Physics-Informed Adaptive-Weight NBeatsx for Short-Term Wind Power Forecasting

Chunyu Li, Li Sheng, *Senior Member, IEEE*, Xiaopeng Xi, and Maiying Zhong

***Abstract*—Accurate and physically interpretable wind power forecasting (WPF) is crucial for ensuring the reliable operation of power grid systems. Wind turbines have operational characteristics significantly influenced by complex environmental factors, such as wind speed fluctuations and intermittency, posing challenges for precise wind power modeling. Although deep learning models have become a promising data-driven solution in WPF, their common "closed-box" nature makes it difficult to balance forecast accuracy with the rationality of physical mechanisms. Therefore, based on the neural basis expansion analysis (NBEATSx) network architecture, this article proposes a multistep WPF method, named physics-informed adaptive-weight NBEATSx. This method realizes the deep integration of physical prior knowledge and data-driven models, providing a novel technical path for solving the joint optimization problem of accuracy and interpretability in WPF. The operational constraints of wind turbines, such as cut-in, rated, cut-out wind speeds, and rated power, are explicitly embedded into the network structure. A dynamic trainable weighting mechanism is leveraged for stack outputs, instead of the traditional aggregation strategy of direct summation. The experimental results based on a dataset of a 2-MW wind turbine show that the proposed method is significantly superior to the benchmark models and NBEATSx variants in terms of forecast accuracy and robustness.**

***Index Terms*—Interpretable neural networks, neural basis expansion analysis (NBEATSx), physics-informed machine learning (PIML), supervisory control and data acquisition (SCADA), wind power forecasting (WPF).**

Received 15 July 2025; revised 29 August 2025; accepted 9 October 2025. This work was supported in part by the National Key R&D Program of China under Grant 2023YFB3307100, in part by the National Natural Science Foundation of China under Grant 62173343, Grant 62033008, and Grant 62233012, in part by the Shandong Provincial Natural Science Foundation under Grant ZR2024MF072 and Grant ZR2025ZD02, and in part by the Research Fund for the Taishan Scholar Project of Shandong Province of China. Paper no. TII-25-4815. *(Corresponding author: Li Sheng.)*

Chunyu Li and Li Sheng are with the Shandong Provincial Engineering Research Center of Intelligent Sensing and Measurement and Control Technology, College of Control Science and Engineering, China University of Petroleum (East China), Qingdao 266580, China (e-mail: B22050006@s.upc.edu.cn; shengli@upc.edu.cn).

Xiaopeng Xi is with the Advanced Center for Electrical and Electronic Engineering, Universidad Técnica Federico Santa María, Valparaíso 2390123, Chile (e-mail: xi.xiaopeng@usm.cl).

Maiying Zhong is with the College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao 266590, China (e-mail: myzhong@sdust.edu.cn).

Digital Object Identifier 10.1109/TII.2025.3623559

I. Introduction

IN THE ongoing global transition toward a low-carbon energy system, wind power has been recognized as a key renewable resource owing to its substantial growth potential. With the development of modern wind turbines toward larger scale (with installed capacity ranging from 7 to 12 MW) and complexity, equipment operation is facing many challenges in reliability and cost-effectiveness [1]. To ensure continuous power generation, condition monitoring techniques have been established as essential support mechanisms [2], [3]. Supervisory control and data acquisition (SCADA) systems are employed to provide critical real-time information regarding turbine performance and operating history, which are often utilized to develop normal behavior models (NBMs) for the detection of performance anomalies [4]. As a class of NBM, wind power forecasting (WPF) models can provide key data support for power system planning, operation scheduling, and stability management, thereby mitigating supply–demand imbalances induced by large-scale wind integration.

From the perspective of temporal horizon, WPF methods are commonly classified into four types [5]: ultra-short-term forecasting (a few minutes ahead), short-term forecasting (several minutes to hours ahead), medium-term forecasting (several hours to a week ahead), and long-term forecasting (one week or more ahead). Among them, short-term forecasting requires high accuracy to reduce wind curtailment rates, optimize daily power generation, and optimize wind turbine maintenance schedules [6]. However, multiple technical bottlenecks continue to challenge this field. On the one hand, the difficulty of forecasting comes from the uncontrollability of the physical process itself: the significant fluctuation and intermittency of wind speed, the high nonlinearity of wind power curve (WPC), and the combined effects of threshold constraints (e.g., cut-in, rated, and cut-out wind speeds) result in complex input data characteristics. On the other hand, the insufficiency of data quality further increases the forecasting difficulty: SCADA data generally have problems, such as low sampling rate, strong noise, heterogeneity, and missing values, which weaken the model's ability to characterize the actual operating patterns [7]. In addition, the error accumulation effect in multistep forecasting and the lack of model robustness remain key factors restricting forecast accuracy [8]. Therefore, the difficulty of WPF is not a single source, but the comprehensive effect of physical uncertainty and data defects, which is also the root cause of the complexity of WPF far exceeding traditional time series forecasting tasks.

Existing data-driven WPF methods can be broadly classified into two categories: stochastic processes and machine learning (ML). Stochastic process methods, such as autoregressive moving average models and Gaussian process regression models [9], model the statistical characteristics of historical wind power or wind speed time series and capture inherent randomness and temporal correlation to some extent. These methods have low computational costs but exhibit significant limitations: on the one hand, they typically rely on stationarity and linearity assumptions, making it difficult to handle the highly complex statistical distributions in wind power monitoring data; on the other hand, 1-D input limits the utilization of multisource information, such as meteorological variables and wind turbine operating states. By contrast, ML methods have achieved superior results in WPF through flexible model structures and diverse feature inputs. Classic algorithms, including support vector machines [10], k-nearest neighbors [11], and artificial neural networks, can learn the nonlinear mapping between monitoring data and power output through kernel functions or shallow networks.

However, these above methods have limitations in modeling complex nonlinear relationships and handling high-dimensional unstructured data. Leveraging the strong nonlinear mapping capabilities and end-to-end automatic feature extraction advantages, deep learning (DL) methods can uncover the deep-seated patterns in wind speed–power conversion from massive multidimensional spatio-temporal data, thus significantly improving WPF accuracy. Recurrent architectures, such as long short-term memory (LSTM) networks [5], [11] and gated recurrent units [12], [13], used internal gating mechanisms to capture short-term sequential dependencies in time-series data. Convolutional neural networks (CNNs) were used to capture localized spatio-temporal features: 1-D/2-D convolutions enhance sensitivity to rapid fluctuations and neighborhood interactions [14], while dilated convolutions were employed by temporal convolutional networks (TCNs) to model long-range temporal dependencies in parallel, thereby mitigating the vanishing-gradient issues of recurrent models [15]. For instance, a hybrid 1-D-CNN-LSTM system [11] improved monthly forecast accuracy by integrating time series modeling with local feature extraction. To integrate spatial topology explicitly, graph neural networks (GNNs) represented wind turbines or sensor arrays as graph nodes and applied graph convolutions or attention to fuse non-Euclidean spatial correlations, overcoming the limitations of purely temporal models. In [6], a graph convolutional network was combined with the maximum information coefficient, and a multiresolution CNN with a self-attention mechanism was adopted.

Recently, Transformer-based architectures have gained considerable attention in time-series and WPF due to their ability to capture both short-range and long-range dependencies through self-attention mechanisms. Informer and its variants introduce sparse attention to address the quadratic complexity of normal Transformers, achieving scalable forecasting on long sequences [16], [17]. Autoformer further improves performance by incorporating series decomposition and an autocorrelation mechanism to better model trend and seasonal components [18], [19], [20]. Other decomposition- and spectral-based models, such as FEDformer, which combines Fourier and wavelet representations, and FFTransformer, which leverages signal decomposition and fast Fourier transform, aim to capture multiscale temporal patterns effectively [20], [21]. Moreover, hybrid frameworks have emerged to exploit complementary strengths: for example, combining TCN with Informer to extract temporal features [16] or integrating GNNs with Transformers to capture spatio-temporal dependencies across wind farms [20]. PatchTST and Crossformer proposed channel-independent patching and cross-dimensional attention, respectively, to further enhance multivariate forecasting [22], [23]. Despite these advances, deep Transformer models rely solely on data-driven mapping without integrating physical priors, such as turbine dynamics and operational constraints. The mapping mechanism without physical constraints not only reduces network interpretability and weakens model robustness under varying operating conditions, climate model changes, or distribution shifts caused by human intervention, but also undermines the reliability of forecast results in practical engineering applications [24].

In the field of power grids, physics-informed machine learning (PIML) mainly consists of four paradigms [25]. 1) Physics-informed loss functions incorporate differential equations, conservation laws, or boundary conditions as regularization terms [26], [27], [28]. To improve the robustness of WPF, the probability distribution of the WPC was used as a physical constraint [26]. In [27], an analytical physical expression of wind power output under extreme events was established to construct the error evaluation function for abrupt features. However, the additional high-order derivative calculation and nonconvex optimization increase the training difficulty and computational overhead. 2) Physics-informed initialization is achieved by pretraining weights via physical model simulations or simplifications. For example, in [29], the model was pretrained on simulated datasets generated by physical models, and then the weights were transferred to the prediction tasks with real data. In fact, the simulation-reality difference may lead to bias, resulting in interference with learning when the simulation model is inaccurate. 3) Hybrid physics-DL models refer to frameworks where DL compensates for unmodeled terms in physical equations [30], or integrates "glass-box" physical model outputs with "closed-box" neural networks through weighted summation or ensemble learning [31]. In [31], medium-term WPF was achieved through the massive expansion of the predictor space defined by weather model outputs and signal selection. However, how to adaptively adjust the weights of the two under different working conditions is still the main problem. In this direction, combining adaptive state-space estimation methods with ML predictors has also been explored [32]. 4) Physics-aware architecture design directly embeds power grid topology, power flow equations, or intermediate physical quantities into network layers and connections, such as hardcoding adjacency matrices in GNNs or assigning voltage or power meanings to hidden neurons [33], [34]. For instance, the partial differential equations (PDE)-assisted network proposed in [35] embedded wind speed evolution PDE into its architecture by designing modules (e.g., interaction units, adaptive frequency gating units) that endowed layer connections with clear physical

meaning via PDE dynamics. However, this paradigm requires in-depth domain knowledge, and the architecture often needs to be redesigned when the grid topology or operating conditions change.

From this discussion, it is evident that achieving deep fusion between physical knowledge and data-driven models remains the core proposition for PIML. Neural basis expansion analysis (NBEATSx) [36] offered a promising foundation for constructing a forecast framework with interpretability. Its unique modular architecture generated forecast results by an additive combination of interpretable basis functions, which could intuitively decompose time-series features, such as trend and seasonality in data. However, in the WPF scenario, the pure data-driven paradigm and rigid structure of NBEATSx gradually show significant limitations. On the one hand, the architecture of NBEATSx employs a static weight aggregation strategy, where the outputs of all blocks are summed into the final prediction with equal weights, leading to insufficient dynamic adaptability of the model to the complex time-series patterns in wind power data. On the other hand, the basis functions design relies on polynomial or Fourier expansion, and lacks explicit modeling of the physical characteristics of wind turbines, resulting in the deviation of forecast values from the actual physical evolution trajectory, especially under extreme wind speed conditions. Therefore, this article develops a high-precision forecast framework for physical constraints and data-driven collaborative optimization. The main contributions of this article are as follows.

1) Replacing the fixed accumulation strategy with a trainable weight matrix enables the model to dynamically adjust the contribution of each stack output according to input time-series features, enhancing adaptability to wind speed patterns.
2) A group of smooth logistic growth models (SLGMs) is used to fit the WPC, and operating parameter constraints are explicitly embedded into the basis functions to ensure the outputs satisfy the nonlinear saturation characteristics of wind turbine operating domains.
3) Transformer encoders are leveraged to capture complex temporal dependencies for generating basis vectors, and residual connections are incorporated to strengthen historical information flow.

The rest of this article is organized as follows. Section II elaborates on the detailed implementation of the proposed WPF approach. Section III presents the experimental results and corresponding analysis. Finally, Section IV concludes this article.

II. Methodology

A. PIAW-NBEATSx Network Structure

This study adopts the multilevel "stack-block" architecture of the NBEATSx model [36], integrating exogenous variables to achieve time series decomposition and forecasting. Blocks serve as the fundamental structural units of this network. All blocks process input data following the same logic, with multiple blocks connected to form a stack via the doubly residual stacking principle. For instance, the input to the nth block ($n = 1, 2, \ldots, N$) within the mth stack ($m = 1, 2, \ldots, M$) comprises two components: the dynamic residual component

$$y_{m,n}^{\text{back}} = \hat{y}_{m,n-1}^{\text{back}} - y_{m,n-1}^{\text{back}} \in \mathbb{R}^{B\times L_1} \tag{1}$$

where B denotes the batch size, L_1 is the length of the backcast window, and the static exogenous variable component $X \in \mathbb{R}^{B\times C\times L_1}$, where C represents the number of exogenous variables. It then outputs two vectors: the block's backcast $\hat{y}_{m,n}^{\text{back}} \in \mathbb{R}^{B\times L_1}$ and forecast $\hat{y}_{m,n}^{\text{for}} \in \mathbb{R}^{B\times L_2}$, where L_2 represents the forecast window length. Doubly residual stacking realizes the structured decomposition of time series by stripping the modeled signal components layer by layer, which is one of the core designs of the NBEATSx model.

Each block contains a multilayer fully connected neural network (FCNN) responsible for learning basis expansion coefficients $\theta_{m,n} = [\theta_{m,n}^{\text{back}}, \theta_{m,n}^{\text{for}}] \in \mathbb{R}^{B\times 2D}$

$$\theta_{m,n} = \text{Linear}\left(\text{FCNN}_{m,n}\left(\hat{y}_{m,n-1}^{\text{back}}, X\right)\right) \tag{2}$$

where D is the number of forecast bases or backcast bases, which depends on the type of stack basis: if the basis is learned from exogenous variables, then $D = D_n$ (the number of selected external covariates), whereas if the basis is predefined (e.g., a set of basis functions), then $D = D_f$ (the number of basis functions). The network incorporates backcast basis $V_{m,n}^{\text{back}} \in \mathbb{R}^{B\times D\times L_1}$ and forecast basis $V_{m,n}^{\text{for}} \in \mathbb{R}^{B\times D\times L_2}$, which can be either predefined basis functions or basis vectors obtained by data learning. Within the basis layers, for each sample $b (b = 1, \ldots, B)$, explicit summation maps the basis expansion coefficients $\theta_{m,n}^{\text{back}} \in \mathbb{R}^{B\times D}$ and $\theta_{m,n}^{\text{for}} \in \mathbb{R}^{B\times D}$ to the backcast $\hat{y}_{m,n}^{\text{back}}$ and forecast $\hat{y}_{m,n}^{\text{for}}$ via the backcast basis $V_{m,n}^{\text{back}}$ and forecast basis $V_{m,n}^{\text{for}}$, respectively. This process can be expressed as follows:

$$\hat{y}_{m,n}^{\text{back}}(b,t) = \sum_{p=1}^{D} \theta_{m,n}^{\text{back}}(b,p) \cdot V_{m,n}^{\text{back}}(b,p,t), t = 1, \ldots, L_1$$

$$\hat{y}_{m,n}^{\text{for}}(b,t) = \sum_{p=1}^{D} \theta_{m,n}^{\text{for}}(b,p) \cdot V_{m,n}^{\text{for}}(b,p,t), t = 1, \ldots, L_2.$$

It can be seen that the basis expansion coefficients $\theta_{m,n}$ essentially assign weights to each basis. These weights are continuously adjusted during training, enabling the model to adaptively combine components, and expand the basis into final forecast results, thus achieving strong fitting performance and interpretability.

Multiple blocks constitute a stack, with each stack specializing in a specific basis type, such as the typical trend basis functions and seasonal basis functions in the NBEATSx model. The input to the mth stack is the backcast $\hat{y}_{m,N}^{\text{back}}$. The forecast output by the stack aggregates the forecasts from each block within it

$$\hat{y}_{m}^{\text{for}} = \sum_{n=1}^{N} \hat{y}_{m,n}^{\text{for}} \in \mathbb{R}^{B\times L_2}. \tag{3}$$

Ultimately, the global output of the model is the sum of all stack forecast results.

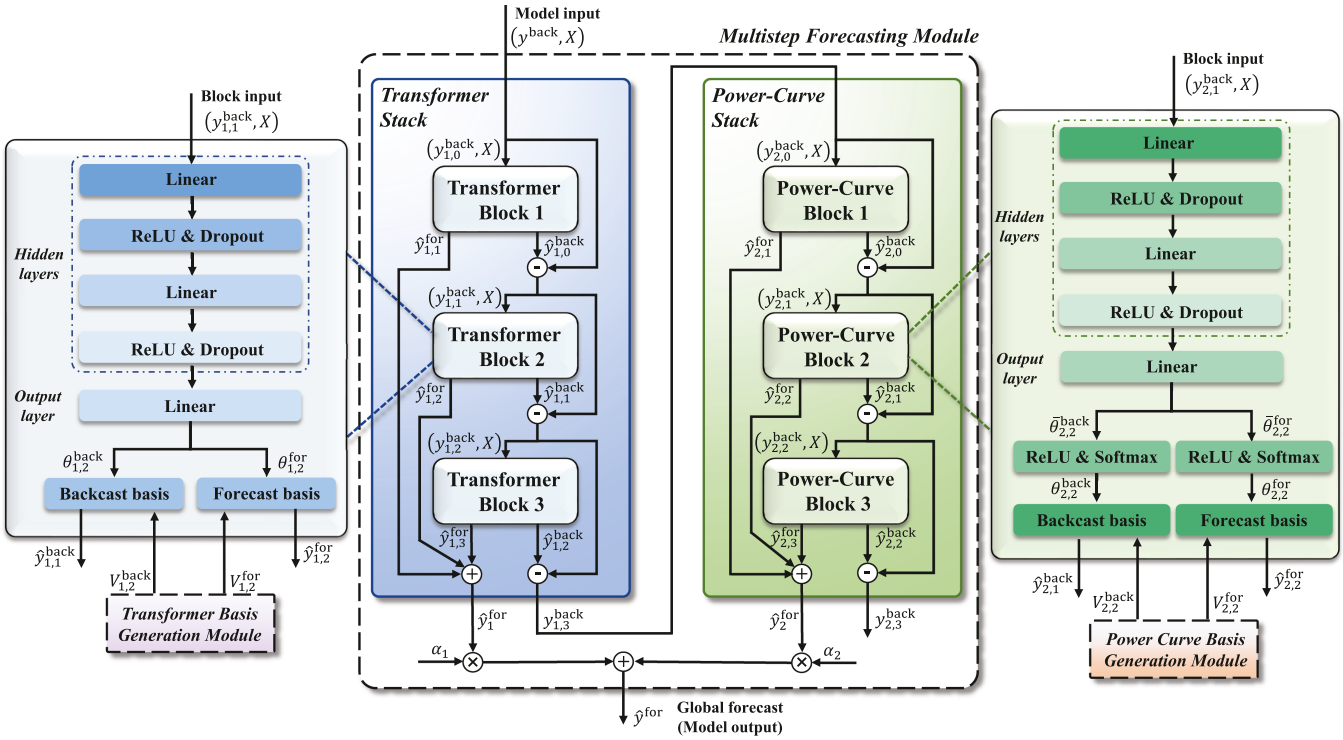

Fig. 1. Network structure details of PIAW-NBEATSx, derived from the multilevel "stack-block" architecture of NBEATSx. The network comprises two stacks: the Transformer stack (blue box), with basis vectors generated by the Transformer Basis Generation Module; the power-curve stack (green box), with predefined basis functions generated by the power curve basis generation module. Each block contains a multilayer FCNN tasked with learning basis expansion coefficients $\theta_{m,n}$. The global forecast $\hat{y}^{\text{for}}$ is constructed via adaptive-weights α_m, adaptively combining the forecasts from all stacks.

By virtue of the superiority of doubly residual stacking and multilevel "stack-block" architecture, NBEATSx can achieve residual propagation and multicomponent separation modeling of time series, thereby combining the nonlinear modeling capability of DL with the interpretability of traditional time series decomposition. However, directly using the original NBEATSx network makes it difficult to achieve accurate WPF. The reasons are that existing basis functions (such as polynomials and Fourier series) have limited expressive power and lack prior knowledge to constrain the training process; furthermore, the static weight aggregation strategy leads to insufficient dynamic adaptability of the model to complex time series patterns in wind power data.

As shown in Fig. 1, the physics-informed adaptive-weight NBEATSx (PIAW-NBEATSx) is proposed, where the global forecast, i.e., model output is the weighted sum of the forecasts of all stacks

$$\hat{y}^{\text{for}} = \sum_{m=1}^{M} \alpha_m \hat{y}_m^{\text{for}} \in \mathbb{R}^{B \times L_2} \tag{4}$$

where α_m are trainable weight coefficients measuring the contribution of each stack to the global forecast (i.e., the final integrated model output). In this way, the model adaptively integrates the physics-informed power-curve stack and the data-driven Transformer stack, so that the global forecast not only inherits the physical consistency of turbine operation, but also captures complex temporal dependencies in wind speed dynamics. This adaptive fusion mechanism ensures that the final forecast is both accurate and physically interpretable.

B. Power Curve Basis Generation

As the core operating characteristic curve of the wind turbine, WPC provides an intuitive depiction of the nonlinear mapping between the input wind speed and the output electric power. It is essentially determined by aerodynamic principles, the electromagnetic properties of the generator, and turbine control logic. As shown in Fig. 2, the nominal WPC reveals three key physical phenomena: 1) Below the cut-in threshold (3 m/s), the turbine remains offline, and output power approaches zero; 2) Between cut-in and rated speeds (3–10 m/s), power increases nonlinearly with pronounced gradient changes; and 3) Above the rated wind speed (10 m/s), power saturates at the nominal capacity (2 MW). This piecewise saturation behavior renders traditional parameteric models (e.g., polynomial regression and sigmoid functions) not flexible enough to capture its true complexity. Moreover, the outliers marked as in Fig. 2(a)–(c) also complicate the process of accurate modeling.

Given this, we employ an SLGM to capture the nonlinear characteristics of the WPC, with its mathematical expression given by

$$\hat{P}(v) = \frac{P^{\max}}{1 + e^{-k(v - v^{\text{half}})}} \cdot \frac{1}{1 + e^{-v^{\text{rated}}(v - v^{\text{cutin}})}} \tag{5}$$

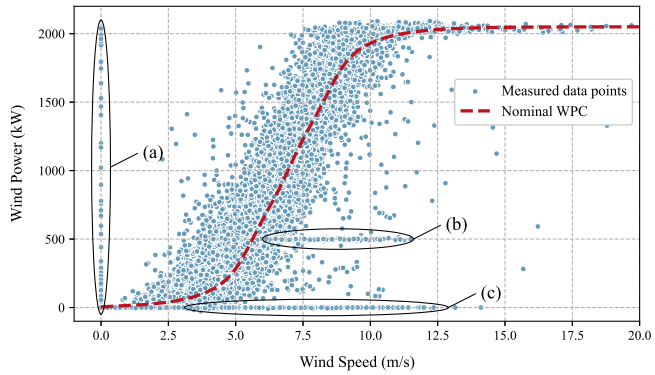

Fig. 2. Wind speed–power scatter distribution for the actual operating points of a 2-MW wind turbine is presented, and the nominal WPC is marked by the red dotted line. Except for the main operating points, the measured data points contain scattered and clustered outliers, such as (a) anemometer malfunctions, (b) shutdown events, and (c) horizontal clusters induced by curtailment strategies.

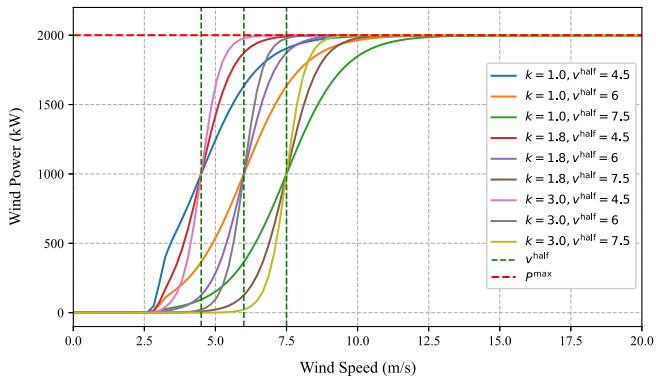

Fig. 3. Shape comparison of SLGMs under different parameters.

where $\hat{P}(v)$ denotes the predicted power at wind speed v, $P^{\max}$ represents the maximum power output (typically the rated power), k is the steepness parameter controlling the rate at which the curve ascends, v^{half} is the wind speed at half-maximum power (i.e., the power output is $P^{\max}/2$ at this wind speed), and v^{cutin} and v^{rated} are the cut-in and rated wind speeds, respectively.

By combining a modified logistic structure with a steepness parameter, we construct the SLGM in (5) using two functional components. Specifically, the first component describes the smooth increase of power with wind speed v, while the second component ensures that the power remains zero before the cut-in wind speed v^{cutin} and stabilizes after the wind speed reaches the rated value v^{rated}. Together, these components enable (5) to generate a curve with a smooth transition in the low wind speed region and gradual steepening as v increases.

The shape of the curve can be adjusted by tuning the key parameters of the SLGM while keeping other parameters constant. For example, setting the parameter k to $K = \{1.0, 1.8, 3.0\}$ and v^{half} to $V^{\text{half}} = \{4.5, 6.0, 7.5\}$, with the number of elements in the two sets denoted as n_k and n_v respectively, yields a set of curves, as shown in Fig. 3.

As illustrated in Fig. 4, the process diagram of power curve basis generation module comprises three core stages as follows.

1) Parameter Processing and Dimension Adjustment: Let $v^{\text{in}}_{m,n} \in \mathbb{R}^{B\times L_1}$ and $v^{\text{out}}_{m,n} \in \mathbb{R}^{B\times L_2}$, respectively, represent the wind speed data in the backcast period and forecast period.

First, the two input components are concatenated along the temporal dimension

$$v_{m,n} = \text{cat}\left(v^{\text{in}}_{m,n}, v^{\text{out}}_{m,n}\right) \in \mathbb{R}^{B\times L}$$

where $L = L_1 + L_2$.

Since $v_{m,n}$ are standardized sequences, the parameters in (5) are adjusted based on the rated cutout wind speed v^{cutout} to scale the horizontal (wind speed) and vertical (power) coordinates of the SLGMs to the range [0,1]

$$\begin{aligned} P\left(v_{m,n}\right) &= L\left(v_{m,n}\right)\cdot N\left(v_{m,n}\right) \\ &= \frac{1}{1+e^{-\bar{k}\left(v_{m,n}-\bar{v}^{\text{half}}\right)}}\cdot\frac{\bar{P}^{\max}}{1+e^{-\bar{v}^{\text{rated}}\left(v_{m,n}-\bar{v}^{\text{cutin}}\right)}} \end{aligned} \tag{6}$$

where $\bar{v}^{\text{cutin}} = v^{\text{cutin}}/v^{\text{cutout}}$, $\bar{P}^{\max} = 1$, $\bar{v}^{\text{rated}} = v^{\text{rated}} \times v^{\text{cutout}}$, $\bar{k} \in \bar{K}$, $\bar{K} = K \times v^{\text{cutout}}$, $\bar{v}^{\text{half}} \in \bar{V}^{\text{half}}$, and $\bar{V}^{\text{half}} = V^{\text{half}}/v^{\text{cutout}}$.

To facilitate tensor broadcasting in the DL framework, the dimensions of parameters $\bar{k}$, $\bar{v}^{\text{half}}$, and the input sequence $v_{m,n}$ are adjusted as follows:

$$\begin{aligned} \tilde{k} &= \text{reshape}\left(\bar{k}\right) \in \mathbb{R}^{1\times1\times n_k\times1\times1} \\ \tilde{v}^{\text{half}} &= \text{reshape}\left(\bar{v}^{\text{half}}\right) \in \mathbb{R}^{1\times1\times1\times n_v\times1} \\ \tilde{v}_{m,n} &= \text{reshape}\left(v_{m,n}\right) \in \mathbb{R}^{B\times L\times1\times1\times1}. \end{aligned}$$

2) Construct the Power Curve Basis Functions: The SLGMs are computed via elementwise broadcasting in tensor computation

$$L\left(\tilde{v}_{m,n}\right) = \frac{1}{1+e^{-\tilde{k}\left(\tilde{v}_{m,n}-\tilde{v}^{\text{half}}\right)}} \in \mathbb{R}^{B\times L\times n_k\times n_v\times1}. \tag{7}$$

The third and fourth dimensions (corresponding to n_k and n_v) are merged, and singleton dimensions are removed to obtain

$$\tilde{L}\left(\tilde{v}_{m,n}\right) = \text{reshape}\left(\text{flatten}\left(L\left(\tilde{v}_{m,n}\right), \text{dim} = 2, 3\right)\right) \tag{8}$$

where $\tilde{L}(\tilde{v}_{m,n}) \in \mathbb{R}^{B\times L\times D}$, and $D = n_k \times n_v$.

Computing the nonlinear scaling term $N(v_{m,n})$ in (6) and expanding its dimensions yields

$$\tilde{N}\left(v_{m,n}\right) = \text{reshape}\left(\frac{\bar{P}^{\max}}{1+e^{-\bar{v}^{\text{rated}}\left(v_{m,n}-\bar{v}^{\text{cutin}}\right)}}\right) \in \mathbb{R}^{B\times L\times1}. \tag{9}$$

The final basis function matrix is then obtained as

$$V_{m,n} = \text{permute}\left(\tilde{L}\left(\tilde{v}_{m,n}\right)\odot\tilde{N}\left(v_{m,n}\right), (0,2,1)\right) \tag{10}$$

where $\odot$ denotes elementwise multiplication operation, and $V_{m,n} \in \mathbb{R}^{B\times D\times L}$. Specifically, permute$(\cdot)$ swaps the second and third dimensions of the tensor from $[B, L, D]$ to $[B, D, L]$.

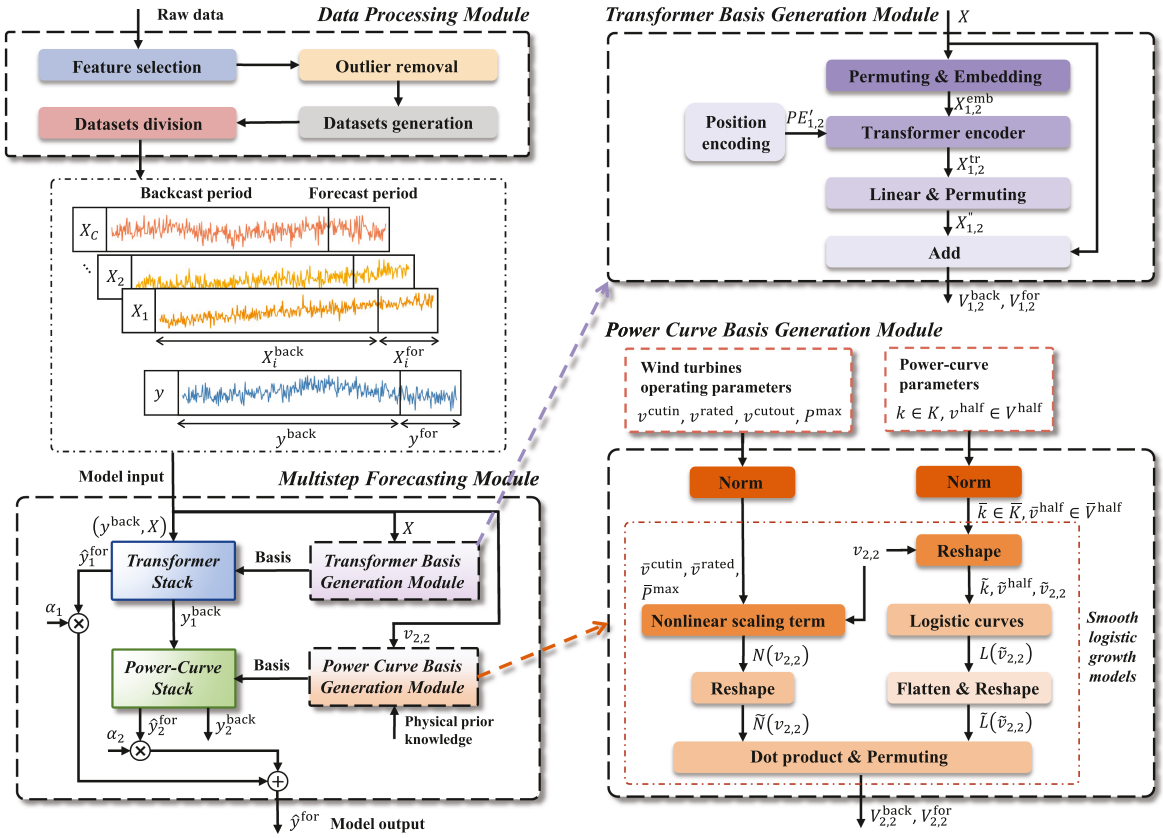

Fig. 4. Overall framework of the short-term WPF method, as well as the details of the generation modules for Transformer basis vectors and power curve basis functions. Key parameters, including cut-in, rated, and cut-out wind speeds, together with rated power (i.e., v^{cutin}, v^{rated}, v^{cutout}, and $P^{\max}$), are explicitly embedded into the power curve basis functions via SLGMs.

Finally, $V_{m,n}$ is split into backcast and forecast basis functions via tensor slicing

$$\begin{cases} V_{m,n}^{\text{back}} = V_{m,n}\left[:,:,1:L_1\right] \in \mathbb{R}^{B\times D\times L_1} \\ V_{m,n}^{\text{for}} = V_{m,n}\left[:,:,L_1+1:L\right] \in \mathbb{R}^{B\times D\times L_2}. \end{cases} \tag{11}$$

3) Basis Expansion Coefficients Processing: For the power-curve block, because the basis expansion coefficients output by the FCNN represent the weights for SLGMs, $\bar{\theta}_{m,n}^{\text{back}}$ and $\bar{\theta}_{m,n}^{\text{for}}$ are first processed using the ReLU activation function to ensure nonnegativity and then normalized via softmax (illustrated in Fig. 1)

$$\theta_{m,n} = \text{softmax}\left(\text{ReLU}\left(\bar{\theta}_{m,n}\right), \text{dim} = -1\right) \in \mathbb{R}^{B\times 2D} \tag{12}$$

where $\text{dim} = -1$ denotes normalization along the last dimension of the tensor. The result is then split into backcast basis expansion coefficients $\theta_{m,n}^{\text{back}} \in \mathbb{R}^{B\times D}$ and forecast basis expansion coefficients $\theta_{m,n}^{\text{for}} \in \mathbb{R}^{B\times D}$.

C. Transformer Basis Generation

As shown in Fig. 4, the generation steps of the Transformer basis include three core stages as follows.

1) Embedding and Positional Encoding: Given input tensors $X^{\text{in}} \in \mathbb{R}^{B\times C\times L_1}$ (backcast period) and $X^{\text{out}} \in \mathbb{R}^{B\times C\times L_2}$ (forecast period), the tensors are first concatenated along the temporal dimension to form $X \in \mathbb{R}^{B\times C\times L}$. Following standard Transformer practice [37], the tensor is transposed to $\mathbb{R}^{B\times L\times C}$ and subjected to linear projection, yielding $X_{m,n}^{\text{emb}} \in \mathbb{R}^{B\times L\times d_{m,n}}$

$$X_{m,n}^{\text{emb}} = \text{Linear}(\text{permute}\left(X_{m,n}, (0,2,1)\right)) \cdot \sqrt{d_{m,n}} \tag{13}$$

where $d_{m,n}$ denotes the embedding dimension and the scaling factor $\sqrt{d_{m,n}}$ stabilizes gradient magnitudes.

Temporal position information is incorporated into the embedded features using sinusoidal encoding functions, as in [37]. The positional encoding matrix $\text{PE}_{m,n} \in \mathbb{R}^{l\times d_{m,n}}$ is precomputed offline, where l denotes the maximum precomputed position encoding length, and dynamically truncated to align with the sequence length L. The final encoded representation is obtained as

$$\bar{X}_{m,n} = \text{Dropout}(X_{m,n}^{\text{emb}} + PE_{m,n}[0:L,:]). \tag{14}$$

2) Transformer Encoder: The Transformer encoder layer is applied, where each layer contains the multihead self-attention, residual connection, and a positionwise feedforward network (FFN). This process is expressed as

$$X_{m,n}^{\text{tr}} = \text{TransformerEncoder}\left(\bar{X}_{m,n}, \bar{M}\right) \tag{15}$$

where $\bar{M} \in \mathbb{R}^{L\times L}$ is a lower triangular causal mask defined as

$$\bar{M}_{ij} = \begin{cases} 0, & \text{if } i \geq j \\ -\infty, & \text{if } i < j \end{cases}$$

which is employed to mask future time steps and preserve causality in the attention mechanism.

3) Output Projection and Residual Connections: An output projection layer maps the Transformer output $X^{\text{tr}}_{m,n}$ from dimension $d_{m,n}$ to the target dimension D (where $D = C$ for blocks with data-learned basis vectors)

$$X^{\text{proj}}_{m,n} = \text{Linear}\left(X^{\text{tr}}_{m,n}\right) \in \mathbb{R}^{B\times L\times D}. \tag{16}$$

Through joint permutation and residual connection operations, the fused representation is derived as

$$V_{m,n} = \text{permute}\left(X^{\text{proj}}_{m,n}, (0,2,1)\right) + X_{m,n} \in \mathbb{R}^{B\times D\times L}. \tag{17}$$

III. Experiments

A. Dataset and Performance Metrics

To validate the performance of the forecasting models, this study utilizes historical data from the entire year of 2023, sourced from a SCADA system in a Chinese wind farm. The dataset contains 37 variables and has a 15-min sampling frequency. As illustrated in Fig. 4, the dataset is constructed through sequential procedures. First, feature selection is implemented by computing a correlation matrix using Pearson correlation coefficients. For groups of variables with approximate perfect collinearity, only one representative variable is retained. This process results in the selection of 13 variables as static exogenous variables X, including wind speed, phase voltage, power factor, blade motor current, blade angle, rotor blade speed, gearbox oil temperature, generator winding temperature, generator drive/nondrive end bearing temperature, environmental temperature, and converter machine/grid side module temperature. Subsequently, physical outliers are removed, and the data are normalized to the range of [0, 1] to complete the preliminary preprocessing.

To evaluate the accuracy of forecasting models, performance metrics, including mean absolute percentage error (MAPE), symmetric MAPE (SMAPE), normalized mean absolute error (NMAE), normalized root-mean-squared error, and coefficient of determination (R^2), are adopted.

B. Experiments Settings and Configurations

In our experiments, the input to the model is arranged as a tensor of shape (B, L_1, D_X), where the batch size is $B = 256$, and the feature dimension per time step is $D_X = 14$. The backcast window length (time steps) L_1 is set to 5 times the forecast window length L_2, with the settings being $L_1 = 25$ and $L_2 = 5$. The dataset is divided into training, validation, and test sets at a ratio of 6:2:2. We train the PIAW-NBEATSx network on the training set using the Adam optimizer with mean squared error (MSE) as the loss function. Hyperparameters are tuned via grid search over three groups: architecture (number of blocks per stack n_{b}, number of fully connected layers per block n_{l}, and hidden dimension per layer n_{h}), regularization (dropout rate d, L1 regularization coefficient λ_1, and L2 regularization coefficient λ_2), and learning rate schedule (initial rate η, decay factor γ, and minimum loss reduction threshold ε). Validation performance drives parameter selection. The optimized network, configured with $n_{\text{b}} = 3$, $n_{\text{l}} = 3$, $n_{\text{h}} = 64$, $d = 0.2$, $\lambda_1 = 10^{-5}$, $\lambda_2 = 0.001$, $\eta = 0.002$, $\gamma = 0.5$, and $\varepsilon = 10^{-5}$, is detailed in Table I. In this setup, each stack comprises three blocks, and every blocks multilayer perceptron (MLP) uses the same hidden dimension. Note that, although we use uniform block counts and dimensions here, one could vary these across stacks or blocks. After training, the model generates five-step forecasts on the test set, which are concatenated to form the full forecast sequence. All experiments are implemented in Python 3.9 and run on an NVIDIA RTX A4000 GPU.

TABLE I
ARCHITECTURE OF THE PIAW-NBEATSX NETWORK

Stack type	Block location	Component type	Parameter configuration	Output Dim.
Transformer stack (3 Blocks)	Transformer blocks 1–3	FNN	Linear(350→64)→ReLU→ Dropout→Linear(64→64)→ ReLU→Dropout→ Linear(64→26)	26
	Transformer basis	Transformer	-	13
		Embedding	Linear(13→64)	64
		Positional encoding	Dropout(0.1)	64
		Transformer encoder (four layers)	For each layer: MultiheadAttention(64→64)→ Dropout→LayerNorm(10^{-5})→ Linear(64→2048)→ReLU→ Linear(2048→64)→Dropout→ LayerNorm(10^{-5})	64
Power-Curve stack (3 Blocks)	Power-Curve blocks 1–3	FNN	Linear(350→64)→ReLU→ Dropout→Linear(64→64)→ ReLU→Dropout→ Linear(64→18)	18
	Power Curve basis	Logistic Basis Functions	-	9

C. Comparisons of Deterministic Forecasting Results

Four benchmark forecasting models are established: the original NBEATSx and three sequence-to-sequence variants (Bi-LSTM, TCN, and Transformer). All models share identical input/output dimensions (input sequence length $L_1 = 25$, input feature dimension $D_X = 14$, and output sequence length $L_2 = 5$) and incorporate a residual shortcut from raw input to output. The NBEATSx model replicates the architecture in Table I but uses classical Trend and Seasonality stacks. Hyperparameters of all models are optimized via grid searches over hidden size, number of layers, dropout rate, learning rate, and batch size within a unified search space, employing the same loss function and early stopping strategy. Key hyperparameters are summarized in Table II.

Table III compares the PIAW-NBEATSx with benchmarks across five metrics. Notably, the benchmark models exhibit extremely high SMAPEs, owing to the strong volatility of real data

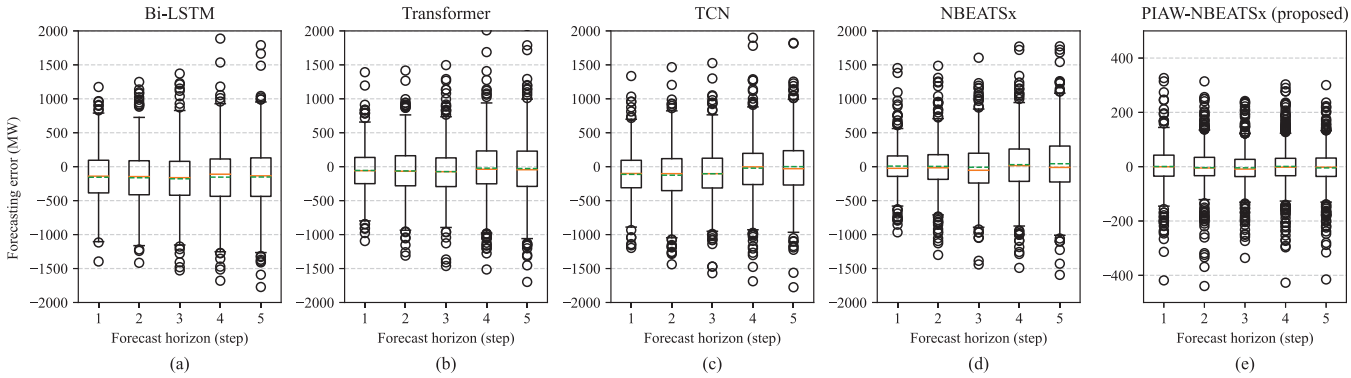

Fig. 5. Comparison of the distributions of forecast errors at each forecast step. (a) Bi-LSTM, (b) Transformer, (c) TCN, (d) NBEATSx, (e) PIAW-NBEATSx (proposed).

TABLE II
HYPERPARAMETER CONFIGURATIONS OF BENCHMARK MODELS

Model	Parameters setting
Bi-LSTM	Hidden units N_{lstm}: 64; number of LSTM layers M_{lstm}: 2; MLP branch: $(2 \times N_{\text{lstm}} \rightarrow N_{\text{lstm}} \rightarrow N_{\text{lstm}}/2 \rightarrow 1)$; batch size B_{lstm}: 64; learning rate: 0.01; dropout rate: 0.3
Transformer	Hidden dimension N_{tf}: 64; number of encoder layers M_{tf}: 2; MLP branch: $(N_{\text{tf}} \rightarrow N_{\text{tf}} \rightarrow N_{\text{tf}}/2 \rightarrow 1)$; batch size B_{tf}: 64; FFN intermediate dimension: $4 \times N_{\text{tf}}$; number of multiple attention heads: 4; learning rate: 0.005; dropout rate: 0.3
TCN	Hidden channels N_{tcn}: 16; number of temporal blocks M_{tcn}: 2; MLP branch: $(N_{\text{tcn}} \rightarrow N_{\text{tcn}} \rightarrow N_{\text{tcn}}/2 \rightarrow 1)$; batch size B_{tcn}: 64; kernel size K_{tcn}: 3; dilation rate: 2^i; learning rate: 0.01; dropout rate: 0.3

TABLE III
PERFORMANCE COMPARISON OF FORECASTING MODELS

Model	SMAPE(%)	NMAE(%)	NMSE(%)	R^2
Bi-LSTM	39.8902	14.4045	3.8014	0.6716
Transformer	40.1339	14.5085	3.7827	0.6735
TCN	38.5865	14.0081	3.7432	0.6769
NBEATSx	38.7019	14.0270	3.6111	0.6883
PI-NBEATSx	9.8081	2.8602	0.2071	0.9821
PIAW-NBEATSx	**9.4423**	**2.5651**	**0.1487**	**0.9872**

and the prevalence of near-zero wind-power values (corresponding to wind turbine shutdown states). Benchmark models have limited capabilities in capturing complex temporal dependencies and multiscale patterns, leading to substantial large forecast error accumulation on low-value samples. By contrast, NBEATSx partially mitigates this issue through its interpretable stacked architecture and signal-decomposition mechanism.

The proposed PIAW-NBEATSx demonstrates significant superiority across all evaluation metrics, achieving the lowest error rates and the highest R^2. This superior performance stems from the novel power-curve stack that explicitly embeds operational parameter constraints into the basis functions, ensuring forecasts comply with the nonlinear saturation characteristics of the WPC. In addition, the integration of a trainable weighting scheme and Transformer stack further enhances forecasting accuracy and stability.

Overall, these comparative results validate the necessity of incorporating prior knowledge for WPF. The proposed method not only improves the interpretability and reliability of forecasting results, but also demonstrates exceptional accuracy and robustness when handling real-world datasets with numerous near-zero samples.

Fig. 5 visualizes the forecast error distribution via boxplots for all models across each forecast steps. As the horizon extends, interquartile ranges broaden and whiskers lengthen, reflecting increasing uncertainty in long-term forecasts. In contrast, PIAW-NBEATSx produces notably compact boxplots with tightly clustered outliers, and its median and mean values (solid and dashed lines) almost coincide at zero, demonstrating minimal bias and high stability. Benchmark models, especially Bi-LSTM and TCN, exhibit substantially wider boxplots and more dispersed outliers, highlighting their greater instability and susceptibility to extreme errors.

As shown in Table IV, this is a comparison of the time and space complexity for the considered models. The definitions of all symbols are provided in Table II and in the description of the experimental settings. Compared with the classical NBEATSx, the proposed PIAW-NBEATSx has higher time and space complexity. This is mainly due to the additional Transformer basis generation module in each block, which introduces extra computations from self-attention and feedforward layers.

D. Discussions of the Proposed Method

1) Interpretability Analysis: Fig. 6(a) compares the global forecast $\hat{y}^{\text{for}}$ produced by the PIAW-NBEATSx network with the observed power curve. Fig. 6(b) and (c) shows two forecast components from the Transformer stack forecast $\hat{y}_1^{\text{for}}$ and the power-curve stack forecast $\hat{y}_2^{\text{for}}$. The evolution of their dynamic weight coefficients α_1 and α_2 during training is plotted in Fig. 7(a). Starting from 1, these weights converge to $\alpha_1 = 0.821$ and $\alpha_2 = 0.324$, indicating that the Transformer stack contributes more significantly to the global forecast than the power-curve stack. Therefore, the temporal dependence modeling plays a dominant role in forecasting, while the stack guided by physical

TABLE IV
COMPARISON OF TIME AND SPACE COMPLEXITY FOR THE CONSIDERED MODELS

Model	Time complexity	Space complexity
Bi-LSTM	$\mathcal{O}\left(M_{\text{lstm}} B_{\text{lstm}} L_1 N_{\text{lstm}}^2\right)$	$\mathcal{O}\left(M_{\text{lstm}} N_{\text{lstm}}^2 + M_{\text{lstm}} B_{\text{lstm}} L_1 N_{\text{lstm}}\right)$
Transformer	$\mathcal{O}\left(M_{\text{tf}} B_{\text{tf}} L_1^2 N_{\text{tf}} + M_{\text{tf}} B_{\text{tf}} L_1 N_{\text{tf}} D_{\text{ff}}\right)$	$\mathcal{O}\left(M_{\text{tf}} N_{\text{tf}} D_{\text{ff}} + M_{\text{tf}} B_{\text{tf}} L_1 D_{\text{ff}}\right)$
TCN	$\mathcal{O}\left(M_{\text{tcn}} B_{\text{tcn}} L_1 N_{\text{tcn}}^2 K_{\text{tcn}}\right)$	$\mathcal{O}\left(M_{\text{tcn}} N_{\text{tcn}}^2 K_{\text{tcn}} + M_{\text{tcn}} B_{\text{tcn}} L_1 N_{\text{tcn}}\right)$
NBEATSx	$\mathcal{O}\left(n_{\text{b}} B L_1 D_X n_{\text{h}} + n_{\text{b}} n_{\text{l}} B n_{\text{h}}^2\right)$	$\mathcal{O}\left(n_{\text{b}} L_1 D_X n_{\text{h}} + n_{\text{b}} n_{\text{l}} n_{\text{h}}^2 + n_{\text{b}} B n_{\text{l}} n_{\text{h}}\right)$
PIAW-NBEATSx	$\mathcal{O}\left(n_{\text{b}} B L_1 D_X n_{\text{h}} + n_{\text{b}} n_{\text{l}} B n_{\text{h}}^2 + n_{\text{b}} n_t B L_1 d d_{\text{ff}}\right)$	$\mathcal{O}\left(n_{\text{b}} L_1 D_X n_{\text{h}} + n_{\text{b}} n_{\text{l}} n_{\text{h}}^2 + n_{\text{b}} n_t B d_{\text{ff}}\right)$

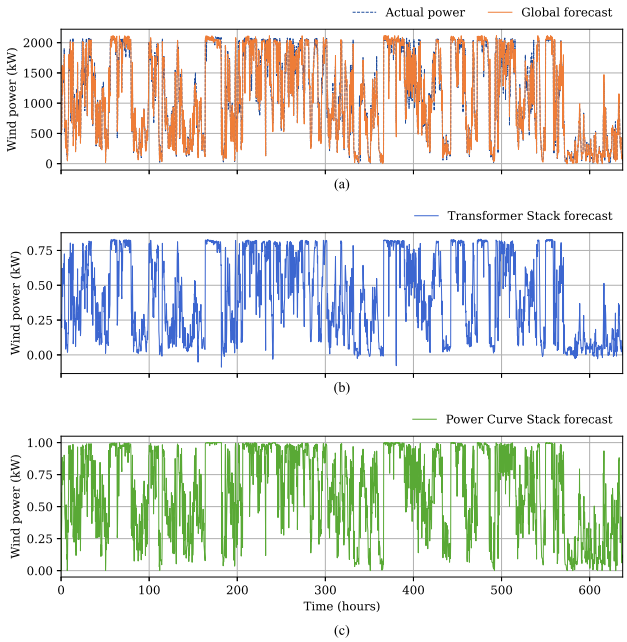

Fig. 6. Global forecast $\hat{y}^{\text{for}}$ and its decomposed forecast components include the Transformer stack forecast $\hat{y}_1^{\text{for}}$ and the power-curve stack forecast $\hat{y}_2^{\text{for}}$.

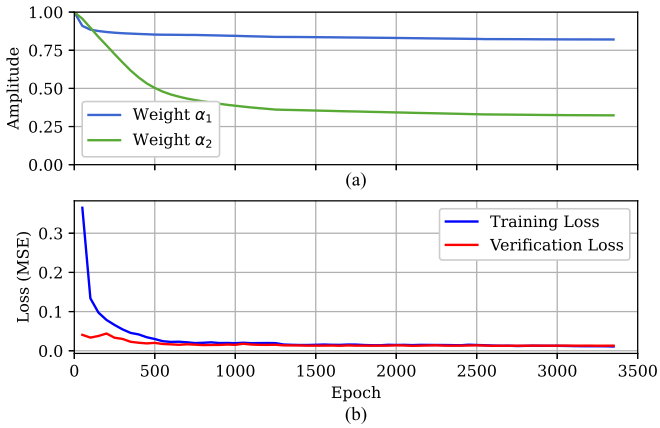

Fig. 7. Variation curves of stack weight coefficients α_m (a) and the loss curves (b) during the training process.

prior information serves as a regularizing supplement. Fig. 7(b) further depicts the training and validation losses, with stable convergence and absence of overfitting.

2) Sensitivity Analysis: To assess the contributions of three core innovations, ablation studies were conducted, focusing on: 1) dynamic trainable weighting, 2) power-curve stack, and 3) Transformer stack. MSE is employed as the loss function for both training and validation processes. PI-NBEATSx denotes a simplified network obtained by removing the dynamic-weight mechanism from the proposed method. Table III demonstrates that removing dynamic-weight mechanism increased NMSE by 39.3% (0.1487%→0.2071%) and degraded R^2 (0.9872→0.9821). This confirms the mechanisms role in adaptively balancing the contributions of multistacks.

Table V reveals the forecasting performance of multiple NBEATSx variants. Each variant uses dedicated stacks to decompose the input time series into specific components. The trend stack employs polynomial basis functions to capture long-term directional patterns. The seasonality stack leverages trigonometric (sine/cosine) basis functions to model periodic oscillations. The identity stack omits basis generation, directly using FCNN outputs $\theta_{m,n}$ [defined in (2)] as its block forecasts $\hat{y}_{m,n}$. In addition, architectures, such as WaveNet [38], TCN, and LSTM networks, can serve as alternative temporal dependence learners. They operate in a manner similar to Transformer encoders by extracting features from the input series and using the network outputs as basis vectors. When these results are considered alongside those in Table III, it becomes clear that the PIAW-NBEATSx architecture achieves a significant performance improvement over all variants. This outcome confirms the necessity of the dual-stack design, as illustrated in Fig. 1.

When replacing the Transformer stack [see Table V(a)], model performance degrades substantially. Traditional stacks (trend and seasonality), due to limitations in their predefined basis functions, lead to higher errors, with SMAPE rising to 19.98–20.03 and NMAE reaching 6.13–6.18. Although general temporal models (WaveNet/TCN) partially mitigate this issue, with SMAPE reduced to 9.40–11.54 and NMAE to 2.80–3.03, their R^2 values (0.9833–0.9853) remain lower than that of the proposed architecture (0.9872). This demonstrates the irreplaceable role of the Transformer as the core adaptive temporal modeling component.

Experiments replacing the power-curve stack [see Table V(b)] reveal its domain-specific value. Although WaveNet achieves an SMAPE of 9.93%, comparable to PIAW-NBEATSx (9.44%), its NMSE (0.1876%) remains higher than that of the original architecture (0.1487%). Traditional stacks again lead to larger errors, with SMAPE ranging from 9.76% to 10.55% and NMSE from 0.1872% to 0.2310%, reflecting mismatched physical characteristics. These results demonstrate that the power-curve stack overcomes the generalization limitations of purely data-driven models in this scenario by embedding physical constraints derived from WPCs. Its role as a carrier of domain knowledge is indispensable for handling nonlinear saturation effects.

TABLE V
PERFORMANCE EVALUATION OF NBEATSX VARIANTS

	(a) Replace only the **Transformer** Stack in PIAW-NBEATSx						(b) Replace only the **Power-Curve** Stack in PIAW-NBEATSx					
Stack type	Trend	Seasonality	Identity	WaveNet	TCN	LSTM	Trend	Seasonality	Identity	WaveNet	TCN	LSTM
SMAPE (%)	20.0324	19.9810	19.8575	9.4041	11.5425	18.9426	10.5528	9.9190	9.7641	9.9302	10.5824	10.2641
NMAE (%)	6.1817	6.1302	6.0818	2.7985	3.0265	5.9907	2.7768	2.7885	2.7267	2.8938	3.2717	2.8341
NMSE (%)	0.8657	0.8490	0.8441	0.1701	0.1934	0.8347	0.1926	0.1872	0.1876	0.1873	0.2310	0.1965
R^2	0.9253	0.9267	0.9271	0.9853	0.9833	0.9280	0.9834	0.9838	0.9838	0.9839	0.9801	0.9830

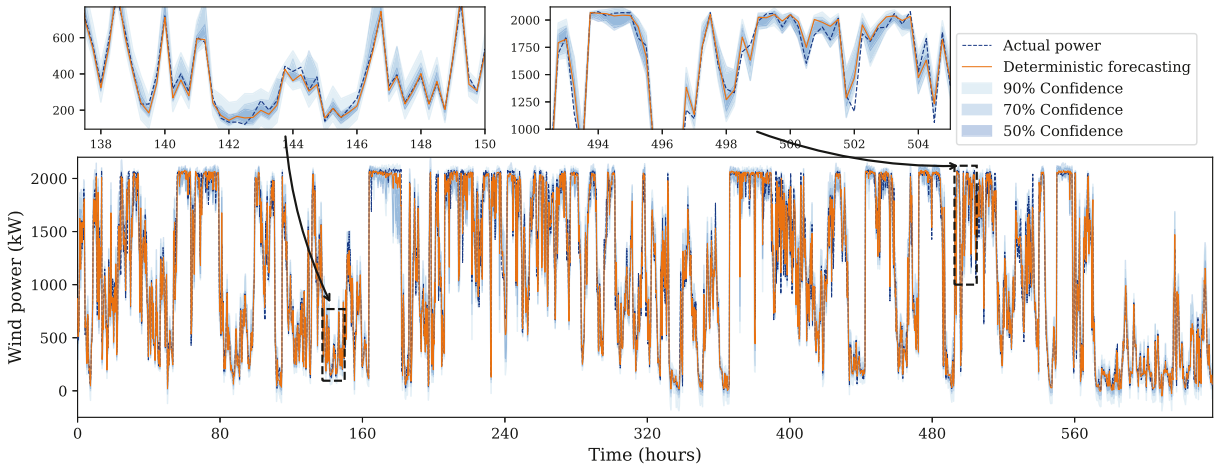

Fig. 8. Forecasting interval coverage at 50%, 70%, and 90%.

The synergistic interaction between the two stacks is the key to the observed performance gains. The Transformer stack serves as a universal engine for extracting complex temporal features, while the power-curve stack injects domain priors to ensure physical consistency. Ablation experiments show that removing either stack leads to significant performance degradation: replacing the Transformer stack causes degradation in overall performance, whereas replacing the power-curve stack results in scenario-specific collapse. By contrast, traditional NBEATSx variants rely on generic basis functions, making it difficult to represent the intricate, domain-specific structures of industrial time series. This validates that PIAW-NBEATSx overcomes these limitations, delivering more accurate, reliable, and interpretable forecasts, and thereby offering a novel approach for research and application in complex wind power time series analysis.

3) Uncertainty Analysis: We assess the PIAW-NBEATSx network's probabilistic forecasts using the Pinball loss, defined as

$$\text{Pinball} = \frac{1}{n_s}\sum_{i=1}^{n_s}\max\left(\tau(y_i-\hat{y}_i),(\tau-1)(y_i-\hat{y}_i)\right) \quad (18)$$

which quantifies the deviations between predicted quantiles and actual values at levels $\tau = \{0.05, 0.15, 0.25, 0.5, 0.75, 0.85, 0.95\}$.

As shown in Fig. 8, the model achieves empirical coverages of 50%, 70%, and 90% with mean interval widths of 82.88 kW, 171.85 kW, and 229.81 kW, respectively. The relatively narrow intervals, particularly at the 50% level, indicate precise uncertainty quantification. Notably, the top-left subplot in Fig. 8 reveals that PIAW-NBEATSx maintains high accuracy and low uncertainty in low-power regimes ($\leq$750 kW). This combination of sharpness and reliability is critical for power system operations.

IV. CONCLUSION

This article introduced PIAW-NBEATSx, a novel framework for short-term WPF that overcame the gap between physical priors and data-driven forecasting. The model embedded wind turbine operational constraints and WPC characteristics directly into its architecture through a dedicated power-curve stack, and captured complex temporal dependencies with a Transformer stack. In addition, it replaced fixed aggregation with a dynamic trainable weighting mechanism that adaptively balanced the contribution of each stack. These ensured physically consistent forecasts, and also enhanced interpretability and robustness against nonphysical outputs. Experimental results on a real dataset demonstrated that PIAW-NBEATSx consistently outperformed benchmark models and multiple NBEATSx variants across key metrics, confirming the effectiveness of its design. Future work will explore the probabilistic uncertainty quantification of WPFs to further strengthen the reliability of forecasting models. This may include integrating stochastic process modeling or Bayesian DL to produce calibrated predictive distributions, as well as exploring quantile regression and ensemble-based strategies to better capture uncertainty under different operating conditions.

REFERENCES

[1] Global Wind Energy Council (GWEC), *Global Wind Report 2024*, Brussels, Belgium: GWEC, Apr. 2024.[Online]. Available: https://www.gwec.net/reports/globalwindreport/2024

[2] L. Sheng, C. Li, M. Gao, X. Xi, and D. Zhou, "A review of SCADA-based condition monitoring for wind turbines via artificial neural networks," *Neurocomputing*, vol. 633, 2025, Art. no. 129830.

[3] S. Chen, R. Yang, M. Zhong, X. Xi, and M. E. Orchard, "FWRF: Mitigating redundant features in fault diagnosis via feature weighted random forest," *IEEE Trans. Instrum. Meas.*, vol. 74, 2025, Art. no. 3519911.

[4] F. Bilendo, N. Lu, H. Badihi, A. Meyer, Ü. Cali, and P. Cambron, "Multi-target normal behavior model based on heterogeneous stacked regressions and change-point detection for wind turbine condition monitoring," *IEEE Trans. Ind. Informat.*, vol. 20, no. 4, pp. 5171–5181, Apr. 2024.

[5] C. Pan, S. Wen, M. Zhu, H. Ye, J. Ma, and S. Jiang, "Hedge backpropagation based online LSTM architecture for ultra-short-term wind power forecasting," *IEEE Trans. Power Syst.*, vol. 39, no. 2, pp. 4179–4192, Mar. 2024.

[6] Y. Song, D. Tang, J. Yu, Z. Yu, and X. Li, "Short-term forecasting based on graph convolution networks and multiresolution convolution neural networks for wind power," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 1691–1702, Feb. 2023.

[7] J. You, R. Yang, Y. Zhan, B. Song, Y. Zhang, and Z. Wang, "BR-MTFL: A novel byzantine resilience enhanced multi-task federated learning framework for high-speed train fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 74, 2025, Art. no. 3518713.

[8] Y. Tang, Y. Zhou, P. Liu, Y. Luo, F. Lin, and F.-J. Chang, "Revolutionizing solar-hydro-wind power forecasts in regional power systems with hybrid machine learning models," *Sol. Energy*, vol. 291, 2025, Art. no. 113391.

[9] X. Xi, D. Zhou, M. Chen, and N. Balakrishnan, "Remaining useful life prediction for fractional degradation processes under varying modes," *Can. J. Chem. Eng.*, vol. 98, no. 6, pp. 1351–1364, 2020.

[10] N. Chen, H. Sun, Q. Zhang, and S. Li, "A short-term wind speed forecasting model based on EMD/CEEMD and ARIMA-SVM algorithms," *Appl. Sci.*, vol. 12, no. 12, 2022, Art. no. 6085.

[11] F. Shahid et al., "1D convolutional LSTM-based wind power prediction integrated with PkNN data imputation technique," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 35, no. 10, 2023, Art. no. 101816.

[12] Z. Zhao et al., "Hybrid VMD-CNN-GRU-based model for short-term forecasting of wind power considering spatio-temporal features," *Eng. Appl. Artif. Intell.*, vol. 121, 2023, Art. no. 105982.

[13] Y. Zhang, Y. Xin, Z. Liu, M. Chi, and G. Ma, "Health status assessment and remaining useful life prediction of aero-engine based on BiGRU and MMoE," *Rel. Eng. System Saf.*, vol. 220, 2022, Art. no. 108263.

[14] X. Zhang, L. Sheng, B. He, and Y. Lu, "A data-driven based hybrid multi-branch framework for AUV navigation," *Ocean Eng.*, vol. 324, 2025, Art. no. 120675.

[15] Y. Li et al., "A TCN-based hybrid forecasting framework for hours-ahead utility-scale PV forecasting," *IEEE Trans. Smart Grid*, vol. 14, no. 5, pp. 4073–4085, Sep. 2023.

[16] M. Gong et al., "Short-term wind power forecasting model based on temporal convolutional network and informer," *Energy*, vol. 283, 2023, Art. no. 129171.

[17] Q. Li, X. Ren, F. Zhang, L. Gao, and B. Hao, "A novel ultra-short-term wind power forecasting method based on TCN and informer models," *Comput. Elect. Eng.*, vol. 120, 2024, Art. no. 109632.

[18] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 22419–22430.

[19] G. Ban, Y. Chen, Z. Xiong, Y. Zhuo, and K. Huang, "The univariate model for long-term wind speed forecasting based on wavelet soft threshold denoising and improved autoformer," *Energy*, vol. 290, 2024, Art. no. 130225.

[20] L. O. Bentsen, N. D. Warakagoda, R. Stenbro, and P. Engelstad, "Spatio-temporal wind speed forecasting using graph networks and novel transformer architectures," *Appl. Energy*, vol. 333, 2023, Art. no. 120565.

[21] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2022, vol. 162, pp. 27268–27286.

[22] H. Zhao, P. Xu, T. Gao, J. J. Zhang, J. Xu, and D. W. Gao, "CPTCFS: CausalPatchTST incorporated causal feature selection model for short-term wind power forecasting of newly built wind farms," *Int. J. Elect. Power Energy Syst.*, vol. 160, 2024, Art. no. 110059.

[23] J. Ding, J. Wang, and J. Wang, "Enhancing significant wave height prediction based on numerical SWAN and crossformer models with adaptive decomposition," *Expert Syst. Appl.*, vol. 291, 2025, Art. no. 128523.

[24] Y. Zhan, R. Yang, Y. Zhang, and Z. Wang, "Mitigating catastrophic forgetting in cross-domain fault diagnosis: An unsupervised class incremental learning network approach," *IEEE Trans. Instrum. Meas.*, vol. 74, 2025, Art. no. 3500614.

[25] B. Huang and J. Wang, "Applications of physics-informed neural networks in power systems-a review," *IEEE Trans. Power Syst.*, vol. 38, no. 1, pp. 572–588, Jan. 2023.

[26] J. Gao, Y. Cheng, D. Zhang, and Y. Chen, "Physics-constrained wind power forecasting aligned with probability distributions for noise-resilient deep learning," *Appl. Energy*, vol. 383, 2025, Art. no. 125295.

[27] Y. Liu, J. Wang, and L. Liu, "Physics-informed reinforcement learning for probabilistic wind power forecasting under extreme events," *Appl. Energy*, vol. 376, 2024, Art. no. 124068.

[28] Y. D. Mammedov, E. U. Olugu, and G. A. Farah, "Weather forecasting based on data-driven and physics-informed reservoir computing models," *Environ. Sci. Pollut. Res.*, vol. 29, no. 16, pp. 24131–24144, 2022.

[29] X. Jia et al., "Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles," *ACM/IMS Trans. Data Sci.*, vol. 2, no. 3, pp. 1–26, 2021.

[30] F. Ye, J. Brodie, T. Miles, and A. A. Ezzat, "AIRU-WRF: A physics-guided spatio-temporal wind forecasting model and its application to the US mid atlantic offshore wind energy areas," *Renewable Energy*, vol. 223, 2024, Art. no. 119934.

[31] N. Kirchner-Bossi, G. Kathari, and F. Porté-Agel, "A hybrid physics-based and data-driven model for intra-day and day-ahead wind power forecasting considering a drastically expanded predictor search space," *Appl. Energy*, vol. 367, 2024, Art. no. 123375.

[32] Z. Xue, Y. Zhang, C. Cheng, and G. Ma, "Remaining useful life prediction of lithium-ion batteries with adaptive unscented kalman filter and optimized support vector regression," *Neurocomputing*, vol. 376, pp. 95–102, 2020.

[33] A. S. Zamzam and N. D. Sidiropoulos, "Physics-aware neural networks for distribution system state estimation," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4347–4356, Nov. 2020.

[34] X. Hu, H. Hu, S. Verma, and Z.-L. Zhang, "Physics-guided deep neural networks for power flow analysis," *IEEE Trans. Power Syst.*, vol. 36, no. 3, pp. 2082–2092, May 2021.

[35] S. Chen, B. Zhang, X. Li, Y. Ye, and K. Lin, "Facilitating interaction between partial differential equation-based dynamics and unknown dynamics for regional wind speed prediction," *Neural Netw.*, vol. 174, 2024, Art. no. 106233.

[36] K. G. Olivares, C. Challu, G. Marcjasz, R. Weron, and A. Dubrawski, "Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with NBEATSx," *Int. J. Forecasting*, vol. 39, no. 2, pp. 884–900, 2023.

[37] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Conf. Neural Inform. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 5998–6008.

[38] Y. Wang, T. Chen, S. Zhou, F. Zhang, R. Zou, and Q. Hu, "An improved wavenet network for multi-step-ahead wind energy forecasting," *Energy Convers. Manage.*, vol. 278, 2023, Art. no. 116709.

Chunyu Li received the B.Eng. and M.Eng. degrees in control science and engineering in 2019 and 2022, respectively, from the China University of Petroleum (East China), Qingdao, China, where she is currently working toward the Doctoral degree in fault diagnosis and interpretability of wind turbines with the College of Control Science and Engineering.

Her research interests include data-driven modeling, fault diagnosis, and reliability analysis.

Li Sheng (Senior Member, IEEE) received the B.Sc. degree in computer science and technology and the M.Sc. degree in computer software and theory from Shandong Normal University, Jinan, China, in 2003 and 2006, respectively, and the Ph.D. degree in control theory and control engineering from Jiangnan University, Wuxi, China, in 2010.

From 2008 to 2009, he was a visiting Ph.D. Student with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD, USA. From 2015 to 2016, he was a Visiting Scholar with the Department of Information Systems and Computing, Brunel University London, London, U.K. He is currently a Professor with the College of Control Science and Engineering, China University of Petroleum (East China), Qingdao, China. He has authored or coauthored more than 70 articles in refereed international journals and is an active reviewer for many such journals. His research interests include stochastic control and filtering, networked control systems, and fault detection and diagnosis for modern systems.

Xiaopeng Xi received the B.Eng. degree in detection guidance and control technology from the Beijing Institute of Technology, Beijing, China, in 2015, and the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2020.

He is a Postdoctoral Fellow with the Advanced Center for Electrical and Electronic Engineering (AC3E), Universidad Técnica Federico Santa María, Valparaíso, Chile. His research interests include fault diagnosis, reliability analysis, degradation modeling, and remaining useful life prediction.

Maiying Zhong received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 1999.

From 2000 to 2001, she was a Visiting Scholar with the University of Applied Sciences Lausitz, Cottbus, Germany. She was a Professor with the School of Control Science and Engineering, Shandong University, Jinan, China, from 2002 to 2008, and as a Postdoctoral Fellow with Central Queensland University, Rockhampton, QLD, Australia, from 2006 to 2007. From 2009 to 2016, she was a Professor with the School of Instrument Science and Opto-Electronics Engineering, Beihang University, Beijing, China. In March 2016, she joined the Shandong University of Science and Technology, Qingdao, China, where she is currently a Professor with the College of Electrical Engineering and Automation. Her research interests include model-based fault diagnosis, fault-tolerant systems and their applications.

Dr. Zhong is a Fellow of the Chinese Association of Automation.