DEEP LEARNING AND ITS APPLICATIONS
(UEC642)

Project Report

Submitted by

Deepit Garg	(102215004)
Shaurya Narang	(102215278)
Kriti Singh	(102215297)
Mrikulesh Minhas	(102215357)

Submitted to

Dr. Gaganpreet Kaur

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Electronics and Communication Engineering Department

July -December 2025

ABSTRACT

Short-term electricity load forecasting plays a critical role in ensuring power system reliability, cost efficiency, and demand–supply balance. This project presents a complete forecasting framework integrating classical Machine Learning models—Random Forest, Gradient Boosting, XGBoost, and LightGBM—and a Deep Learning sequence model (LSTM) to predict hourly electricity load using the PJME dataset.

The proposed workflow includes comprehensive Exploratory Data Analysis (EDA), temporal and lag-based feature engineering, time-aware train-test split (train: pre-2018; test: 2018 onward), and unified evaluation of all models using MAE, RMSE, and MAPE. Visualization modules highlight daily, weekly, monthly, and seasonal trends. LSTM is trained on 72-step sequences after MinMax scaling to capture long-term temporal dependencies.

The results demonstrate that feature-rich boosting models perform strongly on structured data, while LSTM excels at learning patterns across time. The hybrid comparison shows competitive performance relative to contemporary research, providing a practical and interpretable solution for short-term grid forecasting.

CERTIFICATE

This is to certify that the project titled **“Weather-Based Power Load Forecasting using Machine Learning and Deep Learning Techniques”**, submitted by **Deepit Garg (102215004)**, **Shaurya Narang (102215278)**, **Kriti Singh (102215297)**, and **Mrikulesh Minhas (102215357)**, in partial fulfillment of the requirements of the course **UEC642: Deep Learning and Applications**, is an authentic and original piece of work carried out by them during the academic session **July–December 2025**.

The project has been completed under my supervision, and I am satisfied that the work presented is a result of the students’ own effort, dedication, and learning. The report has not been submitted, either wholly or partially, for any other course or evaluation.

Dr. Gaganpreet Kaur Assistant Professor Department of Electronics and Communication Engineering Thapar Institute of Engineering & Technology

ACKNOWLEDGMENT

We would like to express our heartfelt gratitude to **Dr. Gaganpreet Kaur** for her invaluable guidance, motivation, and continuous support throughout the duration of this project. Her expertise in the subject and insightful feedback played a crucial role in shaping the direction and quality of our work.

We also extend our sincere thanks to the **Electronics and Communication Engineering Department** for providing the necessary facilities, academic resources, and an encouraging environment that allowed us to explore, experiment, and learn effectively.

Additionally, we acknowledge the contributions of the **open-source community**, whose tools, libraries, and datasets significantly supported the development and implementation of our forecasting models. Their efforts made it possible for us to execute this project with a high degree of technical depth and reproducibility.

Finally, we are grateful to our peers and families for their constant encouragement and support during the course of this project.

CHAPTER 1: INTRODUCTION

1.1 Background

Modern electrical grids depend heavily on accurate short-term forecasting to manage generation, schedule power transactions, optimize energy markets, and prevent outages. Rising demand variability caused by weather patterns, industrial cycles, and behavioral changes further increases the need for robust prediction models.

Machine Learning models are highly effective in structured forecasting, while Deep Learning—especially LSTM—can capture long-range temporal patterns. A combined evaluation of these approaches provides valuable insights into model reliability and operational feasibility.

1.2 Project Overview

Accurate short-term electricity load forecasting is essential for ensuring grid stability, optimizing energy generation, reducing operational costs, and supporting strategic decision-making in modern power systems. With increasing demand variability and growing reliance on data-driven planning, advanced forecasting frameworks have become indispensable for utilities and grid operators.

This project develops a **comprehensive, end-to-end forecasting pipeline** using the PJME hourly energy load dataset. The workflow integrates both **classical machine learning** and **deep learning sequence models**, enabling a robust comparison between structured feature-based models and temporal neural architectures. The major components of the pipeline include:

- **Loading and preprocessing multi-year hourly load data**, handling missing values, outliers, and timestamp irregularities.
- **Performing extensive Exploratory Data Analysis (EDA)** to uncover daily, weekly, monthly, and seasonal consumption patterns.
- **Engineering time-based and lag-based features** that capture historical dependencies and temporal behavior.
- **Training and comparing multiple machine learning regression models**, including Random Forest, Gradient Boosting, XGBoost, and LightGBM.
- **Developing and evaluating an LSTM-based deep learning model** for sequence prediction and long-range temporal learning.
- **Visualizing predictions, error metrics, and model behavior** using intuitive plots and evaluation dashboards.
- **Selecting the best-performing model** based on objective metrics such as RMSE, MAE, and MAPE.

The entire project is implemented in **Python** and documented in a fully reproducible **Jupyter Notebook (Weather_Based_Power_Load_Forecasting.ipynb)**, ensuring transparency, clarity, and ease of extension for future research.

1.3 Need Analysis

With increasing variability in energy consumption and the rising integration of renewable energy sources, forecasting challenges have intensified. For instance, solar and wind power introduce additional uncertainty into grid operations, requiring more accurate demand predictions. Utilities must also anticipate peak loads to plan reserve margins and optimize power purchases, especially during extreme weather seasons. Therefore, forecasting systems must not only be accurate but also robust and adaptable to changing trends.

Short-term load forecasting directly influences grid stability, economic operation, renewable integration, and efficient peak demand management. Accurate predictions support energy purchasing, distribution infrastructure planning, and prevention of blackouts. This emphasizes the necessity of building an intelligent forecasting model capable of learning from both historical behavior and temporal structure.

1.4 Research Gaps

Despite advancements in forecasting research, several gaps remain unaddressed. Many existing studies silo machine learning or deep learning approaches without thoroughly comparing them under identical preprocessing steps. Similarly, models that rely solely on raw load values ignore contextual elements related to time, seasonality, and historical recurrence, which significantly affect accuracy. There is also limited work combining strong feature engineering with sequence models like LSTM to examine whether hybrid strengths can be achieved.

A review of recent literature suggests limitations in existing forecasting models:

- limited use of rich temporal features
- inadequate modeling of long-term dependencies
- lack of unified ML–DL comparisons
- absence of systematic evaluation under consistent conditions

Addressing these gaps is one of the key motivations behind this project.

1.5 Problem Definition & Scope

Growing electricity consumption and fluctuating energy usage patterns require forecasting systems that can identify behavioral shifts while maintaining reliability. The central challenge is designing a system that captures both short-term fluctuations (hourly or daily) and long-term cycles such as seasonal demand changes. Effective forecasting requires extracting meaningful features, tuning learning algorithms, and validating predictions carefully.

Problem Definition

Given historical hourly electricity load, the task is to predict future hourly load values while capturing periodicity, seasonality, and long-term temporal dependencies.

Scope of Work

- Utilize PJME Hourly Load dataset
- Implement ML models and LSTM
- Perform EDA, feature engineering, and visual exploration
- Evaluate models using MAE, RMSE, and MAPE
- Identify the model with the best predictive behavior

This scope reflects a focused forecasting pipeline without incorporating external variables such as weather or economic indicators.

1.6 Assumptions and Constraints

Forecasting models must operate under certain assumptions, particularly when data limitations exist. Since real-world energy datasets may contain missing or noisy values, preprocessing steps such as interpolation or outlier removal are essential to maintain signal integrity. Additionally, computational constraints play a role in deep learning training, influencing the sequence length and network design.

Assumptions

- The dataset accurately represents true load patterns.
- Missing values filled via interpolation do not distort trends.
- Outliers removed from the top 1% improve generalization.

Constraints

- No weather information included in current implementation.
- LSTM requires more computational resources.
- Chronological splitting is mandatory to avoid data leakage.

These constraints shape the modeling decisions and results interpretation throughout the project.

1.7 Standards Followed

To maintain reliability and reproducibility, the forecasting pipeline adheres to several standards. Time-series forecasting demands strict chronological separation between training and testing sets. Similarly, using widely accepted libraries and methodologies ensures compatibility with existing research and easier model validation.

Standards followed include:

- reproducible random seeds
- standardized ML/DL frameworks such as Scikit-learn and TensorFlow
- chronological train–test split
- consistent metric evaluation
- structured documentation and version control

These practices ensure that the results remain transparent and easy to replicate.

1.8 Approved Objectives

This project aims not only to build models but also to systematically understand how different approaches behave under identical conditions. By organizing the workflow around specific objectives, the project ensures methodological clarity and smooth execution.

The primary objectives are:

1. Conduct extensive EDA to reveal temporal patterns.
2. Engineer time-based and lag-based features.
3. Train multiple ML regression models.
4. Develop and evaluate an LSTM sequence model.
5. Compare ML and DL performance on unified metrics.
6. Identify and justify the best forecasting model.

These objectives guide the structure of the project and the interpretation of its results.

CHAPTER 2: LITERATURE SURVEY

Electricity load forecasting has been widely studied across power systems engineering, applied mathematics, and machine learning domains. With the increasing penetration of renewables and fluctuating energy consumption patterns, researchers have explored a wide range of classical statistical methods, machine learning ensemble models, and deep learning architectures for short-term and long-term load prediction. This chapter provides a structured review of ten recent and influential research works, highlighting their methodologies, contributions, and relevance to the current project. It also discusses the theoretical principles underlying the forecasting problem and examines existing forecasting systems.

2.1 Review of Recent Research Studies

Recent studies focus on combining feature engineering, sequence learning, hybrid modeling, and advanced neural networks to enhance prediction accuracy. The following table summarizes ten relevant research papers, along with their techniques and key findings:

Literature Review Summary

1. **Wen et al. (2024)** – Proposed a **CNN–LSTM hybrid architecture**, demonstrating that convolutional layers effectively extract local temporal features before sequence modeling. Result: significant improvement in forecasting accuracy over standalone LSTM models.

2. **Mu et al. (2023)** – Developed an **LSTM-based Seq2Seq model** tailored for multi-step forecasting. The encoder–decoder structure improved long-range dependency modeling and reduced error accumulation in multi-horizon predictions.

3. **Abumohsen et al. (2023)** – Compared **LSTM, GRU, and RNN** models for load prediction. Concluded that LSTM consistently outperformed other recurrent networks due to its superior memory retention and gating mechanisms.

4. **Gao et al. (2023)** – Introduced an **Adaptive Transformer** model, showcasing the ability of self-attention mechanisms to capture long-range dependencies without sequential processing limitations.

5. **Wang et al. (2024)** – Proposed a **CNN + Extended LSTM hybrid**, where CNN layers extract high-frequency components and LSTM layers handle long-term patterns. Achieved robust performance across diverse seasonal cycles.

6. **Singh et al. (2022)** – Investigated **Random Forest** with engineered lag features. Demonstrated that ensemble tree models can achieve high accuracy when supplied with rich temporal features.

7. **Deb et al. (2021)** – Explored **Boosting Algorithms (XGBoost, GBM)** for load forecasting. Reported that boosting models reduce both bias and variance, offering strong performance on structured datasets.

8. **Zhang et al. (2023)** – Conducted a comprehensive comparison between **LSTM and XGBoost**. Found that while XGBoost excels on feature-based datasets, LSTM outperforms in learning sequential patterns.

9. **Kumar et al. (2024)** – Studied **Seasonal-Trend Decomposition (STL) and SARIMA**, highlighting the importance of isolating seasonal components to improve predictive interpretability.

10. **Bhatt et al. (2023)** – Demonstrated that **feature-rich ML models** incorporating hour, weekday, and seasonal attributes significantly increase forecasting accuracy compared to models using only raw load data.

2.1.1 Theoretical Foundation of Load Forecasting

Electricity load forecasting is fundamentally a **time-series prediction problem**, characterized by periodicity, temporal dependence, and nonlinear patterns. The following theoretical concepts underpin the predictive modeling process:

- **Seasonality** – Electricity consumption follows strong hourly, daily, weekly, and yearly periodic cycles driven by human activities and environmental factors.
- **Trend Components** – Long-term upward or downward trends arise due to economic growth, urban development, or technological adoption.
- **Autocorrelation** – Past load values are highly correlated with future values; therefore, temporal dependencies must be captured using lag features or sequence models.
- **Time-Series Decomposition** – Splitting a signal into trend, seasonal, and residual components improves interpretability and assists in designing better models.
- **Machine Learning Theory** – Ensemble models like Random Forest, XGBoost, and LightGBM reduce variance and handle nonlinear relationships effectively.
- **Deep Learning Theory** – LSTM networks employ gated memory cells capable of learning long-term sequential relationships, making them well-suited for dynamic load patterns.

These theoretical foundations justify the use of both ML and DL approaches in this project.

2.1.2 Existing Systems and Forecasting Approaches

Traditional forecasting systems have primarily relied on **statistical models** such as:

- **ARIMA (Auto-Regressive Integrated Moving Average)**
- **Holt–Winters Exponential Smoothing**
- **SARIMA (Seasonal ARIMA)**

While these models are effective for stationary or moderately varying datasets, they struggle with nonlinear patterns, sudden demand peaks, and long-term seasonal structures.

Modern forecasting solutions incorporate:

- **Gradient Boosting Models (XGBoost, LightGBM)** for structured data and engineered features
- **Hybrid CNN–LSTM Models** for simultaneous short-term and long-term pattern extraction
- **Transformer Models** for better long-range context modeling using self-attention
- **Feature-Rich ML Pipelines** that outperform classical models when lag and temporal attributes are included

However, many existing systems still **lack the combination** of:

- strong feature engineering,
- multiple baseline ML models,
- and sequence-based deep learning comparison
  within a single unified pipeline — which this project aims to deliver.

2.1.3 Summary of Literature Insights

From the reviewed studies, the following insights emerge:

- Temporal and lag-based features significantly enhance ML performance.
- LSTM and attention-based models excel in capturing long-term dependencies.
- Hybrid systems outperform single-model approaches.
- Lack of comprehensive comparative frameworks limits reproducibility across studies.

This motivates the design of a pipeline that is **feature-rich**, **hybrid**, and thoroughly evaluated across both ML and DL architectures.

CHAPTER 3: REQUIREMENT ANALYSIS

A requirement analysis defines the essential capabilities, constraints, and resources needed to successfully implement the proposed forecasting system. This chapter outlines the functional and non-functional requirements, along with software and hardware dependencies. These requirements ensure that the system operates efficiently, produces reliable predictions, and maintains reproducibility across different environments. Proper identification of requirements also helps in guiding the design, development, and execution of the overall forecasting pipeline.

3.1 Functional Requirements

The functional requirements describe the core operations that the forecasting system must perform to achieve the intended objectives. These requirements ensure that the system handles data correctly, generates predictive inputs, trains models effectively, and communicates outputs clearly.

Functional Requirements Include:

- Loading and preprocessing hourly load data:
  The system must be able to ingest multi-year PJME hourly data, convert timestamps to a proper datetime index, handle missing values, remove duplicates, and treat outliers appropriately.

- Generating time-based and lag-based features:
  The system must extract features such as hour, weekday, month, day-of-year, and lagged values (e.g., 24-hour and 168-hour lags) to enrich model input and capture temporal dependencies.

- Training Machine Learning and Deep Learning models:
  The system must support training classical ML regressors (RF, GBM, XGB, LGBM) as well as LSTM-based sequence models.

- Comparing model performance:
  The system must evaluate all models using standardized metrics (MAE, RMSE, MAPE) and identify the best-performing model.

- Producing visual outputs:
  The system must generate prediction plots, error histograms, training-loss curves, and actual-vs-predicted graphs for interpretability and result validation.

These functional requirements ensure that the forecasting pipeline is complete, scalable, and capable of delivering actionable insights.

3.2 Non-Functional Requirements

Non-functional requirements define the quality attributes of the system. They focus on how the system should perform rather than what it should do. For time-series forecasting, maintaining accuracy, robustness, and reproducibility is essential.

Key Non-Functional Requirements:

- High accuracy and low error:
  Forecasts should minimize RMSE and MAPE to ensure reliable short-term load predictions for practical grid applications.

- Robustness across different time periods:

The forecasting model must remain stable across seasonal variations, peak periods, and data fluctuations.

- Reproducibility via fixed random seeds:
  All experiments must use consistent random seeds to ensure repeatable results regardless of system or environment differences.

- Interpretability of ML models:
  Models such as Random Forest and XGBoost should offer interpretable feature importance to help understand the contribution of time-based features.

These non-functional requirements guarantee that the forecasting system remains valid, trustworthy, and meaningful in real-world contexts.

3.3 Software Requirements

The software stack must support data preprocessing, visualization, machine learning, and deep learning model training. All tools selected for the project are open-source, widely used, and compatible with Jupyter Notebook implementations.

Required Software:

- Python 3.8+ – Primary programming language for system development
- TensorFlow – Framework for implementing and training the LSTM model
- Scikit-learn – Library for classical ML models and preprocessing utilities
- Pandas, NumPy – Libraries for efficient data manipulation and numerical operations
- Matplotlib, Seaborn – Visualization libraries for generating plots and analytical graphics

These tools collectively provide a robust ecosystem for implementing an end-to-end forecasting solution.

3.4 Hardware Requirements

The project's computational requirements depend on the complexity of the models and the size of the dataset. While classical ML models are lightweight, sequence models like LSTM require more significant processing capability.

Minimum Hardware Requirements:

- CPU for classical ML training:
  Ensemble models such as Random Forest and XGBoost can be efficiently trained on modern multi-core CPUs.

- GPU recommended for LSTM training:
  A dedicated GPU accelerates matrix computations, enabling faster training and convergence of deep learning models.

- Minimum 8 GB RAM:

Required to load multi-year hourly data, generate lag features, and handle computational overhead during model training.

These hardware specifications ensure smooth execution of the forecasting pipeline without performance bottlenecks.

CHAPTER 4: METHODOLOGY

The methodology defines the systematic workflow followed to build the forecasting pipeline. It covers data preprocessing, exploratory analysis, feature engineering, model training, and evaluation. Each stage is designed to ensure accuracy, reproducibility, and meaningful interpretation of results. The complete methodology is implemented in the Jupyter Notebook *Weather_Based_Power_Load_Forecasting.ipynb*.

4.1 Data Preprocessing

Efficient forecasting begins with clean and well-structured data. The PJME hourly dataset contains multi-year load values indexed by timestamps. Preprocessing ensures high-quality input for feature engineering and model training.

Preprocessing Steps:

- **Timestamp Conversion:** Timestamps are parsed into a proper datetime index to enable time-based operations.
- **Handling Missing Values:** Missing load values are filled using time-based interpolation to preserve trend continuity.
- **Duplicate Removal:** Duplicate timestamps, if any, are removed to maintain data consistency.
- **Outlier Filtering:** The top 1% of extreme values are removed based on statistical thresholds to avoid skewed learning.
- **Resampling & Sorting:** Data is ordered chronologically and resampled if required.

This stage produces a clean and reliable dataset ready for exploratory and predictive modeling.

4.2 Exploratory Data Analysis (EDA)

EDA helps uncover the underlying patterns, seasonality, and trends present in the dataset. It guides decisions about suitable models and features.

Key EDA Visualizations:

- **Hourly, daily, and yearly line plots** to observe overall consumption behavior.
- **Seasonal decomposition** into trend, seasonal, and residual components.
- **Distribution plots** to understand the load variability.
- **Heatmaps** for hourly vs. weekday patterns.
- **Monthly and seasonal consumption trends.**

- **24-hour radial plots** representing cyclical consumption activity.

These analyses reveal strong daily and weekly seasonality, validating the need for lag features and sequence-based learning models.

4.3 Feature Engineering

Feature engineering transforms raw data into informative input variables. Well-designed features significantly improve the performance of classical ML models.

Engineered Features Include:

Time-Based Features

- Hour
- Day of week
- Month
- Quarter
- Year
- Day of year
- Week of year
- Weekend indicator

These features capture recurring temporal patterns.

Lag Features

- **Lag_24:** Load value from exactly 24 hours earlier
- **Lag_168:** Load value from 168 hours earlier (one week earlier)

These features are crucial because electricity usage tends to follow strong daily and weekly habits.

Together, these engineered features form a robust input matrix for ML models.

4.4 Train–Test Split

A time-aware split is essential for forecasting tasks to avoid data leakage.

- **Training Set:** All data before 2018
- **Testing Set:** All data from 2018 onward

This ensures that the model is evaluated on unseen future data, simulating real-world forecasting.

A graphical split visualization is also included in the notebook.

4.5 Machine Learning Models

Several classical regression models are trained using engineered features:

- **Random Forest Regressor**
- **Gradient Boosting Regressor**
- **XGBoost Regressor**
- **LightGBM Regressor**

Each model learns patterns from structured features, making them fast, interpretable, and effective for large datasets.

Evaluation Metrics Used:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Percentage Error (MAPE)

A comparative table is generated to identify the best-performing model.

4.6 Deep Learning Model: LSTM

LSTM (Long Short-Term Memory) networks are ideal for sequential data. They capture long-term dependencies better than classical models.

LSTM Methodology:

1. **Scaling:** Load values are normalized using MinMaxScaler.
2. **Sequence Generation:** 72-step sliding windows are created as input sequences.

3. **Model Architecture:** LSTM layers with dropout regularization to prevent overfitting.
4. **Training:** Model is trained using Adam optimizer with validation monitoring.
5. **Loss Curves:** Training vs validation loss plots are generated.
6. **Evaluation:** Predictions are compared using MAE, RMSE, and MAPE.

LSTM results are added to the ML comparison table.

4.7 Final Model Comparison

The final stage compares all trained models and identifies the one with the lowest RMSE. Additional visualizations include:

- Actual vs Predicted curves
- Error distribution histograms
- Scatter plots of predictions
- Performance bar charts

This ensures a holistic understanding of each model's strengths and limitations.

CHAPTER 5: RESULTS & DISCUSSION

The results reflect the performance of both machine learning and deep learning models trained on the PJME hourly dataset. Visual and numerical evaluation metrics provide a clear understanding of model behavior.

5.1 Model Comparison Table

	MAE	RMSE	MAPE
LSTM	471.379130	613.922368	0.015286
LightGBM	1989.970082	2695.048210	0.061786
Gradient Boosting	2069.840415	2838.367822	0.064307
Random Forest	2093.801423	2861.774214	0.064940
XGBoost	2118.495908	2889.569823	0.065385

This table highlights how boosting models and LSTM perform under the same data and metrics.

5.2 Visual Analysis

Several visual outputs help interpret the model performance:

- **Actual vs Predicted Plot:** Shows how closely model predictions follow true load values.
- **Train–Test Visualization:** Confirms proper chronological splitting.
- **LSTM Loss Curves:** Depict learning progression and potential overfitting.
- **Error Histogram:** Shows distribution of prediction error magnitudes.
- **Scatter Plot:** Illustrates correlation between actual and predicted values.
- **Seasonal Decomposition:** Confirms the dataset's strong seasonality and trend components.

These visuals collectively confirm that the models successfully learn underlying demand patterns.

5.3 Discussion

From the evaluation, several insights emerge:

- Models with well-engineered features (XGBoost, LightGBM) achieve low error rates and high robustness.
- LSTM models capture temporal continuity well, especially during stable consumption periods.
- ML models are faster and more interpretable, while LSTM adds deeper temporal understanding.
- Peak load predictions remain challenging due to sudden demand spikes.

Overall, the results align with patterns found in recent forecasting literature.

CHAPTER 6: CONCLUSION

This project successfully designed and implemented a complete forecasting pipeline using both machine learning and deep learning models. The PJME hourly load dataset was thoroughly analyzed through EDA, feature engineering, and visualization techniques. Classical ML models demonstrated strong performance when supplied with rich temporal features, while LSTM excelled at learning sequential dependencies inherent in time-series data.

The combined comparative approach revealed that hybrid pipelines, which use engineered features alongside sequential modeling, offer reliable forecasting performance suitable for real-world grid management. The methodology, evaluation, and results strengthen the case for adopting data-driven forecasting tools in modern power systems.

CHAPTER 7: PROJECT OUTCOMES

The key outcomes of the project include:

- Development of a complete, reproducible forecasting pipeline
- Extraction of meaningful temporal and lag-based features
- Training of four ML models and an LSTM model
- Identification of the best-performing model
- Insightful visualizations highlighting patterns and errors
- Validation against recent studies and research trends

The project enhances understanding of time-series forecasting and demonstrates practical implementation of hybrid ML–DL systems.

CHAPTER 8: FUTURE SCOPE

While the forecasting pipeline performs effectively, several extensions can further improve accuracy and applicability:

- **Incorporating weather variables** such as temperature, humidity, and wind speed
- **Experimenting with Transformer architectures**, which are state-of-the-art for sequence modeling
- **Building a real-time forecasting dashboard** for power utilities
- **Implementing probabilistic forecasting** (quantile regression, confidence intervals)
- **Including economic and behavioral features** to capture non-seasonal fluctuations
- **Deploying models on cloud platforms** for production-level predictions

These enhancements open pathways for more robust, scalable, and intelligent energy forecasting systems.

REFERENCES

1. Wen et al. (2024). *Deep learning-driven hybrid model for short-term load forecasting*. CNN-LSTM hybrid improves forecasting accuracy over standalone models.
   View Paper
2. Mu et al. (2023). *An improved LSTM-Seq2Seq-based forecasting method for power load prediction*. Demonstrates enhanced sequence learning for long-horizon forecasts.
   View Paper
3. Abumohsen et al. (2023). *Electrical Load Forecasting Using LSTM, GRU, and RNN Models*. Comparing multiple RNN variants; LSTM outperformed others.
   View Paper
4. Gao et al. (2023). *Adaptive Transformer-based deep-learning load forecasting*. Shows Transformer models capture long-range dependencies effectively.
   View Paper
5. Wang et al. (2024). *Load forecasting using CNN and extended LSTM*. CNN layer improves feature extraction before LSTM sequence modeling.
   View Paper
6. Singh et al. (2022). *Random Forest–Based Short-Term Load Forecasting*. Found RF provides robust performance with engineered lag features.
   View Paper
7. Deb et al. (2021). *Boosting Algorithms for Energy Load Forecasting*. Reports superior accuracy for Gradient Boosting and XGBoost on energy datasets.
   View Paper
8. Zhang et al. (2023). *Comparative Study of LSTM and XGBoost for Energy Demand Forecasting*. Concludes LSTM better captures long-term patterns.
   View Paper
9. Kumar et al. (2024). *Seasonal-Trend Decomposition for Electricity Load Forecasting*. Demonstrates seasonal components significantly influence forecasting results.
   View Paper
10. Bhatt et al. (2023). *Feature-rich forecasting models for electricity consumption*. Shows time-based features

(hour, weekday) significantly improve performance.
View Paper

Dataset Link

11. PJME Hourly Energy Load Dataset. *Hourly electricity consumption dataset used for forecasting tasks.* Available on: Kaggle / PJM Interconnection Data Portal.
    View Dataset

Library Documentation

12. TensorFlow (2023). *TensorFlow Deep Learning Framework Documentation*.
    View Documentation
13. Scikit-learn (2023). *Scikit-learn Machine Learning Library Documentation*.
    View Documentation
14. Statsmodels (2023). *Time Series Analysis Tools (ARIMA, SARIMAX, STL)*.
    View Documentation
15. XGBoost Developers (2023). *XGBoost Library Documentation*.
    View Documentation
16. LightGBM Developers (2023). *LightGBM Gradient Boosting Framework Documentation*.
    View Documentation
17. NumPy Developers (2023). *NumPy Numerical Computing Tools*.
    View Documentation
18. Pandas Developers (2023). *Pandas Data Analysis Library Documentation*.
    View Documentation
19. Matplotlib (2023). *Matplotlib Visualization Library*.
    View Documentation
20. Seaborn (2023). *Statistical Data Visualization Library*.
    View Documentation
