



TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

**SHORT-TERM ELECTRICAL LOAD FORECASTING FOR
BANESHWOR FEEDER USING MACHINE AND DEEP
LEARNING MODELS**

Submitted by

SUJIT KOIRALA
(PUL075MSPSE016)

A THESIS REPORT
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE IN POWER SYSTEM ENGINEERING

DEPARTMENT OF ELECTRICAL ENGINEERING
PULCHOWK CAMPUS, LALITPUR, NEPAL

JANUARY, 2026

COPYRIGHT

This thesis may be accessed for academic and research purposes at the Library, Department of Electrical Engineering, Pulchowk Campus, and Institute of Engineering. Permission for substantial reproduction of this report for scholarly use may be granted by the supervisors or, if unavailable, by the Head of Department. Proper acknowledgment must be given to the author and the Department of Electrical Engineering, Pulchowk Campus, Institute of Engineering, for any use of the material herein. Any reproduction or use of this report for commercial purposes without written consent from both the Department and the author is strictly prohibited. Requests for such permissions should be directed to:

Head
Department of Electrical Engineering
Pulchowk Campus, Institute of Engineering
Pulchowk, Lalitpur
Nepal

Certificate of Approval

The undersigned certify that they have read and recommended to the Institute of Engineering for acceptance, a THESIS entitled **Short-Term Electrical Load Forecasting for Baneshwor Feeder Using Machine and Deep Learning Models** submitted by Sujit Koirala (PUL075MSPSE016) as a partial requirement for the Master of Science in Power System Engineering. After review, we recommend its acceptance by the Institute of Engineering.

Amrit Dhakal

Supervisor

Assistant Professor, Department of
Electrical Engineering

Pulchowk Campus, IOE, Tribhuvan
University

Deependra Neupane

Supervisor

Assistant Professor, Department of
Electrical Engineering

Pulchowk Campus, IOE, Tribhuvan
University

Dr. Kamal Chapagain

External Examineer, Department of

Electrical and Electronics Engineering

Kathmandu University

Dr. Bishal Silwal

Program Coordinator, Msc. in Power
System Engineering

Pulchowk Campus, IOE, Tribhuvan
University

Assoc. Prof. Jeetendra Chaudhary

HoD, Department of Electrical Engi-
neering

Pulchowk Campus, IOE, Tribhuvan
University

ABSTRACT

Reliable short-term forecasting of electrical load is essential for the optimal management and planning of modern power systems and electricity markets. As load patterns become more variable due to weather, time-based trends, and social factors, traditional statistical approaches often fail to capture the underlying non-linearities. In this study, short-term load forecasting for the Baneshwor Feeder is addressed using a range of machine learning and deep learning models, with forecast intervals tailored to market operations. The analysis utilizes historical hourly load data and meteorological features such as temperature, solar radiation, and humidity. Data preparation included handling missing values, treating outliers, extracting temporal features, and encoding cyclical time variables. The models explored include Linear Regression, Ridge Regression, Support Vector Regression, Random Forest, Gradient Boosting, and XGBoost, as well as deep learning models like LSTM and GRU. Hyperparameters were optimized to enhance predictive accuracy. Model performance was evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and R-squared (R^2). The GRU model with additional lag features delivered the highest accuracy (RMSE: 0.289, R^2 : 0.879, MAPE: 7.364%), with LSTM as the next best (RMSE: 0.314, R^2 : 0.857, MAPE: 7.612%). Among machine learning models, XGBoost performed best (RMSE: 0.384, R^2 : 0.831, MAPE: 12.693%). These results indicate that the models are suitable for hour-ahead intraday market forecasting. The study demonstrates the value of advanced neural networks and ensemble methods for feeder-level load prediction, supporting operational planning and market participation in power distribution.

Keywords: *short-term load forecasting, machine learning, deep learning, GRU, LSTM, XGBoost, electricity markets, day-ahead market, intraday market, power distribution systems*

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor and faculty members of the Department of Electrical Engineering for their valuable guidance, continuous support, and encouragement throughout the course of this project. Their technical insights and constructive feedback were instrumental in shaping this work.

I am also thankful to the Nepal Electricity Authority and relevant data-providing institutions for making the load and meteorological data available for this study. Their cooperation greatly contributed to the successful completion of the analysis.

Special thanks go to my friends and colleagues for their support, discussions, and motivation during the project period.

Finally, I would like to express my heartfelt appreciation to my family for their constant encouragement and support throughout my academic journey.

Sujit Koirala (PUL075MSPSE016)

TABLE OF CONTENTS

Copyright	i
Approval page	ii
Abstract	iii
Acknowledgements	iv
Table of contents	v
List of tables	viii
List of figures	ix
List of acronyms and abbreviations	x
CHAPTER ONE: INTRODUCTION	1
1.1. Background	1
1.2. Problem Statement	3
1.3. Objectives	5
1.4. Scope and Limitations	5
1.5. Report Organization	7
CHAPTER TWO: LITERATURE REVIEW	9
2.1. Related Works	9
2.2. Electricity Markets and Load Forecasting	12
2.2.1. Structure of Electricity Markets	12
2.2.2. Role of Load Forecasting in Market Operations	14
2.2.3. Relevance to Nepal's Power Sector	15
2.3. Theoretical Background of Forecasting Models	16
2.3.1. Machine Learning Models	16
2.3.2. Deep Learning Models	19
CHAPTER THREE: RESEARCH METHODOLOGY	24
3.1. Overall Workflow	24
3.2. Data Acquisition	26
3.2.1. Data Sources	27

3.2.2.	Load Data Acquisition and Structuring Process	28
3.2.3.	Weather Data Acquisition and Structuring.....	29
3.2.4.	Final Merging of Load and Weather Data	29
3.3.	Data Preprocessing	30
3.4.	Data Input Structure	32
3.4.1.	Raw Data Format	32
3.4.2.	Feature Engineering and Input Vector.....	32
3.4.3.	Input Structure for Machine Learning Models	33
3.4.4.	Input Structure for Deep Learning Models (LSTM/GRU).....	34
3.4.5.	Lag Feature Configuration for Deep Learning	35
3.4.6.	Complete Input Pipeline Summary	35
3.5.	Model Development	36
3.6.	Model Training and Validation	37
3.6.1.	Training of Machine Learning Models	37
3.6.2.	Training of Deep Learning Models	39
3.6.3.	Validation Approach	42
3.7.	Performance Evaluation	42
3.7.1.	Mean Absolute Error (MAE)	43
3.7.2.	Root Mean Squared Error (RMSE)	43
3.7.3.	Mean Absolute Percentage Error (MAPE)	44
3.7.4.	Coefficient of Determination (R^2 Score)	44
CHAPTER FOUR: RESULTS AND DISCUSSION		46
4.1.	Exploratory Data Analysis	46
4.1.1.	Outlier Treatment	51
4.2.	Model Performance Results	54
4.2.1.	Machine Learning Model Performance.....	54
4.2.2.	Deep Learning Model Performance	58
CHAPTER FIVE: CONCLUSION		61
5.1.	Conclusion	61

5.2.	Research Limitations	62
5.3.	Implications	63
5.4.	Recommendation	64
REFERENCES		65

LIST OF TABLES

Table 3.1	Raw Input Data Format	32
Table 3.2	Sample Data Records from Cleaned Dataset	32
Table 3.3	Engineered Feature Set for Model Input	33
Table 3.4	Lag Feature Configurations for Deep Learning Models	35
Table 4.1	Descriptive Statistics of MW Column (After Outlier Removal) ..	54
Table 4.2	Hyperparameter Search Space for Machine Learning Models ..	55
Table 4.3	Machine Learning Models Evaluation Matrix.....	56
Table 4.4	Hyperparameter Search Space for Deep Learning Models	58
Table 4.5	Deep Learning Models Evaluation Matrix	59

LIST OF FIGURES

Figure 1.1	Substation & Transmission Line Network Baneshwor	3
Figure 2.1	Electricity Market Timeline and Forecasting Requirements	13
Figure 2.2	Architecture LSTM	21
Figure 2.3	Architecture GRU	23
Figure 3.1	Methodology Block Diagram	26
Figure 3.2	Feature Correlation Matrix	31
Figure 3.3	Sliding Window Sequence Construction for LSTM/GRU Input .	34
Figure 3.4	Complete Data Input Pipeline from Raw Data to Model Nodes .	36
Figure 4.1	Daily Average Electricity Load Over Time	47
Figure 4.2	Load Distribution by Hour	48
Figure 4.3	Average Load by Month	49
Figure 4.4	Air Temperature Variation Over the Study Period	50
Figure 4.5	Global Solar Radiation Variation Over the Study Period	50
Figure 4.6	Relative Humidity Variation Over the Study Period	51
Figure 4.7	Boxplot of MW Column Before and After Outlier Removal	52
Figure 4.8	Relative Humidity – Outlier Removal Comparison	53
Figure 4.9	Air Temperature – Outlier Removal Comparison	53
Figure 4.10	Global Solar Radiation – Outlier Removal Comparison	53
Figure 4.11	XBoost (Tuned) Actual vs Predicted	57
Figure 4.12	GRU Model Actual vs Predicted Values (Last 200 Test Points) .	60

LIST OF ACRONYMS AND ABBREVIATIONS

ANN	:	Artificial Neural Network
ARIMA	:	Autoregressive Integrated Moving Average
BS	:	Bikram Sambat (Nepali Calendar)
CNN	:	Convolutional Neural Network
DAM	:	Day-Ahead Market
DHM	:	Department of Hydrology and Meteorology
DL	:	Deep Learning
DWT	:	Discrete Wavelet Transform
GRU	:	Gated Recurrent Unit
IDM	:	Intraday Market
LSTM	:	Long Short-Term Memory
MAE	:	Mean Absolute Error
MAPE	:	Mean Absolute Percentage Error
ML	:	Machine Learning
MW	:	Megawatt
NEA	:	Nepal Electricity Authority
PCA	:	Principal Component Analysis
RF	:	Random Forest
RMSE	:	Root Mean Squared Error
RNN	:	Recurrent Neural Network

R^2	:	Coefficient of Determination
RTM	:	Real-Time Market
STLF	:	Short-Term Load Forecasting
SVR	:	Support Vector Regression
TCN	:	Temporal Convolutional Network
XGBoost	:	Extreme Gradient Boosting

CHAPTER ONE: INTRODUCTION

1.1. Background

This research focuses on short-term electrical load forecasting at the feeder level using data-driven machine learning and deep learning techniques. Historical load data combined with weather and temporal features were used to model and predict hourly power demand. Multiple forecasting models were developed and evaluated to identify the most effective approach for accurate and reliable load prediction, with particular relevance to electricity market operations and grid management.

Electricity demand is never constant. It rises and falls with daily routines, temperature changes, business hours, and countless other factors. For a power system operator, being able to predict this demand even just a few hours ahead can make a huge difference. Accurate short-term forecasting helps optimize generation schedules, reduce operational costs, manage peak hours more confidently, and maintain a reliable supply [1, 2]. In modern electricity markets, such forecasts are fundamental to efficient market clearing, optimal bidding strategies, and maintaining supply-demand balance in real-time [3, 4].

The Need for Short-Term Electricity Load Forecasting: Short-term load forecasting has become indispensable in today's power systems for several interconnected reasons. First, from an *operational perspective*, accurate hour-ahead and day-ahead forecasts enable system operators to commit the optimal mix of generating units, schedule maintenance without compromising reliability, and procure reserves efficiently. Second, from an *economic perspective*, forecasting errors translate directly into financial costs—either through over-procurement of expensive peaking generation or through penalties incurred in imbalance settlement mechanisms. Third, from a *market participation perspective*, utilities and load-serving entities must submit bids and offers based on anticipated demand; the accuracy of these forecasts determines their competitiveness and profitability in

day-ahead and intraday markets [3]. Fourth, as power systems integrate more *variable renewable energy sources*, the ability to accurately predict net load (demand minus renewable generation) becomes crucial for maintaining system stability and minimizing curtailment [5].

Short-Term Load Forecasting (STLF) typically focuses on horizons ranging from one hour to a day ahead. These forecasts are critical for economic dispatch, unit commitment, load flow analysis, and real-time operation. In the context of electricity markets, STLF plays a pivotal role across multiple market timelines: intraday markets depend on rolling hour-ahead predictions for position adjustments and real-time balancing, while real-time balancing markets utilize very short-term forecasts for ancillary services and imbalance management [6, 7]. Hour-ahead forecasting is particularly critical as it enables utilities to make timely adjustments to their market positions based on the most recent demand patterns and weather conditions. Traditionally, utilities relied on statistical approaches such as linear regression, ARIMA, exponential smoothing, and Holt-Winters [8, 9]. These techniques can work well when patterns are simple, but they struggle with real-world load curves that are nonlinear, noisy, and influenced by many interacting variables.

Machine Learning models like Random Forest, Support Vector Regression, and XGBoost have shown strong results in energy-related forecasting tasks [1]. Their ability to capture nonlinear relationships makes them a natural fit for electricity load prediction. Likewise, Deep Learning approaches, especially recurrent neural networks such as LSTM and GRU, can learn temporal dependencies more effectively than traditional models [2].

The Baneshwor Feeder of the Nepal Electricity Authority serves a mixed group of consumers in the Baneshwor region of Kathmandu Valley, which is shown in single line diagram shown in Figure 1.1. Its load pattern reflects residential lifestyles, commercial activity, seasonal tourism impacts, and local weather changes. Daily and weekly cycles are clearly visible, but there are also irregularities that simple models fail to capture. As power consumption continues to grow and diversify, the

ability to forecast the feeder's short-term load accurately has become even more important [9]. This creates a strong motivation to investigate how modern ML and DL models can improve forecasting performance for this specific feeder.

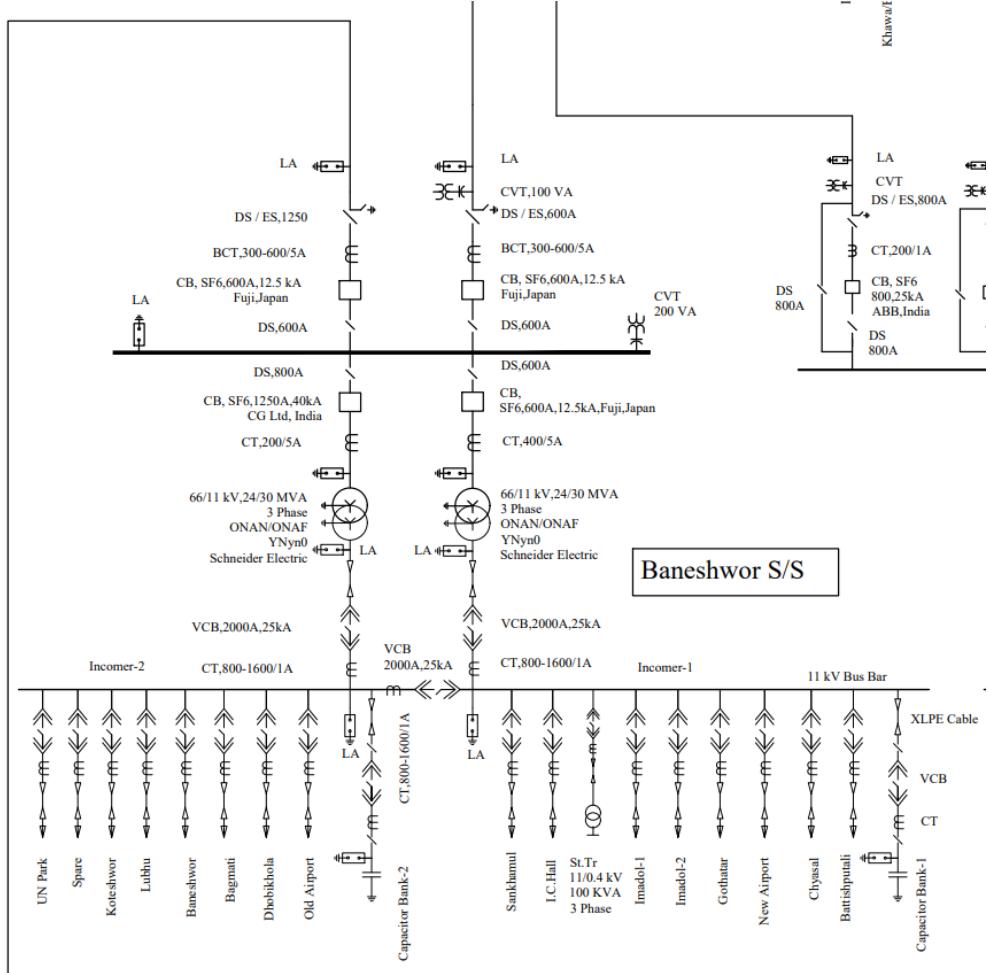


Figure 1.1 Substation & Transmission Line Network Baneshwor

1.2. Problem Statement

The current forecasting practices for the Baneshwor Feeder rely heavily on manual estimation or basic statistical techniques. These methods do not fully capture the nonlinear and dynamic nature of the load profile, especially when multiple influencing factors, like temperature, humidity, rainfall, weekends, and special events come into play. As a result, prediction errors tend to increase during peak hours, sudden weather changes, and seasonal transitions. Although some studies

have explored short-term electrical load forecasting for Kathmandu valley, more comprehensive research is lacking, particularly studies that systematically compare multiple machine learning and deep learning models with hyperparameter tuning, and that frame the forecasting problem within the context of electricity market operations.

Inaccurate short-term forecasts have several consequences for both grid operation and market efficiency. They can affect how generation is scheduled, leading to either unnecessary reserve margins or inadequate supply. They may increase operational costs and technical losses at the distribution level. In the worst cases, poor foresight during high-demand periods can create voltage drops, reliability concerns, or inefficient load-shedding decisions. From an electricity market perspective, forecasting errors directly impact bidding accuracy in intraday markets, lead to costly imbalance settlements in real-time markets, and reduce the economic efficiency of power procurement [10]. As Nepal’s power sector moves toward market liberalization and increased cross-border electricity trade, the financial implications of forecast errors become increasingly significant. Hour-ahead predictions are particularly critical for intraday market operations, where utilities must adjust their positions based on updated demand expectations and respond rapidly to changing load conditions [11].

Despite the availability of historical load and weather data, there has not been a systematic study applying and comparing advanced machine learning and deep learning approaches comprehensively specifically for the Baneshwor Feeder. This lack of a data-driven forecasting system means operators do not yet benefit from models that are capable of learning complex relationships within the data.

This thesis aims to address this accurate short-term electrical load forecasting task by building a complete forecasting framework using multiple ML and DL models, evaluating their performance, hyperparameter tuning and identifying the most suitable approach for accurate short-term load prediction of the Baneshwor Feeder.

1.3. Objectives

To develop and evaluate machine learning and deep learning models for short-term electrical load forecasting of the Baneshwor Feeder to improve prediction accuracy and operational efficiency, with forecasting horizons aligned to hour-ahead intraday market operational timelines.

1. To collect and preprocess historical load data and relevant influencing factors such as weather variables and calendar effects for the Baneshwor Feeder.
2. To implement and evaluate various machine learning models including Support Vector Regression (SVR), Random Forest (RF), and XGBoost, as well as deep learning models such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), using standard error metrics (e.g., RMSE, MAPE, MAE, R-squared).
3. To develop forecasting capabilities suitable for day-ahead scheduling and hour-ahead market operations, enabling accurate demand prediction for electricity market participation and grid balancing.
4. To recommend the most suitable forecasting model for operational use in the Baneshwor Feeder, considering both prediction accuracy and applicability to energy market requirements.

1.4. Scope and Limitations

This study is geographically limited to the Baneshwor Feeder under the Nepal Electricity Authority. The temporal scope focuses on short-term load forecasting with a prediction horizon of one hour ahead, utilizing historical hourly load data as the primary foundation for model development. This forecasting horizon is specifically designed to support hour-ahead intraday market operations, which represent the most dynamic and frequently traded timescale in modern electricity markets [3]. Hour-ahead forecasting enables utilities to make rapid position

adjustments, optimize real-time dispatch, and minimize imbalance costs in competitive market environments. The dataset encompasses historical load data from the Baneshwor Feeder, complemented by weather data including temperature, humidity, and rainfall, along with calendar data distinguishing weekdays, weekends, and holidays.

From a technical perspective, the research implements several machine learning models including Support Vector Regression (SVR), Random Forest, and XGBoost, alongside deep learning architectures such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks. The performance of these models is rigorously evaluated using standard metrics including Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and the coefficient of determination (R-squared). These metrics are directly relevant for assessing model suitability for hour-ahead electricity market applications, where forecast accuracy determines the ability to adjust market positions and minimize imbalance costs [10]. It should be noted that the accuracy of forecasts is inherently dependent on the quality and completeness of the historical data available. Additionally, this study does not extend to medium-term or long term forecasting horizons, and the scope explicitly excludes renewable generation forecasting and electricity price forecasting from its analysis.

Despite the promising results obtained in this study, certain limitations were encountered, primarily related to data availability, model assumptions, and scope of analysis.

1. Dependence on data quality: Forecast accuracy is limited by the completeness and reliability of the historical load and weather data. Missing values, sensor errors, or inconsistent reporting can influence model performance.
2. Model sensitivity to sudden changes: Unexpected events such as outages, festivals, abrupt weather shifts, or abnormal consumption patterns are difficult for data-driven models to predict accurately.

3. Deep learning computation constraints: Training LSTM and GRU models requires more computational resources and time compared to ML models. Their performance may vary depending on the hardware used.
4. Limited feature diversity: Although weather and calendar data are included, other influential factors like economic activities, special events, industrial load profiles, or electricity price signals are not part of the dataset.
5. Generalization across feeders: The models developed in this study are tailored specifically to the Baneshwor Feeder and may not generalize directly to other feeders without retraining or adaptation.
6. Electricity market scope: While the forecasting framework is designed with hour-ahead market timelines in mind, actual market bidding optimization and price forecasting are beyond the scope of this study. The focus remains on load quantity prediction rather than price-volume optimization.

1.5. Report Organization

Thesis structure is as follows; Chapter 1 describes electrical load, importance of short-term electrical load forecasting in the context of both grid operations and electricity markets, methods and techniques developed and used for electrical load forecasting, as well as objectives, scope, and limitations. Chapter 2 is a review of literature, where we critically review previous studies, identify potential gaps, and describe our approach to solve the problem. This chapter also covers the theoretical background of electricity markets and their relationship with load forecasting, as well as the mathematical foundations of machine learning and deep learning models. Chapter 3 describes the overall workflow, data acquisition, model development, training, validation, hyperparameter tuning, and model evaluation. Chapter 4 presents exploratory data analysis, model configurations, hyperparameter search spaces, and comparative model analysis based on evaluation metrics relevant to both operational planning and hour-ahead market participation. Chapter 5 concludes the thesis with research limitations, practical implications for hour-ahead

energy market operations, and future directions.

CHAPTER TWO: LITERATURE REVIEW

This chapter presents a comprehensive overview of existing research related to short term electrical load forecasting using machine learning and deep learning techniques. It examines previously published studies to understand commonly used methodologies, datasets, and performance evaluation approaches in the domain of power system load forecasting. Reviewing prior work helps to identify current research trends, strengths, and limitations of existing models, while also highlighting gaps that motivate the need for this study. By situating the present research within the context of established knowledge, this chapter provides a foundation for model selection and methodological design adopted in this work.

2.1. Related Works

There are many previous works done for electrical load forecasting from short-term electrical load forecasting, to medium-term and long-term. Most of the studies have done short-term load forecasting.

[9] employed Artificial Neural Networks for 24-hour short-term load forecasting, utilizing dew point temperature, dry bulb temperature, and humidity as input features. Their work demonstrated the effectiveness of ANN in capturing the relationship between weather variables and electrical load demand. Similarly, [12] utilized the Prophet model from Meta to perform short-term load forecasting, incorporating time, temperature, humidity, and weather forecast data as features. [13] conducted a study on medium-term load forecasting using ensemble machine learning models. They compared XGBoost and AdaBoost against traditional methods including SVR, decision trees, and Random Forest. Their results highlighted the superior performance of gradient boosting techniques for capturing complex load patterns. [1] tested five machine learning models and found XGBoost to be the most accurate for predictions, using historical load data, weather

information, and holiday indicators as input features. [14] analyzed three popular ML methods for load forecasting: Support Vector Machine, Random Forest, and LSTM. They proposed a fusion forecasting approach that combined outputs from all three models, demonstrating that ensemble methods could improve prediction accuracy beyond individual model performance. Different from above studies, [15] performed a comparison between optimization methods (Particle Swarm Optimization, Dandelion Optimizer, Growth Optimizer) and machine learning models (SVR, ANN) for instantaneous peak electrical load forecasting. They found that ANN combined with Growth Optimizer outperformed other models and identified a strong positive correlation between GDP and peak load demand. [16] conducted a comprehensive evaluation of various machine learning algorithms for power load prediction, including Support Vector Machines, LSTM, ensemble classifiers, and Recurrent Neural Networks. They emphasized the importance of data preprocessing methods, feature selection strategies, and performance assessment metrics in achieving accurate forecasts, they demonstrated that ensemble methods and deep learning approaches consistently outperformed traditional statistical models.

Deep learning is also widely used method for electrical load forecasting, [2] explored deep learning models for electricity demand forecasting in Kathmandu Valley along with machine learning model. They found LSTM demonstrating outstanding performance in terms of MAPE and RMSE. [8] also performed short-term electrical load forecasting for the Gothatar feeder, uses six input features and found that Recurrent Neural Networks outperformed baseline methods including Single Exponential Smoothing, Double Exponential Smoothing, and Hot-Winter's method. [17] conducted a comprehensive comparison of load forecasting methods, including Random Forest, SVR, XGBoost, Multi-Layer Perceptron, LSTM, Conv-1D models and found that LSTM achieved the lowest error rates across multiple evaluation metrics. [18] provided a comprehensive survey on deep learning-based short-term electricity load forecasting covering the past decade. They identified CNN-LSTM hybrid architectures as widely adopted solutions due to exceptional performance in capturing both spatial and temporal features.

Hybrid architecture are also widely adopted for time series forecasting. [19] proposed a hybrid deep learning model combining Gated Recurrent Units and Temporal Convolutional Networks with an attention mechanism for short-term load forecasting. GRU captured long-term dependencies in time series data, while TC-N efficiently learned patterns and features. Their approach demonstrated superior accuracy compared to standalone architectures. [20] developed a hybrid CNN-LSTM framework for short-term individual household load forecasting, CNN layers for feature extraction from input data and LSTM layers for sequence learning. This work demonstrated the effectiveness of combining convolutional and recurrent architectures for handling high volatility in household-level load data. In contrast, [21] proposed a parallel multichannel network approach using 1D CNN and Bidirectional LSTM for load forecasting in smart grids. Unlike traditional stacked CNN-LSTM architectures, their model independently processed spatial and temporal characteristics through parallel channels.

[22] model are also widely used in time-series forecasting task. [23] proposed a sparse transformer-based approach for electricity load forecasting that addressed the computational complexity limitations of standard transformer architectures, sparse attention mechanisms capture temporal dependencies more efficiently, achieving comparable accuracy to RNN-based state-of-the-art methods while being up to 5 times faster during inference. [24] developed a Time Augmented Transformer model for short-term electrical load forecasting, incorporating temporal features and self-attention mechanisms to capture complex dynamic non-linear sequence dependencies. Attention mechanism's capacity to capture complex dynamical patterns in multivariate data contributed to improved forecasting accuracy. [25] proposed a multivariate data slicing transformer neural network for load forecasting in power systems. The transformer model excelled in capturing spatiotemporal relationships by self-attention mechanisms. Their approach demonstrated superior performance in handling the intermittency and volatility characteristics, outperforming traditional statistical models and conventional machine learning methods.

Ensemble methods are also applied in electrical load forecasting task. [26] devel-

oped an enhanced stacked ensemble model combining Random Forest and XGBoost for renewable power and load forecasting, Random Forest first forecast the target variable, followed by XGBoost improving predictions through combination.

2.2. Electricity Markets and Load Forecasting

Electricity markets have fundamentally transformed how power systems are operated, moving from vertically integrated monopolies to competitive market structures where generation, transmission, and distribution are unbundled [4]. In this market-based environment, accurate load forecasting has evolved from a purely technical function to a critical economic activity that directly impacts market efficiency, operational costs, and financial outcomes for all market participants [3].

2.2.1. Structure of Electricity Markets

Modern electricity markets typically operate across multiple timeframes, each serving distinct operational and economic purposes [7]. Understanding these market structures is essential for appreciating why accurate load forecasting at different time horizons has become critically important:

1. **Day-Ahead Market (DAM):** The day-ahead market is typically the primary market for electricity trading, where participants submit bids and offers for delivery during each hour of the following day. Gate closure typically occurs at noon on the day before delivery (D-1). Market clearing determines hourly prices and scheduled quantities based on the intersection of supply and demand curves. Day-ahead load forecasts (24–48 hours ahead) form the foundation of bidding strategies, generation scheduling, and unit commitment decisions. Forecast accuracy at this horizon directly impacts the economic efficiency of day-ahead market outcomes and the financial performance of market participants [10].
2. **Intraday Market (IDM):** The intraday market enables continuous trading of electricity for delivery within the same day, typically with gate closure

times ranging from one hour to 15 minutes before delivery. Market participants can adjust their positions based on updated load forecasts and changing conditions. Hour-ahead load forecasts are critical for intraday market operations, as they enable utilities to optimize their trading positions, reduce imbalance exposure, and respond to demand variations that were not anticipated in day-ahead planning. The intraday market serves as a correction mechanism where participants can “true up” their positions as delivery approaches and more accurate demand information becomes available. The accuracy of hour-ahead forecasts directly determines the effectiveness of intraday trading strategies and the magnitude of imbalance costs [11].

3. **Real-Time Balancing Market (RTM):** The real-time or balancing market operates continuously to maintain instantaneous balance between generation and load. System operators procure ancillary services (frequency regulation, spinning reserves, non-spinning reserves) and manage imbalances through this market. Very short-term forecasts (minutes to one hour ahead) are crucial for efficient real-time operations and minimizing costly emergency interventions. Imbalance settlement typically occurs at prices that are less favorable than day-ahead or intraday prices, creating strong economic incentives for accurate forecasting [27].

Figure 2.1 illustrates the relationship between forecasting horizons and electricity market operational timelines.

Market	Trading Horizon	Forecast Need	Key Use
Day-Ahead	D-1 (noon)	24–48 hours	Unit commitment, bidding
Intraday	Same day	1–12 hours	Position adjustment
Real-Time	Continuous	Minutes–1 hour	Balancing, reserves

Figure 2.1 Electricity Market Timeline and Forecasting Requirements

2.2.2. Role of Load Forecasting in Market Operations

Load forecasting serves multiple critical functions in electricity market operations [5, 10]:

Day-Ahead Bidding and Scheduling: The day-ahead market represents the primary venue for electricity trading, where participants commit to buy or sell electricity for each hour of the following day. Utilities must submit their demand forecasts typically by noon on the day before delivery. These forecasts directly inform bidding strategies, generation scheduling, and unit commitment decisions. Under-forecasting leads to expensive purchases in intraday or real-time markets, while over-forecasting results in selling excess power at potentially unfavorable prices. The economic consequences of day-ahead forecast errors can be substantial, particularly for large load-serving entities managing gigawatt-scale portfolios [10].

Intraday Position Adjustment: As the delivery hour approaches, updated forecasts allow utilities to adjust their market positions through intraday trading. This “fine-tuning” mechanism is essential because day-ahead forecasts, made 24–48 hours in advance, inevitably contain errors due to weather forecast uncertainties and unpredictable demand events. Hour-ahead forecasts enable utilities to respond to changing conditions and minimize the gap between contracted and actual consumption. Accurate hour-ahead predictions reduce reliance on more expensive real-time balancing mechanisms and allow for optimal use of intraday trading opportunities.

Imbalance Cost Management: Deviations between forecasted and actual load create imbalances that must be settled in real-time markets, often at unfavorable prices. The financial impact of imbalances is significant—imbalance prices can be several times higher than day-ahead prices during system stress events, or even negative during periods of oversupply. Accurate hour-ahead forecasts are the most critical tool for minimizing imbalance volumes, as they capture the most recent demand patterns and weather conditions. This enables utilities to make timely adjustments through intraday trading, reducing exposure to volatile real-

time prices and imbalance penalties [11].

Grid Security and Reliability: System operators rely on aggregate load forecasts to ensure sufficient generation reserves, manage transmission constraints, and maintain system frequency. Forecast accuracy at the feeder and substation level contributes to improved aggregate forecasts and more reliable grid operations.

Economic Dispatch Optimization: Generation scheduling and economic dispatch algorithms use load forecasts as primary inputs. More accurate forecasts enable tighter operating reserves, reduced fuel costs, and more efficient utilization of generation assets.

2.2.3. Relevance to Nepal's Power Sector

While Nepal's electricity sector currently operates under a centralized utility model through the Nepal Electricity Authority (NEA), the sector is undergoing significant transformation. Cross-border power trading with India through the India-Nepal electricity exchange, increasing private sector participation in generation, and evolving regulatory frameworks suggest movement toward market-oriented operations. In this context, developing robust load forecasting capabilities at the feeder level becomes strategically important for:

- Optimizing power purchase agreements and import scheduling
- Preparing for potential wholesale market development
- Supporting demand-side management programs
- Enabling efficient integration of distributed energy resources
- Reducing technical and commercial losses through better operational planning

The hour-ahead and day-ahead forecasting capabilities developed in this thesis are directly aligned with these market-relevant timescales, ensuring that the fore-

casting framework can support both current operational needs and future market participation requirements.

2.3. Theoretical Background of Forecasting Models

This section elucidates the mathematical foundations and operational principles underlying the computational models deployed throughout this investigation. Comprehending these theoretical constructs proves indispensable for meaningful interpretation of model behavior and forecasting outcomes within the distribution system context.

2.3.1. Machine Learning Models

2.3.1.1. Linear Regression

Linear Regression constitutes the most elementary predictive framework, postulating a strictly linear mapping between predictor variables and the response quantity. Despite the inherent nonlinearity characterizing electrical consumption phenomena, this model furnishes a baseline reference against which more sophisticated approaches can be quantitatively benchmarked.

Mathematical Formulation:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \quad (2.1)$$

where β_0 is the intercept (bias term), β_i are the coefficients (weights) learned via ordinary least squares minimization, and x_i are the input features.

2.3.1.2. Ridge Regression

Ridge Regression [28] adds L2 regularization to the linear model to reduce overfitting and stabilize coefficient estimates. It is more robust than standard Linear Regression when dealing with many correlated features, which is the case in this study.

Mathematical Formulation:

$$\min_{\beta} (\|y - X\beta\|^2 + \alpha\|\beta\|^2) \quad (2.2)$$

where α controls the strength of L2 regularization, $\|y - X\beta\|^2$ is the residual sum of squares, and $\|\beta\|^2$ is the L2 norm of the coefficient vector.

2.3.1.3. Support Vector Regression (SVR)

SVR [29] models nonlinear relationships by mapping features into a high-dimensional space using kernel functions. It works well for complex regression problems with moderate dataset sizes.

Mathematical Formulation:

SVR finds a function $f(x)$ by solving the following optimization problem:

$$\min_{w,b,\xi,\xi^*} \left(\frac{1}{2}\|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) \right) \quad (2.3)$$

subject to the constraints:

$$y_i - (w \cdot x_i + b) \leq \varepsilon + \xi_i$$

$$(w \cdot x_i + b) - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

where w is the weight vector, b is the bias term, C is the regularization parameter controlling the trade-off between flatness and tolerance of deviations, ε defines the epsilon-insensitive tube, and ξ_i and ξ_i^* are slack variables for points outside the tube.

2.3.1.4. Random Forest Regressor

Random Forest [30] is an ensemble method consisting of multiple decision trees. Each tree is trained on a random subset of features and samples (bootstrap aggregating). It is robust, stable, and handles nonlinearity effectively.

Mathematical Formulation:

The Random Forest prediction is the average of all individual tree predictions:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^T f_t(x) \quad (2.4)$$

where T is the total number of trees in the forest and $f_t(x)$ is the prediction of the t -th decision tree.

2.3.1.5. Gradient Boosting Regressor

Gradient Boosting [31] builds trees sequentially, with each new tree correcting the errors (residuals) of the previous ensemble. It is particularly effective for structured tabular data like load forecasting.

Mathematical Formulation:

At each boosting iteration m , the model is updated as:

$$F_m(x) = F_{m-1}(x) + \nu \cdot h_m(x) \quad (2.5)$$

where $F_{m-1}(x)$ is the ensemble prediction from the previous iteration, $h_m(x)$ is the new tree fitted to the negative gradient (pseudo-residuals), and ν is the learning rate (shrinkage factor) that controls the contribution of each tree.

2.3.1.6. XGBoost Regressor

XGBoost (Extreme Gradient Boosting) [32] is an optimized and regularized implementation of gradient boosting designed for efficiency, scalability, and high accuracy. It was one of the best-performing ML models in this study.

Mathematical Formulation:

XGBoost minimizes the following regularized objective function:

$$\mathcal{L} = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k) \quad (2.6)$$

where $l(y_i, \hat{y}_i)$ is the loss function measuring the difference between actual and predicted values, and $\Omega(f_k)$ is the regularization term for the k -th tree, defined as:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (2.7)$$

where T is the number of leaves in the tree, w_j is the weight (score) of the j -th leaf, γ controls the minimum loss reduction required to make a split, and λ is the L2 regularization term on leaf weights.

2.3.2. Deep Learning Models

To adequately represent the nonlinear dynamics, temporal evolution, and sequential structure inherent in feeder consumption data, two recurrent neural network architectures were constructed and evaluated: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). These networks received training on sliding temporal windows comprising consecutive hourly observations, enabling the architectures to assimilate both proximate and extended temporal dependencies embedded within the dataset. Network optimization employed the Adam algorithm [33], with early termination protocols preventing overtraining and sequence-structured inputs.

2.3.2.1. Long Short-Term Memory (LSTM)

LSTM networks [34] are designed to maintain contextual memory over long sequences through a gating mechanism that controls information flow. This makes them naturally suited for load forecasting, where consumption patterns depend on previous hours. The LSTM architecture addresses the vanishing gradient problem that affects standard RNNs.

Mathematical Formulation:

At each time step t , the LSTM computes the following:

Forget gate (determines what information to discard from cell state):

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.8)$$

Input gate (determines what new information to store):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.9)$$

Candidate cell state (creates new candidate values):

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (2.10)$$

Cell state update (combines old and new information):

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (2.11)$$

Output gate (determines what to output):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.12)$$

Hidden state (final output at time step t):

$$h_t = o_t \odot \tanh(c_t) \quad (2.13)$$

where σ is the sigmoid activation function, \tanh is the hyperbolic tangent activation function, \odot denotes element-wise (Hadamard) product, $[h_{t-1}, x_t]$ represents concatenation of the previous hidden state and current input, W_f , W_i , W_c , and W_o are weight matrices for each gate, b_f , b_i , b_c , and b_o are bias vectors for each gate, c_t is the cell state that carries long-term memory, and h_t is the hidden state output.

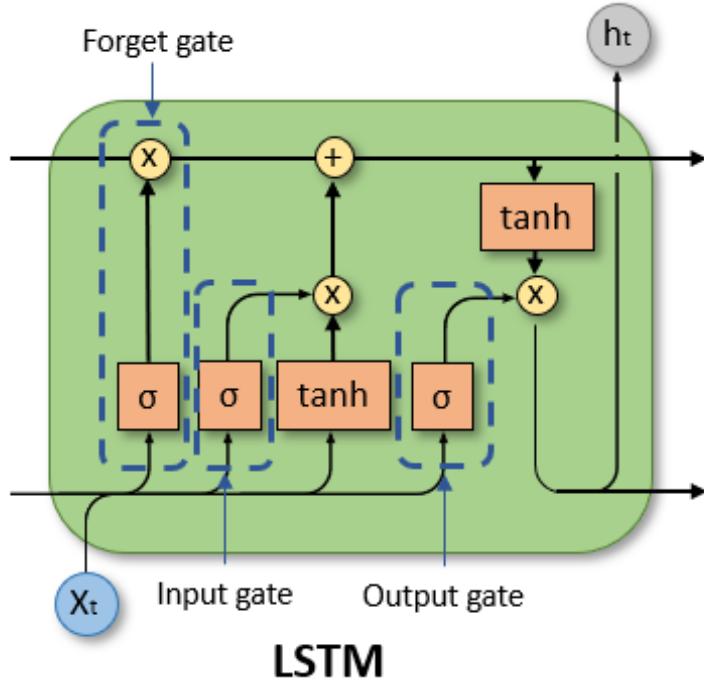


Figure 2.2 Architecture LSTM

2.3.2.2. Gated Recurrent Unit (GRU)

GRU [35] is a streamlined version of LSTM with fewer parameters, combining the forget and input gates into a single update gate and merging the cell state with the hidden state. It often trains faster while achieving comparable performance to LSTM.

Mathematical Formulation:

At each time step t , the GRU computes the following:

Update gate (controls how much past information to keep):

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \quad (2.14)$$

Reset gate (determines how much past information to forget):

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \quad (2.15)$$

Candidate hidden state (computes new candidate activation):

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h) \quad (2.16)$$

Final hidden state (interpolates between previous and candidate state):

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (2.17)$$

where σ is the sigmoid activation function, \tanh is the hyperbolic tangent activation function, \odot denotes element-wise (Hadamard) product, $[h_{t-1}, x_t]$ represents concatenation of the previous hidden state and current input, W_z , W_r , and W_h are weight matrices for the update gate, reset gate, and candidate state, b_z , b_r , and b_h are bias vectors, z_t controls the balance between old and new information, and r_t controls how much of the previous state influences the candidate.

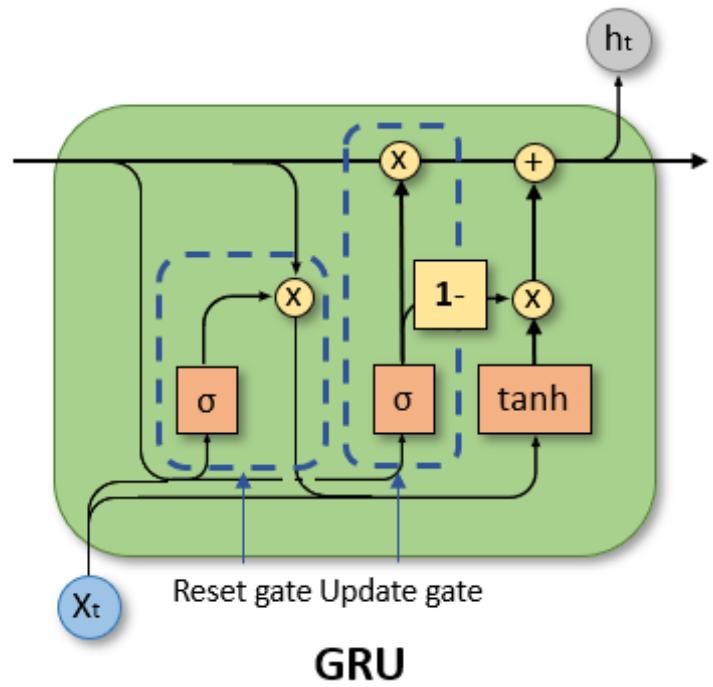


Figure 2.3 Architecture GRU

CHAPTER THREE: RESEARCH METHODOLOGY

This chapter delineates the comprehensive research framework implemented to accomplish the stated thesis objectives. The presentation encompasses experimental design rationale, data procurement procedures, preprocessing protocols, and predictive modeling strategies deployed for feeder-level demand forecasting. The methodological structure ensures rigorous analytical progression, with each procedural component maintaining explicit linkage to the defined research goals. The forecasting framework is specifically designed to support electricity market operational timelines, with hourly predictions enabling both day-ahead scheduling and hour-ahead intraday market operations. A schematic diagram encapsulates the overarching workflow, supplemented by detailed exposition of the chosen machine learning and deep learning algorithms alongside the quantitative assessment techniques applied throughout this investigation.

3.1. Overall Workflow

The predictive modeling pipeline employed in this thesis adheres to a methodical, stage-wise progression, designed to produce forecasts aligned with hour-ahead electricity market operational requirements. Raw hourly demand recordings from the Baneshwor Feeder are aggregated with concurrent meteorological observations (temperature, humidity, precipitation) and temporal markers (weekday/weekend designations, holiday flags) to constitute the comprehensive input feature space. The hourly resolution of the data directly corresponds to the standard settlement periods used in intraday electricity markets, where hour-ahead position adjustments represent the most dynamic and frequently utilized trading mechanism [3]. These unprocessed records initially undergo extensive quality assurance procedures addressing missing entries, anomalous values, and formatting discrepancies through systematic cleaning, imputation algorithms, outlier

remediation, timestamp normalization, and derivation of temporal and cyclical feature representations—thereby generating analysis-ready datasets. Subsequent exploratory analysis interrogates consumption trends, periodic patterns, hourly variation structures, and weather-demand correlations to identify salient predictive features and inform variable selection decisions. Following these preparatory stages, both traditional ML algorithms and deep neural architectures are instantiated, with input features undergoing standardization and organization as tabular matrices for ML implementations while deep architectures receive sequentially-structured inputs. Model calibration proceeds on designated training partitions with validation through walk-forward protocols or holdout assessment, while hyperparameter optimization employs GridSearchCV for ML algorithms and iterative refinement strategies for neural networks to enhance generalization while mitigating overfitting tendencies. Ultimately, all candidate models undergo comparative evaluation using RMSE, MAE, MAPE, and R^2 metrics—standard measures used in electricity market forecasting literature [5]—with predictive accuracy, stability characteristics, and computational efficiency jointly analyzed to recommend optimal forecasting solutions for Baneshwor Feeder hour-ahead operational deployment. This workflow thereby establishes a complete analytical pipeline spanning initial data procurement through final model recommendation, accommodating both conventional and neural network methodologies. Figure 3.1 illustrates the complete methodological structure.

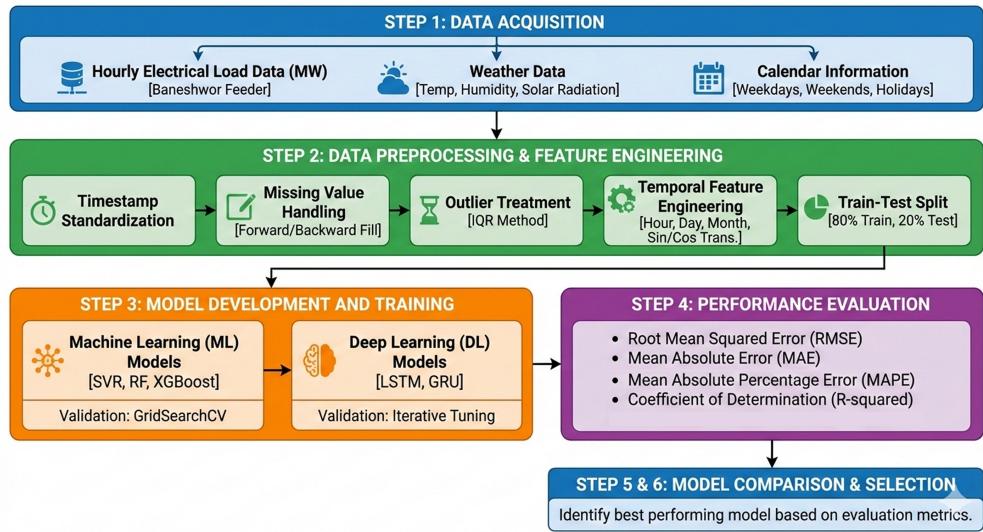


Figure 3.1 Methodology Block Diagram

3.2. Data Acquisition

The initial methodological phase involves systematic data procurement. Archived hourly consumption records from the Baneshwor Feeder constitute the primary dataset. Supplementary meteorological observations—encompassing ambient temperature, atmospheric moisture content, and precipitation measurements—originate from the Department of Hydrology and Meteorology, Nepal. Additionally, weekday and weekend classifications derive from officially published governmental calendrical sources.

The investigative framework relies upon dual primary data streams:

1. Hourly electrical consumption measurements for the Baneshwor Feeder
2. Concurrent hourly meteorological recordings (temperature, humidity, incident solar radiation)

Given that both datasets arrived as unprocessed Excel workbooks exhibiting irregular structural formats, inconsistent temporal indexing, incomplete entries, and multiple worksheets per temporal unit, a comprehensive multi-stage acquisition and restructuring protocol was necessitated. The provenance of both datasets is documented subsequently.

3.2.1. Data Sources

- a) **Electrical Load Data:** The electrical load data used in this study was obtained from the Baneshwor Substation, which operates under the Nepal Electricity Authority (NEA). Hourly feeder load readings were collected from archived operational log sheets maintained by the substation for the years 2079 to 2082 BS. These records provided raw POWER (MW) measurements for the Baneshwor Feeder, along with associated timestamp information. Since the data originated from manually recorded and distributed Excel files, several preprocessing steps—such as header correction, timestamp standardization, and quality checks—were required before the dataset could be used for modeling. This substation-provided dataset forms the core of the forecasting analysis, representing real operational feeder behavior across multiple years.
- b) **Weather Data:** Weather data was sourced from Nepal’s Department of Hydrology and Meteorology (DHM), the official governmental agency responsible for climate and atmospheric measurements. The dataset included hourly records of air temperature, relative humidity, and global solar radiation for the corresponding study period. These variables were essential for capturing the environmental conditions influencing electricity consumption patterns. The DHM dataset required timestamp alignment, interpolation for missing values, and smoothing of extreme readings to ensure compatibility with the load dataset. Once cleaned and synchronized, the weather data served as an important set of exogenous features for both machine learning and deep learning models.

Because the raw files came in varying formats—different months, unpredictable sheet names, Bikram Sambat (BS) dates, mixed day formats, half-hour readings, inconsistent header rows, and multiple sheets per month—a custom data acquisition pipeline was required.

3.2.2. Load Data Acquisition and Structuring Process

The original Excel files provided by the Nepal Electricity Authority (NEA) were highly heterogeneous in structure. Each month consisted of multiple workbooks, with each workbook containing several sheets. In many cases, sheets included mixed headers, irrelevant rows, inconsistent timestamp formats, and non-uniform naming conventions. To address these issues and convert the raw data into a single unified structure suitable for analysis, a multi-stage data processing pipeline was implemented.

In the first stage, a month-wise sheet extraction process was carried out. The script automatically scanned all monthly datasets and identified Excel sheets whose names contained “11KV” or its variations. The extracted sheets were then aligned with their corresponding time periods to ensure correct temporal ordering. Only rows with timestamps recorded at exact hourly intervals (HH:00) were retained, while half-hour readings such as 7:30 were intentionally excluded to maintain uniform hourly resolution. The valid hourly records from each sheet were compiled to produce clean month-wise datasets. At this stage, although the data were organized chronologically, the timestamps were still recorded in the Nepali calendar (BS) and exhibited format inconsistencies.

The second stage focused on date conversion, hour normalization, and daily data structuring. The BS date embedded within each sheet name was extracted and converted into the Gregorian (AD) calendar using Nepali date conversion libraries. Each sheet was read without assuming a fixed header position, allowing the script to dynamically identify the Time column and the corresponding POWER (MW) values. Every day was standardized to contain exactly twenty-four hourly records by indexing hours from 1 to 24, and any missing hours were filled using linear interpolation. Clean and consistent timestamps were then generated in the standard hourly format, ensuring temporal continuity across the dataset. This process resulted in one clean and complete daily record for each calendar day.

In the final stage, all structured monthly and yearly datasets were merged into

a single consolidated file. The script systematically extracted the valid Time and POWER (MW) columns, removed any remaining header fragments, and concatenated the data in chronological order. This process produced a fully unified dataset containing continuous hourly POWER (MW) measurements for the entire study period, which served as the foundation for subsequent data analysis and load forecasting model development.

3.2.3. Weather Data Acquisition and Structuring

Weather data was also provided in raw format with mixed timestamps. Two scripts were developed to clean and align it with the load data.

- a) **Extracting and Cleaning Raw Weather File:** The script located the correct columns for time, temperature, humidity, and solar radiation, then removed any unusable rows. All timestamps were parsed into a consistent datetime format, after which the weather data was filtered to match the exact date range of the load dataset. The timestamps were then formatted as YYYY-MM-DD HH:MM. This stage produced a clean hourly weather dataset.
- b) **Structuring Weather Timestamp Alignment:** Timestamps were shifted so that values such as “HH:45” were aligned to the next hour at “HH+1:00,” and all “24:00” rollover cases were handled correctly. Missing or zero weather values were replaced using nearest-neighbor averages, while NaN solar radiation entries were set to zero. These steps produced the final clean weather file and ensured that all weather variables followed the exact hourly structure required for forecasting.

3.2.4. Final Merging of Load and Weather Data

In the final stage, load and weather datasets were merged into a single unified file. All load timestamps were carefully parsed, including proper handling of the special 24:00 time format, while weather timestamps were standardized to a consistent

format. Weather observations were then precisely aligned with their corresponding load timestamps, and any missing values in weather variables were filled using linear interpolation. The resulting dataset contained synchronized records of time, electrical load (MW), air temperature, global solar radiation, and relative humidity, and served as the primary input for all machine learning and deep learning models used in this thesis.

3.3. Data Preprocessing

Once the load and weather datasets were fully acquired and merged into a single hourly dataset, several preprocessing steps were performed to prepare the data for machine learning and deep learning models. The merged dataset initially contained timestamps in multiple formats, including irregular representations such as “24:00.” All timestamps were therefore parsed and standardized using a custom parsing routine, where “24:00” was shifted to 00:00 of the following day, and the final format was normalized to a consistent hourly representation. Missing electrical load (MW) values caused by incomplete feeder logs and invalid records were handled using a forward-fill followed by backward-fill strategy to preserve temporal continuity without introducing artificial variations. Similarly, missing weather values were treated using linear interpolation, with nighttime solar radiation values set to zero and extreme humidity or temperature readings smoothed using neighboring observations. These steps ensured a clean, continuous, and time-aligned dataset.

After cleaning, temporal feature engineering was applied to capture the inherent daily, weekly, and seasonal patterns in electricity consumption. From each timestamp, multiple time-based features were extracted, including hour, day, month, day of week, week of year, and a weekend indicator. To better represent the cyclical nature of time, sine and cosine transformations were applied to hour, month, and day-of-week values. These cyclic encodings allow machine learning and deep learning models to learn smooth periodic relationships, such as the transition from late night hours to early morning, rather than treating time variables as discontin-

uous linear values. The resulting feature set combined both weather variables and engineered temporal components, forming a comprehensive input representation for modeling.

To understand feature relevance and interdependencies, a correlation analysis was conducted on all numerical variables. The correlation matrix, shown in Figure 3.2, indicates that hour of day has the strongest relationship with electrical load, while global solar radiation shows a moderate positive correlation. Temperature and humidity exhibit weaker but meaningful correlations, and calendar-related variables contribute subtle seasonal trends. Based on this analysis and domain knowledge, all engineered features were retained. After preprocessing, the final dataset contained no missing values, no irregular timestamps, and no half-hour entries, resulting in a fully standardized and reliable hourly time-series dataset used for both machine learning and deep learning model development.



Figure 3.2 Feature Correlation Matrix

3.4. Data Input Structure

This section describes the structure and format of the input data used for training the machine learning and deep learning models. Understanding the data input structure is essential for reproducibility and for applying similar methodologies to other forecasting problems.

3.4.1. Raw Data Format

The final merged dataset consists of hourly records with the following structure:

Table 3.1 Raw Input Data Format

Column	Data Type	Unit	Description
Time	Datetime	YYYY-MM-DD HH:MM	Hourly timestamp
MW	Float	Megawatt (MW)	Electrical load (target variable)
Air Temperature	Float	°C	Ambient temperature
Global Solar Radiation	Float	W/m ²	Solar irradiance
Relative Humidity	Float	%	Atmospheric humidity

Table 3.2 shows a sample of the cleaned dataset used in this study:

Table 3.2 Sample Data Records from Cleaned Dataset

Time	MW	Air Temp (°C)	Solar Rad (W/m ²)	RH (%)
2022-10-19 01:00	0.8	14.5	0.0	88.8
2022-10-19 02:00	0.8	14.4	0.0	87.9
2022-10-19 06:00	1.2	12.1	0.0	100.0
2022-10-19 12:00	1.8	22.0	822.0	46.5
2022-10-19 19:00	2.8	17.9	0.0	76.7

3.4.2. Feature Engineering and Input Vector

After preprocessing and feature engineering, each input sample contains the following features that are fed into the model nodes:

Table 3.3 Engineered Feature Set for Model Input

No.	Feature	Type	Description
1	Hour	Integer (0–23)	Hour of day
2	Day	Integer (1–31)	Day of month
3	Month	Integer (1–12)	Month of year
4	Day_of_Week	Integer (0–6)	Day of week (Mon=0)
5	Week_of_Year	Integer (1–52)	Week number
6	Is_Weekend	Binary (0/1)	Weekend indicator
7	Hour_Sin	Float (-1 to 1)	$\sin(2\pi \cdot \text{Hour}/24)$
8	Hour_Cos	Float (-1 to 1)	$\cos(2\pi \cdot \text{Hour}/24)$
9	Month_Sin	Float (-1 to 1)	$\sin(2\pi \cdot \text{Month}/12)$
10	Month_Cos	Float (-1 to 1)	$\cos(2\pi \cdot \text{Month}/12)$
11	DoW_Sin	Float (-1 to 1)	$\sin(2\pi \cdot \text{DoW}/7)$
12	DoW_Cos	Float (-1 to 1)	$\cos(2\pi \cdot \text{DoW}/7)$
13	Air_Temperature	Float	Scaled temperature
14	Global_Solar_Radiation	Float	Scaled solar radiation
15	Relative_Humidity	Float	Scaled humidity

3.4.3. Input Structure for Machine Learning Models

For machine learning models (Linear Regression, Ridge, SVR, Random Forest, Gradient Boosting, XGBoost), the input is structured as a 2D feature matrix:

$$X_{ML} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix} \quad (3.1)$$

where m is the number of samples and n is the number of features. The target vector is:

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \quad (3.2)$$

where $y^{(i)}$ represents the load value (MW) at the next hour that the model predicts.

3.4.4. Input Structure for Deep Learning Models (LSTM/GRU)

For recurrent neural networks (LSTM and GRU), the input is structured as a 3D tensor to capture temporal sequences:

$$X_{DL} \in \mathbb{R}^{m \times T \times n} \quad (3.3)$$

where m is the number of samples, T is the sequence length (lookback window, typically 24 hours), and n is the number of features per timestep.

Figure 3.3 illustrates how the sliding window approach creates input sequences for the LSTM/GRU models:

	Input Sequence (T=24 hours)					Target
Sample	$t - 24$	$t - 23$	\dots	$t - 1$	t	
1	\mathbf{x}_1	\mathbf{x}_2	\dots	\mathbf{x}_{24}	y_{25}	
2	\mathbf{x}_2	\mathbf{x}_3	\dots	\mathbf{x}_{25}	y_{26}	
3	\mathbf{x}_3	\mathbf{x}_4	\dots	\mathbf{x}_{26}	y_{27}	
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots	\vdots

Figure 3.3 Sliding Window Sequence Construction for LSTM/GRU Input

Each \mathbf{x}_t represents the feature vector at timestep t :

$$\mathbf{x}_t = [\text{Hour}_t, \text{Hour_Sin}_t, \text{Hour_Cos}_t, \dots, \text{Lag}_1, \text{Lag}_3, \dots, \text{Lag}_{48}] \quad (3.4)$$

3.4.5. Lag Feature Configuration for Deep Learning

To enhance temporal pattern recognition, lag features are added to the input vector. Three configurations were tested:

Table 3.4 Lag Feature Configurations for Deep Learning Models

Configuration	Lag Hours	Purpose
Short	[1, 3, 6]	Recent hour patterns
Medium	[1, 3, 6, 12, 24]	Daily patterns
Long (Best)	[1, 3, 6, 12, 24, 48]	Multi-day dependencies

For example, with the long lag configuration, the lag features for predicting load at time t include:

- Lag_1 : Load at $t - 1$ (1 hour ago)
- Lag_3 : Load at $t - 3$ (3 hours ago)
- Lag_6 : Load at $t - 6$ (6 hours ago)
- Lag_{12} : Load at $t - 12$ (12 hours ago)
- Lag_{24} : Load at $t - 24$ (same hour yesterday)
- Lag_{48} : Load at $t - 48$ (same hour two days ago)

3.4.6. Complete Input Pipeline Summary

Figure 3.4 summarizes the complete data input pipeline from raw data to model nodes:

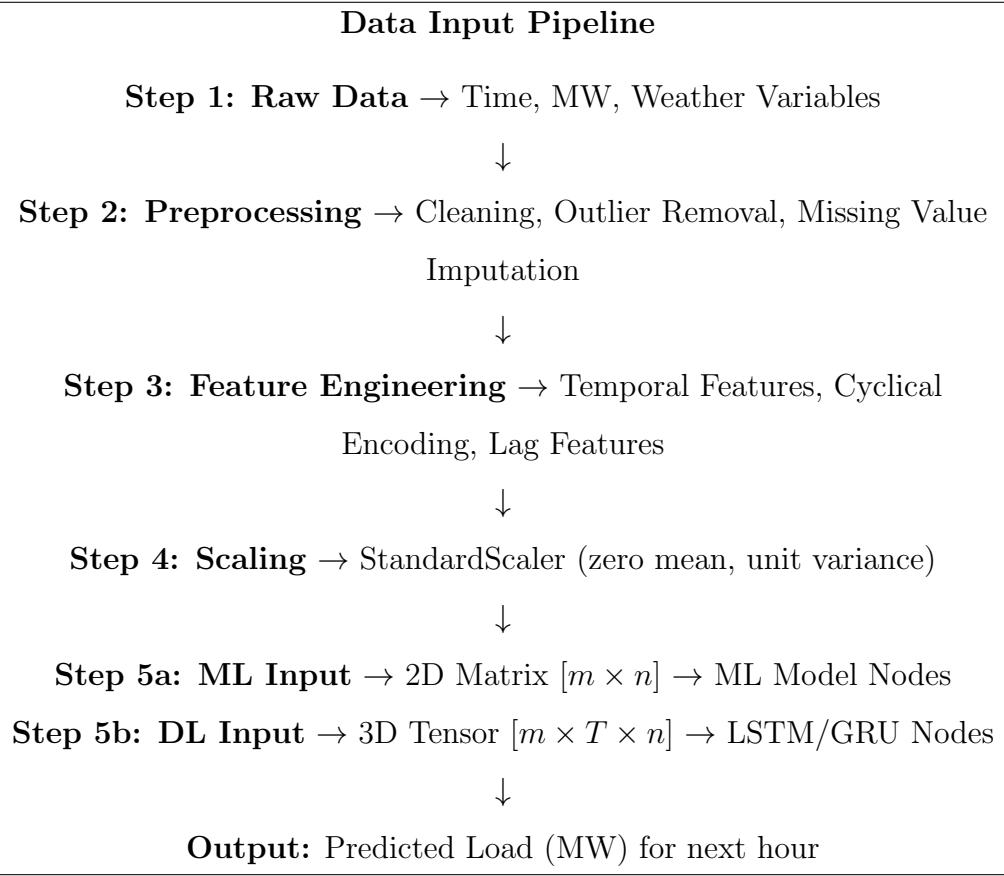


Figure 3.4 Complete Data Input Pipeline from Raw Data to Model Nodes

This structured input representation ensures that both machine learning and deep learning models receive appropriately formatted data that captures temporal patterns, weather influences, and historical load dependencies necessary for accurate hour-ahead load forecasting.

3.5. Model Development

Based on the theoretical foundations presented in Chapter 2, this study implemented multiple forecasting models to predict short-term electrical load for the Baneshwor Feeder. For machine learning, six models were developed: Linear Regression, Ridge Regression, Support Vector Regression (SVR), Random Forest Regressor, Gradient Boosting Regressor, and XGBoost Regressor. For deep learning, two recurrent architectures were implemented: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). All models were trained on the cleaned and

feature-engineered dataset described in the earlier sections, with each model receiving the same input feature set to ensure fair comparison. The machine learning pipeline involved standard scaling of the numerical features, an 80–20 train–test split, and hyperparameter tuning performed using GridSearchCV, while the deep learning models were trained on sliding windows of historical hourly sequences using the Adam optimizer with early stopping to prevent overfitting. Model performance was evaluated using RMSE, MAE, MAPE, and R².

3.6. Model Training and Validation

This stage focuses on how both machine learning (ML) and deep learning (DL) models were trained, tuned, validated, and prepared for final performance comparison. Since ML and DL models require different handling, the training process is presented in two separate parts.

3.6.1. Training of Machine Learning Models

The machine learning models trained in this study include:

1. Support Vector Regression (SVR)
2. Random Forest Regressor (RF)
3. Gradient Boosting Regressor (GBR)
4. Extreme Gradient Boosting (XGBoost)
5. Linear Regression and Ridge Regression (baseline models)

All models used the same final preprocessed dataset described earlier, containing weather features, temporal encodings, and cleaned load values.

3.6.1.1. Input Preparation

For ML models, the dataset was used in a tabular format:

1. **Features (X):** Time features (hour, month, weekday), cyclical encodings, weather variables, and lag features if applied.
2. **Target (y):** Load at the next hour.

Since ML models do not operate on sequences, no sliding window was required.

3.6.1.2. Data Splitting

The dataset was split into 80 percent for training and 20 percent for testing. Shuffling was applied during the split to prevent temporal clustering and to ensure that the machine learning models were exposed to a diverse mix of seasonal and temporal patterns during training.

3.6.1.3. Feature Scaling

Some models required normalized inputs, so StandardScaler was applied to Linear Regression, Ridge, and SVR. Tree-based models such as Random Forest, Gradient Boosting, and XGBoost did not require any scaling.

3.6.1.4. Training Procedure

Each model was trained using its corresponding optimization approach:

- Least squares optimization for Linear Regression and Ridge
- Kernel-based optimization for SVR
- Ensemble tree learning for Random Forest and Gradient Boosting
- Gradient-boosted tree optimization for XGBoost

The training process involved fitting the models on the training set, generating predictions on the test set, and evaluating performance using RMSE, MAE, MAPE, and R².

3.6.1.5. Hyperparameter Tuning

All machine learning models were fine-tuned using GridSearchCV, which tested different combinations of hyperparameters:

- **SVR:** C, epsilon, and gamma
- **Random Forest and Gradient Boosting:** n_estimators, max_depth, and min_samples_split
- **XGBoost:** learning_rate, max_depth, subsample, and colsample_bytree
- **Ridge:** alpha

The best configurations were selected based on the lowest validation error.

3.6.2. Training of Deep Learning Models

Two deep learning models were implemented:

1. Long Short-Term Memory (LSTM)
2. Gated Recurrent Unit (GRU)

Since deep learning models learn from sequences rather than static features, the training process follows a different pipeline.

3.6.2.1. Sequence Construction

A sliding window method was used in which the model received the past N hours as input and predicted the load for the next hour. A typical window size of 24 hours was used, although this value can be adjusted in the implementation.

3.6.2.2. Train–Validation–Test Split

Deep learning models require sequential integrity, so the dataset was split chronologically:

- 70 percent for training
- 15 percent for validation
- 15 percent for testing

No shuffling was applied, ensuring that the model learned from the natural temporal progression of the data.

3.6.2.3. Lag Feature Configurations

To capture temporal dependencies at multiple scales, three different lag configurations were evaluated:

- **Short:** Lags at [1, 3, 6] hours
- **Medium:** Lags at [1, 3, 6, 12, 24] hours
- **Long:** Lags at [1, 3, 6, 12, 24, 48] hours

The extended lag configuration proved most effective for the recurrent models by providing explicit historical context at various temporal resolutions.

3.6.2.4. Data Augmentation

To improve model generalization and increase the effective training dataset size, data augmentation techniques were applied:

- **Noise injection:** Small random noise added to training samples
- **Jittering:** Slight perturbations to feature values
- **Scaling:** Random scaling of feature magnitudes

These augmentation methods doubled the effective training size (2x augmentation factor), helping to reduce overfitting and improve model robustness.

3.6.2.5. Feature Scaling

Two scaling approaches were evaluated:

- **MinMax Scaler:** Scales features to [0, 1] range
- **Standard Scaler:** Standardizes features to zero mean and unit variance

The standard scaler combined with the long lag configuration yielded the best results for the recurrent models.

3.6.2.6. Model Training Configuration

All deep learning models were trained with the following configuration:

- **Optimizer:** Adam
- **Loss Function:** Mean Squared Error (MSE)
- **Batch Size:** Typically 32 or 64
- **Epochs:** Training continued for multiple epochs until early stopping criteria were met
- **Weight Initialization:** Xavier/Glorot initialization (TensorFlow defaults)

3.6.2.7. Regularization and Stability

To avoid overfitting, several regularization techniques were applied:

- Dropout layers were included in the LSTM and GRU models
- Batch normalization was applied where appropriate
- Early stopping was used to monitor validation loss, and training automatically stopped when the validation loss stopped improving for several consecutive epochs

3.6.2.8. Model-Specific Training Notes

- **LSTM:** Processed sequential inputs with a 24-hour lookback window. Achieved strong performance with RMSE of 0.314 and R² of 0.857, demonstrating effective capability in capturing temporal patterns in the load data.
- **GRU:** Provided a computationally lighter alternative to LSTM with slightly better performance, achieving RMSE of 0.289 and R² of 0.879. The GRU architecture proved most effective among the deep learning models for this forecasting task.

3.6.3. Validation Approach

A consistent evaluation strategy was applied across all models:

Machine Learning Models:

- Validated using GridSearchCV with five-fold cross-validation
- Best hyperparameters were chosen based on the minimum validation loss

Deep Learning Models:

- Validated using a 15 percent validation split (chronologically ordered)
- Early stopping was used to prevent overfitting, with patience of approximately 10 epochs
- Best-performing model weights were preserved through checkpointing
- Multiple lag configurations and scaling methods were systematically compared

3.7. Performance Evaluation

To facilitate equitable comparison across machine learning and deep neural network implementations, a standardized battery of statistical accuracy measures

was employed throughout. These metrics quantify the magnitude of discrepancies between model-generated predictions and empirically observed consumption values. This investigation adopted four conventionally utilized regression assessment criteria:

1. Mean Absolute Error (MAE)
2. Root Mean Squared Error (RMSE)
3. Mean Absolute Percentage Error (MAPE)
4. Coefficient of Determination (R^2 Score)

These statistical measures collectively characterize predictive accuracy, algorithmic robustness, and overall forecasting efficacy across the candidate models.

3.7.1. Mean Absolute Error (MAE)

MAE quantifies the average magnitude of prediction deviations from observed values, disregarding error directionality. This metric offers straightforward interpretability and practical utility.

Formula:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (3.5)$$

where n denotes the sample count, y_i represents the measured consumption value, and \hat{y}_i indicates the corresponding model prediction.

Interpretation: Diminished MAE values signify consistently accurate predictions approaching actual consumption levels. This measure demonstrates resilience against distortion from sporadic large-magnitude errors.

3.7.2. Root Mean Squared Error (RMSE)

RMSE constitutes among the most prevalent metrics in demand forecasting applications, preserving measurement units identical to the original consumption values (MW).

Formula:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (3.6)$$

where n represents the observation count, y_i denotes actual consumption, and \hat{y}_i signifies the predicted value.

Interpretation: Reduced RMSE values indicate superior overall predictive accuracy. The quadratic error term imposes heavier penalties on substantial deviations, rendering RMSE a more stringent criterion than MAE for model assessment.

3.7.3. Mean Absolute Percentage Error (MAPE)

MAPE articulates prediction errors as proportional deviations from actual observations, facilitating performance comparisons across disparate feeders or demand scales.

Formula:

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (3.7)$$

where n represents the sample size, y_i indicates measured consumption, and \hat{y}_i denotes the model forecast.

Interpretation: This metric expresses average percentage deviation between predictions and observations, with lower values indicating enhanced performance. The measure becomes mathematically undefined when actual consumption equals zero, though this scenario never materialized given the feeder's continuous operation.

3.7.4. Coefficient of Determination (R^2 Score)

R^2 quantifies the proportion of target variable variance that the predictive model successfully captures and explains.

Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (3.8)$$

where y_i represents measured consumption, \hat{y}_i denotes the predicted quantity, and \bar{y} indicates the arithmetic mean of observations.

Interpretation: An R^2 coefficient of unity signifies flawless prediction capability, whereas zero indicates performance equivalent to naive mean-based forecasting. Elevated R^2 values consequently reflect superior model efficacy. Negative coefficients remain theoretically possible when model predictions underperform relative to simple averaging.

CHAPTER FOUR: RESULTS AND DISCUSSION

This chapter presents the performance of all machine learning and deep learning models trained for short-term load forecasting of the Baneshwor Feeder. All models were evaluated using the same performance metrics (MAE, RMSE, MAPE, R^2), ensuring a fair comparison.

4.1. Exploratory Data Analysis

Preliminary statistical examination was undertaken to characterize the distributional properties of consumption and meteorological variables, discern latent temporal structures, and quantify inter-feature associations prior to model construction. The consolidated dataset encompassed hourly timestamps covering the complete observation period, feeder demand measurements expressed in megawatts, and primary meteorological variables comprising ambient temperature, incident solar irradiance, and atmospheric relative humidity. Additionally, an extensive suite of derived temporal descriptors—including hour index, calendar day, month indicator, weekday designation, and their corresponding sinusoidal transformations—augmented the feature space. Verification procedures confirmed successful missing value remediation, consistent hourly temporal resolution without intermediate gaps, cleaned consumption readings following outlier treatment, and precise chronological alignment between demand and weather observations.

To understand how the load varies over time, the hourly POWER (MW) values were resampled into daily averages and visualized across the entire study period. The resulting trend showed clear daily, weekly, and seasonal fluctuations in the feeder's behavior. Winter months displayed slightly lower solar radiation levels along with moderately higher load during certain intervals. Occasional dips in the curve aligned with known outages or special events. Solar radiation exhibited a strong daytime pattern, reinforcing its moderate correlation with load. Overall,

this broad visualization confirmed that the Baneshwor Feeder operates as a typical mixed-load distribution feeder with pronounced daily cycles and noticeable seasonal influences.

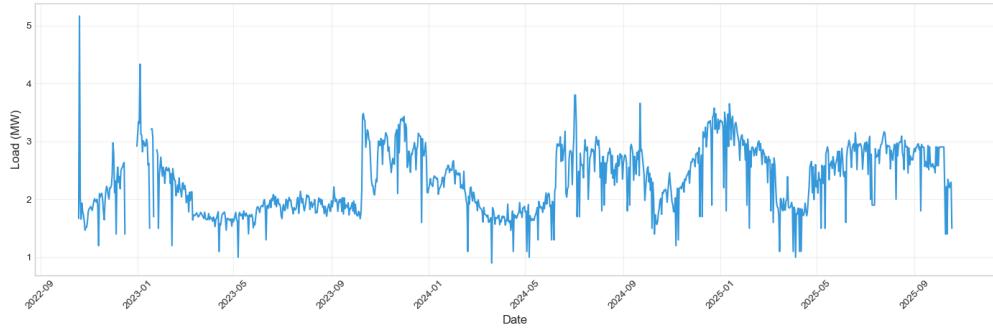


Figure 4.1 Daily Average Electricity Load Over Time

The hourly load values were averaged across the full dataset to understand the feeder's daily consumption pattern. The analysis showed that the minimum load typically occurs around 3:00 AM, which reflects low residential and commercial activity during that time. Load levels begin to rise through the morning and reach a peak at around 19:00, with the average peak load reaching approximately 3.16 MW. This aligns with evening lighting needs and heightened residential usage. A boxplot comparing load against hour of day further illustrated that evening hours exhibit higher variance, while midnight to early-morning hours display more stable and lower demand. These observations confirm that the hour of the day is one of the strongest predictors of load in this feeder.

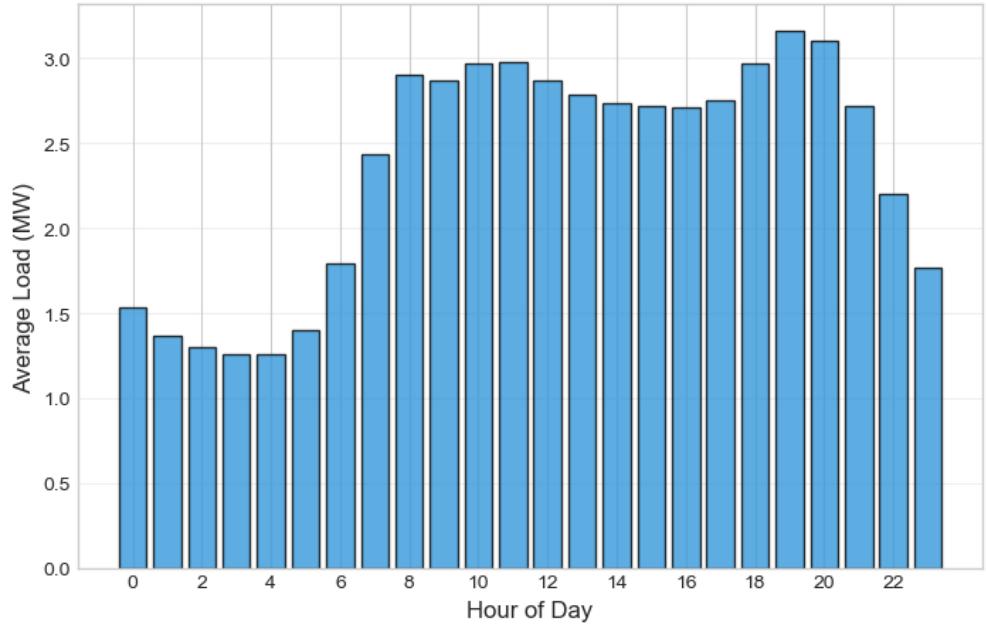


Figure 4.2 Load Distribution by Hour

Monthly averages revealed that warmer months experience higher temperatures, although the corresponding load behavior varies across the year. Seasonal patterns are present but not as dominant as the daily cycles observed in the feeder. Consumption typically increases during festival seasons when household activity rises. These seasonal shifts are effectively captured through the Month feature and its corresponding cyclical encodings.

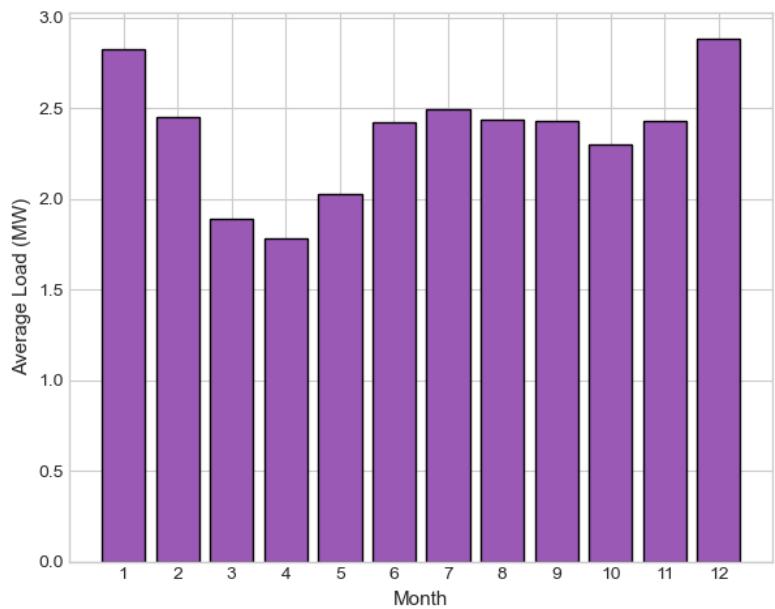


Figure 4.3 Average Load by Month

All weather variables were examined individually to understand their behavior and potential influence on load. Air temperature ranged from roughly 1°C to 33°C and followed a clear daily cycle, with warmer afternoons and cooler nights. Global solar radiation showed a distinct daytime-only pattern, peaking sharply around midday and dropping to zero during nighttime hours. Relative humidity tended to be higher during nighttime and rainy months and displayed a slight inverse relationship with temperature. The temperature–load relationship showed a weak positive correlation, with load increasing moderately as temperatures rise, which is typical for mixed-load areas where fans and cooling appliances see greater use. Solar radiation displayed a moderate positive correlation with load, as higher midday radiation often coincides with active residential and commercial activity. Humidity exhibited a weak negative correlation, since high humidity is generally associated with cloudy or rainy conditions during which daytime load may decrease slightly.

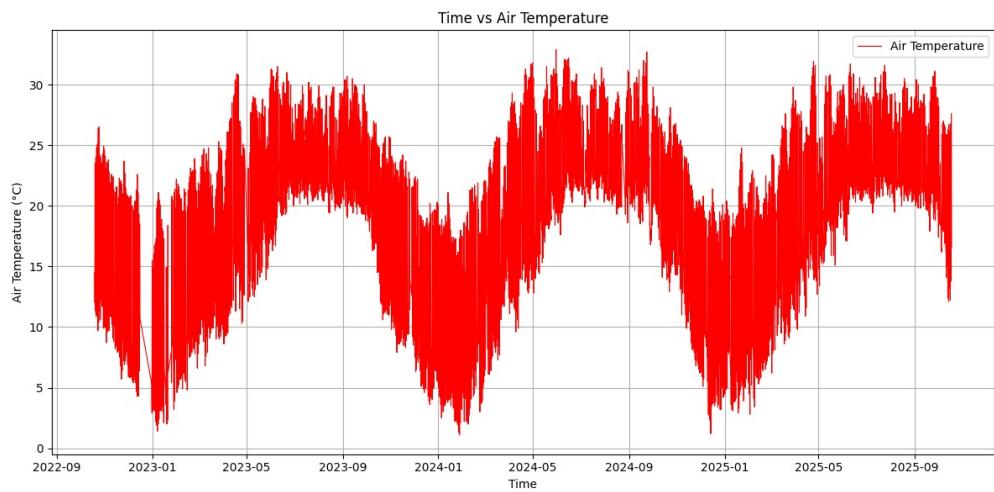


Figure 4.4 Air Temperature Variation Over the Study Period

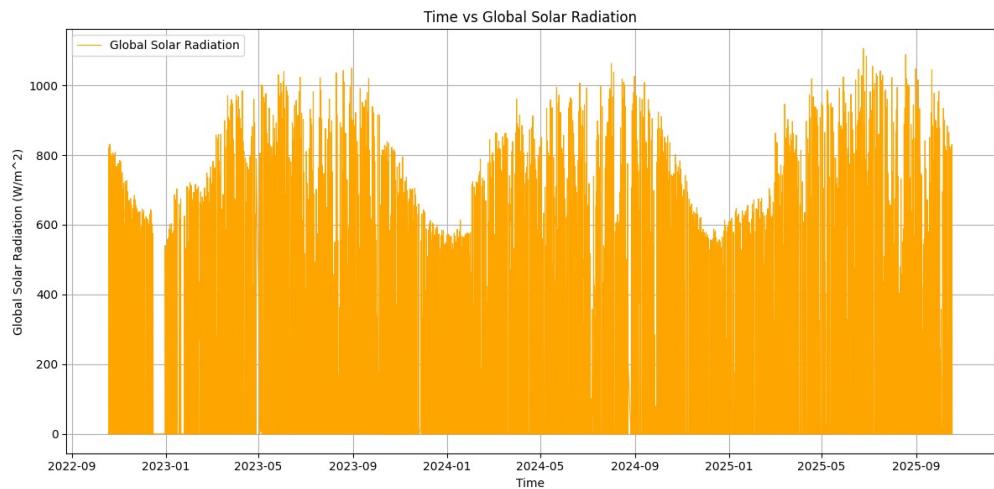


Figure 4.5 Global Solar Radiation Variation Over the Study Period

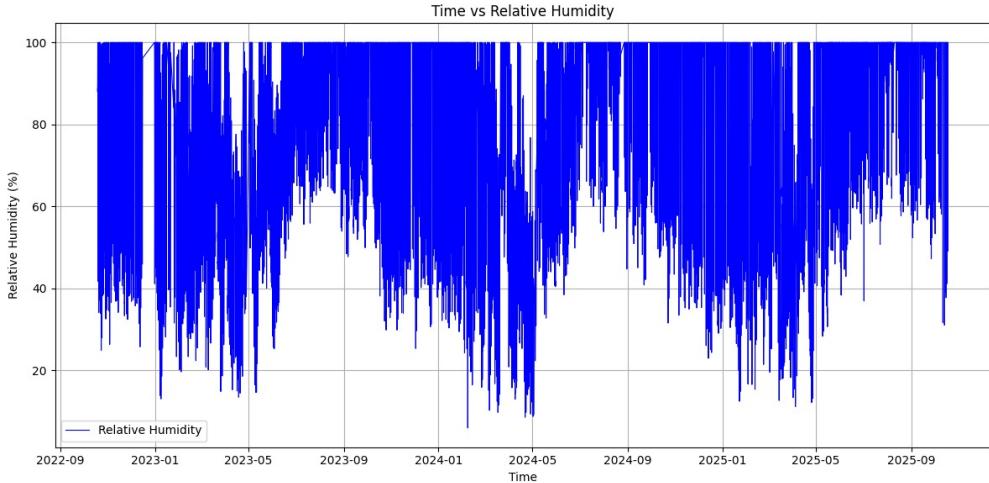


Figure 4.6 Relative Humidity Variation Over the Study Period

4.1.1. Outlier Treatment

During the data quality assessment phase, outliers in the MW (Megawatt) column were identified and treated. For the Baneshwor Feeder, double-digit MW values (i.e., $MW \geq 10$) are technically infeasible given the feeder's capacity and operational characteristics. Such extreme values likely represent sensor errors, data entry mistakes, or measurement anomalies rather than actual consumption.

A total of 56 records with MW values ≥ 10 were identified as outliers, representing approximately 0.23% of the total dataset. These outlier values ranged from 10.30 MW to 404.00 MW, which are clearly outside the feasible operating range for this feeder. A hard threshold approach was employed to remove all records with $MW \geq 10$ MW.

Figure 4.7 presents boxplots of the MW column before and after outlier removal. The left panel shows the original distribution with extreme outliers extending up to 404 MW, while the right panel displays the cleaned distribution with all MW values within the feasible single-digit range (0.00 to 8.50 MW). After outlier treatment, the cleaned dataset contained 24,352 records suitable for model training and evaluation.

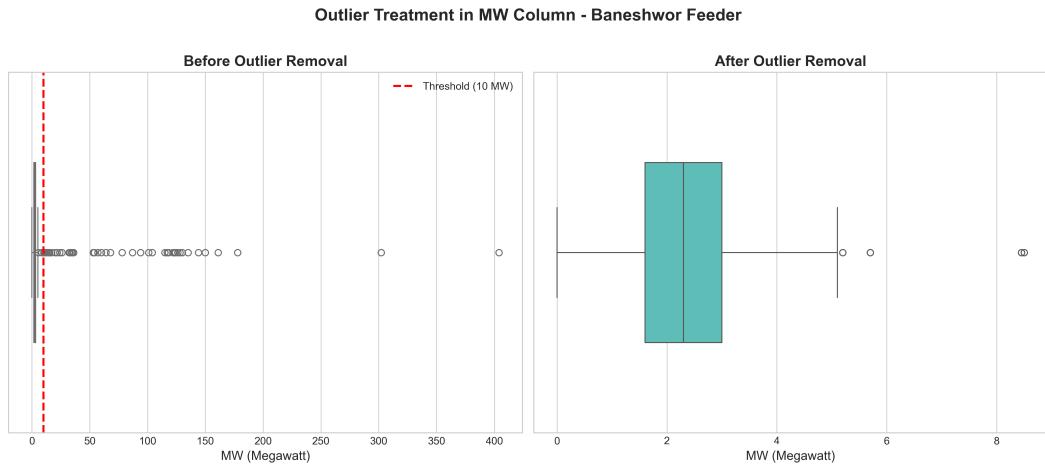


Figure 4.7 Boxplot of MW Column Before and After Outlier Removal

In addition to active power, the exogenous weather variables were also screened for anomalous values. Relative humidity readings below physically realistic levels and a few saturated values above the upper bound were removed, yielding a more compact distribution centered between roughly 60% and 100%. Air temperature exhibited a small number of extreme low and high values that were inconsistent with the local climate; these were treated using the same boxplot-based rule, resulting in a stable range of approximately 1°C to 33°C. Global solar radiation initially contained several unrealistically high spikes (exceeding 1000 W/m²), which were removed so that the cleaned series remained within a plausible envelope of about 0–800 W/m². Figures 4.8–4.10 summarize the effect of the outlier removal procedure on these three weather variables, where the left panel shows the original distribution and the right panel shows the cleaned distribution used for model training.

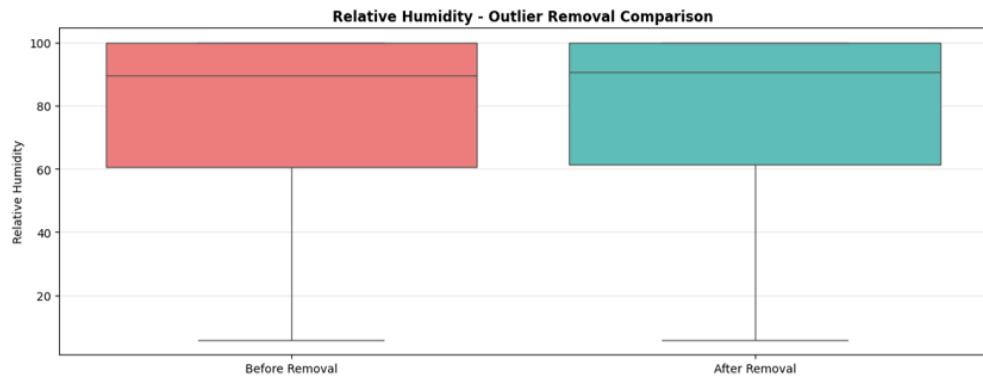


Figure 4.8 Relative Humidity – Outlier Removal Comparison

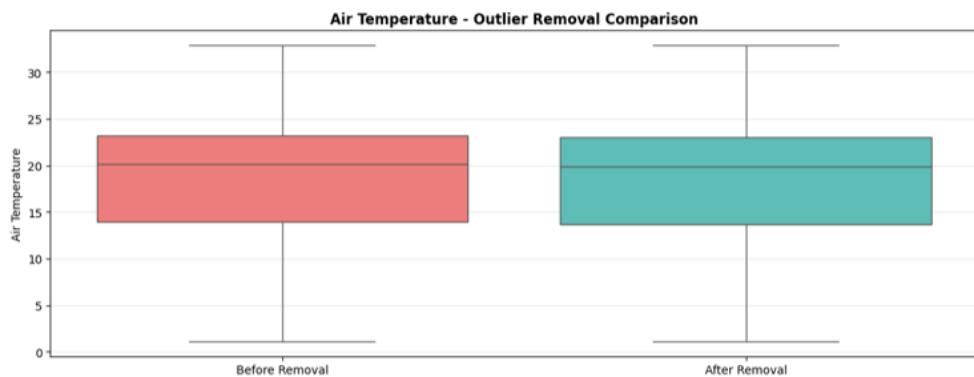


Figure 4.9 Air Temperature – Outlier Removal Comparison

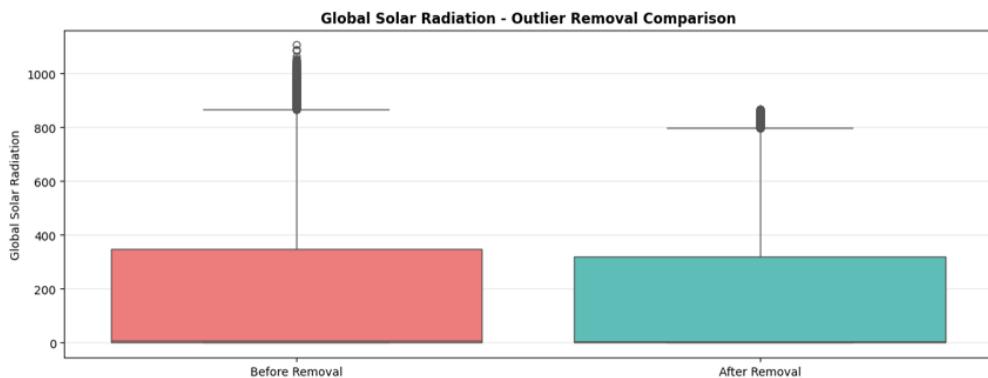


Figure 4.10 Global Solar Radiation – Outlier Removal Comparison

Table 4.1 presents the descriptive statistics of the MW column after outlier removal. The mean load of 2.35 MW and median of 2.30 MW indicate a relatively symmetric distribution. The standard deviation of 0.92 MW reflects moderate variability in hourly demand, while the minimum (0.00 MW) and maximum (8.50

MW) values confirm that all records now fall within the technically feasible range for the Baneshwor Feeder.

Table 4.1 Descriptive Statistics of MW Column (After Outlier Removal)

Statistic	Value
Mean	2.3483 MW
Median	2.3000 MW
Standard Deviation	0.9186 MW
Variance	0.8438 MW ²
Minimum	0.0000 MW
Maximum	8.5000 MW

4.2. Model Performance Results

4.2.1. Machine Learning Model Performance

All machine learning models were trained on the final feature-engineered dataset and evaluated on the test set. To optimize model performance, hyperparameter tuning was conducted using GridSearchCV with five-fold cross-validation. Table 4.2 summarizes the hyperparameter search space explored for each machine learning model. The best-performing configurations were selected based on the lowest validation error.

Table 4.2 Hyperparameter Search Space for Machine Learning Models

Model	Hyperparameters
Ridge Regression	$\alpha = [0.001, 0.01, 0.1, 1, 10, 100]$
Random Forest	Number of trees = [100, 200]
	Max depth = [10, 15, 20]
	Min samples split = [2, 5]
	Min samples leaf = [1, 2]
Gradient Boosting	Estimators = [100, 150, 200]
	Learning rate = [0.05, 0.1, 0.15]
	Max depth = [3, 5, 7]
XGBoost	Estimators = [100, 200]
	Max depth = [4, 6, 8]
	Learning rate = [0.05, 0.1]
	Subsample = [0.8, 1.0]
SVR	C = [1, 10, 100]
	Gamma = [scale, 0.01, 0.1]
	Epsilon = [0.01, 0.1, 0.5]

After tuning, the models were evaluated on the test set. Table 4.3 presents the performance of all machine learning models.

Table 4.3 Machine Learning Models Evaluation Matrix

Sn.No.	Model	MAE	RMSE	MAPE	R ²
1	XGBoost (Tuned)	0.257	0.384	12.693	0.831
2	Random Forest (Tuned)	0.294	0.435	14.290	0.783
3	Random Forest	0.305	0.444	14.825	0.774
4	XGBoost	0.313	0.449	15.242	0.769
5	Gradient Boosting	0.330	0.469	16.062	0.749
6	SVR	0.318	0.483	15.193	0.732
7	Ridge Regression	0.502	0.649	25.021	0.518
8	Linear Regression	0.502	0.649	25.021	0.518

4.2.1.1. Discussion of Results

Machine learning algorithm performance was quantified through MAE, RMSE, MAPE, and R-squared (R^2) to comprehensively assess forecasting capability. Linear Regression and Ridge Regression functioned as baseline comparators, yielding comparatively elevated error metrics and diminished R^2 coefficients, thereby evidencing their constrained capacity for capturing nonlinear interdependencies between consumption demand and predictor variables. Support Vector Regression demonstrated improvement over linear alternatives by reducing error magnitudes; nonetheless, its performance remained subordinate to ensemble-based methodologies, particularly regarding RMSE outcomes.

Tree-based ensemble algorithms exhibited markedly superior performance across all quantitative criteria. Both Random Forest and Gradient Boosting achieved substantial MAE and RMSE reductions while attaining higher R^2 coefficients, reflecting enhanced generalization capabilities and improved nonlinear pattern recognition. Among these, the hyperparameter-optimized XGBoost implementation surpassed all competing machine learning algorithms, achieving optimal RMSE (0.384), maximal R^2 (0.831), and minimal MAPE (12.693%). The performance differential relative to alternative ensemble approaches stems from XG-

Boost's gradient-based optimization strategy, integrated regularization mechanisms, and effective feature interaction handling, which collectively enhance model robustness and predictive reliability.

From an electricity market perspective, the XGBoost model's MAPE of 12.693% represents a significant improvement over baseline statistical methods and provides forecasts of sufficient accuracy for hour-ahead intraday market position adjustments [3]. In typical market operations, forecast errors translate directly into imbalance costs; therefore, reducing MAPE by even a few percentage points can yield substantial economic benefits over time. The model's consistent performance across different load levels, as evidenced by the high R^2 value, indicates reliable predictions during both peak and off-peak periods—a critical requirement for utilities managing their hour-ahead market positions and minimizing real-time imbalance exposure.

Collectively, the comparative assessment confirms that ensemble machine learning approaches, particularly XGBoost, deliver the most dependable and precise predictions for distribution feeder-level short-term demand forecasting applications, with performance characteristics suitable for supporting electricity market operations.

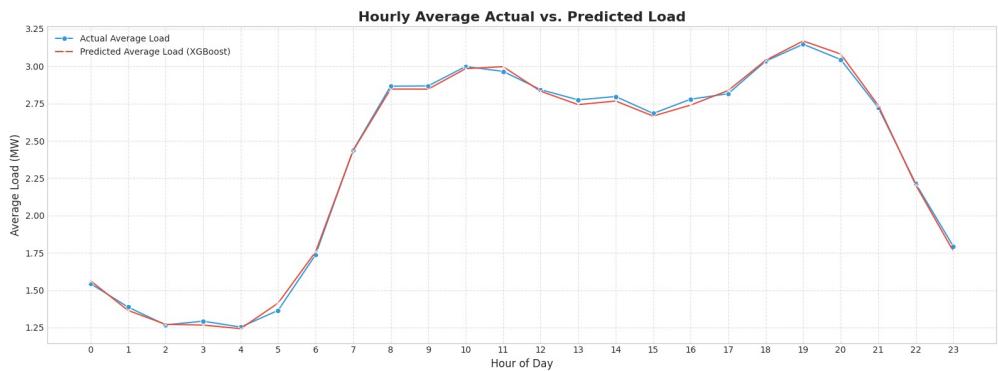


Figure 4.11 XBoost (Tuned) Actual vs Predicted

4.2.2. Deep Learning Model Performance

Deep learning models were trained using lag features and sliding-window sequences to capture temporal dependencies in the feeder load data. Two recurrent architectures were implemented: LSTM and GRU. Comprehensive experiments were conducted across multiple lag configurations and scaling methods to identify optimal configurations. Table 4.4 presents the hyperparameter and configuration search space explored for the deep learning models.

Table 4.4 Hyperparameter Search Space for Deep Learning Models

Component	Values
Lag Feature Configurations	
Short lags	[1, 3, 6] hours
Medium lags	[1, 3, 6, 12, 24] hours
Long lags	[1, 3, 6, 12, 24, 48] hours
Feature Scaling Methods	
MinMax Scaler	Scales to [0, 1] range
Standard Scaler	Zero mean, unit variance
LSTM / GRU Architecture	
Hidden units	[32, 64, 128]
Dropout rate	[0.0, 0.1, 0.2]
Activation function	ReLU
Sequence Modeling (LSTM/GRU)	
Look-back window	24 hours
Batch size	32
Training Parameters	
Learning rate	0.001
Optimizer	Adam
Max epochs	100
Early stopping patience	10 epochs
Data augmentation	2x (noise, jittering, scaling)

After conducting extensive training using the cleaned and augmented dataset, the models were evaluated using the same metrics as the ML models. The best results obtained for each deep learning architecture are presented in Table 4.5. The GRU model with long lag configuration [1, 3, 6, 12, 24, 48] and standard scaling achieved the best overall performance among the deep learning models.

Table 4.5 Deep Learning Models Evaluation Matrix

Sn.No.	Model	MAE	RMSE	MAPE	R²
1	GRU	0.192	0.289	7.364	0.879
2	LSTM	0.205	0.314	7.612	0.857

4.2.2.1. Discussion of Results

Deep neural network implementations—encompassing LSTM and GRU architectures—underwent evaluation employing identical performance criteria to ensure equitable cross-model comparison. Both recurrent architectures demonstrated strong forecasting performance, with GRU achieving the best results among deep learning models.

The GRU architecture achieved MAE of 0.192, RMSE of 0.289, and MAPE of 7.364%, attaining an R² coefficient of 0.879. The LSTM model also performed well, with MAE of 0.205, RMSE of 0.314, and MAPE of 7.612%, achieving an R² of 0.857. Both models demonstrated strong predictive capability, with R² values exceeding 0.85, indicating that the recurrent architectures effectively captured the temporal dependencies present in the feeder load data.

The GRU model's slightly superior performance over LSTM can be attributed to its simpler gating mechanism, which proved more efficient for this particular dataset while requiring fewer parameters and less computational overhead. Both recurrent models successfully learned the sequential patterns in electricity consumption, with the sliding window approach and lag feature configurations enabling effective temporal pattern recognition. Notably, the GRU achieved per-

formance competitive with the tuned XGBoost model ($R^2 = 0.831$), demonstrating that recurrent neural network approaches can provide accurate forecasts when appropriately configured for feeder-level load forecasting tasks.

From an electricity market operations standpoint, the GRU model's MAPE of 7.364% represents excellent forecasting accuracy that would significantly reduce imbalance costs in hour-ahead market participation scenarios. According to [6], forecast errors in the range of 5–10% MAPE are considered acceptable for intraday market operations in most electricity markets, placing both the GRU and LSTM models within the range suitable for practical hour-ahead market applications. The extended lag configuration [1, 3, 6, 12, 24, 48] hours effectively captures dependencies across multiple temporal scales, with particular emphasis on hour-ahead patterns critical for intraday trading and real-time position adjustments. This temporal learning capability makes the recurrent architectures particularly valuable for utilities that must rapidly adjust their market positions based on the most recent demand patterns and conditions.

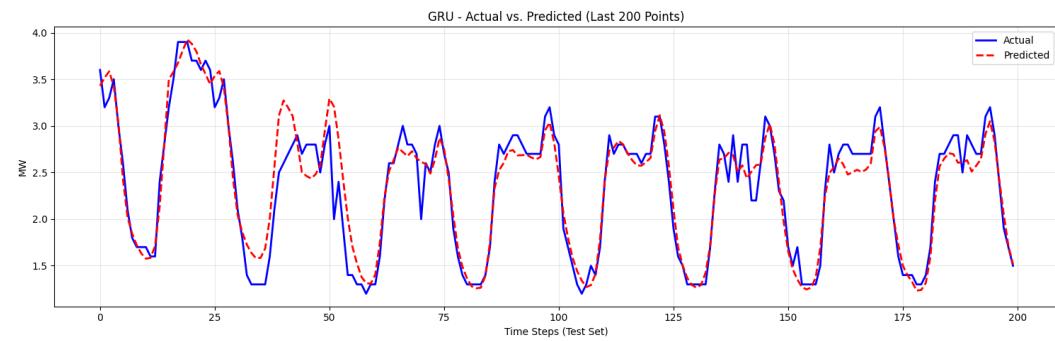


Figure 4.12 GRU Model Actual vs Predicted Values (Last 200 Test Points)

CHAPTER FIVE: CONCLUSION

5.1. Conclusion

This thesis constructed and rigorously assessed an integrated short-term electrical demand forecasting system tailored for the Baneshwor Feeder, employing both conventional machine learning and contemporary deep learning paradigms. The investigation adhered to a structured experimental protocol encompassing data procurement, quality assurance preprocessing, temporal characteristic extraction, and systematic cross-model performance benchmarking within a consistent evaluation framework. Importantly, the forecasting framework was designed with electricity market operational timelines in mind—specifically targeting the hour-ahead intraday market horizon, which represents the most dynamic trading window where utilities must rapidly adjust their positions based on the most recent demand information.

In modern electricity markets, the need for accurate short-term load forecasting cannot be overstated. Day-ahead markets require forecasts 24–48 hours in advance for unit commitment and initial position taking, while intraday markets depend on rolling hour-ahead predictions for position adjustments and real-time balancing. The forecasting models developed in this thesis are specifically aligned with these market operational requirements, enabling utilities to minimize imbalance costs, optimize trading strategies, and maintain supply-demand equilibrium efficiently [3, 11].

Empirical findings demonstrate that both machine learning algorithms and appropriately configured deep learning models can achieve excellent forecasting performance that meets the accuracy requirements for electricity market participation. Among machine learning approaches, the hyperparameter-optimized XG-Boost model achieved strong performance with an RMSE of 0.384, R² of 0.831, and MAPE of 12.693

Among deep neural network implementations, both recurrent architectures demonstrated strong forecasting capability with MAPE values below 8

Collectively, this investigation validates that precise distribution feeder-level short-term demand forecasting remains achievable through both judiciously architected machine learning workflows and appropriately configured recurrent neural network approaches. The forecasting accuracies achieved—particularly the GRU model’s MAPE of 7.364

As Nepal’s power sector evolves toward market-based operations and cross-border trading, the hour-ahead forecasting capabilities demonstrated in this thesis provide a foundation for efficient intraday market participation and grid management. Beyond technical operational efficiency, the implementation of such high-fidelity forecasting systems holds profound implications for the national economy. By minimizing demand-supply mismatches, utility providers can optimize the dispatch of domestic hydroelectric resources, significantly reducing energy wastage and the financial burden of unforeseen power imports. Furthermore, accurate intraday prediction empowers Nepal to maximize revenue in the regional power exchange market by strategically trading surplus energy, thereby strengthening the national trade balance. Ultimately, ensuring a reliable, data-driven, and economically viable power supply fosters industrial productivity and secures the long-term sustainability of the nation’s energy infrastructure.

5.2. Research Limitations

Although the study achieved strong forecasting accuracy, several limitations should be acknowledged:

1. **Dataset size constraints:** While the recurrent models achieved strong performance (GRU $R^2 = 0.879$, LSTM $R^2 = 0.857$), larger datasets spanning additional years could potentially improve model generalization and seasonal pattern recognition.
2. **Irregular load behaviour:** Feeder-level load profiles often contain noise,

outages, fluctuations, and sudden spikes, which can affect deep learning model training without additional contextual variables.

3. **Weather data resolution:** Weather data was available at hourly intervals only; finer granularity or additional environmental factors (e.g., wind speed, rainfall intensity) could further improve models.
4. **No real-time deployment environment:** The study focused on model development and evaluation, and did not include live deployment, automation, or integration with NEA's operational systems.
5. **Electricity market validation:** While the forecasting framework is designed with market timelines in mind, the models have not been validated in actual market trading scenarios. Real market performance may differ due to factors such as price volatility, bidding constraints, and regulatory requirements that were beyond the scope of this study.

These limitations provide important context when interpreting results and designing future enhancements.

5.3. Implications

The findings of this thesis carry several meaningful implications for utilities, researchers, system planners, and electricity market participants:

1. **Practical adoption for distribution feeders:** Both ensemble machine learning models (particularly XGBoost with $R^2 = 0.831$, MAPE = 12.693%) and recurrent deep learning models (GRU with $R^2 = 0.879$, MAPE = 7.364%) can provide accurate, low-error forecasts suitable for operational planning, peak management, and scheduling.
2. **Hour-ahead market readiness:** The forecasting accuracies achieved, particularly the GRU model's MAPE of 7.364%, fall within the acceptable range

for hour-ahead intraday market participation [3]. This positions the forecasting framework as a viable tool for supporting intraday position adjustments, reducing imbalance costs, and optimizing real-time power procurement decisions as Nepal’s electricity sector evolves toward market-based operations.

3. **Value of feature engineering:** Carefully constructed temporal features, lag variables, and weather features significantly improved forecasting accuracy, highlighting the importance of domain knowledge in model design. The extended lag configurations [1, 3, 6, 12, 24, 48] hours combined with sliding window sequences proved effective for capturing hour-ahead temporal dependencies critical for intraday market operations.
4. **Recurrent architecture effectiveness:** Both GRU ($R^2 = 0.879$) and LSTM ($R^2 = 0.857$) demonstrated strong predictive capability, confirming that recurrent neural networks are well-suited for capturing temporal dependencies in electricity load forecasting when properly configured.
5. **Foundation for hour-ahead market participation tools:** Forecasts from both the GRU and XGBoost models can support hour-ahead intraday market position adjustments, real-time balancing operations, demand-side management, smart grid optimization, load shifting strategies, and cross-border power trading decisions.
6. **Transferability:** The pipeline developed in this study can be extended to other NEA feeders with minimal modifications, promoting scalable forecasting across the network and enabling system-wide load aggregation for hour-ahead market operations.

5.4. Recommendation

Although this study demonstrates effective short-term load forecasting at the feeder level using both machine learning and deep learning models, several extensions can be explored to further enhance forecasting accuracy, practical applicability, and market relevance.

1. **Incorporation of Additional Influencing Factors:** Future work may include additional exogenous variables such as holiday indicators, special events, electricity price signals, and socio-economic factors to better capture demand variations that are not explained by weather and temporal features alone.
2. **Advanced Deep Learning Architectures:** Building upon the strong performance of GRU ($R^2 = 0.879$) and LSTM ($R^2 = 0.857$), future work could explore attention mechanisms, transformer-based architectures, or hybrid CNN-RNN models to potentially further improve hour-ahead forecasting accuracy.
3. **Real-Time Deployment and Online Learning:** Future research can focus on deploying the best-performing GRU or XGBoost model in a real-time operational environment, incorporating online or incremental learning techniques to adapt to evolving load patterns and enable continuous hour-ahead predictions.
4. **Integration with Hour-Ahead Market Systems:** As Nepal's power sector moves toward market liberalization and cross-border trading, future work could integrate load forecasts with intraday market optimization algorithms, developing decision support systems that translate hour-ahead demand predictions into optimal position adjustment strategies for intraday markets [10].
5. **Probabilistic Forecasting for Market Risk Management:** Extending the deterministic forecasts developed in this thesis to probabilistic forecasts (prediction intervals, quantile forecasts) would enable better risk quantification for hour-ahead market participation, helping utilities manage the financial uncertainty associated with forecast errors and volatile real-time market prices [3].

REFERENCES

- [1] E. Aguilar Madrid and N. Antonio. “Short-term electricity load forecasting with machine learning”. In: *Information* 12.2 (2021). doi: [10.3390/info12020050](https://doi.org/10.3390/info12020050).
- [2] K. Chapagain et al. “Short-term electricity demand forecasting for Kathmandu Valley, Nepal”. In: *Kathmandu University Journal of Science, Engineering and Technology* 15.3 (2021). doi: [10.3126/kuset.v15i3.63328](https://doi.org/10.3126/kuset.v15i3.63328).
- [3] Rafał Weron. “Electricity price forecasting: A review of the state-of-the-art with a look into the future”. In: *International Journal of Forecasting* 30.4 (2014), pp. 1030–1081.
- [4] Daniel S Kirschen and Goran Strbac. “Fundamentals of Power System Economics”. In: *John Wiley & Sons* (2018).
- [5] Tao Hong and Shu Fan. “Energy forecasting: Past, present, and future”. In: *Foresight: The International Journal of Applied Forecasting* 40 (2016), pp. 43–48.
- [6] Jesus Lago et al. “Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark”. In: *Applied Energy* 293 (2021), p. 116983.
- [7] Mohammad Shahidehpour, Hatim Yamin, and Zuyi Li. “Market operations in electric power systems: Forecasting, scheduling, and risk management”. In: *John Wiley & Sons* (2002).
- [8] S. Acharya, M. C. Luintel, and M. Badrudoza. “Short-term load forecasting of Gothatar feeder of Nepal Electricity Authority using Recurrent Neural Network”. In: *Unpublished manuscript* (2021).
- [9] M. K. Singla et al. “Electrical load forecasting using machine learning”. In: *International Journal* 8.3 (2019).

- [10] Antonio J Conejo, Miguel Carrión, and Juan M Morales. “Decision making under uncertainty in electricity markets”. In: *International Series in Operations Research & Management Science* 153 (2010).
- [11] Hamidreza Zareipour, Kankar Bhattacharya, and Claudio A Canizares. “Electricity market price volatility: The case of Ontario”. In: *Energy Policy* 35.9 (2007), pp. 4739–4748.
- [12] S. Desai et al. “Electrical load forecasting using machine learning”. In: *2021 International Conference on System, Computation, Automation and Networking (ICSCAN)*. 2021, pp. 1–6. doi: [10.1109/ICSCAN53069.2021.9526444](https://doi.org/10.1109/ICSCAN53069.2021.9526444).
- [13] P. Matrenin et al. “Medium-term load forecasting in isolated power systems based on ensemble machine learning models”. In: *Energy Reports* 8 (2022), pp. 612–618. doi: [10.1016/j.egyr.2021.11.175](https://doi.org/10.1016/j.egyr.2021.11.175).
- [14] W. Guo et al. “Machine learning-based methods in short-term load forecasting”. In: *The Electricity Journal* 34.1 (2021), p. 106884. doi: [10.1016/j.tej.2020.106884](https://doi.org/10.1016/j.tej.2020.106884).
- [15] M. Saglam et al. “Instantaneous electricity peak load forecasting using optimization and machine learning”. In: *Energies* 17.4 (2024). doi: [10.3390/en17040777](https://doi.org/10.3390/en17040777).
- [16] S. Jain and A. Gupta. “Comparative analysis of machine learning algorithms for short-term power load prediction”. In: *International Journal of Energy Systems* 19.2 (2024), pp. 55–68.
- [17] M. Cordeiro-Costas et al. “Load forecasting with machine learning and deep learning methods”. In: *Applied Sciences* 13.13 (2023), p. 7933. doi: [10.3390/app13137933](https://doi.org/10.3390/app13137933).
- [18] X. Dong et al. “A decade of deep learning-based short-term electricity load forecasting: A comprehensive review”. In: *Energy AI* 8 (2024), p. 100212.
- [19] J. Wen, H. Li, and T. Zhao. “GRU–TCN hybrid neural network with attention for short-term load forecasting”. In: *Energy* 290 (2024), p. 130423.

- [20] O. Alhussein et al. “Convolutional neural network and long short-term memory hybrid deep learning model for individual load forecasting”. In: *Energies* 13.19 (2020). DOI: [10.3390/en13195396](https://doi.org/10.3390/en13195396).
- [21] H. Hasanat, A. Biswas, and M. Abdullah. “Parallel multichannel CNN–BiLSTM architecture for smart-grid load forecasting”. In: *IEEE Access* 12 (2024), pp. 33211–33225.
- [22] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).
- [23] K. Chan and C. Yeo. “Sparse transformer-based architecture for electricity load forecasting”. In: *Applied Energy* 352 (2024), p. 122211.
- [24] Y. Zhang, Q. Li, and M. Chen. “Time-augmented transformer for short-term electrical load forecasting”. In: *IEEE Transactions on Power Systems* 37.6 (2022), pp. 5214–5225.
- [25] X. Lu and Y. Chen. “Multivariate data-slicing transformer neural network for load forecasting in renewable-integrated power systems”. In: *Electric Power Systems Research* 229 (2024), p. 110138.
- [26] S. Banik and S. Biswas. “A stacked ensemble learning model for renewable power and load forecasting”. In: *Energy Reports* 10 (2024), pp. 112–123.
- [27] Benjamin F Hobbs et al. “Evaluation of a truthful revelation auction in the context of energy markets with nonconcave benefits”. In: *Journal of Regulatory Economics* 18.1 (2001), pp. 5–32.
- [28] Arthur E. Hoerl and Robert W. Kennard. *Ridge regression: Biased estimation for nonorthogonal problems*. Vol. 12. 1. Taylor & Francis, 1970, pp. 55–67.
- [29] Harris Drucker et al. “Support vector regression machines”. In: *Advances in Neural Information Processing Systems* 9 (1997), pp. 155–161.
- [30] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.

- [31] Jerome H. Friedman. “Greedy function approximation: A gradient boosting machine”. In: *Annals of Statistics* 29.5 (2001), pp. 1189–1232.
- [32] Tianqi Chen and Carlos Guestrin. “XGBoost: A scalable tree boosting system”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2016, pp. 785–794.
- [33] Diederik P. Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [34] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780.
- [35] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078* (2014).