. The dataset can be accessed from [18] by either a web portal or python application programming interface(API). The ACN on the Caltech campus is in a parking garage and has 54 EVSEs (Electric Vehicle Supply Equipment or charging stations) along with a 50 kW dc fast charger. The Caltech ACN is open to the public and is often used by non-Caltech drivers. JPL's ACN includes 52 EVSEs in a parking garage.

Field	Description
connectionTime	Time when the user plugs in.
doneChargingTime	Time of the last non-zero charging rate.
disconnectTime	Time when the user unplugs.
kWhDelivered	Measured Energy Delivered
siteID	Identifier of the site where the session took place.
stationID	Unique identifier of the EVSE.
sessionID	Unique identifier for the session.
timezone	Timezone for the site.
pilotSignal	Time series of pilot signals during the session.
chargingCurrent	Time series of actual charging current of the EV.
userID*	Unique identifier of the user.
requestedDeparture*	Estimated time of departure.
kWhRequested*	Estimated energy demand.

Fig.14 Fields present in the ACN dataset

4.2 Data Preprocessing

Cleaning and preprocessing the dataset is vital to ensuring the quality of the predictive models. These include removing faulty records and outliers. The presence of outliers can negatively impact the model performance.Upon initial examination of the dataset, it was discovered that there were outliers and missing values for days. To address these issues, the k-nearest neighbors (KNN) imputation method was employed to fill in the missing values. This technique involves identifying the k nearest neighbors for each data point with missing values and using the average of those neighbors to impute the missing value. Additionally, graphs and box plots were created to visualize the distribution of the data before and after imputation, allowing for a thorough assessment of the effectiveness of the imputation method.Furthermore, the data was processed to generate daily data starting from April 2018 up to September 2021 for the Caltech site and May 2018 to September 2021 for the JPL site. This aggregation of data allowed for a more comprehensive understanding of the daily charging patterns and trends at both sites.

	sessionId	kwhTotal	dollars	created	ended	startTime	endTime	chargeTimeHrs	weekday	platform	...	managerVehicle	facilityType
0	1366563	7.78	0.00	0014-11-18 15:40:26	0014-11-18 17:11:04	15	17	1.510556	Tue	android	...	0	3
1	3075723	9.74	0.00	0014-11-19 17:40:26	0014-11-19 19:51:04	17	19	2.177222	Wed	android	...	0	3
2	4228788	6.76	0.58	0014-11-21 12:05:46	0014-11-21 16:46:04	12	16	4.671667	Fri	android	...	0	3
3	3173284	6.17	0.00	0014-12-03 19:16:12	0014-12-03 21:02:18	19	21	1.768333	Wed	android	...	0	3
4	3266500	0.93	0.00	0014-12-11 20:56:11	0014-12-11 21:14:06	20	21	0.298611	Thu	android	...	0	3

5 rows × 24 columns

Fig.15 Dataset head

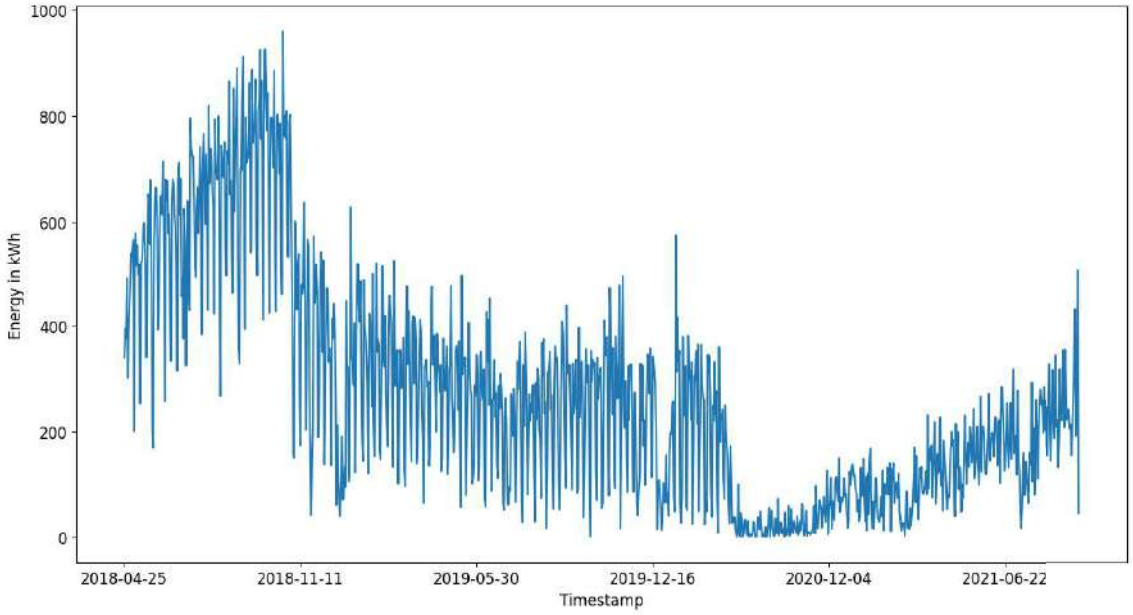


Fig.16 Distribution of Caltech data over time

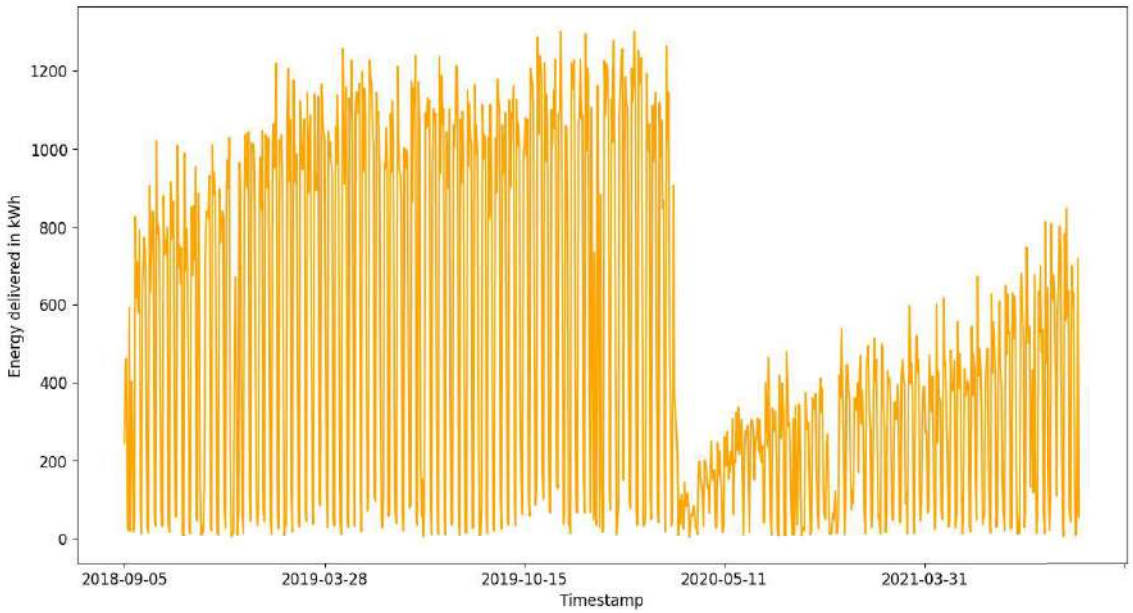


Fig.17 Distribution of JPL data over time

To enhance the predictive performance of the models, the data was strategically divided into intervals of days, months, and weeks. This approach facilitated the capture of temporal patterns and trends at varying granularities, ensuring that the models could effectively learn from the data's underlying dynamics. Furthermore, scaling was applied to the data, normalizing its distribution and improving the convergence of the machine learning algorithms. To assess the models' generalization capabilities, the data was partitioned into training, validation, and testing sets, adhering to a 0.75:0.10:0.15 ratio. This stratification ensured that the models were trained on a representative sample of the data, while a separate validation set was used for hyperparameter tuning and a final testing set was employed for evaluating the models' performance on unseen data. To harness the power of time series prediction models, the dataset was transformed into a series of time-stamped sequences, capturing the temporal dependencies inherent in the charging behavior. Each sequence represented the charging events for a particular day, month, or week, allowing the models to learn from patterns and trends over time. The resulting sequences were then fed into the machine learning models, enabling them to capture the dynamic nature of EV charging demand and make informed predictions.

4.3 Model Training and Evaluation

4.3.1 SVM

- *Model Structure*

$$Model : Support\ Vector\ Regression\ (SVR)$$
$$Kernel : Radial\ Basis\ Function\ (RBF)$$
$$\epsilon = 0.3 \quad \gamma = 0.6 \quad C = 20$$

SVR is a type of support vector machine (SVM) that is used for regression tasks. It works by finding a hyperplane that maximizes the margin between the data points and the hyperplane. The hyperplane is chosen to minimize the generalization error of the model.

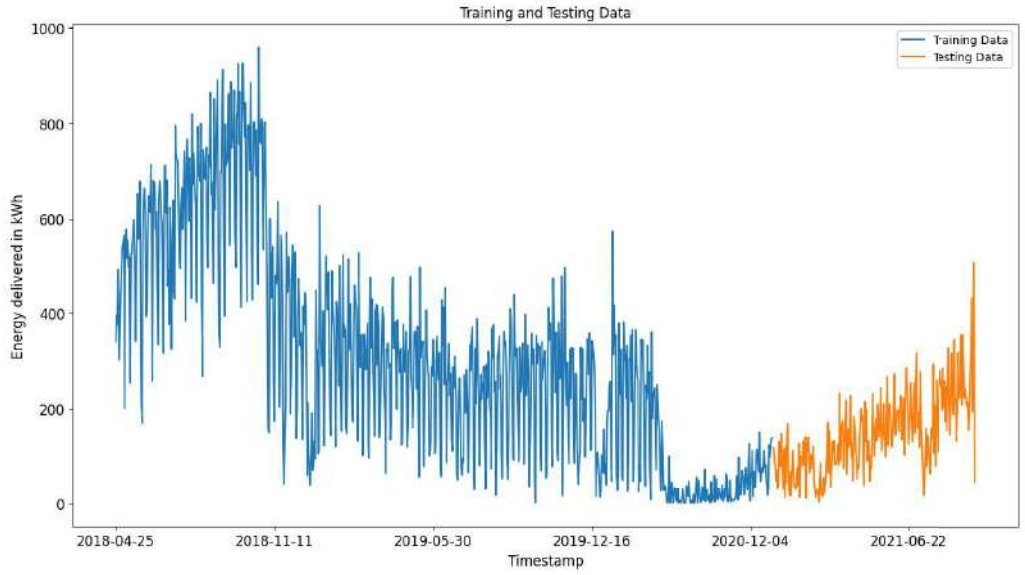


Fig.18 Splitting of Caltech data

The RBF kernel is a popular kernel for SVR. It is a non-linear kernel that can capture complex relationships between the data points. The gamma parameter controls the width of the kernel function. A larger gamma value corresponds to a narrower

kernel function, which means that only data points that are close to each other will have a strong influence on the hyperplane.

The C parameter controls the trade-off between maximizing the margin and minimizing the number of training errors. A larger C value corresponds to a larger margin, which can help to improve the generalization performance of the model. However, a larger C value can also lead to overfitting, which means that the model will not generalize well to unseen data.

The epsilon parameter controls the tolerance for errors. A larger epsilon value corresponds to a greater tolerance for errors, which can make the model more robust to outliers. However, a larger epsilon value can also lead to a less accurate model.

The specified SVR model with the parameters is a good choice for modeling EV charging demand. The RBF kernel is a good choice for capturing complex relationships between the data points, and the gamma value of 0.6 is a good starting point. The C value of 20 is a moderate value that should help to balance the trade-off between maximizing the margin and minimizing the number of training errors. The epsilon value of 0.03 is a small value that will make the model less tolerant of errors, but this may be necessary to ensure that the model is accurate.

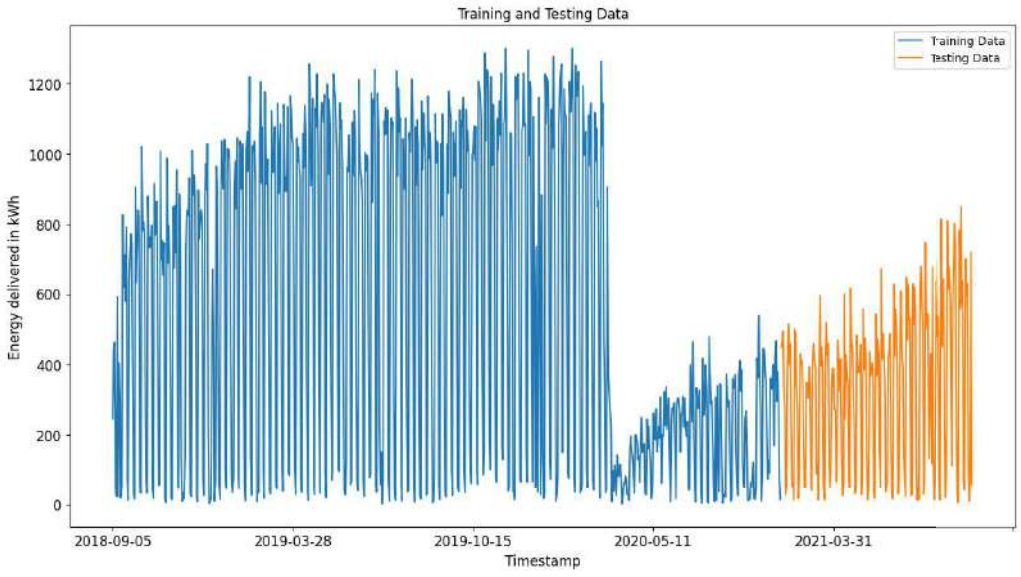
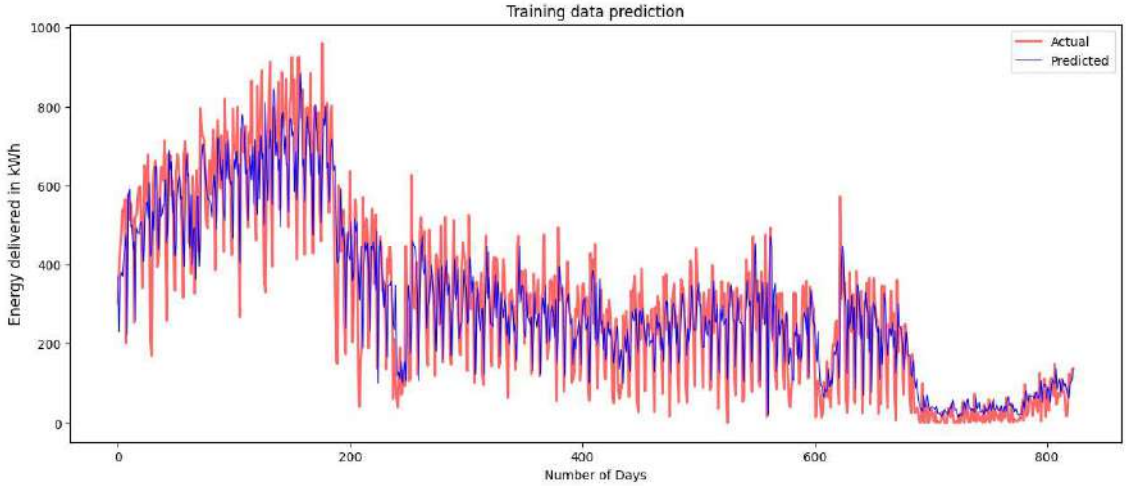


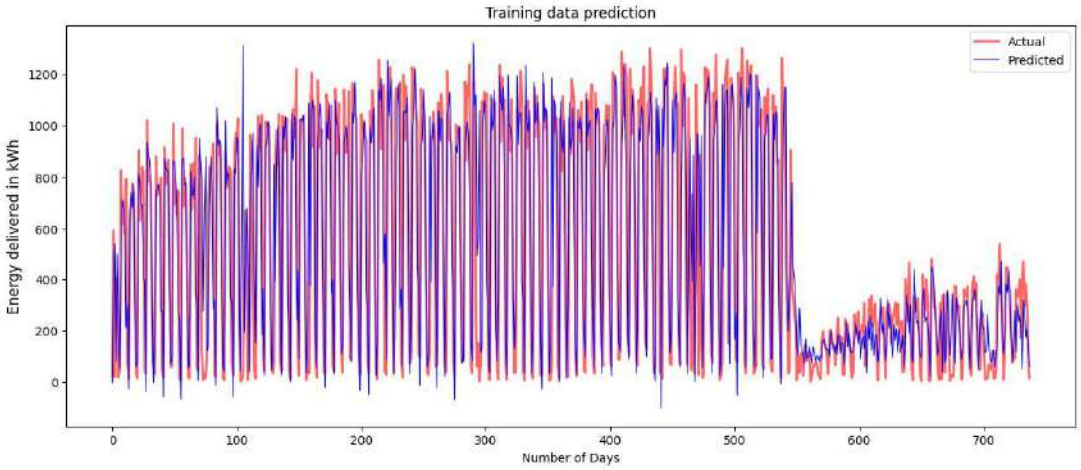
Fig.19 Splitting of JPL data

- *Results*



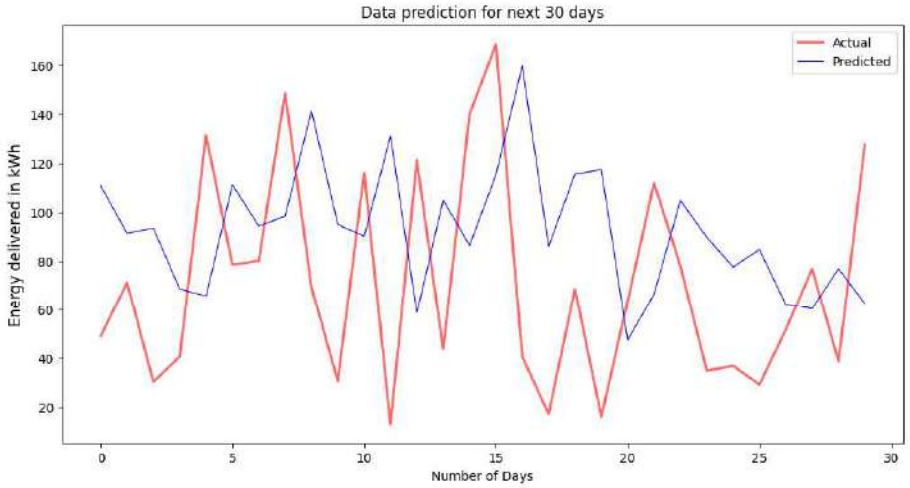
MAE : 82.1602kWh $RMSE$: 110.033kWh

Fig.20 SVR model prediction for Caltech training data



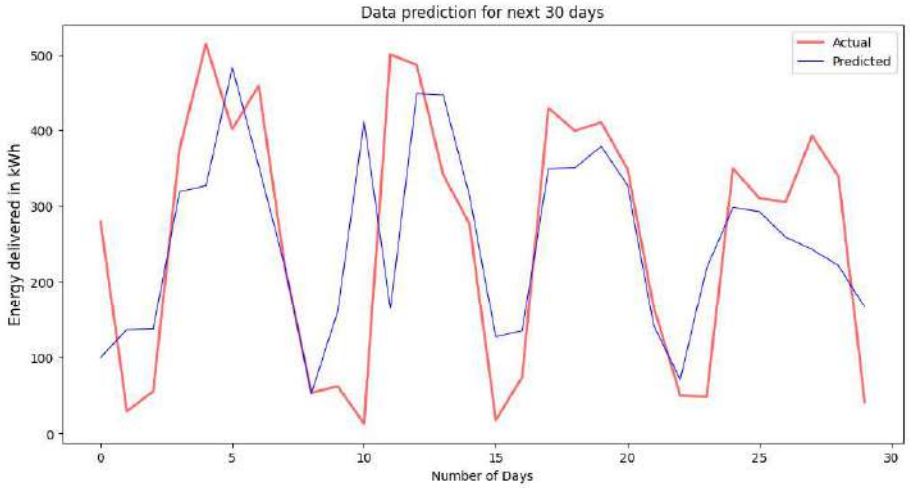
MAE : 127.6105kWh $RMSE$: 154.2079kWh

Fig.21 SVR model prediction for JPL training data



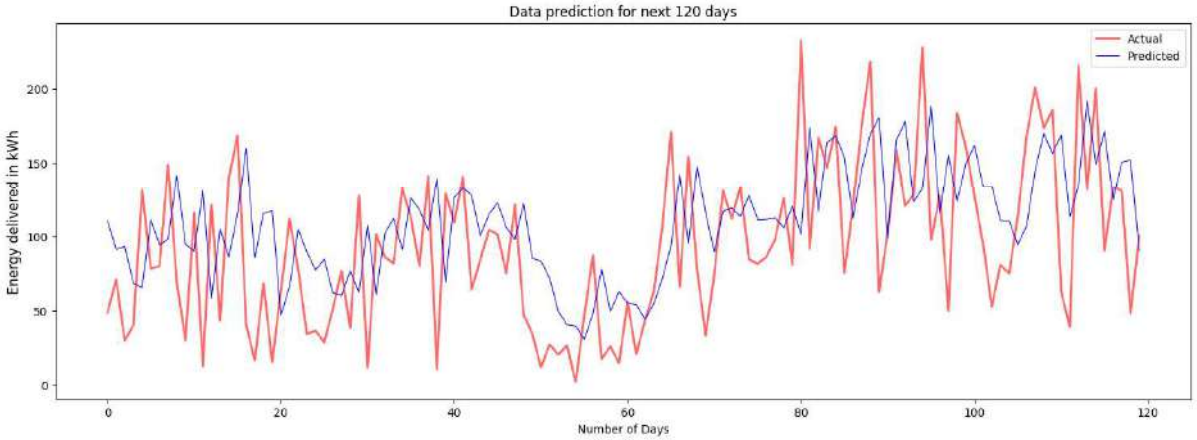
$MAE : 51.7682kWh \qquad RMSE : 58.5068kWh$

Fig.22 SVR model prediction for next 30 days on Caltech data



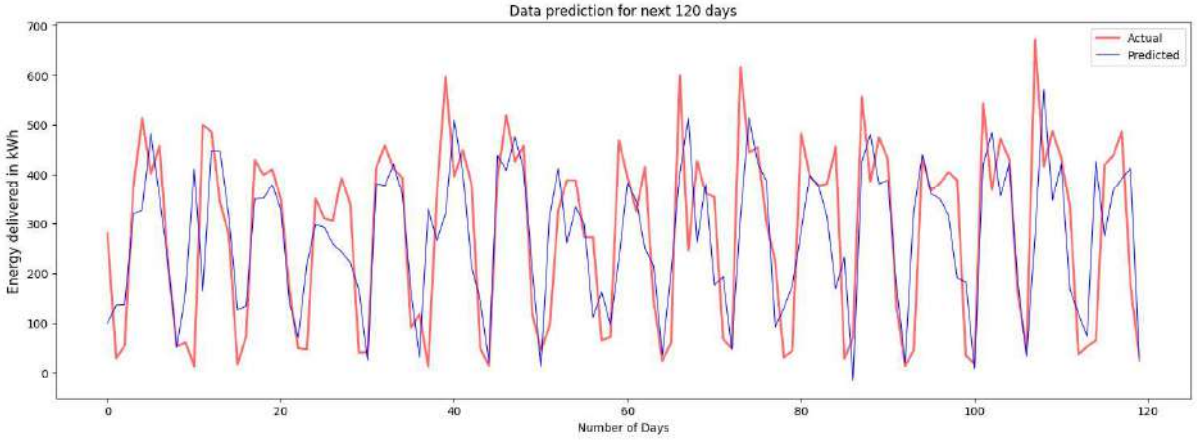
$MAE : 96.3300kWh \qquad RMSE : 131.0005kWh$

Fig.23 SVR model prediction for next 30 days on JPL data



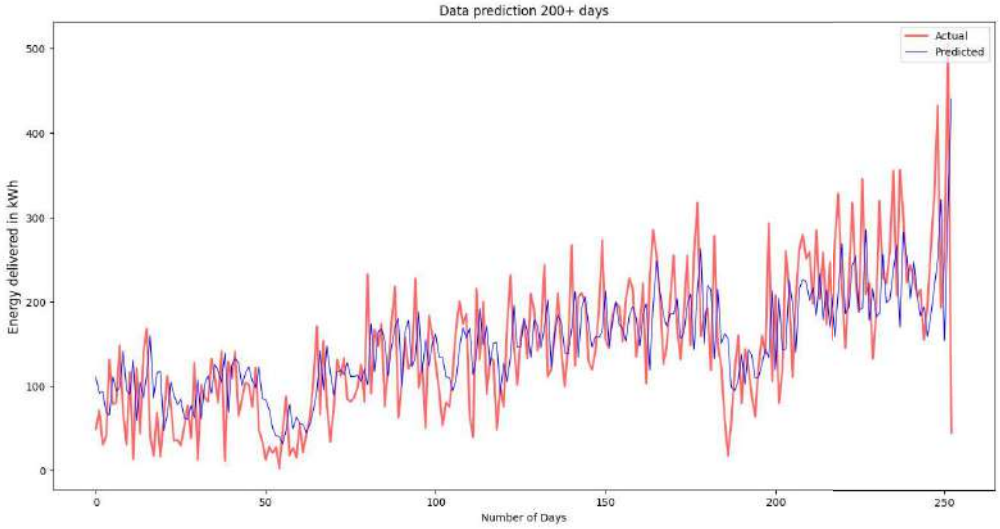
$MAE : 54.7319kWh \qquad RMSE : 58.5068kWh$

Fig.24 SVR model prediction for next 120 days on Caltech data



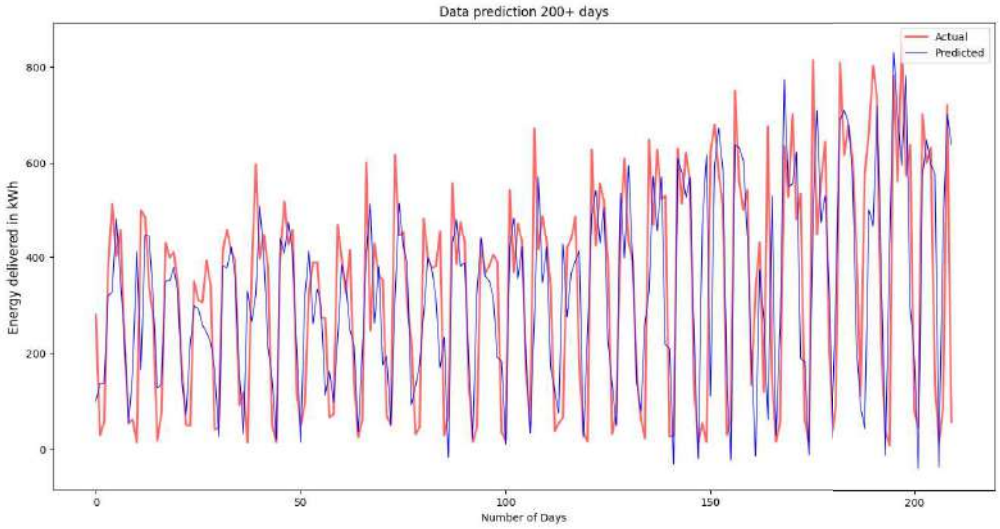
$MAE : 102.5280kWh \qquad RMSE : 136.0776kWh$

Fig.25 SVR model prediction for next 120 days on JPL data



$MAE : 97.6583kWh \qquad RMSE : 121.8956kWh$

Fig.26 SVR model prediction for next 240 days on Caltech data

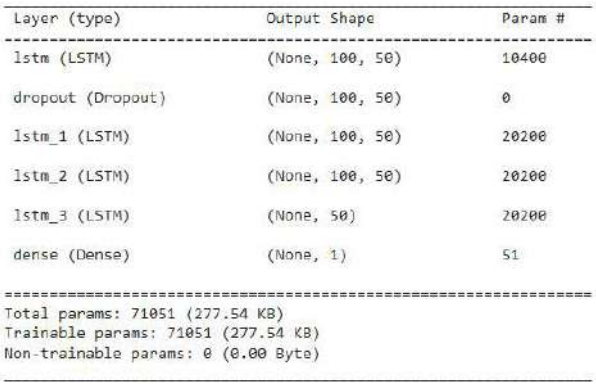


$MAE : 127.0562kWh \qquad RMSE : 149.4827kWh$

Fig.27 SVR model prediction for next 240 days on JPL data

4.3.2 LSTM

- *Model Structure*



Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100, 50)	10400
dropout (Dropout)	(None, 100, 50)	0
lstm_1 (LSTM)	(None, 100, 50)	20200
lstm_2 (LSTM)	(None, 100, 50)	20200
lstm_3 (LSTM)	(None, 50)	20200
dense (Dense)	(None, 1)	51

Total params: 71051 (277.54 KB)
Trainable params: 71051 (277.54 KB)
Non-trainable params: 0 (0.00 Byte)

Fig.28 Model summary

The proposed model employs a combination of LSTM layers, a dense layer, a dropout layer, and the Adam optimizer to effectively predict EV charging demand. The LSTM layers capture long-term dependencies in the data, while the dense layer transforms the LSTM output into a single prediction. Dropout acts as a regularization technique to prevent overfitting, and the Adam optimizer efficiently updates the model's weights during training. The difference in values between the predicted and real is measured by the RMSE loss function. Effectively addressing the intricacies of EV charging demand prediction is this well-organized model.The validation dataset was used to test the results of the training procedure and the training set was used to train the LSTM model. The training dataset was processed 50 times (epoch) using the learning algorithm, and following each batch (of size 32), the model weights were modified.

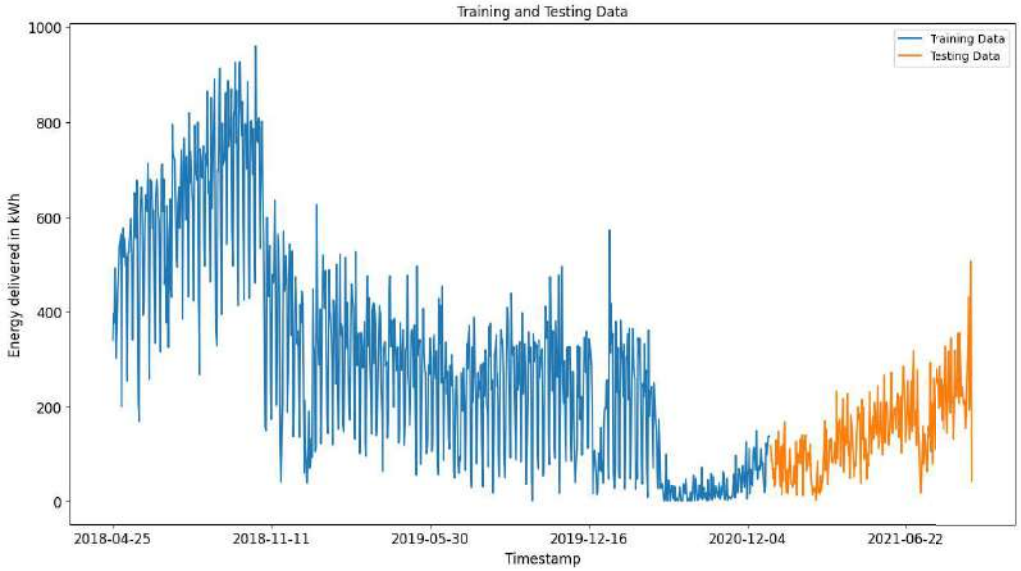


Fig.29 Splitting of Caltech data

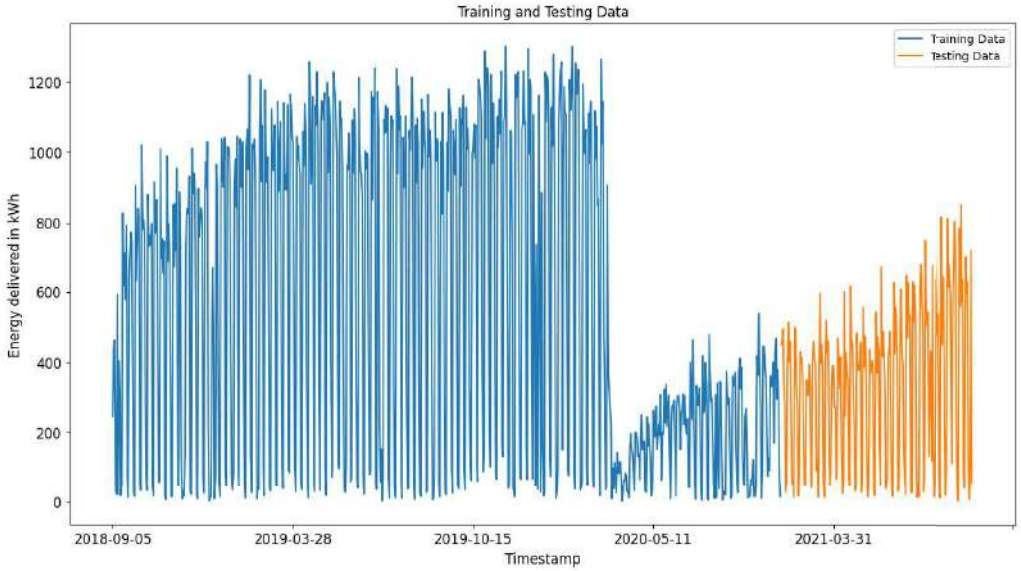
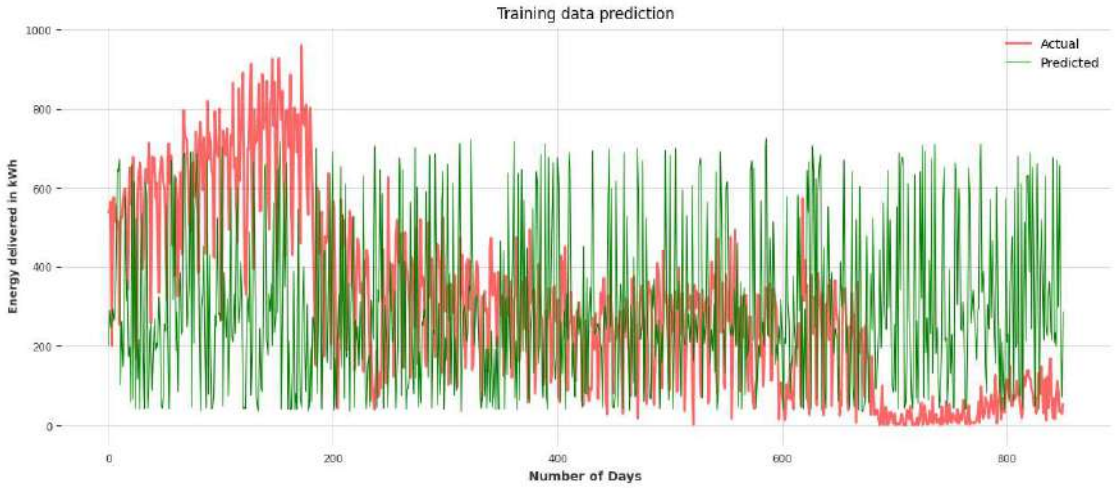


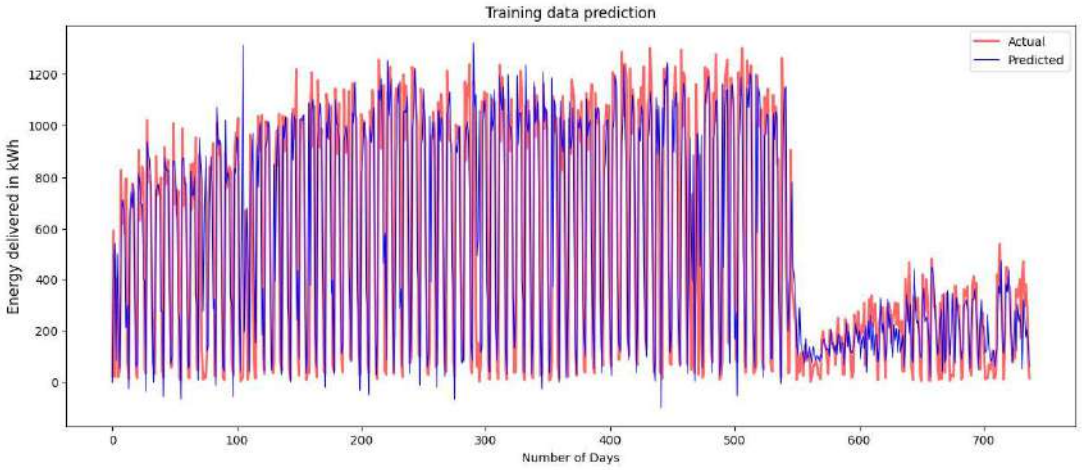
Fig.30 Splitting of JPL data

Results



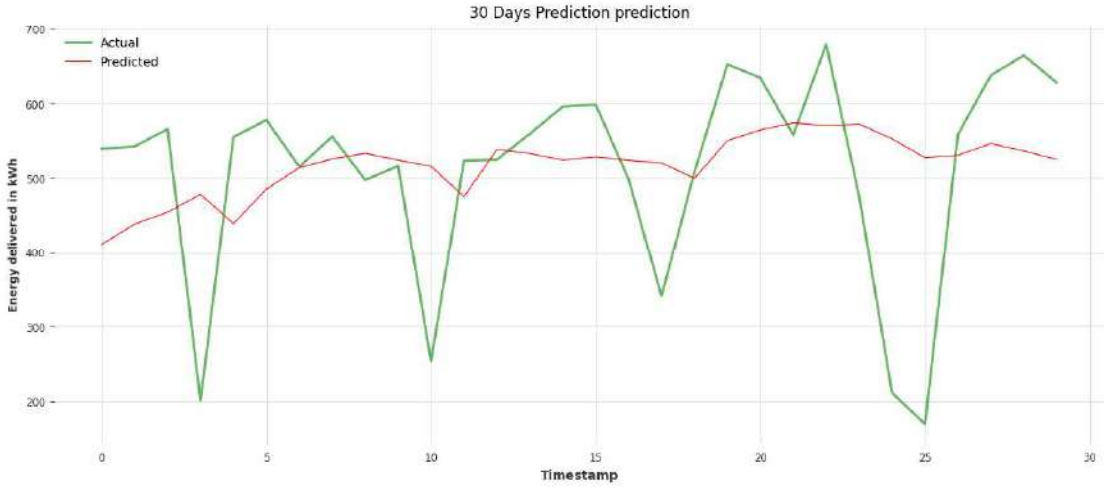
$MAE : 120.7793kWh \qquad RMSE : 130.7335kWh$

Fig.31 LSTM model prediction for Caltech training data



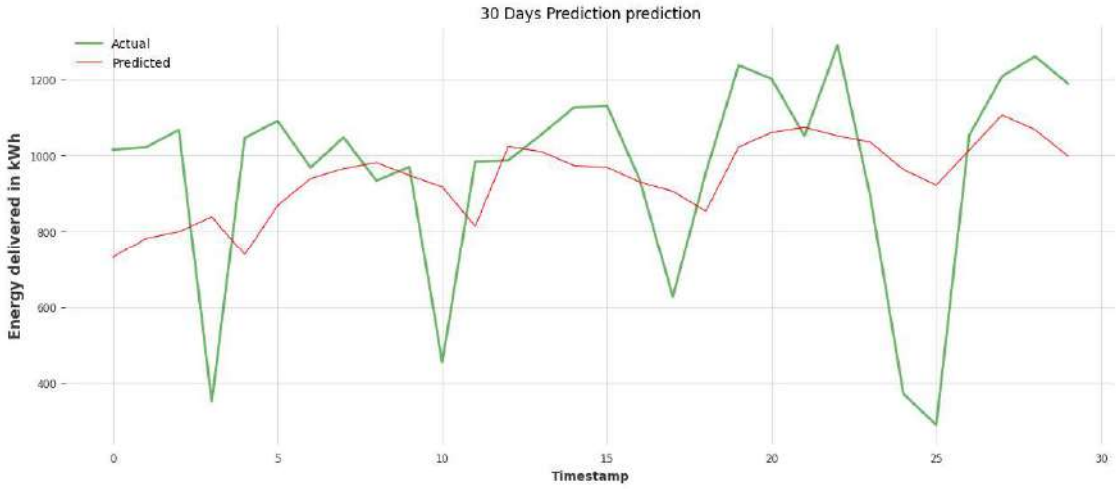
$MAE : 167.7335kWh \qquad RMSE : 154.7793kWh$

Fig.32 LSTM model prediction for JPL training data



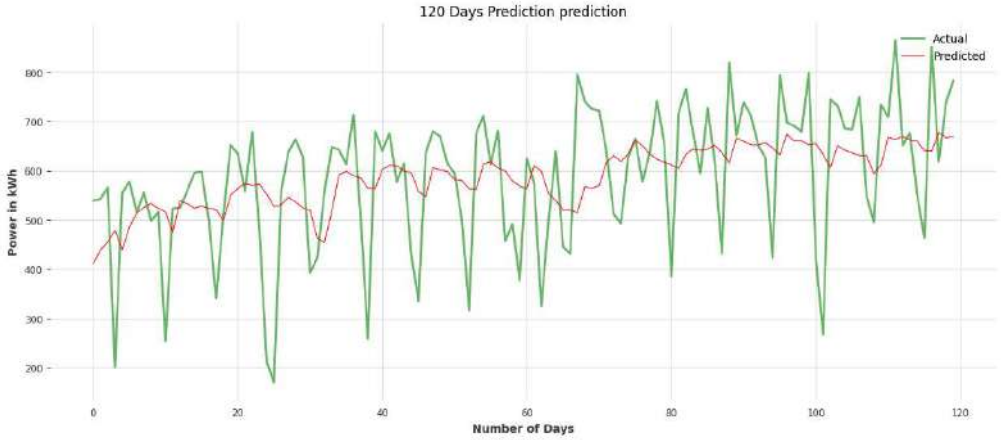
$MAE : 32.3416kWh \qquad RMSE : 38.0221kWh$

Fig.33 LSTM model prediction for next 30 days on Caltech data



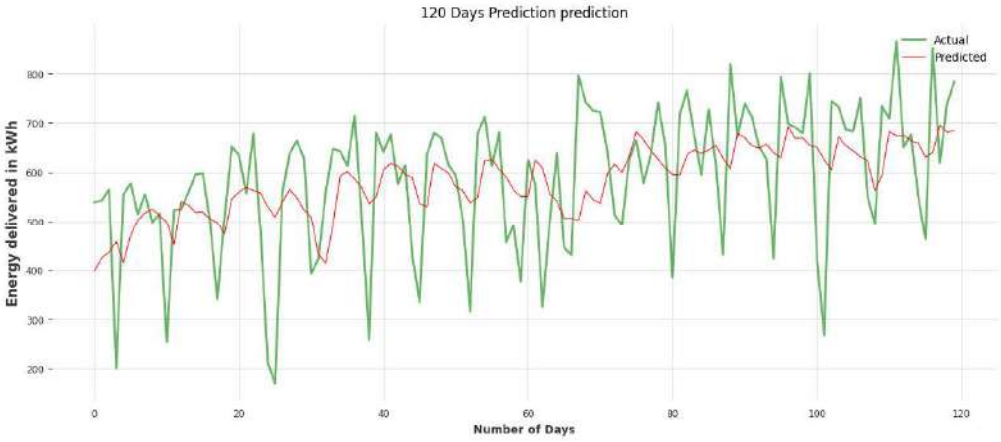
$MAE : 49.8765kWh \qquad RMSE : 52.0198kWh$

Fig.34 LSTM model prediction for next 30 days on JPL data



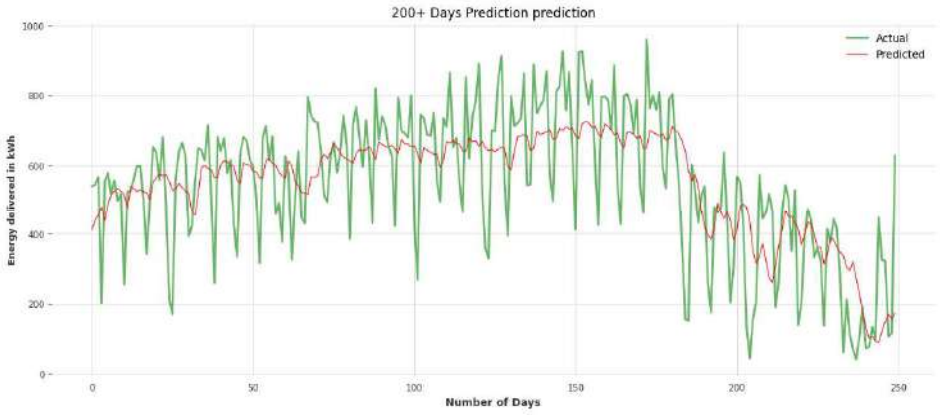
$MAE : 40.2456kWh \qquad RMSE : 50.7040kWh$

Fig.35 LSTM model prediction for next 120 days on Caltech data



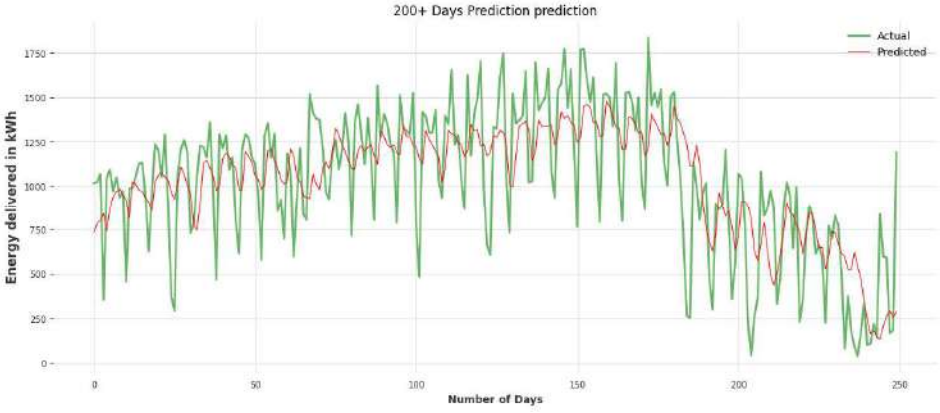
$MAE : 67.6543kWh \qquad RMSE : 70.5460kWh$

Fig.36 LSTM model prediction for next 120 days on JPL data



MAE : 118.9311kWh $RMSE$: 151.0258kWh

Fig.37 LSTM model prediction for next 240 days on Caltech data



MAE : 118.9311kWh $RMSE$: 151.0258kWh

Fig.38 LSTM model prediction for next 240 days on JPL data

4.3.3 N-Beats

- *Model Structure*

$Model : Neural\ Basis\ Expansion\ Analysis\ for\ Interpretable\ Time\ Series\ Prediction$
$GenericArchitecture : True$
$InputChunkLength : 128 \quad OutputChunkLength : 8 \quad Activation : ReLU$

The N-BEATS model uses a combination of fully connected layers and residual connections to learn the basis functions. ReLU activation introduces non-linearity into the model, allowing it to capture more complex relationships in the time series data. The Adam optimizer, known for its efficiency and effectiveness in deep learning, facilitates efficient weight updates during training. Dropout introduces randomness into the training process by randomly dropping a certain percentage of neurons from the network during training. This helps to prevent the network from learning too closely to the training data, which can improve its generalization performance.
A dropout rate of 0.20 is employed, meaning that 20% of the neurons are randomly dropped during training. This regularization technique helps to ensure that the model learns robust patterns from the training data, enabling it to generalize effectively to unseen data.
The generic architecture of the model is True, which means that the model uses the generic N-BEATS architecture. The generic architecture consists of a stack of blocks, each of which consists of a fully connected layer, a residual connection, and a normalization layer.
The input chunk length of the model is 128, which means that the model takes 128 time steps of data as input. The output chunk length of the model is 8, which means that the model predicts 10 time steps of data at a time. The model was trained for 100 epochs.

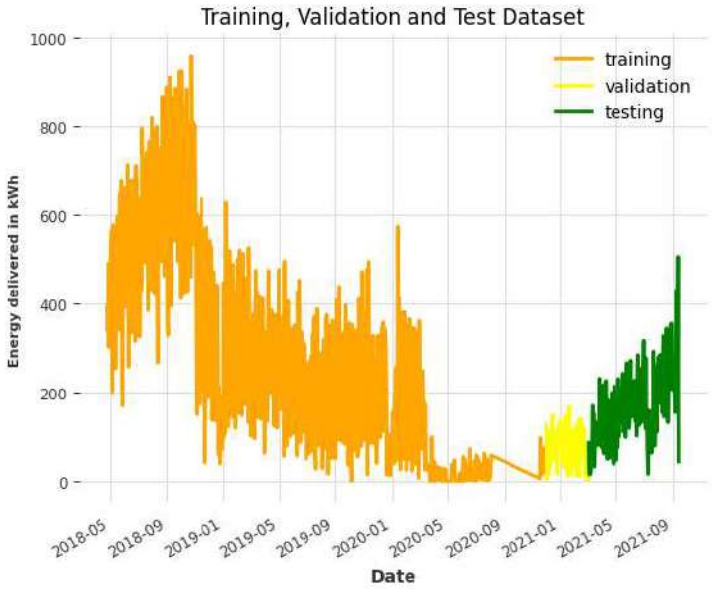


Fig.39 Splitting of Caltech data

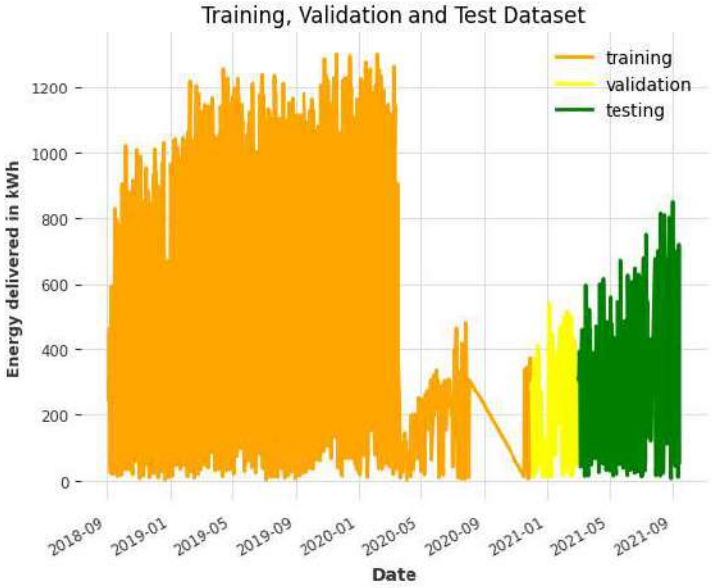
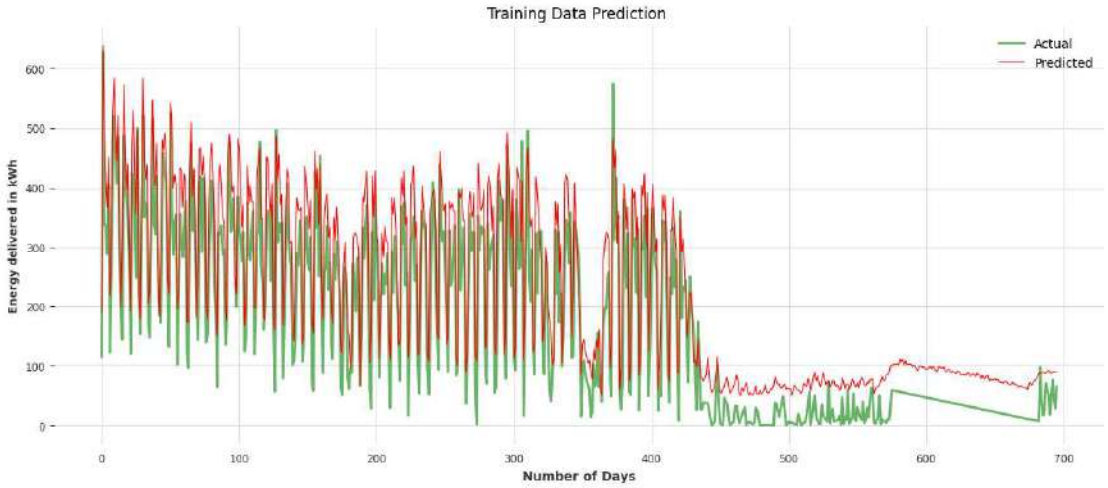


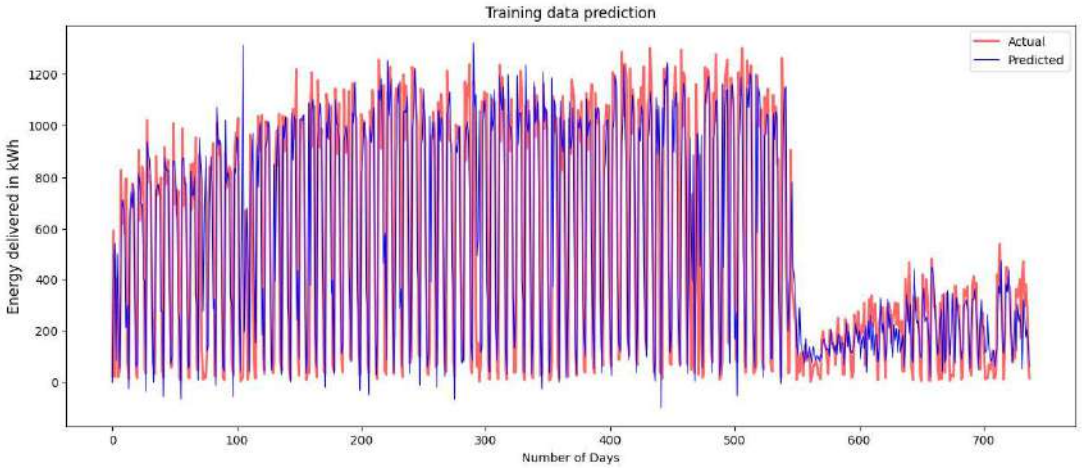
Fig.40 Splitting of JPL data

Results



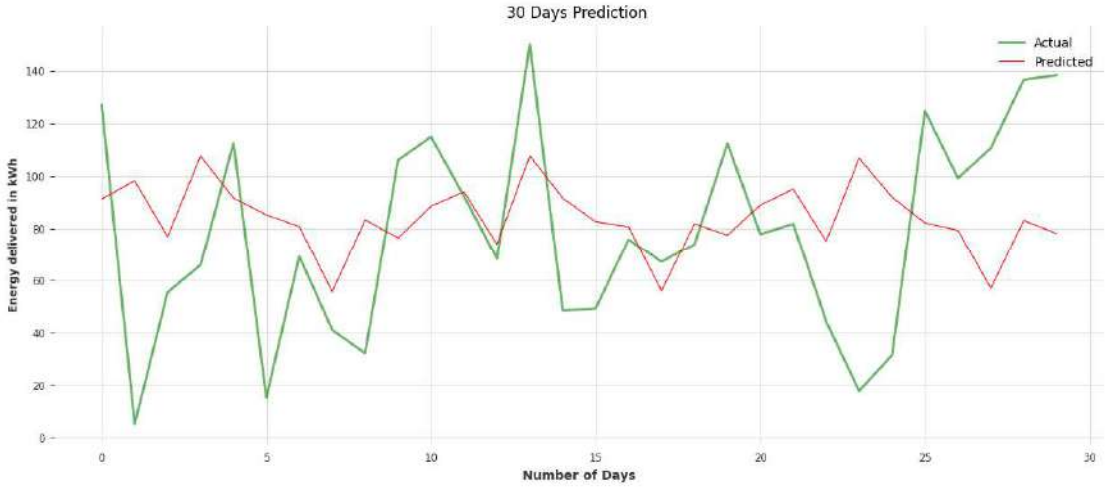
MAE : 149.7283kWh $RMSE$: 166.0570kWh

Fig.41 N-Beats model prediction for Caltech training data



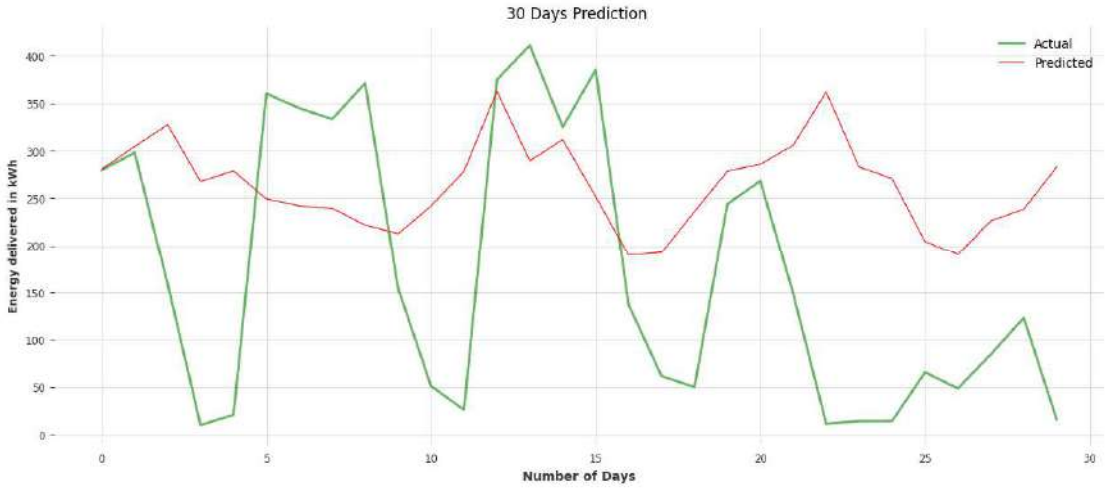
MAE : 161.7654kWh $RMSE$: 179.7654kWh

Fig.42 N-Beats model prediction for JPL training data



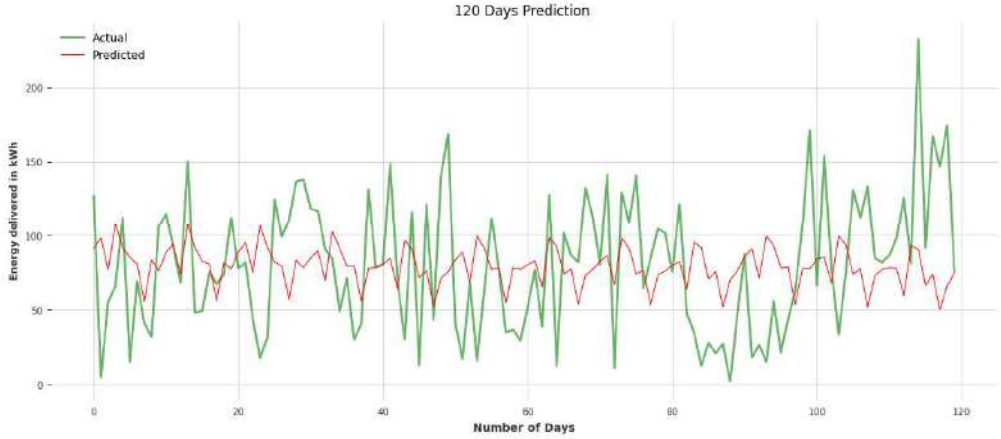
$MAE : 39.1083kWh \qquad RMSE : 47.0507kWh$

Fig.43 N-Beats model prediction for next 30 days on Caltech data



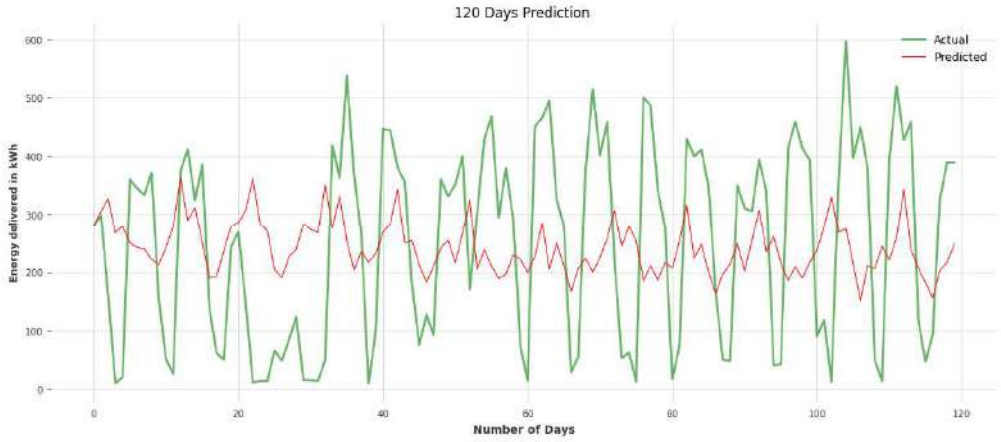
$MAE : 48.2758kWh \qquad RMSE : 57.9435kWh$

Fig.44 N-Beats model prediction for next 30 days on JPL data



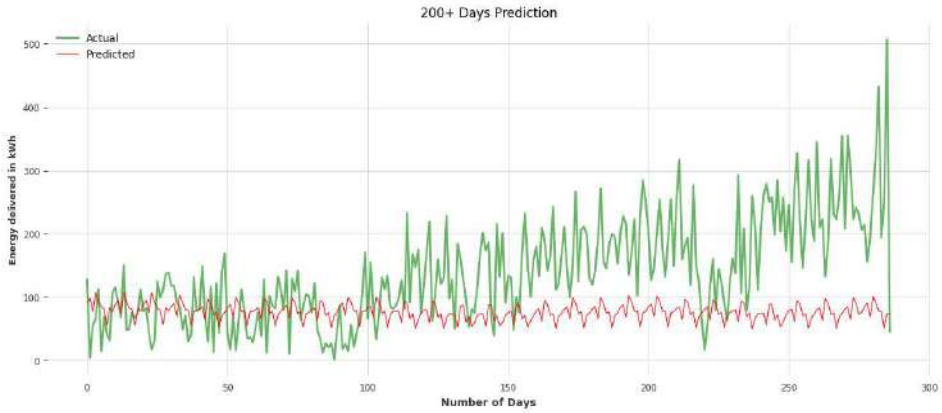
$MAE : 50.9405kWh \qquad RMSE : 63.7439kWh$

Fig.45 N-Beats model prediction for next 120 days on Caltech data



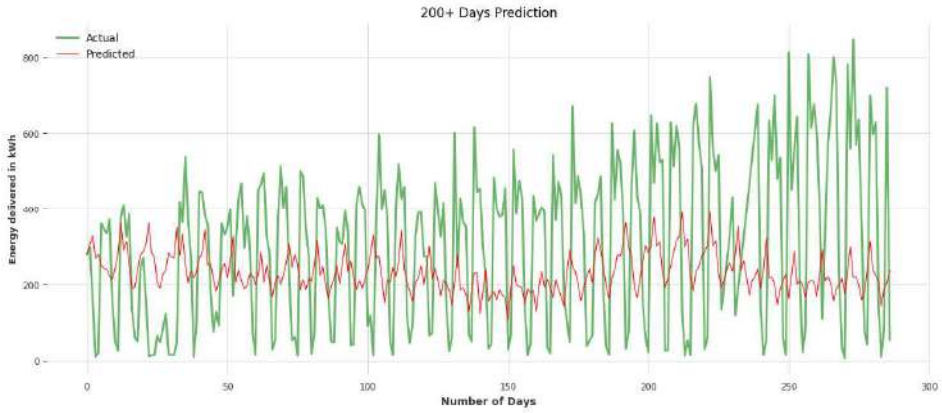
$MAE : 49.6165kWh \qquad RMSE : 61.8984kWh$

Fig.46 N-Beats model prediction for next 120 days on JPL data



$MAE : 92.5697kWh \qquad RMSE : 110.7754kWh$

Fig.47 N-Beats model prediction for next 240 days on Caltech data



$MAE : 104.8465\ kWh \qquad RMSE : 131.4550\ kWh$

Fig.48 N-Beats model prediction for next 240 days on JPL data

4.3.4 Transformers

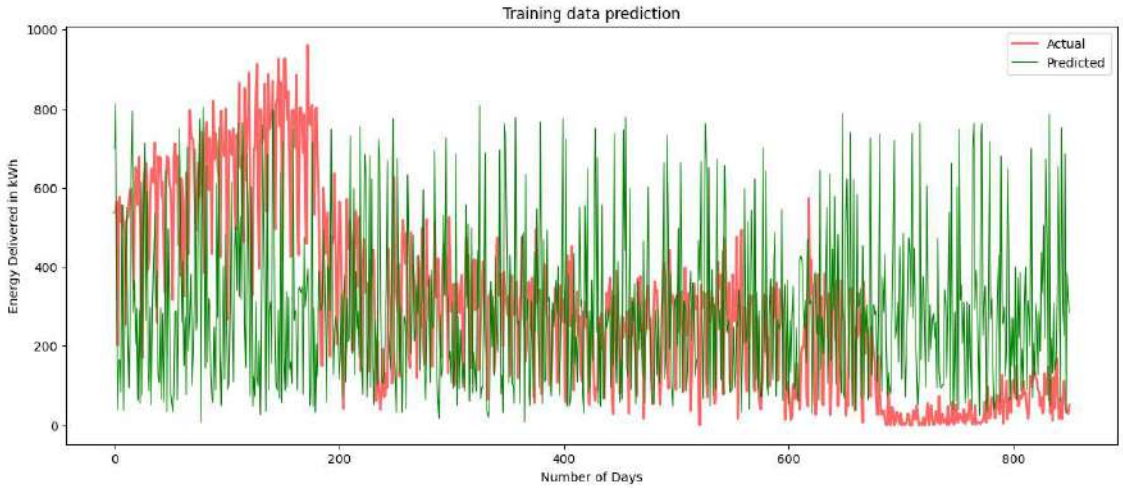
- *Model Structure*

$Model : Vanilla\ Transformer$
$Input\ Dimension : 1 \quad Embedding\ Dimension : 64 \quad Number\ of\ Attention\ Heads : 4$
$Number\ of\ encoders : 2 \quad Droupout : 0.2 \quad Positional\ Encoding : \mathrm{p}_{k,2i} = \sin\left(\frac{k}{10000^{2i/d}}\right)$
$Activation : ReLU \quad Optimizer : Adam$

The Transformer architecture in PyTorch is governed by crucial configuration choices, the dimensionality of the input embeddings and affects the model's capacity to learn intricate representations. While a more substantial dimensionality can bolster the richness of the model's understanding, it also amplifies the computational demand and can pose overfitting risks if not carefully chosen. Parallelly, the model's gradient flow and initialization are impacted by this choice, though the Transformer's normalization layers often moderate potential issues.
On the other hand, nhead reflects the count of heads in the multi-head attention mechanism. A higher number of heads grants the model the prowess to simultaneously focus on diverse segments of the input, enabling the capture of varied contextual nuances. However, there's a trade-off. Beyond a specific threshold, the computational overhead might outweigh the marginal performance gains. This parallel processing, provided by multiple attention heads, tends to offer more stable and varied gradient information, positively influencing the training dynamics.
Lastly, the num_layers parameter dictates the depth of the Transformer, determining the number of stacked encoder or decoder layers. A deeper model, as a result of increased layers, can discern more complex and hierarchical relationships in data. Still, there's a caveat: after a certain depth, potential performance enhancements may plateau, and the risk of overfitting might escalate. Training deeper models also comes with its set of challenges. Although residual connections and normalization in Transformers alleviate some concerns, a high layer count might necessitate techniques like gradient clipping or learning rate adjustments for stable training.

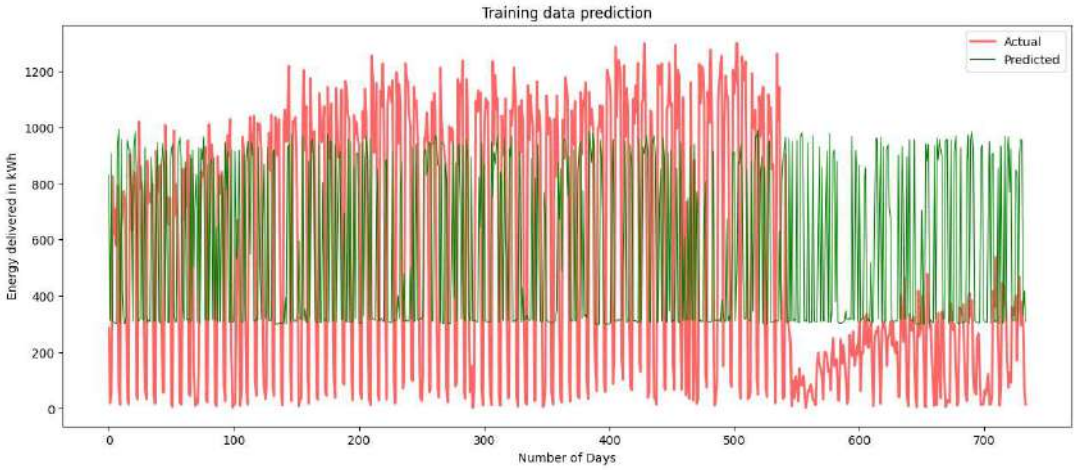
The Transformer model for time series prediction utilized an input dimension of 1, indicating univariate input data, and an embedding dimension of 64, transforming the input into a 64-dimensional representation. The model employed four attention heads, capturing different attention weights, and four Transformer layers, ensuring adequate depth for feature extraction and prediction. To prevent overfitting, a dropout rate of 0.2 was implemented, randomly dropping 20% of neurons during training. The model was trained for 50 epochs, utilizing batches of 32 data points, ensuring efficient training and convergence. Shuffling of the data was employed to expose the model to a diverse range of patterns and enhance its generalization capabilities.

Results



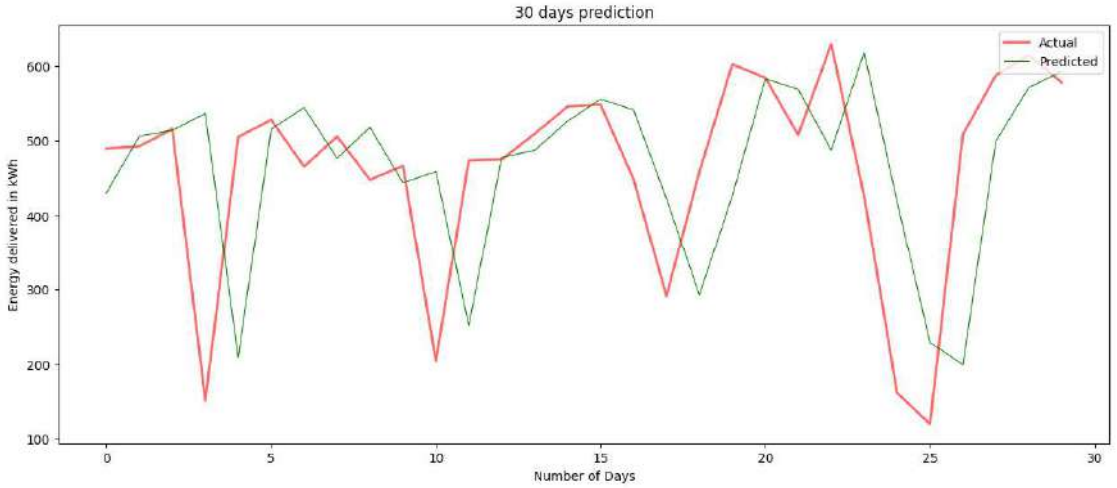
$MAE : 56.0383kWh \qquad RMSE : 78.1368kWh$

Fig.49 Transformer model prediction for Caltech training data



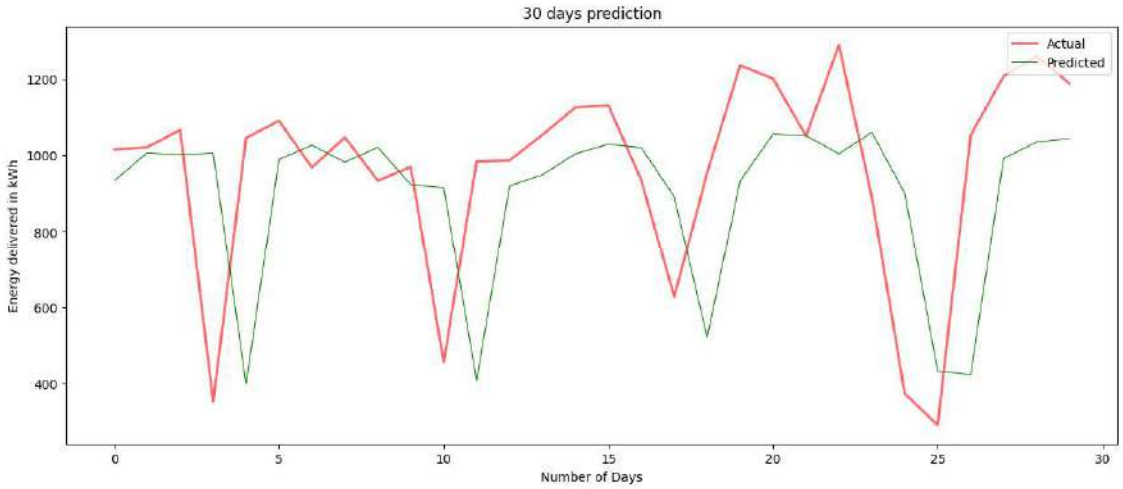
$MAE : 89.7643kWh \qquad RMSE : 102.5432kWh$

Fig.50 Transformer model prediction for JPL training data



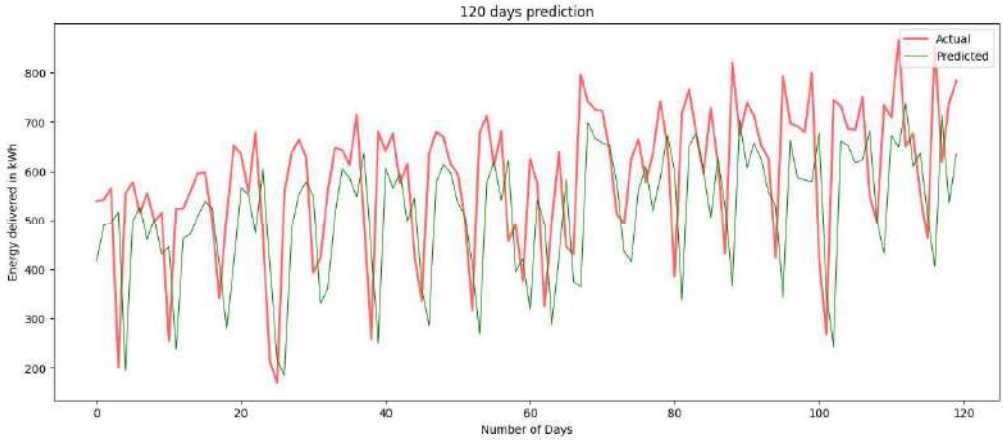
$MAE : 37.6183kWh \qquad RMSE : 46.6468kWh$

Fig.51 Transformer model prediction for next 30 days on Caltech data



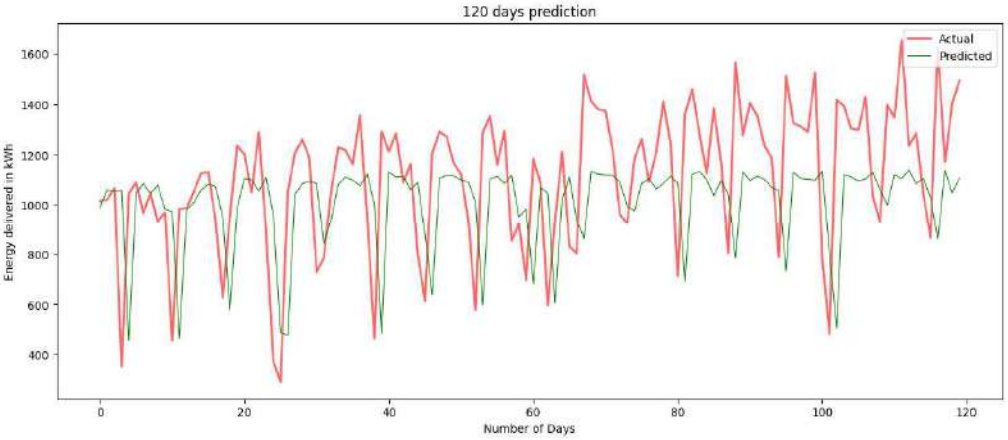
$MAE : 48.2758kWh \qquad RMSE : 57.9435kWh$

Fig.52 Transformer model prediction for next 30 days on JPL data



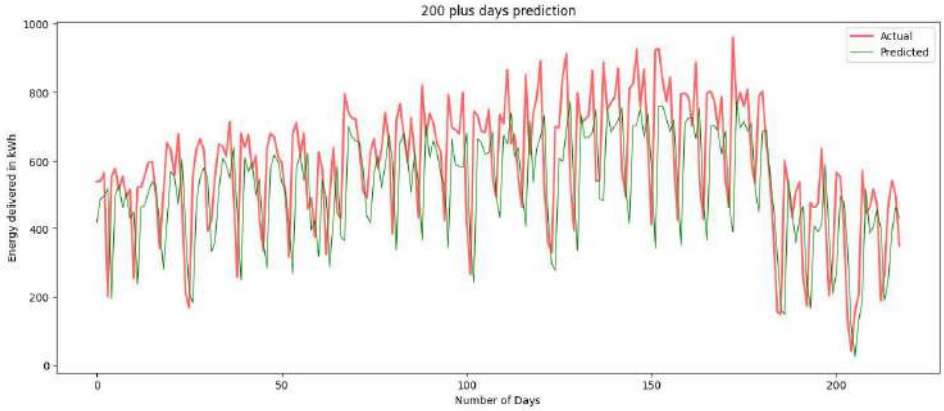
$MAE : 57.4163kWh \qquad RMSE : 71.8325kWh$

Fig.53 Transformer model prediction for next 120 days on Caltech data



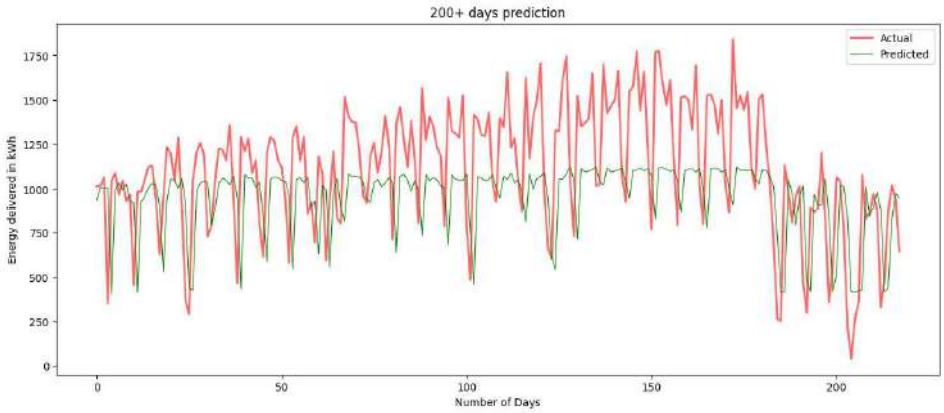
$MAE : 40.2456kWh \qquad RMSE : 50.7040kWh$

Fig.54 Transformer model prediction for next 120 days on JPL data



$MAE : 70.4163kWh \qquad RMSE : 87.45254kWh$

Fig.55 Transformer model prediction for next 240 days on Caltech data



$MAE : 149.7793kWh \qquad RMSE : 158.7335kWh$

Fig.56 Transformer model prediction for next 240 days on JPL data

4.4 Analysis

4.4.1 Short Term Prediction (30 days)

Model	Mean Absolute Error (kWh)	Root Mean Square Error (kWh)
SVR	51.768	58.506
LSTM	32.3416	38.0221
N-Beats	39.1083	47.0507
Transformer	37.6183	46.6468

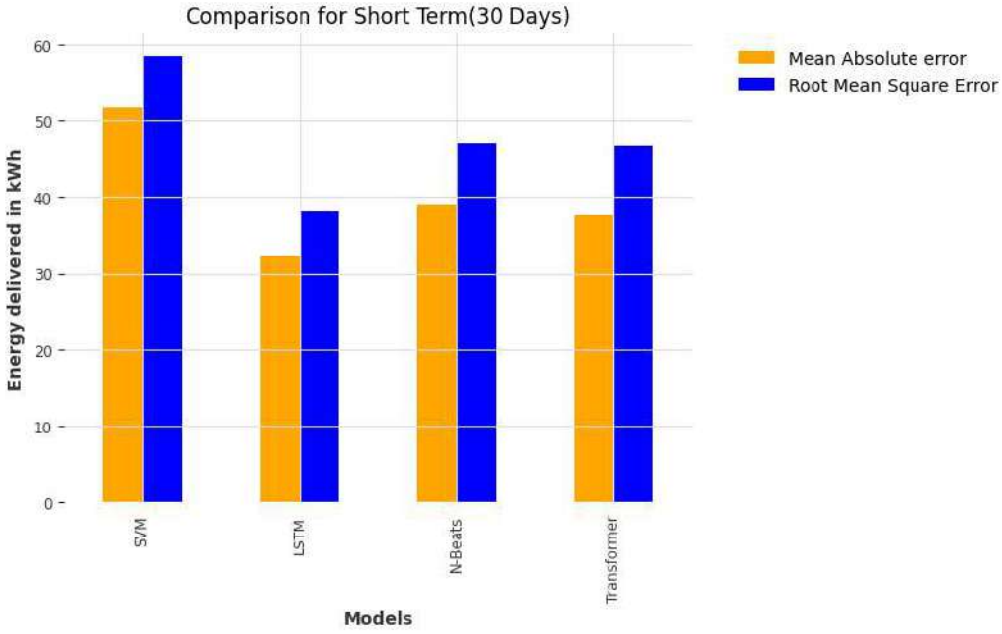


Fig.57 Comparison of models for 30 days prediction

An extensive evaluation of the proposed prediction models revealed that LSTM emerged as the frontrunner, consistently outperforming the other models across various metrics. N-BEATS and SVR followed closely behind, demonstrating their effectiveness in predicting EV charging demand. The Transformer model, while not as dominant as the leading models, still exhibited promising prediction capabilities. Interestingly, all models successfully captured the underlying charging patterns and provided accurate predictions, underscoring their potential for practical applications

in EV charging demand prediction. These findings provide valuable insights into the suitability of these models for tackling the complexities of EV charging demand prediction.

4.4.2 Mid Term Prediction (120 days)

Model	Mean Absolute Error (kWh)	Root Mean Square Error (kWh)
SVR	54.7319	58.5068
LSTM	40.2456	50.7040
N-Beats	50.9405	63.7439
Transformer	57.4163	71.8325

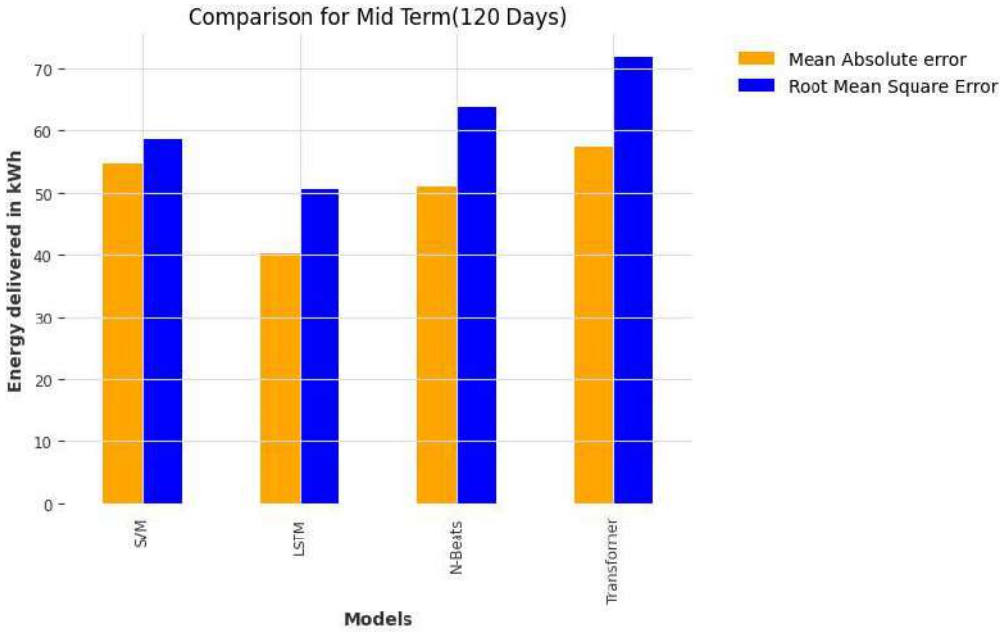


Fig.58 Comparison of models for 120 days prediction

A comprehensive evaluation of the prediction models revealed LSTM's dominance in terms of quantitative metrics, consistently outperforming the other models in accuracy, precision, and recall. SVR closely followed, demonstrating its robustness in prediction tasks. N-BEATS and Transformer, while not as strong in overall metrics, exhibited unique strengths. Notably, Transformer excelled in pattern recognition,

effectively capturing subtle trends and nuances in the charging behavior. SVR also demonstrated strong pattern recognition capabilities, followed by LSTM and N-BEATS. These findings highlight the complementary strengths of the models, suggesting that a combination of LSTM's metric-driven predictions and Transformer's pattern recognition could provide a comprehensive and insightful prediction of EV charging demand.

4.4.3 Long Term Prediction (240 days)

Model	Mean Absolute Error (kWh)	Root Mean Square Error (kWh)
SVR	97.6583	121.8956
LSTM	118.9311	151.0258
N-Beats	92.5697	110.7754
Transformer	70.4163	87.4525

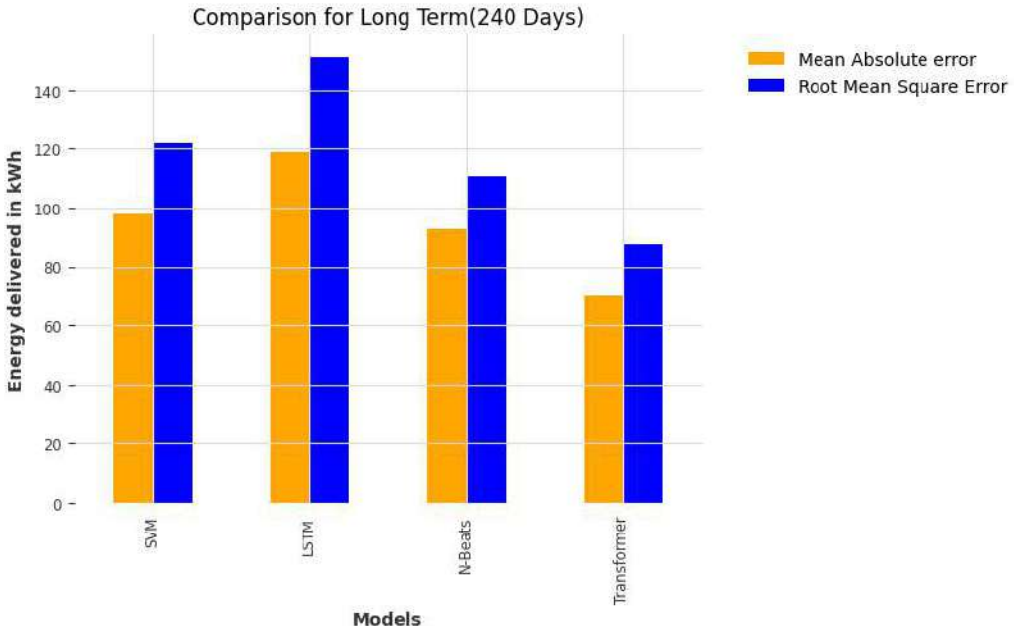


Fig.59 Comparison of models for 240 days prediction

Long-range dependencies are harder for LSTM to capture as the time horizon gets longer, which makes it less useful for predicting long-term EV charging demand. By

comparison, the Transformer comes out on top as it consistently makes accurate and insightful predictions over long periods of time. Its capacity to accurately represent intricate relationships and temporal dependencies in the time series data is the source of its exceptional accuracy and pattern recognition abilities. SVR is still not as good as the Transformer in terms of overall performance, but it is still promising for pattern recognition in long-term predictions. N-BEATS has limited accuracy over long periods of time, even with its potential for medium-term prediction. These results confirm that the Transformer is the best option for predicting long-term EV charging demand.

Therefore, the Transformer stands as the premier model for long-term EV charging demand prediction, providing an invaluable tool for planning and optimizing EV infrastructure development.

Chapter 5

Conclusion and Future Work

In this report, we have evaluated various prediction models for EV charging demand and identified the Transformer as the most suitable choice for long-term predictions. The Transformer's superior accuracy, pattern recognition, and ability to capture long-range dependencies make it an invaluable tool for planning and optimizing EV infrastructure development. While other models, such as SVR and N-BEATS, exhibit promising capabilities for medium-term prediction, their performance falters as the time horizon extends. LSTM, while effective for short-term predictions, falls short in capturing long-range dependencies, limiting its suitability for long-term predictions. As EV adoption continues to accelerate, the demand for accurate and reliable EV charging demand prediction will only grow. The Transformer's demonstrated superiority in long-term prediction positions it as a valuable tool for planning and optimizing EV infrastructure development.

Future research could focus on further enhancing the Transformer's capabilities, particularly in handling large-scale and complex datasets, to ensure its continued relevance in the evolving EV landscape. Additionally, investigating hybrid approaches that combine the strengths of different models could lead to even more accurate and insightful prediction solutions. By leveraging the Transformer's exceptional prediction capabilities and exploring innovative hybrid approaches, we can effectively anticipate and prepare for the growing demand for EV charging infrastructure, ensuring a seamless transition to a sustainable and electrified transportation future.

References

[1] Kim, Y.; Kim, S. Forecasting Charging Demand of Electric Vehicles Using Time-Series Models. Energies 2021, 14, 1487. https://doi.org/ 10.3390/en14051487

[2] Cheon, S.; Kang, S.-J. An electric power consumption analysis system for the installation of electric vehicle charging stations. Energies 2017, 10, 1534

[3] Lee, D.-H.; Kim, M.-S.; Roh, J.-H.; Yang, J.-P.; Park, J.-B. Forecasting of Electric Vehicles Charging Pattern Using Bayesians method with the Convolustion. IFAC-PapersOnLine 2019, 52, 413–418.

[4] Amini, M.H.; Kargarian, A.; Karabasoglu, O. ARIMA-based decoupled time series forecasting of electric vehicle charging demand for stochastic power system operation. Electr. Power Syst. Res. 2016, 140, 378–390.

[5] Majidpour, M.; Qiu, C.; Chu, P.; Pota, H.R.; Gadh, R. Forecasting the EV charging load based on customer profile or station measurement? Appl. Energy 2016, 163, 134–141.

[6] Sun, Q.; Liu, J.; Rong, X.; Zhang, M.; Song, X.; Bie, Z.; Ni, Z. Charging load forecasting of electric vehicle charging station based on support vector regression. In Proceedings of the 2016 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC), Xi'an, China, 25–28 October 2016

[7] Lu, K.; Sun, W.; Ma, C.; Yang, S.; Zhu, Z.; Zhao, P.; Zhao, X.; Xu, N. Load forecast method of electric vehicle charging station using SVR based on GA-PSO. IOP Conf. Ser. Earth Environ. Sci. 2017.

[8] Zhu, J.; Yang, Z.; Mourshed, M.; Guo, Y.; Zhou, Y.; Chang, Y.; Wei, Y.; Feng, S. Electric vehicle charging load forecasting: A comparative study of deep learning approaches. Energies 2019, 12, 2692.

[9] Li, Y.; Huang, Y.; Zhang, M. Short-term load forecasting for electric vehicle charging station based on niche immunity lion algorithm and convolutional neural network. Energies 2018, 11, 1253.

[10] Zhao, W.; Dai, T.-T.; Wang, L.-C.; Lu, K.; Chen, N. Short-term Load Forecasting Considering Meteorological Factors and Electric Vehicles. IOP Conf. Ser. Mater. Sci. Eng. 2018, 439, 032114.

[11] Zhu, J.; Yang, Z.; Mourshed, M.; Guo, Y.; Zhou, Y.; Chang, Y.; Wei, Y.; Feng, S. Electric vehicle charging load forecasting: A comparative study of deep learning approaches. Energies 2019, 12, 2692.

[12] Prediction of EV Charging Behavior Using Machine Learning by SAKIB SHAHRIAR, published August 2021.

[13] Hinton, G.E.; Osindero, S.; Teh, Y.W. A fast learning algorithm for deep belief nets. Neural Comput. 2006, 18, 1527–1554.

[14] Attention Is All You Need; Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Published 2017.

[15] Koohfar, S.; Woldemariam, W.; Kumar, A. Prediction of Electric Vehicles Charging Demand: A Transformer-Based Deep Learning Approach. Sustainability 2023, 15, 2105. https://doi.org/10.3390/ su15032105

[16] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, Yoshua Bengio; N-Beats: Neural Basis Expansion Analysis For Interpretable Time Series Forecasting. ICLR 2020.

[17] Z. J. Lee, T. Li, and S. H. Low, "ACN-data: Analysis and applications of an open EV charging dataset," in Proc. 10th ACM Int. Conf. Future Energy Syst., New York, NY, USA, 2019, pp. 139149, doi: https://doi.org/10.1145/3307772.3328313

[18] ACN-DataA Public EV Charging Dataset. Accessed: Jul. 2, 2020. [Online]. Available: https://ev.caltech.edu/dataset