Computers and Electrical Engineering 126 (2025) 110517

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Physics-informed spatio-temporal network with trainable adaptive feature selection for short-term wind speed prediction

Laeeq Aslam [a], Runmin Zou [a,*], Yaohui Huang [a], Ebrahim Shahzad Awan [b], Sharjeel Abid Butt [a], Qian Zhou [a]

[a] School of Automation, Central South University, Changsha, 410083, Hunan, China
[b] School of Engineering, Design and Built Environment, Western Sydney University, Penrith, 2747, New South Wales, Australia

ARTICLE INFO

Keywords:
Wind speed prediction
Physics-informed neural networks
Spatio-temporal modeling

ABSTRACT

Wind speed prediction (WSP) is essential for optimizing wind power generation, enhancing turbine performance and ensuring grid stability. Wind speed prediction models face challenges in effectively integrating complex spatio-temporal data with physical principles. This limitation reduces their accuracy and reliability for optimizing wind power generation and ensuring grid stability. To solve the issue, this study proposes a physics-informed spatio-temporal network (PISTNet) for short-term WSP that effectively integrates spatio-temporal data with physical principles. The proposed model designs a dynamic feature adapter (DFA) module that dynamically emphasizes relevant temporal and spatial information through adaptive masking mechanisms. It also incorporates an extended advection equation-based physical modeling (EPM) module, which provides physics-based WSP fused with neural network-extracted features using a feature fusion module (FFM). An adaptive physics penalty loss (APPL) function is proposed to enhance model's accuracy to selectively enforce physical constraints based on significant prediction deviations. Comprehensive experiments conducted on four diverse datasets from Hamburg, Herning, Palmerston North, and Silkeborg demonstrate that the proposed model consistently outperforms six state-of-the-art prediction methods across multiple evaluation metrics, achieving up to a 7.1% improvement in RMSE, 8.0% in MAE, 2.3% in $1/R^2$ score, and 9. 5% in SMAPE compared to the closest competitors. The findings highlight the potential of combining data-driven and physics-informed approaches to achieve accurate and reliable WSP, thereby contributing to the advancement of wind energy systems and grid management.

1. Introduction

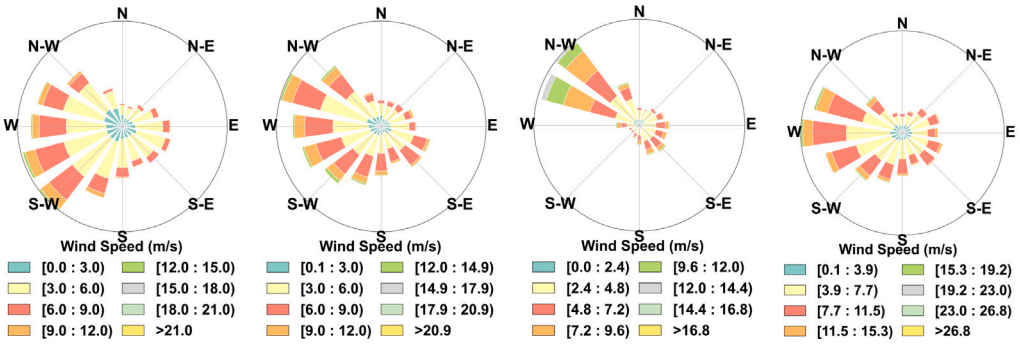
Optimizing wind power generation is essential for sustainable energy solutions. Wind turbines convert air's kinetic energy into electrical power, with power output directly dependent on wind speed. Therefore, wind speed prediction (WSP) is critical for optimizing power generation efficiency. This relationship is described by the wind power equation [1]:

$$P = \frac{1}{2}\rho A v^3 C_p, \tag{1}$$

where P is power output, ρ is air density, A is the swept area of the turbine blades, C_p is the coefficient of performance and v is wind speed. Power output P scales with the cube of wind speed v^3, meaning small wind speed changes cause large power fluctuations.

* Corresponding author.
E-mail address: rmzou@csu.edu.cn (R. Zou).

https://doi.org/10.1016/j.compeleceng.2025.110517
Received 24 January 2025; Received in revised form 10 May 2025; Accepted 6 June 2025
Available online 19 June 2025



(a) Hamburg (b) Herning (c) Palmerston North (d) Silkeborg

Fig. 1. Wind rose diagrams for selected locations.

The other factors (air density, swept area and power coefficient) remain relatively stable for a given turbine, making wind speed the primary determinant of power generation. Therefore, WSP is necessary to optimize turbine performance and maintain grid stability. Despite its importance, accurately predicting wind speed remains a significant challenge due to its stochastic variability, sensitivity to atmospheric turbulence and the interplay of multiple environmental variables across spatial and temporal scales.

To address the inherent complexities in wind speed data, researchers develop various prediction methods categorized into statistical, machine learning and deep learning approaches. Statistical models, such as ARIMA and probability-based methods, are favored for their simplicity and interpretability [2–4]. However, these models often fail to capture the complex non-linear relationships and dynamic spatial–temporal dependencies present in wind speed data. Machine learning techniques, including Support Vector Regression (SVR), Random Forest and Artificial Neural Networks (ANN), model these non-linear patterns and enhance prediction accuracy [5,6]. Deep learning models, particularly Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, demonstrate superior performance in handling intricate temporal and spatial dependencies in wind speed data [5,7]. Additionally, hybrid models that integrate multiple approaches leverage the strengths of each method to further improve prediction accuracy [6]. Comparative analyses indicate that deep learning models consistently outperform traditional statistical methods, underscoring the effectiveness of advanced machine learning and deep learning techniques in enhancing WSP accuracy [5,7]. Despite advancements in machine learning and deep learning techniques, accurately modeling spatio-temporal dependencies and integrating physical principles remains challenging. This requires the development of models that effectively utilize spatio-temporal data and incorporate physical laws governing atmospheric dynamics to improve prediction accuracy and reliability.

The application of spatio-temporal data in WSP is extensively investigated over the past decade [8–12]. Huang et al. [8] integrate LSTM networks with Clayton Copula functions to model the increased coupling and spatial correlation of wind speeds among multiple turbines in a wind farm. This approach effectively captures spatio-temporal dependencies, enhancing prediction accuracy. Moreover, Zhu et al. [9] develop a Predictive Deep Convolutional Neural Network that combines CNNs for spatial feature extraction and Multi-Layer Perceptrons for modeling temporal dependencies. This unified framework outperforms traditional machine learning models by effectively capturing spatio-temporal correlations. Furthermore, Zhang et al. [10] propose a microscale meteorological model integrated with a hybrid deep learning approach. They employ Enhanced Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (ICEEMDAN) for signal decomposition and a hybrid deep learning model for multi-step-ahead predictions, achieving high accuracy in both temporal and spatial forecasting. In another study Agrawal et al. [12] combine CNNs with Spiking Neural Networks to capture space–time characteristics of wind speed. Their model outperforms conventional methods in prediction horizons ranging from 5 min to 1 h, demonstrating improved reliability. Finally, Yan et al. [11] present an end-to-end deep learning architecture utilizing Convolutional LSTM (ConvLSTM), Residual Networks (ResNet) and Three-Dimensional CNNs to predict ultra-short-term wind speeds across multiple locations. This model shows enhanced applicability in complex terrains by effectively capturing spatial and temporal dependencies. Despite these advancements, existing spatio-temporal models often face several limitations. Some models are not end-to-end, requiring separate processes for feature extraction and prediction, which can lead to inefficiencies and reduced accuracy. There is also a lack of efficient feature selection mechanisms, allowing the incorporation of irrelevant spatial features and diminishing prediction performance. Feature selection is particularly crucial for capturing physical phenomena, such as terrain-induced wind flow blockages, where certain areas may be obstructed by hills, causing wind to predominantly flow from specific directions as shown in Fig. 1. This figure displays wind rose diagrams for four different datasets used in this study. It is evident that Hamburg and Herning primarily experience winds from the northwest to southwest, Palmerston North from the northwest and Silkeborg exhibits symmetric patterns with significant western winds. This indicates that for each location, certain features are more significant than those from other locations. Therefore, it is essential for models to dynamically learn location-specific features during the training process. These challenges emphasize the need for more dynamic approaches that combine robust feature selection with physics-informed mechanisms to enhance the reliability and accuracy of WSP.

Feature extraction methods decompose wind speed signals into interpretable components for forecasting. Variational Mode Decomposition (VMD) [13–17] and Empirical Mode Decomposition (EMD) [18–21] isolate intrinsic mode functions from raw data.

For example, EMD-derived components improve prediction accuracy when paired with support vector regression [22]. Advanced EMD variants like EEMD and CEEMDAN further refine decomposition when integrated with machine learning models [23]. Recent studies have advanced decomposition frameworks by integrating adaptive techniques with deep learning architectures. Wu et al. [24] proposed a two-stage EMD-EEMD process optimized using evolutionary algorithms and temporal fusion transformers, achieving reductions in MAPE by 8.4%–15.4% and in RMSE by 22.2%–69% across seasonal datasets. A follow-up study [25] incorporated meteorological feature engineering and wavelet-based frequency analysis, yielding accuracy improvements of 12.6%–65.9% while preserving interpretability through attention mechanisms. These frameworks demonstrate that coupling multi-scale decomposition with domain-specific feature engineering outperforms conventional hybrid models [13,14], establishing new benchmarks for operational forecasting.

Once these features are extracted, optimal feature selection methods identify and prioritize the most relevant features along with the model's hyperparameters [15,26–31]. These heuristic techniques, while effective in isolating significant features, possess inherent limitations. Primarily, they lack adaptability, necessitating separate feature optimization processes in changing wind patterns. Additionally, the computational intensity of these methods makes them unsuitable for deployment on resource-constrained devices such as NVIDIA Jetson platforms or smaller embedded systems. The requirement for extensive parameter tuning not only prolongs the model development lifecycle but also effects its scalability and real-time application capabilities. Consequently, there is a need for dynamic feature selection methods that can adaptively identify relevant features during the training process. Such approaches would reduce computational overhead and enhance the scalability of WSP models, making them more suitable for deployment in diverse and resource-limited environments.

Wind speed dynamics are governed by fundamental physical laws, including the advection equation, which describes the transport of wind speed due to fluid motion. Integrating these physical principles into machine learning models has motivated the development of Physics-Informed Neural Networks (PINNs) for WSP. Recently, several physics-informed models have been proposed. Lagomarsino-Oneto et al. [32] developed supervised learning algorithms that incorporate wind speed and direction measurements alongside topographical data, utilizing statistical methods informed by terrain characteristics to improve prediction accuracy. However, their approach relies primarily on statistical correlations and does not explicitly integrate physical equations or constraints within the model architecture. In contrast, Howland and Dabiri [33] introduced a physics-informed statistical model for wind farm power production by embedding wake superposition and velocity deficit formulations directly into the model structure, thereby enhancing interpretability and prediction accuracy. Nonetheless, their model does not enforce energy conservation, which can result in inaccuracies when downstream turbines extract more power than the available kinetic energy from upstream wakes. Similarly, Aslam et al. [34] integrated Physics-Informed Vectors with neural network architectures such as LSTM and Temporal Convolutional Networks (TCN), incorporating physical knowledge through PIV derived from atmospheric data. They employed a hybrid loss function combining MSE with PIV estimates, facilitating faster convergence and improved performance. However, this approach utilizes a statistical method to compute PIV estimates rather than directly embedding physical laws, making it more of a statistical solution rather than a fully physics-informed model.

Yan et al. [35] implemented a simplified parameterized Navier–Stokes model within their PINN for wind flow field reconstruction to address performance limitations in high Reynolds number turbulent flows, enhancing computational efficiency and training stability. However, this parameterization introduces sensitivity to model inaccuracies, resulting in an imbalanced loss function that inadequately enforces physical constraints. In similar work, Shao et al. [36] developed PIGNN-CFD, a Physics-Informed Graph Neural Network designed for rapid prediction of urban wind fields on unstructured meshes. While PIGNN-CFD leverages graph neural networks to handle complex urban layouts and provides computational speedup compared to traditional CFD simulations, it primarily relies on velocity data from CFD-generated simulations and lacks mechanisms for integrating diverse real-time measurement data such as wind speed, pressure and humidity from multiple spatial locations. Furthermore, PIGNN-CFD does not inherently support real-time adaptability or online deployment, limiting its applicability in dynamic operational environments where wind conditions can change rapidly. Despite recent developments, the use of PINNs for WSP remains limited due to the structure of these models. PINNs combine a physics-based loss with a data loss function, such as mean squared error, modulated by Lagrange multipliers that control the influence of the physics loss. However, physical models for WSP are sensitive to initial conditions, particularly in turbulent environments, leading to errors and deviations from actual behavior. Incorporating these models directly into the training process, even with attenuation through the Lagrange multiplier, does not resolve this issue. The model must minimize both the data loss, based on real wind data and the physics loss, but due to inaccuracies in the physical model, this dual optimization results in suboptimal performance. Therefore, there is a need for a loss function that dynamically imposes a penalty only if the neural network deviates significantly from the physical model predictions.

Hence, the limitations of existing methods are summarized as follows: Convolutional LSTM approaches, although effective in learning hierarchical relationships, are sensitive to noise and irrelevant features. Feature extraction and selection processes are not dynamic and require separate optimization. Additionally, while Convolutional LSTM models can learn both long- and short-term dependencies, they struggle to relate them to the most recent spatio-temporal features. Incorporating physical model predictions also poses a challenge, as errors in these models can significantly degrade overall performance. In response to these challenges, this work introduces three primary contributions that can be summarized as follows:

- **Dynamic Feature Adapter (DFA) Module and Spatio-Temporal Convolutional Recurrent Neural Network (STCRNN)**: The DFA module dynamically filters out noisy and irrelevant features, followed by the STCRNN, which extracts and models both spatial and temporal dependencies from the spatio-temporal data. This combination enhances the model's ability to capture complex patterns and improves prediction accuracy.

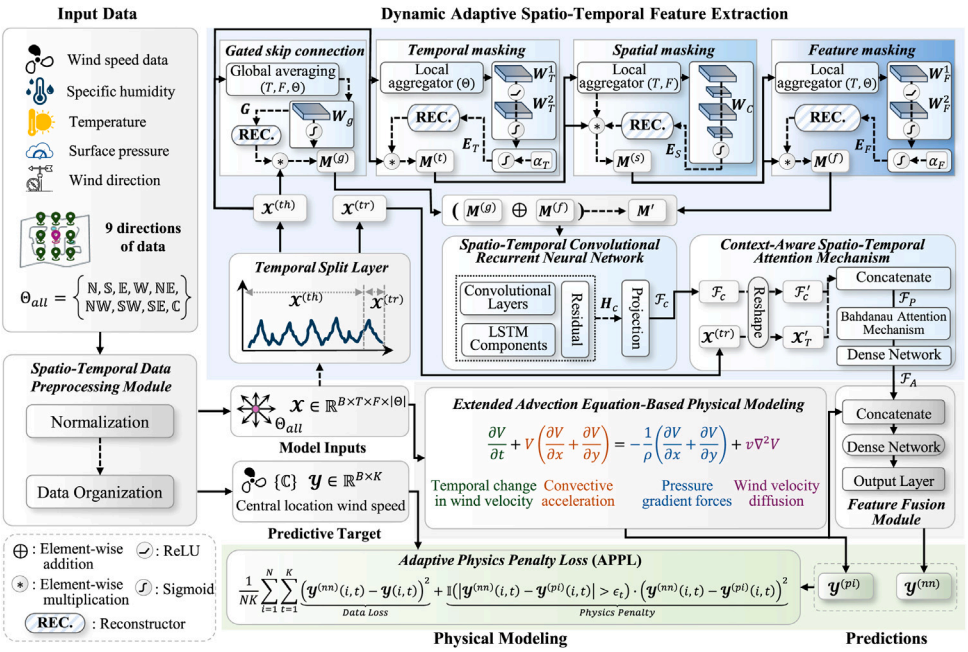


Fig. 2. The overall architecture of the proposed model.

- **Attention Mechanism for Short-Term Context**: Context-Aware Spatio-Temporal Attention Mechanism (CAM) is incorporated to capture the short-term context by relating the features from the past few hours to the most recent hour. This improves the model's ability to generate accurate short-term predictions, which is essential for forecasting in dynamic environments.
- **Integration of Physical Model Predictions with Adaptive Physics Penalty Loss (APPL)**: The model fuses predictions from a physical model based on the advection equation to the features of the neural network using a feature fusion module (FFM) that are then projected to a dense layer for final predictions. Additionally, an Adaptive Physics Penalty Loss is introduced, which applies a penalty only when the data train (neural network) model's predictions deviate significantly from the physical model, with the deviation threshold dynamically adjusted based on terrain-specific physical model performance. This approach ensures the physical model is effectively incorporated without misleading the training process.

The rest of this paper is organized as follows, Section 2 details the proposed methodology, Section 3 describes the experimental setup and results along with a comparison of the proposed model with state-of-the-art methods and finally Section 4 concludes this work.

2. Proposed methodology

This study proposes a Physics-Informed Spatio-Temporal Network (PISTNet) to enhance wind speed prediction by integrating data-driven spatio-temporal feature extraction with physics-based modeling. The framework consists of five key modules. (1) The Spatio-Temporal Data Preprocessing Module structures multi-source atmospheric measurements into normalized tensors with sliding temporal windows to capture directional and temporal dependencies. (2) The Dynamic Adaptive Spatio-Temporal Feature Extraction Module includes four subcomponents: the Temporal Split Layer isolates recent observations; the Dynamic Feature Adapter (DFA) performs hierarchical temporal, spatial, and feature-wise masking with gated skip connections to enable adaptive feature selection; the Spatio-Temporal Convolutional Recurrent Neural Network (STCRNN) fuses convolutional and residual LSTM layers to model complex spatial–temporal interactions; and the Context-Aware Attention Mechanism (CAM) employs Bahdanau attention to integrate recent context with extracted features. (3) The Extended Advection Equation-Based Physical Modeling Module (EPM) introduces a physics-based predictor grounded in fluid dynamics, utilizing the extended advection equation and Runge–Kutta integration to simulate wind velocity evolution. (4) The Feature Fusion Module (FFM) concatenates neural and physical outputs, employing a Mish-activated dense layer to generate a unified representation for final prediction. (5) The Adaptive Physics Penalty Loss (APPL) dynamically imposes physics-based constraints by selectively applying penalty terms only when prediction deviations exceed statistically derived thresholds, thereby balancing empirical accuracy with physical consistency. The overall block diagram is shown in Fig. 2.

2.1. Spatio-temporal data preprocessing module

The dataset consists of spatio-temporal measurements recorded at nine locations: a central location $\mathbb{C}$ and eight surrounding locations in the directions $\Theta_{sur} \in \{\mathbb{N}, \mathbb{S}, \mathbb{E}, \mathbb{W}, \mathbb{NE}, \mathbb{NW}, \mathbb{SW}, \mathbb{SE}\}$. The distance between the central location $\mathbb{C}$ and each surrounding point is D kilometers.

At each location, the recorded features include wind speed, wind direction, surface pressure, humidity and temperature. To ensure comparability across features, each feature $\boldsymbol{x}$ is normalized to the range $[0, 1]$ using min–max normalization, which is defined as:

$$\boldsymbol{x}'_{t,\theta} = \frac{\boldsymbol{x}_{t,\theta} - \boldsymbol{x}_{\theta}^{(\min)}}{\boldsymbol{x}_{\theta}^{(\max)} - \boldsymbol{x}_{\theta}^{(\min)}}, \tag{2}$$

where $\boldsymbol{x}_{t,\theta}$ represents the feature value at time t and location θ. $\boldsymbol{x}_{\theta}^{(\min)}$ and $\boldsymbol{x}_{\theta}^{(\max)}$ are the minimum and maximum values of feature $\boldsymbol{x}$ at location θ across the dataset. Then, the normalized data is segmented into overlapping temporal windows with a step size of S hours. Each temporal window spans T hours of data. For a given window w, the input data $\boldsymbol{X}_w$ and target output $\boldsymbol{Y}_w$ are defined as follows:

$$\boldsymbol{X}_w = \left\{ \boldsymbol{x}'_{t,\theta} \mid t = \{t_0, t_0 + 1, \ldots, t_0 + T - 1\}; \theta \in \Theta_{all} \right\}, \tag{3}$$

$$\boldsymbol{Y}_w = \left\{ \boldsymbol{y}_{t,\theta} \mid t = \{t_0 + T, \ldots, t_0 + T + K - 1\}; \theta \in \{\mathbb{C}\} \right\}, \tag{4}$$

where t_0 is the starting time of window w, $\Theta_{all} = \{\mathbb{C}\} \cup \Theta_{sur}$ represents the set of spatial locations including the central location $\mathbb{C}$ and the eight surrounding directions. For brevity, Θ will be used instead of Θ_{all} in subsequent descriptions. K is the prediction horizon.

The input window $\boldsymbol{X}_w$ includes features from the central location $\mathbb{C}$ as well as the eight surrounding directions. The data is organized into batches, with each batch containing B examples. Each batch is represented as a four-dimensional tensor $\mathcal{X} \in \mathbb{R}^{B \times T \times F \times |\Theta|}$, where B is the batch size, T is the number of time steps, F is the number of features (including wind speed, wind direction, surface pressure, humidity, temperature) and $|\Theta|$ is the number of spatial locations. Each element of the tensor $\mathcal{X}$ is defined as $\mathcal{X}_{b,t,f,\theta}$, where $b \in [1, B]$ indexes the batch, $t \in [1, T]$ indexes the time step, $f \in [1, F]$ indexes the feature and $\theta \in \Theta$ indexes the spatial location. This preprocessing pipeline transforms raw spatio-temporal data into a structured format suitable for neural network training. The normalized data captures temporal dynamics through windowing and spatial dependencies through multiple directions, including the central location. The resulting data structure $\mathcal{X} \in \mathbb{R}^{B \times T \times F \times |\Theta|}$ serves as the input for the proposed model.

2.2. Dynamic adaptive spatio-temporal feature extraction

This module comprises four primary components: a temporal split layer (TSL), DFA module, the spatio-temporal convolutional recurrent neural network (STCRNN) that works as backbone architecture and a context-aware spatio-temporal attention mechanism (CAM). The temporal split layer separates the last time stamp t from the preceding time stamps (up to $t-1$), addressing the need to independently process the most recent temporal information and thereby enhance temporal relevance. The preceding time stamps are processed through the DFA module, which identifies and selects significant features during training, removing the necessity for decomposition and correlation methods and simplifying the feature selection process. The masked input (up to $t-1$) is then fed into the STCRNN backbone architecture, which employs a Convolutional Residual Spatial–Temporal LSTM network to extract spatial and temporal features, thereby improving feature extraction accuracy by capturing dependencies. The separated last time stamp t is integrated with these extracted features through the CAM, which computes a context vector that effectively combines the latest temporal information with the relevant features. This context vector is then projected to the dense layers for final processing. The integration of feature selection and attention enhances the STCRNN backbone by ensuring that only relevant features are processed and that the latest temporal information is effectively utilized, thereby improving the model's overall performance.

2.2.1. Temporal split layer (TSL)

The temporal split layer is a key component in the proposed architecture, designed to separate the input tensor $\mathcal{X} \in \mathbb{R}^{B \times T \times F \times |\Theta|}$ into two distinct parts: historical features $\mathcal{X}^{(th)}$ and the most recent features $\mathcal{X}^{(tr)}$. This partitioning plays a pivotal role in isolating the immediate system state, represented by the latest time step, from the historical data comprising the preceding time steps. Mathematically, the historical features, $\mathcal{X}^{(th)}$, correspond to the first $T-1$ time steps, while the recent features, $\mathcal{X}^{(tr)}$, consist of the features from the final time step. The relationship between these tensors is expressed as:

$$\mathcal{X}^{(th)} = \begin{bmatrix} \boldsymbol{x}'_{1,t} & \boldsymbol{x}'_{1,t+1} & \cdots & \boldsymbol{x}'_{1,t+T-2} \\ \boldsymbol{x}'_{2,t} & \boldsymbol{x}'_{2,t+1} & \cdots & \boldsymbol{x}'_{2,t+T-2} \\ \cdots & \cdots & \cdots & \cdots \\ \boldsymbol{x}'_{F,t} & \boldsymbol{x}'_{F,t+1} & \cdots & \boldsymbol{x}'_{F,t+T-2} \end{bmatrix}, \quad \mathcal{X}^{(tr)} = \begin{bmatrix} \boldsymbol{x}'_{1,t+T-1} \\ \boldsymbol{x}'_{2,t+T-1} \\ \cdots \\ \boldsymbol{x}'_{F,t+T-1} \end{bmatrix}, \quad \mathcal{X} = \begin{bmatrix} \mathcal{X}^{(th)} \\ \mathcal{X}^{(tr)} \end{bmatrix}. \tag{5}$$

For WSP, the latest observations $\mathcal{X}^{(tr)}$ provide an up-to-date representation of spatio-temporal atmospheric conditions, capturing immediate changes and trends critical for short-term forecasting. Meanwhile, the historical features $\mathcal{X}^{(th)}$ are processed through the DFA module and STCRNN backbone, which extract comprehensive spatial and temporal patterns from the preceding $T-1$ time steps. These historical features inform the model about underlying dependencies and long-term trends in wind behavior.

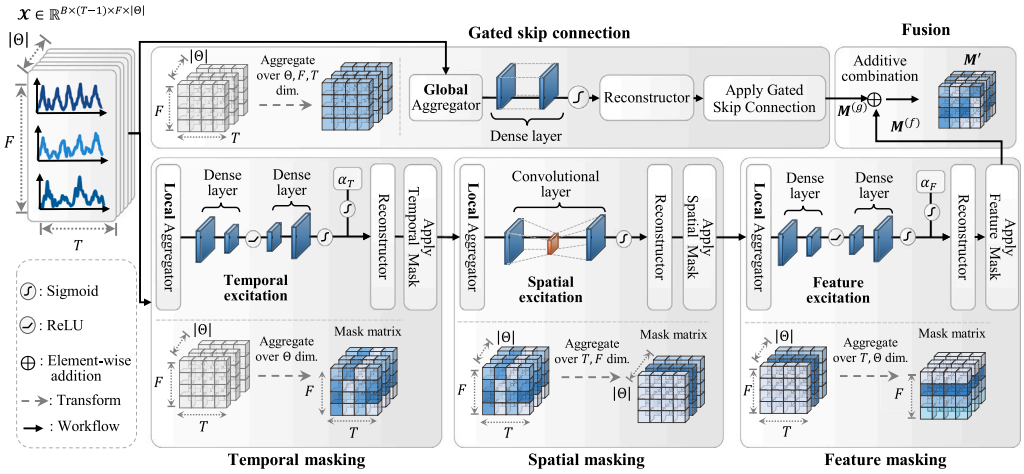


Fig. 3. The framework of the proposed dynamic feature adapter module.

2.2.2. Dynamic feature adaptor (DFA)

Effective feature selection is essential for extracting relevant information from data while minimizing the impact of noise and irrelevant features. The presence of redundant features not only increases computational complexity but also leads to overfitting and reduces the generalization capability of predictive models. Traditional methods often rely on static feature combinations, which are unable to capture the dynamic relationships inherent in the data, limiting the adaptability of the models. In the context of wind speed forecasting, such static approaches fail to address the variability of feature importance across diverse terrains and locations. To overcome these limitations, feature selection can be viewed as an optimization problem in a discrete space, where heuristic methods are typically used to identify suboptimal solutions. However, these methods are computationally expensive and require manual tuning specific to each deployment scenario. Additionally, wind speed forecasting models deployed in diverse environments must account for terrain-specific variations in feature relevance. This variability necessitates a feature selection mechanism that can dynamically adapt during training, rather than relying on predefined hyperparameters.

The proposed DFA module introduces a data-driven approach to dynamically select the significant feature. This addresses the limitations of static methods by adaptively emphasizing relevant features across temporal, spatial and feature dimensions during training. The DFA module enables the model to capture critical dependencies and relationships while maintaining flexibility across diverse terrains and data distributions. Hence, this adaptive mechanism enhances model robustness, reduces computational complexity and improves generalization capability, making it highly effective for wind speed forecasting tasks. The architecture of the DFA module, illustrated in Fig. 3, operates sequentially through three key stages: temporal masking, spatial masking and feature masking. Each stage utilizes a combination of global aggregation and excitation mechanisms to selectively highlight important information. To preserve critical input characteristics and ensure stable gradient flow, a gated skip connection is integrated into the design. This cohesive framework enhances feature representation dynamically, allowing the model to adapt effectively to varying data distributions and feature relevance during training.

Given the input tensor of historical features $\mathcal{X}^{(th)} \in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$. In the first stage, the **temporal masking** operation identifies and emphasizes relevant temporal dependencies across the input tensor. Spatial information is first aggregated through the `LocalAggregator` using global average pooling across the spatial dimensions Θ to produce a temporal aggregated tensor:

$$\boldsymbol{S}_T = \texttt{LocalAggregator}_{\Theta}\left(\mathcal{X}^{(th)}\right), \quad \boldsymbol{S}_T \in \mathbb{R}^{B\times(T-1)\times F}. \tag{6}$$

The summary $\boldsymbol{S}_T$ is then passed through a two-layer dense network to model temporal interactions.

$$\boldsymbol{E}_T = \sigma\left(\boldsymbol{W}_T^2 \cdot \texttt{ReLU}\left(\boldsymbol{W}_T^1 \cdot \boldsymbol{S}_T\right)\right) \cdot \sigma\left(\alpha_T\right), \tag{7}$$

$$\boldsymbol{M}^{(t)} = \mathcal{X}^{(th)} \odot \texttt{Reconstructor}\left(\boldsymbol{E}_T\right), \tag{8}$$

where $\boldsymbol{W}_T^1 \in \mathbb{R}^{\frac{F}{r_T}\times F}$ and $\boldsymbol{W}_T^2 \in \mathbb{R}^{F\times\frac{F}{r_T}}$ are learnable weight matrices, r_T serving as the reduction ratio for temporal features. The activation function $\texttt{ReLU}(\cdot)$ represents the Rectified Linear Unit to introduce non-linearity, while the sigmoid function σ scales the output between 0 and 1. $\texttt{Reconstructor}(\cdot)$ is used to align the data shape of $\boldsymbol{E}_T$ to $\mathcal{X}^{(th)}$. A learnable scaling factor α_T dynamically adjusts the temporal excitation $\boldsymbol{E}_T \in \mathbb{R}^{B\times T\times F}$, enabling adaptive control over the importance of temporal features. The temporal excitation is applied element-wise to the input tensor, producing the temporally masked tensor $\boldsymbol{M}^{(t)}$. Following temporal masking, **spatial masking** is applied to prioritize spatial regions of significance. Temporal and feature dimensions (T and F) are aggregated using global average pooling, producing a spatial summary tensor $\boldsymbol{S}_S$:

$$\boldsymbol{S}_S = \texttt{LocalAggregator}_{T,F}\left(\boldsymbol{M}^{(t)}\right), \quad \boldsymbol{S}_S \in \mathbb{R}^{B\times\Theta}. \tag{9}$$

Then, a convolution operation with kernel size 1×1 is applied to $\boldsymbol{S}_S$, generating a spatial excitation tensor $\boldsymbol{E}_S \in \mathbb{R}^{B\times\Theta}$. The spatial excitation is applied to the temporally masked tensor $\mathcal{X}_T$, resulting in the spatially masked tensor $\mathcal{X}_S$. The spatial masking process is formalized as:

$$\boldsymbol{E}_S = \sigma\Big(\texttt{Conv}\Big(\boldsymbol{S}_S, \boldsymbol{W}_C\Big)\Big), \qquad \boldsymbol{M}^{(s)} = \boldsymbol{M}^{(t)} \odot \texttt{Reconstructor}\big(\boldsymbol{E}_S\big), \tag{10}$$

where $\texttt{Conv}(\cdot)$ indicates the convolution operation with kernel $\boldsymbol{W}_C$. The reconstructor ensures that the excitation $\boldsymbol{E}_S$ aligns correctly with the dimensions of $\boldsymbol{M}^{(t)}$.

The **feature masking** stage targets significant feature channels by aggregating temporal and spatial dimensions (T and Θ). This yields a feature summary tensor $\boldsymbol{S}_F \in \mathbb{R}^{B\times F}$, as follows:

$$\boldsymbol{S}_F = \texttt{LocalAggregator}_{T,\Theta}\Big(\boldsymbol{M}^{(s)}\Big), \quad \boldsymbol{S}_F \in \mathbb{R}^{B\times F}. \tag{11}$$

Then, the obtained $\boldsymbol{S}_F$ is passed through a two-layer dense network, similar to the temporal masking process but parameterized by $\boldsymbol{W}_{F1}$ and $\boldsymbol{W}_{F2}$. A learnable scaling parameter α_F transforms the feature excitation $\boldsymbol{E}_F$, which is applied to the spatially masked tensor $\mathcal{X}_S$, resulting in the feature-masked tensor $\mathcal{X}_F$, as follows:

$$\boldsymbol{E}_F = \sigma\Big(\boldsymbol{W}_F^2 \cdot \texttt{ReLU}\Big(\boldsymbol{W}_F^1 \cdot \boldsymbol{S}_F\Big)\Big) \cdot \sigma\Big(\alpha_F\Big), \tag{12}$$

$$\boldsymbol{M}^{(f)} = \boldsymbol{M}^{(s)} \odot \texttt{Reconstructor}\Big(\boldsymbol{E}_F\Big), \tag{13}$$

where $\boldsymbol{W}_F^1 \in \mathbb{R}^{\frac{F}{r_F}\times F}$ and $\boldsymbol{W}_F^2 \in \mathbb{R}^{F\times\frac{F}{r_F}}$ are learnable weight matrices, with r_F representing the reduction ratio for feature channels. The excitation $\boldsymbol{E}_F \in \mathbb{R}^{B\times F}$ adjusts the spatially masked tensor $\mathcal{X}_S$ to produce the feature masked tensor $\mathcal{X}_F$.

To ensure robust feature propagation and alleviate the vanishing gradient problem, the module incorporates a **gated skip connection** that preserves critical information from the original input tensor $\mathcal{X}$. The gate $\boldsymbol{G}$ is generated by globally averaging $\mathcal{X}$ across all dimensions and scaling the result using a learnable weight matrix $\boldsymbol{W}_g$. The gated tensor $\mathcal{X}_{\text{gated}}$ is obtained by applying the gate to $\mathcal{X}$:

$$\boldsymbol{G} = \sigma\left(\boldsymbol{W}_g \cdot \frac{1}{TF\Theta}\sum_{t=1}^{T}\sum_{f=1}^{F}\sum_{\theta=1}^{\Theta}\mathcal{X}_{b,t,f,\theta,1}^{(th)}\right), \tag{14}$$

$$\boldsymbol{M}^{(g)} = \mathcal{X}^{(th)} \odot \texttt{Reconstructor}\Big(\boldsymbol{G}\Big), \tag{15}$$

where $\boldsymbol{W}_g \in \mathbb{R}^{1\times1}$ is a learnable weight matrix. The gate $\boldsymbol{G} \in \mathbb{R}^{B\times1}$ scales the original input tensor $\mathcal{X}'$ to produce the gated tensor $\boldsymbol{M}^{(g)} \in \mathbb{R}^{B\times T\times\Theta\times1\times F\times1}$.

The final output $\boldsymbol{M}'$ is obtained by combining the feature masked tensor $\boldsymbol{M}^{(f)}$ with the gated tensor $\boldsymbol{M}^{(g)}$:

$$\boldsymbol{M}' = \boldsymbol{M}^{(f)} \oplus \boldsymbol{M}^{(g)}. \tag{16}$$

This design ensures that critical features are adaptively emphasized at multiple levels, while the gated skip connection retains essential information from the input tensor. The DFA module effectively integrates spatio-temporal dependencies, enhances feature representation, and mitigates potential vanishing gradient issues, making it a robust component for complex deep learning architectures.

2.2.3. Spatio-temporal convolutional recurrent neural network (STCRNN)

The proposed method utilizes a STCRNN as the backbone architecture to effectively extract both spatial and temporal dependencies embedded in the input tensor $\boldsymbol{M}' \in \mathbb{R}^{B\times D}$, where $D = (T-1)\times F\times|\Theta|$. This architecture integrates convolutional layers for spatial feature extraction and Long Short-Term Memory (LSTM) units for modeling temporal dynamics. Residual connections are further introduced to alleviate the vanishing gradient problem and enhance feature propagation, allowing the network to model complex spatio-temporal interactions. Formally, the STCRNN backbone can be described as a transformation function $\mathcal{F}_{\text{STCRNN}}$, which processes the input tensor $\boldsymbol{M}'$ through a sequential combination of convolutional and recurrent layers. The intermediate representation is denoted by $\boldsymbol{H}_C$, and the process is expressed as:

$$\boldsymbol{H}_C = \texttt{Residual}\left(\texttt{LSTM}\big(\texttt{Conv}(\boldsymbol{M}')\big)\right). \tag{17}$$

To obtain a compact feature representation suitable for downstream tasks, $\boldsymbol{H}_C$ is flattened and passed through a fully connected (dense) layer with Exponential Linear Unit (ELU) activation. The transformation is defined as:

$$\mathcal{F}_C = \sigma_{\text{ELU}}\Big(\texttt{Flatten}(\boldsymbol{H}_C) \cdot \boldsymbol{W}_{\text{dense}} + \boldsymbol{b}_{\text{dense}}\Big), \tag{18}$$

where $\mathcal{F}_C \in \mathbb{R}^{B\times D}$ is the flattened output of the STCRNN, $\boldsymbol{W}_{\text{dense}} \in \mathbb{R}^{D\times D}$ and $\boldsymbol{b}_{\text{dense}} \in \mathbb{R}^{D}$ are the weights and biases of the dense layer. This process ensures the preservation of the learned spatio-temporal features in a unified, dense representation. The ELU activation function σ_{ELU} is defined as:

$$\sigma_{\text{ELU}}(x) = \begin{cases} x & \text{if } x > 0, \\ \alpha(e^x - 1) & \text{otherwise,} \end{cases} \tag{19}$$

where α is a hyperparameter that controls the function's behavior for negative inputs. The ELU activation introduces non-linearity, facilitating the learning of non linear relationships within the data. The dense projection layer consolidates the spatio-temporal features into a compact representation $\mathcal{F}_C \in \mathbb{R}^{B\times D}$, which is compatible for subsequent fusion with the output of the physical model. This process ensures that the learned features from the STCRNN are efficiently integrated with the physical model's predictions.

The STCRNN architecture excels at capturing spatio-temporal dependencies through the combination of convolutional layers for spatial feature extraction and LSTM units for modeling temporal dynamics. The incorporation of residual connections further enhances the network's ability to learn deep representations by alleviating the vanishing gradient problem. These features collectively contribute to a robust model that can effectively model wind speed data and other time-varying phenomena, while maintaining computational efficiency and predictive accuracy.

2.2.4. Context-aware spatio-temporal attention mechanism (CAM)

The CAM module integrates temporal context from the Temporal Split Layer with spatial–temporal features extracted by the STCRNN to enhance feature extraction for WSP. The output of the Temporal Split Layer, denoted $\mathcal{X}^{(tr)} \in \mathbb{R}^{B\times 1\times F\times|\Theta|}$, is reshaped into $\mathcal{X}'_T \in \mathbb{R}^{B\times 1\times F'}$, where $F' = F\times|\Theta|$. Simultaneously, the output of the STCRNN, $\mathcal{F}_C \in \mathbb{R}^{B\times D}$ with $D = (T-1)\times F\times|\Theta|$, is reshaped into $\mathcal{F}'_C \in \mathbb{R}^{B\times(T-1)\times F'}$. The reshaped tensors $\mathcal{X}'_T$ and $\mathcal{F}'_C$ are then concatenated along the temporal dimension to form the combined feature tensor $\mathcal{F}_P \in \mathbb{R}^{B\times T\times F'}$, which serves as the input to the attention mechanism.

This attention mechanism employs the Bahdanau attention mechanism to compute a context vector. The attention scores $\alpha_{t,s}$ are calculated as follows:

$$e_{t,s} = \boldsymbol{v}^{\top}\tanh\left(\boldsymbol{W}_1\mathcal{F}_P^{(s)} + \boldsymbol{W}_2\mathcal{X}_T^{\prime(t)}\right), \tag{20}$$

$$\alpha_{t,s} = \frac{\exp(e_{t,s})}{\sum_{s'=1}^{T}\exp(e_{t,s'})}, \tag{21}$$

$$\boldsymbol{c}_t = \sum_{s=1}^{T}\alpha_{t,s}\mathcal{F}_P^{(s)}, \tag{22}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{U\times F'}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{U\times F'}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^U$ is a weight vector. The feature vectors at temporal steps s and t are denoted by $\mathcal{F}_P^{(s)}$ and $\mathcal{X}_T^{\prime(t)}$, respectively. The context vector $\boldsymbol{c}_t \in \mathbb{R}^{F'}$ is computed for each time step t.

Then, the context vectors $\boldsymbol{c}_t$ are concatenated with the corresponding feature vectors and passed through a dense layer with U units, followed by an ELU activation function to produce the refined feature representation $\mathcal{F}_A \in \mathbb{R}^{B\times U}$:

$$\mathcal{F}_A = \sigma_{\mathrm{ELU}}(\boldsymbol{c}\cdot W_{\mathrm{attn}} + b_{\mathrm{attn}}), \tag{23}$$

where $W_{\mathrm{attn}} \in \mathbb{R}^{U\times U}$ and $b_{\mathrm{attn}} \in \mathbb{R}^U$ are weight matrices and σ_{ELU} denotes the ELU activation function. The dense layer, combined with the ELU activation, introduces non-linearity and enables the model to capture intricate interactions within the data.

The resulting refined feature vector $\mathcal{F}_A \in \mathbb{R}^{B\times U}$ is then fused with the output of the physical model to generate WSP. The attention mechanism allows the model to dynamically assign weights to different temporal and spatial components, enabling it to focus on the most relevant features for prediction. By applying Bahdanau attention scores, the model captures the intricate interactions between temporal context and spatial–temporal features. The subsequent dense layer with ELU activation further processes the features, ensuring their compatibility with the feature fusion process. This approach enhances the accuracy and efficiency of wind speed forecasting by allowing the model to focus attention on the most significant temporal and spatial components of the data, thereby improving predictive performance.

2.3. Extended advection equation-based physical modeling module (EPM)

The physical model for WSP is grounded in the extended advection equation, integrating fluid dynamics principles to simulate wind-related variables within the atmospheric domain. This model complements the neural network-based feature extraction by providing physically-informed predictions, thereby enhancing wind speed forecast accuracy and reliability. The extended advection equation governing wind velocity V is expressed as:

$$\frac{\partial V}{\partial t} + V\left(\frac{\partial V}{\partial x} + \frac{\partial V}{\partial y}\right) = -\frac{1}{\rho}\left(\frac{\partial P}{\partial x} + \frac{\partial P}{\partial y}\right) + \nu\nabla^2 V. \tag{24}$$

This equation captures: (i) the temporal change in wind velocity $\frac{\partial V}{\partial t}$, (ii) the convective acceleration $V\left(\frac{\partial V}{\partial x} + \frac{\partial V}{\partial y}\right)$, (iii) the pressure gradient forces $-\frac{1}{\rho}\left(\frac{\partial P}{\partial x} + \frac{\partial P}{\partial y}\right)$ and (iv) wind velocity diffusion due to turbulent effects $\nu\nabla^2 V$, where ρ represents air density and ν is the kinematic viscosity. To numerically solve Eq. (24), we use the fourth-order Runge–Kutta (RK4) method, which provides a robust solution for temporal integration of wind velocity. The RK4 update for wind velocity V at each time step Δt is as follows:

$$k_1 = \left.\frac{dV}{dt}\right|_{V_{\mathrm{current}}}, \tag{25}$$

$$k_2 = \left.\frac{dV}{dt}\right|_{V_{\mathrm{current}}+\frac{\Delta t}{2}k_1}, \tag{26}$$

$$k_3 = \left.\frac{dV}{dt}\right|_{V_{\text{current}}+\frac{\Delta t}{2}k_2}, \tag{27}$$

$$k_4 = \left.\frac{dV}{dt}\right|_{V_{\text{current}}+\Delta t\cdot k_3}, \tag{28}$$

$$V_{\text{future}} = V_{\text{current}} + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{29}$$

where $\frac{dV}{dt}$ is derived from the extended advection equation, V_{current} is the current wind velocity, V_{future} is the predicted wind velocity at the next time step and Δt is the time step size. The RK4 method updates the wind velocity iteratively, accurately capturing the nonlinear dynamics of wind behavior. The input to the physical model is the tensor $\mathcal{X} \in \mathbb{R}^{B\times T\times F\times|\Theta|}$, which also serves as input to the neural network. The model extracts wind speed V and pressure P to compute spatial gradients $\frac{\partial V}{\partial x}$, $\frac{\partial V}{\partial y}$, $\frac{\partial P}{\partial x}$ and $\frac{\partial P}{\partial y}$, as well as the Laplacian $\nabla^2 V$. These quantities are derived using central difference approximations and capture the spatial behavior of the wind.

The prediction process applies the RK4 method to iteratively update wind velocity, generating a sequence of predictions for the forecast horizon. The gradients and Laplacian are recalculated at each step and the RK4 update equations are used to predict future wind velocities. Final predictions are denormalized to revert preprocessing, ensuring they are in the original scale of wind speed measurements. The output of the model is a vector of predicted wind speeds for the forecast horizon, structured as $\mathcal{Y}^{(\text{pi})} \in \mathbb{R}^{B\times K}$, where K is the number of future time steps for prediction. This vector is then fused with neural network-extracted features to generate the final forecasts, combining both physics-based and data-driven insights for enhanced prediction accuracy. While the physical model is robust, it remains sensitive to input variables such as spatial gradients and pressure, making it vulnerable to data noise. Moreover, assumptions like two-dimensional flow and constant air density limit its applicability. Despite these limitations, the model offers valuable wind speed estimates that improve the neural network's learning process by grounding predictions in atmospheric dynamics. The integration of both models enhances forecast accuracy and robustness by combining the reliability of physical laws with the adaptability of machine learning.

2.4. Feature fusion module (FFM)

The Feature Fusion Module (FFM) combines neural network-extracted features with predictions from a physical model to create a comprehensive representation for Wind Speed Prediction (WSP). Specifically, the neural network outputs, denoted as $\mathcal{F}_A \in \mathbb{R}^{B\times U}$, are concatenated with the physical model predictions, $\mathcal{Y}^{(\text{pi})} \in \mathbb{R}^{B\times K}$. This results in a unified feature vector $\mathcal{F} \in \mathbb{R}^{B\times(U+K)}$, which integrates both data-driven insights and physics-based predictions. This combined feature vector $\mathcal{F}_O$ is passed through a dense layer with a Mish activation function to effectively fuse the two representations. The fusion process is mathematically expressed as:

$$\mathcal{F}_{\text{fused}} = \text{Mish}(\mathcal{F}_O \cdot W_{\text{fusion}} + b_{\text{fusion}}), \tag{30}$$

where $W_{\text{fusion}} \in \mathbb{R}^{(U+K)\times(U+K)}$ and $b_{\text{fusion}} \in \mathbb{R}^{U+K}$ are the learnable weights and biases of the fusion layer, respectively. The Mish activation function is defined as:

$$\text{Mish}(x) = x \cdot \tanh\left(\ln(1+e^x)\right). \tag{31}$$

The Mish function was selected for its smooth, non-monotonic properties, which enhance gradient flow and improve representation learning. Its ability to maintain smoothness across a wide range of input values makes it particularly effective for fusing diverse features, ensuring minimal distortion while preserving their interactions. After fusion, the joint representation $\mathcal{F}_{\text{fused}}$ is passed to the final prediction layer using another dense layer with K units and a linear activation function:

$$\mathcal{Y}^{(\text{nn})} = \mathcal{F}_{\text{fused}} \cdot W_{\text{final}} + b_{\text{final}}, \tag{32}$$

where $W_{\text{final}} \in \mathbb{R}^{(U+K)\times K}$ and $b_{\text{final}} \in \mathbb{R}^K$ are the weights and biases of the final layer. The linear activation function ensures that the output predictions $\mathcal{Y}^{(\text{nn})}$ retain their continuous nature, crucial for regression tasks such as WSP. The Feature Fusion Module integrates data-driven and physics-based insights, providing a robust estimate of wind speed. By combining the strengths of both neural network features and physical model predictions, this approach enhances prediction accuracy. It leverages the complementary benefits of both methods, improving the model's learning ability and enabling more accurate and reliable wind speed forecasts.

2.5. Adaptive physics penalty loss (APPL)

The APPL optimizes WSP models by integrating data-driven and physics-based components. In contrast to traditional PINN loss functions, which uniformly apply penalties through Lagrange multipliers to enforce physical constraints, APPL conditionally enforces these constraints based on the magnitude of prediction deviations. Specifically, APPL establishes percentile-based thresholds ϵ_t for each prediction horizon t using the entire training dataset. These thresholds are derived from the distribution of absolute errors between the ground truth wind speeds $\mathcal{Y}$ and the physics-based predictions $\mathcal{Y}^{(\text{PI})}$, ensuring that only significant deviations contribute to the loss function.

Formally, APPL is defined as:

$$\mathcal{L}_{\text{APPL}} = \frac{1}{NK}\sum_{i=1}^{N}\sum_{t=1}^{K}\underbrace{\left(\mathcal{Y}^{(\text{nn})}(i,t) - \mathcal{Y}(i,t)\right)^2}_{\text{Data Loss}} + \underbrace{\mathbb{I}\left(\left|\mathcal{Y}^{(\text{nn})}(i,t) - \mathcal{Y}^{(\text{pi})}(i,t)\right| > \epsilon_t\right)\cdot\left(\mathcal{Y}^{(\text{nn})}(i,t) - \mathcal{Y}^{(\text{pi})}(i,t)\right)^2}_{\text{Physics Penalty}}, \tag{33}$$

where $\mathcal{L}_{\text{APPL}}$ represents the total loss, N is the number of training samples and K denotes the number of prediction horizons. $\mathcal{Y}$ corresponds to the ground truth wind speed, $\mathcal{Y}^{(\text{nn})}$ to the neural network's prediction and $\mathcal{Y}^{(\text{pi})}$ to the physics-based prediction. The indicator function $\mathbb{I}(\cdot)$ activates the physics-based penalty only when the absolute difference between the neural network's prediction and the physics-based prediction exceeds the threshold ϵ_t.

The threshold ϵ_t for each horizon t is computed as the Pth percentile of the absolute errors $|\mathcal{Y}_{\text{train}}(i,t) - \mathcal{Y}^{(\text{pi})}_{\text{train}}(i,t)|$ across all training samples:

$$\epsilon_t = \text{Percentile}_P\left(\left|\mathcal{Y}_{\text{train}}(i,t) - \mathcal{Y}^{(\text{pi})}_{\text{train}}(i,t)\right|\right). \tag{34}$$

This formulation ensures that ϵ_t represents the error distribution for each prediction horizon by utilizing the full training dataset. APPL applies physics-based penalties when deviations between the neural network's predictions and the physics-based model exceed statistically significant thresholds. This selective penalty prevents overfitting from the physical model error, focusing the model on addressing inaccuracies. By incorporating physics-based constraints in a data-dependent manner, APPL aligns the neural network's predictions with the physics model where significant deviations occur, improving the accuracy and reliability of wind speed forecasts. This approach balances data-driven optimization with physical laws, yielding a robust and generalizable predictive model.

The pseudocode of the overall workflow of the proposed scheme is provided in Algorithm 1.

Algorithm 1 Wind Speed Prediction Model Workflow

Require: **Data:** D
Hyperparameters: Batch size B, Window size T, Step size S, Prediction horizon K, Learning rate η, Epochs E, Percentile P
Ensure: Trained model parameters and evaluation metrics saved in `metrics.json`
Data Preprocessing
Normalize all features in D
Segment data into overlapping windows $\mathcal{W}$
Structure input tensor $\mathcal{X} \in \mathbb{R}^{B\times T\times F\times|\Theta|}$
Compute Epsilon Thresholds
Compute absolute errors on training set: errors $\leftarrow |\mathcal{Y}_{\text{train}} - \mathcal{Y}^{(\text{pi})}_{\text{train}}|$
Determine thresholds: $\epsilon_t \leftarrow \text{Percentile}_P(\text{errors}_t)$ for each horizon $t \in \{1, \ldots, K\}$
Initialize Model Parameters
Initialize θ_{DFA}, θ_{STCRNN}, θ_{CAM}, θ_{FFM}
for epoch $e = 1$ to E **do**
  for each batch b in $\mathcal{W}$ **do**
    Forward Pass
    $\boldsymbol{Y}_{\text{final}} \leftarrow \text{ForwardPass}(\mathcal{X}_b, \theta)$
    Compute APPL Loss
    $\mathcal{L}_{\text{APPL}} \leftarrow \text{ComputeAPPL}(\mathcal{Y}^{(\text{nn})}, \mathcal{Y}, \mathcal{Y}^{(\text{pi})}, \epsilon_t)$
    Backward Pass and Optimization
    Update parameters: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}_{\text{APPL}}$
  end for
end for
Testing Phase
$\{\text{RMSE}, \text{MAE}, \frac{1}{R^2}, \text{SMAPE}\} \leftarrow \text{EvaluateMetrics}(D_{\text{test}}, \theta)$

3. Experimental results

This section presents the performance results of the proposed model and includes ablation studies conducted on each of the four datasets collected from Hamburg, Herning, Palmerston North and Silkeborg. The ablation studies assess the impact of the DFA module, CAM, EPM, FFM and the APPL loss function on model's performance. Each subsection provides detailed insights into the model's performance for the respective dataset, highlighting the contributions of individual components and their influence on the overall results. Finally, the performance of the model is compared with six recently proposed state-of-the-art methods.

3.1. Dataset description

This study utilizes four datasets from NASA's Prediction of Worldwide Energy Resources (POWER) dataset [37], with all features measured at a 1-hour resolution. The data was collected from four cities in three countries, including Hamburg and Silkeborg in Denmark, Herning in Denmark and Palmerston North in New Zealand. Table 1 outlines the data collection periods, total durations and data splits. Each dataset is partitioned chronologically into 80% for training, 10% for validation and 10% for testing to ensure comprehensive model training and unbiased evaluation.

Measurements were obtained from a central point and eight surrounding locations in the cardinal directions, at distances of 10 km (Hamburg), 50 km (Herning), 100 km (Palmerston North) and 30 km (Silkeborg). The features include temperature (2 m), wind speed and direction (10 m), air pressure (surface) and humidity (2 m). Wind speed at 10 meters corresponds to typical wind turbine installation heights. Table 2 summarizes wind speed statistics, indicating mean values ranging from 4.78 m/s (Hamburg) to 6.94 m/s (Silkeborg), with standard deviations between 2.39 m/s and 3.30 m/s. The varying distances between measurement points facilitate analysis under different spatial configurations. Finally, Seasonal variations are presented in Table 3, illustrating differences

Table 1
Data collection periods and data splits.

Location	Period	Total Hours	Train (80%)	Validation (10%)	Test (10%)
Hamburg	Jan 2019–Dec 2022	35,064	28,051	3,506	3,507
Herning	Jan 2020–Jul 2023	30,936	24,749	3,094	3,093
Palmerston North	Jan 2020–Jul 2023	30,936	24,749	3,094	3,093
Silkeborg	Jan 2020–Dec 2023	35,064	28,051	3,506	3,507

Table 2
Wind speed statistics and spatial distances.

Parameter	Hamburg	Herning	Palmerston North	Silkeborg
Mean (m/s)	4.78	5.57	5.32	6.94
Median (m/s)	4.37	5.25	4.81	6.66
Std Dev (m/s)	2.39	2.71	2.83	3.30
Min (m/s)	0.02	0.07	0.02	0.07
Max (m/s)	20.97	20.88	16.84	26.81
Distance (KM)	10	50	100	30

Table 3
Seasonal average wind speeds (m/s).

Season	Hamburg	Herning	Palmerston North	Silkeborg
Spring	4.96	5.44	4.84	6.53
Summer	3.97	4.71	5.31	6.14
Autumn	4.61	5.71	6.01	7.36
Winter	5.60	6.40	5.28	7.74

in average wind speeds across spring, summer, autumn and winter for each location. For example, Silkeborg shows an increase from 6.14 m/s in summer to 7.74 m/s in winter, reflecting seasonal influences on wind patterns.

The datasets exhibit temporal variability through fluctuating wind speeds and seasonal trends, necessitating models that can capture both short-term and long-term patterns. Spatially, the datasets span diverse geographical regions with varying distances between measurement points, providing a comprehensive basis for evaluating model performance under different spatial configurations. The combination of temporal fluctuations, distinct spatial behaviors and geographical diversity renders the datasets suitable for assessing the proposed model's performance relative to existing models, ensuring robustness and generalizability in WSP across varied conditions.

3.2. Experimental setup

The experimental framework is established to assess the performance of the proposed WSP model in comparison with existing models under standard conditions. All models undergo training for up to 2000 epochs, employing an early stopping mechanism with a patience of 10 epochs to prevent overfitting. The Adam optimizer is utilized with a learning rate of 0.001 and a gradient clipping value of 1.0 to ensure stable convergence. For loss functions, MSE is applied to the baseline models, whereas the proposed model exclusively employs the APPL loss function, as detailed in proposed methodology. The prediction task involves using input features from 48 h (denoted as T) across nine spatial dimensions (theta = 9), with each dimension providing five features. The model predicts wind speed for the next six hours ($K = 6$) with a step size of 4 h ($S = 4$) between consecutive windows. A batch size of 32 is maintained throughout the training process. Training is conducted on an NVIDIA GeForce RTX 3080 Ti GPU and model inference is performed on NVIDIA Jetson Nano devices to verify deployability on edge hardware. All results presented in this paper are conducted on NVIDIA Jetson Nano using the test subset of each dataset. Table 4 summarizes the key experimental parameters.

3.3. Evaluation metrics

The performance of the WSP models is evaluated using four metrics including RMSE, MAE, Reciprocal of the' Coefficient of Determination ($1/R^2$) and Symmetric Mean Absolute Percentage Error (SMAPE). These metrics provide a comprehensive assessment of the models' predictive accuracy and robustness. RMSE measures the square root of the average squared differences between the predicted wind speeds ($\hat{y}_i$) and the actual wind speeds (y_i). It is sensitive to large errors, making it useful for identifying models that may produce significant outliers:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}. \tag{35}$$

Table 4
Summary of experimental parameters.

Parameter	Value
Number of Epochs	2000
Early Stopping Patience	10 epochs
Optimizer	Adam
Learning Rate	0.001
Gradient Clipping Value	1.0
Batch Size	32
Loss Function (SOTA Models)	MSE
Loss Function (Proposed Model)	APPL
Input Time Stamp (T)	48 h
Spatial Dimension (Θ)	9
Features per Dimension (F)	5
Step Size between Windows (S)	4 h
Percentile(P)	80%
Prediction Horizon (K)	6 h
Training Hardware	NVIDIA RTX 3080 Ti
Deployment Hardware	NVIDIA Jetson Nano

MAE calculates the average of the absolute differences between the predicted and actual values, providing a straightforward interpretation of the average prediction error in the same units as wind speed. It is less sensitive to outliers compared to RMSE, offering a more robust measure of central tendency of errors:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|. \tag{36}$$

The reciprocal of the R^2 score ($1/R^2$) offers an alternative perspective on the model's explanatory power. While the R^2 score indicates the proportion of variance in the actual wind speeds explained by the model, its reciprocal inversely relates to the model's performance, where lower values correspond to better predictive capability:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}, \tag{37}$$

$$\frac{1}{R^2} = \frac{1}{1 - \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}}. \tag{38}$$

SMAPE evaluates the accuracy of the forecasts by measuring the relative error between the predicted and actual values, scaled by the average of the absolute actual and predicted values. This metric is particularly useful for understanding percentage errors in predictions, allowing for comparison across different scales of wind speed:

$$\mathrm{SMAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{\frac{|\hat{y}_i| + |y_i|}{2}}, \tag{39}$$

where N is the number of observations, y_i represents the actual wind speed at instance i, $\hat{y}_i$ represents the predicted wind speed at instance i and $\overline{y}$ is the mean of the actual wind speeds. A consistent decrease in RMSE, MAE, $1/R^2$ and SMAPE indicates an improvement in model performance. By taking the reciprocal of R^2, this study ensure that a lower value corresponds to better performance, maintaining consistency with the other metrics where lower values denote improvement. This suite of metrics collectively evaluates the models from different perspectives, ensuring a thorough and nuanced assessment of their capabilities in accurately prediction wind speed.

3.4. Results and ablation study

This section evaluates the performance of the proposed model and presents an ablation study. The scatter plots in Figs. 4, 5, 6, and 7 illustrate the strong agreement between predicted and actual wind speeds in all data sets.

Horizon 1, predictions closely cluster along the identity line, demonstrating high short-term accuracy. Although longer prediction horizons introduce a natural increase in dispersion, the model consistently captures the overall trends, affirming its robustness in practical forecasting scenarios. A detailed analysis of the MAE and MSE against data density confirms the model's robust performance across a wide range of wind conditions. In particular, although the Silkeborg dataset includes extreme wind speeds (with a maximum of 26.81 m/s), such cases are exceptionally rare due to limited data availability. Our findings show that while prediction errors in these high wind speed ranges are slightly elevated, their infrequency means they have a negligible impact on the overall performance. Since the primary focus of this work is to accurately capture average wind behavior, we emphasize that the model remains highly effective for the vast majority of conditions, and the limited occurrence of extreme events does not warrant a dedicated analysis in this context.

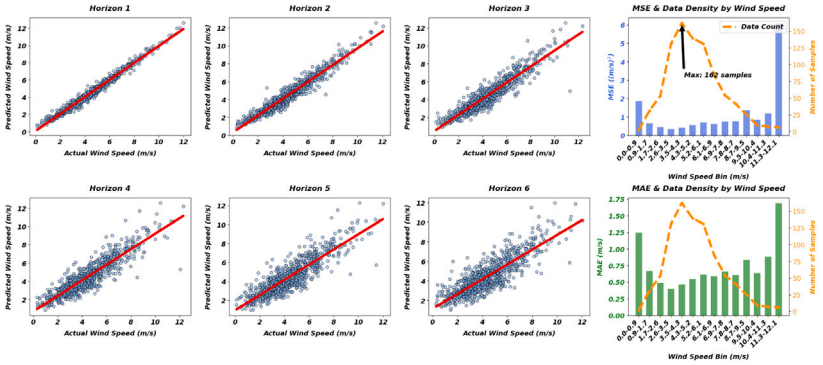


Fig. 4. Scatter plot of predicted vs. actual wind speeds for the Hamburg dataset.

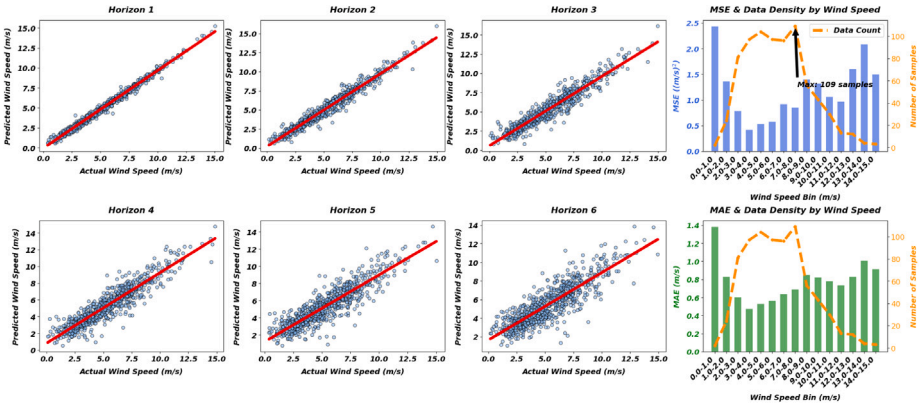


Fig. 5. Scatter plot of predicted vs. actual wind speeds for the Herning dataset.

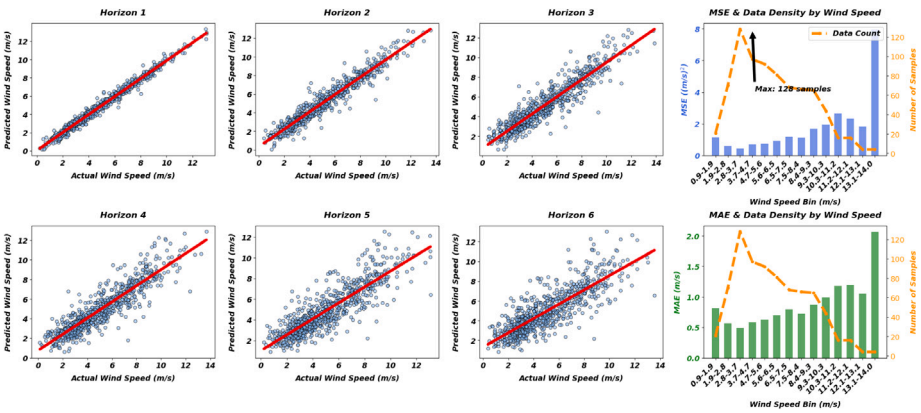


Fig. 6. Scatter plot of predicted vs. actual wind speeds for the Palmerston North dataset.

For the ablation study, we incrementally add components to the backbone architecture, STCRNN, which forms the basis of the initial model, PISTNet V1. The study begins with the addition of the DFA module, resulting in the model PISTNet V2. Next, the CAM module is incorporated, producing the model PISTNet V3. Finally, the FFM layer, which combines the predictions from the EPM and the features extracted from the CAM module, is added, resulting in PISTNet V4. The final model, PISTNet, includes the APPL loss function. We compare the performance of each of these models in this study. The Table 5 summarizes each model version progressively builds on the previous one, with the final model incorporating all the discussed components.

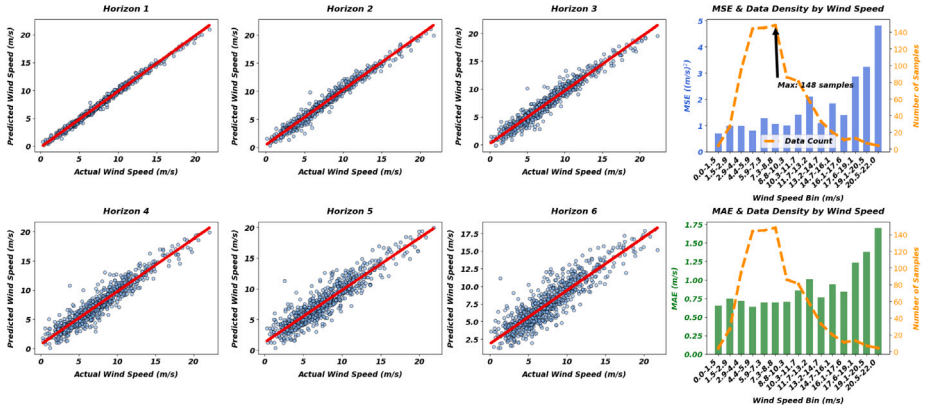


Fig. 7. Scatter plot of predicted vs. actual wind speeds for the Silkeborg dataset.

Table 5
Overview of model versions and added components.

Model Version	STCRNN	DFA	CAM	FFM	APPL
PISTNet V1	✓	–	–	–	–
PISTNet V2	✓	✓	–	–	–
PISTNet V3	✓	✓	✓	–	–
PISTNet V4	✓	✓	✓	✓	–
PISTNet	✓	✓	✓	✓	✓

Table 6
Average performance metrics across six prediction horizons on the Hamburg dataset.

Model	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
PISTNet V1	0.7893	0.6066	1.2277	8.3385
PISTNet V2	0.7895	0.6014	1.2426	8.1932
PISTNet V3	0.7923	0.6030	1.23	8.1711
PISTNet V4	0.7771	0.5914	1.213	8.007
PISTNet	**0.7334**	**0.5540**	**1.1915**	**7.6024**

3.4.1. Ablation study on Hamburg dataset

Table 6 presents the average performance metrics of the baseline PISTNet V1 model and its successive enhancements on the Hamburg dataset, evaluated across six prediction horizons. Moving from PISTNet V1 to PISTNet V2 results in a marginal increase in RMSE from 0.7893 m/s to 0.7895 m/s and an improvement in the $1/R^2$ Score from 1.2277 to 1.2426. Meanwhile, the MAE decreases slightly from 0.6066 m/s to 0.6014 m/s and the SMAPE drops from 8.3385% to 8.1932%. The addition of the CAM module in PISTNet V3 further raises the RMSE to 0.7923 m/s and the $1/R^2$ Score to 1.23, while MAE increases slightly to 0.6030 m/s. However, the SMAPE continues to improve, decreasing to 8.1711%. This increase in RMSE despite the decrease in MAE suggests that while the model's predictions may exhibit slightly higher variability, it is becoming better at predicting the general trend of wind speed, especially for smaller values, which results in a reduction in the average absolute error (MAE). The introduction of the FFM layer in PISTNet V4 leads to more significant improvements, reducing the RMSE to 0.7771 m/s, MAE to 0.5914 m/s, $1/R^2$ Score to 1.213 and SMAPE to 8.007%. Finally, the application of the APPL loss function in the final PISTNet model results in the most significant performance gains, achieving the lowest RMSE of 0.7334 m/s, MAE of 0.5540 m/s, $1/R^2$ Score of 1.1915 and SMAPE of 7.6024%. The progressive improvements in these metrics highlight that each module contributes to the model's overall performance, with the integration of EPM predictions and the APPL loss function yielding the most pronounced benefits.

Overall, the progressive enhancements to the PISTNet V1 architecture consistently improve the model's ability to capture complex spatial–temporal dependencies and predictive patterns, resulting in superior performance metrics on the Hamburg dataset. Each successive modification builds upon the previous one, incrementally refining the model's predictive capabilities. Specifically, Fig. 8 provides a horizon-by-horizon comparison of RMSE, MAE, $1/R^2$ Score and SMAPE, illustrating the performance gains achieved through successive model improvements across different prediction horizons. The black line in the figure represents the overall improvement from the backbone PISTNet V1 model to the proposed PISTNet model. In terms of RMSE, the most significant improvement of 11.4% occurs at the first prediction horizon, followed by a decreasing rate of improvement, with the 6th-hour prediction showing a 5.6% improvement compared to PISTNet V1. Similar patterns are observed in MAE and SMAPE, indicating that the benefits of the proposed model enhancements diminish as the prediction horizon increases, except for an outlier at the third

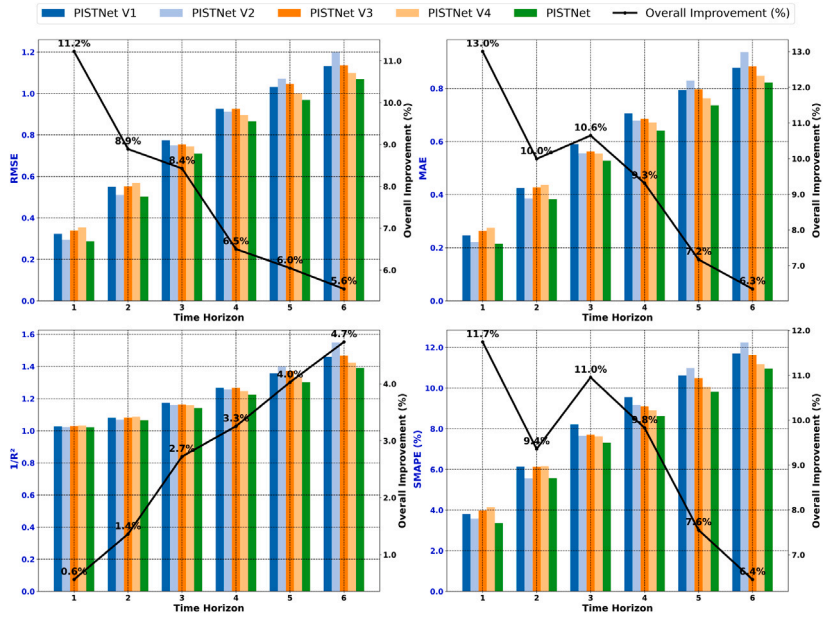


Fig. 8. Percentage improvement across prediction horizons (1 to 6) for each model variant on the Hamburg dataset.

Table 7
Average Performance metrics across six prediction horizons on the Herning dataset.

Model	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
PISTNet V1	0.9154	0.6961	1.1744	8.1604
PISTNet V2	0.9061	0.6988	1.1741	8.3164
PISTNet V3	0.8756	0.6616	1.1648	8.1577
PISTNet V4	0.8606	0.6554	1.1590	7.7884
PISTNet	**0.8511**	**0.6443**	**1.1535**	**7.7088**

horizon, where the improvement surpasses that of the second horizon. Conversely, the $1/R^2$ Score generally improves with longer horizons, suggesting that while the physics-informed enhancements offer greater improvements at shorter prediction horizons, they also significantly aid in better fitting the model for longer time horizons. In summary, the proposed model improvements demonstrate a robust ability to enhance prediction accuracy, particularly in longer-term predictions, by integrating STCRNN, DFA, CAM, FFM and APPL loss thereby creating a more accurate and reliable prediction system for WSP on Hamburg dataset.

3.4.2. Ablation study on Herning dataset

Table 7 presents the average performance metrics of the PISTNet V1 model and its subsequent enhancements on the Herning dataset across six prediction horizons. The transition from PISTNet V1 to PISTNet V2, achieved by incorporating the DFA block, results in a slight improvement in the RMSE, which decreases from 0.9154 to 0.9061 m/s. However, this enhancement leads to a marginal increase in both the Mean Absolute Error (MAE) and the SMAPE, suggesting a small rise in sensitivity to larger errors while having minimal impact on the mean error. The addition of the CAM layer to form PISTNet V3 brings more notable improvements, reducing the RMSE to 0.8756 m/s, the MAE to 0.6616 m/s and improving the SMAPE to 8.15%. This improvement indicates that the CAM layer effectively highlights relevant spatial–temporal features, thereby enhancing prediction accuracy. The subsequent integration of the FFM layer to create PISTNet V4 further reduces the RMSE and MAE to 0.8606 m/s and 0.6554 m/s, respectively, while lowering the SMAPE to 7.79%. These results demonstrate that incorporating predictive information from physical models significantly improves performance by providing more accurate and contextually relevant data. Finally, the addition of the APPL loss function in the training process of PISTNet, along with the improvements from PISTNet V4, leads to the best performance, achieving an RMSE of 0.8511 m/s, an MAE of 0.6443 m/s and a SMAPE of 7.71%. This highlights the cumulative benefits of each added module in enhancing the model's prediction accuracy.

Overall, the progressive enhancements to the PISTNet V1 architecture consistently improve the model's ability to capture complex spatial–temporal dependencies and predictive patterns, resulting in superior performance metrics on the Herning dataset. Each successive modification builds upon the previous one, incrementally refining the model's predictive capabilities. Fig. 9 additionally, provides a horizon-by-horizon comparison of RMSE, MAE, $1/R^2$ Score and SMAPE, illustrating the performance gains achieved through successive model improvements across different prediction horizons. The black line in the figure represents the overall

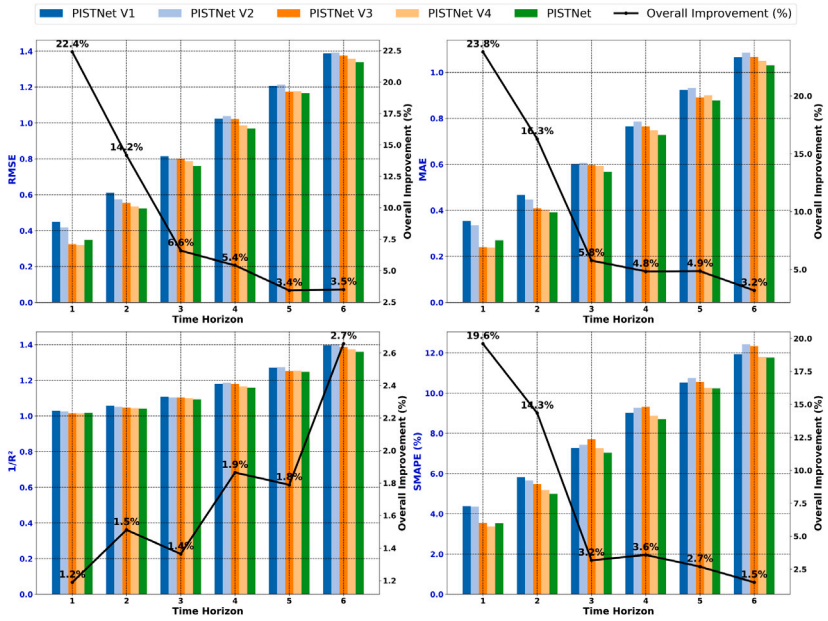


Fig. 9. Percentage improvement across prediction horizons (1 to 6) for each model variant on the Herning dataset.

Table 8
Average performance metrics across six prediction horizons on the Palmerston North dataset.

Model	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
PISTNet V1	1.0709	0.8010	1.2640	9.5402
PISTNet V2	1.0090	0.7462	1.2273	9.3199
PISTNet V3	1.0102	0.7405	1.2224	8.8277
PISTNet V4	0.9869	0.7332	1.2225	8.7589
PISTNet	**0.9534**	**0.7055**	**1.2021**	**8.5484**

improvement from the PISTNet V1 model to the proposed PISTNet. Specifically, in terms of RMSE, the most significant improvement of 22.4% occurs at the first prediction horizon, followed by a decreasing rate of improvement, with the 6th-hour prediction showing a 3.5% improvement compared to STCRNN. Similar patterns are observed in MAE and SMAPE, indicating that the benefits of the proposed model enhancements diminish as the prediction horizon increases. Conversely, the $1/R^2$ Score generally improves with longer horizons, except for an outlier at the third time horizon. This suggests that while the FFM and APPL shows more improvements for shorter prediction horizons, they significantly aid in better fitting the model for longer time horizons.

3.4.3. Ablation study on Palmerston North dataset

Table 8 presents the performance of the PISTNet V1 model and its successive enhancements on the Palmerston North dataset. Incorporating the DFA block leads to a reduction in RMSE from 1.0709 to 1.0090 m/s and a decrease in MAE from 0.8010 to 0.7462 m/s.

The $1/R^2$ score slightly declines from 1.2640 to 1.2273, while SMAPE improves from 9.5402% to 9.3199%. The addition of the CAM mechanism in PISTNet V3 results in a slight increase in RMSE to 1.0102 m/s, but it further reduces MAE to 0.7405 m/s. The $1/R^2$ score decreases slightly to 1.2224, while SMAPE improves to 8.8277%. Introducing the FFM layer in PISTNet V4 leads to additional gains, lowering RMSE to 0.9869 m/s and MAE to 0.7332 m/s. The $1/R^2$ score stabilizes at 1.2225 and SMAPE improves further to 8.7589%. Finally, the integration of the APPL loss function in PISTNet results in the best performance, reducing RMSE to 0.9534 m/s, MAE to 0.7055 m/s, $1/R^2$ to 1.2021 and SMAPE to 8.5484%. Fig. 10 shows the percentage improvement in RMSE, MAE and SMAPE across the six horizons for each model. The highest improvement in RMSE is 31.3% at the first horizon, which decreases at longer horizons. RMSE increases at the fourth and sixth horizons. SMAPE and MAE show the greatest improvement at shorter horizons, with smaller gains at longer horizons. The $1/R^2$ score follows the trends observed in the Hamburg and Herning datasets, with the highest improvement (9.9%) at the sixth horizon.

3.4.4. Ablation study on Silkeborg dataset

Table 9 shows the average performance metrics of the backbone PISTNet V1 model and its successive enhancements on the Silkeborg dataset across six prediction horizons. Introducing the DFA block to make PISTNet V2 reduces the RMSE from 1.0646 to

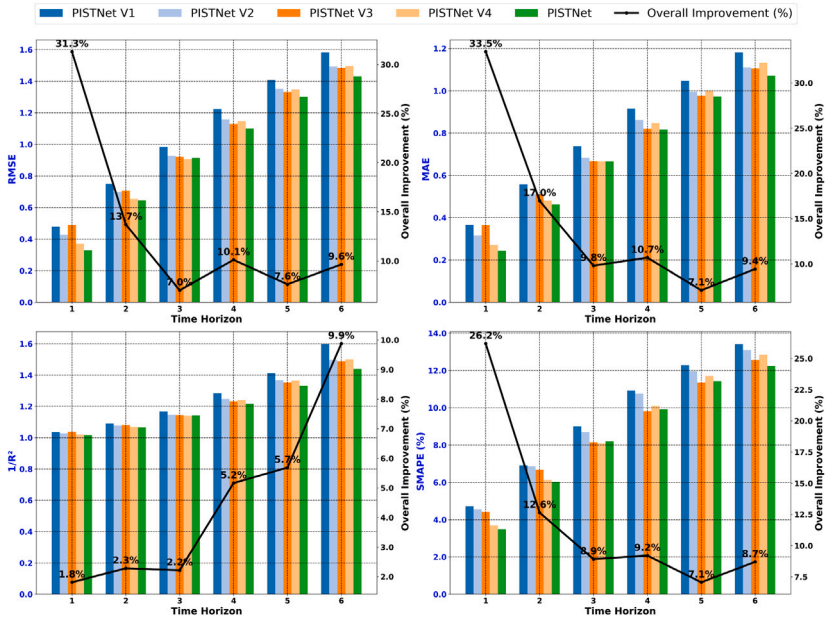


Fig. 10. Percentage improvement across prediction horizons (1 to 6) for each model variant on the Palmerston North dataset.

Table 9
Average performance metrics across six prediction horizons on the Silkeborg dataset.

Model	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
PISTNet V1	1.0646	0.7990	1.1102	6.9331
PISTNet V2	1.0400	0.7790	1.1070	6.8919
PISTNet V3	1.0510	0.7805	1.1085	6.8470
PISTNet V4	1.0374	0.7695	1.1047	6.6870
PISTNet	**1.0155**	**0.7563**	**1.0997**	**6.6344**

1.0400 m/s and the MAE from 0.7990 to 0.7790 m/s. The $1/R^2$ Score decreases from 1.1102 to 1.1070 and the SMAPE decreases from 6.9331% to 6.8919%. Furthermore, the addition of the CAM results in an RMSE of 1.0510 m/s, a MAE of 0.7805 m/s, a $1/R^2$ Score of 1.1085 and a SMAPE of 6.8470% in PISTNet V3. Further integration of FFM in existing model results in PISTNet V4 that lowers the RMSE to 1.0374 m/s and the MAE to 0.7695 m/s. The $1/R^2$ Score changes to 1.1047 and the SMAPE decreases to 6.6870%. Finally, the proposed model PISTNet with integration of APPL function results in an RMSE of 1.0155 m/s, an MAE of 0.7563 m/s, a $1/R^2$ Score of 1.0997 and a SMAPE of 6.6344%. Each model variant exhibits changes in performance metrics, with the addition of FFM and the APPL loss function contributing to the most notable improvements. Fig. 11 presents the percentage improvement in RMSE, MAE and SMAPE across six prediction horizons for each model variant. The RMSE shows an improvement of 7.9% at the first horizon, followed by a 0.2% improvement at the second horizon and a 7.0% improvement at the third horizon. Subsequent horizons exhibit a 5.4% decrease and a 1.9% improvement at the sixth horizon. The MAE follows a similar pattern. The SMAPE shows the highest improvement of 5.6% at the fourth horizon and 3.8% at the first horizon. The $1/R^2$ Score improvement percentage varies, decreasing from the first to the second horizon, increasing until the fifth horizon and then decreasing again. These patterns are different from the pattern observed in the previous datasets. The primary reason for this is the high variability (SD = 3.30 m/s) and extreme outliers (max = 26.81 m/s), as given in the Table 2.

3.5. Comparison with state-of-the-art (SOTA) methods

This section compares the performance of the proposed model with eight SOTA models, each employing different architectures and techniques for WSP. The models include Conv-LSTM [38], CoReSTL [11], Mixformer [39], PI-LSTM [34], SAGWO-LSTM [26], SG-Filtered TCN-LSTM [40], Inverted Transformer [41], and PatchTST[42]. Among these, Conv-LSTM and CoReSTL [11,38] are based on convolutional LSTM architectures, with CoReSTL specifically designed for WSP using spatio-temporal data. Mixformer [39] is a transformer-based model that incorporates a spatio-temporal Gaussian mixture attention (ST-GMA) layer to capture intricate temporal and spatial dependencies for more accurate predictions. PI-LSTM integrates physical knowledge into an LSTM framework for improved WSP. The SAGWO-LSTM model [26] combines feature selection through correlation analysis of wind turbine data and applies an enhanced grey wolf optimization algorithm (SAGWO) to optimize the LSTM, making it particularly relevant for

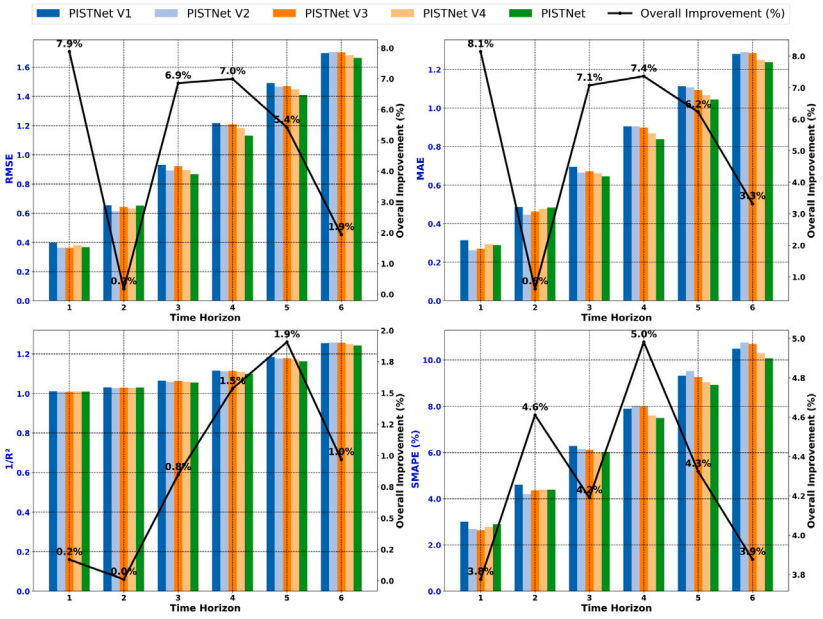


Fig. 11. Percentage improvement across prediction horizons (1 to 6) for each model variant on the Silkeborg dataset.

Table 10
Average performance metrics on the Hamburg dataset.

Model name	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE ↓
Conv-LSTM [38]	1.4894	1.1846	2.2917	14.9112
CoReSTL[11]	0.7894	0.6066	1.2277	8.3385
PI-LSTM[34]	0.9105	0.7063	1.3554	9.4530
Mixformer [39]	0.8003	0.6169	1.2320	8.5559
SAGWO-LSTM[26]	0.8240	0.6385	1.2490	8.7803
iTransformer [41]	1.2343	0.9353	1.7022	11.9558
PatchTST [42]	1.0815	0.8071	1.4558	10.6308
TCN-LSTM with Savitzky–Golay filter [40]	0.9379	0.7386	1.3324	9.7010
PISTNet	**0.7334**	**0.5540**	**1.1915**	**7.6025**

feature engineering and optimization using heuristic methods. The SG-Filtered TCN-LSTM [40] is a hybrid model that merges a temporal convolution network (TCN) with an LSTM network and uses the Savitzky–Golay filter for feature selection on the outputs of both models. The Inverted Transformer [42] leverages the self-attention mechanism to capture long-range dependencies in sequential data, making it highly effective for time series forecasting tasks such as WSP. PatchTST [42]adapts the Vision Transformer (ViT) approach to handle multivariate time series by dividing the data into patches, allowing it to capture both local and global dependencies over long forecasting horizons. These eight models span a wide range of techniques, including convolutional LSTM, transformer-based models, physics-informed LSTM, heuristic optimization, and hybrid models with feature selection. Comparing the proposed model against these diverse architectures, almost all of which have been introduced in the past two years, demonstrates its superior performance across various evaluation metrics.

3.5.1. Comparison on Hamburg dataset

The comparison of the proposed model with other SOTA models on the Hamburg dataset is presented in Table 10. The table shows the average performance metrics for each model across multiple evaluation criteria, including RMSE, MAE, $1/R^2$ score and SMAPE.

From Table 10, the results clearly demonstrate that the PISTNet model outperforms all other models across all four evaluation metrics, RMSE, MAE, $1/R^2$ score and SMAPE. Specifically, it achieves the lowest RMSE of **0.7334** m/s, the lowest MAE of **0.5540** m/s, the lowest $1/R^2$ score of **1.1915** and the lowest SMAPE of **7.6025**%, establishing a new benchmark for WSP using the Hamburg spatio-temporal dataset. In comparison, the next best-performing model is CoReSTL, with an RMSE of 0.7894 m/s and an MAE of 0.6066 m/s, yet it falls short of the proposed model in all metrics. Inverted Transformer and PatchTST are both designed for capturing long-range dependencies across time and space. However, Inverted Transformer struggles in short-term wind speed prediction because its self-attention mechanism, while effective for global patterns, does not capture local spatial dependencies and

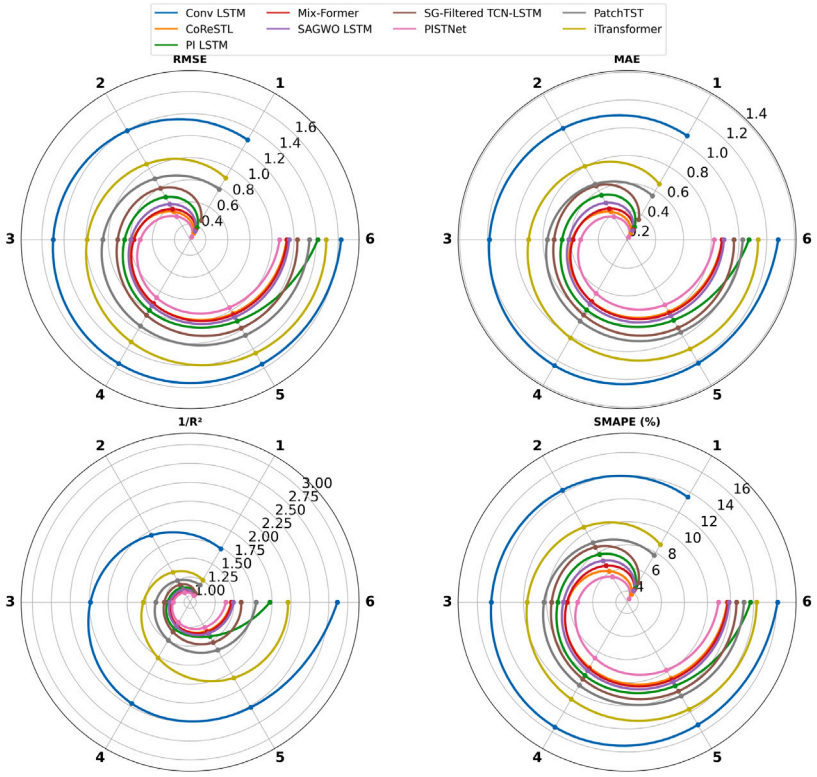


Fig. 12. Performance comparison across different time horizons for each metric on the Hamburg dataset.

Table 11
Average performance metrics on the Herning dataset.

Model name	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE ↓
Conv-LSTM [38]	1.5314	1.2048	1.5774	13.7232
CoReSTL[11]	0.9154	0.6961	1.1744	8.1604
PI-LSTM[34]	0.9656	0.7445	1.1867	8.5598
Mixformer [39]	0.9557	0.7346	1.1961	8.6692
SAGWO-LSTM[26]	1.0172	0.7713	1.2181	8.7800
iTransformer [41]	1.2343	0.9353	1.7022	11.9558
PatchTST [42]	1.4933	1.1331	1.6101	12.0709
TCN-LSTM with Savitzky–Golay filter [40]	0.9489	0.7282	1.1938	8.6064
PISTNet	**0.8511**	**0.6443**	**1.1535**	**7.7088**

immediate temporal fluctuations, which are crucial for accurate 6-hour forecasts. On the other hand, PatchTST, designed to handle local spatial features via its patch-based approach, performs better in this case because it preserves both local spatial relationships and fine-grained temporal patterns, making it more suitable for short-term predictions. However, PatchTST and models such as Mixformer and PI-LSTM also exhibit competitive results but do not surpass the PISTNet model in terms of RMSE and MAE. Table 10 presents the average results of the proposed model, whereas the time horizon comparison for the Hamburg dataset is illustrated in Fig. 12. In the radial charts, the inner-most circle, highlighted in pink, distinctly showcases the superior performance of the proposed model relative to existing models at each time horizon. Conversely, the Conv-LSTM model exhibits the poorest performance across all time horizons, as indicated by its outermost circle in blue on the chart. Additionally, it is observable that as the prediction horizon extends, the prediction accuracy for all models tends to diminish, underscoring the inherent challenges of long-term prediction. Nevertheless, the proposed model maintains a relatively stable performance even with increasing time horizons, underscoring its robustness and efficacy.

3.5.2. Comparison study on Herning dataset

The comparison of the proposed model with other SOTA on the Herning dataset, shown in Table 11, highlights the average performance metrics across RMSE, MAE, $1/R^2$ score, and SMAPE. The PISTNet model outperforms all other models, achieving the lowest RMSE (**0.8511** m/s), MAE (**0.6443** m/s), $1/R^2$ score (**1.1535**), and SMAPE (**7.7088**%), setting a new benchmark for WSP using

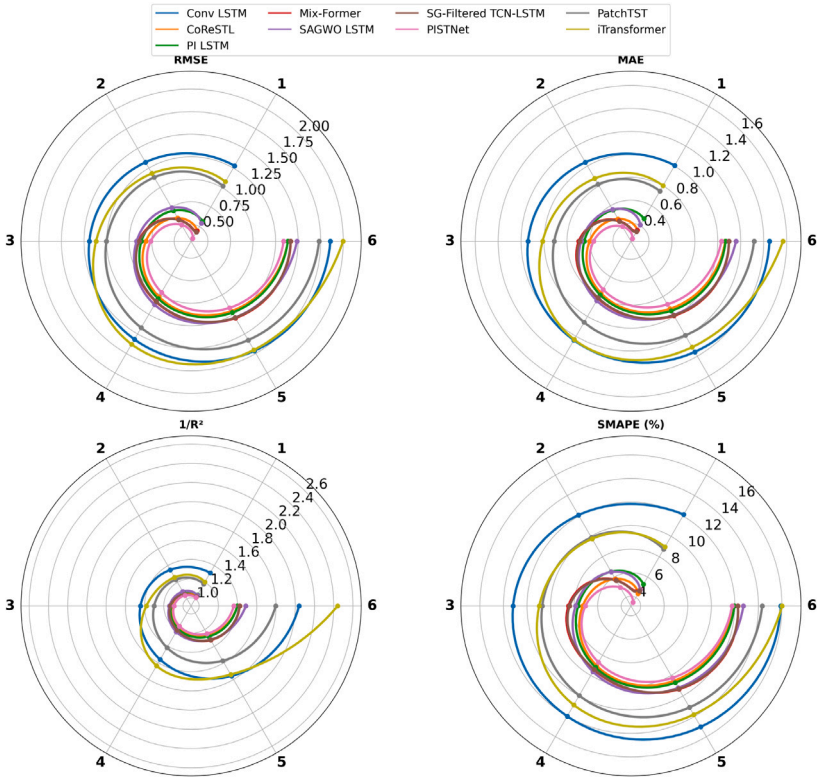


Fig. 13. Performance comparison across different time horizons for each metric on Herning dataset.

Table 12
Average performance metrics on the Palmerston North dataset.

Model name	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
Conv-LSTM [38]	1.7235	1.3641	1.8243	15.6465
CoReSTL [11]	1.0709	0.8010	1.2644	9.5403
PI-LSTM [34]	1.2462	0.9674	1.3816	11.0748
Mixformer [39]	1.0160	0.7665	1.2301	9.4461
SAGWO-LSTM [26]	1.2561	0.9655	1.3577	11.0587
iTransformer [41]	1.6481	1.2080	1.7462	13.5556
PatchTST [42]	1.5706	1.1397	1.6251	12.9815
TCN-LSTM with Savitzky–Golay filter [40]	1.0998	0.8367	1.2675	10.0442
PISTNet	**0.9534**	**0.7055**	**1.2021**	**8.5484**

spatio-temporal datasets. In contrast, the next best-performing model, CoReSTL, has an RMSE of 0.9154 m/s and MAE of 0.6961 m/s, while models like Mixformer, iTransformer, PatchTST and PI-LSTM also show competitive results but fail to surpass PISTNet in both RMSE and MAE.

Table 11, shows the average results of the proposed model, whereas the time horizon comparison for the Hamburg dataset is given in Fig. 13 . In the radial charts, the inner-most circle, shown in pink, clearly illustrates the performance of the proposed model relative to the existing models at each time horizon. On the other hand, the Conv-LSTM model demonstrates the poorest performance across all time horizons, as indicated by its outer most circle in blue color on the chart. Moreover, it is evident that, as the prediction horizon increases, the prediction accuracy for all models tends to decrease, reflecting the natural challenge of long-term prediction. However, the proposed model maintains a relatively stable performance even with an increasing time horizon, highlighting its robustness and effectiveness.

3.5.3. Comparison study on Palmerston North dataset

Table 12 presents the average performance metrics of various models on the Palmerston North dataset. The Conv-LSTM model exhibits the highest RMSE of 1.7235 m/s, MAE of 1.3641 m/s, $1/R^2$ Score of 1.8243 and SMAPE of 15.6465%. The CoReSTL model shows improved performance with an RMSE of 1.0709 m/s, MAE of 0.8010 m/s, $1/R^2$ Score of 1.2644 and SMAPE of 9.5403%.

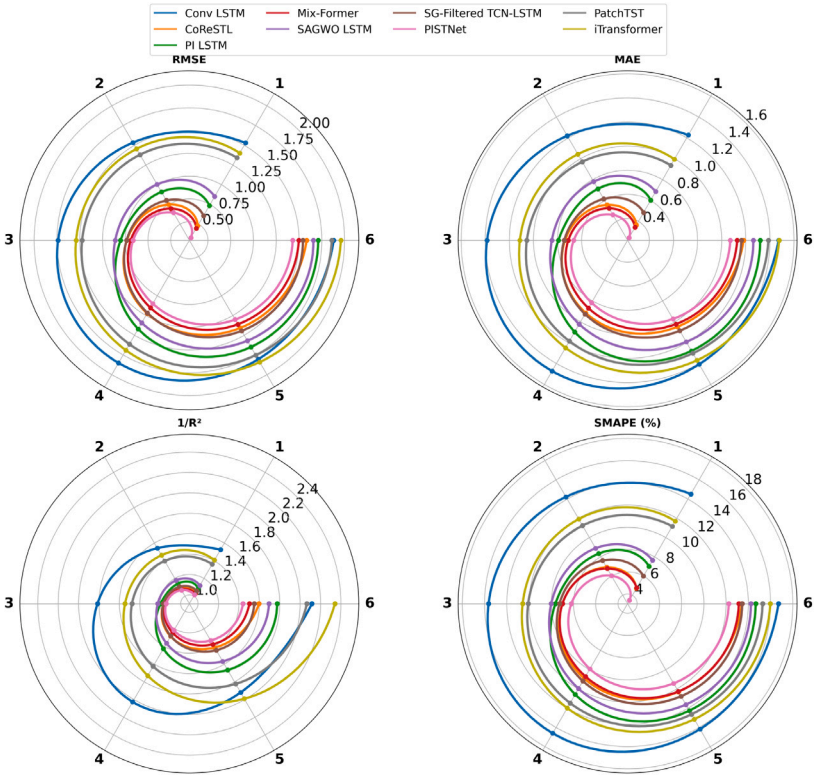


Fig. 14. Performance comparison across different time horizons for each metric on the Palmerston North dataset.

The PI-LSTM and SAGWO-LSTM models have RMSE of 1.2462 m/s and 1.2561 m/s, respectively, with corresponding MAE of 0.9674 m/s and 0.9655 m/s. The Mixformer model achieves an RMSE of 1.0160 m/s, MAE of 0.7665 m/s, $1/R^2$ Score of 1.2301 and SMAPE of 9.4461%. PatchTST and iTransformer still fail to focus on dependencies that are significant for short-term wind speed prediction. The TCN-LSTM with Savitzky–Golay filter model records an RMSE of 1.0998 m/s, MAE of 0.8367 m/s, $1/R^2$ Score of 1.2675 and SMAPE of 10.0442%. The proposed PISTNet model demonstrates the lowest RMSE of 0.9534 m/s, MAE of 0.7055 m/s, $1/R^2$ Score of 1.2021 and SMAPE of 8.5484%. Fig. 14 illustrates the performance of each model based on the average RMSE, MAE and SMAPE metrics. The PISTNet model shows the lowest values across all metrics, indicating better performance compared to the other models. The Conv-LSTM model has the highest errors, while models like Mixformer and TCN-LSTM with Savitzky–Golay filter perform better than the CoReSTL, but not as well as the proposed model.

3.5.4. Comparison study on Silkeborg dataset

Table 13 presents the average performance metrics of SOTA models on the Silkeborg dataset. The Conv-LSTM model has an RMSE of 2.1496 m/s, MAE of 1.6985 m/s, $1/R^2$ Score of 1.4883 and SMAPE of 13.5069%. The CoReSTL model shows improved performance with an RMSE of 1.0647 m/s, MAE of 0.7991 m/s, $1/R^2$ Score of 1.1102 and SMAPE of 6.9331%. The PI-LSTM model records an RMSE of 1.2218 m/s, MAE of 0.9295 m/s, $1/R^2$ Score of 1.1367 and SMAPE of 7.8002%. The Mixformer model achieves an RMSE of 1.1131 m/s, MAE of 0.8286 m/s, $1/R^2$ Score of 1.1172 and SMAPE of 7.1150%. The SAGWO-LSTM model reports an RMSE of 1.5991 m/s, MAE of 1.2457 m/s, $1/R^2$ Score of 1.2433 and SMAPE of 9.5705%. The TCN-LSTM with Savitzky–Golay filter model has an RMSE of 1.1779 m/s, MAE of 0.8964 m/s, $1/R^2$ Score of 1.1281 and SMAPE of 7.6413%. The iTransformer model has an RMSE of 2.4975 m/s, MAE of 1.6528 m/s, $1/R^2$ Score of 1.7889 and SMAPE of 12.4539%, and the PatchTST model shows an RMSE of 2.5213 m/s, MAE of 1.6744 m/s, $1/R^2$ Score of 1.8171 and SMAPE of 12.7038%. The proposed PISTNet model demonstrates the lowest RMSE of 1.0155 m/s, MAE of 0.7564 m/s, $1/R^2$ Score of 1.0998 and SMAPE of 6.6345%.

Fig. 15 illustrates the performance of each model based on the average RMSE, MAE, and SMAPE metrics. The PISTNet model has the lowest values in all metrics, indicating superior performance compared to the other models. The Conv-LSTM model has the highest errors, while models such as Mixformer and TCN-LSTM with Savitzky–Golay filter perform better than the CoReSTL, but do not match the performance of the proposed model. In addition, the iTransformer and PatchTST models, despite having higher error metrics such as RMSE and MAE, show relatively higher $1/R^2$ scores, indicating a stronger correlation with the true values. However, both models demonstrate higher SMAPE values, which suggests that while they may maintain a reasonable relationship with the true data, they are less precise in their predictions compared to models like PISTNet.

Table 13
Average performance metrics on the Silkeborg dataset.

Model name	Average			
	RMSE ↓	MAE ↓	$1/R^2$ Score ↓	SMAPE (%) ↓
Conv-LSTM [38]	2.1496	1.6985	1.4883	13.5069
CoReSTL [11]	1.0647	0.7991	1.1102	6.9331
PI-LSTM [34]	1.2218	0.9295	1.1367	7.8002
Mixformer [39]	1.1131	0.8286	1.1172	7.1150
SAGWO-LSTM [26]	1.5991	1.2457	1.2433	9.5705
iTransformer [41]	2.4975	1.6528	1.7889	12.4539
PatchTST [42]	2.5213	1.6744	1.8171	12.7038
TCN-LSTM with Savitzky–Golay filter [40]	1.1779	0.8964	1.1281	7.6413
PISTNet	**1.0155**	**0.7564**	**1.0998**	**6.6345**

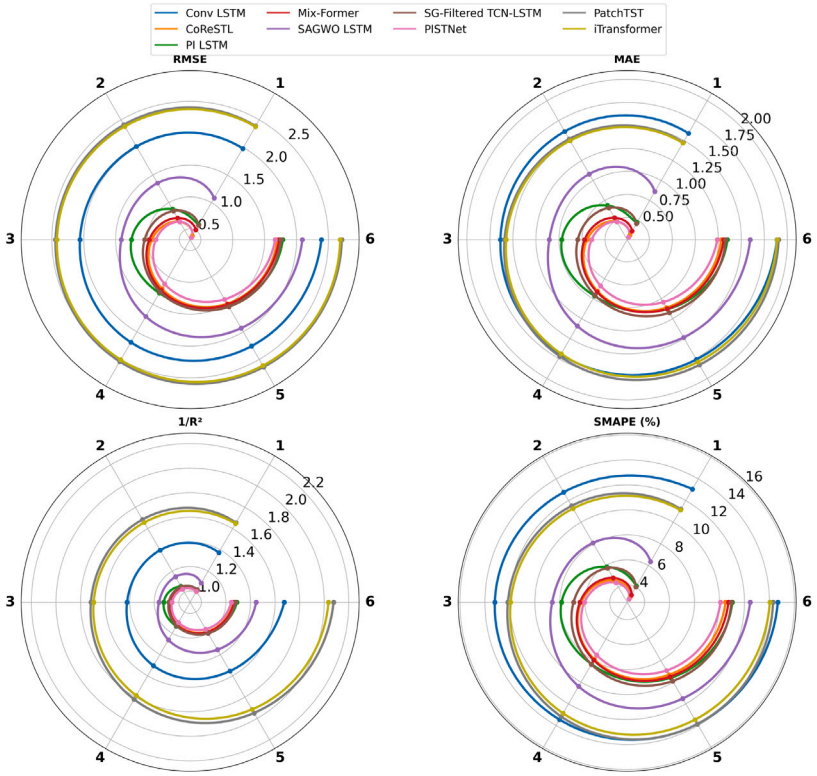


Fig. 15. Performance comparison of models on the Silkeborg dataset.

3.6. Discussion

The proposed PISTNet demonstrates robust performance across diverse wind speed datasets, achieving improved accuracy by integrating spatio-temporal data with physics-informed modeling. However, the framework's design and assumptions require deeper analysis to fully understand their impact on its strengths, limitations, and potential applications.

3.6.1. Assumption-driven error sources and adaptive compensation

The extended advection equation simplifies atmospheric dynamics by assuming two-dimensional flow and constant air density. Ignoring vertical motion limits the model's ability to capture terrain-driven updrafts or downdrafts, such as those occurring in mountainous regions or convective systems. Similarly, constant density neglects variations caused by temperature, humidity, and altitude, leading to inaccuracies in pressure gradient force calculations during rapid thermal changes or in high-altitude regions. Additionally, subgrid-scale turbulence is approximated via the Laplacian diffusion term ($\nu\nabla^2 V$), which oversmooths small-scale fluctuations critical in urban or unstable boundary layers. These simplifications introduce errors in scenarios involving strong stratification, vertical momentum transfer, or fine-grained turbulence.

The hybrid design of PISTNet mitigates these limitations by integrating physics with data-driven adaptations. The neural network compensates for unresolved physics through the DFA module, which dynamically prioritizes spatial features influenced

by terrain or vertical motion cues, and the CAM module, which contextualizes temporal dependencies. Furthermore, the APPL loss function selectively enforces physical constraints only when deviations between neural network predictions and the physical model exceed statistically derived thresholds. This ensures the model retains the interpretability of physics while leveraging data-driven flexibility to address inherent simplifications, balancing accuracy and generalizability without requiring computationally intensive 3D simulations.

3.6.2. Scalability and generalizability

This study focuses on wind speed prediction, PISTNet's modular design enables potential adaptation to other meteorological variables (e.g., temperature, humidity) with targeted adjustments. For such extensions, the advection equation in the EPM module would need replacement with domain-specific physics (e.g., heat/diffusion equations for temperature), and input features would require augmentation (e.g., solar radiation for thermal modeling). The APPL loss function could also be retuned to enforce constraints aligned with the new variable's dynamics, such as thermal diffusion instead of advective transport. This adaptability stems from decoupling physics-based components (EPM, APPL) from data-driven modules (DFA, CAM). While the current implementation prioritizes wind speed dynamics, the framework's separation of physics and data layers allows future customization for other spatio-temporal tasks without architectural redesign. Future work will explore these adaptations to validate the model's universality in multi-variable meteorological forecasting applications.

4. Conclusion

This study presents a physics-informed spatio-temporal neural network (PISTNet) for short-term wind speed prediction (WSP), which integrates spatio-temporal data and physical principles to improve model accuracy. The model incorporates a dynamic feature adapter (DFA) module, a spatio-temporal convolutional recurrent neural network (STCRNN), a context-aware spatio-temporal attention mechanism (CAM), a feature fusion module (FFM) to integrate neural network features with physics-based predictions and an adaptive physics penalty loss (APPL) function to selectively apply physical constraints based on prediction deviations. Extensive experiments on four diverse datasets (Hamburg, Herning, Palmerston North and Silkeborg) demonstrate the effectiveness of the proposed model, showing significant improvements over six state-of-the-art methods across multiple evaluation metrics, including RMSE, MAE, $1/R^2$ score and SMAPE. Ablation studies reveal that the incorporation of physics-based predictions and the APPL loss function notably enhances prediction performance, ensuring reliable wind speed forecasts that support wind energy optimization and grid stability.

The innovation of this work lies in the integration of physical principles with deep learning techniques, which helps to enhance prediction accuracy and reliability. The findings show that the PISTNet model significantly outperforms existing methods, particularly in terms of its ability to handle complex spatio-temporal relationships in wind data. Looking to the future, we plan to focus on enhancing the performance and applicability of the proposed PISTNet model by incorporating advanced physical models to account for complex atmospheric phenomena such as turbulence and three-dimensional flow dynamics. Additionally, integrating real-time data streams and online learning techniques will allow the model to adapt to evolving atmospheric conditions and handle changing wind patterns over time. Expanding the model to predict multiple meteorological variables, such as temperature and humidity, could further improve its forecasting capabilities for renewable energy optimization. Moreover, incorporating transfer learning techniques to adapt the model to different geographical regions and environmental conditions will enhance its versatility and accuracy across diverse wind farms. These future directions will build upon the success of this study and contribute to the development of more robust and adaptive wind speed prediction models.

CRediT authorship contribution statement

Laeeq Aslam: Conceptualization, Methodology, Software, Writing – original draft. **Runmin Zou:** Supervision, Project administration, Validation. **Yaohui Huang:** Conceptualization, Methodology, Writing – original draft, Data analysis. **Ebrahim Shahzad Awan:** Data curation, Investigation. **Sharjeel Abid Butt:** Visualization. **Qian Zhou:** Software, Validation.

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.

Appendix A. Sensitivity analysis discussion

The sensitivity analysis is conducted using the Silkeborg dataset, chosen for its significant temporal variability, characterized by a wide range of wind speeds and the highest standard deviation. These characteristics amplify the impacts of the parameters, making it ideal for identifying critical dependencies of the architecture. Six key parameters were systematically varied while others were kept constant: batch size (16–64), learning rate (0.001–0.1), physics loss percentile (70–90), reduction factors for temporal/feature dimensions (8–24) used need to adjusted in DFA module and units in CAM (16–64). Fig. A.16 illustrates the percentage changes in MAE in these ranges, revealing an exponential sensitivity to the learning rate (lr > 0.05 causing >160% degradation) and moderate batch size effects (+16.35% at B=16).

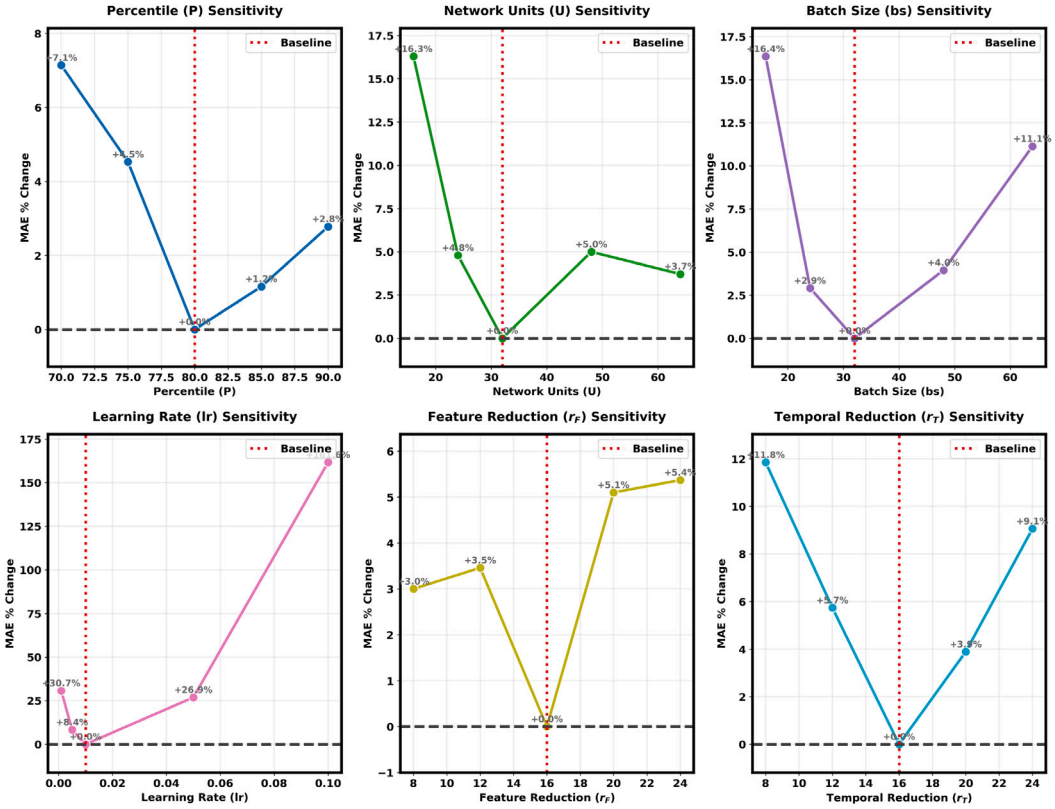


Fig. A.16. MAE percentage changes across parameter variations.

Table A.14
Parameter sensitivity results (Baseline MAE = 0.756)

Parameter	Test range	Max % Δ MAE	Critical threshold	Stable range
Learning Rate (lr)	0.001–0.1	+161.63	>0.05	0.005–0.01
Batch Size (B)	16–64	+16.35	<24, > 48	24–48
Temporal Reduction (r_T)	8–24	+11.85	> 20	12–16
Units (U)	16–64	+16.29	<24	32–64
Feature Reduction (r_F)	8–24	+5.37	> 20	8–20
Percentile (P)	70–90	+7.14	> 80	75–85

The numerical results in Table A.14 demonstrate three sensitivity regimes. Learning rate dominates with 161.63% MAE increase at lr=0.1, while temporal reduction shows linear degradation (+11.85% at r_T=8). Architectural parameters exhibit surprising stability — feature reduction (r_F) remains below +5.37% and percentile threshold (P) under +7.14% . Batch size reveals a U-shaped relationship, with both small (B=16) and large (B=64) values degrading performance by >11%, suggesting an optimal mid-range (24–48). The critical range for dense units (U) in attention layer is > 20 with an optimal value at 32 units.

The analysis suggests two implementation priorities: First, strict learning rate control through adaptive scheduling or gradient clipping, given its exponential sensitivity. Second, architectural preservation of temporal resolution, as reducing r_T below 12, increases MAE by >5%. Batch size requires balanced selection (24–48) to avoid under-fitting or over-fitting, while other parameters show sufficient stability for flexible implementation. These findings enable robust model deployment across varying advection regimes while maintaining physical consistency.

Appendix B. Mathematical definitions of local aggregator and reconstructor

Table B.15 summarizes the operations used in the DFA module. All dimensions are defined relative to the input tensor $\mathcal{X}^{(th)} \in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$.

Table B.15
Mathematical definitions of Local Aggregator and Reconstructor operations.

Operation	Input shape	Operation	Output shape
Temporal Aggregation	$\mathcal{X}^{(th)} \in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$	$\boldsymbol{S}_T = \frac{1}{\Theta}\sum_{\theta=1}^{\Theta} \mathcal{X}^{(th)}_{b,t,f,\theta}$	$\boldsymbol{S}_T \in \mathbb{R}^{B\times(T-1)\times F}$
Temporal Reconstructor	$\boldsymbol{E}_T \in \mathbb{R}^{B\times(T-1)\times F}$	$\texttt{Reconstructor}(\boldsymbol{E}_T) = \text{tile}(\boldsymbol{E}_T, [1,1,1,\Theta])$	$\in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$
Spatial Aggregation	$\boldsymbol{M}^{(t)} \in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$	$\boldsymbol{S}_S = \frac{1}{(T-1)F}\sum_{t=1}^{T-1}\sum_{f=1}^{F} \boldsymbol{M}^{(t)}_{b,t,f,\theta}$	$\boldsymbol{S}_S \in \mathbb{R}^{B\times\Theta}$
Spatial Reconstructor	$\boldsymbol{E}_S \in \mathbb{R}^{B\times\Theta}$	$\texttt{Reconstructor}(\boldsymbol{E}_S) = \text{tile}(\boldsymbol{E}_S, [1,(T-1),F,1])$	$\in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$
Feature Aggregation	$\boldsymbol{M}^{(s)} \in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$	$\boldsymbol{S}_F = \frac{1}{(T-1)\Theta}\sum_{t=1}^{T-1}\sum_{\theta=1}^{\Theta} \boldsymbol{M}^{(s)}_{b,t,f,\theta}$	$\boldsymbol{S}_F \in \mathbb{R}^{B\times F}$
Feature Reconstructor	$\boldsymbol{E}_F \in \mathbb{R}^{B\times F}$	$\texttt{Reconstructor}(\boldsymbol{E}_F) = \text{tile}(\boldsymbol{E}_F, [1,(T-1),1,\Theta])$	$\in \mathbb{R}^{B\times(T-1)\times F\times\Theta}$

Data availability

Data will be made available on request.

References

[1] Manwell JF, McGowan JG, Rogers AL. Wind energy explained: Theory, design and application. John Wiley & Sons; 2009.
[2] Yatiyana E, Rajakaruna S, Ghosh A. Wind speed and direction forecasting for wind power generation using ARIMA model. In: Proceedings of the 2017 australasian universities power engineering conference. IEEE; 2017, p. 1–6.
[3] Tyass I, Khalili T, Rafik M, Abdelouahed B, Raihani A, Mansouri K. Wind speed prediction based on statistical and deep learning models. Int J Renew Energy Dev 2023;12(2):288–99.
[4] Cadenas E, Rivera W, Campos-Amezcua R, Heard C. Wind speed prediction using a univariate ARIMA model and a multivariate NARX model. Energies 2016;9(2):109.
[5] Li X, Li K, Shen S, Tian Y. Exploring time series models for wind speed forecasting: A comparative analysis. Energies 2023;16(23):7785.
[6] Dhakal R, Sedai A, Pol S, Parameswaran S, Nejat A, Moussa H. A novel hybrid method for short-term wind speed prediction based on wind probability distribution function and machine learning models. Appl Sci 2022;12(18):9038.
[7] Peña-Gallardo R, Medina-Rios A. A comparison of deep learning methods for wind speed forecasting. In: Proceedings of the 2020 IEEE international autumn meeting on power, electronics and computing, vol. 4, IEEE; 2020, p. 1–6.
[8] Huang Y, Zhang B, Pang H, Wang B, Lee KY, Xie J, et al. Spatio-temporal wind speed prediction based on Clayton Copula function with deep learning fusion. Renew Energy 2022;192:526–36.
[9] Zhu Q, Chen J, Zhu L, Duan X, Liu Y. Wind speed prediction with spatio–temporal correlation: A deep learning approach. Energies 2018;11(4):705.
[10] Zhang D, Hu G, Song J, Gao H, Ren H, Chen W. A novel spatio-temporal wind speed forecasting method based on the microscale meteorological model and a hybrid deep learning model. Energy 2024;288:129823.
[11] Yan B, Shen R, Li K, Wang Z, Yang Q, Zhou X, et al. Spatio-temporal correlation for simultaneous ultra-short-term wind speed prediction at multiple locations. Energy 2023;284:128418.
[12] Agrawal T, Sheoran S, Pasari S, Shukla S. Wind speed prediction with spatio temporal correlation using convolutional and spiking neural networks. In: Proceedings of the 2022 12th international conference on cloud computing, data science and engineering. IEEE; 2022, p. 197–200.
[13] Parri S, Teeparthi K, Kosana V. A hybrid methodology using VMD and disentangled features for wind speed forecasting. Energy 2024;288:129824.
[14] Liu Z, Liu H. A novel hybrid model based on GA-VMD, sample entropy reconstruction and BiLSTM for wind speed prediction. Measurement 2023;222:113643.
[15] Li Q, Ren X, Zhang F, Gao L, Hao B. A novel ultra-short-term wind power forecasting method based on TCN and informer models. Comput Electr Eng 2024;120:109632.
[16] Zhang C, Li Z, Ge Y, Liu Q, Suo L, Song S, et al. Enhancing short-term wind speed prediction based on an outlier-robust ensemble deep random vector functional link network with AOA-optimized VMD. Energy 2024;296:131173.
[17] He Y, Wang W, Li M, Wang Q. A short-term wind power prediction approach based on an improved dung beetle optimizer algorithm, variational modal decomposition, and deep learning. Comput Electr Eng 2024;116:109182.
[18] Yan Y, Wang X, Ren F, Shao Z, Tian C. Wind speed prediction using a hybrid model of EEMD and LSTM considering seasonal features. Energy Rep 2022;8:8965–80.
[19] Shang Z, Chen Y, Chen Y, Guo Z, Yang Y. Decomposition-based wind speed forecasting model using causal convolutional network and attention mechanism. Expert Syst Appl 2023;223:119878.
[20] Suo L, Peng T, Song S, Zhang C, Wang Y, Fu Y, et al. Wind speed prediction by a swarm intelligence based deep learning model via signal decomposition and parameter optimization using improved chimp optimization algorithm. Energy 2023;276:127526.
[21] Ai X, Li S, Xu H. Wind speed prediction model using ensemble empirical mode decomposition, least squares support vector machine and long short-term memory. Front Energy Res 2023;10:1043867.
[22] Ren Y, Suganthan PN, Srikanth N. A novel empirical mode decomposition with support vector regression for wind speed forecasting. IEEE Trans Neural Netw Learn Syst 2014;27(8):1793–8.
[23] Ren Y, Suganthan PN, Srikanth N. A comparative study of empirical mode decomposition-based short-term wind speed forecasting methods. IEEE Trans Sustain Energy 2014;6(1):236–44.
[24] Wu B, Wang L. Two-stage decomposition and temporal fusion transformers for interpretable wind speed forecasting. Energy 2024;288:129728.
[25] Wu B, Yu S, Peng L, Wang L. Interpretable wind speed forecasting with meteorological feature exploring and two-stage decomposition. Energy 2024;294:130782.
[26] Zhu A, Zhao Q, Yang T, Zhou L, Zeng B. Wind speed prediction and reconstruction based on improved grey wolf optimization algorithm and deep learning networks. Comput Electr Eng 2024;114:109074.
[27] Hu H, Li Y, Zhang X, Fang M. A novel hybrid model for short-term prediction of wind speed. Pattern Recognit 2022;127:108623.

[28] Aslam L, Zou R, Awan ES, Hussain SS, Shakil KA, Wani MA, et al. Hardware-centric exploration of the discrete design space in transformer–LSTM models for wind speed prediction on memory-constrained devices. Energies 2025;18(9):2153.
[29] Zhao X, Liu HP, Jin H, Cao S, Tang G. An improved hybrid model for wind power forecasting through fusion of deep learning and adaptive online learning. Comput Electr Eng 2024;120:109768.
[30] Sheng A, Xie L, Zhou Y, Wang Z, Liu Y. A hybrid model based on complete ensemble empirical mode decomposition with adaptive noise, GRU network and whale optimization algorithm for wind power prediction. IEEE Access 2023;11:62840–54.
[31] Namboodiri V, Goyal R. A novel hybrid ensemble wind speed forecasting model employing wavelet transform and deep learning. Comput Electr Eng 2025;121:109820.
[32] Lagomarsino-Oneto D, Meanti G, Pagliana N, Verri A, Mazzino A, Rosasco L, et al. Physics informed machine learning for wind speed prediction. Energy 2023;268:126628.
[33] Howland MF, Dabiri JO. Wind farm modeling with interpretable physics-informed machine learning. Energies 2019;12(14):2716.
[34] Aslam L, Zou R, Awan E, Butt SA. Integrating physics-informed vectors for improved wind speed forecasting with neural networks. In: Proceedings of the 14th Asian control conference. IEEE; 2024, p. 1902–7.
[35] Yan C, Xu S, Sun Z, Lutz T, Guo D, Yang G. A framework of data assimilation for wind flow fields by physics-informed neural networks. Appl Energy 2024;371:123719.
[36] Shao X, Liu Z, Zhang S, Zhao Z, Hu C. PIGNN-CFD: A physics-informed graph neural network for rapid predicting urban wind field defined on unstructured mesh. Build Environ 2023;232:110056.
[37] Stackhouse P. NASA prediction of worldwide energy resource (POWER). Recuper El 2014;27.
[38] Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-K, Woo W-c. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: Proceedings of the 28th Annual Conference on Neural Information Processing Systems, vol. 28, Montreal, Quebec, Canada; 2015, p. 802–10.
[39] Wu T, Ling Q. Mixformer: Mixture transformer with hierarchical context for spatio-temporal wind speed forecasting. Energy Convers Manage 2024;299:117896.
[40] Liu S, Xu T, Du X, Zhang Y, Wu J. A hybrid deep learning model based on parallel architecture TCN-LSTM with Savitzky-Golay filter for wind power prediction. Energy Convers Manage 2024;302:118122.
[41] Liu Y, Hu T, Zhang H, Wu H, Wang S, Ma L, et al. Itransformer: Inverted transformers are effective for time series forecasting. 2023, arXiv preprint arXiv:2310.06625.
[42] Nie Y, H. Nguyen N, Sinthong P, Kalagnanam J. A time series is worth 64 words: Long-term forecasting with transformers. In: International conference on learning representations. 2023.