Hardware-Centric Exploration of the Discrete Design Space in Transformer-LSTM Models for Wind Speed Prediction on Memory-Constrained Devices

Laeeq Aslam , Runmin Zou * , Ebrahim Shahzad Awan , Sayyed Shahid Hussain , Kashish Ara Shaki * , Mudasir Ahmad Wani , Muhammad Asim

Posted Date: 26 February 2025

doi: 10.20944/preprints202502.2066.v1

Keywords: wind speed prediction; power forecasting; hyper-parameter tuning; model size optimization; renewable energy management; on-device deployment

Article

Hardware-Centric Exploration of the Discrete Design Space in Transformer-LSTM Models for Wind Speed Prediction on Memory-Constrained Devices

Laeeq Aslam [1], Runmin Zou [1,*], Ebrahim Shahzad Awan [2], Sayyed Shahid Hussain [1], Kashish Ara Shaki [3,*], Mudasir Ahmad Wani [4] and Muhammad Asim [4]

[1] School of Automation, Central South University, Changsha 410083, Hunan, China
[2] School of Engineering, Design and Built Environment, Western Sydney University, Penrith, 2747, New South Wales, Australia
[3] Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
[4] EIAS Data Science Lab, College of Computer and Information Sciences, and with Center of Excellence in Quantum and Intelligent Computing, Prince Sultan University, Riyadh 11586, Saudi Arabia
* Correspondence: Runmin Zou: rmzou@csu.edu.cn; Kashish Ara Shaki:kashakil@pnu.edu.sa

Abstract: Wind is one of the most important resources in the renewable energy basket. However, questions regarding wind as a sustainable solution, especially when it faces challenges such as upfront cost, visual impact, noise pollution and bird collisions. These challenges arise from commercial windmills whereas for domestic small-scale windmills these challenges are limited. On the other hand, accurate wind speed prediction (WSP) is crucial for optimizing power management in renewable energy systems. Existing research focuses on proposing model architectures and optimizing hyper-parameters to improve model performance. This approach often results in larger models, which are hosted on cloud servers. Such models face challenges, including bandwidth utilization leading to data delays, increased costs, security risks, concerns about data privacy and the necessity for continuous internet connectivity. Such resources are not available for domestic windmills. To overcome these obstacles, this work proposes a transformer model integrated with Long Short-Term Memory (LSTM) units, optimized for memory-constrained devices (MCD). A contribution of this research is the development of a novel cost function that balances the reduction of mean square error with the constraints of model size. This approach enables model deployment on low-power devices, avoiding the challenges of cloud-based deployment. The model, with its tuned hyper-parameters, outperforms recent methodologies in terms of mean squared error, mean absolute error, model size and R-squared scores across three different datasets. This advancement leads the way for more dynamic and secure on-device wind speed prediction (WSP) applications, representing a step forward in renewable energy management.

Keywords: wind speed prediction; power forecasting; hyper-parameter tuning; model size optimization; renewable energy management; on-device deployment

1. Introduction

Wind energy is an important alternative to fossil fuels, contributing significantly to global renewable energy capacity. The Global Wind Report 2022 [1] noted a 94 GW increase in 2021, although growth slowed in major markets like China and the United States. Offshore wind saw significant expansion, particularly in China. However, one of the significant challenges in wind energy management is intermittency, which refers to the unpredictable nature of wind. Wind speeds, influenced by weather patterns, fluctuate throughout the day, resulting in inconsistent power output. However, this problem is usually solved by predicting wind speed before time and then managing other available resources

 doi:10.20944/preprints202502.2066.v1

integration with the wind energy system for sustainable power generation. Various prediction techniques have been developed, which can be categorized into physical models, statistical models and machine learning (ML) models.

Numerical Weather Prediction (NWP) models use physical principles for wind speed prediction (WSP), eliminating the need for historical data and model training. These models simulate atmospheric conditions to provide location-specific forecasts, as demonstrated by Brabec et al. [2]. However, their performance is sensitive to initial physical conditions, where minor errors can affect results. Another study by Moreno et al. [3] points out the complexity and high computational costs associated with NWP models. Additionally, accurate input data and substantial computational resources are required, since inaccuracies in initial atmospheric data can lead to forecast errors. Nevertheless, when integrated into hybrid models, NWP models become valuable tools for wind speed prediction, particularly in regions lacking historical wind data [2,4].

On the other hand, statistical models treat WSP as a stochastic process, using historical data to identify time-variable relationships. Common statistical models include autoregressive (AR), moving average (MA), autoregressive integrated moving average (ARIMA) and the Kalman filter [5]. These models have been applied extensively due to their simplicity and relatively low computational cost. However, they assume linear relationships, which may not accurately capture the nonlinear characteristics of wind speed (WS) time series. Torres et al. [6] apply an ARMA model to predict hourly mean WS in Spain, demonstrating that while such models provide reasonable short-term forecasts, they struggle with the nonlinear and random nature of wind speeds. Li et al. [7] utilize ARMA-based approaches for WSP, showing that these models can be effective under certain conditions but have limitations in capturing complex wind patterns. Collectively, these statistical models contribute to understanding WS patterns by providing a foundational approach to time series analysis. However, their limitations in capturing nonlinearity and randomness highlight the need for more advanced methods that can model the complex behavior of wind speeds.

Machine learning (ML) models, including artificial neural networks (ANN), support vector machines (SVM) and deep learning methods, are applied in wind speed prediction (WSP) for their ability to model nonlinear relationships [8]. Unlike statistical models, ML models do not assume normality of residuals or stationarity of the time series. ANNs model wind speed as a nonlinear system, as demonstrated by Bechrakis [9] and Mohandes et al. [10] show the potential of SVMs for WSP. These models capture patterns in wind speed data but require large datasets and significant computational power for training. Deep learning models, such as LSTM networks, capture long-term dependencies in wind speed data. Combining these approaches can improve prediction accuracy. Hybrid models (HM) integrate various techniques to maximize their benefits. Geng et al. [11] use LSTM networks for short-term WSP, achieving higher accuracy than traditional methods. Cai et al. [12] apply Extreme Gradient Boosting (XGBoost) for WSP, effectively handling large datasets and using regularization to prevent overfitting. Aslam et al. [13] propose a physics-informed machine learning approach with a novel cost function to enhance WSP by collecting features from surrounding locations to predict future wind speeds. ANNs and SVMs model nonlinear relationships, while LSTM and XGBoost handle sequential data and large datasets, respectively. These techniques together improve prediction accuracy and model complex wind speed behaviors.

Hybrid models combine multiple approaches to leverage their strengths and address individual limitations. HM often merge statistical models with ML techniques, such as combining ARIMA with ANN or SVM to improve prediction quality [14]. Recent studies enhance time series data preprocessing using Singular Spectrum Analysis (SSA) and wavelet transforms to increase model accuracy [15]. Advances in HM include transformer models, initially developed for natural language processing, applied to WSP. Wan et al. [16] and Pan et al. [17] integrate convolutional neural networks (CNN) with LSTM networks to capture spatial and temporal dependencies in wind speed data. The attention mechanism is added to these models to focus on relevant parts of the input sequence, improving forecasting accuracy [18]. By combining different methodologies, HM provide a balanced approach

 doi:10.20944/preprints202502.2066.v1

that enhances WSP accuracy and reliability, managing the complexities and nonlinearity in wind speed data effectively.

Since intermittency is just one challenge related to wind energy utilization, besides this, some of the other fundamental challenges in wind power production include visual impact [19–21], noise pollution [22,23], bird and bat collisions [24–26], upfront costs [27–29] and grid integration [30,31]. These issues primarily arise from commercial wind energy projects and raise questions about the sustainability of wind energy as a solution. Although these challenges are significant for commercial wind energy projects, increasing domestic small-scale windmill installations can reduce the impact of these problems.

In existing studies such as [32–34], the challenge of intermittency and predictions is mostly studied in the context of commercial windmills, which can afford large cloud servers to run sophisticated ML models for accurate predictions. In these projects, the focus is on improving prediction accuracy and the size of the models is not a significant concern. For small windmills, however, accessing a server to run such models presents issues like latency, data privacy, security, bandwidth usage, cloud server costs and energy consumption. Therefore, designing smaller models that predict WS with minimal error can avoid these challenges and efficiently manage energy locally. Deploying ML models on edge devices is a potential solution, but this area is not well-studied in the context of domestic wind energy production. Addressing these challenges for small windmills, this research investigates a DDS comprising the hyper-parameters of a transformer LSTM hybrid model in such a way that not only the prediction accuracy of such models is improved but also the size of the models is significantly reduced.

The primary contributions of this research can be summarized as follows:

- This work presents a hybrid baseline model (HBM) that combines a Transformer model with LSTM, designed for optimal tuning with variable hyper-parameters.
- Introduces a novel cost function and utilizes Genetic Algorithms for optimizing discrete design spaces (DDS) influenced by model hyper-parameters.
- Incorporates hardware-specific performance evaluations at each optimization step, ensuring effective deployment on targeted hardware.

This research advances the understanding and application of transformer LSTM models for wind speed prediction on low-power devices. It explores trade-offs and offers solutions for challenges related to memory constraints and prediction accuracy.

2. Proposed Methodology

This study utilizes a baseline model with two main components: a Transformer encoder and LSTM layers. The Transformer encoder extracts features by handling complex data patterns, while the LSTM layers, as the prediction head, capture temporal dependencies. This section first details the Transformer-LSTM baseline architecture. It then discusses model performance metrics, which are integral to the cost and fitness functions. Finally, a hardware-centric DDS optimization scheme is presented.

2.1. Baseline Architecture

The baseline model's architecture is shown in Figure 1. First, the input features go through a dense embedding layer. This layer changes the dimensions of the input data using equation 1:

$$\text{Dimension} = N_H \times H_S \tag{1}$$

Here, N_H is the number of heads and H_S is the head size.

Next, positional embeddings are added using sinusoidal functions:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{2}$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 February 2025 doi:10.20944/preprints202502.2066.v1

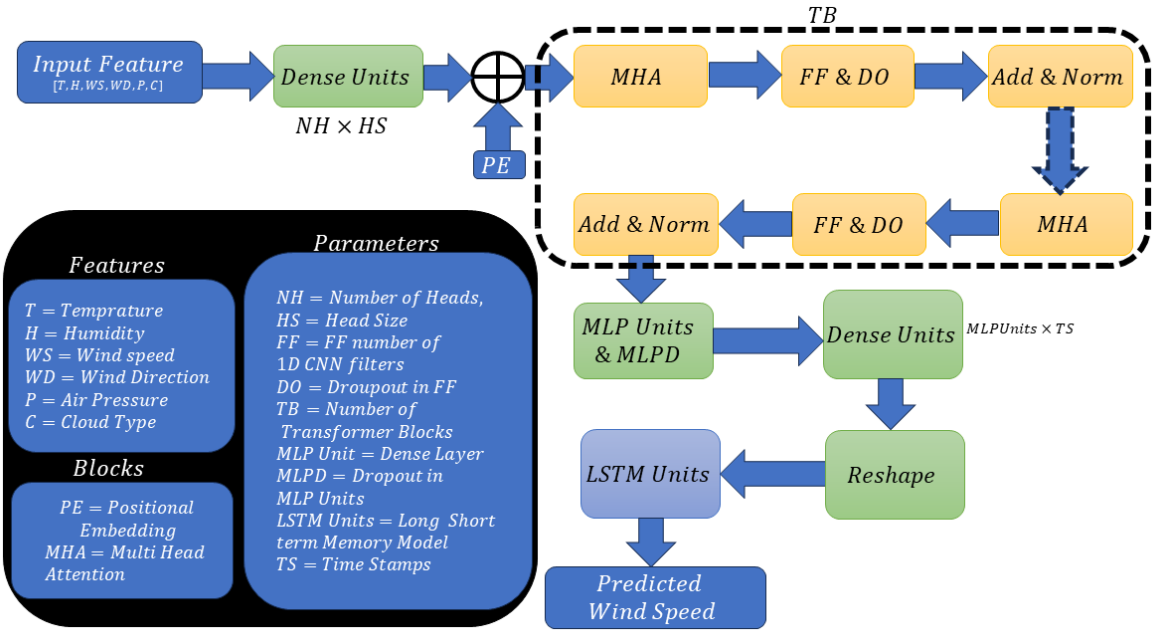

Figure 1. Block diagram of the proposed baseline model.

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{3}$$

where, pos is the position in the sequence, i is the dimension index and d_{model} is the dimensions. Then, the features pass to the Multi-Head Attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

Where, Q, K and V are the query, key and value matrices and d_k is the dimension of the keys. This computes attention scores, focusing on key parts of the input data. Next, the features pass through a 1D convolutional neural network (CNN) layer that applies a ReLU activation function, adding non-linearity to help the model learn complex patterns. Following the CNN layer, an Add and Normalize layer is used:

$$\text{Layer Output} = \text{LayerNorm}(\text{Input} + \text{Output}) \tag{5}$$

This layer normalizes the data, stabilizing the learning process. The model then concatenates T_B transformer blocks, each with a Multi-Head Attention mechanism, a CNN layer, a feed-forward network and an Add and Normalize layer. Next, a layer of MLP Dense Units is added. These dense layers enhance the representation. The features then pass through more dense layers, adjusting dimensions to match the number of time stamps. This is followed by a reshape layer, organizing features by time stamps. The final dense layer allocates features to specific time stamps for better prediction accuracy. Finally, a layer of LSTM units acts as a regression head to predict future events, suitable for tasks like forecasting wind speeds. The LSTM uses several gates to manage information flow:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{6}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{7}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{8}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{9}$$

 doi:10.20944/preprints202502.2066.v1

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{10}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{11}$$

These equations (6 to 11) show how the LSTM manages information. The forget gate (f_t) discards irrelevant data, the input gate (i_t) updates the cell state with new information and the output gate (o_t) outputs part of the cell state. These mechanisms help the LSTM make accurate predictions based on historical data.

This baseline model is better than a purely LSTM or Transformer model because it leverages the strengths of both architectures. The Transformer's attention mechanism excels at capturing contextual relationships in the data, while the LSTM is adept at handling long-term dependencies in sequential data. This combination allows the model to address both short-term and long-term patterns effectively, leading to more accurate predictions.

2.2. Model Performance Metrics

The performance of the model is quantitatively assessed using various metrics, such as MSE, MAE, the R^2 score, the model's storage size on disk and its latency. The formulas for these metrics are as follows:

R^2 Score (Coefficient of Determination):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{12}$$

Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{13}$$

Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{14}$$

In the above equations, y_i denotes the actual wind speed, $\hat{y}_i$ the predicted WS and $\bar{y}$ the average of the actual wind speeds. MSE is sensitive to larger errors due to its squaring of error terms, whereas MAE is sensitive to the median of errors as it considers the absolute value of errors.

2.3. Cost and Fitness Functions

The proposed baseline model has an ability to capture time series data patterns. However, such models have a high MS if not optimized properly. Hence, there is a trade-off between prediction accuracy, quantified by the MSE and MS.

Navigating this balance can be seen as optimization of a DDS. Unlike a continuous space where we have infinite options to select the values of the parameters, in a DDS choices are quantized, offering only a finite set of possibilities. The pivotal components of our model, all within this DDS, comprise:

- **Head Size** (HS): Influences the granularity of attention and overall model capacity.
- **Number of Heads** (NH): Enables concurrent attention mechanisms, facilitating the model's understanding of diverse data facets.
- **Feed-Forward Dimension** (FF): Signifies the inner processing capability of the feed-forward network within the Transformer.
- **Number of Transformer Blocks** (TB): Multiple blocks augment the model's depth, enhancing its ability to discern complex patterns.
- **MLP Units** ($MLPU$): Governs the capacity of the dense layers within the model.
- **Dropout** (DO): Acts as a regularization knob in the transformer encoder section.
- **MLP Dropout** ($MLPD$): Acts as a regularization knob in the dense layer section, mitigating the risk of overfitting.
- **LSTM Units** ($LSTMU$): Determines the temporal processing power of the LSTM layers.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 February 2025 doi:10.20944/preprints202502.2066.v1

The optimization problem, considering these parameters, can be given as in equation 15.

$$\min_{A} C \quad \text{such that} \quad AX - C = 0 \tag{15}$$

Where,

$$X = \begin{bmatrix} HS & NH & FF & TB & MLPU & DO & MLPD & LSTMU \end{bmatrix}^T \tag{16}$$

represents the known configurations of the model, consolidated into a column vector. Our objective is to identify the optimal coefficient matrix A to minimize the cost C. The cost metric C symbolizes the equilibrium between accuracy and MS, defined as in equation 17.

$$C = \epsilon \times \left(\frac{\text{MSE}}{\alpha}\right) + (1 - \epsilon) \times \left(\frac{\text{Size of Model}}{\beta}\right) \tag{17}$$

where ϵ varies between 0 and 1, serving as a balancing agent between the model's predictive prowess (MSE scaled by α in m/s) and its computational footprint (Size of Model scaled by β in bytes).

Each column of A is subject to specific upper threshold constraints, with different data type requirements: $0 \leq A_1 \leq$ mhs, $0 \leq A_2 \leq$ mnh, $0 \leq A_3 \leq$ mffd, $0 \leq A_4 \leq$ mnb, $0 \leq A_5 \leq$ mmlpu, $0 \leq A_6 \leq$ mdo, $0 \leq A_7 \leq$ mmlpdo, $0 \leq A_8 \leq$ mlstmu. Here, the upper limits (mhs, mnh, mffd, etc.) represent the maximum permissible values for each parameter (Max_Head_Size, Max_Number_of_Heads, etc.), ensuring the model's feasibility and efficiency within the specified constraints.

As in this work, a Genetic Algorithm (GA) inspired technique is employed to identify models that minimize prediction error while also maintaining a minimal size. Consequently, we have devised a fitness function that incorporates an additional constraint to ensure model effectiveness. This constraint ensures that the model must achieve a positive R^2 score, which is a standard metric for assessing the goodness of fit in regression models. The fitness function, therefore, integrates this additional criterion and is formulated as in equation 18.

$$\text{Fitness} = C - \lambda \times \text{R2 score} \tag{18}$$

This method ensures the chosen model balances accuracy and size while meeting a basic standard of predictive performance, as measured by the R^2 score.

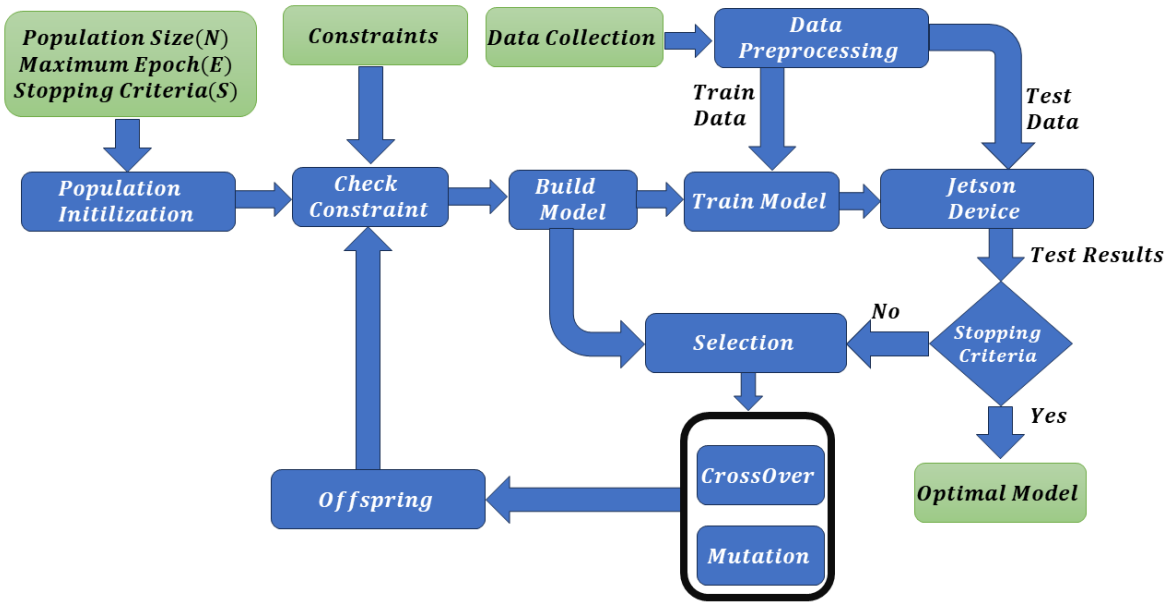

Figure 2. Block Diagram of the Proposed Scheme

 doi:10.20944/preprints202502.2066.v1

2.4. Hardware-Centric DDS Optimization Scheme

In this section, the proposed hardware-centric DDS optimization algorithm is discussed. The proposed methodology is graphically explained in Figure 2. This scheme works on the optimization principles of the Genetic Algorithm. The details of the proposed scheme are as follows:

1. A Genetic Algorithm initiates a random generation of models, where each gene represents a specific aspect of the model and one chromosome can be given as X in equation 16. Once these models are generated, they are passed through a constraint check.
2. These models are then built and trained for a limited duration of 25 epochs. Post-training, they are deployed on a memory-constrained device to evaluate their performance along with the test data.
3. The fitness of these models is calculated based on the test results, using equation 18, where λ is a Lagrange multiplier that penalizes models with a negative R^2 score after 25 epochs.
4. The GA process for generating new models involves selection, mutation and crossover by considering the following aspects:
    - If there is an improvement in performance over the last three generations, the two best models (minimum fitness value) are selected for mutation and crossover.
    - If performance does not improve, one best parent and a second randomly selected gene are used for mutation and crossover.
    - During mutation or crossover, if any gene exceeds its predefined constraint limits, a random value within the permissible range is reassigned to maintain the MS within the required memory space.
5. The process returns to step 2 unless a stopping criterion is met.
6. Finally, the top 5 models from the experiment are selected for further training over an extended number of epochs. They are then re-evaluated on the memory-constrained device to ascertain their final performance metrics.

In this work, models are trained on a server and tested on MCD like the Jetson. This dual environment and hardware-centric approach is more practical compared to the standard procedure of neural architecture search, where models are trained and tested simultaneously on the same machines, typically servers. In our research, models are trained on servers to accelerate the training process and then tested on the actual hardware. Since results may vary when models are deployed to such devices, testing them before selecting the best parents ensures the selection of solutions that are optimal for the specific hardware. For transferring trained models from the server to the Jetson device, Google's gRPC protocol is employed, ensuring efficient and reliable model transfer.

Training lasts for 25 epochs to quickly assess model behavior and adaptability. A model's performance during this period, indicated by MSE and R^2 scores, determines its potential for further development or the need for reevaluation. Predefined limits on hyper-parameters prevent training failures in memory-constrained environments, ensuring models remain within available memory space and are practical for real-world deployment.

Furthermore, an adaptive Genetic Algorithm strategy introduces new random genes when performance stagnates over three generations. This keeps the search space diverse and dynamic, promoting exploration and increasing the chance of finding effective models. Hence, the top 5 models are selected based on initial performance metrics for extended training beyond 25 epochs. This approach ensures promising candidates are refined further. To enhance understanding of the proposed methodology, the following pseudo code succinctly outlines the scheme:

3. Results and Comparative Analysis

This section discusses the results and a comparative analysis of our proposed methodology, starting with a detailed overview of the datasets and parameter configurations.

 doi:10.20944/preprints202502.2066.v1

Algorithm 1 Hardware-Centric DDS Optimization Algorithm

Require: N (population size), E (maximum epochs), S (stopping criteria)
Ensure: Optimized models M^*

$P \leftarrow \text{Initialize}(N)$ ▷ Generate initial population
$G \leftarrow 0$ ▷ Generation counter
while $\neg\text{StoppingCriteria}(G, S)$ **do**
  for all $m \in P$ **do**
    $\text{Train}(m, E)$
    $\text{Eval}(m)$
    $m.\varphi \leftarrow \epsilon \times \text{MSE}(m) + (1 - \epsilon) \times \text{Size}(m) - \lambda \times R^2(m)$
  end for
  $P' \leftarrow \emptyset$ ▷ New population
  if $\text{Improved}(P, 3)$ **then**
    $\{p_1, p_2\} \leftarrow \text{SelectBest}(P, 2)$
  else
    $\{p_1, p_2\} \leftarrow \{\text{SelectBest}(P, 1), \text{SelectRandom}(P)\}$
  end if
  for $i \leftarrow 1$ **to** $N/2$ **do**
    $\{c_1, c_2\} \leftarrow \text{Crossover}(p_1, p_2)$
    $\text{Mutate}(c_1)$
    $\text{Mutate}(c_2)$
    $\text{Constraint}(c_1)$
    $\text{Constraint}(c_2)$
    $P' \leftarrow P' \cup \{c_1, c_2\}$
  end for
  $P \leftarrow P'$
  $G \leftarrow G + 1$
end while
$M^* \leftarrow \text{SelectTop}(P, 5)$
$\text{ExtendedTraining}(M^*)$
$\text{ReEval}(M^*)$
return M^*

 doi:10.20944/preprints202502.2066.v1

3.1. Datasets and Parameter Configurations

The datasets used in this research were collated by the National Renewable Energy Laboratory (NERL), focusing on three significant locations in Pakistan: Gwadar, Pasni and Jhimpir. Pakistan, being the fifth largest country, has been facing an energy crisis for decades and the resources available to the government are insufficient to install commercial windmills. Instead, domestic small-scale windmills can be installed by the local government with a lesser budget, supporting the United Nations Sustainable Development Goals (SDGs), particularly SDG 7 (Affordable and Clean Energy).

The datasets record meteorological variables at 60-minute intervals, including temperature, wind speed (WS), wind direction, atmospheric pressure and cloud type. This study focuses on DDS for MCD, targeting one-step-ahead forecasts. The dataset is pre-processed to include the past 24 hours' measurements and then normalized. The model aims to estimate WS (in m/s) for the upcoming 25th hour, denoted as $\hat{y}_i$. After prediction, denormalization transforms $\hat{y}_i$ back to the actual scale to determine the real-world wind speed, y_i.

The maximum values for the model parameters are represented as a row vector $\mathbf{a}_{\max}$: mhs $= 512$, mnh $= 16$, mffd $= 8$, mnb $= 8$, mmlpu $= 256$, mdo $= 1$, mmlpdo $= 1$ and mlstmu $= 128$. Thus, $\mathbf{a}_{\max} = [512, 16, 8, 8, 256, 1, 1, 128]$. The models are trained on an Nvidia GeForce RTX 3080 Ti and, to ensure practical deployment on MCD, the models are optimized for the Nvidia Jetson Nano. The Nvidia Jetson Nano is designed for edge computing, featuring a quad-core ARM Cortex-A57 CPU and an integrated GPU, making it energy-efficient and suitable for real-time applications in renewable energy management.

3.2. Results

In this study, the parameter λ in Equation 18 is set to 0.001. Additionally, α is chosen as 1 m/s and β is set to 1,048,576 Bytes. These parameter values are intuitively determined by evaluating the performance of the baseline model, which has the maximum possible size for hyper-parameters. Additionally, five distinct experiments are conducted, each varying the value of ϵ in Equation 18. The chosen values for ϵ in these experiments are 0.01, 0.25, 0.5, 0.75 and 0.99. In the experiment where ϵ is 0.01, the primary focus is on minimizing the MS, with a negligible concern for the MSE. As the value of ϵ increases to 0.25, there is a noticeable shift in priority, placing greater emphasis on reducing MSE while lessening the importance of MS. This trend continues with higher values of ϵ; as ϵ increases, the significance given to minimizing MSE grows, consequently reducing the emphasis on the size of the model.

Initial training over 25 epochs yields five distinct models for each case. Upon fine-tuning these models using the dataset from Gwadar city, results are obtained for varying values of epsilon. These are depicted in Figures **??** to **??**. Each figure comprises six subplots: the first subplot summarizes the results of the five models, while the remaining five subplots detail the DDS each model occupies. For instance, Figure **??** features a model in the first row and second column, highlighted in purple, with the optimal parameters $X = [310, 5, 2, 2, 100, 0.21, 0.31, 37]$. The notation 'NH 5/16' indicates that the optimal number of heads is 5, within the maximum allowed limit of 16.

3.2.1. **Experiment with** $\epsilon = 0.01$

Figure **??** showcases the top five models for $\epsilon = 0.01$, focusing on the trade-off between MS and performance. The most efficient model achieves an impressive balance with a total cost of 0.28, an MSE of 0.11 m/s and a MS of only 0.28 MB. The second model, with a total cost of 0.33, an MSE of 0.22 m/s and a MS of 0.33 MB, shows both an increase in size and MSE. This makes it a less favorable choice, as it does not efficiently balance the trade-off between size and performance.

In contrast, the third model demonstrates a significant improvement in performance at the expense of increased size. It registers a total cost of 0.45, an MSE of 0.08 m/s and a MS of 0.45 MB, indicating a substantial enhancement in accuracy for a reasonable increment in MS. The fourth and fifth models,

 doi:10.20944/preprints202502.2066.v1

with slightly different sizes of 0.46 MB and 0.47 MB but identical total costs of 0.46 and MSE of 0.06 m/s, showcase a consistent level of high performance for these larger models.

3.2.2. **Experiment with** $\epsilon = 0.25$

Figure **??** presents the top five models for $\epsilon = 0.25$, where the balance between MS and performance is considered under different criteria compared to the earlier experiment. The best-performing model in this setup achieves a total cost of 0.29, an MSE of 0.06 m/s and a MS of 0.36 MB, demonstrating an efficient balance between accuracy and compactness. The second model exhibits a total cost of 0.31, an MSE of 0.09 m/s and a MS of 0.38 MB. This model indicates a preference for slightly larger size while still maintaining high accuracy.

Moving to the third model, it presents a total cost of 0.39, an increased MSE of 0.26 m/s and a MS of 0.44 MB. This suggests a shift towards accommodating a larger MS in exchange for a moderate increase in error rates. The fourth model shows an increase in both total cost (0.44) and MS (0.48 MB) but with a slightly higher MSE of 0.3 m/s, indicating a trade-off for a larger MS against a marginal increase in error performance. Interestingly, the final model, despite having the largest size at 0.6 MB, achieves a low MSE of 0.05 m/s with a total cost of 0.46. This outcome suggests that a larger MS can significantly improve accuracy, although at the cost of increased resource consumption. Detailed optimal parameters for these models are depicted in Figure **??**.

3.2.3. **Experiment with** $\epsilon = 0.50$

In the experiment characterized by $\epsilon = 0.5$, as illustrated in Figure **??**, a distinct set of outcomes is observed, which leans towards a more balanced trade-off between MS and performance. The first model achieves a total cost of 0.25, an MSE of 0.05 m/s and a MS of 0.44 MB, demonstrating a favorable balance between accuracy and size.

The second model, with a slightly larger size of 0.46 MB, also records a total cost of 0.25 but achieves a slightly lower MSE of 0.04 m/s, indicating an incremental improvement in accuracy. Moving on to the third model, the total cost rises to 0.3, the MSE to 0.04 m/s and the MS to 0.56 MB. This model suggests a preference for a larger size while maintaining a low error rate, balancing between cost and performance.

The fourth model, significantly larger at 0.8 MB, demonstrates a total cost of 0.42 and an MSE of 0.04 m/s. This indicates a considerable enhancement in accuracy, which comes with a substantial increase in size. Finally, the fifth model, with a size of 0.85 MB, shows a total cost of 0.44 and an MSE of 0.04 m/s. Although similar in size to the fourth model, it presents a slightly higher total cost, thus marking it as slightly less efficient in comparison. Detailed parameters for these models, including their specific configurations, are further elucidated in Figure **??**.

3.2.4. **Experiment with** $\epsilon = 0.75$

The experiment with $\epsilon = 0.75$, as shown in Figure **??**, reveals a compelling set of results, emphasizing a greater focus on model accuracy while also considering MS. The first model in this series achieves a total cost of 0.218, an MSE of 0.05 m/s and a MS of 0.721 MB. This indicates a shift towards prioritizing accuracy while maintaining a moderately large size.

The second model, with a MS of 0.927 MB, has a total cost of 0.267 and an MSE of 0.047 m/s. This model suggests further enhancement in accuracy, with a noticeable increase in size. Advancing to the third model, the total cost rises to 0.324, the MSE to 0.076 m/s and the MS to 1.065 MB, reflecting a continued emphasis on reducing the error rate, albeit at the expense of larger model dimensions.

The fourth model, at 1.191 MB, shows a cost of 0.343 and an MSE of 0.06 m/s. Despite its size, it balances well between size and accuracy. The fifth model, the largest at 1.311 MB, records a cost of 0.368 and an MSE of 0.054 m/s. Although the largest, it maintains good accuracy, illustrating the balance between size and performance. Details of these models are in Figure **??**.

 doi:10.20944/preprints202502.2066.v1

3.2.5. **Experiment with $\epsilon = 0.99$**

The final experiment, with $\epsilon = 0.99$ as shown in Figure **??**, represents an approach with emphasis on MSE reduction, while placing minimal constraints on the MS. Hence, the first model achieves a total cost of 0.059, an MSE of 0.047 m/s and a MS of 1.217 MB, showing a significant emphasis on accuracy. The second model, with a total cost of 0.062, an MSE of 0.05 m/s and a slightly smaller size of 1.163 MB, maintains a similar accuracy level with a marginal reduction in size.

Moreover, the third model, with a total cost of 0.063, an MSE of 0.057 m/s and a smaller MS of 0.655 MB, presents a better balance between size and accuracy compared to its predecessors. The fourth model, at 0.781 MB, shows a higher cost of 0.177 and an MSE of 0.171 m/s, indicating improvements in both size and accuracy. Finally, the fifth model, despite a slightly smaller size of 0.719 MB, records the highest cost of 0.207 and an MSE of 0.202 m/s. This highlights the difficulty of optimizing for high accuracy while maintaining a reasonable MS. Detailed insights into the model configurations and trade-offs are provided in Figure **??**. This study shows the top five models for each epsilon value, highlighting trade-offs. For example, in Experiment 1 with $\epsilon = 0.01$, an MSE of 0.08 m/s or less is acceptable. We also limit MS to under 500 KB. The third model is suitable at 0.455 MB and 0.078 m/s MSE. This approach emphasizes that model selection depends on specific application requirements and constraints.

3.3. Optimal Models Performance Across Datasets

The models and their results presented above are searched using the Gwadar dataset. However, when models with optimal hyper-parameters are trained and then tested on data from two other datasets, Jhimpir and Pasni, they have also shown acceptable performance. Table 1.

Table 1. Optimal Models Performance Across Datasets

Dataset	ϵ	Size (MB)	MSE (m/s)	MAE (m/s)	R^2	Latency (Sec)
Jhimpir	0.01	0.2862	0.0848	0.2180	0.9732	0.0848
	0.25	0.3812	0.0932	0.2284	0.9706	0.0341
	0.50	0.4587	0.0442	0.1573	0.9860	0.1557
	0.75	0.7232	0.0442	0.1573	0.9860	0.1620
	0.99	1.2104	0.0451	0.1583	0.9858	0.6929
Gwadar	0.01	0.2836	0.1065	0.2404	0.9600	0.0915
	0.25	0.3643	0.0568	0.1744	0.9786	0.1992
	0.50	0.4416	0.0522	0.1654	0.9804	0.1773
	0.75	0.7211	0.0498	0.1591	0.9850	0.3022
	0.99	1.2168	0.0470	0.1592	0.9823	0.6224
Pasni	0.01	0.2849	0.5513	0.5620	0.7900	0.0876
	0.25	0.3829	0.6610	0.6160	0.7514	0.0311
	0.50	0.4413	0.5412	0.5892	0.7964	0.1610
	0.75	0.7233	0.3263	0.4346	0.8772	0.2982
	0.99	1.2107	0.2856	0.4065	0.8925	0.6639

3.4. Analyzing the Influence of ϵ on Model Metrics

This section explores the relationship between the parameter ϵ and various model performance indicators, specifically MSE, MAE, R^2 score and MS. Figure 8 presents these relationships graphically, allowing for a comprehensive comparison of how changes in ϵ affect each metric. As it can be seen, a trend emerges showing that an increase in ϵ leads to changes in both MSE and MAE. This shows a direct impact of ϵ on minimizing these error metrics. Furthermore, a positive correlation between ϵ and the R^2 score is also shown, suggesting improved predictive performance with larger ϵ values, while the MS is increasing as the value of ϵ increases.

 doi:10.20944/preprints202502.2066.v1

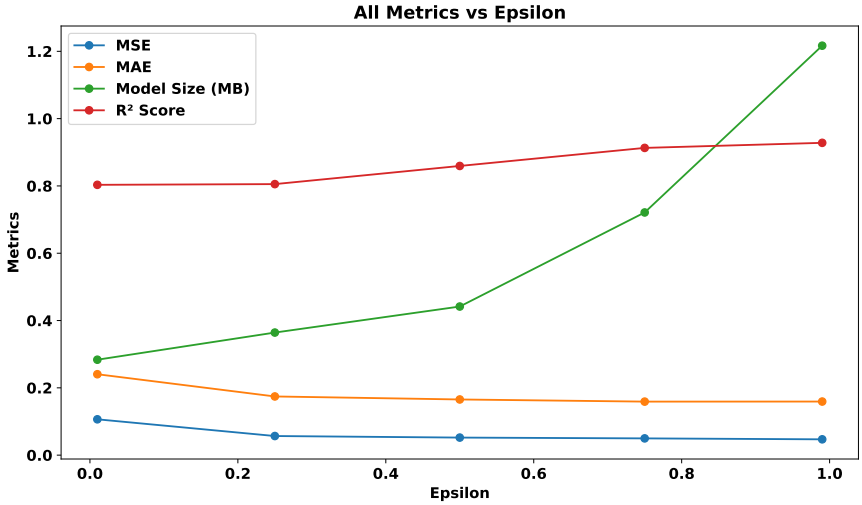

Figure 8. Impact of varying ϵ values on MSE, MAE, MS and R^2 Score

3.5. Comparison

This section compares the proposed optimal model with three existing schemes. Firstly, this work compares with DeepAR [35], an autoregressive recurrent neural network model for probabilistic forecasting with our proposed scheme. The second model for comparison is a CNN-based probabilistic forecasting framework [36]. Lastly, this work investigates the CNN-LSTM model [37], which integrates CNN and LSTM for wind power prediction.

Table 2. Comparison with Existing Schemes

Dataset	Scheme	Size (MB)	MSE (m/s)	MAE (m/s)	R^2
Jhimpir	DeepAR[35]	100.3615	0.0921	0.2321	0.8974
	DeepTCN[36]	15.6832	0.0751	0.2075	0.9332
	CNN-LSTM[37]	12.6971	0.0793	0.2251	0.9162
	Proposed Scheme	**1.2104**	**0.0451**	**0.1583**	**0.9858**
Gwadar	DeepAR[35]	100.3628	0.0903	0.2561	0.9600
	DeepTCN[36]	15.6756	0.0698	0.1945	0.9489
	CNN-LSTM[37]	12.6348	0.0815	0.2678	0.8756
	Proposed Scheme	**1.2168**	**0.0470**	**0.1592**	**0.9823**
Pasni	DeepAR[35]	100.5483	0.4361	0.5620	0.7900
	DeepTCN[36]	15.6540	0.4275	0.4598	0.8219
	CNN-LSTM[37]	12.3256	0.5017	0.5321	0.7545
	Proposed Scheme	**1.2107**	**0.2856**	**0.4065**	**0.8925**

In the comprehensive comparative analysis presented in Table 2, the proposed scheme consistently outperforms existing models like DeepAR, DeepTCN and CNN-LSTM across all datasets. For example, in the Jhimpir dataset, our proposed scheme achieves up to 98.78% reduction in size, 51.89% improvement in MSE, 31.80% improvement in MAE and a 9.80% increase in R^2 score compared to DeepAR. Similar trends are observed in the Gwadar and Pasni datasets, with substantial improvements in all metrics. Specifically, against DeepTCN in Gwadar, the proposed scheme shows a 92.24% reduction in size, 32.66% better MSE, 18.15% better MAE and a 3.52% higher R^2 score. Similarly, in Pasni, against CNN-LSTM, the proposed scheme demonstrates a 90.18% reduction in size, 43.07% improvement in MSE, 23.60% improvement in MAE and an 18.29% increase in R^2 score.

4. Conclusion

In this study, a Hybrid Baseline Model (HBM) was developed, integrating transformers for feature extraction with Long Short-Term Memory (LSTM) networks for time series forecasting, specifically

 doi:10.20944/preprints202502.2066.v1

tailored for Wind Speed Prediction (WSP) on low-power, memory-constrained devices (MCD). The combination of transformers and LSTM networks enhanced the model's ability to extract features and process sequential data, as evidenced by significant reductions in mean squared error (MSE) and mean absolute error (MAE) compared to existing methods. A novel cost function was introduced, effectively managing the trade-offs between prediction accuracy and model size (MS), which is crucial for deployment on devices with limited resources. The results demonstrated up to a 92.24% reduction in model size and a 51.03% improvement in MSE over traditional models like DeepAR, highlighting the effectiveness of the proposed approach. The successful real-time deployment on Jetson Nano devices further confirmed the practical applicability of the models. Training on a server and testing on MCD ensured both accuracy and feasibility for real-world scenarios. Additionally, the use of a genetic algorithm for iterative optimization, based on actual test results, further enhanced the model's performance and adaptability.

Author Contributions: Conceptualization, Laeeq Aslam and Runmin Zou; methodology, Ebrahim Shahzad Awan, Sayyed Shahid Hussain, and Laeeq Aslam; software, Laeeq Aslam; validation, Laeeq Aslam, Ebrahim Shahzad Awan, and Sayyed Shahid Hussain; formal analysis, Laeeq Aslam; investigation, Laeeq Aslam and Muhammad Asim; resources, Laeeq Aslam and Muhammad Asim; writing—original draft preparation, Laeeq Aslam, Mudasir Ahmad Wani, and Kashish Ara Shaki; writing—review and editing, Laeeq Aslam, Runmin Zou, Mudasir Ahmad Wani, and Kashish Ara Shaki; visualization, Laeeq Aslam; supervision, Runmin Zou; project administration, Runmin Zou; funding acquisition,Muhammad Asim and Kashish Ara Shaki . All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R757), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. Also, the authors would like to acknowledge the support of Prince Sultan University.

Data Availability Statement: The data used in this study are sourced from the National Renewable Energy Laboratory (NREL), available at https://www.nrel.gov/.

Acknowledgments: Acknowledgement: The authors would like to acknowledge the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R757), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The Author would also like to acknowledge Prince Sultan University and EIAS Lab for their valuable support.

Conflicts of Interest: Declare conflicts of interest or state "The authors declare no conflicts of interest.".

References

1. Lee, J.; Zhao, F. Global Wind Report 2022, 2022.
2. Brabec, M.; Craciun, A.; Dumitrescu, A. Hybrid numerical models for wind speed forecasting. *Journal of Atmospheric and Solar-Terrestrial Physics* **2021**, *220*, 105669.
3. Moreno, S.; Mariani, V.; dos Santos Coelho, L. Hybrid multi-stage decomposition with parametric model applied to wind speed forecasting in Brazilian northeast. *Renewable Energy* **2021**, *164*, 1508–1526.
4. Nascimento, E.G.S.; de Melo, T.A.; Moreira, D.M. A transformer-based deep neural network with wavelet transform for forecasting wind speed and wind energy. *Energy* **2023**, *278*, 127678.
5. Huang, H.; Chen, J.; Sun, R.; Wang, S. Short-term traffic prediction based on time series decomposition. *Physica A* **2022**, *585*, 126441.
6. Torres, J.; et al. Forecast of hourly average wind speed with ARMA models in Navarre (Spain). *Solar Energy* **2005**, *79*, 65–77.
7. Li, L.; et al. ARMA model-based wind speed prediction for large radio telescope. *Acta Astronomica Sinica* **2022**, *63*, 70.
8. Moreno, S.; dos Santos Coelho, L. Wind speed forecasting approach based on singular spectrum analysis and adaptive neuro fuzzy inference system. *Renewable Energy* **2018**, *126*, 736–754.
9. Bechrakis, D.; Sparis, P. Wind speed prediction using artificial neural networks. *Wind Engineering* **1998**, pp. 287–295.
10. Mohandes, M.A.; Halawani, T.O.; Rehman, S.; Hussain, A.A. Support vector machines for wind speed prediction. *Renewable energy* **2004**, *29*, 939–947.

 doi:10.20944/preprints202502.2066.v1

11. Geng, D.; Zhang, H.; Wu, H. Short-term wind speed prediction based on principal component analysis and LSTM. *Applied Sciences* **2020**, *10*, 4416.
12. Cai, R.; Xie, S.; Wang, B.; Yang, R.; Xu, D.; He, Y. Wind speed forecasting based on extreme gradient boosting. *IEEE Access* **2020**, *8*, 175063–175069.
13. Aslam, L.; Zou, R.; Awan, E.; Butt, S.A. Integrating Physics-Informed Vectors for Improved Wind Speed Forecasting with Neural Networks. In Proceedings of the Asian Control Conference, July 2024.
14. Liu, H.; Tian, H.; Li, Y. Comparison of two new ARIMA-ANN and ARIMA-Kalman hybrid methods for wind speed prediction. *Applied Energy* **2012**, *98*, 415–424.
15. Zhang, Y.; Le, J.; Liao, X.; Zheng, F.; Li, Y. A novel combination forecasting model for wind power integrating least square support vector machine, deep belief network, singular spectrum analysis and locality-sensitive hashing. *Energy* **2019**, *168*, 558–572.
16. Wan, A.; Chang, Q.; et al. Short-term power load forecasting for combined heat and power using CNN-LSTM enhanced by attention mechanism. *Energy* **2023**, *282*, 128274.
17. Pan, S.; et al. Oil well production prediction based on CNN-LSTM model with self-attention mechanism. *Energy* **2023**, *284*, 128701.
18. Li, W.; Li, Y.; Garg, A.; Gao, L. Enhancing real-time degradation prediction of lithium-ion battery: A digital twin framework with CNN-LSTM-attention model. *Energy* **2024**, *286*, 129681.
19. Ata Teneler, A.; Hassoy, H. Health effects of wind turbines: a review of the literature between 2010-2020. *International journal of environmental health research* **2023**, *33*, 143–157.
20. Gkeka-Serpetsidaki, P.; Papadopoulos, S.; Tsoutsos, T. Assessment of the visual impact of offshore wind farms. *Renewable Energy* **2022**, *190*, 358–370.
21. Bilgili, M.; Alphan, H. Visual impact and potential visibility assessment of wind turbines installed in Turkey. *Gazi University Journal of Science* **2022**, *35*, 198–217.
22. Lehnardt, Y.; Barber, J.R.; Berger-Tal, O. Effects of wind turbine noise on songbird behavior during nonbreeding season. *Conservation Biology* **2024**, *38*, e14188.
23. Teff-Seker, Y.; Berger-Tal, O.; Lehnardt, Y.; Teschner, N. Noise pollution from wind turbines and its effects on wildlife: A cross-national analysis of current policies and planning regulations. *Renewable and Sustainable Energy Reviews* **2022**, *168*, 112801.
24. Zolotoff-Pallais, J.M.; Perez, A.M.; Donaire, R.M. General Comparative Analysis of Bird-Bat Collisions at a Wind Power Plant in the Department of Rivas, Nicaragua, between 2014 and 2022. *European Journal of Biology and Biotechnology* **2024**, *5*, 1–7.
25. Richardson, S.M.; Lintott, P.R.; Hosken, D.J.; Economou, T.; Mathews, F. Peaks in bat activity at turbines and the implications for mitigating the impact of wind energy developments on bats. *Scientific Reports* **2021**, *11*, 3636.
26. Choi, D.Y.; Wittig, T.W.; Kluever, B.M. An evaluation of bird and bat mortality at wind turbines in the Northeastern United States. *PLoS One* **2020**, *15*, e0238034.
27. Barter, G.E.; Sethuraman, L.; Bortolotti, P.; Keller, J.; Torrey, D.A. Beyond 15 MW: A cost of energy perspective on the next generation of drivetrain technologies for offshore wind turbines. *Applied Energy* **2023**, *344*, 121272.
28. Wiser, R.; Millstein, D.; Bolinger, M.; Jeong, S.; Mills, A. The hidden value of large-rotor, tall-tower wind turbines in the United States. *Wind Engineering* **2021**, *45*, 857–871.
29. Turc Castellà, F.X. Operations and maintenance costs for offshore wind farm. Analysis and strategies to reduce O&M costs. Master's thesis, Universitat Politècnica de Catalunya, 2020.
30. Abdelateef Mostafa, M.; El-Hay, E.A.; ELkholy, M.M. Recent trends in wind energy conversion system with grid integration based on soft computing methods: comprehensive review, comparisons and insights. *Archives of Computational Methods in Engineering* **2023**, *30*, 1439–1478.
31. Ahmed, S.D.; Al-Ismail, F.S.; Shafiullah, M.; Al-Sulaiman, F.A.; El-Amin, I.M. Grid integration challenges of wind energy: A review. *Ieee Access* **2020**, *8*, 10857–10878.
32. Suo, L.; Peng, T.; Song, S.; Zhang, C.; Wang, Y.; Fu, Y.; Nazir, M.S. 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.
33. Saini, V.K.; Kumar, R.; Al-Sumaiti, A.S.; Sujil, A.; Heydarian-Forushani, E. Learning based short term wind speed forecasting models for smart grid applications: An extensive review and case study. *Electric Power Systems Research* **2023**, *222*, 109502.
34. Zhang, Y.; Pan, G.; Chen, B.; Han, J.; Zhao, Y.; Zhang, C. Short-term wind speed prediction model based on GA-ANN improved by VMD. *Renewable energy* **2020**, *156*, 1373–1388.

 doi:10.20944/preprints202502.2066.v1

35. Salinas, D.; Flunkert, V.; Gasthaus, J.; Januschowski, T. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* **2020**, *36*, 81–91.
36. Chen, Y.; Kang, Y.; Chen, Y.; Wang, Z. Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing* **2020**, *399*, 491–501.
37. Wu, Q.; Guan, F.; Lv, C.; Huang, Y. Ultra-short-term multi-step wind power forecasting based on CNN-LSTM. *IET Renewable Power Generation* **2021**, *15*, 1019–1029.