A Hybrid Ensemble Empirical Mode Decomposition–Temporal Convolutional Network Framework with LightGBM Residual Correction for Day-Ahead Electric Load Forecasting

Trung Dung Nguyen, Tuan Anh Nguyen
FEET, Industrial University of Ho Chi Minh City (IUH), Ho Chi Minh City, Viet Nam

ARTICLE INFORMATION

Article History:

Received 08 June 2026
Revised 04 August 2026
Accepted 17 September 2026

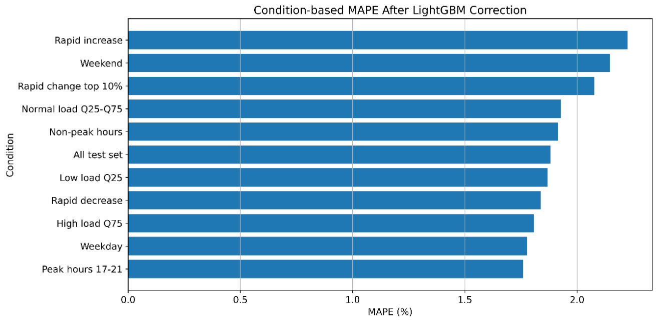
Keywords:

Day-Ahead Load Forecasting;
Ensemble Empirical Mode Decomposition;
LightGBM;
Residual Correction;
Temporal Convolutional Network

Corresponding Author:

Tuan Anh Nguyen,
FEET, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Viet Nam.
Email: nguyentrungdung@iuh.edu.vn.

ABSTRACT



Accurate day-ahead load forecasting is essential for generation scheduling, reserve allocation, electricity-market operation, and reliable power-system management. However, half-hourly electricity demand is nonlinear and non-stationary, with short-term fluctuations, daily and weekly periodicity, and slowly varying trends occurring simultaneously. This study proposes a unified hybrid framework that combines ensemble empirical mode decomposition, a temporal convolutional network, and validation-calibrated LightGBM residual correction for direct day-ahead forecasting. The main contribution is the coordinated use of multi-scale signal representation, direct multi-output temporal learning, and controlled residual correction within a single forecasting pipeline. Ensemble empirical mode decomposition separates the original load series into nine intrinsic mode functions and one residue, which are organized as a synchronized ten-channel input. The temporal convolutional network then maps a seven-day historical window of 336 half-hourly observations directly to the subsequent 48 load values, avoiding recursive error propagation. LightGBM is subsequently trained to estimate structured residuals using the base forecast, forecast horizon, calendar variables, historical lags, rolling statistics, and decomposition-derived features. To limit overcorrection, the residual model is trained on an earlier validation subset, while a separate chronological calibration subset selects the correction coefficient, which was 0.30 in the reported experiment. Experiments using Queensland electricity-demand data from 2015 to 2019 yielded an RMSE of 156.92 MW, an MAE of 116.48 MW, a MAPE of 1.8825%, a WAPE of 1.8746%, and an R^2 of 0.9692 for the proposed framework. Under the stored offline evaluation setting, these error values were lower than those reported for the standalone LightGBM and raw-series temporal convolutional network baselines. Because the baseline pipelines contain closely matched but not completely identical test-origin sets, the comparisons are interpreted descriptively rather than as fully paired statistical evidence. In addition, EEMD is applied offline to the complete series; therefore, the reported results represent a decomposition-assisted benchmark rather than a strictly causal real-time forecasting implementation. The findings suggest that multi-scale representation, temporal convolution, and calibrated residual learning can provide complementary benefits under the considered experimental setting.

Document Citation:

T. D. Nguyen and T. A. Nguyen, “A Hybrid Ensemble Empirical Mode Decomposition–Temporal Convolutional Network Framework with LightGBM Residual Correction for Day-Ahead Electric Load Forecasting,” *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 8, no. 5, pp. 1346-1373, 2026, DOI: 10.12928/biste.v8i5.17013.

1. INTRODUCTION

Accurate short-term load forecasting (STLF) is essential for the secure, reliable, and economical operation of modern power systems. In particular, day-ahead load forecasts support unit commitment, generation scheduling, reserve allocation, electricity-market bidding, maintenance planning, and demand-side management [1][2]. Forecasting errors can directly affect operational decisions. Underestimation may result in insufficient reserve capacity and increased system risk, whereas overestimation may lead to unnecessary generation commitment and higher operating costs. The importance of forecasting accuracy is especially evident at the day-ahead horizon, where system operators require the complete future load trajectory rather than a single predicted value. For half-hourly electricity-demand data, this task involves predicting 48 consecutive load values representing the subsequent 24-hour period.

Despite its operational importance, day-ahead load forecasting remains challenging because electricity demand is nonlinear, non-stationary, and governed by multiple interacting temporal mechanisms. At short time scales, the load series contains abrupt ramps, local fluctuations, and transient variations. At intermediate time scales, intraday and weekly patterns arise from working schedules, consumer behavior, and recurring human activities. At longer time scales, seasonal effects and gradual structural changes influence the underlying demand level. These temporal components are superimposed in the observed load series, making it difficult for a single forecasting model to represent all relevant behaviors simultaneously. Forecasting difficulty may also vary according to the prediction horizon, load level, weekday or weekend status, peak and non-peak periods, and the occurrence of rapid load changes.

Early STLF studies were primarily based on traditional statistical techniques, including linear regression [3]-[7], exponential smoothing [8]-[11], autoregressive models [12][13], autoregressive integrated moving average (ARIMA) models [14]-[18], seasonal ARIMA models [19], and related time-series approaches. These methods remain attractive because of their relatively simple structures, low computational requirements, and ease of interpretation. They can perform effectively when the underlying electricity-demand process is approximately linear, stationary, and stable. However, their performance may deteriorate when the load series contains strong nonlinearities, complex temporal interactions, abrupt changes, or evolving operating conditions. Their reliance on predefined statistical assumptions may also restrict their ability to represent heterogeneous temporal patterns.

To overcome these limitations, machine-learning methods have been widely introduced into STLF. Representative approaches include support vector regression [20]-[24], decision trees [25]-[27], random forests [28]-[31], gradient boosting [32]-[36], XGBoost [37]-[42], and LightGBM [43]-[47]. These models can capture nonlinear relationships among historical demand, calendar variables, rolling statistics, and other engineered predictors without requiring restrictive assumptions regarding the underlying data distribution. Tree-based ensemble methods are particularly effective for heterogeneous tabular features and can efficiently represent complex nonlinear interactions.

Among these methods, LightGBM is attractive because of its strong nonlinear approximation capability and computationally efficient tree-building mechanism. It can effectively exploit recent demand values, daily and weekly lags, rolling statistics, and calendar-related information. However, the performance of feature-based machine-learning approaches remains highly dependent on the quality and relevance of the engineered predictors. In multi-step day-ahead forecasting, separate models are also frequently trained for individual forecast horizons. Such a strategy increases model proliferation and prevents the learner from developing a shared representation of the complete future load trajectory.

Deep-learning methods provide an alternative approach by learning temporal representations directly from sequential data. Recurrent neural networks [48]-[50], long short-term memory networks [51]-[55], gated recurrent units [56]-[59], convolutional neural networks [60]-[64], and temporal convolutional networks [65]-[69] have all been applied to electricity-load forecasting. Recurrent models such as LSTM and GRU can retain information over extended temporal intervals, but their sequential computation may increase training time and limit parallelization. CNN-based models provide efficient local pattern extraction; however, conventional convolutional architectures may require careful structural design to capture long-range temporal dependencies.

Temporal convolutional networks provide a particularly suitable architecture for long-window sequence modeling. Through temporal convolution, dilation, and residual connections, a TCN can enlarge its receptive field and capture short- and long-range dependencies while maintaining efficient parallel computation [70]-[73]. For day-ahead forecasting, the TCN can be connected to a multi-output prediction layer so that all future horizons are generated simultaneously. This direct forecasting strategy avoids the accumulation of errors associated with recursively feeding previous predictions into subsequent forecasting steps. Nevertheless, when a TCN is trained directly on the original load series, the same network must simultaneously model rapid

Vol. 8, No. 5, October 2026, pp. 1346-1373

fluctuations, intraday periodicity, weekly recurrence, and slower demand variations within a single latent representation.

Signal decomposition provides a complementary mechanism for addressing this challenge. Empirical mode decomposition adaptively separates a nonlinear and non-stationary signal into intrinsic mode functions and a residual component, whereas ensemble empirical mode decomposition improves decomposition stability through noise-assisted repeated decomposition and ensemble averaging. By separating the original demand series into components corresponding to different temporal scales, EEMD can provide a more explicit representation of high-frequency fluctuations, intermediate oscillations, and low-frequency variations [74]-[77].

Many decomposition-based forecasting methods train an independent predictive model for each decomposed component and subsequently reconstruct the final forecast by summing the component-level predictions. Although straightforward, this strategy may require a large number of separate submodels and may not explicitly capture interactions among different temporal scales. An alternative multi-channel strategy is to supply all decomposed components simultaneously to a single forecasting model. This design reduces model proliferation and allows the downstream learner to capture cross-scale relationships within a unified temporal representation.

The present study adopts this multi-channel strategy. The original Queensland electricity-demand series is decomposed into nine intrinsic mode functions and one residual component, resulting in ten synchronized input channels. These components are jointly processed by a TCN using a seven-day historical window containing 336 half-hourly observations. The network directly predicts all 48 load values of the following day in a single forward pass. In this framework, EEMD is responsible for constructing an explicit multi-scale representation of the historical signal, while the TCN learns the direct mapping from the decomposed seven-day history to the complete next-day load trajectory.

However, decomposition and deep temporal modeling may not eliminate all systematic forecasting errors. The remaining residuals may still depend on the forecast horizon, time of day, recent load level, daily and weekly lags, rolling statistical behavior, or the current state of the decomposed components. Treating these residuals as purely random noise may therefore discard useful predictive information. A secondary learner can instead be trained to identify predictable error patterns and refine the base forecast.

For this purpose, the proposed framework employs LightGBM as a residual-correction model. Unlike the standalone LightGBM baseline, which directly predicts future load values, the correction model estimates only the residual remaining after the EEMD-TCN base forecast. Its inputs include the base prediction, forecast-horizon variables, recent and lagged load values, rolling statistics, calendar information, and decomposition-derived features available at the forecast origin. This separation of responsibilities allows the TCN to focus on the primary temporal forecasting task, while LightGBM learns structured deviations that remain after the first forecasting stage.

A further methodological challenge is that unrestricted residual correction may overadjust an already reasonable base forecast. The proposed framework therefore separates residual-model training from correction-strength calibration. LightGBM is trained using the earlier portion of the validation period, while a separate chronological calibration subset is used to select the correction coefficient. The final prediction is obtained by adding a calibrated fraction of the predicted residual to the EEMD-TCN base forecast. This procedure reduces the risk of excessive correction and ensures that test labels are not used to train the residual model or determine the correction magnitude.

Motivated by these considerations, this study develops a three-stage day-ahead forecasting framework integrating EEMD-based multi-scale representation, direct TCN forecasting, and validation-calibrated LightGBM residual correction. The framework is evaluated using half-hourly Queensland electricity-demand data collected from 2015 to 2019. A seven-day input window is used to predict the complete 48-step load profile of the following day. The proposed framework is compared with a standalone LightGBM model and a raw-series TCN baseline. The principal contributions of this study are summarized as follows.

First, a multi-channel EEMD-TCN architecture is developed in which nine intrinsic mode functions and one residual component are jointly supplied to a single TCN. Unlike conventional component-wise forecasting approaches, the proposed architecture does not require an independent forecasting model for each decomposed component and allows interactions among different temporal scales to be learned directly.

Second, the TCN performs direct multi-output day-ahead forecasting by mapping a seven-day historical window of 336 half-hourly observations to the complete 48-step next-day load trajectory. This strategy avoids recursive error propagation and enables the model to learn a shared representation of the entire forecasting horizon.

Third, a LightGBM residual-correction stage is introduced to model structured errors remaining after the EEMD-TCN base forecast. Rather than independently reconstructing the future load trajectory, the correction model uses the base prediction, forecast horizon, temporal information, historical lags, rolling statistics, and decomposition-derived features to estimate the remaining residual.

Fourth, residual-model training and correction-strength calibration are separated chronologically. The residual model is trained on one portion of the validation period, while a separate calibration subset is used to select the correction coefficient. This design prevents test-label information from influencing the residual-learning and calibration stages.

Fifth, the proposed framework is examined against feature-based machine-learning and deep temporal-learning baselines using overall forecasting accuracy, horizon-wise behavior, error distributions, condition-based performance, and model-specific diagnostic analyses. These evaluations provide a descriptive assessment of the model behavior under the stored offline experimental setting. Because the evaluated pipelines contain slightly different valid test-origin sets, the reported comparisons are not interpreted as a fully paired statistical evaluation.

Finally, the study explicitly distinguishes between offline decomposition-assisted benchmarking and strictly causal operational forecasting. It also reports that the standalone LightGBM and TCN-based pipelines contain slightly different valid test-origin sets. These limitations are considered when interpreting the numerical comparisons, which are treated as descriptive evidence within the reported experimental configuration rather than as fully paired operational validation.

The remainder of this paper is organized as follows. Section II reviews statistical, machine-learning, deep-learning, decomposition-based, and residual-correction methods for short-term load forecasting. Section III presents the proposed EEMD-TCN-LightGBM methodology. Section IV describes the dataset, experimental settings, forecasting configuration, and evaluation methods. Section V presents and discusses the experimental results. Finally, Section VI concludes the study and outlines directions for future research.

2. RELATED WORK

Short-term load forecasting has been extensively investigated using statistical, machine-learning, deep-learning, decomposition-based, and hybrid modeling approaches. Traditional forecasting methods include regression, autoregressive models, exponential smoothing, and autoregressive integrated moving average techniques. These approaches remain attractive because of their relatively simple structures, low computational requirements, and interpretability [78][79]. They can provide satisfactory performance when electricity-demand patterns are approximately linear and stable. However, their reliance on predefined statistical assumptions may limit their ability to represent nonlinear, non-stationary, and rapidly changing load behavior.

Machine-learning methods provide greater modeling flexibility by formulating load forecasting as a supervised learning problem. Support vector regression, random forests, gradient boosting, XGBoost, and LightGBM have been applied using historical demand, lagged variables, rolling statistics, calendar information, and other exogenous predictors [80]-[81]. These methods can efficiently model nonlinear relationships, particularly when temporal information is represented through carefully engineered tabular features. LightGBM is especially suitable for this setting because of its computationally efficient tree-construction mechanism and its ability to process heterogeneous predictors. Nevertheless, the accuracy of feature-based models remains strongly dependent on feature selection and engineering. Moreover, multi-horizon day-ahead forecasting may require an independent estimator for each future step, increasing the number of models and limiting the ability to learn a shared representation of the complete future trajectory.

Deep-learning approaches reduce dependence on manually engineered predictors by learning temporal representations directly from sequential inputs. Recurrent neural networks, long short-term memory networks, gated recurrent units, convolutional neural networks, and related hybrid architectures have therefore been widely investigated for load forecasting. Recurrent models can capture long-term temporal dependencies, but their sequential computation may increase training cost and restrict parallel processing. Convolutional approaches provide efficient local feature extraction; however, conventional architectures may require careful structural design to represent dependencies spanning long historical windows [82][83].

Temporal convolutional networks provide an alternative architecture for long-window sequence modeling. Through temporal convolution, dilation, and residual connections, TCNs can expand their receptive fields and capture both short- and long-range dependencies while retaining efficient parallel computation. They are also well suited to direct multi-output forecasting, in which the complete future trajectory is generated in a single forward pass. This strategy avoids recursive error propagation and enables the model to learn relationships among multiple forecast horizons. However, when a TCN is applied directly to a raw load sequence, the network must simultaneously represent short-term fluctuations, intraday periodicity, weekly recurrence, and slowly varying demand patterns within a single input channel.

Signal-decomposition techniques have consequently been combined with forecasting models to provide more explicit representations of nonlinear and non-stationary time series. Empirical mode decomposition adaptively separates a signal into intrinsic mode functions and a residual component, whereas ensemble empirical mode decomposition improves decomposition stability through noise-assisted repeated decomposition and ensemble averaging. Related approaches, including complete ensemble empirical mode decomposition with adaptive noise and variational mode decomposition, have also been applied to energy forecasting. These techniques aim to separate complex load behavior into components associated with different temporal scales before prediction [84]-[86].

Decomposition-based forecasting methods generally adopt either a component-wise or a multi-channel modeling strategy. In component-wise approaches, each decomposed component is predicted using an independent model, and the resulting forecasts are subsequently summed to reconstruct the final load prediction. Although this design simplifies the forecasting task assigned to each learner, it may require numerous component-specific models, increase computational and maintenance requirements, and overlook interactions among temporal scales.

In a multi-channel strategy, all decomposed components are instead supplied simultaneously to a single forecasting model. This approach reduces model proliferation and enables the downstream learner to capture interactions among high-frequency fluctuations, intermediate oscillations, and low-frequency trends. However, decomposition and multi-channel temporal modeling alone may still leave structured errors associated with the forecast horizon, time of day, recent demand level, temporal variability, or the current decomposition state.

Residual learning provides a complementary mechanism for exploiting such remaining error structure. After a primary model produces a base forecast, a secondary learner can be trained to estimate the difference between the observed and predicted values. The predicted residual is then used to refine the original forecast. Residual-correction models may employ autoregressive, neural-network, or ensemble-learning techniques. Nevertheless, an unrestricted correction model may overadjust already accurate predictions, particularly when the secondary learner is highly flexible or when the correction magnitude is selected using the same data employed for residual-model training.

The existing literature therefore demonstrates the individual value of signal decomposition, deep temporal forecasting, and residual learning. However, these mechanisms are frequently implemented as separate modeling strategies. Decomposition-based methods often employ multiple component-specific predictors, deep sequence models commonly process the original load signal directly, and residual-learning methods do not necessarily exploit an explicit multi-scale representation of the demand series. In addition, residual-model training and correction-strength selection are not always separated chronologically, which may increase the risk of overly optimistic evaluation.

The present study integrates these three research directions within a unified forecasting pipeline. Ensemble empirical mode decomposition is first used to construct a synchronized multi-scale representation of the original load series. Nine intrinsic mode functions and one residual component are then supplied jointly to a single temporal convolutional network, which directly predicts the complete 48-step day-ahead load trajectory. LightGBM is subsequently employed only as a residual-correction model rather than as a second independent load forecaster. Its role is to estimate structured deviations remaining after the EEMD-TCN base prediction.

A separate chronological calibration subset is further used to determine the correction magnitude. This design distinguishes residual-model learning from correction-strength selection and prevents test observations from influencing either stage. Accordingly, the research gap addressed in this study is not the absence of EEMD, TCN, or residual learning individually, but the lack of their coordinated use in a direct multi-output framework with controlled residual correction.

Based on this gap, the study addresses three principal questions within an offline benchmark setting. First, it examines the forecasting behavior obtained when synchronized multi-channel EEMD inputs are combined with a TCN, relative to a raw-series TCN baseline. Second, it investigates whether the residuals of the EEMD–TCN base forecast retain patterns that can be modeled using the available historical, temporal, and decomposition-derived features. Third, it evaluates the effect of validation-calibrated LightGBM residual correction on the stored forecast outputs. These questions motivate the methodology presented in the following section, while the operational implications remain subject to causal decomposition and fully aligned evaluation in future work.

3. PROPOSED METHOD

This section presents the proposed hybrid framework for direct day-ahead short-term load forecasting. The methodology integrates three complementary stages: EEMD-based multi-scale decomposition, TCN-based temporal forecasting, and LightGBM-based residual correction. Rather than assigning the entire forecasting task to a single model, the framework separates signal representation, primary sequence prediction, and forecast-error correction. EEMD first transforms the original load series into multiple synchronized components associated with different temporal scales. These components are then jointly processed by a TCN to generate the complete next-day load trajectory. Finally, LightGBM models the predictable residual remaining in the TCN forecast, while a validation-based calibration mechanism controls the magnitude of the final correction.

Let y_t denote the observed electrical load at half-hourly time index t. Because the data are sampled every 30 min, each complete day contains 48 observations. The forecasting task uses the most recent seven days of historical information to predict all 48 half-hourly load values of the following day. Accordingly, the historical input length is $L = 336$, while the forecasting horizon is $H = 48$.

For a forecast origin t, the historical input sequence is defined as

$$\mathbf{x}_t = [y_{t-L+1}, y_{t-L+2}, \dots, y_t]^{\mathrm{T}}, L = 336 \tag{1}$$

where $\mathbf{x}_t$ contains only load observations available up to and including the forecast origin t. The corresponding day-ahead target vector is:

$$\mathbf{Y}_t = [y_{t+1}, y_{t+2}, \dots, y_{t+H}]^{\mathrm{T}}, H = 48 \tag{2}$$

where $\mathbf{Y}_t$ contains the complete load trajectory of the following 24 h.

The forecasting task is formulated as direct multi-output prediction. Therefore, all 48 future load values are generated simultaneously rather than recursively predicting one step at a time. This formulation avoids repeatedly feeding earlier predictions back into the model and therefore eliminates the mechanism through which recursive forecast errors can accumulate across the prediction horizon.

In the proposed framework, however, the raw historical sequence $\mathbf{x}_t$ is not directly supplied to the final forecasting model. The complete process consists of three successive stages. First, EEMD decomposes the original load signal into multiple intrinsic mode functions and a residual component. Second, these components are aligned as synchronized channels and supplied jointly to a TCN, which produces the complete 48-step base forecast. Third, LightGBM estimates the structured residual remaining in the base prediction, and a validation-selected coefficient controls the magnitude of the final correction.

The overall forecasting process can therefore be summarized as follows: Historical load → EEMD multi-scale decomposition → Multi-channel TCN forecasting → LightGBM residual correction → Final day-ahead forecast.

The following subsections describe the individual modeling stages and their integration into the complete proposed framework.

3.1. EEMD-Based Multi-Scale Decomposition

The first stage of the proposed framework uses ensemble empirical mode decomposition to reveal the multi-scale temporal structure contained in the original electricity-demand series. Electrical load is nonlinear and non-stationary and may simultaneously exhibit short-term fluctuations, repeated daily and weekly patterns, and slower variations in the overall demand level. When these behaviors are mixed within a single raw sequence, the forecasting model must learn several temporal dynamics from the same representation. EEMD is therefore introduced as a preprocessing and representation stage before temporal forecasting.

EEMD extends empirical mode decomposition by repeatedly adding independent white-noise realizations to the original signal and applying EMD to each perturbed sequence. Let $y(t)$ denote the original load signal. For the m-th ensemble trial, the perturbed sequence is defined as

$$y_m(t) = y(t) + \varepsilon w_m(t), m = 1, \dots, M \tag{3}$$

where $w_m(t)$ is an independent white-noise realization, M denotes the number of ensemble trials, and ε controls the noise amplitude.

EMD is then applied independently to each perturbed signal. The corresponding intrinsic mode functions obtained from all ensemble trials are averaged to form the final EEMD components. The original load signal can be represented as

$$y(t) = \sum_{k=1}^{K} c_k\,(t) + r(t) \tag{4}$$

where $c_k(t)$ denotes the k-th intrinsic mode function and $r(t)$ represents the residual component.

The extracted components describe load behavior at different characteristic scales. Earlier IMFs generally capture faster oscillations and short-term fluctuations, whereas later IMFs represent progressively slower variations. The residual component describes the remaining low-frequency structure. EEMD therefore transforms the original one-dimensional signal into a set of synchronized components that provide different temporal views of the same underlying load process.

In the present study, the EEMD procedure uses 30 ensemble trials and a noise width of 0.05. The resulting decomposition contains nine intrinsic mode functions and one residual component. Thus, the load state at time $t$is represented by the ten-dimensional vector

$$\mathbf{z}_t = [c_1(t), c_2(t), \dots, c_9(t), r(t)]^{\mathrm{T}} \in \mathbb{R}^{10} \tag{5}$$

For each forecast origin, the component vectors from the previous 336 half-hourly observations are aligned chronologically to form the historical multi-channel input

$$\mathbf{Z}_t \in \mathbb{R}^{336\times 10} \tag{6}$$

where the first dimension corresponds to the seven-day historical window and the second dimension corresponds to the nine IMFs and the residual component.

The proposed framework uses this multi-channel representation rather than forecasting the decomposition components independently. In conventional component-wise approaches, a separate predictive model is often trained for each IMF and the resulting component forecasts are then summed. Although such a strategy is straightforward, it increases the number of models and may not explicitly capture interactions among different temporal scales.

In contrast, all EEMD components in the proposed method are supplied jointly to a single TCN. This design allows the network to learn both the temporal evolution of individual components and the relationships among components associated with different scales. For example, a short-term fluctuation represented by a high-frequency IMF may have different forecasting implications depending on the slower operating state reflected by later IMFs or the residual component.

The EEMD stage is therefore used only to construct a structured multi-scale representation of the historical load series. It does not independently generate the future load forecast. The actual mapping from the decomposed seven-day history to the complete next-day load trajectory is learned by the multi-channel TCN described in the following subsection.

In the present implementation, EEMD is applied once to the complete load series before the forecasting windows are constructed. Consequently, the decomposed representation is generated under an offline setting and is not strictly causal, because the decomposition of an earlier observation may be influenced by later observations in the complete series. The reported results should therefore be interpreted as an offline decomposition-assisted benchmark designed to examine the behavior of the integrated modeling architecture.

Accordingly, the numerical results are not presented as evidence of real-time operational performance. A deployment-oriented implementation would require EEMD to be recalculated using only information available up to each forecast origin, for example through rolling-window or expanding-window decomposition. Such causal decomposition may produce different components and forecasting performance from those reported in the present study. This distinction is retained throughout the interpretation of the experimental results.

The output of this stage is therefore a 336×10multi-channel historical representation, which is subsequently processed by the TCN to generate the direct 48-step day-ahead base forecast.

3.2. Multi-Channel TCN Forecasting Model

After the original load series has been transformed into ten synchronized EEMD components, the next stage learns the relationship between the multi-scale historical representation and the complete next-day load trajectory. A temporal convolutional network is used as the primary forecasting model because it can capture temporal dependencies over long historical windows through convolutional operations and an expanded receptive field.

For each forecast origin t, the TCN receives the multi-channel input

$$\mathbf{Z}_t \in \mathbb{R}^{336\times 10} \tag{7}$$

where the temporal dimension contains the previous 336 half-hourly observations and the feature dimension contains the nine IMFs and the residual component.

Before model training, each decomposition channel is standardized independently. Let $z_{t,k}$ denote the value of the k-th component at time t. The standardized value is calculated as

$$\tilde{z}_{t,k} = \frac{z_{t,k} - \mu_k}{\sigma_k} \tag{8}$$

where μ_k and σ_k are the mean and standard deviation of component k, respectively. For the reported offline benchmark, these parameters are estimated from the combined model-development period, comprising the training and validation observations, and are subsequently applied unchanged to the test period. No test observation is used to estimate the scaling parameters. This procedure is therefore described as development-set normalization rather than strictly training-only normalization. The target load is standardized using the same principle and transformed back to the original MW scale after prediction.

The TCN extracts temporal features from the standardized multi-channel sequence using dilated temporal convolutions. A generic dilated convolution can be expressed as

$$\mathbf{h}(t) = \sum_{i=0}^{q-1} \mathbf{W}_i\, \mathbf{x}(t - di) \tag{9}$$

where q is the kernel size, $d$denotes the dilation factor, and $\mathbf{W}_i$ represents the learnable convolutional weights.

Increasing the dilation factor allows the receptive field to expand without requiring a proportional increase in network depth. This property is particularly suitable for the present forecasting task because the seven-day historical window contains short-term persistence, repeated daily behavior, and weekly recurrence.

The implementation uses the default TCN() layer provided by the keras-tcn library. The study does not introduce a separate hyperparameter-optimization procedure for the TCN architecture. The purpose is to examine the contribution of EEMD-based multi-scale representation and residual correction around a standard temporal-convolutional forecasting model.

The output of the TCN is connected to a fully connected layer containing 48 units. The resulting base forecast is represented as

$$\hat{\mathbf{Y}}_t^{\text{base}} = \left[\hat{y}_{t+1}^{\text{base}}, \hat{y}_{t+2}^{\text{base}}, \dots, \hat{y}_{t+48}^{\text{base}}\right]^{\mathrm{T}} \in \mathbb{R}^{48} \tag{10}$$

All 48 future load values are therefore generated in a single forward pass. This direct multi-output strategy allows the TCN to learn a shared representation of the complete next-day load profile and avoids recursive error propagation.

The model is trained using mean squared error and the Adam optimizer. The batch size is set to 32 and the maximum number of training epochs is 50. Early stopping monitors validation loss with a patience of eight epochs and restores the best-performing model weights. In addition, the learning rate is reduced by a factor of 0.5 after four consecutive validation epochs without improvement.

The resulting EEMD-TCN prediction serves as the base forecast of the proposed framework. Although the multi-channel representation allows the TCN to exploit information across different temporal scales, systematic errors may still remain. The next stage therefore models the predictable structure contained in these residuals.

3.3. LightGBM Residual Correction

The EEMD-TCN model is responsible for producing the main day-ahead forecast. However, its prediction errors may still exhibit systematic patterns associated with the forecast horizon, recent load behavior, time of day, load level, or the current state of the decomposed signal. The proposed framework therefore introduces LightGBM as a secondary learner to model the remaining residual.

For forecast origin $t$and horizon h, the residual of the EEMD-TCN base forecast is defined as

$$e_{t,h} = y_{t+h} - \hat{y}_{t+h}^{\text{base}}, h = 1, \dots, 48 \tag{11}$$

A positive residual indicates underestimation by the base model, whereas a negative residual indicates overestimation. If the residual were purely random, a secondary correction model would not be expected to provide a systematic improvement. Residual learning therefore examines whether the EEMD-TCN has left predictable error structure unmodeled.

For each forecast point, a structured correction feature vector is constructed using only information available at the forecast origin or known deterministically for the target timestamp. No future observed load value is included.

The correction features contain several complementary types of information. The base forecast and its squared value describe the predicted load magnitude and allow the correction model to identify load-dependent bias. Forecast-horizon variables describe the lead time and enable the model to learn horizon-specific error behavior.

Historical load information includes the load at the forecast origin and lags at 1, 2, 48, 96, and 336 half-hourly steps. These predictors represent immediate persistence, one- and two-day recurrence, and weekly recurrence. One-day and one-week rolling means and standard deviations are also included to describe the recent load level and variability.

Calendar features characterize both the forecast origin and the target timestamp and include hour, minute, day of week, month, weekend status, and cyclical temporal encodings. Target-time calendar variables are permitted because they are known in advance and do not contain future load information.

The current EEMD state is also included through the values of the nine IMFs and the residual component at the forecast origin. These predictors allow the correction model to determine whether forecast errors are associated with a particular multi-scale condition of the load signal.

LightGBM then learns the mapping from the correction features to the EEMD-TCN residual:

$$\hat{e}_{t,h} = g_{\boldsymbol{\psi}}(\boldsymbol{\phi}_{t,h}) \tag{12}$$

where $\boldsymbol{\phi}_{t,h}$denotes the correction feature vector and $g_{\boldsymbol{\psi}}$represents the LightGBM model.

The implementation uses the default LGBMRegressor() configuration. This choice keeps the correction stage simple and avoids introducing a separate hyperparameter-optimization problem. Importantly, LightGBM is not used as a second independent load forecaster. Its role is restricted to predicting the residual left by the EEMD-TCN base forecast.

Applying the entire predicted residual may lead to excessive correction. To control this effect, the validation period is divided chronologically into two subsets. The first 70% of validation origins are used to train the residual model, while the remaining 30% are reserved for correction calibration.

Candidate correction coefficients are evaluated over the set

$$\mathcal{A} = \{0, 0.05, 0.10, \dots, 1.00\} \tag{13}$$

For a given coefficient α, the corrected prediction is calculated as

$$\hat{y}_{t+h}^{\text{final}} = \hat{y}_{t+h}^{\text{base}} + \alpha \hat{e}_{t,h} \tag{14}$$

The coefficient that produces the lowest MAPE on the calibration subset is retained and subsequently applied unchanged to the test period. In the stored experiment, the selected value was $\alpha^* = 0.30$.

This calibration mechanism prevents the predicted residual from being applied automatically at full strength. It also ensures that the correction magnitude is selected without using test labels. The resulting corrected predictions form the final output of the complete proposed framework.

3.4. Proposed EEMD-TCN-LightGBM Framework

The proposed framework integrates EEMD, TCN, and LightGBM into a sequential day-ahead forecasting architecture. Each stage performs a distinct but complementary function. EEMD provides multi-scale signal representation, the TCN performs the primary temporal forecasting task, and LightGBM corrects predictable residual errors that remain after the base prediction has been generated.

The first stage transforms the original seven-day historical load sequence into a ten-channel representation containing nine IMFs and one residual component. Instead of forecasting these components independently, the proposed model aligns them as synchronized channels and supplies them jointly to the TCN.

The second stage maps the resulting 336×10input matrix directly to the complete 48-step next-day load trajectory. The EEMD-TCN base prediction can be represented as

$$\hat{\mathbf{Y}}_t^{\text{base}} = f_{\boldsymbol{\theta}}(\mathbf{Z}_t) \tag{15}$$

where $f_{\boldsymbol{\theta}}$denotes the TCN forecasting model and $\mathbf{Z}_t$is the multi-channel EEMD input.

The third stage estimates the remaining residual using LightGBM. The predicted residual is then scaled by the validation-selected coefficient before being added to the base forecast. The complete final output is therefore expressed as

$$\hat{\mathbf{Y}}_t^{\mathrm{final}} = \hat{\mathbf{Y}}_t^{\mathrm{base}} + \alpha^* \hat{\mathbf{E}}_t \tag{16}$$

where $\hat{\mathbf{E}}_t$ contains the 48 predicted residuals and $\alpha^*$is the correction coefficient selected on the calibration subset.

The complete forecasting process can therefore be summarized as follows:

Historical load → EEMD decomposition → Multi-channel representation → TCN base forecast → LightGBM residual prediction → Calibrated correction → Final day-ahead forecast.

The proposed architecture differs from conventional component-wise decomposition methods because the IMFs are not assigned independent forecasting models. Instead, all components are processed jointly by a single TCN. This allows the model to learn cross-scale relationships while avoiding the need to train and maintain multiple component-specific neural networks.

The framework also differs from conventional forecast ensembles. LightGBM does not produce an independent load forecast that is averaged with the TCN output. Instead, it explicitly estimates the residual of the base forecast. Its role is therefore corrective rather than generative.

A further characteristic of the proposed framework is the separation between primary forecasting, residual learning, and correction calibration. The TCN is trained using the training period. Its validation predictions are then used to construct residual targets. The earlier part of the validation period is used to train LightGBM, while the later part is used only to select the correction coefficient. The test period is evaluated only after all model parameters and the correction coefficient have been fixed.

The three modeling stages therefore address different aspects of the forecasting problem. EEMD addresses the difficulty of representing a non-stationary signal containing multiple temporal scales. The TCN addresses the direct prediction of the complete next-day trajectory from a long historical sequence. LightGBM addresses the structured forecast error that remains after the primary temporal model has been applied.

By assigning these functions to separate but connected stages, the proposed framework avoids requiring a single model to simultaneously perform signal decomposition, long-range temporal learning, and residual correction. The complete method is therefore based on complementary task specialization rather than simply increasing the complexity of one forecasting model.

The resulting architecture produces one final point forecast for each of the 48 half-hourly horizons. The next subsection describes the complete training, calibration, and forecasting procedure used to implement the proposed framework.

3.5. Overall Algorithm and Flowchart

Figure 1 presents the complete workflow of the proposed EEMD–TCN–LightGBM framework for direct 48-step day-ahead load forecasting. The proposed method follows a sequential hybrid architecture consisting of three complementary modeling stages. First, ensemble empirical mode decomposition is used to transform the original electricity-demand series into multiple temporal components. Second, a temporal convolutional network generates the primary day-ahead forecast from the resulting multi-channel representation. Third, LightGBM estimates the structured residual errors remaining in the EEMD–TCN forecast. A validation-selected calibration coefficient is then applied to control the contribution of the predicted residual and reduce the risk of excessive correction. The complete procedure includes data preparation, chronological data partitioning, EEMD decomposition, multi-channel sample construction, TCN training, residual-model training, correction-strength calibration, test-set forecasting, final residual correction, and performance evaluation.

As shown in Figure 1, the forecasting procedure begins with the collection and preparation of QLD1 electricity-demand data recorded at 30-minute intervals from 2015 to 2019. The observations are arranged chronologically and examined for invalid timestamps, missing target values, and duplicated records. After data cleaning, the complete series is divided chronologically into training, validation, and test periods without random shuffling. This chronological organization prevents the direct use of test targets during TCN training, residual-model fitting, and correction-coefficient calibration. However, because EEMD is applied offline to the complete load series, the resulting decomposition is not strictly causal, as discussed in Section 3.1.

The cleaned load series is subsequently decomposed using ensemble empirical mode decomposition. This stage separates the original nonlinear and non-stationary signal into nine intrinsic mode functions and one residual component. The resulting components represent load variations occurring at different temporal scales, ranging from short-term fluctuations to slower changes in the underlying demand level. Rather than predicting each component independently, the decomposed signals are aligned chronologically and combined to form a synchronized ten-channel representation. This design allows the forecasting model to process the different temporal components jointly and learn interactions among multiple time scales.

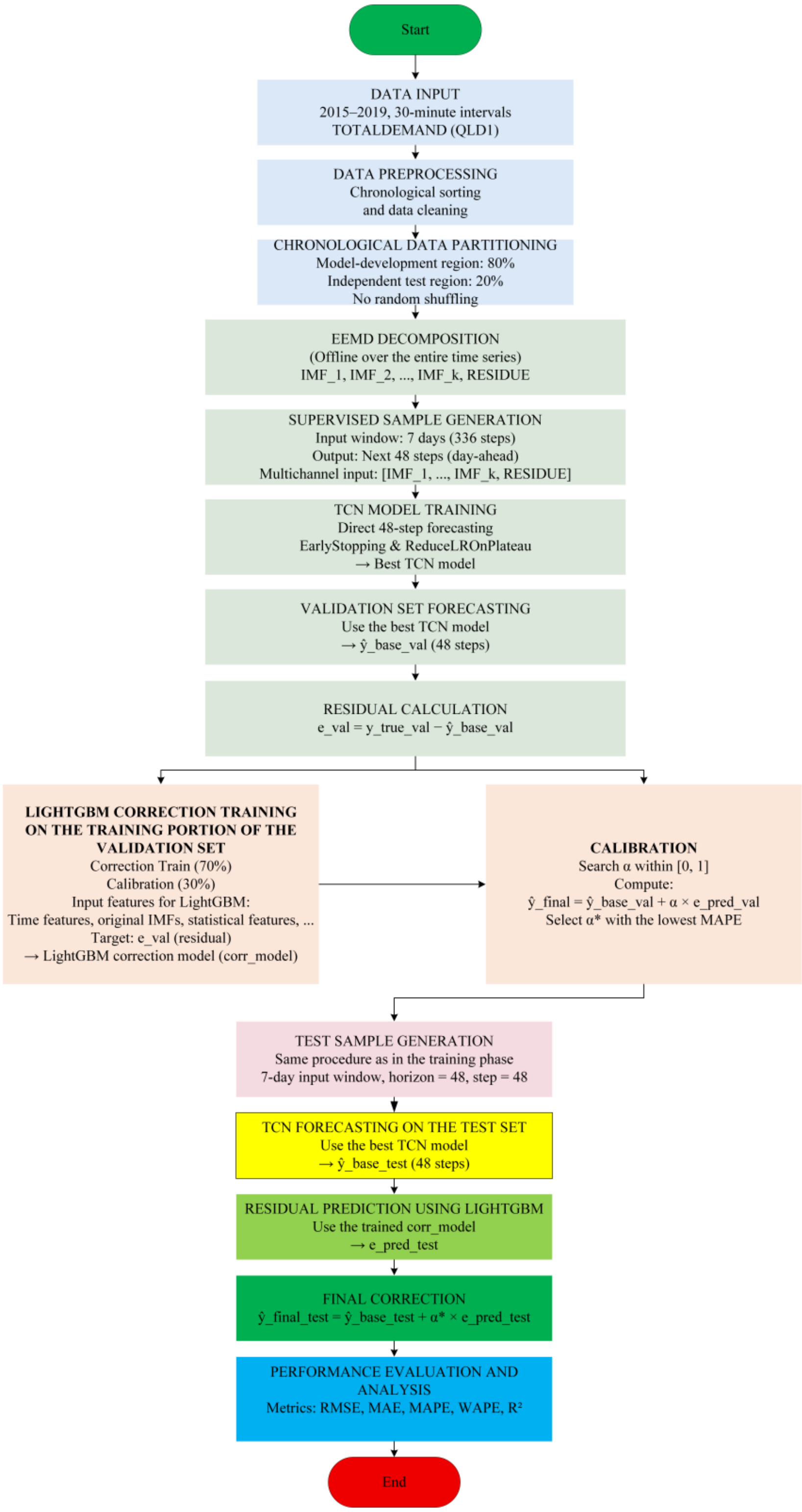


Figure 1. Overall workflow of the proposed EEMD–TCN–LightGBM framework for direct 48-step day-ahead electric load forecasting

4. EXPERIMENTAL SETUP

This section describes the experimental design used to evaluate the proposed EEMD–TCN–LightGBM framework. The forecasting task is formulated as direct day-ahead prediction using half-hourly electricity-demand data from Queensland, Australia. Each forecasting sample uses a fixed seven-day historical window to predict the complete 48-step load profile of the following day. To preserve temporal ordering, all data partitions are constructed chronologically without random shuffling. This partitioning prevents test targets from being used directly in model fitting or calibration. Nevertheless, because EEMD is applied offline to the complete series, the proposed experiment is interpreted as a decomposition-assisted benchmark rather than a strictly causal forecasting pipeline. The test forecasts are generated using non-overlapping daily windows. The following subsections describe the dataset and preprocessing procedure, forecasting configuration, computational environment and reproducibility settings, and performance-evaluation methods.

4.1. Dataset and Data Preparation

The experiments use electricity-demand data from Queensland, Australia. Five annual data files covering the period from 2015 to 2019 are combined to construct the complete study series. The forecasting target is the TOTALDEMAND variable for the QLD1 region, while the corresponding observation time is provided by the SETTLEMENTDATE field.

The original data are recorded at 30-minute intervals, resulting in 48 scheduled observations per day. The complete study period contains 1,826 calendar days and 87,648 scheduled half-hourly intervals before data cleaning. The original temporal resolution is retained throughout the study because the forecasting objective is to predict the complete intraday demand profile at its native half-hourly resolution.

The five annual data files are first concatenated and arranged in chronological order. The SETTLEMENTDATE field is converted to a datetime format, and TOTALDEMAND is converted to a numeric variable. Only observations associated with the

QLD1 region are retained. Records containing invalid timestamps or invalid target values are removed. When duplicate timestamps are identified, the last available record is retained to ensure that each timestamp is represented by a single demand value.

The preprocessing procedure is intentionally conservative. No temporal aggregation, resampling, smoothing, outlier clipping, or interpolation is applied to the target series. Consequently, the observed demand variability, including rapid changes and extreme values present in the original records, is preserved. This design reduces the possibility that the reported forecasting improvements are caused by additional smoothing or outlier-removal procedures rather than by the evaluated forecasting models.

All preprocessing operations are completed before the supervised forecasting samples are generated. The cleaned observations remain chronologically ordered, and no random reordering is applied at any stage. Model inputs and targets are subsequently constructed using historical and future observations according to their temporal positions. Statistics required for model normalization are estimated from the model-development period and then applied unchanged to the independent test period. The principal dataset and preprocessing settings are summarized in Table 1. After preprocessing, the cleaned chronological series is used to construct the historical input windows and corresponding day-ahead targets according to the forecasting configuration described in the following subsection.

Table 1. Dataset and data preparation settings

Item	Setting
Region	Queensland (QLD1)
Study period	2015–2019
Target variable	TOTALDEMAND
Timestamp field	SETTLEMENTDATE
Sampling interval	30 min
Scheduled observations per day	48
Calendar days	1,826
Scheduled observations before cleaning	87,648
Temporal aggregation	None
Resampling	None
Smoothing	None
Outlier clipping	None
Interpolation	None
Data ordering	Chronological
Random shuffling	None

4.2. Forecasting Configuration

The experiments are formulated as a direct day-ahead load-forecasting task. At each forecast origin, the models use the electricity-demand observations from the preceding seven days to predict the complete load profile of the subsequent 24 h. Because the data are recorded at 30-minute intervals, the historical input length is fixed at ($L=336$), corresponding to seven days of observations, while the forecasting horizon is fixed at ($H=48$), corresponding to the 48 half-hourly load values of the following day.

The TCN-based models employ a direct multi-output forecasting strategy. Accordingly, all 48 future load values are generated simultaneously in a single forward pass rather than recursively. This strategy avoids repeatedly feeding earlier predictions into subsequent forecasting steps and therefore prevents the accumulation of recursive forecasting errors across the day-ahead horizon.

During test evaluation, the interval between two consecutive forecast origins is set to ($S=48$). Because one complete day contains 48 half-hourly observations, consecutive test origins are separated by 24 h. Consequently, the resulting 48-step day-ahead forecasting windows do not overlap.

The dataset is divided chronologically, and random train–test shuffling is not applied. For the TCN-based pipelines, the earliest 80% of valid forecast origins constitute the model-development region, whereas the latest 20% constitute the independent test region. Within the model-development region, the final 10% of forecast origins are reserved for validation, while the remaining earlier origins are used for TCN training. This chronological partitioning preserves the temporal order of the demand series and ensures that observations from the test period are not used during model development.

Under the implemented TCN-based sampling procedure, the dataset contains 62,831 training origins, 6,981 validation origins, and 364 non-overlapping test origins. Because each test origin generates 48 forecasts, the TCN-based evaluation contains a total of 17,472 predicted test points.

For the proposed framework, the validation period is further divided chronologically into two non-overlapping subsets. The earlier 70% of the validation origins are used to train the LightGBM residual-correction model, whereas the later 30% are reserved exclusively for correction-coefficient calibration. This division produces 4,886 residual-training origins and 2,095 calibration origins. Separating these subsets prevents the same validation observations from being used both to train the residual model and to determine the correction magnitude.

Three forecasting systems are evaluated. The first is a standalone LightGBM baseline that uses engineered historical-demand, lagged, rolling-statistical, and calendar features. A separate default LGBMRegressor model is trained for each of the 48 forecasting horizons.

The second system is a raw-series TCN baseline that receives a (336x1) historical input sequence and directly predicts the subsequent 48 load values. The single input channel contains the original electricity-demand series.

The third system is the proposed EEMD–TCN–LightGBM framework. Its TCN receives a (336x10) multi-channel input consisting of nine intrinsic mode functions and one residual component obtained through EEMD. The TCN first generates the complete 48-step base forecast. LightGBM subsequently estimates the structured residual remaining in this prediction, and the validation-selected correction coefficient controls the contribution of the estimated residual to the final forecast. The principal configurations of the evaluated forecasting systems are summarized in Table 2.

Table 2. Forecasting configurations of the evaluated models

Setting	LightGBM baseline	TCN baseline	Proposed model
Input representation	Engineered historical and calendar features	Raw load sequence	EEMD components
Input dimension	Horizon-specific tabular input	(336x1)	(336x10)
Historical information	Lagged and rolling features	Seven days	Seven days
Forecast horizon	48 steps	48 steps	48 steps
Forecast period	24 h	24 h	24 h
Forecasting strategy	48 horizon-specific models	Direct multi-output	Direct multi-output
Main learner	LGBMRegressor	TCN + Dense (48)	TCN + Dense (48)
Residual correction	None	None	LightGBM
Test-origin step size	48	48	48
Test windows	Non-overlapping	Non-overlapping	Non-overlapping

Because the standalone LightGBM pipeline constructs lagged and rolling predictors before removing incomplete feature rows, its stored evaluation contains 360 non-overlapping test origins. In contrast, the TCN-based pipelines contain 364 non-overlapping test origins. Metrics are calculated using the valid forecasts available within each pipeline, and no missing prediction is imputed. The resulting test periods are closely matched but not completely identical. Therefore, the percentage differences reported between the models are interpreted as descriptive comparisons of the stored experiments rather than as results from a fully paired test design. No formal claim of statistical superiority is made on the basis of these model-specific test sets.

4.3. Computational Environment and Reproducibility

All experiments are implemented in Python. TensorFlow and Keras are used to construct and train the TCN-based forecasting models, while the temporal convolutional layers are implemented using the keras-tcn package. The EEMD procedure is performed using PyEMD, and LightGBM is employed for both the standalone machine-learning baseline and the residual-correction stage of the proposed framework.

To improve experimental reproducibility, a fixed random seed of 42 is applied to Python, NumPy, TensorFlow, and the EEMD procedure wherever supported. The same seed is retained across the evaluated models to reduce variability caused by random initialization and stochastic training operations.

The TCN-based models are optimized using the Adam optimizer with mean squared error as the training loss. The batch size is set to 32, and the maximum number of training epochs is limited to 50. Validation loss is monitored throughout model training. Early stopping is applied with a patience of eight epochs, and the model weights corresponding to the lowest validation loss are restored. In addition, the learning rate is reduced by a factor of 0.5 when the validation loss does not improve for four consecutive epochs.

For the proposed framework, EEMD is implemented using 30 ensemble trials and a noise width of 0.05. The decomposition produces nine intrinsic mode functions and one residual component, resulting in ten synchronized input channels. Each decomposition channel is standardized independently. The forecasting target is also standardized before TCN training. All scaling parameters are estimated from the model-development data and then applied unchanged during test evaluation.

The LightGBM residual-correction model uses the default LGBMRegressor configuration. Candidate correction coefficients are evaluated from 0 to 1 with an interval of 0.05 using the dedicated calibration subset. Under the reported experimental setting, the selected correction coefficient is 0.30. This value is fixed before test evaluation and is not adjusted using test observations.

To reduce the effect of implausible residual adjustments, the final corrected forecasts are clipped to the minimum and maximum load values observed in the model-development period. These limits are determined exclusively from the training and validation observations and are subsequently applied unchanged to the test forecasts. No test target is used to determine the clipping limits, train the residual model, or select the correction coefficient. The principal computational and implementation settings are summarized in Table 3.

To support subsequent analysis and verification, the model predictions, actual target values, residuals, training histories, horizon-wise errors, daily errors, condition-based metrics, feature-importance values, and statistical-test inputs are retained. The present study focuses on forecasting accuracy and model behavior under a fixed software workflow. Hardware-dependent execution-time comparisons are not included because a standardized cross-model timing protocol was not retained for all experiments. The reported reproducibility information therefore concentrates on the algorithmic settings, random seed, training controls, decomposition configuration, and residual-calibration procedure.

Table 3. Computational and implementation settings

Item	Setting
Programming language	Python
Deep-learning framework	TensorFlow/Keras
TCN package	keras-tcn
Decomposition package	PyEMD
Tree-based learner	LightGBM
Random seed	42
Optimizer	Adam
Training loss	Mean squared error
Batch size	32
Maximum training epochs	50
Early-stopping patience	8 epochs
Learning-rate reduction patience	4 epochs
Learning-rate reduction factor	0.5
EEMD ensemble trials	30
EEMD noise width	0.05
EEMD output	9 IMFs and 1 residual component
Residual learner	Default LGBMRegressor
Correction-coefficient range	0–1
Correction-coefficient interval	0.05
Selected correction coefficient	0.30

Vol. 8, No. 5, October 2026, pp. 1346-1373

4.4. Evaluation Methods

Forecasting performance is evaluated using complementary absolute, percentage-based, and goodness-of-fit measures. The principal metrics are root mean squared error, mean absolute error, mean absolute percentage error, weighted absolute percentage error, and the coefficient of determination.

Let y_i and $\hat{y}_i$ denote the actual and predicted load values for Ntest observations. RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{17}$$

MAE is calculated as

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} | y_i - \hat{y}_i | \tag{18}$$

MAPE is defined as

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

WAPE is calculated as

$$\text{WAPE} = 100\frac{\sum_{i=1}^{N} | y_i - \hat{y}_i |}{\sum_{i=1}^{N} | y_i |} \tag{20}$$

The coefficient of determination is

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{21}$$

where $\bar{y}$denotes the mean observed test load.

RMSE gives greater weight to large forecast deviations, MAE measures the average error directly in MW, MAPE and WAPE describe relative forecasting accuracy, and R^2 measures agreement between the predicted and observed load variation. MSE and sMAPE are also calculated and retained in the saved outputs for supplementary comparison (Table 4).

Table 4. Diagnostic evaluation conditions

Condition	Definition
Peak load	Actual load ≥ 90th percentile
Weekday	Monday–Friday
Weekend	Saturday–Sunday
Peak hours	17:00–21:00
Non-peak hours	Outside 17:00–21:00
Rapid change	Absolute load change ≥ 90th percentile
Rapid increase	Large positive load change
Rapid decrease	Large negative load change
Low load	Below 25th percentile
Normal load	25th–75th percentile
High load	Above 75th percentile

In addition to overall test accuracy, the forecasting results are examined across the 48 forecast horizons and across complete daily trajectories. Because the test windows are non-overlapping, each test origin corresponds directly to one full day-ahead prediction.

Additional evaluations are conducted under several operating conditions. Peak-load observations are defined using the 90th percentile of actual test demand. Weekdays and weekends are evaluated separately, and peak hours are defined as the period from 17:00 to 21:00.

Rapid load changes are identified using the 90th percentile of the absolute point-to-point load variation. Rapid increases and rapid decreases are then examined separately. Load-level conditions are defined using the 25th and 75th percentiles of actual test demand.

These evaluations are used to describe the forecasting-error patterns observed across different horizons and operating conditions, rather than relying only on aggregate test metrics. Because the evaluated pipelines do not contain exactly identical target timestamps, no formal paired significance test is reported. The overall, horizon-wise, and condition-based results are therefore interpreted descriptively within their respective stored evaluation sets.

5. RESULTS AND DISCUSSION

5.1. Overall Forecasting Performance

Table 5 summarizes the stored day-ahead forecasting results of the standalone LightGBM baseline, the raw-series TCN baseline, and the proposed EEMD–TCN–LightGBM framework. The reported metrics include MSE, RMSE, MAE, MAPE, sMAPE, WAPE, and R^2. Lower values of the error metrics indicate smaller forecasting deviations, whereas a higher R^2 indicates stronger agreement between the observed and predicted load variations.

Table 5. Overall day-ahead forecasting performance under the stored evaluation setting

Model	MSE	RMSE (MW)	MAE (MW)	MAPE (%)	sMAPE (%)	WAPE (%)	R^2
LightGBM	57,473.76	239.74	168.87	2.7527	2.7334	2.7192	0.9283
TCN	63,051.46	251.10	177.82	2.8720	2.8421	2.8618	0.9212
Proposed EEMD-TCN-LightGBM	24,623.13	156.92	116.48	1.8825	1.8827	1.8746	0.9692

The standalone LightGBM baseline reports an RMSE of 239.74 MW, an MAE of 168.87 MW, and a MAPE of 2.7527%. The raw-series TCN reports an RMSE of 251.10 MW, an MAE of 177.82 MW, and a MAPE of 2.8720%. Under their respective stored test constructions, LightGBM therefore produces lower error values than the raw-series TCN across the reported metrics. This observation suggests that the engineered lagged, rolling-statistical, and calendar features provide an effective representation for the considered dataset.

The proposed EEMD–TCN–LightGBM framework reports an MSE of 24,623.13, an RMSE of 156.92 MW, an MAE of 116.48 MW, a MAPE of 1.8825%, an sMAPE of 1.8827%, a WAPE of 1.8746%, and an R^2 of 0.9692. These values are lower than the error values reported for the two baseline pipelines under the stored experimental setting.

Relative to the standalone LightGBM results, the descriptive reductions in RMSE, MAE, and MAPE are 34.55%, 31.02%, and 31.61%, respectively. Relative to the raw-series TCN results, the corresponding descriptive reductions are 37.51%, 34.50%, and 34.45%. Similar patterns are observed for the remaining error metrics. These percentages quantify the differences between the stored model outputs but should not be interpreted as results from an exactly paired test because the LightGBM and TCN-based pipelines contain slightly different valid test-origin sets.

The lower reported errors of the proposed framework are consistent with the intended complementary roles of its three stages. EEMD provides a multi-scale input representation, the TCN learns the direct 48-step temporal mapping, and LightGBM estimates part of the remaining structured residual. However, the present results do not isolate the individual contribution of each stage through a complete ablation experiment. Therefore, the observed differences are attributed to the integrated pipeline as a whole rather than quantitatively assigned to EEMD or residual correction separately.

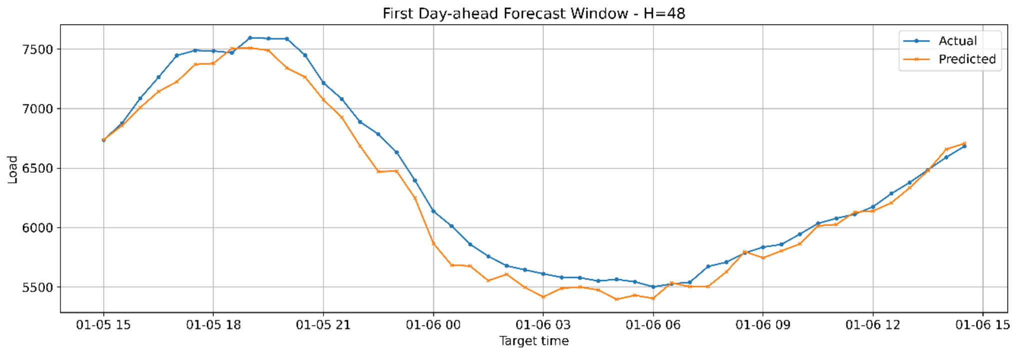
A further limitation is that EEMD is performed offline on the complete load series. Consequently, Table 5 represents an offline decomposition-assisted benchmark and not a strictly causal real-time forecasting evaluation. The reported numerical improvements should therefore be interpreted within this experimental scope. They indicate promising behavior of the integrated framework but do not establish that the same performance would be maintained under rolling causal decomposition or on an exactly paired operational test set.

Overall, the stored results show that the proposed pipeline reports lower error values across all considered metrics. The following subsections examine the corresponding forecast trajectories, error distributions, horizon-wise behavior, operating-condition results, and model-specific diagnostic patterns. These analyses are descriptive and are intended to characterize the behavior of the evaluated pipelines rather than provide formal statistical proof of superiority.

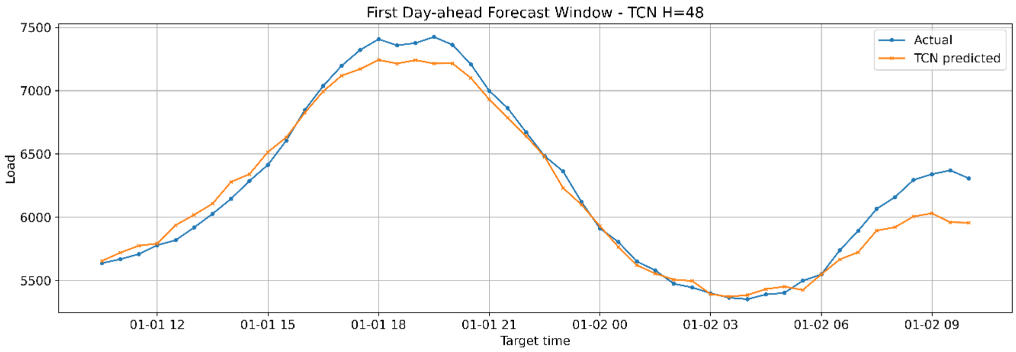
5.2. Comparative Forecasting Behavior

Although Table 5 summarizes the aggregate error values reported by the three forecasting pipelines, these metrics do not show how each model follows the temporal shape of an individual day-ahead load trajectory. Figure 2 therefore presents representative 48-step forecasts produced by the LightGBM baseline, the raw-series

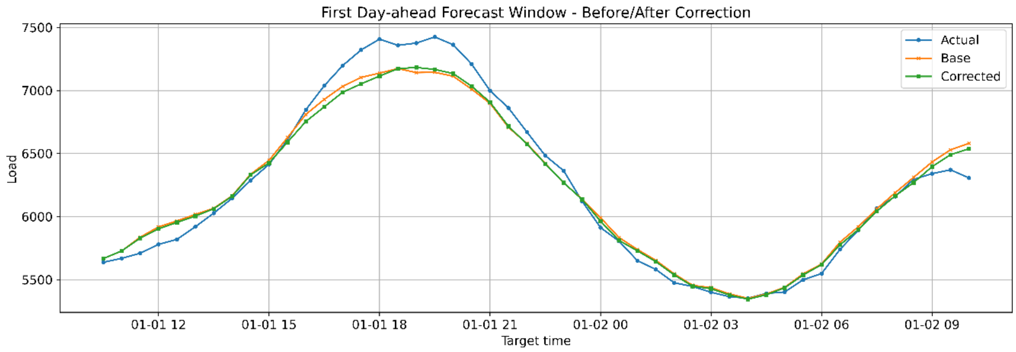
TCN baseline, and the proposed framework. For the proposed model, both the EEMD-TCN base forecast and the final corrected prediction are shown to illustrate the effect of LightGBM residual correction.



(a) LightGBM baseline



(b) Raw-series TCN baseline



(c) Proposed EEMD-TCN-LightGBM framework before and after residual correction

Figure 2. Representative 48-step day-ahead forecasting results: (a) LightGBM baseline, (b) raw-series TCN baseline, and (c) proposed EEMD-TCN-LightGBM framework before and after residual correction

As shown in Figure 2(a), LightGBM captures the overall daily demand pattern, including the high-load period, the subsequent decline, the overnight low-load interval, and the following recovery. However, visible deviations remain around the peak and during the descending part of the trajectory, where the model tends to underestimate the observed load. This behavior indicates that the engineered features provide a strong representation of recurring demand patterns but do not fully reproduce all rapid temporal variations.

The raw-series TCN in Figure 2(b) also reproduces the general shape of the next-day load profile. The model follows the main rising and falling trends closely, particularly around the central portion of the forecasting horizon. Nevertheless, it underestimates the daily peak and shows a larger deviation during the final recovery period. These results are consistent with Table 5, where the raw TCN performs slightly worse than LightGBM despite learning the complete 48-step trajectory through a shared temporal representation.

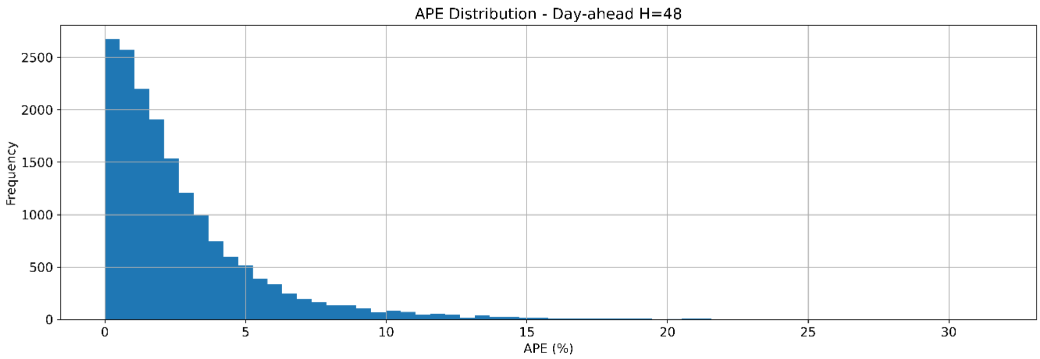
Figure 2(c) illustrates the behavior of the proposed framework before and after residual correction. The EEMD-TCN base forecast already captures the principal shape of the actual load trajectory, while the corrected prediction introduces relatively small adjustments rather than substantially changing the overall forecast. This

Vol. 8, No. 5, October 2026, pp. 1346-1373

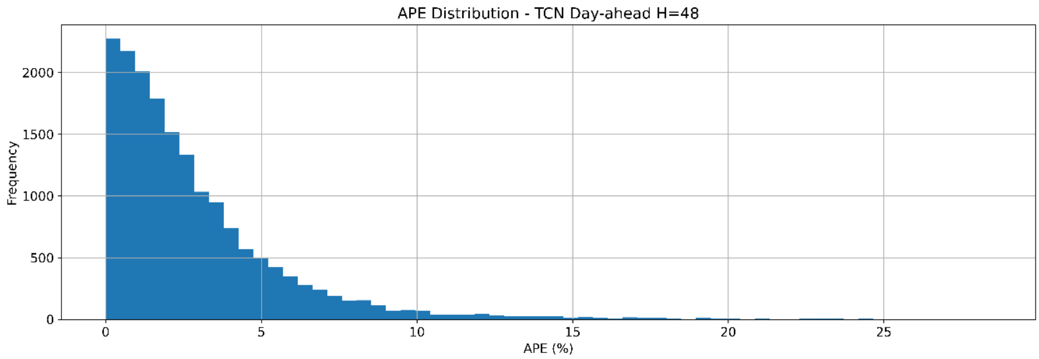
confirms that LightGBM acts as a secondary error-correction model rather than as an independent forecaster. The calibrated correction moves the base prediction closer to the observations in several parts of the trajectory, particularly where systematic local deviations remain.

Overall, the representative trajectories illustrate that the proposed framework follows the selected observed day-ahead profile more closely than the two baselines in the displayed example. The LightGBM correction modifies the EEMD–TCN base output through relatively small adjustments rather than replacing the underlying forecast trajectory. Because the figure presents representative windows rather than the complete test distribution, it should be interpreted as a qualitative illustration of model behavior and not as independent evidence of general superiority.

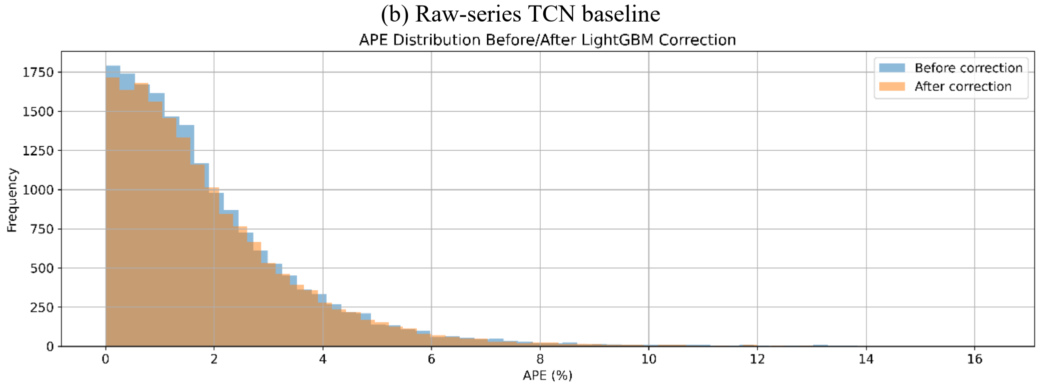
The representative trajectories in Figure 2 illustrate how the models behave over individual day-ahead windows. However, a single forecast window cannot fully characterize performance over the complete test set. To provide a broader view of point-wise forecasting accuracy, Figure 3 presents the distributions of absolute percentage error (APE) for the two baseline models and the proposed framework before and after residual correction.



(a) LightGBM baseline



(b) Raw-series TCN baseline



(c) Proposed framework before and after LightGBM residual correction

Figure 3. Absolute percentage error distributions of the evaluated forecasting approaches: (a) LightGBM baseline, (b) raw-series TCN baseline, and (c) proposed EEMD-TCN framework before and after LightGBM residual correction

As shown in Figure 3(a) and Figure 3(b), the APE distributions of both baseline models are strongly right-skewed. Most predictions are concentrated in the low-error region, while a smaller number of observations form a long upper tail. This pattern indicates that both LightGBM and the raw-series TCN provide relatively accurate forecasts for a large proportion of the test samples but still produce occasional larger deviations under more difficult operating conditions.

The longer error tails observed for the baseline models are consistent with their higher aggregate MAPE values in Table 5. Although most forecast points remain within a relatively small percentage-error range, the presence of higher-error observations increases the overall forecasting error and suggests that certain load transitions or operating conditions remain difficult for the individual baseline models.

Figure 3(c) directly compares the EEMD-TCN base forecast before correction with the final output after LightGBM residual correction. The two distributions largely overlap, indicating that the correction stage preserves the general behavior of the base forecast rather than substantially altering all predictions. However, the corrected distribution shows a modest shift toward lower APE values and a reduction in part of the higher-error region. This confirms that the residual model acts mainly through targeted adjustments to systematic errors.

The relatively small visual difference between the before- and after-correction distributions is also consistent with the calibrated correction strategy. Because only a fraction of the predicted residual is added to the base forecast, the final model is designed to improve systematic deviations without producing large changes to already accurate predictions. The correction stage therefore refines the EEMD-TCN output rather than replacing its underlying forecast structure.

Figure 3 provides a descriptive comparison of the APE distributions generated by the stored model outputs. The proposed framework exhibits a more concentrated distribution and a smaller high-error region in the displayed evaluation. However, because the baseline pipelines do not contain completely identical target sets, the distributions should not be interpreted as a formally paired statistical comparison.

5.3. Horizon-Wise Performance and Effect of Residual Correction

The error distributions in Figure 3 describe overall point-wise accuracy but do not show whether forecasting difficulty changes with lead time. Figure 4 therefore compares MAPE across the complete 48-step day-ahead horizon. The first two panels show the horizon-wise behavior of the LightGBM and raw-series TCN baselines, whereas the third panel compares the proposed EEMD-TCN forecast before and after LightGBM residual correction.

As shown in Figure 4(a), the LightGBM baseline maintains relatively moderate errors over the early and middle forecast horizons, but its MAPE increases substantially after approximately horizon 35. The largest errors occur near the end of the forecasting window, where MAPE exceeds 5%. This pattern indicates that the horizon-specific feature-based models become less accurate for the later part of the next-day trajectory.

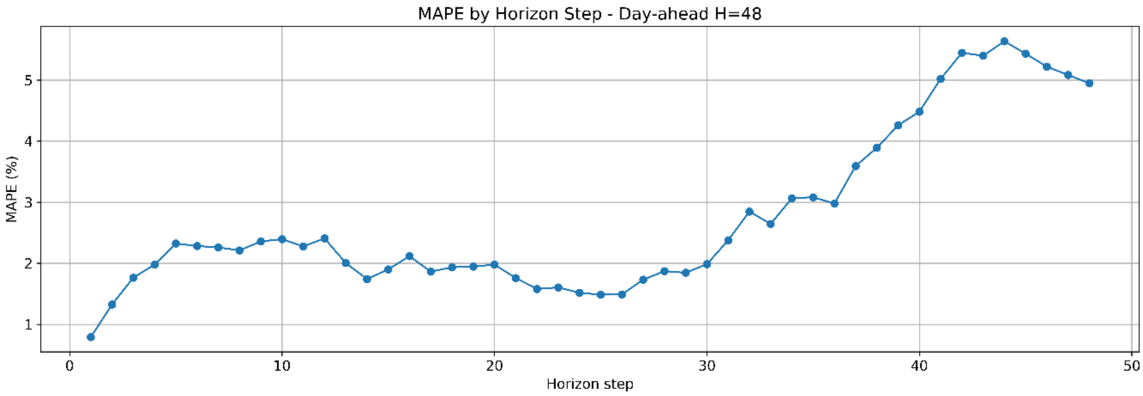
A similar late-horizon deterioration is observed for the raw-series TCN in Figure 4(b). The TCN also shows higher errors during several early-to-middle horizons and then experiences a sharp increase after approximately horizon 39. The final horizons reach MAPE values of around 5%, confirming that longer lead times remain the most difficult part of the direct day-ahead forecasting task.

Figure 4(c) shows a clearly lower error level for the EEMD-TCN base model. Over most of the first 40 horizons, MAPE remains approximately between 1.4% and 2.2%, before increasing during the final portion of the forecast. Under the stored evaluation setting, the EEMD–TCN base output exhibits lower and more stable horizon-wise errors than the raw-series TCN output over most of the forecasting horizon. Because the present study does not include a complete component-level ablation on an identical paired test set, this pattern should not be interpreted as an isolated quantitative effect of EEMD alone.

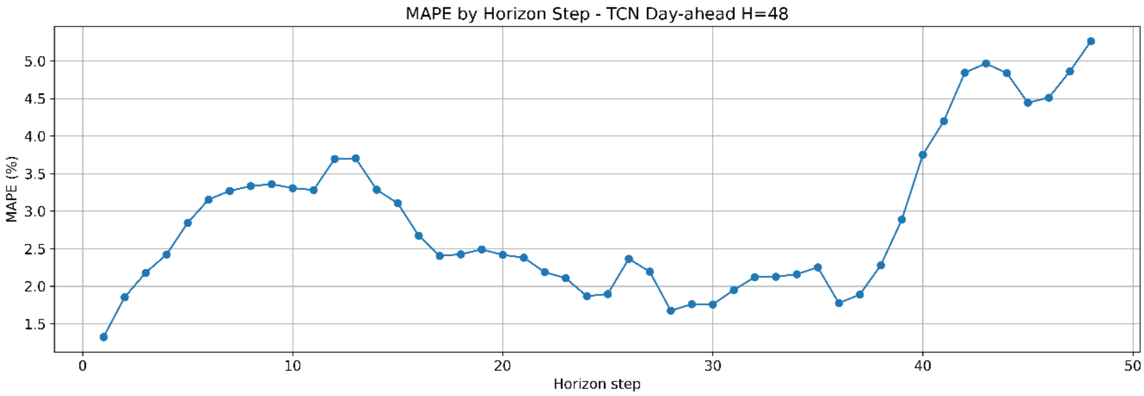
The LightGBM residual correction further reduces MAPE at many horizons, particularly during the later forecast steps where the base-model error increases most strongly. The improvement is not completely uniform; at several early and middle horizons, the corrected curve is slightly above the base forecast. However, the correction produces more consistent reductions across the difficult final horizons, where larger errors have a stronger effect on overall forecasting performance.

Overall, Figure 4 indicates that the reported forecasting errors tend to increase toward the later portion of the 48-step horizon. Within the proposed pipeline, residual correction reduces MAPE at many horizons, particularly near the end of the forecasting window, although it is not beneficial at every step. These patterns describe the stored horizon-wise outputs and support the use of a calibrated rather than unrestricted correction. Because the evaluated pipelines are not based on exactly identical target sets, the cross-model curves are interpreted descriptively.

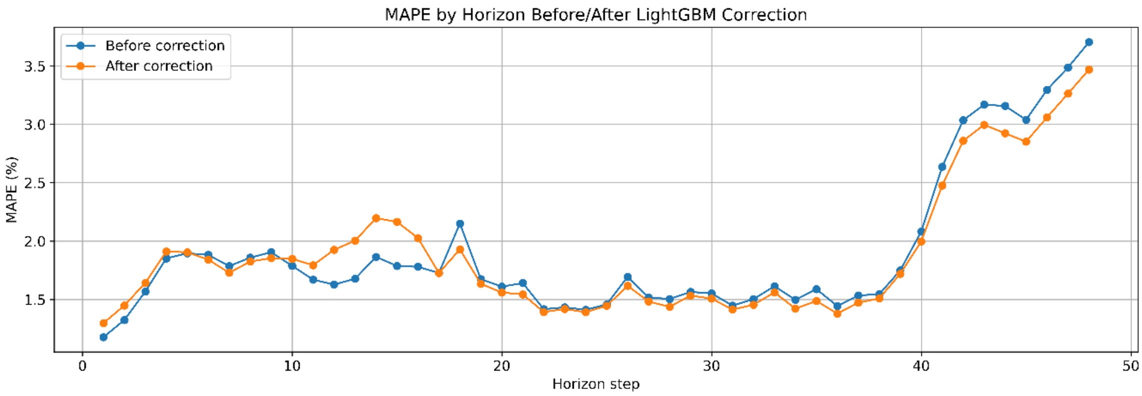
Vol. 8, No. 5, October 2026, pp. 1346-1373



(a) LightGBM baseline



(b) Raw-series TCN baseline



(c) Proposed EEMD-TCN framework before and after LightGBM residual correction

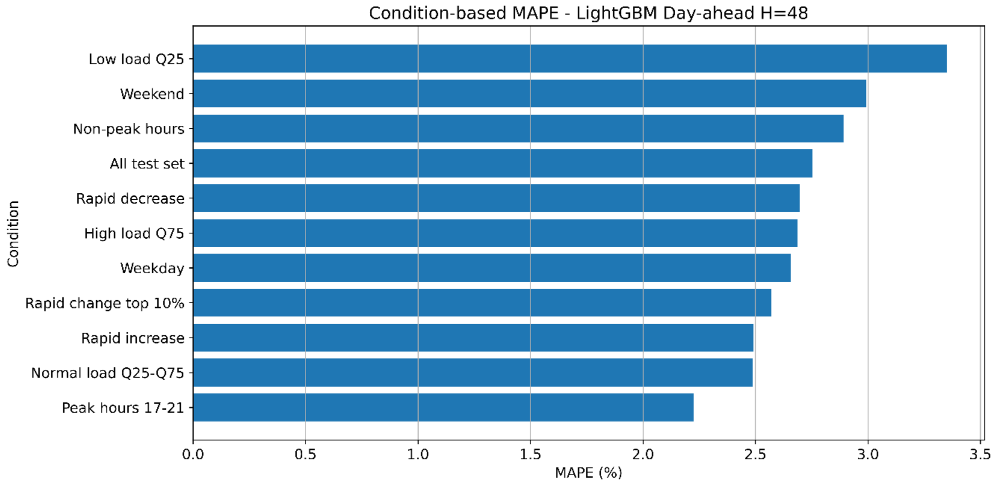
Figure 4. Horizon-wise MAPE comparison over the 48-step forecasting horizon: (a) LightGBM baseline, (b) raw-series TCN baseline, and (c) proposed EEMD-TCN framework before and after LightGBM residual correction

5.4. Condition-Based Performance

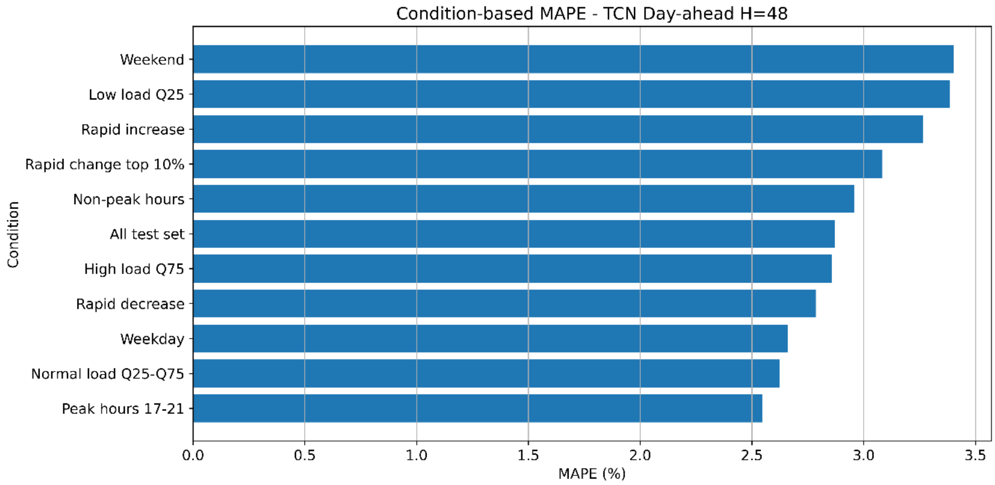
The horizon-wise results in Figure 4 show that forecasting difficulty varies across lead times. However, forecast accuracy may also depend on the operating condition of the power system. Figure 5 therefore compares the MAPE of the three evaluated models under different load levels, calendar conditions, peak and non-peak hours, and rapid load variations.

As shown in Figure 5(a), the LightGBM baseline exhibits its highest error under low-load conditions, followed by weekends and non-peak hours. In contrast, the lowest MAPE is obtained during peak hours. This result suggests that the engineered lag and calendar features represent regular high-demand patterns effectively, while lower-load periods remain more difficult because percentage errors become more sensitive to absolute deviations.

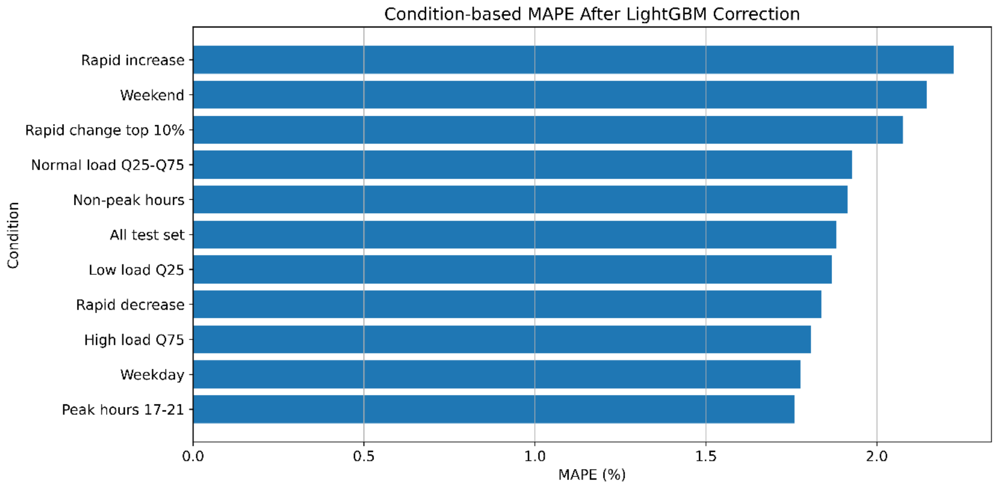
Vol. 8, No. 5, October 2026, pp. 1346-1373



(a) LightGBM baseline



(b) Raw-series TCN baseline



(c) Proposed EEMD-TCN-LightGBM framework

Figure 5. Condition-based forecasting performance of the evaluated models: (a) LightGBM baseline, (b) raw-series TCN baseline, and (c) proposed EEMD-TCN-LightGBM framework

The TCN results in Figure 5(b) show a similar dependence on operating conditions. The largest errors occur during weekends, low-load periods, and rapid increases, whereas peak-hour and normal-load conditions are forecast more accurately. Compared with LightGBM, the raw-series TCN is particularly affected by rapid load changes, indicating that a single-channel temporal representation may have greater difficulty following abrupt transitions.

Figure 5(c) shows that the proposed framework reports lower condition-specific MAPE values than the two baseline pipelines across the displayed categories. Most of the reported values remain below 2%, while relatively higher errors occur during rapid increases, weekends, and abrupt load changes. Lower values are observed during peak-hour, weekday, and high-load conditions.

These results provide a descriptive characterization of model behavior under different operating conditions. Because the baseline pipelines contain slightly different valid test-origin sets, the comparisons should not be interpreted as fully paired evidence of robustness. Moreover, because a complete component-level ablation is not available, the observed patterns are attributed to the integrated EEMD–TCN–LightGBM framework rather than separately to EEMD or residual correction.

5.5. Model Interpretation and Computational Cost

The previous analyses focused on forecasting accuracy across overall, horizon-wise, and operating-condition perspectives. Figure 6 examines the internal behavior of the evaluated models from a different perspective, highlighting the information used by the LightGBM models and the TCN's training dynamics.

The feature-ranking results in Figure 6(a) show that the standalone LightGBM forecast is driven primarily by a combination of target-time information and recent historical demand. The future hour is the most influential predictor, followed by the 24-step load lag, the load observed at the forecast origin, and shorter historical lags such as 12 and 8 steps. This ranking confirms the strong role of intraday position and recent load persistence in day-ahead forecasting. Daily and weekly temporal encodings, together with rolling statistics, also contribute to the prediction, although their importance is lower than that of the dominant load and time variables.

The training behavior of the raw-series TCN is shown in Figure 6(b). Training loss decreases continuously throughout the optimization process, indicating that the network progressively fits the historical sequence. Validation loss also declines rapidly during the early epochs and reaches its lowest region around the middle of training. After this point, further reductions in training loss are not accompanied by comparable validation improvement, and the validation curve begins to fluctuate slightly. This pattern supports the use of early stopping and best-weight restoration to limit overfitting and retain the model state with the strongest validation performance.

A different importance structure appears in the residual-correction model in Figure 6(c). The dominant predictors are target index, target hour, and the EEMD-TCN base prediction. Their importance indicates that the remaining forecast error depends strongly on both the position within the 48-step horizon and the magnitude of the original base forecast. Target-time cyclical variables and calendar features also contribute substantially, suggesting that residual bias varies systematically across different parts of the daily load cycle.

The correction model also assigns meaningful importance to rolling weekly and daily statistics and to decomposition-derived variables such as IMF 8 and IMF 9 at the forecast origin. This finding supports the design of the residual-learning stage. The remaining forecast error is not explained solely by the base prediction; it also contains information related to recent load variability, temporal position, and the current multi-scale state of the demand signal.

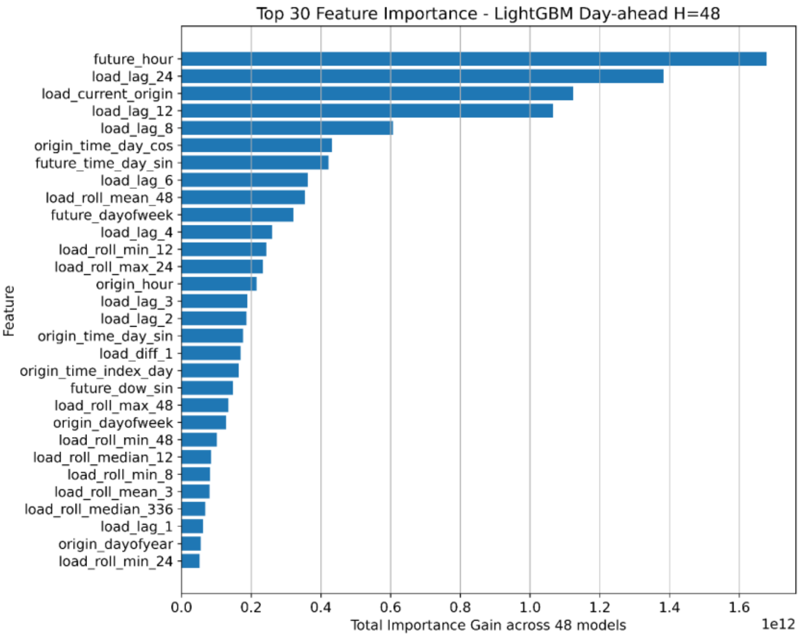
The contrast between Figure 6(a) and Figure 6(c) illustrates the different roles assigned to the two LightGBM models. The standalone LightGBM model relies mainly on historical demand and calendar variables to generate the future load forecast, whereas the residual-correction model places greater emphasis on the EEMD–TCN base prediction, forecast position, and temporal context. The first model predicts electricity demand directly, while the second estimates structured deviations remaining after the base forecast.

These importance values describe how the fitted LightGBM models use the available predictors within the reported dataset. They should not be interpreted as evidence of physical causality or as proof that the same feature ranking would be obtained for another region or forecasting period.

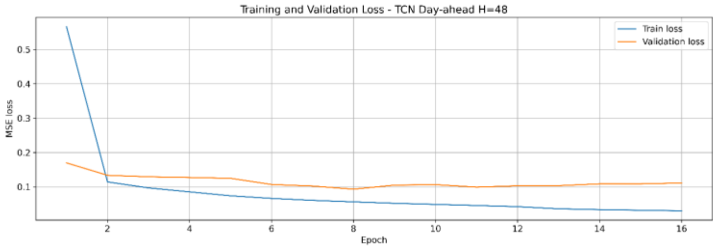
Taken together, the diagnostic panels are consistent with the intended task separation of the proposed architecture. However, because a complete component-level ablation is not available, the diagnostic results do not independently quantify the contribution of each stage to the final forecasting performance.

The present analysis focuses on forecasting accuracy and model-specific diagnostic behavior. A direct computational-cost comparison is not reported because a standardized timing protocol was not retained consistently across the standalone LightGBM, TCN, EEMD, and residual-correction stages. Accordingly, no claim is made that the proposed framework is computationally more efficient than the baseline models.

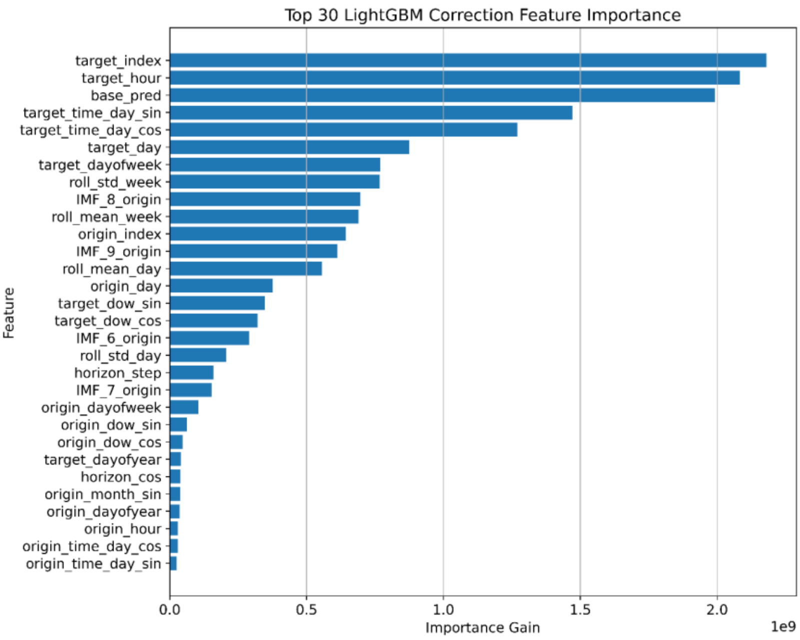
Vol. 8, No. 5, October 2026, pp. 1346-1373



(a) Feature importance of the standalone LightGBM baseline



(b) Training and validation loss of the raw-series TCN



(c) Feature importance of the LightGBM residual-correction model

Figure 6. Model-specific diagnostic results: (a) feature importance of the standalone LightGBM baseline, (b) training and validation loss of the raw-series TCN, and (c) feature importance of the LightGBM residual-correction model

6. CONCLUSION

This study presented a hybrid EEMD–TCN framework with LightGBM residual correction for direct day-ahead electric load forecasting. The framework separates the forecasting process into three stages. EEMD transforms the original demand series into nine intrinsic mode functions and one residual component, a TCN maps the resulting seven-day multi-channel history directly to the complete 48-step next-day profile, and LightGBM estimates structured residual errors. A correction coefficient selected on a separate chronological calibration subset controls the magnitude of the final adjustment.

Using half-hourly Queensland electricity-demand data from 2015 to 2019, the proposed framework reported an RMSE of 156.92 MW, an MAE of 116.48 MW, a MAPE of 1.8825%, a WAPE of 1.8746%, and an R^2 of 0.9692. Under the stored experimental setting, these values were lower than those reported for the standalone LightGBM and raw-series TCN pipelines. The horizon-wise, error-distribution, and condition-based analyses also showed favorable descriptive patterns for the proposed framework.

The results suggest that multi-scale input representation, direct temporal convolutional forecasting, and calibrated residual learning can perform complementary roles within an integrated forecasting pipeline. However, the present experiment does not provide a complete ablation analysis that isolates the individual contribution of EEMD and residual correction. Therefore, the numerical differences are interpreted as the performance of the complete framework rather than as separate quantitative gains attributable to each component.

Two methodological limitations are central to the interpretation of the results. First, EEMD is applied offline to the complete load series before forecasting windows are generated. The reported experiment is therefore an offline decomposition-assisted benchmark and not a strictly causal real-time forecasting implementation. Second, the standalone LightGBM pipeline contains 360 valid test origins, whereas the TCN-based pipelines contain 364. Consequently, the reported cross-model percentage differences are descriptive and do not constitute a fully paired statistical comparison.

Future work should evaluate the framework using rolling-window or expanding-window causal decomposition, common forecast origins and target timestamps, and additional electricity markets. A complete ablation study and repeated-run evaluation would also be useful for isolating the contribution and stability of the individual modeling stages.

REFERENCES

[1] Y. Kong, Y. Kong, Y. Chen, J. Du, Y. Yang, and Q. Xu, "A bidding strategy for virtual power plants in the day-ahead market," *Energies*, vol. 18, no. 18, p. 4874, 2025, https://doi.org/10.3390/en18184874.

[2] K. K. Miskiw, E. Kraft, and S.-E. Fleten, "Coordinated bidding in sequential electricity markets: Effects of price-making," *Energy Economics*, vol. 144, p. 108316, 2025, https://doi.org/10.1016/j.eneco.2025.108316.

[3] A. Petrusic and A. Janjic, "Fuzzy Multiple Linear Regression for the Short Term Load Forecast," in *2020 55th International Scientific Conference on Information, Communication and Energy Systems and Technologies ICEST*, pp. 102–105, 2020, https://doi.org/10.1109/ICEST49890.2020.9232776.

[4] L. Pérez-Domínguez, H. Garg, D. Luviano-Cruz, and J. L. García Alcaraz, "Estimation of linear regression with the dimensional analysis method," *Mathematics*, vol. 10, no. 10, p. 1645, 2022, https://doi.org/10.3390/math10101645.

[5] H. M. Al-Hamadi, "Long-term electric power load forecasting using fuzzy linear regression technique," in *2011 IEEE Power Engineering and Automation Conference*, vol. 3, pp. 96–99, 2011, https://doi.org/10.1109/PEAM.2011.6135023.

[6] L. Han, P. Pinson, and J. Kazempour, "Trading data for wind power forecasting: A regression market with lasso regularization," *Electric Power Systems Research*, vol. 212, p. 108442, 2022, https://doi.org/10.1016/j.epsr.2022.108442.

[7] V. Urošević and A. M. Savić, "Temporal clustering for accurate short-term load forecasting using Bayesian multiple linear regression," *Applied Intelligence*, vol. 55, no. 1, p. 10, 2025, https://doi.org/10.1007/s10489-024-05887-z.

[8] M. Pleños, "Time Series Forecasting Using Holt-Winters Exponential Smoothing: Application to Abaca Fiber Data," *Zeszyty Naukowe Szkoły Główniej Gospodarstwa Wiejskiego w Warszawie - Problemy Rolnictwa Światowego*, vol. 22, no. 2, pp. 17–29, 2022, https://doi.org/10.22630/prs.2022.22.2.6.

[9] S. Sawalha and G. Al-Naymat, "An Adaptive Holt–Winters Model for Seasonal Forecasting of Internet of Things IoT Data Streams," *Internet of Things*, vol. 6, no. 3, pp. 1–18, 2025, https://doi.org/10.3390/iot6030039.

[10] U. I. Lezama Lope, A. Benavides-Vázquez, G. Santamaría-Bonfil, and R. Z. Ríos-Mercado, "Fast and Efficient Very Short-Term Load Forecasting Using Analogue and Moving Average Tools," *IEEE Transactions on Learning Technologies*, vol. 21, no. 9, pp. 1015–1021, 2023, https://doi.org/10.1109/tla.2023.10251808.

[11] A. Jarraya *et al*., "On-Site temperature and irradiance forecast tuning for improved load prediction in buildings," *Energy and Buildings*, vol. 337, p. 115642, 2025, https://doi.org/10.1016/j.enbuild.2025.115642.

[12] S. Deng, X. Dong, L. Tao, J. Wang, Y. He, and D. Yue, "Multi-type load forecasting model based on random forest and density clustering with the influence of noise and load patterns," *Energy*, vol. 307, p. 132635, 2024, https://doi.org/10.1016/j.energy.2024.132635.

[13] S. Sarker, A. Sarker, A. Razzak, and Z. R. Khan, "Hybrid Deep Learning-Based Multivariate Forecasting of Relative Humidity and Air Temperature: A Case Study of the Rangpur Division, Bangladesh," *Expert Systems with Applications*, vol. 299, p. 133184, 2026, https://doi.org/10.1016/j.eswa.2026.133184.
[14] U. Samal and A. Kumar, "Enhancing Software Reliability Forecasting Through a Hybrid ARIMA-ANN Model," *Arabian Journal for Science and Engineering*, vol. 49, no. 5, pp. 6845–6859, 2023, https://doi.org/10.1007/s13369-023-08486-1.
[15] S. Chen, R. Lin, and W. Zeng, "Short-Term Load Forecasting Method Based on ARIMA and LSTM," in *2022 IEEE 25th International Conference on Computer Communication Technology ICCT*, pp. 1913–1917, 2022, https://doi.org/10.1109/ICCT56141.2022.10073051.
[16] J. A. C. Dias *et al*., "Enhanced Carbon Flux Forecasting via STL Decomposition and Hybrid ARIMA-ES-LSTM Model in Amazon Forest," *IEEE Access*, vol. 13, pp. 84713–84726, 2025, https://doi.org/10.1109/ACCESS.2025.3561166.
[17] H. Zhang, X. Liu, X. Gao, and Z. Liu, "Hybrid ARIMAX model integrating optimal time-lag selection and markov chain error correction for residential heating load forecasting," *Energy Reports*, vol. 14, pp. 5259–5273, 2025, https://doi.org/10.1016/j.egyr.2025.11.082.
[18] X. Guo, Q. Zhao, D. Zheng, Y. Ning, and Y. Gao, "A short-term load forecasting model of multi-scale CNN-LSTM hybrid neural network considering the real-time electricity price," *Energy Reports*, vol. 6, pp. 1046–1053, 2020, https://doi.org/10.1016/j.egyr.2020.11.078.
[19] H. Nagpal, A. Soppelsa, H. Madsen, and R. Fedrizzi, "An adaptive multi-seasonal ARIMA approach for domestic hot water load forecasting: A pilot study," *Energy and Buildings*, vol. 347, p. 116226, 2025, https://doi.org/10.1016/j.enbuild.2025.116226.
[20] G. F. Fan, R. T. Zhang, C. C. Cao, Y. H. Yeh, and W. C. Hong, "Applications of empirical wavelet decomposition, statistical feature extraction, and antlion algorithm with support vector regression for resident electricity consumption forecasting," *Nonlinear Dynamics*, vol. 111, no. 21, pp. 20139–20163, 2023, https://doi.org/10.1007/s11071-023-08922-9.
[21] Z. Zhang, Y. Dong, and W. C. Hong, "Long Short-Term Memory-Based Twin Support Vector Regression for Probabilistic Load Forecasting," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 1, pp. 1764–1778, 2023, https://doi.org/10.1109/TNNLS.2023.3335355.
[22] S. Sreekumar, J. Verma, A. Sujil, and R. Kumar, "One day forth forecasting of hourly electrical load using genetically tuned Support Vector Regression for smart grid frame work," in *2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences RAECS*, pp. 1–6, 2016, https://doi.org/10.1109/RAECS.2015.7453405.
[23] Z. Zhang, Q. Zhang, H. Liang, and B. Gorbani, "Optimizing electric load forecasting with support vector regression/LSTM optimized by flexible Gorilla troops algorithm and neural networks a case study," *Scientific Reports*, vol. 14, no. 1, pp. 1–20, 2024, https://doi.org/10.1038/s41598-024-73893-9.
[24] M. Safari, A. H. Rabiee, and J. Joudaki, "Developing a Support Vector Regression SVR Model for Prediction of Main and Lateral Bending Angles in Laser Tube Bending Process," *Materials*, vol. 16, no. 8, p. 3251, 2023, https://doi.org/10.3390/ma16083251.
[25] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning: A First Course for Engineers and Scientists*. 2022, https://doi.org/10.1017/9781108919371.
[26] N. N. Sergeev and P. V. Matrenin, "Enhancing Efficiency of Ensemble Machine Learning Models for Short-Term Load Forecasting through Feature Selection," in *2022 IEEE 23rd International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices EDM*, pp. 368–371, 2022, https://doi.org/10.1109/EDM55285.2022.9855022.
[27] Y. Chen, C. Lin, Y. Zhang, J. Liu, and D. Yu, "Day-ahead load forecast based on Conv2D-GRU_SC aimed to adapt to steep changes in load," *Energy*, vol. 302, p. 131814, 2024, https://doi.org/10.1016/j.energy.2024.131814.
[28] M. Ahmed, T. A. Elgalhud, M. Navid, S. Mir, and K. Grolinger, "Federated Online Learning for adaptive load forecasting across decentralized nodes," *International Journal of Electrical Power & Energy Systems*, vol. 169, p. 110779, 2025, https://doi.org/10.1016/j.ijepes.2025.110779.
[29] U. Singh and S. Vadhera, "Random Forest and Xgboost Technique for Short-Term Load Forecasting," in *2022 1st International Conference on Sustainable Technology for Power and Energy Systems STPES*, pp. 1–6, 2022, https://doi.org/10.1109/STPES54845.2022.10006635.
[30] V. Veeramsetty, K. R. Reddy, M. Santhosh, A. Mohnot, and G. Singal, "Short-term electric power load forecasting using random forest and gated recurrent unit," *Electrical Engineering*, vol. 104, no. 1, pp. 307–329, 2022, https://doi.org/10.1007/s00202-021-01376-5.
[31] S. Wang and J. Ma, "A Novel Ensemble Model for Load Forecasting: Integrating Random Forest, XGBoost, and Seasonal Naive Methods," in *2023 2nd Asian Conference on Frontiers of Power and Energy ACFPE*, pp. 114–118, 2023, https://doi.org/10.1109/ACFPE59335.2023.10455494.
[32] E. Nziyumva, R. Hu, C.-Y. Hsu, and J. Niyogisubizo, "Electrical load forecasting using hybrid of extreme gradient boosting and light gradient boosting machine," in *The International Conference on Image, Vision and Intelligent Systems ICIVIS 2021*, vol. 813, pp. 1083–1093, 2022, https://doi.org/10.1007/978-981-16-6963-7_95.

[33] P. Saini and S. K. Parida, "A novel probabilistic gradient boosting model with multi-approach feature selection and iterative seasonal trend decomposition for short-term load forecasting," *Energy*, vol. 294, p. 130975, 2024, https://doi.org/10.1016/j.energy.2024.130975.
[34] Z. Qinghe, X. Wen, H. Boyan, W. Jong, and F. Junlong, "Optimised extreme gradient boosting model for short term electric load demand forecasting of regional grid system," *Scientific Reports*, vol. 12, no. 1, pp. 1–12, 2022, https://doi.org/10.1038/s41598-022-22024-3.
[35] M. Mohamed, F. E. Mahmood, M. A. Abd, M. Rezkallah, and A. Hamadi, "Load Demand Forecasting Using eXtreme Gradient Boosting XGBoost," in *2023 IEEE Industry Applications Society Annual Meeting IAS*, pp. 1–7, 2023, https://doi.org/10.1109/IAS54024.2023.10406613.
[36] S. Singh and M. M. Tripathi, "A Comparative Analysis of Extreme Gradient Boosting Technique with Long Short-Term Memory and Layered Recurrent Neural Network for Electricity Demand Forecas," in *2021 6th International Conference on Recent Trends in Electronics, Information & Communication Technology RTEICT*, pp. 297–302, 2021, https://doi.org/10.1109/RTEICT52294.2021.9573988.
[37] G. Suo, L. Song, Y. Dou, Z. Cui, and A. X. Principle, "Multi-dimensional Short-term Load Forecasting Based on XGBoost and Fireworks Algorithm," in *2019 18th International Symposium on Distributed Computing and Applications for Business Engineering and Science DCABES*, pp. 245–248, 2019, https://doi.org/10.1109/DCABES48411.2019.00068.
[38] L. Zhang and D. Jánošík, "Enhanced short-term load forecasting with hybrid machine learning models: CatBoost and XGBoost approaches," *Expert Systems with Applications*, vol. 241, p. 122686, 2024, https://doi.org/10.1016/j.eswa.2023.122686.
[39] M. Shafiuzzaman *et al*., "Enhanced very short-term load forecasting with multi-lag feature engineering and prophet-XGBoost-CatBoost architecture," *Energy*, vol. 335, p. 137981, 2025, https://doi.org/10.1016/j.energy.2025.137981.
[40] J. Zhou and M. Ding, "Short-term load forecasting for holidays based on similar days selecting and XGBoost model," in *2023 IEEE 8th International Conference on Power and Renewable Energy ICPRE*, pp. 1–5, 2023, https://doi.org/10.1109/ICPS58381.2023.10128055.
[41] L. Semmelmann, S. Henni, and C. Weinhardt, "Load forecasting for energy communities: a novel LSTM-XGBoost hybrid model based on smart meter data," *Energy Informatics*, vol. 5, no. 1, pp. 1–21, 2022, https://doi.org/10.1186/s42162-022-00212-9.
[42] K. Mishra, A. Yadav, and N. D. Londhe, "Load demand forecasting of chhattisgarh state for demand management system with optimal feature selection using XGBOOST approach," *Electric Power Systems Research*, vol. 253, p. 112567, 2026, https://doi.org/10.1016/j.epsr.2025.112567.
[43] F. Wang, "Net Load Segmented Forecasting Method For Data Center Based on GS-LightGBM Model," in *2023 IEEE IAS Global Conference on Renewable Energy and Hydrogen Technologies GlobConHT*, pp. 1–6, 2023, https://doi.org/10.1109/GlobConHT56829.2023.10087415.
[44] Z. Fang, J. Zhan, J. Cao, L. Gan, and H. Wang, "Research on Short-Term and Medium-Term Power Load Forecasting Based on STL-LightGBM," in *2022 2nd International Conference on Electrical Engineering and Control Science IC2ECS*, pp. 1047–1051, 2022, https://doi.org/10.1109/IC2ECS57645.2022.10088145.
[45] Y. Miao and Z. Chen, "Short-term Load Forecasting Based on Echo State Network and LightGBM," in *2023 IEEE International Conference on Predictive Control of Electrical Drives and Power Electronics PRECEDE*, pp. 1–6, 2023, https://doi.org/10.1109/PRECEDE57319.2023.10174609.
[46] Z. Wang, L. Chen, and C. Wang, "Weekly periodicity feature-driven CNN-GRU and LightGBM hybrid model for short-term multi-step building load forecasting," *Energy*, vol. 355, p. 141163, 2026, https://doi.org/10.1016/j.energy.2026.141163.
[47] Y. Tan, Z. Teng, C. Zhang, G. Zuo, Z. Wang, and Z. Zhao, "Long-Term Load Forecasting Based on Feature fusion and LightGBM," in *2021 IEEE 4th International Conference on Power and Energy Applications ICPEA*, pp. 104–109, 2021, https://doi.org/10.1109/ICPEA52760.2021.9639313.
[48] V. Gundu and S. P. Simon, "Quality analysis of combined similar day and day ahead short-term load forecasting using recurrent neural networks," *Sādhanā*, vol. 47, no. 1, p. 12, 2022, https://doi.org/10.1007/s12046-022-01825-2.
[49] S. Khan *et al*., "Optimizing load demand forecasting in educational buildings using quantum-inspired particle swarm optimization QPSO with recurrent neural networks RNNs:a seasonal approach," *Scientific Reports*, vol. 15, no. 1, pp. 1–19, 2025, https://doi.org/10.1038/s41598-025-04301-z.
[50] B. Predić *et al*., "Cloud-load forecasting via decomposition-aided attention recurrent neural network tuned by modified particle swarm optimization," *Complex & Intelligent Systems*, vol. 10, no. 2, pp. 2249–2269, 2024, https://doi.org/10.1007/s40747-023-01265-3.
[51] S. Cantillo-Luna, R. Moreno-Chuquen, and J. A. Lopez Sotelo, "Intra-day Electricity Price Forecasting Based on a Time2Vec-LSTM Neural Network Model," in *2023 IEEE Colombian Conference on Applications of Computational Intelligence ColCACI*, pp. 1–6, 2023, https://doi.org/10.1109/ColCACI59285.2023.10225803.
[52] C. Wang, X. Li, Y. Shi, W. Jiang, Q. Song, and X. Li, "Load forecasting method based on CNN and extended LSTM," *Energy Reports*, vol. 12, pp. 2452–2461, 2024, https://doi.org/10.1016/j.egyr.2024.07.030.
[53] P. Deka and P. Raquel, "Novel NSGA-II optimized LSTM and GRU models for short-term forecasting of residential heating load," *Energy and Buildings*, vol. 344, p. 115999, 2025, https://doi.org/10.1016/j.enbuild.2025.115999.
[54] Z. Sheng, Z. An, H. Wang, G. Chen, and K. Tian, "Residual LSTM based short-term load forecasting," *Applied Soft Computing*, vol. 144, p. 110461, 2023, https://doi.org/10.1016/j.asoc.2023.110461.

[55] O. Rubasinghe *et al*., "A Novel Sequence to Sequence Data Modelling Based CNN-LSTM Algorithm for Three Years Ahead Monthly Peak Load Forecasting," *IEEE Transactions on Power Systems*, vol. 39, no. 2, pp. 1932–1947, 2024, https://doi.org/10.1109/TPWRS.2023.3271325.

[56] L. Oumiguil and A. Nejmi, "Short-term PV plant power forecasting model using CNN-GRU with XGBoost algorithm," *Scientific African*, vol. 31, p. e03184, 2026, https://doi.org/10.1016/j.sciaf.2026.e03184.

[57] Q. Zhang *et al*., "Multi-load forecasting for regional integrated energy systems using a hybrid ResNet-GRU-MTL model with dynamic grey wolf optimization," *Renewable Energy*, vol. 256, p. 124690, 2026, https://doi.org/10.1016/j.renene.2025.124690.

[58] G. Liu *et al*., "Medium- and long-term load reliability forecasting method of power system based on CNN-GRU hybrid neural network," *Results in Engineering*, vol. 28, p. 108263, 2025, https://doi.org/10.1016/j.rineng.2025.108263.

[59] W. Li and W. Zhang, "Hybrid GRU-EKAN model for short-term load forecasting of industrial air compressors," *Measurement*, vol. 267, p. 120631, 2026, https://doi.org/10.1016/j.measurement.2026.120631.

[60] L. Fan, Y. Wan, Q. Xiao, and K. Liu, "Correction of near-surface methane concentrations using a CNN-RF hybrid model based on multi-scale feature extraction," *Atmospheric Pollution Research*, vol. 17, no. 8, p. 103078, 2026, https://doi.org/10.1016/j.apr.2026.103078.

[61] Z. Tian, W. Liu, W. Jiang, and C. Wu, "CNNs-Transformer based day-ahead probabilistic load forecasting for weekends with limited data availability," *Energy*, vol. 293, p. 130666, 2024, https://doi.org/10.1016/j.energy.2024.130666.

[62] D. Wang, "Forecasting language proficiency levels with CNN-LSTMs and real-time speech data," *Egyptian Informatics Journal*, vol. 32, p. 100807, 2025, https://doi.org/10.1016/j.eij.2025.100807.

[63] L. Manai, W. Mchara, M. Abdellatif, and M. Raissi, "WOA-FCM-CNN-WNN-informer: An advanced hybrid deep learning model for ultra-accurate PV power forecasting in electric mobility," *Intelligent Systems with Applications*, vol. 29, p. 200630, 2026, https://doi.org/10.1016/j.iswa.2026.200630.

[64] J. Ran, Y. Gong, Y. Hu, and J. L. Cai, "EV load forecasting using a refined CNN-LSTM-AM," *Electric Power Systems Research*, vol. 238, p. 111091, 2025, https://doi.org/10.1016/j.epsr.2024.111091.

[65] J. Zhang, X. Li, and Y. Ding, "Leveraging homogeneity-fusion-based TCN for enhanced batch residential load forecasting," *Electric Power Systems Research*, vol. 259, p. 113189, 2026, https://doi.org/10.1016/j.epsr.2026.113189.

[66] J. Tian, H. Liu, W. Gan, Y. Zhou, N. Wang, and S. Ma, "Short-term electric vehicle charging load forecasting based on TCN-LSTM network with comprehensive similar day identification," *Applied Energy*, vol. 381, p. 125174, 2025, https://doi.org/10.1016/j.apenergy.2024.125174.

[67] S. Pang *et al*., "A hybrid TCN-BiLSTM short-term load forecasting model for ship electric propulsion systems combined with multi-step feature processing," *Ocean Engineering*, vol. 316, p. 119808, 2025, https://doi.org/10.1016/j.oceaneng.2024.119808.

[68] S. A. Alsamraee and S. Khanna, "A hybrid transformer – TCN – GRU based model for thermal load forecasting of a large university campus," *Energy*, vol. 344, p. 140114, 2026, https://doi.org/10.1016/j.energy.2026.140114.

[69] J. Wen, Y. Peng, W. Zhang, X. Huang, and Z. Wang, "Short-term power load forecasting based on TCN-LSTM model," in *2024 IEEE 6th International Conference on Civil Aviation Safety and Information Technology ICCASIT*, pp. 734–738, 2024, https://doi.org/10.1109/ICCASIT62299.2024.10827868.

[70] Y. Feng, J. Zhu, P. Qiu, X. Zhang, and C. Shuai, "Short-term Power Load Forecasting Based on TCN-BiLSTM-Attention and Multi-feature Fusion," *Arabian Journal for Science and Engineering*, vol. 49, no. 11, pp. 15301–15317, 2024, https://doi.org/10.1007/s13369-024-09351-5.

[71] Y. Wang, J. Chen, X. Chen, X. Zeng, and S. Member, "Short-Term Load Forecasting for Industrial Customers Based on TCN-LightGBM," *IEEE Transactions on Power Systems*, vol. 36, no. 3, pp. 1984–1997, 2021, https://doi.org/10.1109/TPWRS.2020.3028133.

[72] H. Li, S. Li, Y. Wu, Y. Xiao, Z. Pan, and M. Liu, "Short-term power load forecasting for integrated energy system based on a residual and attentive LSTM-TCN hybrid network," *Frontiers in Energy Research*, vol. 12, p. 1384142, 2024, https://doi.org/10.3389/fenrg.2024.1384142.

[73] Y. Li, Y. Ye, Y. Xu, L. Li, X. Chen, and J. Huang, "Two-stage forecasting of TCN-GRU short-term load considering error compensation and real-time decomposition," *Earth Science Informatics*, vol. 17, no. 5, pp. 4381–4397, 2024, https://doi.org/10.1007/s12145-024-01456-7.

[74] C. Fan, C. Ding, L. Xiao, F. Cheng, and Z. Ai, "Deep belief ensemble network based on MOEA/D for short-term load forecasting," *Nonlinear Dynamics*, vol. 105, no. 3, pp. 2405–2430, 2021, https://doi.org/10.1007/s11071-021-06742-3.

[75] G. F. Fan, Y. R. Liu, H. Z. Wei, M. Yu, and Y. H. Li, "The new hybrid approaches to forecasting short-term electricity load," *Electric Power Systems Research*, vol. 213, p. 108759, 2022, https://doi.org/10.1016/j.epsr.2022.108759.

[76] F. Yang, X. Fu, Q. Yang, and Z. Chu, "Decomposition strategy and attention-based long short-term memory network for multi-step ultra-short-term agricultural power load forecasting," *Expert Systems with Applications*, vol. 238, p. 122226, 2024, https://doi.org/10.1016/j.eswa.2023.122226.

[77] M. Yamasaki, R. Z. Freire, L. O. Seman, S. F. Stefenon, V. C. Mariani, and L. dos Santos Coelho, "Optimized hybrid ensemble learning approaches applied to very short-term load forecasting," *International Journal of Electrical Power & Energy Systems*, vol. 155, p. 109579, 2024, https://doi.org/10.1016/j.ijepes.2023.109579.

[78] S. Karamolegkos and D. E. Koulouriotis, "Advancing short-term load forecasting with decomposed Fourier ARIMA: A case study on the Greek energy market," *Energy*, vol. 325, p. 135854, 2025, https://doi.org/10.1016/j.energy.2025.135854.

[79] A. Kumar, A. J. Singh, and S. Kumar, "Forecasting wind speed over multiple horizons: Superiority of deep learning and decomposition-driven hybrid models," *Unconventional Resources*, vol. 9, p. 100296, 2026, https://doi.org/10.1016/j.uncres.2025.100296.

[80] Y. Liang, S. Wang, and X. Ma, "Research and Prediction Based on Multiple Linear Regression and Machine Learning," in *2024 IEEE 3rd International Conference on Electrical Engineering, Big Data and Algorithms EEBDA*, pp. 1398–1401, 2024, https://doi.org/10.1109/EEBDA60612.2024.10485891.

[81] A. Rafati, M. Joorabian, E. Mashhour, and H. R. Shaker, "Machine learning-based very short-term load forecasting in microgrid environment: evaluating the impact of high penetration of PV systems," *Electrical Engineering*, vol. 104, no. 4, pp. 2667–2677, 2022, https://doi.org/10.1007/s00202-022-01509-4.

[82] A. Irankhah, M. H. Yaghmaee, and S. Ershadi-Nasab, "Optimized short-term load forecasting in residential buildings based on deep learning methods for different time horizons," *Journal of Building Engineering*, vol. 84, p. 108505, 2024, https://doi.org/10.1016/j.jobe.2024.108505.

[83] K. Song *et al*., "Short-term load forecasting based on CEEMDAN and dendritic deep learning," *Knowledge-Based Systems*, vol. 294, p. 111729, 2024, https://doi.org/10.1016/j.knosys.2024.111729.

[84] J. Wang, L. Zhao, J. Gu, Y. Lei, and J. Su, "A hybrid model for short-term load forecasting in integrated energy systems based on adaptive signal decomposition and BResNet-transformer," *Electric Power Systems Research*, vol. 260, p. 113302, 2026, https://doi.org/10.1016/j.epsr.2026.113302.

[85] D. Venkateswaran, R. Venkateswaran, C. Shin, W. Choi, and Y. Cho, "Novel VMD-Informer-LSTM architecture for solar radiation forecasting with signal decomposition and sparse attention for smart energy applications," *Expert Systems with Applications*, vol. 305, p. 130612, 2026, https://doi.org/10.1016/j.eswa.2025.130612.

[86] K. P. Wai, C. H. Koo, Y. F. Huang, and W. C. Chong, "Enhancing water quality index forecasting: a novel approach integrating signal decomposition and deep learning models for multi-step ahead predictions," *Ain Shams Engineering Journal*, vol. 17, no. 6, p. 104133, 2026, https://doi.org/10.1016/j.asej.2026.104133.

AUTHOR BIOGRAPHY

Trung Dung Nguyen, was born in 1976 in Vietnam. He received a Master's degree in Electrical Engineering from Ho Chi Minh City University of Technology and Education, Vietnam, in 2017. He lectures at the Faculty of Electrical Engineering Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Vietnam. His research focuses on applications of metaheuristic algorithms in power system optimization, optimal control, and model predictive control.
Email: nguyentrungdung@iuh.edu.vn

Tuan Anh Nguyen, received a B.Sc. degree from the Industrial University of Ho Chi Minh City, Vietnam 2012, and an M.Sc. degree from the HCMC University of Technology and Education, Vietnam, in 2015. He is a Lecturer with the Faculty of Electrical Engineering Technology at the Industrial University of Ho Chi Minh City, Vietnam. His main research interests include power quality and load forecasting.
Email: nguyenanhtuan@iuh.edu.vn