Electric Load Forecasting of Godishala Village Substation

Tapas Kumar Maiti
tapas_kumar@dau.ac.in
Mentor, Dhirubhai Ambani University
Gandhinagar, India

Chandan Pandit
202418043@dau.ac.in
Dhirubhai Ambani University
Gandhinagar, India

Vivek Gautam
202418053@dau.ac.in
Dhirubhai Ambani University
Gandhinagar, India

1 PROJECT DESCRIPTION

Electric power load data is derived from hourly voltage (V), current (I), and power factor (pf) measurements at the 33/11 kV substation in Godishala, Huzurabad, Telangana, India, spanning January 1 to December 31, 2021. The dataset includes 8760 hourly data points, with 3-phase active power calculated in kilowatts (kW). Additional variables include day status (weekday: 0, weekend: 1), season (Winter: 1, Summer: 2, Rainy: 0), temperature (°F), and humidity (%). Missing values (66 hours) due to maintenance or outages were imputed using the average of corresponding hours from the previous and next valid days, based on weekday rules. Load statistics are:

- Mean Load: 2130 kW
- Standard Deviation: 1302 kW
- Minimum Load: 412 kW
- Peak Load: 6306 kW

2 PROJECT OBJECTIVE

The objective is to forecast electric demand for Godishala village using historical data from the local 33/11 kV substation, enabling efficient power management.

3 DATASET SOURCE

The dataset is sourced from:
`https://data.mendeley.com/datasets/tj54nv46hj/1`

4 DATASET LOCATION

Data was collected from the 33/11 kV substation in Godishala village, Huzurabad mandal, Karimnagar district, Telangana, India, serving residential, commercial, and small industrial consumers.

5 DATASET DESCRIPTION

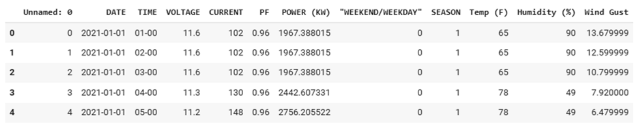
The dataset variables are described below:

Column	Description
DATE	Calendar date (e.g., 2021-01-01)
TIME	Hour of the day (e.g., 13:00)
VOLTAGE (V)	Line voltage in volts
CURRENT (A)	Current in amperes
PF	Power factor (0 to 1)
POWER (kW)	Real power consumed
WEEKEND	0 = Weekday, 1 = Weekend
SEASON	Winter = 1, Summer = 2, Rainy = 0
Temperature (°F)	Ambient temperature
Humidity (%)	Humidity percentage
Wind Gust	Maximum wind speed (mph or km/h)

Table 1: Dataset Variables

6 VISUALIZATIONS

Key visual insights from the data are presented below:



	Unnamed: 0	DATE	TIME	VOLTAGE	CURRENT	PF	POWER (KW)	"WEEKEND/WEEKDAY"	SEASON	Temp (F)	Humidity (%)	Wind Gust
0	0	2021-01-01	01-00	11.6	102	0.96	1967.388015	0	1	65	90	13.679999
1	1	2021-01-01	02-00	11.6	102	0.96	1967.388015	0	1	65	90	12.599999
2	2	2021-01-01	03-00	11.6	102	0.96	1967.388015	0	1	65	90	10.799999
3	3	2021-01-01	04-00	11.3	130	0.96	2442.607331	0	1	78	49	7.920000
4	4	2021-01-01	05-00	11.2	148	0.96	2756.205522	0	1	78	49	6.479999

Figure 1: Dataset Columns Overview

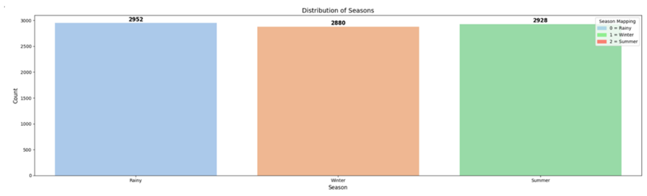


Figure 2: Bar Chart of Seasons
Insight: The data is nearly balanced across seasons (Rainy: 2952, Winter: 2880, Summer: 2928), reducing seasonal bias in forecasting models and enabling effective capture of seasonal load patterns (e.g., higher summer demand).

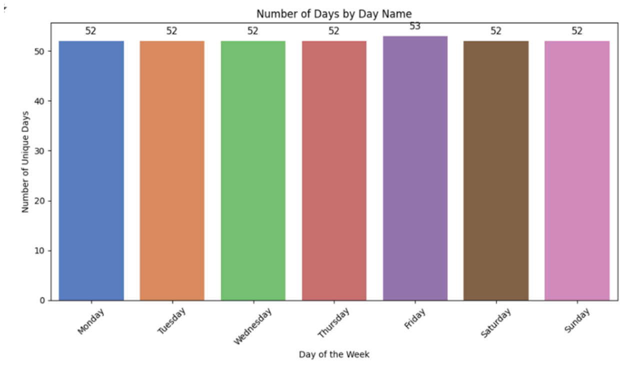


Figure 3: Bar Chart of Daily Load Distribution
Insight: Daily load data is nearly uniform (52 days each, except Friday: 53), ensuring accurate modeling of weekday vs. weekend patterns without introducing bias.

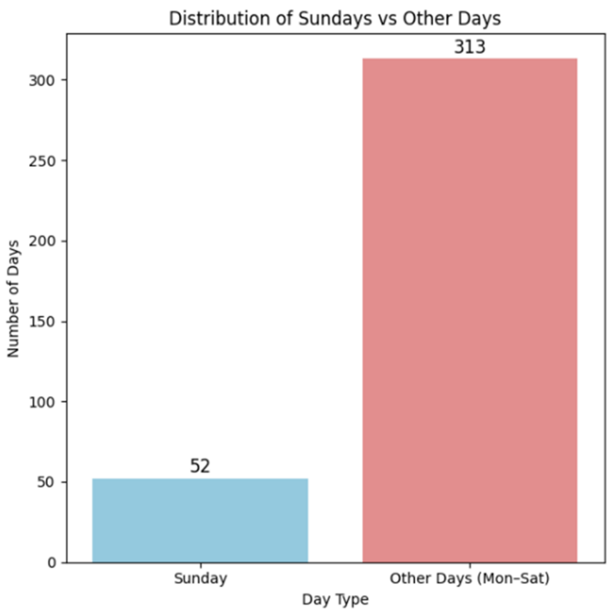
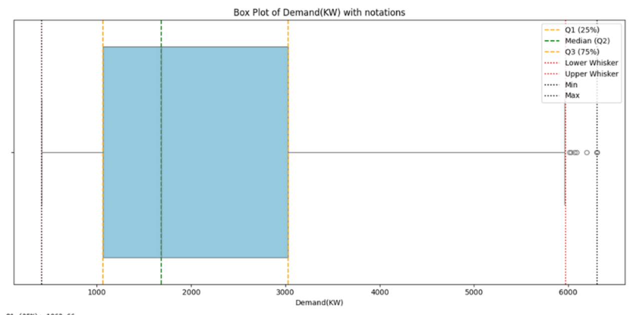


Figure 4: Bar Chart: Sunday vs Other Days
Insight: Sundays (52) vs. other days (313) reflect 14% weekend data, enabling analysis of typically lower Sunday demand and supporting more tailored load forecasting models.



```
Q1 (25%): 1062.66
Median (Q2): 1683.69
Q3 (75%): 3025.77
IQR: 1963.11
Lower Whisker: 412.03
Upper Whisker: 5970.44
Min: 412.03
Max: 6306.21
Number of Outliers: 7
```

Figure 5: Box Plot of Demand (kW)
Insight: Demand is right-skewed, with 50% of values between 1062.66–3025.77 kW. The presence of 7 outliers above 5970.44 kW may indicate equipment anomalies or unexpected surges.

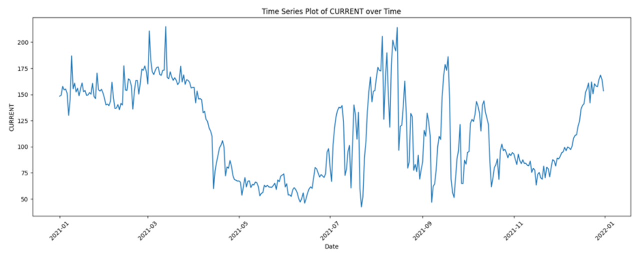


Figure 6: Time Series Plot of Current (A)
Insight: Current shows seasonal peaks during summer (likely due to cooling load) along with clear daily and weekly cycles, making it well-suited for time series modeling and load forecasting.

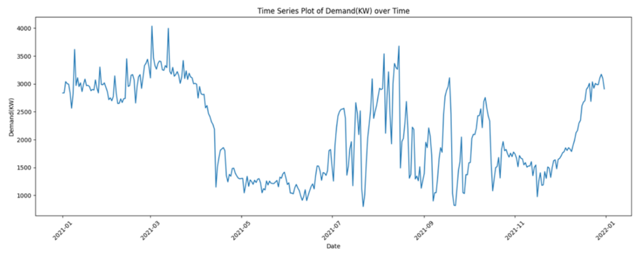


Figure 7: Time Series Plot of Demand (kW)
Insight: Demand peaks in summer and dips in winter, showing strong seasonal and daily/weekly cycles. These patterns make the data suitable for time series models like SARIMA or Prophet.

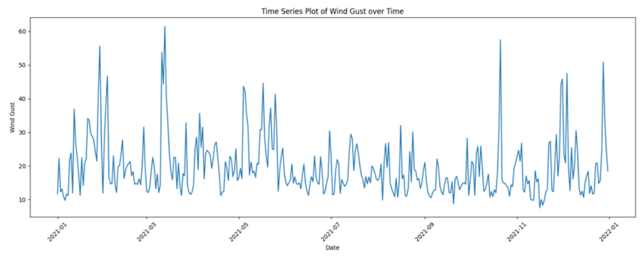


Figure 8: Time Series Plot of Wind Speed
Insight: Wind speed fluctuates most during the rainy season but shows minimal correlation with demand, indicating low predictive value for load forecasting models.

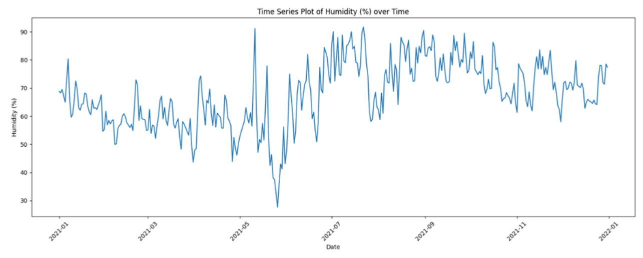


Figure 9: Time Series Plot of Humidity (%)
Insight: Humidity peaks during the rainy season and shows a negative correlation with temperature (-0.65), which may indirectly influence demand patterns through temperature-driven consumption behavior.

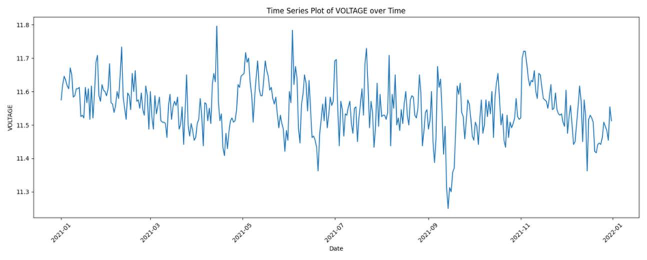


Figure 10: Time Series Plot of Voltage (V)
Insight: Voltage remains generally stable with minor fluctuations and shows low correlation with demand, indicating limited usefulness as a predictive feature in forecasting models.

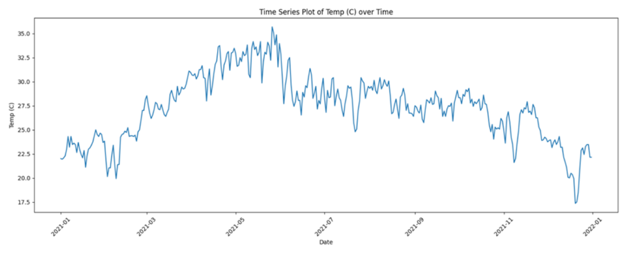


Figure 11: Time Series Plot of Temperature (°F)
Insight: Temperature peaks in summer and dips in winter, closely aligning with demand patterns. This seasonal variation makes temperature a valuable feature for improving forecasting accuracy.

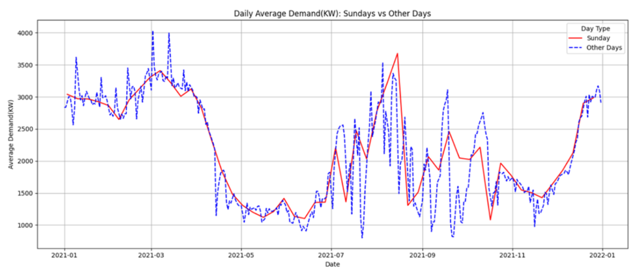


Figure 12: Time Series Plot of Demand: Sunday vs Other Days
Insight: Sundays consistently show lower demand compared to other days, reflecting reduced weekend usage. Capturing this pattern is critical for improving forecasting accuracy, especially in weekly cycle models.

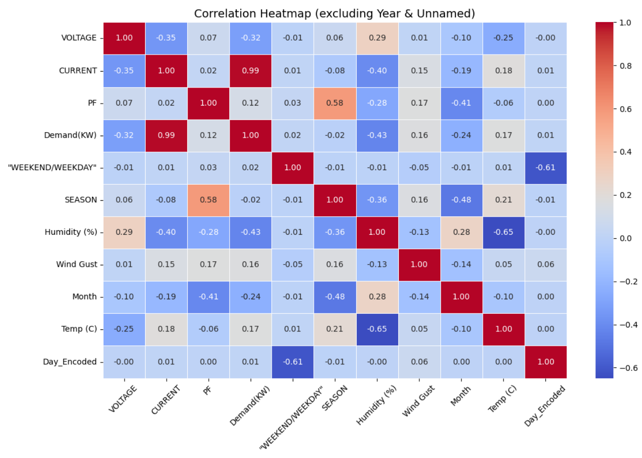


Figure 13: Correlation Heatmap
Insight: Demand shows a strong positive correlation with current (0.99) and a weak correlation with power factor (0.12). Season influences power factor (0.58), while temperature and humidity are negatively correlated (-0.65). Calendar-based features exhibit low correlation, suggesting limited direct predictive power.

6.1 Time Series Decomposition:

Decomposition is the process of separating a time series into its constituent components: *Trend*, *Seasonal*, and *Residual*. This technique helps understand the underlying patterns in the data and prepares it for forecasting or anomaly detection.

- **Actual:** The original observed time series data.
- **Trend:** The long-term direction or movement in the data, indicating increasing or decreasing patterns over time.
- **Seasonal:** Repeating short-term cycles or periodic fluctuations (e.g., daily, weekly, yearly).
- **Residual:** The random noise or irregular component left after removing trend and seasonality.

By analyzing these components individually, we can select appropriate forecasting models (e.g., SARIMA for seasonal data, Prophet for additive seasonality and holidays) and better interpret the behavior of the system being monitored.

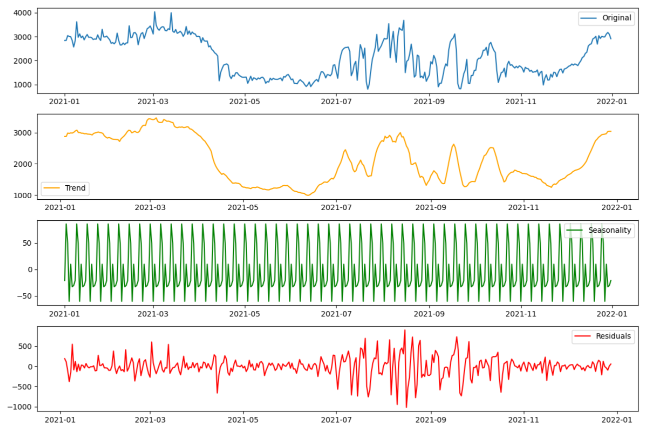


Figure 14: Time Series Decomposition of Demand
Insight: Demand experiences a sharp drop in April–May followed by a steady rise. The decomposition reveals strong weekly seasonality and noticeable mid-year residual volatility, making the data well-suited for models like SARIMA or Prophet.

6.2 Augmented Dickey-Fuller (ADF) Test

The Augmented Dickey-Fuller (ADF) test is a statistical test used to check whether a time series is stationary or not. Stationarity means the statistical properties of the series such as mean and variance remain constant over time. Many time series forecasting models require the data to be stationary for reliable performance.

Why use ADF test?

- To test the null hypothesis that the series has a unit root (i.e., is non-stationary).
- To determine if differencing or other transformations are needed to make the series stationary.
- Helps in selecting appropriate time series models (e.g., ARIMA).

ADF Test Results:

- ADF Statistic: **-4.551621**
- p-value: **0.000158**
- Critical Values:
  - 1%: -3.431
  - 5%: -2.862
  - 10%: -2.567
- **Conclusion:** Since the p-value < 0.05 and the ADF statistic is less than all critical values, we reject the null hypothesis of non-stationarity. This confirms the data is stationary.

7 NAIVE FORECASTING

Naive Forecasting

The Naive Forecasting model is one of the simplest time series prediction methods. It assumes that the value at the next time step is the same as the last observed value. Despite its simplicity, it is widely used as a baseline to assess the performance of more complex models.

Working Principle: For a given time series $\{y_t\}$, the Naive model predicts:

$$\hat{y}_{t+1} = y_t$$

This means the forecast for the next time step is simply the most recent observed value.

Limitation: The model does not account for any underlying trend, seasonality, or pattern in the data. This makes it unsuitable for time series with strong temporal structures.

Evaluation: The Naive model served as a baseline. It achieved poor R^2 scores across all datasets (0.09–0.25), indicating limited predictive power. The constant forecast failed to capture hourly dynamics, making it unsuitable for practical demand forecasting.

Table 2: Naive Forecasting Evaluation Metrics

Dataset	MAE	MSE	RMSE	R^2
Train	727.46	1,493,013.78	1221.89	0.2487
Validation	671.40	1,286,016.45	1134.03	0.0724
Test	560.92	912,084.25	955.03	0.0928

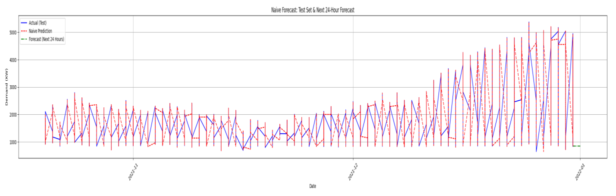


Figure 15: Naive Forecast vs Actual Demand for 24 Hours

Date & Time	Predicted Demand (kW)
2021-12-31 01:00:00	839.84
2021-12-31 02:00:00	839.84
2021-12-31 03:00:00	839.84
2021-12-31 04:00:00	839.84
2021-12-31 05:00:00	839.84
⋮	⋮
2021-12-31 20:00:00	839.84
2021-12-31 21:00:00	839.84
2021-12-31 22:00:00	839.84
2021-12-31 23:00:00	839.84
2022-01-01 00:00:00	839.84

Table 3: Naive Forecast for the Next 24 Hours (First and Last 5)

7.1 Autocorrelation and Partial Autocorrelation

7.1.1 Definition of ACF and PACF. **Autocorrelation Function (ACF)**
ACF measures the correlation between the time series and its own lagged values.

$$\mathrm{ACF}(k) = \frac{\sum_{t=k+1}^{T}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T}(y_t - \bar{y})^2} \tag{1}$$

where:

- y_t is the time series value at time t
- $\bar{y}$ is the mean of the series
- k is the lag

Partial Autocorrelation Function (PACF)
PACF measures the correlation between the series and its lagged values after removing the effect of shorter lags.

$$\mathrm{PACF}(k) = \phi_{kk} \tag{2}$$

where ϕ_{kk} is the last coefficient in the autoregression of order k that best fits the series.

7.1.2 Interpretation of ACF and PACF Plots. **ACF Plot (Top):**

- ACF gradually declines slowly.
- Many lags up to lag 20 are significantly positive (outside the blue confidence interval).
- This behavior is typical of an AR or ARMA process with persistence.
- Suggests strong autocorrelation — current demand values are highly dependent on past values.

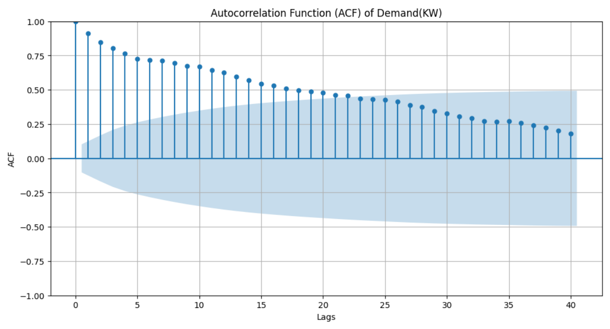


Figure 16: Autocorrelation Function (ACF) Plot

PACF Plot (Bottom):

- PACF shows a sharp drop after lag 1.
- Only the first lag is significantly different from zero.
- Indicates an AR(1) process is sufficient to explain the autocorrelation structure.

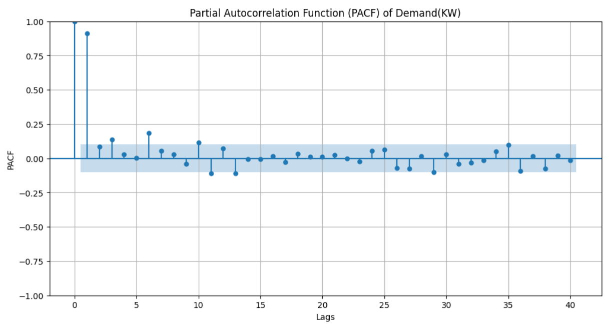


Figure 17: Partial Autocorrelation Function (PACF) Plot

8 AR(1) MODEL

The Autoregressive model of order 1, denoted as AR(1), is a fundamental statistical model for time series forecasting. It assumes that the current value of the series is primarily dependent on its immediate past value with some noise.

How It Works

In an AR(1) model, the future value $\hat{y}_t$ is modeled as a linear combination of its previous value y_{t-1} and a stochastic error term ϵ_t:

$$\hat{y}_t = \phi y_{t-1} + \epsilon_t$$

Where:

- $\hat{y}_t$ is the predicted value at time t,
- ϕ is the autoregressive coefficient (lag-1),
- ϵ_t is white noise (error term).

Strength: Effectively captures short-term temporal correlations and is computationally efficient.

Limitation: Its linear structure restricts it from modeling nonlinear dynamics or sudden shifts in load.

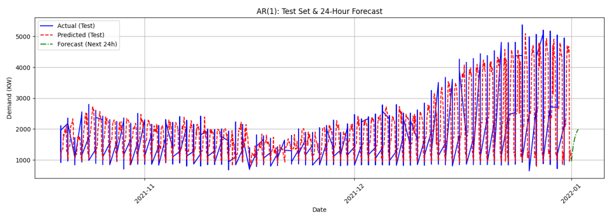


Figure 18: AR(1) Forecast vs Actual Demand (Next 24 Hours)

Model Evaluation

The AR(1) model significantly improved performance, with consistent R^2 above 0.82 across all data splits and decreasing MAE and RMSE. The forecast exhibited a smooth rising trend in demand, making it suitable for short-term load planning.

Table 4: AR(1) Model Evaluation for Demand (kW)

Dataset	MAE	MSE	RMSE	R^2
Train	315.48	335065.02	578.85	0.8314
Validation	268.37	232863.42	482.56	0.8327
Test	220.49	180451.48	424.80	0.8204

Table 5: AR(1) Forecast for the Next 24 Hours (First and Last 5)

Date & Time	Predicted Demand (kW)
2022-01-01 01:00:00	952.74
2022-01-01 02:00:00	1055.75
2022-01-01 03:00:00	1149.74
2022-01-01 04:00:00	1235.49
2022-01-01 05:00:00	1313.73
⋮	⋮
2022-01-01 20:00:00	1922.54
2022-01-01 21:00:00	1940.58
2022-01-01 22:00:00	1957.04
2022-01-01 23:00:00	1972.06
2022-01-02 00:00:00	1985.76

9 VANILLA RNN MODEL

Recurrent Neural Networks (RNNs) are a class of neural networks designed for sequence data, making them well-suited for time series forecasting tasks such as electric load prediction. Unlike traditional feedforward neural networks, RNNs maintain a memory of previous inputs through recurrent connections, allowing them to capture temporal dependencies.

How It Works

At each time step t, the RNN updates its hidden state h_t using the current input x_t and the previous hidden state h_{t-1}. The prediction $\hat{y}_t$ is then made from the hidden state:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

$$\hat{y}_t = W_{hy}h_t + b_y$$

Where:

- W_{hh}, W_{xh}, W_{hy} are weight matrices,
- b_h, b_y are bias terms,
- σ is the activation function (typically tanh or ReLU),
- h_t is the hidden state capturing sequence information,
- $\hat{y}_t$ is the predicted output.

Strength: RNNs capture sequential patterns and temporal dynamics, especially useful for multistep load forecasting.

Limitation: Vanilla RNNs struggle with long-term dependencies due to vanishing gradients. More advanced variants like LSTM and GRU are often preferred in such cases.

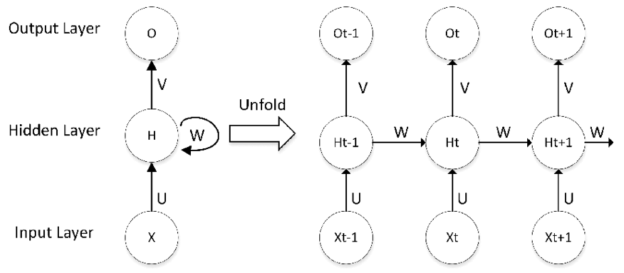


Figure 19: Vanilla RNN Architecture for Load Forecasting

9.1 Best Hyperparameters

units1: 128, **l2_reg:** 0.001, **dropout1:** 0.2, **units2:** 32, **dropout2:** 0.3, **lr:** 0.001

9.2 Performance Metrics

Table 6: Vanilla RNN Model Evaluation

Dataset	MAE	MSE	R^2
Train	236.76	131452.21	0.9316
Validation	293.55	180950.32	0.8443
Test	178.64	94557.82	0.9084

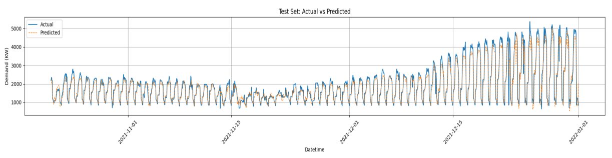


Figure 20: Vanilla RNN Forecast vs Actual Demand (Next 24 Hours)

9.3 Forecast for the Next 24 Hours

Table 7: Vanilla RNN Forecast (Next 24 Hours)

Date & Time	Predicted Demand (kW)
2021-12-31 01:00:00	1952.49
2021-12-31 02:00:00	1809.87
2021-12-31 03:00:00	1809.02
2021-12-31 04:00:00	1954.13
2021-12-31 05:00:00	2374.45
⋮	⋮
2021-12-31 20:00:00	1553.07
2021-12-31 21:00:00	1217.96
2021-12-31 22:00:00	948.90
2021-12-31 23:00:00	695.20
2022-01-01 00:00:00	920.27

10 BIDIRECTIONAL LSTM MODEL

Bidirectional Long Short-Term Memory (BiLSTM) networks are a powerful variant of RNNs designed to capture both past and future dependencies in sequence data. Unlike standard LSTMs that process input in one temporal direction (past to future), BiLSTMs run two LSTMs in parallel: one forward and one backward. This allows the model to capture richer temporal representations, making it particularly effective for electric load forecasting with hourly trends and seasonal dependencies.

How LSTM Works Internally

Each LSTM unit contains three types of gates that regulate the flow of information:

- **Forget Gate:** Decides what information to discard from the previous cell state.

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

- **Input Gate:** Decides what new information to add to the cell state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad \tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

- **Output Gate:** Determines the output at current time step based on the updated cell state.

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

- **Cell State Update:**

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

- **Hidden State:**

$$h_t = o_t \cdot \tanh(C_t)$$

Where σ is the sigmoid activation, x_t is the input at time t, and h_{t-1} is the previous hidden state.

BiLSTM Combination:

A BiLSTM processes input in both forward and backward directions:

$$\overrightarrow{h_t} = \text{LSTM}(x_t, \overrightarrow{h_{t-1}}), \quad \overleftarrow{h_t} = \text{LSTM}(x_t, \overleftarrow{h_{t+1}})$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$$

This concatenated output h_t enhances temporal understanding by integrating past and future context.

Strength: Captures bidirectional temporal patterns for better accuracy. **Limitation:** Computationally more expensive than unidirectional models.

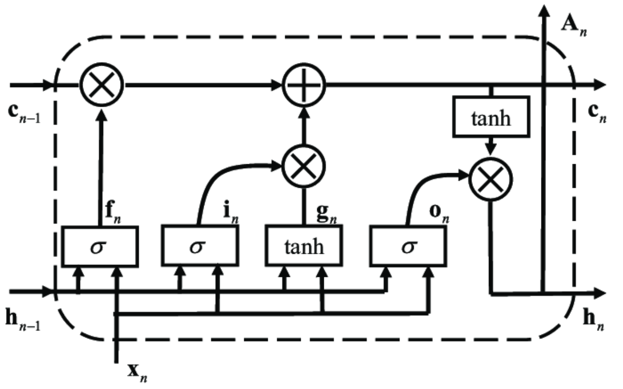


Figure 21: Bidirectional LSTM Architecture for Load Forecasting

10.1 Best Hyperparameters

The optimal architecture and training configuration was identified using Keras Tuner. The best hyperparameters obtained are:

- **Units in LSTM Layer 1:** 128
- **L2 Regularization:** 0.001
- **Dropout Rate after Layer 1:** 0.2
- **Units in LSTM Layer 2:** 32
- **Dropout Rate after Layer 2:** 0.2
- **Learning Rate:** 0.001

These hyperparameters contributed to improved generalization and convergence speed during model training.

10.2 Performance Metrics

Table 8: Bidirectional LSTM Model Evaluation

Dataset	MAE	MSE	R^2
Train	240.52	139570.82	0.9157
Validation	217.79	106907.48	0.8825
Test	179.60	97415.74	0.9139

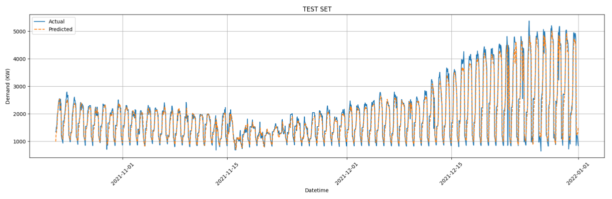


Figure 22: Bidirectional LSTM Forecast vs Actual Demand (Next 24 Hours)

10.3 Forecast for the Next 24 Hours

Table 9: Bidirectional LSTM Forecast (First and Last 5 Hours)

Date & Time	Predicted Demand (kW)
2021-12-31 01:00:00	1574.33
2021-12-31 02:00:00	1784.07
2021-12-31 03:00:00	2131.56
2021-12-31 04:00:00	2705.95
2021-12-31 05:00:00	3571.82
⋮	⋮
2021-12-31 20:00:00	4413.03
2021-12-31 21:00:00	3971.18
2021-12-31 22:00:00	3580.77
2021-12-31 23:00:00	3291.15
2022-01-01 00:00:00	3136.65

11 BIDIRECTIONAL GRU MODEL

Bidirectional Gated Recurrent Unit (BiGRU) networks are a streamlined variant of RNNs designed for capturing temporal dependencies while using fewer parameters than LSTMs. Like BiLSTMs, BiGRUs operate in both forward and backward directions, enabling the model to learn contextual information from past and future time steps, making it suitable for electric load forecasting where such dependencies are crucial.

How GRU Works Internally

GRU units simplify the architecture of LSTMs by combining the forget and input gates into a single update gate, and using a reset gate to manage the flow of information:

- **Update Gate:** Determines how much of the past information to carry forward.

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

- **Reset Gate:** Controls how much past information to forget.

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

- **Candidate Activation:** Calculates the candidate hidden state.

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

- **Final Hidden State:**

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

Where σ is the sigmoid activation function, x_t is the current input, and h_{t-1} is the previous hidden state.

BiGRU Combination:

$$\overrightarrow{h_t} = \text{GRU}(x_t, \overrightarrow{h_{t-1}}), \quad \overleftarrow{h_t} = \text{GRU}(x_t, \overleftarrow{h_{t+1}})$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$$

This bidirectional structure improves the model's ability to interpret sequential patterns with minimal overhead.

Strength: Efficient memory usage and fewer parameters than BiLSTM. **Limitation:** May underperform on tasks requiring longer-term memory compared to LSTM.

Best Hyperparameters

- **Units (Layer 1):** 64
- **Dropout (Layer 1):** 0.3
- **Units (Layer 2):** 64
- **Dropout (Layer 2):** 0.3
- **L2 Regularization:** 0.001
- **Learning Rate:** 0.0005

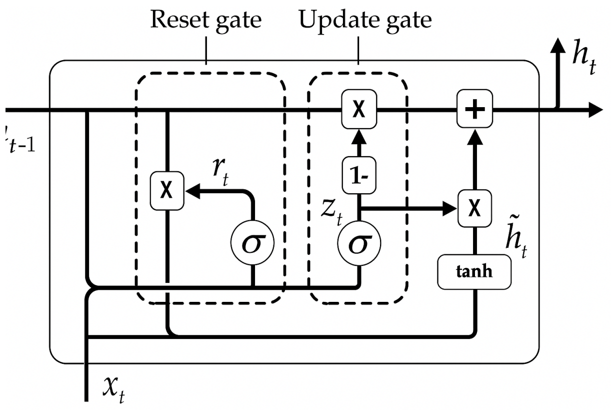


Figure 23: Bidirectional GRU Architecture for Load Forecasting

Model Performance

Table 10: Performance Metrics for Bidirectional GRU Model

Dataset	MSE	MAE	R^2 Score
Train	124,707.66	223.58	0.9188
Validation	109,835.57	222.84	0.8800
Test	99,876.38	181.99	0.9133

These results indicate the BiGRU model generalizes well, maintaining low error across all datasets and achieving high R^2 scores above 0.91 on both train and test sets.

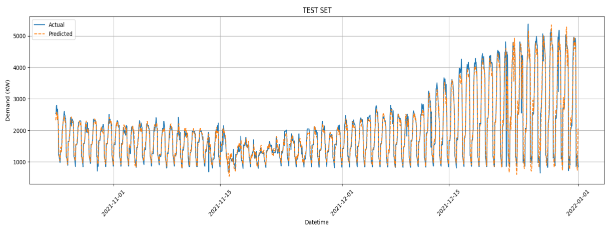


Figure 24: Bidirectional GRU Forecast vs Actual Demand (Next 24 Hours)

11.1 Forecast for the Next 24 Hours

Table 11: Bidirectional GRU Forecast (First and Last 5 Hours)

Date & Time	Predicted Demand (kW)
2021-12-31 01:00:00	1812.23
2021-12-31 02:00:00	2059.99
2021-12-31 03:00:00	2323.11
2021-12-31 04:00:00	2602.56
2021-12-31 05:00:00	2922.39
⋮	⋮
2021-12-31 20:00:00	5777.07
2021-12-31 21:00:00	4742.32
2021-12-31 22:00:00	3741.43
2021-12-31 23:00:00	2854.48
2022-01-01 00:00:00	2179.31

12 CONCLUSION

This study explored advanced recurrent neural network architectures—Vanilla RNN, Bidirectional LSTM (BiLSTM), and Bidirectional GRU (BiGRU)—for short-term electric load forecasting. To ensure proper model design, statistical analyses including the Augmented Dickey-Fuller (ADF) test, Autocorrelation Function (ACF), and Partial Autocorrelation Function (PACF) were performed.

ADF Test: The Augmented Dickey-Fuller test was used to determine the stationarity of the electricity demand time series. The test yielded a statistic of **-4.551621** with a p-value of **0.000158**, which is below the 5% significance level. The ADF statistic was also lower than all critical values (1%: -3.431, 5%: -2.862, 10%: -2.567), allowing us to reject the null hypothesis of non-stationarity and conclude that the data is stationary.

ACF and PACF: The ACF plot showed a slow, gradual decay across many lags, indicating strong autocorrelation and long-memory behavior. The PACF plot exhibited a sharp cutoff after lag 1, which is characteristic of an AR(1) process. These insights confirmed the presence of significant short-term dependencies, validating the use of autoregressive-based models like RNNs for this dataset.

Vanilla RNN: The Vanilla RNN effectively captured the temporal dependencies in the data, achieving strong performance metrics: $R^2 = 0.9316$ (train) and $R^2 = 0.9084$ (test), with low MAE and MSE. Although the validation $R^2 = 0.84$ was slightly lower, this indicates only mild overfitting. The RNN aligned well with the ACF/PACF-informed temporal structure.

Strengths:

- Captures nonlinear and sequential patterns in electric demand data.
- Aligns well with the AR(1)-like temporal structure highlighted by PACF.
- Stationarity confirmed via ADF test improves training stability.

Limitations:

- Struggles with long-term dependencies.
- Sensitive to sequence formatting and input scaling.

Bidirectional LSTM (BiLSTM): BiLSTM extended RNNs with bidirectional processing and long-term memory capabilities. It achieved $R^2 = 0.9157$ (train), 0.8825 (validation), and 0.9139 (test). BiLSTM showed superior generalization and the ability to model complex, long-range dependencies, making it highly suitable for seasonal and hourly electric demand patterns.

Strengths:

- Captures both forward and backward temporal dependencies.
- Excellent for complex or long-term demand forecasting.

Limitations:

- High computational cost and training time.
- Requires more careful hyperparameter tuning.

Bidirectional GRU (BiGRU): BiGRU offered a computationally efficient alternative to BiLSTM. With optimized hyperparameters (64 units, dropout = 0.3, learning rate = 0.0005), it delivered comparable results: $R^2 = 0.9188$ (train), 0.8800 (validation), and 0.9133 (test). BiGRU maintained low MAE and MSE across datasets and provided fast, stable training.

Strengths:

- Faster training with performance similar to BiLSTM.
- Efficient memory usage due to simpler architecture.

Limitations:

- May slightly underperform on very long sequences.
- Still requires robust preprocessing and tuning.

Final Insights: All three RNN-based models outperformed traditional forecasting methods in accuracy and adaptability. The ADF test confirmed data stationarity, while ACF and PACF analysis indicated a strong AR(1)-like structure with short-term autocorrelation. This validated the use of sequence models such as RNNs. While mild overfitting was observed—especially in Vanilla RNN—techniques like dropout and early stopping helped mitigate it. Among the models, BiLSTM and BiGRU achieved the highest test performance, with BiGRU offering the best trade-off between forecasting accuracy and computational efficiency.

13 FUTURE SCOPE

While this project focused on forecasting electric demand for the 33/11 kV Godishala substation using historical data, several important extensions can significantly enhance the accuracy, scalability, and applicability of the work. One promising direction is the integration of synthetic city-scale load generation frameworks such as the **pyCity** ecosystem (pycity_base, pycity_calc, and pycity_street). These tools enable the creation of realistic, high-resolution electrical demand profiles for entire cities or districts, even in situations where measured data is limited or unavailable.

13.1 1. Synthetic Data Generation Using pyCity

The pyCity framework allows simulation of hourly electricity demand using:

- Real weather data (via EPW files),
- Building archetypes (residential, commercial, industrial),
- Occupancy models,
- Standard Load Profiles and stochastic appliance models.

In future work, this project can be extended by generating realistic electrical load data for **Gandhinagar city** using pyCity. This would include:

- Extracting building footprints and street networks through OpenStreetMap using pycity_street.
- Assigning building types and demand characteristics.
- Simulating hourly demand for a full year using pycity_calc.
- Exporting the aggregated load curves as CSV for forecasting.

This integration would allow researchers to validate their forecasting models not only on real substation data but also on simulated city-wide datasets.

13.2 2. Digital Twin of Gandhinagar

Using pyCity, a **digital twin** of Gandhinagar can be created. This virtual city model would enable:

- District-level load estimation,
- EV charging impact analysis,
- Rooftop solar adoption scenarios,
- Feeder-level congestion studies,
- Renewable integration planning.

Such simulation capability is extremely valuable for smart grid research and urban energy planning.

13.3 3. Forecasting Under Future Scenarios

Current forecasting is based on historical demand. With pyCity, the project can be extended to simulate:

- Population growth and increased household consumption,
- Seasonal variations with future climate conditions,
- Penetration of electric vehicles,
- Solar rooftop adoption,
- Industrial growth in specific zones.

This allows evaluating how demand may evolve under different policy or infrastructure changes.

13.4 4. Handling Missing or Incomplete Real Data

Real substation datasets often contain:

- Missing values,
- Sensor noise,
- Inconsistent sampling,
- Outage periods.

Synthetic data from pyCity can serve as:

- A benchmark for comparison,
- A filler for missing periods,
- Training data for models requiring long historical sequences.

13.5 5. Multi-Building and Multi-Feeder Forecasting

In the future, forecasting can be extended beyond a single substation to:

- Colony-level load prediction,
- Feeder-level balancing,
- Aggregated demand across wards or districts.

The modular nature of pyCity supports hierarchical modeling, making it possible to forecast at various spatial resolutions.

13.6 Summary

Integrating pyCity into this project will transform it from substation-level forecasting to full **urban energy modelling**. Combining real measured data with high-fidelity synthetic data can significantly improve model robustness and support advanced research in smart grids, renewable integration, and urban planning.

REFERENCES

[1] Load Forecasting Techniques for Power System: Research Challenges and Survey. Available at: https://www.researchgate.net/publication/361688270_Load_Forecasting_Techniques_for_Power_System_Research_Challenges_and_Survey

[2] Electric Load Forecasting Using Machine Learning Techniques: A Review. Available at: https://www.mdpi.com/1996-1073/16/5/2283

[3] A Review on Load Forecasting Techniques for Smart Grid. Available at: https://ijcrt.org/papers/IJCRT2207165.pdf

[4] Short-Term Load Forecasting Using LSTM Based Recurrent Neural Network. Available at: https://ieeexplore.ieee.org/document/9812604

[5] A Review on Forecasting Techniques in Smart Grid. Available at: https://jesit.springeropen.com/articles/10.1186/s43067-020-00021-8

[6] Methods and Models for Electric Load Forecasting: A Comprehensive Review. Available at: https://www.researchgate.net/publication/339403486_Methods_and_Models_for_Electric_Load_Forecasting_A_Comprehensive_Review