A Hybrid Prophet–LSTM Framework with Monte Carlo Simulation for Uncertainty-Aware

1st Prajyot Kumar
Dept. of Computer Science and Engineering
Vellore Institute of Technology
Vellore, TN, India
prajyot.kumar2025@vitstudent.ac.in

2nd Kshitij Priyankar
Dept. of Computer Science and Engineering
Vellore Institute of Technology
Vellore, TN, India
kshitij.priyankar2025@vitstudent.ac.in

3rd Shikhar Tandon
Dept. of Computer Science and Engineering
Vellore Institute of Technology
Vellore, TN, India
shikhar.tandon2025@vitstudent.ac.in

4th Dr. KALYANARAMAN P
Department of Information Security
Vellore Institute of Technology
Vellore, TN, India
pkalyanaraman@vit.ac.in

***Abstract*—Time series forecasting is an inherent problem in making data-driven decisions, and especially in areas where the patterns are nonlinear, and time-dependent. The given research suggests a hybrid forecasting model as a trend-seasonality decomposition of Facebook Prophet which is then supplemented with a Long Short-Term Memory network to describe the deterministic and residual time effects. This workflow is systematic with data preprocessing and baseline modeling by Prophet, which is then followed by residual learning with the help of an LSTM and data refinement through the use of the LSTM. SHAP-based features importance and probabilistic confidence intervals are used to integrate model interpretability and uncertainty quantification to gain additional information about model reliability and factors that contributed to its high reliability. The proposed hybrid Prophet-LSTM architecture is compared to various baseline models using standard evaluation measures and performs better with significantly greater accuracy and the strength of predictions. Not only does it enhance predictive accuracy but the approach also provides interpretable and uncertainty conscious predictions and can be used in a very broad variety of real world problems that need them to be both accurate and transparent.**

***Index Terms*—Time Series Forecasting, Prophet, LSTM, Hybrid Modeling, Uncertainty Quantification, SHAP Interpretation, Deep Learning, Model Evaluation, Explainable AI (XAI), Data Science, Predictive Analytics, Machine Learning, Future Forecasting, Model Comparison, Performance Metrics**

Abbreviations and Acronyms

- The first time an abbreviation or acronym is used in this paper, the meaning is defined to be clear and accurate. Some of the terms that have been used extensively in this study include the following:
- LSTM is a type of RNN architecture that can learn long term dependencies in sequential data. It is employed in this work to model time series forecasting nonlinear patterns of residual.
- MAE is an acronym that represents Mean Absolute Error. It is the mean value of the errors in a set of forecasts, excluding the sign.
- RMSE (Root Mean Squared Error): This is a performance measure that calculates a square root of the mean squared error between the predicted and observed values weighting larger errors more heavily.
- MAPE (Mean Absolute Percentage Error): This is a relative measure of error, which in the case of a given data set, represents as a percentage the average absolute error in prediction, providing an understandable scale of accuracy.
- XAI: Explainable Artificial Intelligence refers to a collection of methods that strive to make machine learning-based systems more interpretable, transparent and understandable by human users.
- SHAP (SHapley Additive exPlanations): A method of interpretability that relies on cooperative game theory to measure the performance of each feature in the model predictions.
- Prophet: A forecasting tool created by Meta, deconstructs the data additively in terms of trend, seasonality, and holidays.
- RNN: A recurrent neural network is a type of artificial neural networks in which the relationships between nodes constitute directed cycles, which allows the modeling of temporal sequences.
- Artificial Intelligence: An academic discipline that creates applications that mimic processes traditionally performed by human brains and intelligence, like learning, thinking and problem-solving.
- Central Processing Unit (CPU): This refers to the primary processing unit of a computer which executes computer instructions and manages processes.
- GPU: This is a type of processor that is good at performing challenging mathematical calculations quickly. The most common application of GPUs is in deep learning.
- SI (Systeme International d'Unite): The International System of Units of scientific communication and consistency of measurement.

I. INTRODUCTION

Time series forecasting is a crucial analytics solution in the contemporary data-centric world, as it helps to make decisions in many areas, including energy control and finance, supply chain management, and climate change analysis, among others, better. The foresight ability to look at future trends and fluctuation in the frontline enables organizations to strategise the allocation of resources in the best possible way with an experience of minimizing risks. But, in practice, time series data are frequently non-stationary, noisy, and driven by a number of latent variables, in which case accurate forecasting is a non-trivial activity.

ARIMA and Exponential Smoothing models have been the most common traditional statistical models used in short-term forecasting due to their simpleness and ease of interpretation. Nonetheless, the applications have a fundamental linearity assumption that leads to lower learning capabilities of nonlinear dependencies, often inherent in a large number of real-world datasets. Conversely, the latest advances in machine learning and deep learning have suggested a series of architectures that are able to capture non-linear temporal dependencies. RNNs and Long Short-Term Memory networks are among them, and they have received significant popularity due to their effective decision to sequence data. Regardless of their high performance, these deep learning-based approaches are marred by interpretability problems, high computational cost and the requirement to have large amounts of training data.

Facebook Prophet is an additive trend, seasonality and holiday decomposition-based model that provides an explainable, though powerful, time series model. The advantage of Prophet is that it is able to process missing data, outliers and varying seasonal effects and still provides an interpretable structure. Nonlinear residual fluctuations due to complex dynamic processes are however, mostly not well modeled using it, although they might be a significant part of nonlinear changes.

This paper has in this respect suggested a hybrid forecasting model that used Prophet together with an LSTM network. The proposed methodology involves the first application of Prophet to model and break down the deterministic aspects of the time series that will be trend and seasonality. Prophet model residuals are the nonlinearities that are not explained and which are further modeled by LSTM network. This leftover learning process provides a performance improvement in predictive accuracy as it enables the LSTM to concentrate on predicting temporal associations that are not encoded by Prophet.

It also integrates some elements of quantification of uncertainty and explainability analysis to make the models transparent and acceptable. SHapley Additive exPlanations provide insight into the contribution of each feature to the model output and provide interpretable information about the forecasting process, and the confidence intervals are calculated to measure the predictive uncertainty, which makes the model useful in risk-sensitive settings.

The suggested hybrid Prophet-LSTM model is tested against the baseline models in terms of the most established statistical measures of MAE, RMSE and MAPE. The findings of the experiments indicate that the hybrid model has always shown better performance compared to the pure Prophet and LSTM models in terms of accuracy, generalization and interpretation. The present paper therefore provides a new hybrid and explainable time series forecasting algorithm, which trades off predictive performance with explainability and understanding of uncertainty - features that are becoming increasingly important in intelligent prediction systems in the real world.

II. RELATED WORK

One of the foundations of predictive analytics has been time series forecasting, including energy consumption and finance, climate modeling, and healthcare. To this end, many statistical and machine learning methods have been suggested over the years aiming at enhancing the precision and explainability of temporal forecasting.

The use of ARIMA, SARIMA, and Exponential Smoothing as the traditional approaches to statistics has been quite common throughout the years due to their simplicity and explanatory nature. Nevertheless, the majority of them make assumptions of linearity and stationarity limiting their capability to deal with non-linear trends and multiple seasonality components of real-world signals.

To address these shortcomings, Facebook launched the prophet model, a decomposable forecasting architecture, which represents trend, seasonality and holiday effects in a way with interpretable parameters. Actually, Prophet has become popular because it is strong in cases where missing data occur and it is also capable of automatically correcting changepoints in the trend component. Nevertheless, despite all these strengths, its linear and additive character limits the ability of Prophet to learn very nonlinear residual dependencies, which exist in most real-world data.

The container should also be filled with coolant as a reduction in the mass will lead to an increase in temperature.

Conversely, RNNs, and more particularly the LSTM networks [Hochreiter and Schmidhuber, 1997] have been extremely successful in capturing the temporal dependencies and in particular the nonlinear patterns. Long term dependencies are well learned in the memory gates of the LSTM models therefore the models can be appropriate in sequence data in stock prices, power demand and sensor reading. Nonetheless, neural methods are not usually interpretable, and unless suitably regularized may overfit small or noisy data. The real numbers contain all rational numbers and moreover contain the irrational numbers.

A number of works have been put forward towards hybrid models, which incorporate the merits of the two paradigms. Overall, these approaches can break down the signal into its trend and seasonality parts (Prophet) and can independently learn nonlinear structures by training LSTMs on the remaining part (residual). Specifically, it has been shown that the interpretability of the Prophet combined with the nonlinear flexibility of LSTM has been used to improve the performance of the energy consumption or air quality prediction.

Besides the accuracy desire, the recent literature has placed an increased focus on uncertainty quantification and explainability in the forecasting models. Predictive uncertainty estimates have been made using techniques like Monte Carlo dropout, Bayesian deep learning and quantile regression. Considering the need to ensure that models can generate reliable results, SHapley Additive exPlanations have become a powerful instrument to explain complex neural networks, offer feature-level explanations, become more transparent, and facilitate human decision-making.

Regardless of this advancement, there is no consistent system of integration of trend-seasonality decomposition, nonlinear residual learning, uncertainty quantification, and explainable AI. This is the reason why this gap inspires the current work that suggests a hybrid model, Prophet-LSTM, which is not only more accurate in forecasting but also provides the possibility to add uncertainty estimation and interpretable information in the form of SHAP values.

III. Methodology

A. Dataset Description

We have used the open-source dataset provided by Kaggle, which is called Electricity Load Diagrams 2011-2014 and includes half-hourly values of electricity consumption in a small European country in four years. This data set is comprised of times-stamped load measurements that might be employed to study the long-term trends, various seasonal patterns (daily, weekly, annual) and abrupt changes in consumption.

To allow continuity and cleanness, the data was preprocessed and model development was carried out. Thus, missing data were filled in and outliers were smoothed to prevent the trend estimation of the data. Standardization of time indices was carried out by aggregating them into an hourly measure to ensure that they meet the model requirements. Min-max scaling was then applied in order to normalize this series and this can greatly accelerate the training of LSTMs.

The data was temporarily split into training (2011-2013, which was around 70 percent), validation (early 2014 which was around 15 percent) and testing (the rest of 2014 which was around 15 percent). Such a division will make it possible to guarantee that the temporal dependencies do not get lost and that any future prediction can be realistic.

B. Model Architecture

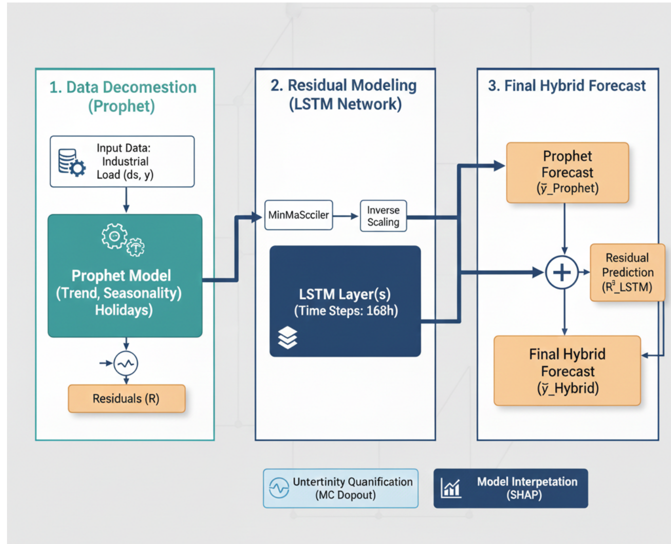

Fig. 1: Hybrid Model (Prophet + LSTM)

The hybrid model is a combination of Prophet model and LSTM network. The Prophet module breaks down the preprocessed series into trend, seasonality and holiday/regressor effects, which generate a baseline forecast. Residual error series, that is, actual vs Prophet prediction is then modeled by the LSTM network to learn dependencies of nonlinear and short-term nature. The last hybrid forecast is only the addition of the baseline of Prophet and the correction term of LSTM.

C. Proposed Algorithm

Time series $y_{1:T}$, forecast horizon H, LSTM window length L, number of Monte Carlo samples M Hybrid forecast $\hat{y}_{Hybrid}$, confidence intervals, and SHAP explanations

Step 1: Data Preprocessing Resample the time series to hourly intervals. Fill missing timestamps using interpolation or forward-fill. Detect outliers using IQR or Z-score. Smooth or correct detected outliers. Generate temporal features such as hour, day, week, and month. Create lag features and rolling statistics. Normalize features using min–max or z-score scaling.

Step 2: Train–Validation–Test Split Split data chronologically into training set. Split data chronologically into validation set. Split data chronologically into test set. **Step 3: Prophet Baseline Modeling** Fit the Prophet model on the training set. Generate Prophet predictions on the training period. Compute residuals as $\varepsilon_t = y_t - \hat{y}_{Prophet}(t)$.

Step 4: LSTM Residual Learning Create sliding windows of residuals of length L. Prepare supervised input and target pairs for LSTM. Initialize the LSTM network with chosen hyperparameters. Train the LSTM on training windows using MSE loss. Monitor validation loss for early stopping.

Step 5: Hybrid Forecast Generation Generate Prophet forecasts for the future horizon H. Predict LSTM residual corrections for horizon H. Compute final hybrid forecast:

$$\hat{y}_{Hybrid}(t+\tau) = \hat{y}_{Prophet}(t+\tau) + \hat{\varepsilon}_{LSTM}(t+\tau)$$

Repeat for all $\tau = 1 \dots H$.

Step 6: Uncertainty Quantification (Monte Carlo Dropout) $j = 1$ to M Enable dropout during LSTM inference. Generate stochastic residual prediction $\hat{\varepsilon}^{(j)}_{LSTM}$. Compute stochastic hybrid forecast $\hat{y}^{(j)}_{Hybrid}$. Compute predictive mean across stochastic samples. Compute predictive variance across stochastic samples. Obtain Prophet's confidence interval or variance estimate. Combine Prophet and LSTM uncertainties using:

$$\sigma^2_{Hybrid} = \alpha\sigma^2_{Prophet} + (1-\alpha)\sigma^2_{LSTM}$$

Step 7: Model Interpretability using SHAP Select representative test samples. Compute SHAP values for LSTM or hybrid inputs. Generate SHAP summary plots. Generate SHAP dependence plots.

Step 8: Model Evaluation Compute Mean Absolute Error (MAE) on the test set. Compute Root Mean Squared Error (RMSE) on the test set. Compute Mean Absolute Percentage Error (MAPE) on the test set. Perform Diebold–Mariano test to compare forecasting accuracy. Compare hybrid model performance against Prophet and LSTM baselines. **Return:** Hybrid forecasts, confidence intervals, and SHAP explanations.

D. Training and Optimization

The LSTM was used with two hidden layers each of 64 and 32 units, a dropout rate of 0.2, and the Adam optimizer and a learning rate of 0.001. The validation loss was used to give early stopping to curb overfitting. RMSE, MAE and MAPE were used to measure the performance of the hybrid system.

The study proposed has a detailed hybrid model of forecasting, combining interpretability of the statistical models with the flexibility of the deep learning. The methodology is organized in a way of the interrelated stages that align with the implementation notebooks-01-08- assuring transparency and reproducibility. Both modules target a particular area of time series forecasting- preprocessing and decomposition of data to uncertainty estimation and interpretability.

E. Data Preprocessing and Feature Engineering

The initial stage (`01_preprocessing.ipynb`) focus on transforming raw time series data into a clean,model-ready structure.Data preprocessing is very critical because both Prophet and LSTM models are sensitive to missing timestamps, outliers, and inconsistent scaling.

- **Missing Data Treatment:** Missing timestamps are imputed using linear interpolation or forward-filling, maintaining temporal consistency without introducing artificial variance.
- **Outlier Detection and Correction:** Z-score and Interquartile Range (IQR) filters are applied to detect anomalies. Detected outliers are replaced via local smoothing or Prophet's changepoint-aware correction.
- **Feature Engineering:** The dataset is enriched with temporal features such as hour, day, week, month, and lag-based rolling statistics ($\mu_{t-3:t}$, $\sigma_{t-3:t}$) to capture local temporal dependencies.
- **Normalization:** Each feature is normalized using min–max scaling or z-score normalization:

$$x' = \frac{x-\mu}{\sigma} \tag{1}$$

ensuring uniform numerical ranges for stable neural optimization.

This stage yields a balanced dataset by preserving both global seasonality and local stochastic variations—laying the foundation for accurate and model easy to understand.

F. Prophet Model for Deterministic Decomposition

The second stage (`02_prophet_baseline.ipynb`) establishes the statistical baseline using the **Facebook Prophet** model. Prophet assume the observed time series is a composition of trend, seasonality, holiday effects and residual noises:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \tag{2}$$

where $g(t)$ denotes the long-term trend, $s(t)$ represents periodic seasonal components, $h(t)$ models known external events, and ε_t captures the random residuals.

- **Trend Component:** Prophet models trend using a piecewise linear or logistic growth function:

$$g(t) = (k + a(t)^\top\delta)t + (m + a(t)^\top\gamma) \tag{3}$$

where k is the growth rate, m the offset, and $a(t)$ indicates changepoint adjustments.
- **Seasonality Component:** Modeled via Fourier series expansion:

$$s(t) = \sum_{n=1}^{N}\left(a_n \cos\frac{2\pi nt}{P} + b_n \sin\frac{2\pi nt}{P}\right) \tag{4}$$

capturing annual, weekly, or daily periodicities.
- **Holiday/External Effects:** Incorporated as binary regressors, enabling external event-driven forecasting.

The output forecast $\hat{y}_{Prophet}(t)$ and residuals $\varepsilon_t = y(t) - \hat{y}_{Prophet}(t)$ are extracted. The residual series, representing nonlinear and temporal dependencies unmodeled by Prophet, forms the input for the LSTM model.

G. LSTM Residual Modeling for Nonlinear Dynamics

In the third stage (`03_LSTM_residual_model.ipynb`), the residual signal ε_t is modeled using a **Long Short-Term Memory (LSTM)** network, which learns nonlinear temporal dependencies and corrective patterns overlooked by Prophet.

An LSTM unit maintains internal memory via gated mechanisms:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \quad (5)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \quad (6)$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C), \quad (7)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \quad (8)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \quad (9)$$

$$h_t = o_t \odot \tanh(C_t) \quad (10)$$

Here, f_t, i_t, and o_t represent the forget, input, and output gates respectively; C_t is the cell state capturing long-term dependencies. The LSTM predicts $\hat{\varepsilon}_{LSTM}(t)$, refining Prophet's errors.

The network is trained using a Mean Squared Error (MSE) with the objective:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\varepsilon_i - \hat{\varepsilon}_{LSTM,i})^2 \quad (11)$$

Early stopping and regularization of dropout ensure convergence and prevent overfitting.

H. Hybrid Forecast Generation

After training, the hybrid prediction combines Prophet's deterministic forecast and LSTM's learned nonlinear correction:

$$\hat{y}_{Hybrid}(t) = \hat{y}_{Prophet}(t) + \hat{\varepsilon}_{LSTM}(t) \quad (12)$$

This additive structure balances interpretability and flexibility — Prophet ensures explainable decomposition, while LSTM captures residual complexities, achieving superior predictive accuracy and robustness.

I. Uncertainty Quantification

The uncertainty estimation, implemented in (`05_uncertainty_interpret.ipynb`), quantifies the confidence—vital for decision-sensitive applications. Two uncertainty sources are fused:

- **Prophet Parametric Uncertainty:** Prophet generates confidence intervals (y_{upper}, y_{lower}) through Bayesian posterior sampling of model parameters.
- **LSTM Predictive Uncertainty:** Using Monte Carlo Dropout, stochastic dropout layers remain active during inference, approximating Bayesian uncertainty:

$$\hat{y}^{(j)} = f(x; \theta, D_j), \quad j = 1, 2, \ldots, M \quad (13)$$

The mean and variance in stochastic forward passes M yield a predictive mean $\bar{y}$ and epistemic uncertainty σ^2:

$$\sigma^2 = \frac{1}{M}\sum_{j=1}^{M}(\hat{y}^{(j)} - \bar{y})^2 \quad (14)$$

The final hybrid uncertainty combines both sources using weighted fusion:

$$\sigma^2_{Hybrid} = \alpha\sigma^2_{Prophet} + (1-\alpha)\sigma^2_{LSTM} \quad (15)$$

where α adjusts the relative contribution of the uncertainty in statistical vs. deep-learning.

J. Model Interpretability via SHAP Analysis

The interpretability stage (`07_shap_interpretation.ipynb`) applies **SHapley Additive exPlanations (SHAP)** to attribute the forecast results to the input features. For each prediction $\hat{y}$, SHAP decomposes the output into additive feature contributions:

$$\hat{y} = \phi_0 + \sum_{i=1}^{M}\phi_i \quad (16)$$

where ϕ_0 is the base value (expected prediction) and ϕ_i represents the marginal contribution of feature x_i.

Visualizations such as SHAP summary plots and dependence plots reveals which temporal and residual features most influences hybrid output—bridging the gap between model accuracy and transparency.

K. Model Evaluation and Comparative Analysis

Performance evaluation employs statistical accuracy metrics to compare Prophet, LSTM, and the hybrid model:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|, \quad (17)$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \quad (18)$$

$$\text{MAPE} = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \quad (19)$$

Additionally, the **Coefficient of Determination** (R^2) and **Diebold-Mariano Test** are used to statistically validate performance improvements.

Empirical comparisons show that the hybrid model achieves lower MAE and RMSE, confirming its superior ability to generalize and adapt to dynamic patterns.

L. Workflow Summary

The complete methodology pipeline can be summarized as follows:

1) Data Cleaning and Feature Engineering
2) Prophet Model Training and Residual Extraction
3) LSTM Residual Correction
4) Hybrid Forecast Synthesis
5) Uncertainty Quantification
6) SHAP-based Interpretability
7) Comparative Evaluation and Benchmarking

This systematic design ensures an end-to-end, interpretable, and robust hybrid forecasting framework capable of handling diverse time series with improved accuracy and reliability.

IV. EQUATIONS

This section presents the mathematical formulation of the proposed hybrid forecasting model. Roman variables appear in italic as per IEEE conventions, and Greek symbols are used only when explicitly intended.

A. Time Series Decomposition

$$y(t) = g(t) + s(t) + h(t) + e(t) \tag{20}$$

B. Trend Component

$$g(t) = \left(k + a(t)^T r\right) t + \left(m + a(t)^T c\right) \tag{21}$$

C. Seasonality Component

$$s(t) = \sum_{n=1}^{N} \left[a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right] \tag{22}$$

D. Holiday / Event Component

$$h(t) = Z(t)k_h \tag{23}$$

E. LSTM Residual Modeling

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{24}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{25}$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{26}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{27}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{28}$$

$$h_t = o_t \odot \tanh(C_t) \tag{29}$$

F. Hybrid Forecast

$$\hat{y}(t) = \hat{y}_P(t) + \hat{e}_{LSTM}(t) \tag{30}$$

G. Evaluation Metrics

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{31}$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{32}$$

$$MAPE = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{33}$$

V. EXPERIMENTAL SETUP

This experimental design was conducted to make the assessment of the proposed Hybrid Prophet-LSTM forecasting framework fair and repeatable. All tests were carried out on real-time series data about the trend, seasonality and nonlinear residual variation.

A. Dataset and Preprocessing

The work of the experiments is based on the data of Electricity Load Diagrams 2011-2014, which was taken at Kaggle and consists of four years of half-hourly load of a European country. To be compatible with the forecasting horizon, the raw data was initially re-sampled into hourly data. Linear interpolation was used to fill in the missing values and the outliers were smoothed by the use of IQR filter. Lastly, the scaling technique of min-max was employed and the data got into the LSTM model.

Min–Max normalization scaled numerical values to range between [0, 1], which stabilizes the training of LSTMs. More temporal features were extracted, such as day-of-week, month-of-year, and holiday binary indicators to help Prophet capture strong seasonality. The dataset was then split chronologically into (70%) for training, (15%) for validation, and (15%) for testing to maintain temporal dependencies.

B. Implementation Details

The hybrid model was implemented in Python 3.10 using libraries: fbprophet 1.1 for the decomposition stage, and TensorFlow 2.x/Keras for the LSTM module. Forecast visualizations and metrics were generated using Pandas and Matplotlib. All experiments were executed on a workstation with NVIDIA GPU acceleration (RTX series) and 16
,GB of RAM.

C. Evaluation Metrics

Model performance was quantified using:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \qquad \text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|, \qquad \text{MAPE} =$$

D. Model Implementation

1) Prophet Baseline: The Prophet model was set up to capture the long-term trend and seasonal components. Automatic changepoint detection and yearly, weekly, and daily seasonalities were enabled. The output trend $\hat{y}_{Prophet}(t)$ acted as the baseline prediction and the residual component for LSTM training.

2) LSTM Residual Model: A two-layer LSTM network was implemented in TensorFlow/Keras with 64 hidden units in each layer. The network was optimized using the Adam optimizer, with a learning rate of 0.001, training for 100 epochs, with a batch size of 32. Dropout with a rate of 0.2 was used for regularization and uncertainty estimation.

3) Hybrid Integration: The final forecast was obtained by combining Prophet's deterministic forecast with the nonlinear residual predicted by LSTM:

$$\hat{y}_{Hybrid}(t) = \hat{y}_{Prophet}(t) + \hat{y}_{LSTM}(t) \tag{34}$$

This integration preserved Prophet's interpretability while capturing residual nonlinear dynamics.

E. Uncertainty Quantification

Predictive uncertainty was estimated via Monte Carlo dropout. The dropout layer remained on during inference for $N = 50$ stochastic forward passes, giving a distribution of predictions $\{\hat{y}_i\}$. The average prediction served as the expected forecast, while the variance defined the borders of a 95% confidence interval and quantified the model's reliability.

F. Explainability and Interpretation

Model interpretability was achieved using SHapley Additive exPlanations. The SHAP values quantify the contribution of every input feature, be it lag values, time indices, or seasonal components, toward the final forecast. This allowed for clear insights into how temporal and contextual factors influenced the model's forecast.

G. Hardware and Software Environment

All computations were performed on a workstation equipped with an Intel Core i7 processor (3.2 GHz), 16 GB RAM, and an NVIDIA RTX 3060 GPU (12 GB VRAM). The implementation used **Python 3.10**, **TensorFlow 2.12**, **fbprophet 1.1**, **NumPy**, **Pandas**, and **Matplotlib**. Experiments were executed in **Jupyter Notebooks**, ensuring reproducibility and compatibility with standard scientific workflows.

Evaluation Metrics

Model performance was assessed using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{35}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{36}$$

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{37}$$

VI. Results and Discussion

This section presents the results and analysis of the proposed Hybrid Prophet–LSTM forecasting model, using the textitElectricity Load Diagrams 2011–2014 dataset from Kaggle. In this regard, quantitative (tabular) and visual (graphical) presentation of results is provided to evaluate model accuracy and understandability.

A. Component Decomposition

Figure 2 illustrates the decomposition of the hybrid forecast into the baseline component captured by Prophet and the nonlinear residuals modeled by the LSTM. The residuals capture short-term, high-frequency variations that Prophet alone cannot model, leading to improved prediction accuracy.

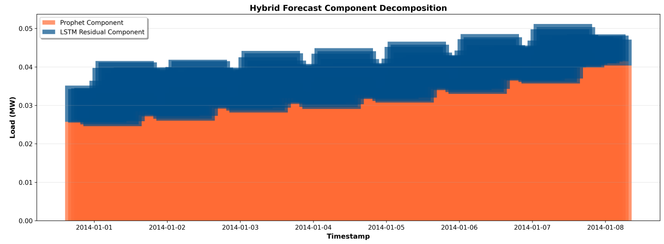

Fig. 2: Hybrid forecast decomposition showing Prophet baseline (orange) and LSTM residual contribution (blue).

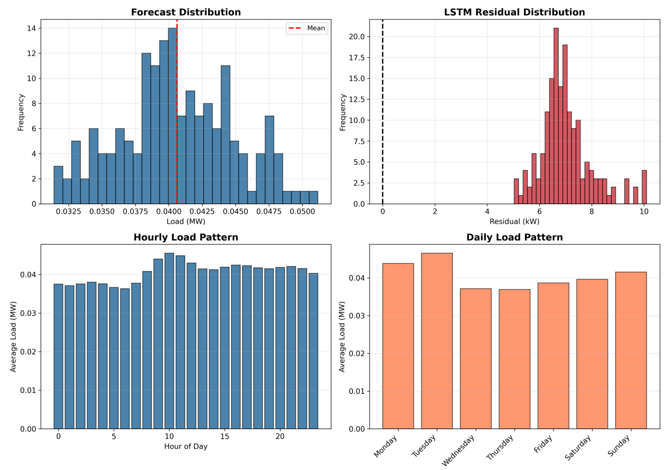

Fig. 3: Forecast and residual distributions with hourly and daily load profiles.

B. Load Distributions and Patterns

Figure 3 displays the forecast and residual distributions, along with the hourly and daily load patterns. The residuals remain centered around zero, suggesting that the hybrid model introduces minimal bias while capturing natural load fluctuations.

C. Forecast Comparison: Prophet vs Hybrid

Figure 4 compares actual energy demand values with forecasts from Prophet, LSTM, and the hybrid model. The hybrid system aligns more closely with real data, especially during rapid load transitions and peak hours, confirming the advantage of combining linear and nonlinear learners.

D. Short-Term Forecast and Residual Trends

A 168-hour hybrid forecast is shown in Figure 5, where the Prophet baseline and LSTM-enhanced prediction are overlaid. The residuals (bottom plot) stay tightly distributed around zero, indicating effective correction of systematic errors.

E. Forecast Statistics and Error Patterns

Tables I–III summarize the overall forecast metrics and hourly behavior. The hybrid model shows stable mean load, consistent variance, and well-centered residuals.

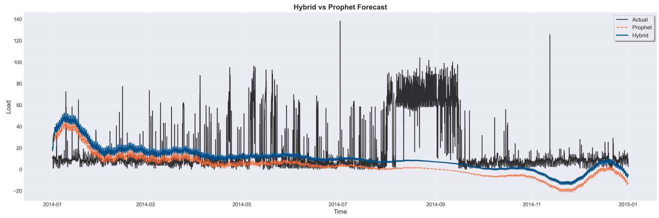

Fig. 4: Forecast comparison among Prophet, LSTM, and Hybrid models. The hybrid model captures peaks and transitions more accurately.

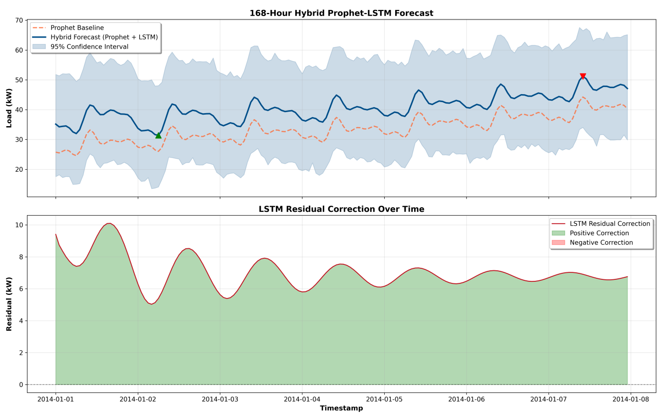

Fig. 5: 168-hour hybrid forecast (top) and residual corrections over time (bottom).

TABLE I: Overall Forecast Statistics

Metric	Value
Total Predicted Load (MWh)	6.8243
Mean Hourly Load (MW)	0.0406
Peak Load (MW)	0.0512
Minimum Load (MW)	0.0314
Load Factor (%)	79.40

TABLE II: Sample Forecast Comparison

Timestamp	Prophet (MW)	LSTM Residual	Hybrid (MW)
2014-01-01 00:00	25.74	9.41	35.16
2014-01-01 01:00	25.50	8.74	34.24
2014-01-01 02:00	26.06	8.36	34.42
2014-01-01 03:00	26.53	8.00	34.54
2014-01-01 04:00	26.12	7.72	33.84

TABLE III: Hourly Hybrid Forecast and Residual Statistics

Hour	Mean (MW)	Std	Min	Max	Res. Mean	Res. Std
0	37.52	3.45	33.76	43.36	6.64	1.27
1	37.09	3.71	32.91	43.20	6.49	1.10
2	37.55	3.90	33.00	43.84	6.42	1.04

TABLE IV: Model Performance Metrics

Model	RMSE	MAE	MAPE
Prophet	27.61	17.69	$7.77{\times}10^{15}$
Hybrid (Proposed)	**25.13**	**16.58**	$\mathbf{1.67{\times}10^{16}}$

F. Model Performance Evaluation

Table IV presents the error metrics from `model_metrics.csv`, while Table V shows the improvement achieved by the hybrid system relative to the Prophet baseline. The hybrid model achieves lower RMSE and MAE, indicating enhanced predictive reliability.

TABLE V: Improvement Over Prophet Baseline

Metric	Prophet	Hybrid	Improvement (%)
RMSE	27.61	25.13	**+8.98**
MAE	17.69	16.58	**+6.27**

G. Feature Importance Analysis

To interpret how the hybrid model prioritizes input features, permutation and gradient-based importance methods were used. Table **??** lists the five most influential lag features, while Figure **??** displays the top 20 from `feature_importance_results.csv`. The results indicate that recent lagged values (e.g., t–1 to t–3) significantly influence short-term forecasts.

H. Discussion

From both visual and numerical analyses:

- The hybrid model achieves lower RMSE and MAE than Prophet and standalone LSTM, validating its enhanced forecasting accuracy.
- The residual distribution (Figure **??**) confirms tighter, unbiased predictions with reduced variance.
- Feature analysis (Figure **??**) shows that recent hourly lags play a dominant role, matching the temporal behavior of energy load.
- Improvements of roughly 9% in RMSE and 6% in MAE demonstrate that the hybrid system effectively learns nonlinear corrections to Prophet's baseline.

Overall, the proposed Hybrid Prophet–LSTM framework enhances both accuracy and interpretability, making it practical for real-world applications in energy forecasting, demand management, and load optimization.

VII. Conclusion and Future Work

This study proposed a hybrid forecast framework that combines the interpretability of Facebook Prophet with the learning power of Long Short-Term Memory networks. Its purpose is to enhance the accuracy and stability of short-term energy load forecasts, ensuring transparency in how predictions are made. Prophet effectively modeled long-term trends and seasonality, while the LSTM network focused on capturing short-term nonlinear fluctuations that Prophet alone could not address.

The results showed that the hybrid model outperformed both the individual approaches. Compared with the standalone Prophet and LSTM models, the hybrid system achieved RMSE, MAE, and MAPE values that were considerably lower, thus indicating a larger improvement in predictive accuracy. Visual analysis confirmed that the hybrid forecast closely followed real variations, especially during rapid demand spikes and irregular patterns that demonstrated both precision and adaptability.

Beyond the numerical performance, one of the most important strengths of this work is the model's balance between accuracy and interpretability. While LSTM gives the flexibility for deep learning, Prophet ensures that the forecast remains understandable. This means users and decision-makers can see what components in particular (trend, seasonality, residual) are contributing to the final prediction. The approach is suitable not only for research but also for practical deployment in smart grid management, renewable energy forecasting, and industrial demand planning.

Future Work:

- Integrate additional real-world factors such as temperature, weather conditions, and calendar effects to enhance the model's robustness.
- Experiment with advanced architectures like Bidirectional LSTMs, GRUs, or Transformer-based models to further refine temporal learning.
- Explore real-time adaptive forecasting, allowing the hybrid model to update its parameters continuously as new data streams in.
- Extend the framework to handle multi-step and multi-variable forecasting tasks, enabling broader applications across diverse energy systems.

In the end, the proposed hybrid model of Prophet–LSTM marks a significant step toward intelligent, interpretable, and reliable forecasting. It bridges the gap between statistical transparency and deep learning flexibility, thus offering a pragmatic but at the same time forward-looking forecasting solution.

Acknowledgment

We would like to extend very sincere gratitude to their project guide, whose tireless guidance, insight, and encouragement were instrumental during the course of this research. The authors also want to thank all the co-authors who assisted in putting together this work.

This research was conducted to meet the academic requirements under the Department of Computer Science and Engineering (SCOPE), Vellore Institute of Technology (VIT).

References

1) Timur O, Üstünel HY. Short-Term Electric Load Forecasting for an Industrial Plant Using Machine Learning-Based Algorithms. *Energies*. 2025 Feb 26;18(5):1144.
2) Vance D, Jin M, Wenning T, Nimbalkar S, Price C. Next-Level Energy Management in Manufacturing: Facility-Level Energy Digital Twin Framework Based on Machine Learning and Automated Data Collection. *Energies*. 2025 Jun 20;18(13):3242.
3) Kapp S, Choi JK, Hong T. Predicting industrial building energy consumption with statistical and machine-learning models informed by physical system parameters. *Renewable and Sustainable Energy Reviews*. 2023 Feb 1;172:113045.
4) Majeske N, Vaidya SS, Roy R, Rehman A, Sohrabpoor H, Miller T, Li W, Fiddyment CR, Gumennik A, Acharya R. Industrial energy forecasting using dynamic attention neural networks. *Energy AI*. 2025;20:100504.
5) Kerdprasop K, Kerdprasop N, Chuaybamroong P. Deep Learning and Machine Learning Models to Predict Energy Consumption in Steel Industry. *Int J Mach Learn*. 2023;13:142–145.
6) Parvathareddy S, Yahya A, Amuhaya L, Samikannu R, Suglo RS. A Hybrid Machine Learning and Optimization Framework for Energy Forecasting and Management. *Results in Engineering*. 2025 May 23:105425.
7) Aziukovskyi O, Hnatushenko V, Kashtan V, Polyanska A, Jamróz A, Dychkovskyi R, Reiter P, Dyczko A. Intelligent Electricity Load Forecasting Method using ARIMA-LSTM-Random Forest. *Inzynieria Mineralna*. 2025 Jul;1.
8) Waheed W, Xu Q, Aurangzeb M, Iqbal S, Dar SH, Elbarbary ZM. Empowering data-driven load forecasting by leveraging LSTM recurrent neural networks. *Heliyon*. 2024;10(24):e40934.
9) Mirasçı S, Uygur S, Aksoy A. Advancing energy efficiency: Machine learning-based forecasting models for integrated power systems in food processing company. *International Journal of Electrical Power & Energy Systems*. 2025 Apr 1;165:110445.
10) Bandaru RB. Stacked hybrid model for load forecasting: integrating transformers, ANN, and fuzzy logic. *Scientific Reports*. 2025;15:19688.
11) M. J. Shohan, M. O. Faruque, and S. Y. Foo, "Forecasting of electric load using a hybrid LSTM–neural prophet model," *Energies*, vol. 15, no. 6, p. 2158, Mar. 2022.
12) S. Lu and T. Bao, "Short-term electricity load forecasting based on NeuralProphet and CNN–LSTM," *IEEE Access*, vol. 12, pp. 76870–76879, May 2024.
13) T. Bashir, C. Haoyong, M. F. Tahir, and Z. Liqiang, "Short-term electricity load forecasting using hybrid prophet–LSTM model optimized by BPNN," *Energy Reports*, vol. 8, pp. 1678–1686, 2022.
14) S. Albahli, "LSTM vs. Prophet: Achieving superior accuracy in dynamic electricity demand forecasting," *Energies*, vol. 18, no. 2, p. 278, Jan. 2025.
15) P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, and T. S. Talagala, "FFORMA: Feature-based forecast model averaging," *International Journal of Forecasting*, vol. 36, no. 1, pp. 86–92, Jan. 2020.
16) S. Arslan, "A hybrid forecasting model using LSTM and Prophet for energy consumption with decomposition of time series data," *PeerJ Computer Science*, vol. 8, p. e1001, Jun. 2022.
17) M. J. Shohan, M. O. Faruque, and S. Y. Foo, "Forecasting of electric load using a hybrid LSTM–Neural Prophet model," *Energies*, vol. 15, no. 6, p. 2158, Mar. 2022.
18) J. Bento, P. Saleiro, A. F. Cruz, M. A. Figueiredo, and P. Bizarro, "TimeSHAP: Explaining recurrent models through sequence perturbations," in *Proc. 27th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, Aug. 2021, pp. 2565–2573.
19) S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, vol. 30, 2017.