Energy 335 (2025) 137753

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy

Cabin: A collaborative and adaptive framework for wind power forecasting integrating ambient variables

Senzhen Wu [a], Yu Chen [a], Xinhao He [a], Zhijin Wang [a,*], Xiufeng Liu [b,*], Yonggang Fu [a]

[a] College of Computer Engineering, Jimei University, Yinjiang Road 185, Xiamen, 361021, Fujian, China
[b] Department of Technology, Management and Economics, Technical University of Denmark, 2800, Kgs. Lyngby, Denmark

ARTICLE INFO

Dataset link: https://github.com/xiufengliu/cabin-wind-forecasting

Keywords:
Wind power forecasting
Adaptive modeling
Collaborative feature integration
Ambient variable utilization
Energy systems reliability
Kolmogorov–Arnold Networks (KAN)

ABSTRACT

Accurate wind power forecasting is paramount for maintaining grid stability and facilitating the efficient integration of renewable energy. However, the inherent variability of wind patterns and their complex dependencies on meteorological conditions pose significant forecasting challenges. This paper introduces *Cabin*, a novel framework designed for enhanced wind power prediction by effectively integrating historical wind power data with ambient variables such as temperature, wind speed, and direction. Cabin's architecture features two key modules: an Ambient Representation Module (ARM) for extracting multi-dimensional, context-aware features, and a Collaboration of Ambient Variables (CAV) module that synergistically integrates these features using temporal convolutions and Kolmogorov–Arnold Networks (KAN) for adaptive non-linear modeling. This collaborative and adaptive design allows Cabin to handle heterogeneous data fusion and accommodate varying data completeness through three distinct configurations. Comprehensive evaluations on two public benchmark wind power datasets (TWPF and GWPF) demonstrate Cabin's consistent superiority over a comprehensive suite of 34 state-of-the-art baseline models, achieving significant improvements in forecast error metrics, including reductions in Mean Squared Error (MSE) by up to 48.63% and Coefficient of Variation of Root Mean Squared Error (CV-RMSE) by up to 28.33% compared to the least performant baseline. These substantial gains are statistically validated by Diebold–Mariano tests, and new experiments further confirm Cabin's robust performance on hourly resolution data, underscoring its efficacy as a powerful tool for advancing predictive reliability in renewable energy systems.

1. Introduction

The increasing global reliance on wind energy as a cornerstone of sustainable power necessitates precise and robust forecasting tools. Accurate wind power forecasting is essential for optimizing grid integration, maintaining system stability, and minimizing energy imbalance costs [1,2]. Events such as the grid overload in Germany during the 2022 storm season, where unpredictable wind patterns exacerbated risks to power network stability, underscore this critical need. The inherent variability and unpredictability of wind power, stemming from intricate and interdependent atmospheric variables like wind speed, direction, and temperature [3], present substantial forecasting challenges. Consequently, advanced forecasting models must adeptly capture these spatiotemporal dependencies. Furthermore, real-world deployments often grapple with incomplete or inconsistent data, demanding adaptable forecasting approaches that maintain performance under such conditions. These challenges are multifaceted, encompassing not only data quality issues but also the distinct requirements of forecasting at different temporal scales, such as the daily level essential for strategic grid planning and day-ahead market operations. These complexities highlight the urgent need for innovative techniques that can intelligently leverage diverse data sources, particularly ambient (exogenous) meteorological variables, for precise and resilient wind power predictions.

Wind power generation forecasting is inherently complex due to the stochastic nature of wind and its dependence on numerous meteorological factors. Wind speed and direction exhibit non-stationary and non-Gaussian characteristics across various spatial and temporal scales, often rendering traditional modeling approaches inadequate [4,5]. While stochastic programming models have been developed to account for these uncertainties [6,7], and sophisticated machine learning approaches can handle high-dimensional datasets [8], a key challenge remains: effectively modeling the collaborative interplay between historical wind power data and multiple ambient variables. Many existing methods struggle to capture these interactions comprehensively,

* Corresponding authors.
E-mail addresses: szwbyte@gmail.com (S. Wu), yychenpro@gmail.com (Y. Chen), xhhdaxx@gmail.com (X. He), zhijin@jmu.edu.cn (Z. Wang), xiuli@dtu.dk (X. Liu), yonggangfu@jmu.edu.com (Y. Fu).

https://doi.org/10.1016/j.energy.2025.137753
Received 25 January 2025; Received in revised form 19 July 2025; Accepted 25 July 2025
Available online 22 August 2025

limiting their ability to fully exploit available information.

Traditional forecasting models, including linear regression, Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs), often face difficulties in modeling the intricate interactions between historical wind power data and ambient variables. Autoregressive models, for example, primarily rely on historical wind power data, thereby potentially neglecting crucial influences from exogenous variables like temperature and pressure [9]. CNNs, while proficient at identifying spatial patterns, may not adequately address long-range temporal dependencies [10]. Recurrent architectures like Long Short-Term Memory (LSTM) networks, designed for temporal dynamics, can encounter issues such as vanishing or exploding gradients with extended sequences [11]. Many such models tend to process wind power generation and ambient variables as separate information streams or with limited interaction, failing to leverage the potential of a unified representation that encapsulates their joint influence.

Recent advancements have underscored the necessity of integrating multiple influencing factors into cohesive frameworks. Hybrid models combining autoregressive techniques with neural networks have shown improved performance by capturing both linear and nonlinear relationships [12]. Ensemble approaches using diverse machine learning techniques have also demonstrated gains by considering a broader range of meteorological factors [13]. However, the challenge of developing models that can inherently and dynamically account for the *collaborative* nature of these multiple influencing factors — meaning how their combined, interactive effects shape future wind power — persists. This involves more than simple concatenation; it requires mechanisms to learn complex, synergistic relationships.

To address these limitations, we propose *Cabin* (a name for our model architecture), a novel framework for wind power forecasting that distinguishes itself through specialized modules designed for sophisticated feature representation and integration. The architecture centrally features an **Ambient Representation Module (ARM)**, which processes input data comprising both historical wind power and ambient variables to extract multi-dimensional features. The ARM achieves this by applying learnable transformations and dimension-specific softmax normalization — strategically across sample, temporal, and feature axes — to effectively emphasize the most salient patterns and interdependencies within the data. This unique multi-axis weighting mechanism differentiates it from standard attention layers found in Transformer-based models, which typically focus on temporal or feature dimensions in a predefined manner. Subsequently, a **Collaboration of Ambient Variables (CAV)** module takes these rich features generated by the ARM and synthesizes them. Within the CAV module, a unified representation is learned that models the intricate interactions between historical wind power and the ambient variables. This module employs temporal convolutions to capture time-dependent relationships and, critically, incorporates Kolmogorov–Arnold Networks (KAN) [14]. The use of KANs for non-linear adaptive integration, instead of traditional fixed-activation MLPs, allows Cabin to model complex non-linear dependencies with adaptive activation functions (splines), thereby enhancing both interpretability and predictive accuracy with potentially fewer parameters. This provides a genuine contribution beyond existing Transformer and graph-based models which typically rely on simpler activation functions or concatenation-based fusion. The term "collaborative" in Cabin specifically refers to this deep, learned integration and joint modeling of historical power data and ambient variables within the CAV module, a process facilitated by the expressive features generated by ARM. This methodological approach moves beyond simple input concatenation to actively model their combined influence on forecasting outcomes.

Furthermore, Cabin is designed to be "adaptive", catering to varying data availability scenarios commonly encountered in practical applications. This adaptability is realized through three distinct architectural configurations. The first, an **only-target framework**, is tailored for situations where the model utilizes exclusively historical wind power data. The second, a **data-first framework**, concatenates historical wind power and ambient variable data at the input stage, allowing them to be processed jointly through the model's backbone. The third configuration, a **learning-first framework**, processes wind power and ambient variables through separate initial representation learning pathways; the specialized features learned for each data type are then fused before the final predictions are generated. These configurations, which are detailed further in Section 3.6, ensure Cabin's versatility and enable robust performance regardless of the completeness of available ambient data. Such adaptability is crucial for real-world deployment where data streams can often be inconsistent or incomplete.

The principal contributions of this paper are threefold:

(1) **A novel collaborative feature integration mechanism** embodied in the ARM and CAV modules, that jointly models historical wind power and ambient variables. This approach, particularly with ARM's multi-axis importance discerning and the innovative use of KANs in CAV for non-linear adaptive integration, captures complex, non-linear dynamics often missed by methods treating data sources with simpler forms of integration or fixed activations, leading to enhanced prediction accuracy.
(2) **An adaptive Cabin model architecture** with three distinct configurations (data-first, only-target, and learning-first) to accommodate varying data completeness. This design ensures versatility and sustained performance for real-world applications characterized by diverse data availability.
(3) **Comprehensive evaluation of Cabin across multiple public benchmark datasets against 34 diverse state-of-the-art baseline models.** Results demonstrate significant improvements in error metrics (MAE, MSE, CV-RMSE, and R^2 score), statistically validated by Diebold–Mariano tests. New experiments further confirm Cabin's robust performance on hourly resolution data, underscoring its efficacy in improving grid stability and prediction reliability across temporal granularities.

The remainder of this paper is structured as follows: Section 2 reviews related work in wind power forecasting, including advances in modeling techniques and frameworks for adaptive and multi-source forecasting. Section 3 details the methodology of the Cabin model, including its architectural variants, the ARM, and the CAV module. Section 4 describes the experimental setup, datasets, evaluation metrics, and comparative analysis. Section 5 presents and discusses the experimental results, including performance comparisons, sensitivity analyses, and ablation studies. Lastly, Section 6 provides conclusions and outlines directions for future research. An Appendix lists abbreviations used.

2. Related work

Advances in wind power forecasting have spurred a diverse array of modeling techniques and frameworks aimed at enhancing accuracy and robustness. Traditional statistical methods have been progressively augmented and, in many cases, surpassed by machine learning and deep learning models, which are better equipped to capture the complex, non-linear dependencies inherent in wind power data. This section reviews key advancements in modeling techniques and explores innovative frameworks that incorporate adaptive and multi-source learning, setting the context for the unique contributions of the proposed Cabin model.

2.1. Advances in modeling techniques for wind power forecasting

Forecasting wind power generation is challenging due to intricate temporal and spatial dependencies in meteorological and environmental data. Traditional statistical methods like Autoregression (AR) and Vector Autoregression (VAR) have been applied for linear relationships

and short-term dependencies [15]. However, they often struggle with the high-dimensional, non-linear nature of wind data [16]. To address these limitations, machine learning and deep learning approaches such as Convolutional Neural Networks (CNNs) for spatial feature extraction and Recurrent Neural Networks (RNNs) for temporal patterns have become prevalent. Models like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) have shown superior performance in learning complex temporal patterns [17,18]. Hybrid CNN-LSTM architectures further improved accuracy by combining spatial and temporal feature learning [19]. While these methods advance univariate or simplistic multivariate forecasting, Cabin aims to improve upon the integration of multiple exogenous variables through its ARM and CAV modules, specifically designed for collaborative feature learning from heterogeneous sources.

Transformer-based architectures, with their self-attention mechanisms, have marked significant progress in time series forecasting, particularly for capturing long-range dependencies. The Informer model [20] addressed scalability for long sequences, showing potential in renewable energy. However, while effective for temporal dependencies, standard Transformers may require adaptation for optimal multivariate forecasting where inter-variable relationships are key [21]. Cabin's CAV module, particularly with KANs, offers a different approach to modeling these complex interactions, potentially offering more specialized non-linear mapping than standard feed-forward layers or fixed-activation functions in Transformers.

Graph Neural Networks (GNNs) have emerged for modeling spatial dependencies among wind turbines or farms, such as the Adaptive Multi-channel Graph Convolutional Network (AM-GCN) [22] and adaptive graph residual networks [23]. While powerful for explicit spatial structures, GNNs often rely on consistent graph topology. Cabin, while not explicitly a GNN, focuses on the "collaborative" aspect of feature integration from multiple variables for a single site or co-forecasting multiple sites as distinct features, offering robustness even with variable data completeness through its adaptive configurations.

Recent research highlights the importance of hybrid and sophisticated data integration frameworks. For instance, GMDH abductive neural networks using SCADA data [24] and multiview GRU models for ultra-short-term forecasting [25] emphasize multi-source data fusion. Cabin builds upon these ideas by proposing a structured approach (ARM and CAV) for feature extraction and collaborative integration of ambient intelligence (i.e., meteorological data) with historical power data. Many wind power forecasting studies have indeed considered multiple input variables [26,27]. Cabin's distinction lies in its specific architectural choices for this integration: the multi-axis attention-like weighting in ARM (through dimension-specific softmax) and the KAN-enhanced fusion in CAV, aiming for a more nuanced understanding of variable interdependencies than standard concatenation or generic neural network layers.

2.2. Frameworks for adaptive and multi-source forecasting

The increasing complexity and variability in wind power forecasting have driven research towards models that are not only accurate but also adaptive to changing conditions and capable of leveraging information from multiple sources or sites. While Cabin's "collaborative" nature refers to the integration of different types of variables (power and ambient) for a given forecasting task, and its "adaptive" nature refers to its flexible configurations for data availability, it is useful to review broader concepts of collaboration and adaptation in the literature to position our work.

Federated learning and transfer learning are prominent examples of collaborative frameworks in a multi-site context, enabling knowledge sharing across wind farms while addressing data privacy and heterogeneity. Tang et al. [28] developed a privacy-preserving model combining these techniques. Zhang et al. [29] introduced the Multi-Source and Temporal Attention Network (MSTAN) for probabilistic forecasting, using multi-source data and attention. These approaches focus on inter-site collaboration, which is different from Cabin's intra-site variable collaboration but shares the goal of leveraging diverse information.

Multi-source data fusion techniques are directly relevant. Jonas [30] integrated static turbine data with observational and meteorological forecasts using self-attention. Haupt et al. [31] emphasized combining AI with physics-based methods. These exemplify leveraging diverse data types, a principle central to Cabin's use of ambient variables.

Hybrid learning approaches blending various data sources and modeling techniques enhance adaptability. Jin et al. [32] used an ensemble of Gaussian processes for time-varying wind patterns. Wu and Xu [33] developed a spatial–temporal adaptive model for regional heterogeneity. Xie et al. [34] introduced a spatiotemporal GNN with a self-adaptive adjacency matrix. These highlight the trend towards dynamic and adaptive models.

Adaptive, graph-based frameworks address spatial variability. Wang et al. [35] proposed AG-MGAT, a multi-graph attention network. Li et al. [36] introduced HSTGCN for multi-modal wind and PV power prediction. While these focus on spatial relationships explicitly via graphs, Cabin's ARM module performs a form of adaptive feature weighting across samples, time, and features, which can be seen as a non-explicit way of adapting to input characteristics.

Reinforcement learning (RL) has also been applied for real-time adaptability in resource optimization related to wind power [37,38]. While Cabin is not an RL model, the drive for dynamic adaptation is a shared theme. The discussion of these broader adaptive and collaborative frameworks (federated learning, transfer learning, specific GNNs, RL) serves to illustrate the landscape of advanced forecasting. Cabin contributes to this landscape by focusing on a specific type of collaboration (joint modeling of historical power and ambient variables through ARM/CAV) and adaptation (structural variants for data completeness), rather than employing these other specific techniques directly.

3. Methodology

3.1. Cabin overview

The Cabin model, illustrated in Fig. 1, addresses the challenges of wind power forecasting by synergistically integrating historical power data and exogenous ambient variables into a unified, adaptive framework. Cabin is not an acronym but the designated name for our proposed architecture. Unlike traditional methods that may process these data types with limited interaction, Cabin employs a structured collaborative representation mechanism. This mechanism captures both short- and long-term dependencies and dynamically adapts to fluctuations in wind power data through its specialized modules. The model architecture comprises two main modules—the Ambient Representation Module (ARM) and the Collaboration of Ambient Variables (CAV)—which work sequentially to transform raw data into accurate predictions.

The ARM module performs multi-dimensional feature extraction across sample, temporal, and feature dimensions, isolating essential patterns within historical wind power data and exogenous ambient data (e.g., temperature, wind speed). This layered extraction, detailed in Section 3.4, enables Cabin to capture complex temporal dynamics and inter-variable dependencies, providing a comprehensive view of the factors influencing wind power. The CAV module then synthesizes these representations to generate the final forecast. As described in Section 3.5, it first applies a linear transformation to unify the extracted features, followed by temporal convolution to capture time-dependent relationships, and finally, incorporates Kolmogorov–Arnold Networks (KAN) [14]. The use of KANs allows Cabin to model complex non-linear dependencies with adaptive activation functions, thereby enhancing Cabin's flexibility in learning specific patterns unique to wind power

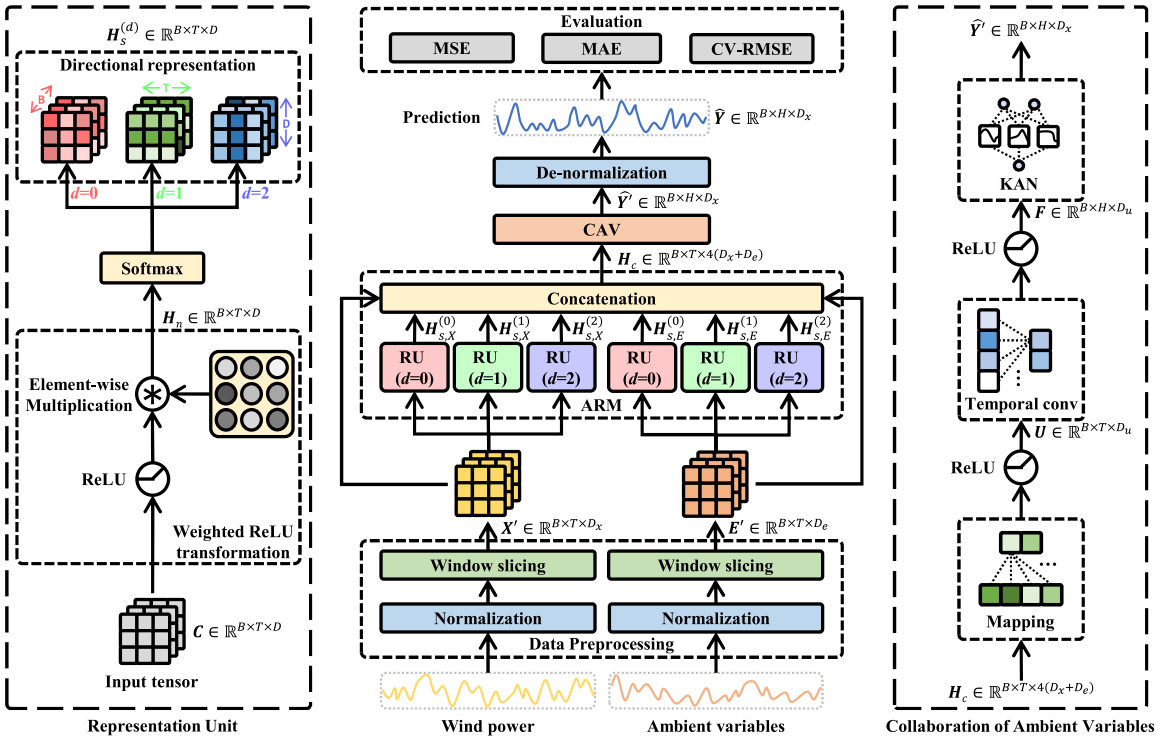


Fig. 1. Workflow of the Cabin model. It illustrates (a) data preprocessing including normalization and window slicing, (b) the Ambient Representation Module (ARM) performing multi-dimensional feature extraction via Representation Units (RUs) that apply weighted ReLU and dimension-specific softmax operations (sample, temporal, feature-wise) to both historical wind power (X') and ambient variable (E') inputs, and (c) the Collaboration of Ambient Variables (CAV) module which integrates these extracted features using a mapping unit, temporal convolutions, and Kolmogorov–Arnold Networks (KAN) to generate final wind power forecasts ($\hat{Y}$). This process captures temporal, feature-level, and non-linear dependencies.

data, potentially improving interpretability and predictive accuracy compared to models with fixed activation functions.

Together, ARM and CAV form an end-to-end pipeline. The "collaborative" aspect is realized through ARM's joint processing of historical and ambient data features and CAV's sophisticated fusion of these representations. The "adaptive" nature stems from Cabin's flexible architecture, offering three configurations (Section 3.6) to handle varying data availability, making it a robust tool for wind power forecasting.

3.2. Problem formulation

Wind turbine active power forecasting involves predicting future turbine output based on historical data and ambient (exogenous) variables. Wind speed and direction, while meteorological, are considered exogenous as their generation is external to the turbine's power output process itself, serving as inputs to it [26]. Formally, let $\boldsymbol{X}^* \in \mathbb{R}^{N\times D_x}$ and $\boldsymbol{E}^* \in \mathbb{R}^{N\times D_e}$ represent the normalized historical wind power and ambient variable time series, respectively, where N is the number of time steps, D_x is the number of wind turbines (or distinct power generation series being forecast, allowing for multi-site forecasting if $D_x > 1$), and D_e is the number of ambient variables.

To facilitate model training, we apply a window slicing function $W(\cdot)$ to $\boldsymbol{X}^*$ and $\boldsymbol{E}^*$, generating multiple windowed samples. Each input window slides one time step forward to generate the next sample, creating a sequence of training instances. For evaluation, predictions are made for corresponding future windows in the test set. If the prediction horizon $H > 1$, this means the model predicts a sequence of H future values. The evaluation metrics are then computed by comparing the predicted sequences against the actual future sequences over the entire test period. The windowed inputs are denoted by $\boldsymbol{X}^w = W^T_{1:N-T-H-1}(\boldsymbol{X}^*) \in \mathbb{R}^{N_w\times T\times D_x}$ and $\boldsymbol{E}^w = W^T_{1:N-T-H-1}(\boldsymbol{E}^*) \in \mathbb{R}^{N_w\times T\times D_e}$, where $N_w = N - T - H + 1$ is the number of window slices, and T represents the length of each historical observation window. Detailed preprocessing steps are provided in Section 3.3.

For a batch of samples, we define the batched wind power and ambient variable samples as $\boldsymbol{X}' = \{\boldsymbol{x}^w_n, \boldsymbol{x}^w_{n+1}, \ldots, \boldsymbol{x}^w_{n+B-1}\} \in \mathbb{R}^{B\times T\times D_x}$ and $\boldsymbol{E}' = \{\boldsymbol{e}^w_n, \boldsymbol{e}^w_{n+1}, \ldots, \boldsymbol{e}^w_{n+B-1}\} \in \mathbb{R}^{B\times T\times D_e}$, where $\boldsymbol{x}^w_n$ and $\boldsymbol{e}^w_n$ denote the nth window slice samples in $\boldsymbol{X}^w$ and $\boldsymbol{E}^w$, respectively, and B is the batch size.

The forecasting objective is to learn a mapping function f that predicts wind power values $\hat{\boldsymbol{Y}}$ based on the processed historical data. For wind power forecasting without exogenous variables (only-target configuration), this function can be expressed as:

$$\hat{\boldsymbol{Y}} \leftarrow f(\boldsymbol{X}'), \tag{1}$$

where $\hat{\boldsymbol{Y}} \in \mathbb{R}^{B\times H\times D_x}$ is the predicted output for the upcoming H time steps. When integrating exogenous variables (data-first or learning-first configurations), the forecasting process includes both wind power and ambient variable inputs, described as:

$$\hat{\boldsymbol{Y}} \leftarrow f(\boldsymbol{X}', \boldsymbol{E}'), \tag{2}$$

The loss function, defined as the mean squared error (MSE) $\mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}})$, measures the discrepancy between the true wind power values $\boldsymbol{Y} \in \mathbb{R}^{B\times H\times D_x}$ and the predicted values $\hat{\boldsymbol{Y}}$. Minimizing this loss involves optimizing the learnable parameters θ of the mapping function f. Formally, this optimization problem is given by:

$$\min_{\theta} \mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) \tag{3}$$

To efficiently address this optimization, we employ the Adam optimizer, a variant of stochastic gradient descent (SGD) that adapts the learning rate for each parameter dynamically, incorporating momentum and weight decay mechanisms to enhance convergence. For clarity, the mathematical symbols used throughout are summarized in Table 1.

Table 1
Mathematical notation.

Symbol	Definition
$\boldsymbol{X}, \boldsymbol{E}$	Original wind power and ambient variable time series
$\boldsymbol{X}^*, \boldsymbol{E}^*$	Normalized wind power and ambient variable time series
$\boldsymbol{X}^w, \boldsymbol{E}^w$	Windowed slices of $\boldsymbol{X}^*$ and $\boldsymbol{E}^*$ for input features
$\boldsymbol{X}', \boldsymbol{E}'$	Batched input samples from $\boldsymbol{X}^w$ and $\boldsymbol{E}^w$
$\boldsymbol{Y}^w$	Windowed slices of $\boldsymbol{X}^*$ for target future values
$\boldsymbol{Y}$	Batched target samples from $\boldsymbol{Y}^w$
$\hat{\boldsymbol{Y}}$	Predicted wind power values for the horizon H
N	Total number of time steps in the original series
N_w	Number of windowed samples generated
T	Length of the historical observation window (input sequence length)
H	Prediction horizon (output sequence length)
D_x, D_e	Number of features for wind power and ambient variables, respectively
B	Batch size
$W(\cdot)$	Window slicing function
$f(\cdot)$	Forecasting mapping function
$\mathcal{L}(\cdot)$	Loss function
θ	Learnable parameters of the model
$[;]$	Concatenation operation

3.3. Data preprocessing

To standardize the data and improve training efficiency, Min-Max normalization is applied, scaling data values within the range $[0, 1]$. The normalization formula is given by:

$$\boldsymbol{d}^* = \frac{\boldsymbol{d} - \min(\boldsymbol{d})}{\max(\boldsymbol{d}) - \min(\boldsymbol{d})}, \tag{4}$$

where $\boldsymbol{d}$ represents the original data values (a specific feature column), and $\boldsymbol{d}^*$ is the normalized result. De-normalization is achieved through:

$$\boldsymbol{d} = \boldsymbol{d}^* \cdot (\max(\boldsymbol{d}) - \min(\boldsymbol{d})) + \min(\boldsymbol{d}), \tag{5}$$

which restores data to its original scale. Here, $\min(\boldsymbol{d})$ and $\max(\boldsymbol{d})$ denote the minimum and maximum values in the training portion of $\boldsymbol{d}$, respectively, and are stored to apply to validation/test data and for de-normalization.

The initial time series data for wind power and ambient variables, denoted by $\boldsymbol{X}$ and $\boldsymbol{E}$, are normalized to $\boldsymbol{X}^*$ and $\boldsymbol{E}^*$. Next, window slicing is applied to segment these normalized time series into fixed-length, sequential samples. For the daily forecasting task central to this study, the historical observation window length T was set to 10 days. This value was determined through preliminary experiments to strike an optimal balance: it is long enough to capture potential weekly cyclical patterns and short-term trends, yet short enough to remain highly responsive to the most recent weather dynamics, thus preventing the dilution of predictive signals with outdated information.

These samples serve as structured input data for the model to capture temporal relationships effectively. The slicing is performed using a window function $W(\cdot)$, resulting in the following segmented representations for input features and corresponding target values:

$$\boldsymbol{X}^w = W^T_{1:N-T-H+1}(\boldsymbol{X}^*) = \{\boldsymbol{x}^*_{t:t+T-1} \mid t = 1, \ldots, N-T-H+1\} \tag{6}$$

$$\boldsymbol{E}^w = W^T_{1:N-T-H+1}(\boldsymbol{E}^*) = \{\boldsymbol{e}^*_{t:t+T-1} \mid t = 1, \ldots, N-T-H+1\} \tag{7}$$

$$\boldsymbol{Y}^w = W^H_{T+1:N-H+1}(\boldsymbol{X}^*) = \{\boldsymbol{x}^*_{t+T:t+T+H-1} \mid t = 1, \ldots, N-T-H+1\} \tag{8}$$

Here, $\boldsymbol{X}^w \in \mathbb{R}^{N_w \times T \times D_x}$ and $\boldsymbol{E}^w \in \mathbb{R}^{N_w \times T \times D_e}$ represent the collections of sequential input slices for wind power and ambient variables, respectively, with $N_w = N - T - H + 1$ denoting the number of generated windows. Correspondingly, $\boldsymbol{Y}^w \in \mathbb{R}^{N_w \times H \times D_x}$ represents the future wind power values associated with each input window slice.

By structuring data in this manner, the model can more effectively leverage historical and ambient information for accurate forecasting.

3.4. Ambient Representation Module (ARM)

The Ambient Representation Module (ARM) is crafted to capture intricate dependencies within the input data, focusing on sample importance, temporal dynamics, and feature interrelationships to strengthen the model's predictive accuracy for wind power. The ARM processes both wind power data, $\boldsymbol{X}' \in \mathbb{R}^{B \times T \times D_x}$, and ambient variable data, $\boldsymbol{E}' \in \mathbb{R}^{B \times T \times D_e}$, where B is batch size, T is sequence length, and D_x / D_e are feature dimensions. This multi-faceted view is crucial for representing the complex factors influencing wind power.

To implement this, ARM applies three distinct Representation Units (RUs), each targeting a specific data dimension: sample, temporal, and feature. The term "axes" refers to these dimensions of the input tensor:

- **Sample axis (dimension B):** Operations along this axis consider relationships or relative importance across different samples within a batch for a specific time step and feature.
- **Temporal axis (dimension T):** Operations along this axis focus on dependencies or salient points across the time steps within each sample and for each feature.
- **Feature axis (dimension D_x or D_e):** Operations along this axis capture interactions or relative importance among different features (e.g., different turbines' power, or power vs. temperature) at a specific time step for a given sample.

Each RU incorporates a weighted ReLU transformation followed by softmax normalization along its designated axis, allowing the module to emphasize significant patterns.

Within each RU, for an input tensor $\boldsymbol{C} \in \mathbb{R}^{B \times T \times D}$ (where $\boldsymbol{C}$ could be $\boldsymbol{X}'$ or $\boldsymbol{E}'$, and D is D_x or D_e), a weighted transformation is applied:

$$\boldsymbol{H}_n = \mathrm{ReLU}(\boldsymbol{C}) \odot \boldsymbol{W}_c, \tag{9}$$

where $\boldsymbol{W}_c \in \mathbb{R}^{T \times D}$ is a learnable weight matrix, initialized using Glorot uniform initialization [39], and broadcasted across the batch dimension B. Here, $\boldsymbol{C}$ is the normalized input data, predominantly in $[0, 1]$. Applying ReLU first ($\mathrm{ReLU}(\boldsymbol{C})$) ensures that any minor negative values due to numerical precision are zeroed out; for positive inputs, $\mathrm{ReLU}(\boldsymbol{C}) = \boldsymbol{C}$. The subsequent element-wise multiplication $\odot \boldsymbol{W}_c$ then allows $\boldsymbol{W}_c$ to act as learnable scaling factors or gates for each (time, feature) pair across all samples in the batch. This selectively amplifies or dampens input signals based on learned importance.

Following this transformation, dimension-specific softmax normalization is applied:

$$\boldsymbol{H}^{(0)}_s[b,t,k] = \frac{\exp(\boldsymbol{H}_n[b,t,k])}{\sum_{b'=1}^{B} \exp(\boldsymbol{H}_n[b',t,k])} \quad \text{(Sample Softmax)}, \tag{10}$$

The Sample Softmax (Eq. (10)) normalizes activations for each specific time step t and feature k across all B samples in the batch. This highlights which samples in the current batch exhibit stronger (or more relevant) activations for that particular (t, k) point. While operating across samples within a batch, this is primarily for feature re-scaling based on batch statistics and does not fundamentally alter the i.i.d. assumption for SGD if gradients are computed per sample before averaging, as is standard. It can aid in stabilizing learning by emphasizing relative signal strengths within the mini-batch context.

$$\boldsymbol{H}^{(1)}_s[b,t,k] = \frac{\exp(\boldsymbol{H}_n[b,t,k])}{\sum_{t'=1}^{T} \exp(\boldsymbol{H}_n[b,t',k])} \quad \text{(Temporal Softmax)}, \tag{11}$$

The Temporal Softmax (Eq. (11)) normalizes activations across all T time steps for each sample b and feature k. This identifies the relative importance of different time points within the input sequence for that specific feature and sample.

$$\boldsymbol{H}^{(2)}_s[b,t,k] = \frac{\exp(\boldsymbol{H}_n[b,t,k])}{\sum_{k'=1}^{D} \exp(\boldsymbol{H}_n[b,t,k'])} \quad \text{(Feature-wise Softmax)}, \tag{12}$$

The Feature-wise Softmax (Eq. (12)) normalizes activations across all D features for each sample b and time step t. This highlights the relative importance of different input features at that specific time point for that sample.

Here, $\boldsymbol{H}_s^{(0)}, \boldsymbol{H}_s^{(1)}, \boldsymbol{H}_s^{(2)} \in \mathbb{R}^{B\times T\times D}$ are the dimension-specific normalized representations. Each softmax operation enables ARM to capture dominant data patterns along different axes, refining feature extraction.

Subsequently, ARM concatenates these dimension-normalized representations along with the original input data ($\boldsymbol{X}'$ and $\boldsymbol{E}'$), forming a comprehensive feature set for the CAV module:

$$\boldsymbol{H}_c = [\boldsymbol{H}_{s,X}^{(0)}; \boldsymbol{H}_{s,X}^{(1)}; \boldsymbol{H}_{s,X}^{(2)}; \boldsymbol{X}'; \boldsymbol{H}_{s,E}^{(0)}; \boldsymbol{H}_{s,E}^{(1)}; \boldsymbol{H}_{s,E}^{(2)}; \boldsymbol{E}'], \tag{13}$$

where $\boldsymbol{H}_c \in \mathbb{R}^{B\times T\times 4(D_x+D_e)}$ denotes the combined data representation. Specifically, $\boldsymbol{H}_{s,X}^{(0)}, \boldsymbol{H}_{s,X}^{(1)}, \boldsymbol{H}_{s,X}^{(2)} \in \mathbb{R}^{B\times T\times D_x}$ are representations from $\boldsymbol{X}'$, while $\boldsymbol{H}_{s,E}^{(0)}, \boldsymbol{H}_{s,E}^{(1)}, \boldsymbol{H}_{s,E}^{(2)} \in \mathbb{R}^{B\times T\times D_e}$ are from $\boldsymbol{E}'$. This enriched feature set enhances the model's capacity for robust forecasting.

3.5. Collaboration of Ambient Variables (CAV) module

The Collaboration of Ambient Variables (CAV) module processes the integrated features $\boldsymbol{H}_c$ from ARM to predict future wind power $\hat{\boldsymbol{Y}} \in \mathbb{R}^{B\times H\times D_x}$. It employs a layered approach: a mapping unit, a temporal convolutional layer, and a Kolmogorov–Arnold Network (KAN) module.

3.5.1. Mapping unit

The mapping unit transforms $\boldsymbol{H}_c$ into a unified representation $\boldsymbol{U} \in \mathbb{R}^{B\times T\times D_u}$ using a linear transformation followed by ReLU activation:

$$\boldsymbol{U}_{b,t,:} = \sigma(\boldsymbol{H}_c[b,t,:]\boldsymbol{W}_u + \boldsymbol{b}_u), \quad \forall b \in [1,B], t \in [1,T] \tag{14}$$

where $\boldsymbol{W}_u \in \mathbb{R}^{4(D_x+D_e)\times D_u}$ and $\boldsymbol{b}_u \in \mathbb{R}^{D_u}$ are the learnable weight matrix and bias vector. This operation is applied independently to the feature vector at each time step t for each sample b in the batch. σ is ReLU. D_u is a hyperparameter.

3.5.2. Temporal convolutional layer

To capture local temporal patterns from the unified representation $\boldsymbol{U}$, a temporal convolutional layer is applied. This layer uses 1D convolutions along the time axis (T). While 2D convolutions (Conv2D) could be used if spatial relationships between turbines (features D_x) were explicitly structured as a 2D grid, 1D convolutions are more standard for multivariate time series where each feature channel is treated as a separate sequence or where inter-feature relationships are learned by subsequent layers (like KAN here) rather than by convolutional kernels across features [40]. Our approach focuses on temporal patterns per unified feature D_u first. The layer processes $\boldsymbol{U} \in \mathbb{R}^{B\times T\times D_u}$. The layer applies H filters of size T, each with "same" padding and ReLU activation, yielding enhanced temporal features:

$$\boldsymbol{F}^{(h)}[b,:,:] = \mathrm{ReLU}\left(\sum_{t'=1}^{T} \boldsymbol{W}_f^{(h)}[t',:]\cdot \boldsymbol{U}[b,t',:] + \boldsymbol{b}_f^{(h)}\right), \quad \text{for } h = 1,\dots,H, \tag{15}$$

where $\boldsymbol{W}_f^{(h)} \in \mathbb{R}^{T\times D_u}$ and $\boldsymbol{b}_f^{(h)} \in \mathbb{R}^{D_u}$ denote the weights and biases of the hth temporal filter. Each filter processes the full $T \times D_u$ slice of the input tensor $\boldsymbol{U} \in \mathbb{R}^{B\times T\times D_u}$ to produce a D_u-dimensional feature vector for each batch element.

The resulting output $\boldsymbol{F}^{(h)} \in \mathbb{R}^{B\times 1\times D_u}$ (reshaped from $\mathbb{R}^{B\times D_u}$ for clarity) is then concatenated along the filter dimension:

$$\boldsymbol{F} = [\boldsymbol{F}^{(1)}; \boldsymbol{F}^{(2)}; \dots; \boldsymbol{F}^{(H)}] \in \mathbb{R}^{B\times H\times D_u}. \tag{16}$$

This operation yields H distinct feature vectors of dimension D_u per sample, which are subsequently mapped to the desired output dimension D_x via a learnable transformation.

3.5.3. Kolmogorov–Arnold Network (KAN) module

To handle complex non-linear dependencies, KAN replaces traditional MLP layers. KANs approximate multivariate functions using compositions of learnable univariate functions (splines) on the edges of a neural network-like structure, rather than fixed activation functions on nodes summed with weighted inputs [14]. This offers greater flexibility in learning arbitrary function shapes, potentially improving accuracy and interpretability with fewer parameters than equivalently expressive MLPs for certain problems.

Given $\boldsymbol{F} \in \mathbb{R}^{B\times H\times D_u}$ from the temporal layer, KAN is applied to transform the feature dimension D_u to D_x for each of the $B \times H$ instances. Effectively, KAN acts as a powerful non-linear mapping $g : \mathbb{R}^{D_u} \to \mathbb{R}^{D_x}$. For each element (b,h) in $B\times H$, let $\boldsymbol{f}_{b,h} \in \mathbb{R}^{D_u}$ be the input to KAN. A KAN layer l with input dimension m and output dimension n is defined as:

$$\mathrm{KAN}(\boldsymbol{z})_j = \sum_{i=1}^{m} \phi_{j,i}(\boldsymbol{z}_i), \quad j = 1,\dots,n \tag{17}$$

where $\phi_{j,i}$ are learnable univariate spline functions. Multiple KAN layers can be stacked:

$$\boldsymbol{Z}^{(l)} = \mathrm{KAN}_l(\boldsymbol{Z}^{(l-1)}), \tag{18}$$

where $\boldsymbol{Z}^{(0)} = \boldsymbol{f}_{b,h}$. The final KAN layer KAN_L outputs $\hat{\boldsymbol{y}}_{b,h} \in \mathbb{R}^{D_x}$:

$$\hat{\boldsymbol{y}}_{b,h} = \mathrm{KAN}_L(\boldsymbol{Z}^{(L-1)}). \tag{19}$$

Collecting these outputs for all b,h gives $\hat{\boldsymbol{Y}} \in \mathbb{R}^{B\times H\times D_x}$. KAN's adaptability in function approximation is key for capturing the non-linear relationships between the temporal features and the final wind power output, potentially offering better generalization than fixed-activation MLPs [41,42].

3.6. Cabin architectural variants for adaptability

To accommodate different levels of data availability and to explore various strategies for integrating exogenous variables, Cabin is proposed with three architectural variants. These variants primarily differ in how historical wind power data ($\boldsymbol{X}'$) and ambient variable data ($\boldsymbol{E}'$) are introduced and processed before or within the main ARM and CAV modules. This provides adaptability to real-world scenarios where ambient data might be complete, partially available, or entirely missing.

The rationale for these configurations is to provide flexibility:

- **Only-target:** Serves as a baseline or for situations where reliable ambient data is unavailable.
- **Data-first:** Assumes ambient data is available and reliable, allowing early fusion to potentially capture low-level interactions. This is suitable when variables are expected to have direct, intertwined effects from the start.
- **Learning-first:** Allows specialized feature extraction for historical power and ambient variables separately before fusion. This can be beneficial if the two data types have distinct characteristics that are better learned independently first, or if one data source is noisier than the other.

The choice of architecture can be guided by data availability and preliminary experiments on a validation set to determine which fusion strategy yields better performance for a specific dataset.

3.6.1. Only-target framework

In this configuration, Cabin relies exclusively on historical wind power data. The input to the ARM module is solely $\boldsymbol{X}'$. Eq. (13) would simplify as terms related to $\boldsymbol{E}'$ and $\boldsymbol{H}_{s,E}$ would be absent. The CAV module then processes these ARM-derived features from $\boldsymbol{X}'$ to generate predictions $\hat{\boldsymbol{Y}}$. This setup is essential when ambient data is missing, unreliable, or not deemed beneficial.

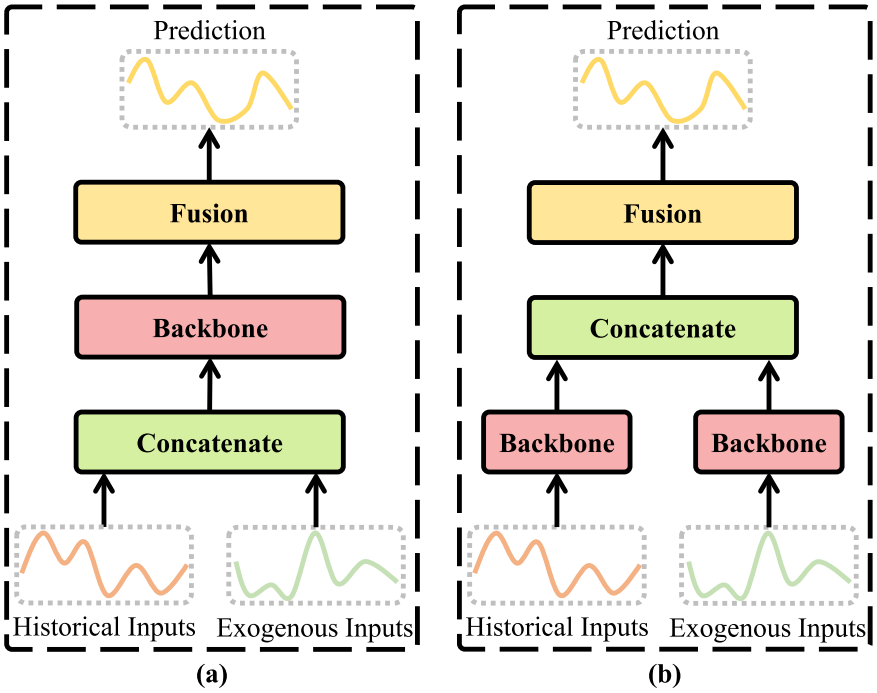


Fig. 2. Conceptual illustration of fusion strategies guiding Cabin's variants: (a) Data-first framework: Historical power data ($\boldsymbol{X}'$) and ambient variable data ($\boldsymbol{E}'$) are concatenated early and processed jointly by a shared backbone (representing ARM and part of CAV). (b) Learning-first framework: $\boldsymbol{X}'$ and $\boldsymbol{E}'$ are processed by separate backbone pathways (e.g., distinct ARMs or parts of ARM) to learn specialized representations before their outputs are fused (e.g., at the CAV stage). The "Only-target" framework would conceptually only have the $\boldsymbol{X}'$ pathway from (b) leading to prediction without any $\boldsymbol{E}'$ input or fusion.

3.6.2. Data-first framework

Illustrated conceptually in Fig. 2(a), the data-first framework concatenates historical wind power data $\boldsymbol{X}'$ and ambient variable data $\boldsymbol{E}'$ along their feature dimensions at the very beginning: $\boldsymbol{C}_{\text{input}} = [\boldsymbol{X}'; \boldsymbol{E}'] \in \mathbb{R}^{B\times T\times(D_x+D_e)}$. This combined tensor is then fed as a single input into the ARM module. The ARM extracts multi-dimensional features from this concatenated input, and the CAV module subsequently generates predictions. This approach allows the model to learn interactions between historical power and ambient variables from the earliest processing stages.

3.6.3. Learning-first framework

Conceptually shown in Fig. 2(b), the learning-first framework processes historical wind power data $\boldsymbol{X}'$ and ambient variable data $\boldsymbol{E}'$ through separate ARM pathways initially (or distinct parts of a larger ARM). This means $\boldsymbol{X}'$ goes through its own set of RUs to produce $\boldsymbol{H}_{c,X} = [\boldsymbol{H}^{(0)}_{s,X}; \boldsymbol{H}^{(1)}_{s,X}; \boldsymbol{H}^{(2)}_{s,X}; \boldsymbol{X}']$, and $\boldsymbol{E}'$ goes through its own set to produce $\boldsymbol{H}_{c,E} = [\boldsymbol{H}^{(0)}_{s,E}; \boldsymbol{H}^{(1)}_{s,E}; \boldsymbol{H}^{(2)}_{s,E}; \boldsymbol{E}']$. These specialized, separately learned representations $\boldsymbol{H}_{c,X}$ and $\boldsymbol{H}_{c,E}$ are then concatenated before being fed into the CAV module (or a part of CAV designed for fusion). This allows for tailored feature extraction for each data type before their interactions are modeled.

For baseline models in our experiments, when applying these fusion frameworks, the "backbone" in Fig. 2 refers to the main architecture of the baseline model itself. For "data-first", baselines received concatenated input. For "learning-first", baselines processed power and ambient data separately up to a point, and their intermediate representations were then fused (e.g., by concatenation followed by a linear layer) before final prediction, if the baseline architecture allowed such modification. Cabin inherently supports these via its modular ARM/CAV design.

4. Experimental setup

4.1. Datasets and experimental design considerations

We utilized two public benchmark wind turbine datasets:

(1) **Turkey Wind Power Forecasting (TWPF) Dataset** [43]: Data from a single Goldwind GW87/1500 turbine (1500 kW capacity) in Turkey, from Jan 1 to Dec 31, 2018, at 10 min intervals. Includes active power and ambient variables like wind speed, direction, temperature.
(2) **Greece Wind Power Forecasting (GWPF) Dataset** [44]: Hourly data from 18 geographically dispersed locations in Greece, from Jan 1, 2017, to Dec 31, 2020. Aggregate installed capacity of 6792.7 MW. Includes active power and relevant meteorological data for each location.

Both datasets were resampled to a daily frequency. For TWPF, 10 min data was aggregated (e.g., averaged for ambient variables, summed or averaged for power depending on the target definition) to daily. GWPF hourly data was similarly aggregated. The datasets were split chronologically: the first 70% for training, the next 10% for validation (hyperparameter tuning), and the final 20% for testing. This ensures that the model is validated and tested on data chronologically after the training data, mimicking a real-world forecasting scenario.

The prediction horizon H was set to one day. The historical observation window T was 10 days. For daily data, this means $T = 10$ (input time steps) and $H = 1$ (output time step). Batch size B was 8, which means 8 instances (each instance being 10 past days) form a batch.

4.1.1. Justification for daily aggregation and overfitting mitigation

A critical aspect of our experimental design is the decision to aggregate the high-resolution source data to a daily temporal scale. This choice warrants careful justification, particularly concerning the suitability for deep learning models and the risk of overfitting.

The aggregation to a daily scale was a deliberate choice aligned with specific, high-value forecasting tasks in energy systems management. While high-frequency forecasting is vital for real-time operations, daily forecasting addresses distinct challenges such as day-ahead energy market bidding, unit commitment, and grid load balancing over a 24-h cycle. This lower-frequency task requires models to identify longer-term dependencies and filter the high-frequency stochastic noise inherent in wind generation, presenting a unique scientific challenge.

We mitigated the risk of overfitting through a multi-pronged strategy:

(1) **Training Sample Expansion:** The limited number of raw daily data points was expanded using a sliding window approach ($T = 10, H = 1$). For the TWPF dataset, the 255-day training split yields 245 overlapping training samples. Similarly, for the GWPF dataset, the 1022-day training split generates 1012 samples. This standard technique provides a richer set of training instances for the model to learn from.
(2) **Strict Validation Protocol:** To ensure an unbiased estimate of generalization performance, we adhered to the 70%/10%/20% chronological split. The validation set was used exclusively for hyperparameter tuning and implementing an early stopping criterion. The test set was completely held-out and used only for a single, final evaluation of the fully trained models. This protocol prevents any information leakage from the test set into the model development process.
(3) **Regularization and Comparative Evidence:** All models were trained with L2 regularization to discourage overly complex solutions. The consistent outperformance of Cabin against 34 diverse baselines, all subjected to the same rigorous protocol, provides the most compelling evidence of its superior architectural design. This suggests that Cabin is genuinely more effective at extracting generalizable patterns from this challenging data, rather than simply being better at overfitting.

Furthermore, to demonstrate Cabin's robustness across different temporal scales, we have conducted experiments on hourly resolution data, presented in Appendix B.

4.2. Performance criterion

To evaluate performance, we used Mean Squared Error (MSE), Mean Absolute Error (MAE), Coefficient of Variation of Root Mean Squared Error (CV-RMSE), and the R-squared (R^2) score.

(1) Mean Square Error (MSE): $\frac{1}{N_s H D_x} \sum_{i=1}^{N_s} \sum_{h}^{H} \sum_{d}^{D_x} (Y_{i,h,d} - \hat{Y}_{i,h,d})^2$. Penalizes large errors.
(2) Mean Absolute Error (MAE): $\frac{1}{N_s H D_x} \sum_{i=1}^{N_s} \sum_{h}^{H} \sum_{d}^{D_x} |Y_{i,h,d} - \hat{Y}_{i,h,d}|$. Average magnitude of errors.
(3) Coefficient of Variation of Root Mean Square Error (CV-RMSE): $\frac{\sqrt{\text{MSE}}}{\text{mean}(Y)}$. Normalized RMSE.
(4) R-squared (R^2) Score: $1 - \frac{\sum (Y_{i,h,d} - \hat{Y}_{i,h,d})^2}{\sum (Y_{i,h,d} - \text{mean}(Y))^2}$. Proportion of variance explained.

Here N_s is the number of samples in the test set. For multi-turbine datasets like GWPF (where $D_x > 1$), these metrics are computed by averaging the errors/squared errors over all turbines D_x, all prediction steps H (here $H = 1$), and all samples N_s, providing a single performance score for the entire dataset.

4.3. Baseline models for comparison

We compared Cabin against 34 diverse time series forecasting models, detailed in Table 2. This comprehensive set includes traditional statistical models, various neural network architectures (RNNs, CNNs, Transformers), graph-based models, and other recent advanced time series forecasting methods. While many are general-purpose time series models, their state-of-the-art performance makes them strong benchmarks for evaluating Cabin's novel architectural contributions to multivariate forecasting, demonstrated here on wind power data. Several baselines also have specific mechanisms for handling multivariate inputs.

4.4. Model configuration and training settings

All models were trained by minimizing MSE loss using the Adam optimizer [67] with L2 regularization. Hyperparameters for all models (Cabin and baselines) were tuned using a grid search strategy on the validation set (10% of data, as described in Section 4.1). Each model configuration was trained until validation MSE showed no improvement for 10 consecutive epochs (early stopping). The hyperparameter configuration yielding the lowest validation MSE was selected. The specific hyperparameter ranges for baseline models are detailed in Table 3. For Cabin, key hyperparameters tuned included the unified representation dimension D_u (range $\{16, 32, 64, 128, 256, 512\}$), learning rate (range $\{0.0001, 0.001, 0.01\}$), and KAN specific parameters like spline order and grid size based on KAN library defaults and common ranges. Due to the extensive number of models and hyperparameter combinations, results reported are based on a single run with fixed random seeds for initialization to ensure reproducibility. While averaging over multiple runs would provide variance estimates, the consistency of Cabin's superior performance across datasets and metrics provides confidence in its robustness.

We implemented all models using PyTorch v2.0.1. Experiments were run on a server with an Intel Xeon Gold 5218R CPU, 256 GB RAM, and four Tesla V100-PCIE-16 GB GPUs.

4.5. Implementation of Cabin variants

The three architectural variants of Cabin—only-target, data-first, and learning-first—were implemented as described in Section 3.6. For baseline models, similar principles were applied to test their performance with and without exogenous data, and with early vs. later fusion if their architecture permitted straightforward adaptation:

- **Only target baseline:** Model trained using only historical wind power data.
- **Data-first baseline:** Historical power and ambient data were concatenated along the feature dimension and fed as input to the baseline model. A final linear layer was used to ensure the output dimension matched D_x if the baseline's architecture did not inherently handle the expanded input to produce the correct output shape. For baselines that could adjust output dimensionality internally based on input (e.g. many Transformers), this linear layer was not needed.
- **Learning-first baseline:** For baselines amenable to this, two separate instances of the model (or its initial layers) processed historical power and ambient data independently. Their output representations were then concatenated and passed through a simple fusion layer (e.g., a linear layer) to produce the final prediction. This was feasible for models with clear feature extraction stages.

This experimental setup allows for a fair comparison of Cabin's specialized fusion mechanisms against more generic applications of fusion to standard baseline architectures.

Table 2
List of baseline models for comparison.

Category	Model name	Description
Traditional models	GAR [45], AR [45], VAR [45]	GAR uses a unified weight structure; AR applies variable-specific weights for distinct dynamics; VAR models multivariate dependencies between input and output variables.
RNNs	LSTM [46], GRU [47], ED [48]	LSTM captures long-term dependencies using memory cells; GRU reduces parameters by combining gates; ED processes input sequences with an encoder–decoder structure.
CNN-based models	CNN1D [49], CNN-RNN [49], LSTNet [49]	CNN1D models sequences with convolution; CNN-RNN combines CNN with RNN for enhanced temporal feature extraction; LSTNet integrates CNN and RNN to capture both long- and short-term dependencies.
	TCN [40]	TCN uses causal convolutional layers to model sequential data, serving as an alternative to recurrent networks.
Enhanced linear models	DLinear [21], NLinear [21]	DLinear decomposes data into trend, seasonal, and residual components; NLinear applies non-linear transformations for capturing complex patterns.
Attention-based models	TPA [50]	Temporal Pattern Attention (TPA) focuses on significant historical time steps through attention mechanisms to enhance forecasting accuracy.
	DSANet [51]	Dual Self-Attention Network (DSANet) leverages self-attention to capture both short- and long-term dependencies in time series.
Transformer-based models	Transformer [52], Informer [20]	Transformer relies on attention for sequence modeling; Informer introduces ProbSparse self-attention for efficient long-sequence forecasting.
	Autoformer [53], FEDformer [54]	Autoformer uses decomposition to capture trend and seasonal features; FEDformer combines seasonal-trend decomposition with transformer architecture.
	STAEformer [55], Crossformer [56], Triformer [57]	STAEformer applies spatial–temporal attention for multivariate forecasting; Crossformer captures cross-dimension dependencies; Triformer integrates temporal, feature, and instance attention mechanisms for enhanced forecasting.
	FiLM [58]	Frequency improved Legendre Memory (FiLM) enhances long-term forecasting by incorporating frequency domain information.
	NHiTS [59]	Neural Hierarchical Interpolation for Time Series (NHiTS) improves multi-horizon forecasting using hierarchical interpolation.
	PatchTST [60]	Patch-based Time Series Transformer (PatchTST) applies patching techniques to effectively capture local dependencies in time series data.
Graph-based models	StemGNN [61]	Spectral Temporal Graph Neural Network (StemGNN) combines graph neural networks with temporal convolution for spatiotemporal modeling.
	AGCRN [62]	Adaptive Graph Convolutional Recurrent Network (AGCRN) combines adaptive graph convolution with recurrent networks for spatiotemporal forecasting.
	GAIN [45]	Graph Ambient Intelligent Network (GAIN) integrates graph attention with meteorological data to enhance multivariate time series forecasting.
Collaborative models	MSL [63]	Multivariate Shapelet Learning (MSL) extracts meaningful patterns from historical data to enhance forecasting.
	TCOAT [64]	Temporal Collaborative Attention (TCOAT) captures dependencies by focusing on global time steps and applying multi-directional attention for spatiotemporal modeling.
	CoDR [65]	Collaborative Directional Representation (CoDR) uses fluctuation extraction to model directional dependencies.
	CTRL [66]	Collaborative Temporal Representation Learning (CTRL) employs collaborative representation learning to improve forecasting accuracy and robustness.

5. Result and analyses

5.1. Performance comparison with baseline models

This section analyzes Cabin's performance against baselines using the only-target framework results (Tables 4 and 5).

Overall Performance of Cabin (Only-target): In the only-target scenario (using only historical wind power), Cabin consistently outperforms all 34 baseline models across both datasets and all metrics (MSE, MAE, CV-RMSE, R^2).

- On TWPF (Table 4): Cabin achieves MSE of 128.741, MAE of 9.150, CV-RMSE of 0.549, and R^2 of 0.669. Compared to the least performant FiLM (MSE 250.633), this is a 48.63% MSE reduction. Against the strongest baseline CTRL (MSE 139.854), Cabin shows a 7.95% MSE improvement.
- On GWPF (Table 5): Cabin achieves MSE of 83.141, MAE of 7.017, CV-RMSE of 0.450, and R^2 of 0.619. Compared to the least performant StemGNN (MSE 221.126), this is a 62.40% MSE reduction. Against the strongest baseline CTRL (MSE 87.077, though its MAE is slightly better), Cabin shows a 4.52% MSE improvement.

It is noteworthy that while the R^2 scores do not approach unity, the values achieved by Cabin (up to 0.693 on TWPF and 0.643 on GWPF) represent a significant improvement over the baseline models and are substantial for the highly stochastic and weakly periodic nature of daily aggregated wind power. For operational purposes, where minimizing prediction error is paramount, Cabin's marked reductions in MSE and MAE are particularly impactful. Furthermore, our extensive benchmark includes several state-of-the-art models built on decomposition principles (e.g., DLinear, Autoformer, PatchTST). The fact that Cabin consistently outperforms these methods suggests that its end-to-end representation learning is more effective for this type of non-stationary data than explicit decomposition-based approaches, which often rely on stronger periodic signals than are present in daily wind patterns.

These results highlight the efficacy of Cabin's ARM for feature extraction and CAV (with KAN) for modeling complex temporal dependencies even with only historical power data. The multi-dimensional attention in ARM and non-linear modeling by KAN likely contribute to superior pattern recognition.

Table 3
Hyper-parameter settings for baseline models.

Model	Parameter	Option range
LSTM GRU ED	Hidden size	$\{2^4, 2^5, 2^6\}$
CNN1D	CNN kernel size	3–9 (2 per step)
	CNN out channels	$\{2^2, 2^3, 2^4, 2^5, 2^6\}$
CNNRNN	GRU hidden size	$\{2^4, 2^5, 2^6\}$
	GRU layers	1–3 (1 per step)
CNNRNNRes	Residual window size	1–7 (1 per step)
	Residual ratio	0.1–0.5 (0.1 per step)
LSTNet	Skip window size	1–7 (1 per step)
	Skip GRU hidden size	$\{2^4, 2^5, 2^6\}$
	Skip GRU layers	1–3 (1 per step)
TCN	CNN kernel size	3–9 (2 per step)
	CNN out channels	$\{2^2, 2^3, 2^4, 2^5, 2^6\}$
DLinear	Decomposition kernel size	3–9 (2 per step)
TPA	CNN kernel size	3–9 (2 per step)
	CNN out channels	$\{2^2, 2^3, 2^4, 2^5, 2^6\}$
	GRU hidden size	$\{2^4, 2^5, 2^6\}$
	GRU layers	1–3 (1 per step)
	Highway window size	1–7 (1 per step)
DSANet	CNN kernel size	3–9 (2 per step)
	Attention layers	1–3 (1 per step)
	The numbers of heads	$\{2^2, 2^3, 2^4\}$
	The dimension of the model	$\{2^4, 2^5, 2^6\}$
Transformer	Encoder layers	1–3 (1 per step)
Informer	Decoder layers	1–3 (1 per step)
Autoformer	The label length	1–10 (1 per step)
FEDformer	The numbers of heads	$\{2^2, 2^3, 2^4\}$
STAEformer	The dimension of the model	$\{2^4, 2^5, 2^6\}$
Crossformer	The sequence length	3–7 (2 per step)
Triformer	The patch size	{2, 5}
FiLM	The dimension of the model	$\{2^4, 2^5, 2^6\}$
NHiTS	The number of blocks	1–3 (1 per step)
	The layer number	1–3 (1 per step)
	The hidden size	$\{2^4, 2^5, 2^6\}$
	The pooling size	2–8 (2 per step)
PatchTST	Encoder layers	1–3 (1 per step)
	The numbers of heads	$\{2^2, 2^3, 2^4\}$
	The dimension of the model	$\{2^4, 2^5, 2^6\}$
	The patch length	{2, 5}
	The patch stride	{1, 2, 5}
StemGNN	Block size	1–10 (1 per step)
	Leaky rate	0.1–0.3 (0.1 per step)
AGCRN	The layer number	1–3 (1 per step)
	The hidden dimension size	$\{2^4, 2^5, 2^6\}$
GAIN	GAT hidden size	$\{2^4, 2^5, 2^6\}$
	The number of heads of GAT	$\{2^0, 2^1, 2^2, 2^3, 2^4\}$
MSL	Shapelet size	$\{2^2, 2^3, 2^4, 2^5, 2^6\}$
TCOAT	Residual window size	1–7 (1 per step)
	Residual ratio	0.1–0.5 (0.1 per step)
CoDR	Horizon	1–3 (1 per step)
	The hidden size	$\{2^4, 2^5, 2^6\}$
CTRL	RNN hidden size	$\{2^4, 2^5, 2^6\}$
	RNN layers	1–3 (1 per step)

Baseline Model Category Analysis (Only-target):

- **Traditional Linear Models (GAR, AR, VAR):** Perform adequately for short-term trends but are outperformed by neural models, indicating limitations in capturing non-linearities.
- **RNN-based (LSTM, GRU, ED) & CNN-based (CNN1D, CNNRNN, LSTNet, TCN):** Generally improve over linear models. LSTMs and GRUs effectively capture temporal dependencies. Hybrid CNN-RNN models like LSTNet show good results, but Cabin's specialized ARM and KAN-enhanced CAV seem to extract and model features more effectively. TCNs, while powerful, did not consistently outperform simpler CNNs or RNNs here.
- **Enhanced Linear Models (DLinear, NLinear):** Did not significantly outperform basic linear models, suggesting their decomposition or normalization techniques were not sufficient for the complexity of this wind data.
- **Transformer-based (Transformer, Informer, Autoformer, etc.):** Performance varies. Informer and NHiTS show strong results among Transformers. However, some complex variants (e.g., Autoformer, FEDformer) underperform simpler Transformers or even RNNs on these datasets, suggesting that their intricate mechanisms for long-sequence forecasting might not always translate to superior performance on shorter daily sequences or require extensive tuning. PatchTST performs reasonably well. FiLM consistently performed poorly.
- **Graph-based (StemGNN, STID, AGCRN, GAIN):** Performance is mixed. STID showed good results on TWPF. StemGNN was among the weakest. These models are designed for spatio-temporal data; their application to single-site (TWPF) or multi-site treated as multivariate time series (GWPF without explicit graph structure input) might not fully leverage their strengths.
- **Collaborative Representation Models (MSL, TCOAT, CoDR, CTRL):** CTRL stands out as the best performing baseline model on both datasets in the only-target setting. This highlights the potential of models focusing on learning rich representations. Cabin builds on this concept but with a different architectural approach (ARM's multi-axis attention, KANs) which ultimately yields further improvements.

The consistent superiority of Cabin suggests its architectural design, particularly the ARM's ability to discern feature importance along multiple axes and CAV's KAN-based non-linear mapping, provides a more effective way to model wind power dynamics than the approaches taken by the diverse set of baselines.

Fig. 3 visually compares predictions from Cabin, the best baseline (CTRL), and a less effective baseline (FiLM) against actual wind power values. Cabin demonstrates superior tracking of wind fluctuations, particularly in capturing peaks and rapid changes, which is crucial for grid operations. CTRL also performs well but Cabin often shows a closer fit. FiLM struggles to capture the dynamics, highlighting the advantage of more sophisticated architectures like Cabin.

The Diebold–Mariano (DM) test represents a statistical approach for assessing and contrasting the forecasting capabilities of various prediction models. Statistical test results including DM values and corresponding p-values across different datasets are summarized in the following table. Statistical significance is denoted as: ***, **, *, and None representing significance thresholds of 1%, 5%, 10% and beyond 10%, respectively. As shown in Fig. 4, the results indicate that Cabin demonstrates statistically significant performance improvements compared to baseline models in most cases, with particularly strong significance levels observed on the GWPF dataset. The heat maps reveal consistent patterns of significant differences between Cabin and baseline models across different frameworks, providing robust statistical evidence for Cabin's forecasting effectiveness.

5.2. Fusion performance comparison and analyses

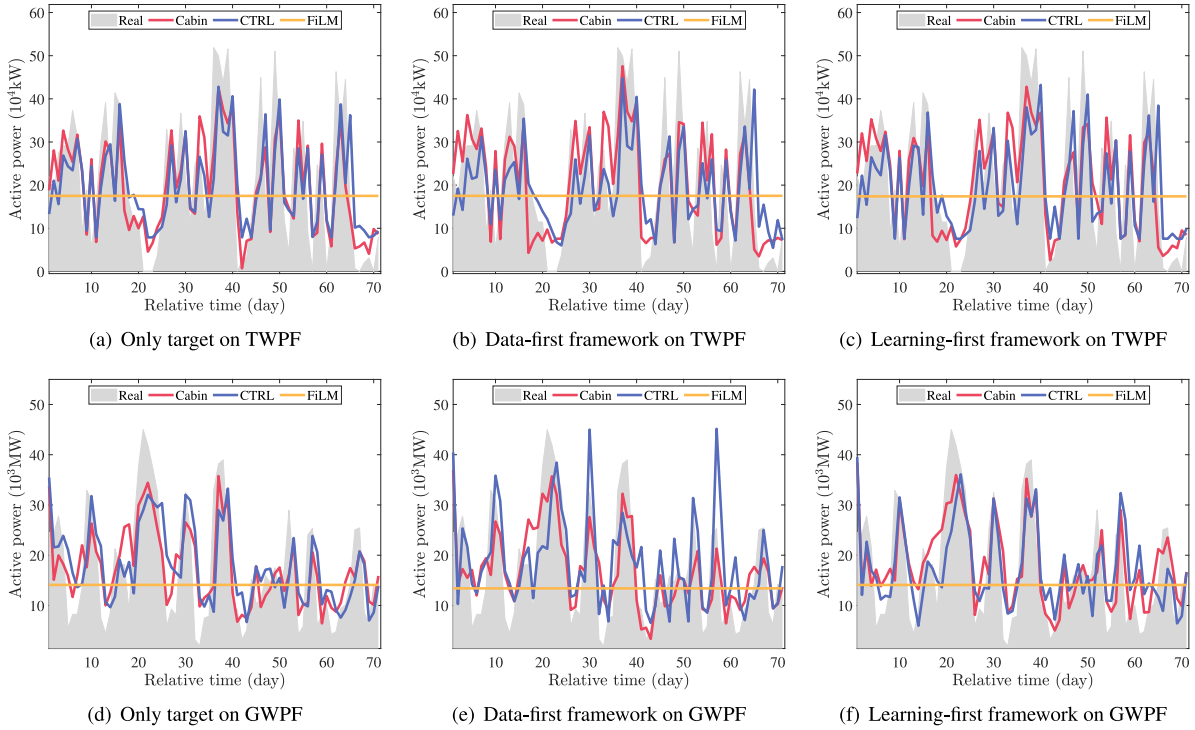
This section analyzes the impact of integrating exogenous ambient variables using the data-first and learning-first frameworks, comparing performance to the only-target framework (Tables 4 and 5).

Cabin's Performance with Fusion Frameworks: Cabin consistently benefits from the integration of ambient variables, outperforming its only-target version and all baseline models under both data-first and learning-first frameworks.

- On TWPF: Cabin's MSE improves from 128.741 (only-target) to 122.110 (data-first, 5.15% improvement) and further to 119.368

Table 4
Performance comparison on the TWPF dataset. Best results in bold, second-best underlined. Wavy lines indicate worst performance among all models for that metric and framework.

Model	Only target				Data-first framework				Learning-first framework			
	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑
GAR	162.797	10.605	0.618	0.587	163.004	10.535	0.618	0.586	160.845	10.416	0.614	0.592
AR	162.691	10.607	0.617	0.587	153.367	10.217	0.600	0.611	153.787	10.051	0.600	0.610
VAR	162.663	10.604	0.617	0.587	153.155	10.153	0.599	0.611	153.666	10.104	0.600	0.610
LSTM	157.044	10.104	0.607	0.601	162.278	10.149	0.617	0.588	165.956	10.672	0.624	0.579
GRU	160.839	10.340	0.614	0.592	160.202	10.167	0.613	0.593	163.480	10.538	0.619	0.585
ED	161.547	10.677	0.615	0.590	172.482	10.986	0.636	0.562	156.308	10.479	0.605	0.603
CNN1D	163.852	10.713	0.620	0.584	162.963	10.643	0.618	0.586	157.221	10.444	0.607	0.601
CNNRNN	158.670	10.515	0.610	0.597	161.068	10.267	0.614	0.591	158.768	10.199	0.610	0.597
CNNRNNRes	163.197	10.692	0.618	0.586	161.739	10.542	0.616	0.589	170.849	11.008	0.633	0.566
LSTNet	160.044	10.410	0.612	0.594	156.086	10.160	0.605	0.604	151.965	10.140	0.597	0.614
TCN	169.090	10.752	0.630	0.571	173.383	10.525	0.637	0.560	160.007	10.517	0.612	0.594
DLinear	162.961	10.612	0.618	0.586	162.933	10.619	0.618	0.586	150.828	9.973	0.595	0.617
NLinear	172.452	10.997	0.636	0.562	163.087	10.608	0.618	0.586	149.317	9.991	0.592	0.621
TPA	159.243	10.563	0.611	0.596	158.556	10.711	0.610	0.597	174.460	10.427	0.639	0.557
DSANet	162.903	10.492	0.618	0.586	177.044	10.792	0.644	0.546	198.229	11.470	0.682	0.491
Transformer	163.935	10.652	0.620	0.584	168.727	10.849	0.629	0.572	144.193	10.019	0.581	0.634
Informer	158.295	10.678	0.609	0.598	175.968	10.812	0.642	0.548	156.416	10.290	0.605	0.603
Autoformer	179.490	11.100	0.649	0.540	182.061	11.119	0.653	0.532	220.033	11.563	0.718	0.435
FEDformer	176.392	10.983	0.643	0.548	181.271	10.642	0.652	0.534	201.589	11.634	0.687	0.482
STAEformer	163.272	10.747	0.619	0.585	171.073	10.398	0.633	0.565	203.479	11.000	0.691	0.477
Crossformer	164.949	10.725	0.622	0.581	150.748	10.537	0.594	0.617	153.069	10.459	0.599	0.611
Triformer	240.477	12.726	0.751	0.382	234.921	12.630	0.742	0.396	233.083	12.599	0.739	0.401
FiLM	250.633	13.529	0.766	0.356	235.188	12.635	0.742	0.395	234.925	12.631	0.742	0.396
NHiTS	158.401	10.380	0.609	0.598	145.519	9.991	0.584	0.626	174.309	10.996	0.639	0.552
PatchTST	166.280	10.716	0.624	0.578	195.501	11.640	0.677	0.497	179.534	10.906	0.649	0.539
StemGNN	235.464	12.639	0.743	0.395	234.721	12.627	0.742	0.397	235.898	12.646	0.744	0.394
STID	144.912	9.818	0.583	0.627	174.859	10.743	0.640	0.550	145.597	9.825	0.584	0.625
MAGNet	164.917	10.715	0.622	0.581	242.161	12.140	0.753	0.378	168.691	10.710	0.629	0.572
AGCRN	162.386	10.649	0.617	0.588	205.810	11.762	0.695	0.471	185.548	11.268	0.659	0.523
GAIN	163.103	10.447	0.618	0.586	173.866	10.813	0.638	0.558	168.321	10.734	0.628	0.573
MSL	154.822	10.306	0.602	0.607	181.126	10.852	0.652	0.534	146.832	10.069	0.587	0.622
TCOAT	164.905	10.782	0.622	0.581	151.337	10.084	0.596	0.616	157.440	10.157	0.607	0.600
CoDR	162.286	10.621	0.617	0.588	187.444	11.066	0.663	0.518	166.875	10.699	0.625	0.576
CTRL	139.854	9.176	0.573	0.640	169.054	10.882	0.629	0.571	156.279	10.459	0.605	0.603
Cabin (Ours)	**128.741**	**9.150**	**0.549**	**0.669**	**122.110**	**8.908**	**0.535**	**0.686**	**119.368**	**9.020**	**0.529**	**0.693**



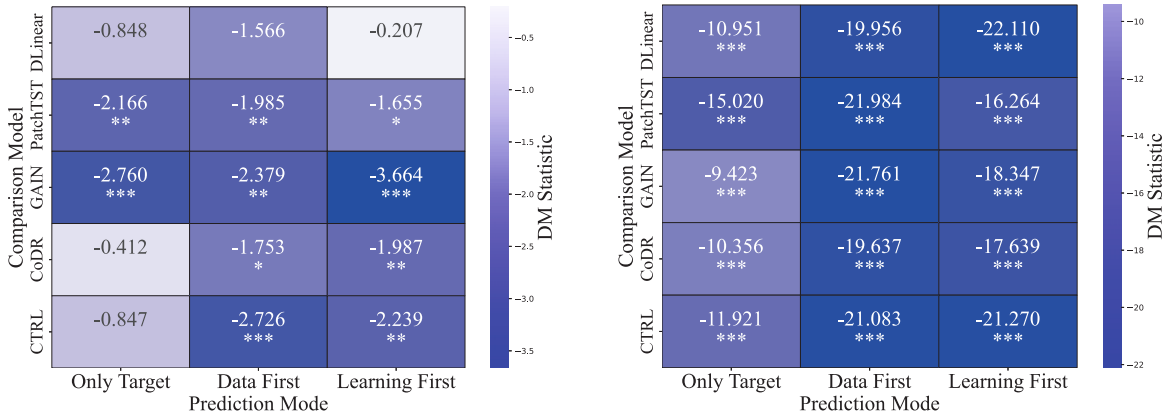
(a) Only target on TWPF (b) Data-first framework on TWPF (c) Learning-first framework on TWPF

(d) Only target on GWPF (e) Data-first framework on GWPF (f) Learning-first framework on GWPF

Fig. 3. Visualized comparisons of ground truth values with predictions from Cabin, CTRL (best performing baseline), and FiLM (a representative poorly performing baseline) on test sets of TWPF (a–c) and GWPF (d–f) datasets across the three architectural frameworks. This selection allows for assessing Cabin against a strong competitor and illustrating the range of predictive capabilities. Cabin consistently tracks actual wind power fluctuations more closely, especially at peaks and troughs, across different frameworks and datasets.

Table 5
Performance comparison on the GWPF dataset. Best results in bold, second-best underlined. Wavy lines indicate worst performance.

Model	Only target				Data-first framework				Learning-first framework			
	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑
GAR	105.908	7.936	0.508	0.515	96.556	7.844	0.485	0.558	95.960	7.645	0.484	0.561
AR	105.561	7.951	0.507	0.517	100.355	8.098	0.495	0.540	99.131	7.956	0.492	0.546
VAR	105.433	8.176	0.507	0.517	106.619	8.533	0.510	0.512	99.307	7.970	0.492	0.545
LSTM	97.204	7.755	0.487	0.555	96.906	7.817	0.486	0.556	96.425	7.818	0.485	0.559
GRU	101.228	8.102	0.497	0.536	102.063	7.896	0.499	0.532	97.621	7.843	0.488	0.553
ED	101.541	8.147	0.497	0.535	98.704	7.769	0.491	0.548	98.686	8.088	0.490	0.548
CNN1D	105.229	8.065	0.506	0.518	100.419	8.027	0.495	0.539	99.129	7.761	0.492	0.546
CNNRNN	102.439	8.090	0.500	0.531	99.543	7.942	0.493	0.544	98.094	7.891	0.489	0.551
CNNRNNRes	100.544	7.834	0.495	0.539	100.169	8.223	0.494	0.541	97.286	7.924	0.487	0.554
LSTNet	101.634	7.945	0.498	0.534	100.394	8.064	0.495	0.540	96.273	7.997	0.484	0.559
TCN	108.375	8.312	0.514	0.504	101.295	8.242	0.497	0.536	106.643	8.317	0.510	0.511
DLinear	106.378	8.099	0.509	0.513	97.004	7.908	0.486	0.556	96.676	7.715	0.485	0.557
NLinear	110.001	8.180	0.518	0.496	98.658	7.742	0.490	0.548	95.961	7.816	0.484	0.561
TPA	104.613	8.246	0.505	0.521	104.883	8.097	0.506	0.519	120.587	8.537	0.542	0.447
DSANet	103.621	8.055	0.503	0.525	96.282	7.843	0.484	0.559	99.895	8.098	0.493	0.542
Transformer	106.000	8.329	0.508	0.515	104.818	7.961	0.505	0.520	112.237	8.787	0.523	0.486
Informer	100.120	8.157	0.494	0.541	99.847	7.952	0.493	0.543	102.588	8.304	0.500	0.530
Autoformer	108.551	8.205	0.514	0.503	121.105	8.586	0.543	0.445	114.298	8.057	0.528	0.476
FEDformer	115.985	8.543	0.532	0.469	110.139	8.276	0.518	0.495	130.486	9.158	0.564	0.402
STAEformer	104.323	8.248	0.504	0.522	110.796	8.324	0.520	0.492	110.295	8.124	0.519	0.494
Crossformer	106.125	8.351	0.508	0.514	99.913	8.112	0.494	0.542	108.418	8.276	0.514	0.503
Triformer	219.667	11.021	0.732	0.003	219.667	11.021	0.732	0.003	132.126	9.291	0.568	0.395
FiLM	218.009	10.997	0.729	0.011	218.009	10.997	0.729	0.011	222.293	11.059	0.736	−0.018
NHiTS	100.570	7.870	0.495	0.539	99.316	7.999	0.492	0.545	102.399	8.004	0.500	0.531
PatchTST	107.465	8.040	0.512	0.508	96.478	7.923	0.485	0.558	98.587	8.078	0.490	0.548
StemGNN	221.126	11.107	0.734	−0.003	217.965	10.996	0.729	0.011	214.352	10.951	0.723	0.028
STID	104.691	8.083	0.505	0.520	101.929	8.266	0.498	0.533	98.762	8.010	0.491	0.548
MAGNet	110.675	8.534	0.519	0.493	100.901	7.934	0.496	0.538	100.650	8.048	0.495	0.539
AGCRN	102.683	8.153	0.500	0.530	97.963	7.794	0.489	0.551	103.001	7.801	0.501	0.528
GAIN	102.940	7.971	0.501	0.528	92.918	7.689	0.476	0.575	91.382	7.775	0.472	0.582
MSL	104.389	8.156	0.504	0.522	103.629	8.124	0.503	0.525	97.903	7.743	0.489	0.552
TCOAT	100.376	7.873	0.495	0.540	95.887	7.767	0.483	0.561	93.982	7.560	0.479	0.570
CoDR	99.564	7.887	0.493	0.544	97.144	7.961	0.487	0.555	93.773	7.708	0.478	0.570
CTRL	87.077	**6.723**	0.461	0.601	104.225	8.288	0.504	0.522	94.069	7.608	0.479	0.570
Cabin (Ours)	**83.141**	7.017	**0.450**	**0.619**	**77.890**	**6.823**	**0.436**	**0.643**	**82.785**	**7.153**	**0.449**	**0.621**



(a) TWPF DM test results (Daily resolution)

(b) GWPF DM test results (Daily resolution)

Fig. 4. Diebold–Mariano test results comparing Cabin against baseline models for daily wind power forecasting. Heat maps show statistical significance levels: dark colors indicate significant differences, confirming Cabin's superior performance across both datasets.

(learning-first, 7.28% improvement over only-target). This indicates that for TWPF, a learning-first approach to fuse ambient data is most beneficial for Cabin.

- On GWPF: Cabin's MSE improves from 83.141 (only-target) to 77.890 (data-first, 6.32% improvement). The learning-first variant (MSE 82.785) is slightly worse than only-target on MSE for GWPF but still very competitive and better than data-first on MAE. This suggests that for GWPF with Cabin, early fusion (data-first) is more effective.

The differential benefit of data-first vs. learning-first across datasets highlights that the optimal fusion strategy can be data-dependent. Cabin's architecture allows for exploring these options. The ARM and CAV modules are clearly effective in leveraging the additional information from ambient variables.

Baseline Models with Fusion Frameworks:

- Many baselines also show improvements with fusion, but often inconsistently. For example, linear models (AR, VAR) benefit, especially with the learning-first setup. Some RNN/CNN hybrids like LSTNet also improve.

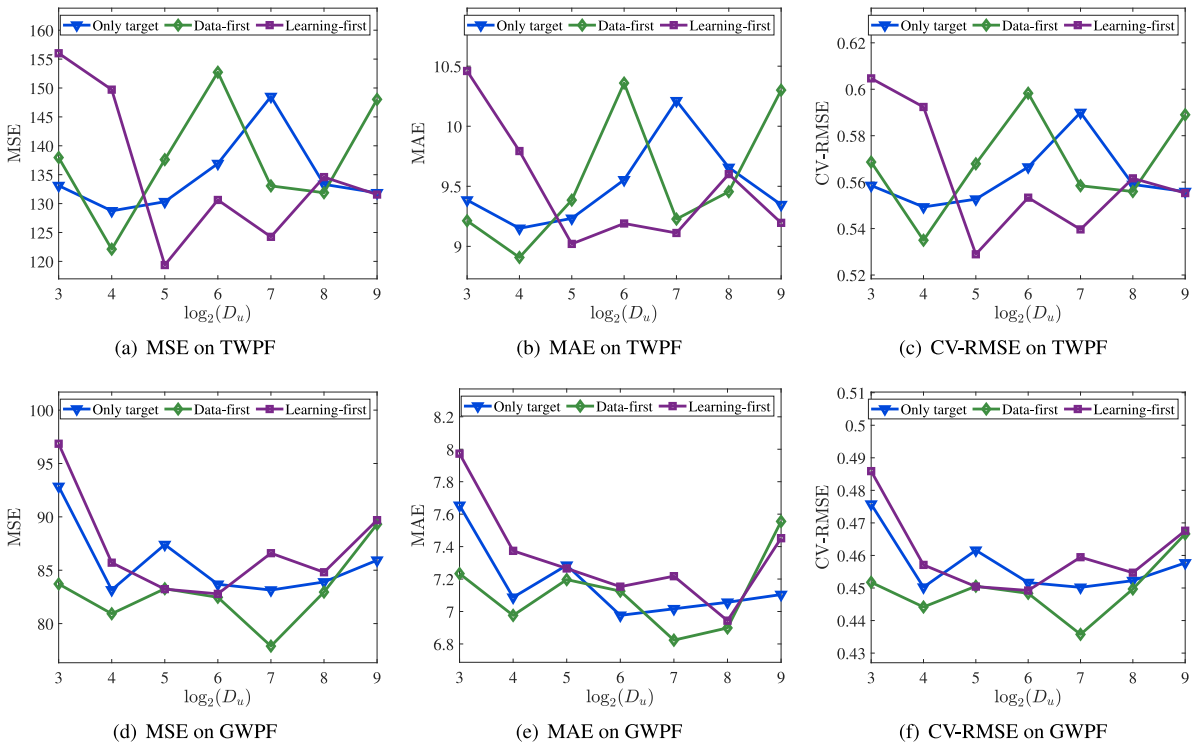


Fig. 5. Performance sensitivity to the Unified Representation Number (D_u) in the CAV module's mapping unit for Cabin on TWPF (a–c) and GWPF (d–f) datasets, across its three architectural variants. Results show optimal D_u varies (e.g., 16–32 for TWPF, 64–128 for GWPF), indicating data/architecture dependency. Both too low and too high D_u can degrade performance, highlighting a trade-off between model capacity and overfitting/efficiency.

- Transformer-based models show mixed results. Some, like Transformer and Crossformer on TWPF, benefit significantly from fusion, particularly with learning-first or data-first respectively. Others, like NHiTS, sometimes see performance degradation with fusion, suggesting their architectures are not inherently designed to optimally integrate additional asynchronous variables without careful adaptation.
- GAIN (a graph-based model) shows strong performance with fusion on GWPF, becoming the best baseline under fusion for that dataset, indicating its graph attention mechanism can effectively incorporate exogenous information.
- CTRL, the best only-target baseline, does not consistently maintain its top rank among baselines when fusion is applied (e.g., on TWPF data-first, its MSE increases). This suggests that simply adding exogenous data to a strong univariate model does not guarantee improvement without specialized fusion mechanisms.

Cabin's consistent top performance across all three frameworks underscores its robust design for incorporating ambient variables. The ARM's multi-dimensional feature extraction and CAV's KAN-based synthesis appear particularly adept at exploiting these additional data sources, more so than the ad-hoc fusion applied to many baselines(see Fig. 5).

5.3. Sensitivity analyses

We further investigate the sensitivity of Cabin to the dimensionality of the unified representation D_u in the CAV module. As shown in Fig. 5, the results exhibit a clear dataset- and architecture-dependent pattern. For the TWPF dataset, panels (a)–(c) demonstrate that relatively small values of D_u (between 16 and 32) yield the most favorable performance across all metrics (MSE, MAE, CV-RMSE). Increasing D_u beyond this range often leads to higher errors, suggesting that excessive capacity introduces overfitting when training samples are limited. In contrast, the GWPF dataset, illustrated in panels (d)–(f), favors larger values of D_u (64 to 128). This is particularly evident in Fig. 5(f), where the CV-RMSE sharply decreases when D_u increases from 16 to 64, then stabilizes before rising again at $D_u = 512$. The richer and more diverse GWPF dataset thus benefits from a higher representation capacity, but overly large dimensions eventually harm generalization. Across both datasets, all three Cabin variants (only-target, data-first, learning-first) follow similar trends, though the learning-first configuration tends to be more robust to increases in D_u. These observations confirm that Cabins performance is sensitive to the choice of D_u, and that a balanced selection is crucial: too small values limit representational power, while too large values lead to inefficiency and potential overfitting. We further investigate the sensitivity of Cabin to the dimensionality of the unified representation D_u in the CAV module. As shown in Fig. 5, the results exhibit a clear dataset- and architecture-dependent pattern. For the TWPF dataset, panels (a)–(c) demonstrate that relatively small values of D_u (between 16 and 32) yield the most favorable performance across all metrics (MSE, MAE, CV-RMSE). Increasing D_u beyond this range often leads to higher errors, suggesting that excessive capacity introduces overfitting when training samples are limited. In contrast, the GWPF dataset, illustrated in panels (d)–(f), favors larger values of D_u (64 to 128). This is particularly evident in Fig. 5(f), where the CV-RMSE sharply decreases when D_u increases from 16 to 64, then stabilizes before rising again at $D_u = 512$. The richer and more diverse GWPF dataset thus benefits from a higher representation capacity, but overly large dimensions eventually harm generalization. Across both datasets, all three Cabin variants (only-target, data-first, learning-first) follow similar trends, though the learning-first configuration tends to be more robust to increases in D_u. These observations confirm that Cabin's performance is sensitive to the choice of D_u, and that a balanced selection is crucial: too small values limit representational power, while too large values lead to inefficiency and potential overfitting.

Table 6
Ablation study of ARM components in Cabin. Best results in bold, second-best underlined, worst in wavy lines.

Dataset	Model variant	Only target			Data-first framework			Learning-first framework		
		MSE	MAE	CV-RMSE	MSE	MAE	CV-RMSE	MSE	MAE	CV-RMSE
TWPF	Cabin (Full)	**128.741**	**9.150**	**0.549**	**122.110**	**8.908**	**0.535**	**119.368**	**9.020**	**0.529**
	w/o ARM (Temporal Softmax)	150.537	10.357	0.594	143.366	10.055	0.580	147.854	10.166	0.589
	w/o ARM (Feature-wise Softmax)	147.859	10.287	0.589	148.754	10.274	0.590	148.977	10.339	0.591
	w/o ARM (Sample Softmax)	153.445	10.456	0.600	154.499	10.371	0.602	153.272	10.380	0.599
GWPF	Cabin (Full)	**83.141**	**7.017**	**0.450**	**77.890**	**6.823**	**0.436**	**82.785**	**7.153**	**0.449**
	w/o ARM (Temporal Softmax)	88.299	7.280	0.464	85.255	7.235	0.456	85.970	7.376	0.458
	w/o ARM (Feature-wise Softmax)	84.549	7.139	0.454	82.551	7.104	0.449	85.247	7.279	0.456
	w/o ARM (Sample Softmax)	101.380	8.001	0.497	102.370	8.103	0.500	97.355	7.782	0.487

5.4. Ablation study

Table 6 presents the ablation study on ARM's softmax components. The full Cabin model consistently performs best. Removing Sample Softmax causes the most significant performance drop across all scenarios (e.g., MSE increases by 19.19% on TWPF only-target, 31.43% on GWPF data-first). This underscores its critical role in dynamically weighting or normalizing samples within a batch for feature extraction. Temporal Softmax and Feature-wise Softmax also contribute significantly; their removal leads to notable performance degradation, confirming their importance in capturing temporal dynamics and feature interdependencies, respectively. The learning-first framework sometimes shows slightly more resilience to component removal than data-first, possibly because separate initial processing offers some robustness. However, all components are clearly beneficial. This study primarily focuses on ARM. The CAV module, especially its KAN component, is crucial for effectively modeling the non-linear relationships from the rich features ARM provides. KANs are chosen over standard MLPs for their potential in superior function approximation for complex patterns and improved interpretability via learnable activation functions, as discussed in recent KAN literature [14,41]. While a direct ablation of KAN vs. MLP in CAV was not performed due to the extensive existing experiments, the strong overall performance of Cabin, which integrates KAN, supports its contribution. Future work could explore this specific ablation.

6. Conclusions and future work

This paper introduced Cabin, an adaptive and collaborative framework for wind power forecasting. Cabin's core strength lies in its structured approach to integrating historical power data with ambient meteorological variables. The Ambient Representation Module (ARM) effectively extracts multi-dimensional features by discerning importance across sample, temporal, and feature axes. The Collaboration of Ambient Variables (CAV) module, leveraging temporal convolutions and, notably, Kolmogorov–Arnold Networks (KAN), synergistically synthesizes these features to model complex non-linear dependencies. Cabin's "collaborative" nature refers to this sophisticated joint modeling, while its "adaptive" characteristic is demonstrated through three configurations (only-target, data-first, learning-first) that cater to varying data availability, ensuring robustness. Comprehensive evaluations against 34 baselines on two public benchmark datasets showed Cabin's superior performance, achieving significant error reductions (e.g., MSE reduction up to 48.63%, CV-RMSE up to 28.33%, and strong R^2 scores). These results are further statistically validated by Diebold–Mariano tests. Moreover, new experiments detailed in Appendix B demonstrate Cabin's robust and competitive performance on hourly resolution data, confirming its adaptability across different temporal granularities. These findings collectively highlight Cabin's potential to enhance predictive reliability, thereby contributing to grid stability and efficient renewable energy integration.

Future work could involve further refinement of ARM's attention mechanisms and CAV's KAN architecture for even greater adaptability to highly dynamic conditions. Exploring the interpretability aspects of KAN within Cabin could yield insights into influential factors. Extending Cabin to incorporate explicit spatial modeling for large wind farms or investigating its application to other renewable energy sources like solar power are promising directions. Additionally, conducting more extensive ablation studies, for instance, comparing KANs with traditional MLPs within the CAV module, could further delineate component contributions. Developing mechanisms for online learning or continual adaptation would also enhance Cabin's practical utility in real-time operational settings. Finally, a direct comparative study against hybrid models that combine signal decomposition techniques, such as Variational Mode Decomposition (VMD), with recurrent networks could further benchmark the effectiveness of Cabin's end-to-end learning paradigm.

CRediT authorship contribution statement

Senzhen Wu: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Data curation, Conceptualization. **Yu Chen:** Writing – review & editing, Writing – original draft, Software, Data curation. **Xinhao He:** Writing – original draft, Software, Methodology, Data curation. **Zhijin Wang:** Writing – review & editing, Writing – original draft, Supervision, Software, Resources, Project administration, Methodology, Investigation, Conceptualization. **Xiufeng Liu:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Formal analysis. **Yonggang Fu:** Writing – review & editing, Resources, Methodology.

Data and code availability

The datasets used in this study are publicly available: TWPF [43] and GWPF [44]. The code for the Cabin model and experiments will be made available on a public repository (e.g., GitHub) upon publication of this manuscript. A link will be provided in the final version.

Declaration of competing interest

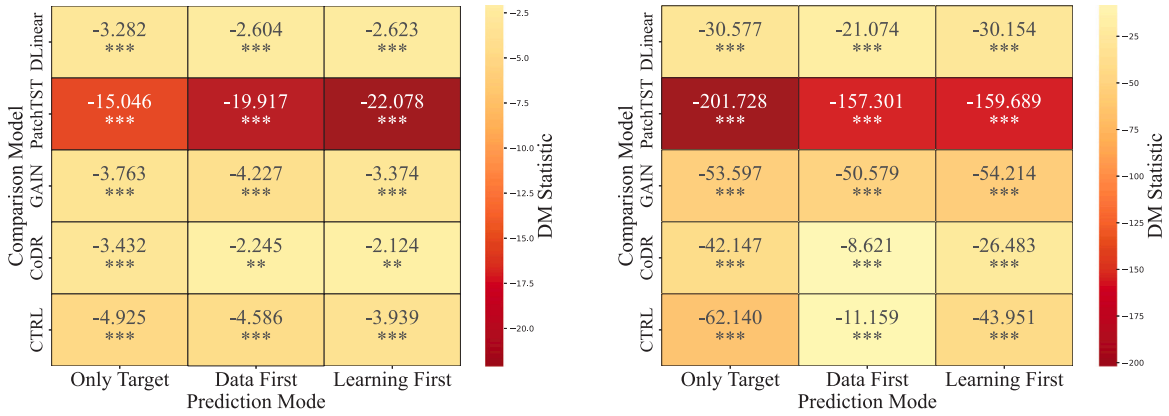
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partially supported in part by the BEGONIA project, which has received funding from the European Commission under grant agreement No. 01133306.

Appendix A. Abbreviations and meanings

See Table A.7.



(a) TWPF DM test results (Hourly resolution)

(b) GWPF DM test results (Hourly resolution)

Fig. B.6. Diebold–Mariano test results comparing Cabin against baseline models for hourly wind power forecasting. Heat maps show statistical significance levels: dark colors indicate significant differences, confirming Cabin's superior performance across both datasets.

Table A.7
Abbreviations and meanings.

Abbreviation	Meaning
AGCRN	Adaptive Graph Convolutional Recurrent Network
AR	Autoregression
ARM	Ambient Representation Module
CAV	Collaboration of Ambient Variables module
CNN	Convolutional Neural Network
CNN1D	One-Dimensional CNN
CNNRNN	Convolutional Recurrent Neural Network
CNNRNNRes	Residual Convolutional RNN
CoDR	Collaborative Directional Representation
CTRL	Collaborative Temporal Representation Learning
CV-RMSE	Coefficient of Variation of RMSE
DLinear	Decomposition-Linear
DSANet	Dual Self-Attention Network
ED	Encoder-Decoder
FEDformer	Frequency Enhanced Decomposed Transformer (official: Seasonal-Trend Decomposition with Fourier Mix)
FiLM	Frequency improved Legendre Memory (official: Feature-Wise Linear Modulation)
GAIN	Graph Ambient Intelligent Network
GAR	Global Autoregression
GRU	Gated Recurrent Unit
GWPF	Greece Wind Power Forecasting
KAN	Kolmogorov–Arnold Networks
LSTM	Long Short-Term Memory
LSTNet	Long- and Short-Term Network
MAE	Mean Absolute Error
MSE	Mean Square Error
MSL	Multivariate Shapelet Learning
MAGNet	Multi-scale Attention and Evolutionary Graph Structure Network
NHiTS	Neural Hierarchical Interpolation for Time Series
NLinear	Non-Linear
PatchTST	Patch-based Time Series Transformer
R^2	R-squared Score (Coefficient of Determination)
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
STAEformer	Spatio-Temporal Adaptive Embedding Transformer
StemGNN	Spectral Temporal Graph Neural Network
STID	Spatial-Temporal Identity
TCN	Temporal Convolutional Net
TCOAT	Temporal Collaborative Attention
TPA	Temporal Pattern Attention
TWPF	Turkey Wind Power Forecasting
VAR	Vector Autoregression

Appendix B. Performance comparison with on hourly resolution datasets

The performance of Cabin on both the TWPF and GWPF datasets is evaluated in this appendix using hourly resolution data to demonstrate its robustness across different temporal scales. Tables B.8 and B.9 present comprehensive comparisons against five advanced representative baseline models, including DLinear, PatchTST, GAIN, CoDR, and CTRL. These baselines were selected to represent different model categories: enhanced linear models (DLinear), patch-based transformers (PatchTST), graph-based approaches (GAIN), and collaborative representation methods (CoDR, CTRL). The experimental setup follows the same three-framework configuration (only-target, data-first, learning-first) with appropriate hyperparameter tuning for hourly forecasting. The input window length is set to 24 h with a prediction horizon of 1 h ahead. The batch size is set to 32 for the baseline models.

The results on the TWPF hourly dataset show that Cabin performs competitively across all metrics and frameworks. Cabin achieves MSE scores of 5.031 (only-target), 5.393 (data-first), and 5.399 (learning-first), which represent improvements over CoDR, one of the stronger baselines, which achieves MSE values of 6.354, 6.166, and 6.104 respectively. PatchTST appears to face challenges with the hourly wind power dynamics, showing higher error rates (MSE ranging from 15.005 to 48.776). Among other baselines, DLinear demonstrates stable performance across frameworks, while GAIN and CTRL show reasonable effectiveness.

For the GWPF hourly dataset, Cabin maintains competitive performance with MSE scores of 24.112 (only-target), 25.673 (data-first), and 24.003 (learning-first). The learning-first framework achieves a notable R^2 score of 0.994. CoDR proves to be a strong baseline competitor, achieving competitive performance in data-first (MSE: 26.781) and learning-first (MSE: 27.268) frameworks. DLinear performs reasonably well in the only-target scenario. PatchTST shows some limitations for this particular forecasting task with higher MSE values in fusion frameworks. These findings suggest that Cabin's architecture demonstrates consistent performance across datasets with varying temporal resolutions and data characteristics.

The DM-test results for the hourly datasets are presented in Fig. B.6. The results suggest that Cabin shows competitive performance compared to most baseline models across both datasets, with statistically significant improvements observed in many scenarios, particularly in the only-target and learning-first frameworks. This provides additional evidence supporting Cabin's effectiveness and adaptability to different temporal resolutions.

Table B.8
Performance comparison on the TWPF dataset (hourly). Best results in bold, second-best underlined. Wavy lines indicate worst performance.

Model	Only target				Data-first framework				Learning-first framework			
	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑
DLinear	6.287	1.644	0.263	0.904	6.299	1.647	0.263	0.904	6.284	1.640	0.263	0.904
PatchTST	15.005	2.807	0.406	0.772	48.776	5.036	0.733	0.258	19.775	3.577	0.467	0.699
GAIN	6.475	1.646	0.267	0.902	6.933	1.726	0.276	0.895	6.593	1.658	0.269	0.900
CoDR	6.354	1.647	0.264	0.903	6.166	1.652	0.261	0.906	6.104	1.640	0.259	0.907
CTRL	6.958	1.714	0.277	0.894	7.102	1.781	0.280	0.892	6.796	1.702	0.274	0.897
Cabin (Ours)	**5.031**	**1.488**	**0.235**	**0.923**	**5.393**	**1.576**	**0.244**	**0.918**	**5.399**	**1.590**	**0.244**	**0.918**

Table B.9
Performance comparison on the GWPF dataset (hourly). Best results in bold, second-best underlined. Wavy lines indicate the worst performance.

Model	Only target				Data-first framework				Learning-first framework			
	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑	MSE ↓	MAE ↓	CV-RMSE ↓	R^2 ↑
DLinear	27.277	3.833	0.062	0.993	28.451	3.930	0.063	0.992	27.666	3.874	0.062	0.993
PatchTST	196.812	10.802	0.165	0.948	98.306	7.491	0.117	0.974	92.808	7.285	0.114	0.975
GAIN	29.521	4.003	0.064	0.992	33.425	4.242	0.068	0.991	31.979	4.181	0.067	0.992
CoDR	28.673	3.944	0.063	0.992	26.781	3.827	0.061	0.993	27.268	3.859	0.062	0.993
CTRL	31.315	4.130	0.066	0.992	26.813	3.821	0.061	0.993	29.791	4.051	0.064	0.992
Cabin (Ours)	**24.112**	**3.568**	**0.058**	**0.994**	**25.673**	**3.716**	**0.060**	**0.993**	**24.003**	**3.584**	**0.058**	**0.994**

Data availability

The code for the Cabin model and all experiments described in this manuscript is publicly available on GitHub at: "https://github.com/xiufengliu/cabin-wind-forecasting" The repository includes the complete implementation of the Cabin framework, example usage scripts, and documentation for reproducing the experimental results presented in the paper.

References

[1] Gilbert C, Browell J, McMillan D. Leveraging turbine-level data for improved probabilistic wind power forecasting. IEEE Trans Sustain Energy 2019;11(3):1152–60. http://dx.doi.org/10.1109/tste.2019.2920085.

[2] Phipps K, Lerch S, Andersson M, Mikut R, Hagenmeyer V, Ludwig N. Evaluating ensemble post-processing for wind power forecasts. Wind Energy 2022;25(8):1379–405. http://dx.doi.org/10.1002/we.2736.

[3] Baidya Roy S, Traiteur JJ. Impacts of wind farms on surface air temperatures. Proc Natl Acad Sci 2010;107(42):17899–904. http://dx.doi.org/10.1073/pnas.1000493107.

[4] Chen P, Pedersen T, Bak-Jensen B, Chen Z. ARIMA-based time series model of stochastic wind power generation. IEEE Trans Power Syst 2009;25(2):667–76. http://dx.doi.org/10.1109/tpwrs.2009.2033277.

[5] Snaiki R, Jamali A, Rahem A, Shabani M, Barjenbruch BL. A metaheuristic-optimization-based neural network for icing prediction on transmission lines. Cold Reg Sci & Technol 2024;104249. http://dx.doi.org/10.1016/j.coldregions.2024.104249.

[6] Vespucci MT, Maggioni F, Bertocchi MI, Innorta M. A stochastic model for the daily coordination of pumped storage hydro plants and wind power plants. Ann Oper Res 2012;193:91–105. http://dx.doi.org/10.1007/s10479-010-0756-4.

[7] Yan J, Li P, Huang Y. A short-term wind power scenario generation method based on conditional diffusion model. In: 2023 IEEE sustainable power and energy conference. ISPEC, IEEE; 2023, p. 1–6. http://dx.doi.org/10.1109/ispec58282.2023.10403004.

[8] Dai Y, Zhang M, Jiang F, Zhang J, Liu M, Hu W. Wind speed multi-step prediction based on the comparison of wind characteristics and error correction: Focusing on periodic thermally-developed winds. Eng Appl Artif Intell 2024;136:108924. http://dx.doi.org/10.1016/j.engappai.2024.108924.

[9] Depci T, İnci M, Savrun MM, Büyük M. A review on wind power forecasting regarding impacts on the system operation, technical challenges, and applications. Energy Technol 2022;10(8):2101061. http://dx.doi.org/10.1002/ente.202101061.

[10] Sari AP, Suzuki H, Kitajima T, Yasuno T, Prasetya DA, Arifuddin R. Short-term wind speed and direction forecasting by 3DCNN and deep convolutional LSTM. IEEJ Trans Electr Electron Eng 2022;17(11):1620–8. http://dx.doi.org/10.1002/tee.23669.

[11] Cali U, Sharma V. Short-term wind power forecasting using long-short term memory based recurrent neural network model and variable selection. Int J Smart Grid Clean Energy 2019;8(2):103–10. http://dx.doi.org/10.1016/j.energy.2020.118980.

[12] Singh PK, Singh N, Negi R. Short-term wind power forecasting using wavelet-based hybrid recurrent dynamic neural networks. Int J Perform Eng 2019;15(7):1772. http://dx.doi.org/10.1109/iria53009.2021.9588723.

[13] Wang Z, Zhang J, Zhang Y, Huang C, Wang L. Short-term wind speed forecasting based on information of neighboring wind farms. IEEE Access 2020;8:16760–70. http://dx.doi.org/10.1109/access.2020.2966268.

[14] Liu Z, Wang Y, Vaidya S, Ruehle F, Halverson J, Soljačić M, Hou TY, Tegmark M. Kan: Kolmogorov-arnold networks. 2024, http://dx.doi.org/10.48550/arxiv.2404.19756, arXiv preprint arXiv:2404.19756.

[15] Kim Y, Hur J. An ensemble forecasting model of wind power outputs based on improved statistical approaches. Energies 2020;13(5):1071. http://dx.doi.org/10.3390/en13051071.

[16] Zhang Y, Kong X, Wang J, Wang H, Cheng X. Wind power forecasting system with data enhancement and algorithm improvement. Renew Sustain Energy Rev 2024;196:114349. http://dx.doi.org/10.1016/j.rser.2024.114349.

[17] Shahid F, Zameer A, Muneeb M. A novel genetic LSTM model for wind power forecast. Energy 2021;223:120069. http://dx.doi.org/10.1016/j.energy.2021.120069.

[18] Xiao Y, Zou C, Chi H, Fang R. Boosted GRU model for short-term forecasting of wind power with feature-weighted principal component analysis. Energy 2023;267:126503. http://dx.doi.org/10.1016/j.energy.2022.126503.

[19] Ramadevi B, Kasi VR, Bingi K. Hybrid LSTM-based fractional-order neural network for Jeju Island's wind farm power forecasting. Fract Fract 2024;8(3):149. http://dx.doi.org/10.3390/fractalfract8030149.

[20] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. Virtual Event: AAAI Press; 2021, p. 11106–15. http://dx.doi.org/10.1609/aaai.v35i12.17325.

[21] Zeng A, Chen M, Zhang L, Xu Q. Are transformers effective for time series forecasting? In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, 2023, p. 11121–8. http://dx.doi.org/10.1609/aaai.v37i9.26317.

[22] Wang X, Zhu M, Bo D, Cui P, Shi C, Pei J. Am-gcn: Adaptive multi-channel graph convolutional networks. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020, p. 1243–53. http://dx.doi.org/10.48550/arxiv.2007.02265.

[23] Li R, Lu H, Cui C, Ma S. Adaptive graph residual network for hand shape estimation in single images. In: Fourth international conference on computer vision and information technology. CVIT 2023, vol. 12984, SPIE; 2024, p. 46–56. http://dx.doi.org/10.1117/12.3013289.

[24] Abdel-Aal RE, Elhadidy MA, Shaahid S. Modeling and forecasting the mean hourly wind speed time series using GMDH-based abductive networks. Renew Energy 2009;34(7):1686–99. http://dx.doi.org/10.1016/j.renene.2009.01.001.

[25] Xiong B, Fu M, Cai Q, Li X, Lou L, Ma H, Meng X, Wang Z. Forecasting ultra-short-term wind power by multiview gated recurrent unit neural network. Energy Sci Eng 2022;10(10):3972–86. http://dx.doi.org/10.1002/ese3.1263.

[26] Barber S, Lima LAM, Sakagami Y, Quick J, Latiffianti E, Liu Y, Ferrari R, Letzgus S, Zhang X, Hammer F. Enabling co-innovation for a successful digital transformation in wind energy using a new digital ecosystem and a fault detection case study. Energies 2022;15(15):5638. http://dx.doi.org/10.3390/en15155638.

[27] Jenkel L, Jonas S, Meyer A. Privacy-preserving fleet-wide learning of wind turbine conditions with federated learning. Energies 2023;16(17):6377. http://dx.doi.org/10.3390/en16176377.

[28] Tang Y, Zhang S, Zhang Z. A privacy-preserving framework integrating federated learning and transfer learning for wind power forecasting. Energy 2024;286:129639. http://dx.doi.org/10.1016/j.energy.2023.129639.
[29] Zhang H, Yan J, Liu Y, Gao Y, Han S, Li L. Multi-source and temporal attention network for probabilistic wind power prediction. IEEE Trans Sustain Energy 2021;12(4):2205–18. http://dx.doi.org/10.1109/tste.2021.3086851.
[30] Jonas S, Winter K, Brodbeck B, Meyer A. Bias correction of wind power forecasts with SCADA data and continuous learning. In: Journal of physics: conference series. vol. 2767, IOP Publishing; 2024, 092061. http://dx.doi.org/10.1088/1742-6596/2767/9/092061.
[31] Haupt SE, McCandless TC, Dettling S, Alessandrini S, Lee JA, Linden S, Petzke W, Brummet T, Nguyen N, Kosović B, et al. Combining artificial intelligence with physics-based methods for probabilistic renewable energy forecasting. Energies 2020;13(8):1979. http://dx.doi.org/10.3390/en13081979.
[32] Jin H, Shi L, Chen X, Qian B, Yang B, Jin H. Probabilistic wind power forecasting using selective ensemble of finite mixture Gaussian process regression models. Renew Energy 2021;174:1–18. http://dx.doi.org/10.1016/j.renene.2021.04.028.
[33] Wu H, Xu Z. Multi-energy load forecasting in integrated energy systems: A spatial-temporal adaptive personalized federated learning approach. IEEE Trans Ind Inform 2024;20(10):12262–74. http://dx.doi.org/10.1109/tii.2024.3417297.
[34] Xie Y, Zheng J, Taylor G, Hulak D. A short-term wind power prediction method via self-adaptive adjacency matrix and spatiotemporal graph neural networks. Comput Electr Eng 2024;120:109715. http://dx.doi.org/10.1016/j.compeleceng.2024.109715.
[35] Wang Y, Yang Z, Ma J, Jin Q. A wind speed forecasting framework for multiple turbines based on adaptive gate mechanism enhanced multi-graph attention networks. Appl Energy 2024;372:123777. http://dx.doi.org/10.1016/j.apenergy.2024.123777.
[36] Li Z, Ye L, Song X, Luo Y, Pei M, Wang K. Heterogeneous spatiotemporal graph convolution network for multi-modal wind-PV power collaborative prediction. IEEE Trans Power Syst 2024;39(4):5591–608. http://dx.doi.org/10.1109/tpwrs.2023.3342636.
[37] Alhartomi M, Salh A, Audah L, Alzahrani S, Alzahmi A. Enhancing sustainable edge computing offloading via renewable prediction for energy harvesting. IEEE Access 2024. http://dx.doi.org/10.1109/access.2024.3404222.
[38] Xu X, Xu K, Zeng Z, Tang J, He Y, Shi G, Zhang T. Collaborative optimization of multi-energy multi-microgrid system: A hierarchical trust-region multi-agent reinforcement learning approach. Appl Energy 2024;375:123923. http://dx.doi.org/10.1016/j.apenergy.2024.123923.
[39] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the 14th international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings; 2011, p. 315–23.
[40] Bai S, Kolter JZ, Koltun V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. 2018, http://dx.doi.org/10.48550/arxiv.1803.01271, Cs. LG.
[41] Li C, Liu X, Li W, Wang C, Liu H, Yuan Y. U-KAN makes strong backbone for medical image segmentation and generation. 2024, http://dx.doi.org/10.48550/arxiv.2406.02918, arXiv preprint arXiv:2406.02918.
[42] Vaca-Rubio CJ, Blanco L, Pereira R, Caus M. Kolmogorov-arnold networks (kans) for time series analysis. 2024, http://dx.doi.org/10.48550/arXiv.2405.08790, ArXiv Preprint.
[43] Isen B. Turkey wind power forecasting (Turkey WPF) data collection. 2018.
[44] Vartholomaios A, roStamatis Karlos, Kouloumpris E, Tsoumakas G. Short-term renewable energy forecasting in Greece using prophet decomposition and tree-based ensembles. In: Proceedings of the database and expert systems applications - DEXA 2021 workshops. vol. 1479, Linz, Austria: Springer; 2021, p. 227–38. http://dx.doi.org/10.1007/978-3-030-87101-7_22.
[45] Wang Z, Liu X, Huang Y, Zhang P, Fu Y. A multivariate time series graph neural network for district heat load forecasting. Energy 2023;278:127911. http://dx.doi.org/10.1016/j.energy.2023.127911.
[46] Siami-Namini S, Tavakoli N, Namin AS. The performance of LSTM and BiLSTM in forecasting time series. In: Proceedings of the 3rd international conference on smart systems and inventive technology. 2019, p. 3285–92. http://dx.doi.org/10.1109/bigdata47090.2019.9005997.
[47] Chung J, Gulcehre C, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. 2014, arXiv abs/1412.3555.
[48] Cho K, van Merrienboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: Encoder-decoder approaches. In: SSST'14. Doha, Qatar: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation; 2014, p. 103–11. http://dx.doi.org/10.3115/v1/w14-4012.
[49] Wu Y, Yang Y, Nishiura H, Saitoh M. Deep learning for epidemiological predictions. In: Proceedings of the 41st international ACM SIGIR conference on research & development in information retrieval. New York, NY, USA: Association for Computing Machinery; 2018, p. 1085–8. http://dx.doi.org/10.1145/3209978.3210077.
[50] Shih S, Sun F, Lee H. Temporal pattern attention for multivariate time series forecasting. Mach Learn 2019;108(8–9):1421–41. http://dx.doi.org/10.1007/s10994-019-05815-0.
[51] Huang S, Wang D, Wu X, Tang A. DSANet: Dual self-attention network for multivariate time series forecasting. In: Proceedings of the 28th ACM international conference on information and knowledge management. Beijing, China: ACM; 2019, p. 2129–32. http://dx.doi.org/10.1145/3357384.3358132.
[52] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Lu, Polosukhin I. Attention is all you need. In: Proceedings of the 30ty advances in neural information processing systems. vol. 30, Long Beach, CA, USA: Curran Associates, Inc.; 2017, p. 5998–6008. http://dx.doi.org/10.5555/3295222.3295349.
[53] Wu H, Xu J, Wang J, Long M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In: Proceedings of the 34th advances in neural information processing systems. Virtual; 2021, p. 22419–30. http://dx.doi.org/10.48550/arxiv.2106.13008.
[54] Liu H, Dong Z, Jiang R, Deng J, Deng J, Chen Q, Song X. Spatio-temporal adaptive embedding makes vanilla transformer SOTA for traffic forecasting. In: Proceedings of the 32nd ACM international conference on information and knowledge management. New York, NY, USA: ACM; 2023, p. 4125–9. http://dx.doi.org/10.1145/3583780.3615160.
[55] Zeng H, Duan X, Huang X, Cui Q. STA-former: encoding traffic flows with spatio-temporal associations in transformer networks for prediction. Clust Comput 2024;27(7):9693–714. http://dx.doi.org/10.1007/s10586-024-04462-y.
[56] Zhang Y, Yan J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: Proceedings of the 11th international conference on learning representations. OpenReview.net; 2023.
[57] Liu L, Lyu J, Liu S, Tang X. TriFormer: A multi-modal transformer framework for mild cognitive impairment conversion prediction. In: Proceedings of the 20th international symposium on biomedical imaging. 2023, p. 1–4. http://dx.doi.org/10.1109/isbi53787.2023.10230709.
[58] Zhou T, Ma Z, Wang X, Wen Q, Sun L, Yao T, Yin W, Jin R. Film: Frequency improved Legendre memory model for long-term time series forecasting. In: Proceedings of the 36th advances in neural information processing systems. vol. 35, New Orleans, LA, USA; 2022, p. 12677–90. http://dx.doi.org/10.48550/arxiv.2109.03254.
[59] Challu C, Olivares KG, Oreshkin BN, Garza Ramirez F, Mergenthaler Canseco M, Dubrawski A. NHITS: Neural hierarchical interpolation for time series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, 2023, p. 6989–97. http://dx.doi.org/10.1609/aaai.v37i6.25854.
[60] Nie Y, Nguyen NH, Sinthong P, Kalagnanam J. A time series is worth 64 words: Long-term forecasting with transformers. In: Proceedings of the 11th international conference on learning representations. Kigali, Rwanda; 2023, p. 1–12. http://dx.doi.org/10.48550/arxiv.2211.14730.
[61] Cao D, Wang Y, Duan J, Zhang C, Zhu X, Huang C, Tong Y, Xu B, Bai J, Tong J, et al. Spectral temporal graph neural network for multivariate time-series forecasting. In: Proceedings of the 29th advances in neural information processing systems. Virtual; 2020, p. 17766–78. http://dx.doi.org/10.48550/arxiv.2103.077191.
[62] Bai L, Yao L, Li C, Wang X, Wang C. Adaptive graph convolutional recurrent network for traffic forecasting. In: Proceedings of the 22nd advances in neural information processing systems. vol. 33, Curran Associates, Inc.; 2020, p. 17804–15.
[63] Wang Z, Cai B. COVID-19 cases prediction in multiple areas via shapelet learning. Appl Intell 2022;52(1):595–606. http://dx.doi.org/10.1007/s10489-021-02391-6.
[64] Hu Y, Liu H, Wu S, Zhao Y, Wang Z, Liu X. Temporal collaborative attention for wind power forecasting. Appl Energy 2024;357:122502. http://dx.doi.org/10.1016/j.apenergy.2023.122502.
[65] Wang Z, Liu H, Wu S, Liu N, Liu X, Hu Y, Fu Y. Explainable time-varying directional representations for photovoltaic power generation forecasting. J Clean Prod 2024;468:143056. http://dx.doi.org/10.1016/j.jclepro.2024.143056.
[66] Hu Y, Wu S, Chen Y, He X, Xie Z, Wang Z, Liu X, Fu Y. CTRL: Collaborative temporal representation learning for day-ahead wind power forecasting. In: Proceedings of the 8th international conference on electronic information technology and computer engineering. Haikou, China: ACM; 2024, http://dx.doi.org/10.1145/3711129.3711336.
[67] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Proceedings of the 3rd international conference for learning representations. San Diego, CA, USA: OpenReview.net; 2015, p. 1–15. http://dx.doi.org/10.1145/1830483.1830503.