IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 May 2025, accepted 15 June 2025, date of publication 23 June 2025, date of current version 9 July 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3581973

RESEARCH ARTICLE

Dynamic Optimization of Recurrent Networks for Wind Speed Prediction on Edge Devices

LAEEQ ASLAM[1], (Graduate Student Member, IEEE), RUNMIN ZOU[2], (Senior Member, IEEE), EBRAHIM SHAHZAD AWAN[3], SAYYED SHAHID HUSSAIN[1], MUHAMMAD ASIM[4], SAMIA ALLAOUA CHELLOUG[5], AND MOHAMMED A. ELAFFENDI[4]

[1]School of Automation, Central South University, Changsha, Hunan 410083, China
[2]School of Automation, Institute of Smart Energy and Advanced Control, Central South University, Changsha, Hunan 410083, China
[3]School of Engineering, Design, and Built Environment, Western Sydney University, Penrith, NSW 2747, Australia
[4]EIAS Data Science Laboratory, College of Computer and Information Sciences, Center of Excellence in Quantum and Intelligent Computing, Prince Sultan University, Riyadh 11586, Saudi Arabia
[5]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

Corresponding authors: Runmin Zou (rmzou@csu.edu.cn) and Samia Allaoua Chelloug (SAChelloug@pnu.edu.sa)

This work was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R239), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

ABSTRACT Accurate wind speed prediction (WSP) remains essential for optimizing energy management in small-scale domestic windmills. Server-dependent machine learning models, commonly deployed in wind farms, prove infeasible for domestic systems due to high costs and energy demands. While edge computing offers a viable alternative, current WSP methods prioritize hyperparameter optimization without constraining model size (MS), resulting in memory-intensive architectures incompatible with resource-limited devices. To address this gap, we propose a framework that co-optimizes the discrete hyperparameter spaces of Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) and Temporal Convolutional Network (TCN) models under strict memory constraints. An adaptive Simulated Annealing algorithm with memory-based rejection (aSAR) navigates the discrete design space, employing nine objective functions that balance Mean Absolute Percentage Error (MAPE) against model compactness. Evaluations on wind datasets from Chile, Kazakhstan and Mongolia demonstrate that aSAR-optimized models reduce prediction errors by up to 54.17% and decrease MS by 98.75% relative to state-of-the-art techniques. The results highlight significant regional performance variations, underscoring the necessity of location-specific architecture selection. This work establishes a systematic approach for deploying memory-efficient WSP models on edge devices, advancing sustainable energy solutions for decentralized wind power systems.

INDEX TERMS Adaptive simulated annealing, deep learning, edge devices, hyperparameter optimization, renewable energy, wind speed prediction.

I. INTRODUCTION

Wind energy is an important renewable energy source, with the global installed capacity reaching 837 gigawatts (GW) in 2021, showing an annual growth rate of 17% [1], [2]. Wind energy works by turning the kinetic energy of wind into mechanical power or electricity, which can be used to power homes, businesses and industries. Unlike fossil fuels, wind energy does not produce greenhouse gases or other pollutants during operation, making it a key part of the shift to clean energy and reducing climate change. Accurate prediction of wind speed(WS) is crucial for the effective management of wind energy systems. The efficiency and reliability of wind power generation depend directly on WSP because the energy produced by wind turbines is proportional to the cube of the WS [3]. Therefore, precise WSP is essential for optimizing grid stability, improving the scheduling of energy production and reducing reliance on fossil fuels [4], [5]. Moreover,

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

adding wind energy in grids adds more sustainable energy and helps in reducing carbon footprint of electricity generation.

A. EXISTING RESEARCH

Methods for WSP can be categorized into physical models, statistical models, machine learning (ML) models and hybrid models. Physical models, such as Numerical Weather Prediction (NWP) models, rely on atmospheric data and meteorological readings for WSP. These models are suitable for long-term predictions but require accurate boundary and initial conditions [6], [7]. However, their dependency on parameterization schemes and the accuracy of initial conditions limits their effectiveness for short-term forecasts [8], [9]. Further, other physical models like the Mesoscale Model 5 and the Weather Research and Forecasting model, which use numerical simulations, excel in long-term predictions but are computationally intensive, making them less suitable for short-term WSP [10], [11]. In contrast, statistical methods use historical wind speed data along with environmental features such as temperature and pressure for prediction. Common models include autoregressive moving average (ARMA) and integrated autoregressive moving average (ARIMA), which identify trends in historical data [12], [13]. ARMA and ARIMA models, due to their use of auto-correlation, have gained significant attention in WSP prediction, with studies showing that ARIMA models can outperform neural networks [14]. Additionally, ARIMA-ARCH and ARMA models have been proposed to address heteroskedasticity, a common issue in wind speed prediction(WSP) due to seasonal wind patterns and complex weather interactions [15]. However, multiple studies [16], [17] have shown that, while statistical models outperform other methods for long-term WSP, they struggle with short-term fluctuations and nonlinear effects typical of wind speed, making them less effective for unstable short-term WSP.

Machine learning (ML) techniques such as artificial neural networks (ANNs), support vector machines (SVMs) and deep learning architectures gain attention by effectively modeling these nonlinear relationships without boundary condition assumptions [18], [19]. For example, Moreno et al. [20] use ANNs' ability to uncover nonlinear patterns in wind speed data, while Mohandes et al. [19] demonstrate SVMs' robustness and predictive accuracy. More recently, multiple studies optimize long short-term memory (LSTM) networks to capture long-term temporal dependencies, surpassing traditional approaches like support vector regression [21], [22], [23]. Additionally, methods such as Extreme Gradient Boosting (XGBoost) prove effective for handling large datasets and mitigating overfitting through regularization [24]. Physics-informed ML approaches further enhance prediction precision by integrating spatial features and custom loss functions [25].

Recent research explores the combination of different model architectures to take advantage of their complementary strengths, resulting in more powerful and robust WSP solutions. These hybrid models integrate traditional statistical methods with machine learning algorithms to enhance both accuracy and reliability. For example, Liu et al. [26] demonstrate that coupling ARIMA with ANNs or SVMs yields superior predictive performance. Building on this, later studies incorporate advanced preprocessing techniques such as Singular Spectrum Analysis (SSA) and wavelet transforms to further improve prediction quality [27]. Moving beyond classical hybrids, researchers combine multiple deep learning models, for instance, convolutional neural networks (CNNs) with long- and short-term memory (LSTM) networks to effectively capture spatial and temporal features in wind speed data [28], [29]. More recently, transformer models enhanced with feature selection techniques apply to WSP, utilizing attention mechanisms to prioritize the most relevant input sequences and thereby improve forecasting accuracy [30], [31], [32].

B. LIMITATIONS OF EXISTING WORK

Current WSP research focuses primarily on optimizing deep learning models [21], [22], [23], [33], [34] or hybrid architectures [35], [36] to improve prediction accuracy. These methods assume deployment on servers with sufficient memory to store model checkpoints and perform inference. Such online servers suit large commercial wind farms, which justify the bandwidth and infrastructure costs. However, these farms face challenges aligned with United Nations Sustainable Development Goals, including noise pollution [37], [38], operational expenses [39], [40], [41], visual impact [42], [43], [44], and grid integration difficulties [45], [46]. Small-scale turbines reduce these challenges by operating independently, requiring minimal grid connection and incurring lower costs. Deploying server-dependent WSP models on small-scale systems reduces their economic viability, as remote inference demands costly infrastructure. This situation creates a need for WSP models optimized jointly for accuracy and compactness, enabling deployment on memory-limited edge devices such as the Jetson Nano.

Recent work proposed by aslam at el. [47] partially addresses this need by proposing a hybrid model optimized for size and accuracy via linear objective functions. Two key limitations remain. First, a single baseline model is designed is optimized for a single site but applied universally, overlooking spatial variability caused by topography, seasonal changes, pressure gradients and mesoscale dynamics. Unlike stable data domains(solar irradiance), wind patterns vary widely across regions and no single architecture performs well everywhere. Second, scalability issues arise under high-variability conditions. Although the hybrid model fits edge constraints on low-variability data, its complexity grows substantially in complex environments such as mountainous or coastal areas, exceeding edge device memory. Therefore, effective small-scale WSP deployment requires location-specific model architectures optimized within strict memory limits.

C. CONTRIBUTIONS

To address these limitations, this work proposes a location-specific optimization of fundamental time series prediction architectures, including recurrent models (LSTM, GRU) and temporal models Temporal Convolutional Network (TCN). It shows that comprehensive optimization of these models, under size constraints, can result in models that are not only smaller in size, but also outperform larger models for specific locations. In this research, the discrete hyperparameter space of LSTM, GRU and TCN models is optimized specifically for memory-constrained devices using aSAR. The main contributions of this research are as follows:

- This work optimizes the discrete hyper-parameter space of LSTM, GRU and TCN models for WSP on memory-constrained devices, using nine objective functions, including exponential scaling, linear weighted combination, inverse scaling, quadratic penalty, harmonic mean, logarithmic combination, product of inverses, focal loss-inspired and combined log-linear objective functions.
- The aSAR algorithm is proposed that optimizes hyperparameters, model architectures and feature selection for WSP. It dynamically adjusts the number of key features using the Pearson correlation coefficient and determines the optimal input time stamps for the dataset. Additionally, aSAR incorporates a memory-based rejection mechanism, which tracks previously rejected solutions to prevent revisiting suboptimal regions of the solution space. This memory mechanism ensures more efficient exploration by adding a penalty for solutions that are too similar to those already rejected. By applying aSAR to wind speed datasets from Kazakhstan, Mongolia and Chile, the algorithm improves prediction accuracy and efficiency, addressing challenges such as computational constraints and model complexity in small-scale applications.
- The top three models from each objective function are first extensively trained using the datasets and then evaluated using the Analytic Hierarchy Process (AHP), which ranks them based on MAP (MAPE) and MS. The top models for each dataset are identified as aSAR-TCN, aSAR-LSTM and aSAR-LSTM, collectively referred to as aSAR-Models.Finally the best architecture among these models can be deployed on memory constrained devices.

The rest of the paper is structured as follows: Section II discusses the proposed methodology, Section III discusses the results of the proposed method and Section IV concludes the overall work.

II. PROPOSED METHODOLOGY

This work proposes a methodology for optimizing predictive models in WSP by integrating model size (MS) and Mean Absolute Percentage Error (MAPE) into nine objective functions. These functions optimize three baseline models, Temporal Convolutional Networks (TCNs), Long Short-Term Memory networks (LSTMs) and Gated Recurrent Units (GRUs), balancing accuracy and computational efficiency. These models were selected for their complementary strengths in temporal modeling, TCNs capture long-range dependencies via dilated convolutions, while LSTMs/GRUs excel at learning sequential patterns through gated mechanisms, both critical for wind speed's time-varying dynamics. The optimized models are evaluated and ranked using the Analytic Hierarchy Process (AHP), which compares their performance based on accuracy and complexity. The best-performing models are analyzed to identify the most suitable architecture for each location, selecting the model and hyperparameters that best fit the dataset.The hyperparameter space is discrete, as parameters are either categorical (e.g., activation functions) or integer-valued (e.g., layer count, units per layer), disallowing continuous optimization techniques such as use of KKT conditions with gradient-based methods. Although the adaptive estimator method proposed by Tutsoy et al. [48] can be used for hyperparameter learning by framing it as an adaptive control problem, its practical use is limited. This is because it assumes linear system dynamics and relies on temporal difference learning, which leads to slow convergence and poor conditioning in the discrete hyperparameter space. Therefore, heuristic or metaheuristic algorithms are necessary to explore this discrete space efficiently. Hence, this work proposes use of the aSAR algorithm to optimize this discrete hyperparameter space. The methodology aims to outperform traditional time-series prediction models by prioritizing computational efficiency through compact architectures and optimized hyperparameters. The overall approach is illustrated in Figure 1.

A. OPTIMIZATION OBJECTIVES

The primary goal is to explore the discrete hyper-parameter space of LSTM, GRU and TCN models to enhance prediction accuracy while minimizing MS. We introduce nine objective functions that balance these criteria. Before describing them, we define the accuracy metric as MAPE, which evaluates a model's predictive error in relative terms and is defined by

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \times 100, \tag{1}$$

where Y_i is the actual wind speed at time i, $\hat{Y}_i$ is the forecasted wind speed and N is the number of observations. Because MAPE divides the absolute error by the actual value, it is scale-independent and can be used across datasets of different magnitudes. It remains interpretable as it reflects proportional error instead of absolute difference. Its reliance on absolute values, rather than squared terms, makes it less sensitive to large outliers, although very small Y_i values can inflate its ratio.

We propose nine objective functions that combine MAPE and MS in distinct ways. The parameter ϵ often appears as a control knob to adjust the influence of MS relative to MAPE,

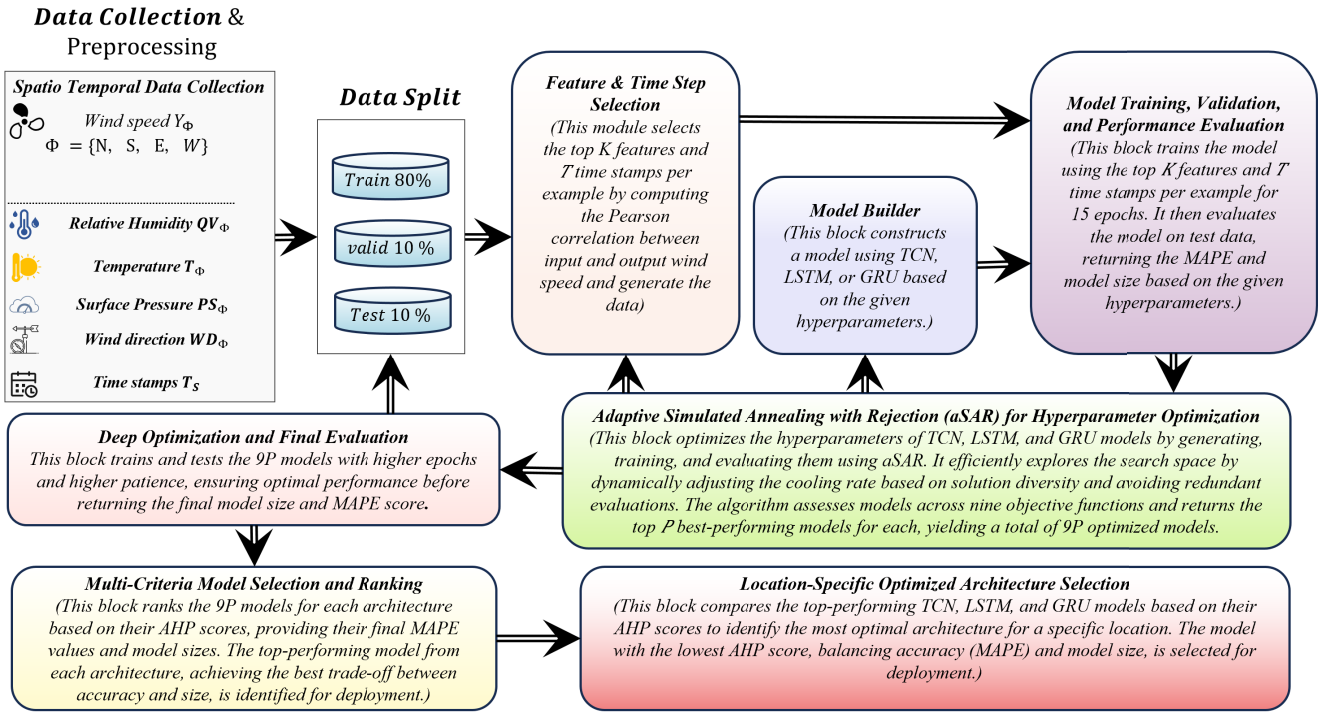

FIGURE 1. Overall block diagram of the proposed methodology.

while some formulations include additional parameters that further shape the trade-off. The first objective function, the Linear Weighted Combination (LWC), is given by

$$f_{\text{LWC}}(Y, \hat{Y}, MS) = (1-\epsilon)\,\text{MAPE} \; - \; \epsilon \; \ln(\text{MS}), \quad (2)$$

and adds a log-based penalty for MS to MAPE in a linear fashion. The parameter ϵ splits attention between accuracy and MS, with larger ϵ penalizing size more heavily. LWC keeps these terms additive, providing a straightforward trade-off, although the subtraction of $\epsilon \ln(\text{MS})$ can sometimes produce abrupt changes when balancing improvements in MAPE against the size penalty.

Next, the Inverse Scaling (IS) function,

$$f_{\text{IS}}(Y, \hat{Y}, MS) = \frac{\text{MAPE}}{1 + \epsilon \; \ln(\text{MS})}, \quad (3)$$

divides MAPE by $1 + \epsilon \ln(\text{MS})$. This setup restrains overly large architectures because the denominator increases gradually with ln(MS), smoothing out the penalty for moderate expansions. Unlike a linear subtraction, embedding the size term in the denominator can enforce more continuous and incremental penalization of complexity.

The Logarithmic Combination (LC) function,

$$f_{\text{LC}}(Y, \hat{Y}, MS) = \ln\big(1 + \text{MAPE}\big) \; - \; \epsilon \; \ln\big(1 + \ln(\text{MS})\big), \quad (4)$$

applies a logarithmic transform to both MAPE and MS. This moderates the effect of large values since both the error term and MS are smoothed inside log operations. Large jumps in MAPE or MS thus do not immediately cause extreme changes in the objective, providing a gentler constraint on expansion than ratio- or difference-based methods.

Meanwhile, the Combined Log and Linear (CLL) function,

$$f_{\text{CLL}}(Y, \hat{Y}, MS) = \ln\big(1 + \text{MAPE}\big) \; - \; \epsilon \, \text{MS}, \quad (5)$$

applies a logarithmic term to MAPE but penalizes MS linearly. This can be more aggressive than using ln(MS) if ϵ is large, because the subtraction by ϵ MS grows quickly with model complexity. Unlike LC, CLL can therefore offer a stronger disincentive for large architectures, though it may over-penalize size in certain contexts if ϵ is not finely tuned.

Another formulation is the Harmonic Mean (HM) function,

$$f_{\text{HM}}(Y, \hat{Y}, MS) = \frac{2\,\text{MAPE}}{\text{MAPE} + \left(\frac{1}{\ln(\text{MS})}\right)}, \quad (6)$$

which weaves MAPE and an inverse log of MS into a harmonic mean. This structure encourages both terms to remain small since the harmonic mean is dominated by larger denominators. In practice, it can push models toward balancing improved accuracy and reasonable size, rather than excelling at one objective at the cost of the other.

The Quadratic Penalty (QP) function,

$$f_{\text{QP}}(Y, \hat{Y}, MS) = \text{MAPE} - \epsilon \left(\ln(\text{MS})\right)^2, \quad (7)$$

imposes a squared-log penalty on MS. Relative to a direct ln(MS) term, $\left(\ln(\text{MS})\right)^2$ grows more quickly once MS becomes large, creating a sharper incentive to keep models smaller. However, ϵ must be chosen carefully to avoid ignoring accuracy altogether.

An exponential approach is taken by the Exponential Scaling (ES) function,

$$f_{\mathrm{ES}}(Y, \hat{Y}, MS) = \mathrm{MAPE} \cdot \mathrm{MS}^{-\epsilon}, \tag{8}$$

Because $\mathrm{MS}^{-\epsilon}$ quickly decreases the objective as MS grows, strongly disfavoring large architectures. Compared with a ratio-based penalty, ES compresses MAPE through multiplication and tuning ϵ adjusts how severely it dampens the score for larger models.

An alternative multiplicative structure arises in the Product of Inverses (PI) function,

$$f_{\mathrm{PI}}(Y, \hat{Y}, MS) = \mathrm{MAPE} \cdot \frac{1-\epsilon}{\ln(\mathrm{MS})}, \tag{9}$$

Here, a reciprocal ln(MS) factor heavily penalizes growth in MS. A slight increase in MS causes ln(MS) to rise, shrinking its reciprocal and thereby reducing the score. This can be stricter than adding or subtracting a long term, though its exact impact depends on how quickly ln(MS) grows and on ϵ.

Finally, the Focal Loss Inspired (FLI) function,

$$f_{\mathrm{FLI}}(Y, \hat{Y}, MS) = \mathrm{MAPE}^{\gamma}\,(1-\epsilon)\; -\; \epsilon\,\big(\ln(\mathrm{MS})\big)^{\alpha}, \tag{10}$$

incorporates exponents γ and α to further shape how each term behaves. Raising MAPE to γ can either concentrate on reducing moderate errors or amplify smaller ones, while $(\ln(\mathrm{MS}))^{\alpha}$ targets the size penalty. This extra flexibility requires more tuning but allows one to place highly customized emphasis on accuracy versus complexity.

These nine functions offer varied strategies for balancing model accuracy and MS in neural network hyper-parameter optimization. Each has parameters that must be adjusted according to the specific goals of a project, whether that is prioritizing minimal error, reducing complexity, or achieving a practical compromise between the two. By selecting and tuning an appropriate objective function, one can guide the learning process toward models that efficiently capture patterns while remaining suitably lightweight.

B. DISCRETE HYPER-PARAMETER SPACE

This section describes the discrete hyper-parameter space for LSTM, GRU and TCN models. The goal is to optimize model performance by exploring hyper-parameters, including architecture and input parameters. Since the recurrent models, including LSTM and GRU, share almost the same hyperparameters, they are discussed in a separate subsection, while TCN, which has different hyperparameters, is addressed in another subsection. The input feature section is presented separately.

1) DISCRETE HYPER-PARAMETER SPACE FOR RECURRENT MODELS

In LSTM models, the architecture consists of three gates: forget, input and output. The forget gate determines which information to discard:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big), \tag{11}$$

where σ is the sigmoid function, h_{t-1} is the previous hidden state and x_t is the current input. The input gate regulates updates to the cell state:

$$i_t = \sigma\big(W_i \cdot [h_{t-1}, x_t] + b_i\big), \tag{12}$$

and the candidate cell state is computed as:

$$\tilde{C}_t = \tanh\big(W_C \cdot [h_{t-1}, x_t] + b_C\big). \tag{13}$$

The cell state updates as:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t, \tag{14}$$

and the output gate determines the next hidden state:

$$o_t = \sigma\big(W_o \cdot [h_{t-1}, x_t] + b_o\big), \tag{15}$$

yielding:

$$h_t = o_t \cdot \tanh\big(C_t\big). \tag{16}$$

The GRU model simplifies the LSTM by merging the forget and input gates into an update gate and using a reset gate. The update gate is defined as:

$$z_t = \sigma\big(W_z \cdot [h_{t-1}, x_t] + b_z\big), \tag{17}$$

and the reset gate as:

$$r_t = \sigma\big(W_r \cdot [h_{t-1}, x_t] + b_r\big). \tag{18}$$

Key hyperparameters for LSTM and GRU models include the number of layers (N), units per layer, dropout rate (D) and activation function (A). The number of layers controls network depth, affecting its ability to learn complex patterns. More layers may improve performance but increase MS and computational cost. Similarly, more units per layer enhance performance but raise MS and training time. The dropout rate prevents overfitting by deactivating neurons during training. Higher dropout rates reduce overfitting but may hinder learning, while lower rates risk overfitting. The activation function introduces non-linearity, with choices like ReLU, tanh, or sigmoid impacting performance and training efficiency.

2) DISCRETE HYPER-PARAMETER SPACE FOR TEMPORAL CONVOLUTIONAL MODELS

The TCN model uses temporal convolutions with hyperparameters such as the number of filters F, kernel size K_s and dilation factors Δ. A dilated convolution is defined as:

$$y_t = \sum_{i=0}^{k-1} w_i \cdot x_{t-d \cdot i}. \tag{19}$$

Increasing filters or kernel size enlarges the receptive field, improving temporal dependency capture but increasing MS. Dilations and padding expand the receptive field without linearly increasing parameters, balancing complexity and performance.

3) INPUT FEATURE SELECTION

Since each model receives spatio-temporal input for a given terrain, certain input features, such as temperature from the north, may have a greater impact on wind speed compared to features from other directions. However, this effect depends on the terrain. Hence, while optimizing parameters, this work dynamically adjusts K and selects the top K input features. These features are chosen based on their Pearson correlation with the target variable (wind speed):

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}, \tag{20}$$

where X_i and Y_i are sample points and $\bar{X}$ and $\bar{Y}$ are their means. Selecting strongly correlated features improves accuracy and reduces complexity. A smaller K reduces computational cost, while a larger K may improve accuracy at the expense of complexity.

The sequence length T determines the temporal span the model learns from. Longer sequences capture more dependencies but increase computational burden, while shorter sequences may miss long-term patterns. Optimizing T balances temporal depth and efficiency. Table 1 summarizes the constraints and ranges for each hyper-parameter, guiding optimization to improve performance and efficiency.

C. ADAPTIVE SIMULATED ANNEALING WITH MEMORY-BASED REJECTION MECHANISM

Simulated Annealing with Rejection (SAR) is an optimization technique and can be used for optimal hyperparameter search. The algorithm starts with an initial high-temperature T_0 to facilitate exploration of the solution space. At each iteration, a new solution is generated by perturbing the current solution. If the new solution has a lower cost, it is accepted; otherwise, it is accepted with a probability given by:

$$P(\Delta E, T) = \exp\left(\frac{\Delta E}{T}\right) \tag{21}$$

where $\Delta E = \text{current\_cost} - \text{new\_cost}$ is the difference in cost between the current and the new solutions. The temperature T is gradually decreased according to a cooling schedule:

$$T = T_0 \times \alpha^k \tag{22}$$

where α is the cooling rate and k is the iteration number. The algorithm continues until the temperature reaches a stopping threshold T_{stop} or after a set number of iterations N, as indicated by the termination condition:

$$T \leq T_{\text{stop}} \quad \text{or} \quad \text{iteration} \geq N \tag{23}$$

This structure allows SAR to balance exploration and exploitation, accepting worse solutions early in the process to escape local minima and refining solutions as the temperature decreases. However, SAR suffers from premature convergence, especially when the solutions become concentrated around local optima, leading to inefficient exploration. Additionally, the fixed cooling schedule does not adjust based on the solution diversity, which may hinder further exploration when the search space still has unexplored regions. These issues motivate the need for a more adaptive approach.

To overcome this challenge, this work proposes an adaptive SAR (aSAR) algorithm in which the cooling rate adjusts dynamically based on the diversity of the current solutions. Specifically, the temperature update is adjusted using the fitness variance σ_k^2 of the solutions at iteration k, which is defined as:

$$\sigma_k^2 = \frac{1}{N}\sum_{i=1}^{N}(f_i - \bar{f})^2 \tag{24}$$

where f_i is the fitness of the i-th solution, $\bar{f}$ is the average fitness and N is the number of solutions. When the fitness variance is large, the solutions are diverse and the algorithm maintains a higher temperature to encourage exploration. As the variance decreases and the solutions converge, the temperature decreases more rapidly to focus on exploitation.

The cooling rate modifies as:

$$\alpha_k = \alpha_0 \times \left(1 + \beta \times \sigma_k^2\right) \tag{25}$$

where α_0 is the base cooling rate and β is a hyperparameter controlling the influence of the fitness variance. The temperature update equation becomes:

$$T_{k+1} = T_k \times \alpha_k \tag{26}$$

The adaptive cooling rate α_k is determined by the fitness variance σ_k^2, which quantifies the diversity of solutions. When σ_k^2 is large (indicating diverse solutions), α_k increases proportionally to $\beta\sigma_k^2$, slowing the temperature decay (T_k) to prolong exploration. Conversely, when σ_k^2 decreases (solutions converge), α_k approaches α_0, accelerating cooling to prioritize exploitation. This feedback loop ensures the algorithm self-adjusts to the solution landscape. The hyperparameters β, λ, and $D_{\max}$ are tuned empirically: β scales the influence of σ_k^2 on α_k and is normalized by the fitness range ($\beta = 1/(\max(f) - \min(f))$) during initialization to ensure stability. The maximum similarity threshold $D_{\max}$ is set as a fraction (e.g., 10%) of the search space diameter, while the penalty strength λ is initialized small (e.g., $\lambda = 0.1$) and adjusted via grid search to balance exploration and rejection penalties. This parameterization ensures robustness across problems while retaining the adaptive benefits of aSAR.

Additionally, aSAR introduces a memory-based rejection sampling mechanism that tracks previously rejected solutions. This mechanism prevents the algorithm from revisiting suboptimal regions by adding a penalty for solutions too similar to those already rejected. The Euclidean distance $D(\theta, \theta_{\text{rej}})$ between the current solution θ and a rejected solution θ_{rej} measures the similarity. The acceptance probability is modified as:

$$P_{\text{accept}} = \exp\left(\frac{\Delta E}{T}\right) \times \left(1 - \lambda \times \min\left(1, \frac{D(\theta, \theta_{\text{rej}})}{D_{\max}}\right)\right) \tag{27}$$

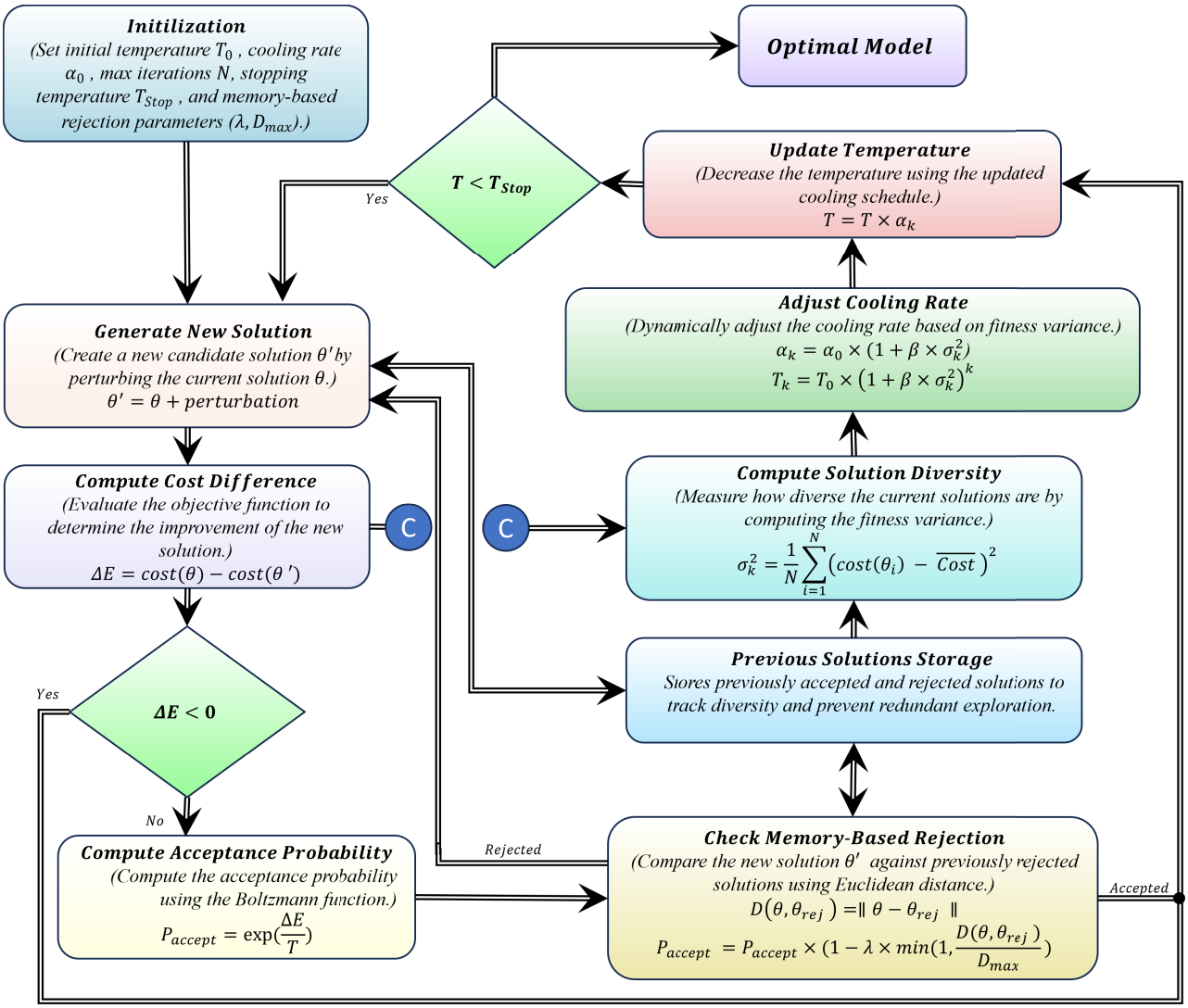

FIGURE 2. Block diagram of the aSAR optimization method.

where λ is a hyperparameter that regulates the strength of the penalty, while $D_{\max}$ sets a threshold defining the maximum permissible similarity for rejection. If the distance between the current solution and a previously rejected solution is small, the penalty term increases, reducing the likelihood of acceptance. This approach prevents the algorithm from revisiting similar solutions, thereby enhancing the efficiency of solution space exploration. The overall block diagram of the aSAR optimization method is illustrated in Figure 2.

The modifications introduced in aSAR, particularly the adaptive cooling schedule and memory-based rejection, address the issues of premature convergence and redundant exploration present in traditional aSAR. By adjusting the cooling rate based on solution diversity and preventing the algorithm from revisiting previously explored suboptimal regions, aSAR ensures more efficient exploration and exploitation of the search space.

D. MODEL EVALUATION AND RANKING

The multi-criteria selection process operates through two complementary mechanisms. The first identifies Pareto-optimal solutions using nine competing objective functions during aSAR optimization for each architecture (TCN, LSTM and GRU). The second performs a final ranking via the AHP. Algorithm 1 formalizes this dual-stage approach as follows:

$$\{f_1, \ldots, f_9\} \quad \rightarrow \quad \text{AHP Ranking} \tag{28}$$

where the left-hand side represents multi-criteria optimization and the right-hand side denotes decision making for the best architecture. Each objective function f_j (defined in Equations 2–10) expresses a distinct trade-off between accuracy, measured by MAPE and efficiency, measured by MS. The aSAR algorithm (Algorithm 2) explores this nine-dimensional criterion space via adaptive simulated

TABLE 1. Summary of hyper-parameters and constraints for LSTM, GRU and TCN models.

Hyper-parameter	LSTM	GRU	TCN
Number of Layers (N)	$N_i \in [N_{min}, N_{max}]$	$N_i \in [N_{min}, N_{max}]$	$N_i \in [N_{min}, N_{max}]$
Units (U)	$U_i \in [U_{min}, U_{max}]$	$U_i \in [U_{min}, U_{max}]$	–
Dropout Rate (D)	$D \in [D_{min}, D_{max}]$	$D \in [D_{min}, D_{max}]$	$D \in [D_{min}, D_{max}]$
Activation (A)	$A \in \{A_1, A_2, A_3\}$	$A \in \{A_1, A_2, A_3\}$	$A \in \{A_1, A_2, A_3\}$
K Features (K)	$K \in [K_{min}, K_{max}]$	$K \in [K_{min}, K_{max}]$	–
Time Stamp (T)	$T \in [T_{min}, T_{max}]$	$T \in [T_{min}, T_{max}]$	–
Number of Filters (F)	–	–	$F \in [F_{min}, F_{max}]$
Kernel Size (K_s)	–	–	$K_s \in [K_{s,min}, K_{s,max}]$
Number of Stacks (S)	–	–	$S \in [S_{min}, S_{max}]$
Dilations (Δ)	–	–	$\Delta \in \{\Delta_1, \Delta_2, \ldots, \Delta_n\}$
Padding (P)	–	–	$P \in \{\text{causal, same}\}$

annealing, generating P Pareto-optimal candidates per objective function.

Following optimization of the hyperparameters for each model using nine objective functions, we obtain P top-performing models for each objective function, resulting in a total of $9P$ models. These models are trained on the complete dataset and their MAPE and MS are recomputed. MAPE is defined as the mean of the absolute percentage differences between the predicted and actual values, indicating the model's prediction accuracy. To select the best models from this set, we apply the AHP, a method used to compare multiple criteria. In this case, we compare MAPE and MS for each model. The first step involves normalizing the MAPE and MS values to bring them onto a common scale. Normalization is done using the formula:

$$\text{Normalized Value} = \frac{\text{Original Value} - \text{Min Value}}{\text{Max Value} - \text{Min Value}} + \epsilon \tag{29}$$

where ϵ is a small constant added to avoid division by zero. Next, we create two pairwise comparison matrices: one for MAPE and one for MS. Each matrix has size $9P \times 9P$, where $9P$ represents the total number of models. For the MAPE matrix $\mathbf{A}_{\text{MAPE}}$, each element $\mathbf{A}_{\text{MAPE},ij}$ is calculated as:

$$\mathbf{A}_{\text{MAPE},ij} = \frac{\text{Normalized MAPE}_i}{\text{Normalized MAPE}_j} \tag{30}$$

Similarly, for the MS matrix $\mathbf{A}_{\text{Size}}$, each element $\mathbf{A}_{\text{Size},ij}$ is calculated as:

$$\mathbf{A}_{\text{Size},ij} = \frac{\text{Normalized Size}_i}{\text{Normalized Size}_j} \tag{31}$$

These matrices allow for a comparison of each model against every other model in terms of MAPE and MS. To find the relative importance of each model, we calculate the principal eigenvector for each matrix by solving:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \tag{32}$$

where $\mathbf{A}$ is the pairwise comparison matrix, λ is the eigenvalue and $\mathbf{v}$ is the eigenvector. The principal eigenvector, corresponding to the largest eigenvalue, is then normalized to obtain the priority vector:

$$\mathbf{v}_{\text{normalized}} = \frac{\mathbf{v}}{\sum \mathbf{v}} \tag{33}$$

The consistency of the comparisons is checked using the consistency ratio (CR). The consistency index (CI) is calculated as:

$$\text{CI} = \frac{\lambda_{\max} - 9P}{9P - 1} \tag{34}$$

where $\lambda_{\max}$ is the largest eigenvalue of the pairwise comparison matrix and $9P$ is the size of the matrix. The consistency ratio is then computed as:

$$\text{CR} = \frac{\text{CI}}{\text{RI}} \tag{35}$$

where RI is the random index that depends on the size of the matrix and is obtained from standard tables. The CR measures how consistent the comparisons are compared to a perfectly consistent matrix. A CR less than 0.1 is acceptable and indicates consistency in the pairwise comparisons. If the CR exceeds 0.1, it suggests that the comparisons might be inconsistent and the comparison process should be revised. Ensuring consistency is crucial as it validates the decision-making process and ensures that the comparisons between the models are logically sound.

Finally, the overall AHP score for each model is computed by combining the normalized priority vectors for MAPE $\mathbf{v}_{\text{normalized, MAPE}}$ and MS $\mathbf{v}_{\text{normalized, Size}}$ as:

$$\text{AHP Score} = 0.5 \times \mathbf{v}_{\text{normalized, MAPE}} + 0.5 \times \mathbf{v}_{\text{normalized, Size}} \tag{36}$$

The models are ranked based on their AHP scores, with the lowest scores indicating the best models. This method provides a balanced evaluation of both accuracy and complexity, ensuring the selected models are suitable for deployment in memory-constrained environments. This AHP process helps make a clear and fair decision in selecting the optimal models. The pseudo code of the general framework is given by algorithm 1 and 2

Algorithm 1 Proposed Methodology

Initialize hyperparameter spaces for LSTM, GRU, TCN from Table 1;
for *model* $m \in \{LSTM, GRU, TCN\}$ **do**
  for *objective function* $f_j \in \{1, \ldots, 9\}$ **do**
    Optimize hyperparameters using aSAR (Algorithm 2) with f_j;
    Store top P models based on f_j scores;
  end for
end for
Train all $9 \times P$ models on full dataset;
Compute MAPE (Eq. 1) and Model Size (MS) for each model;
Normalize MAPE and MS using Eq. 29;
Construct pairwise comparison matrices (Eqs. 30, 31);
Calculate priority vectors via eigen decomposition (Eq. 32);
Check CR using Eqs. 34, 35;
if $CR < 0.1$ **then**
  Compute AHP scores (Eq. 36);
  Rank models by AHP scores;
end if
Select top-ranked model for each deployment location;

Algorithm 2 aSAR Optimization Algorithm

Input: Initial temp T_0, cooling rate α_0, β, λ, $D_{\max}$
Initialize population Θ with random solutions from hyperparameter space;
Evaluate initial fitness $E(\theta) \forall \theta \in \Theta$;
Initialize rejection memory $M_{\text{rej}} \leftarrow \emptyset$;
Compute initial fitness variance σ_0^2 (Eq. 24);
$k \leftarrow 0$;
while $T_k > T_{stop}$ ***and*** $k < N_{max}$ **do**
  foreach $\theta_i \in \Theta$ **do**
    $\theta_{\text{new}} \leftarrow \text{Perturb}(\theta_i)$;
    Evaluate $E_{\text{new}} \leftarrow f(\theta_{\text{new}})$;
    $\Delta E \leftarrow E_{\text{new}} - E(\theta_i)$;
    $P_{\text{accept}} \leftarrow \exp(\Delta E / T_k)$;
    foreach $\theta_{rej} \in M_{rej}$ **do**
      $D \leftarrow \text{Euclidean Distance}(\theta_{\text{new}}, \theta_{\text{rej}})$;
      if $D < D_{max}$ **then**
        $P_{\text{accept}} \leftarrow P_{\text{accept}} \times (1 - \lambda)$;
      end if
    end foreach
    if $\Delta E < 0$ ***or*** $random() < P_{accept}$ **then**
      $\theta_i \leftarrow \theta_{\text{new}}$;
      $E(\theta_i) \leftarrow E_{\text{new}}$;
    end if
    else
      $M_{\text{rej}} \leftarrow M_{\text{rej}} \cup \{\theta_{\text{new}}\}$;
    end if
  end foreach
  Compute σ_k^2 from current population Θ;
  $\alpha_k \leftarrow \alpha_0 \times (1 + \beta \sigma_k^2)$;
  $T_{k+1} \leftarrow T_k \times \alpha_k$;
  $k \leftarrow k + 1$;
end while
Return Best solution $\theta^* \in \Theta$;

III. RESULTS AND COMPARISON

This section explains the results and comparison of the proposed scheme with existing models. It first outlines the experimental setup and the performance evaluation metrics, followed by the results of the proposed method. The section concludes with a comparison between the optimized aSAR-LSTM, aSAR-TCN and aSAR-LSTM models and their non-optimized counterparts using the Wilcoxon signed-rank test.

A. EXPERIMENTAL SETUP

Since this study focuses on optimizing the hyperparameters of LSTM, TCN and GRU models for deployment on memory-constrained devices. Therefore, the memory constraint is the hard constraint hence, we have placed constraints on the maximum size of different hyper-parameters. Memory constraints necessitate precise hyper-parameters selection to minimize both MS and MAPE. The learning rate is fixed at 1×10^{-4} across all models, with independent optimization of each model's objective function while maintaining the same base architecture. The top three models selected by these objective functions are converted to the Open Neural Network Exchange (ONNX) format before deployment on NVIDIA Jetson devices. The Jetson Nano features a quad-core ARM Cortex-A57 CPU (1.43 GHz), a 128-CUDA-core Maxwell GPU (921 MHz) and 4 GB LPDDR4 RAM. It runs Ubuntu through the NVIDIA JetPack SDK, optimized for edge-AI frameworks including TensorFlow, PyTorch and OpenCV, with power consumption ranging from 5W to 10W (5 V DC), making it ideal for edge deployment. TensorFlow provides an open-source utility for converting model checkpoints to ONNX format, preserving the model's architecture and parameters for cross-platform compatibility. Post-conversion, the ONNX models execute efficiently on NVIDIA Jetson hardware while remaining portable to other edge devices such as the Intel Movidius Neural Compute Stick, Google Coral Edge TPU and Raspberry Pi systems. Table 2 summarizes the hyperparameters and constraints used for LSTM, GRU and TCN models:

The aSAR algorithm is employed for hyper-parameter optimization. The initial temperature is set at $T_0 = 1000$, with a stopping threshold $T_{\text{stop}} = 1$and a base cooling rate $\alpha_0 = 0.99$. The adaptive cooling rate α_k is modulated by the fitness variance scaling factor $\beta = 0.1$ and the memory-based rejection mechanism uses $\lambda = 0.2$ and $D_{\max} = 0.5$. The population size is $N = 10$ and the algorithm terminates after $K_{\max} = 100$ iterations. For model training, the dataset is split into 80% training, 10% validation and 10% testing. Training

TABLE 2. Summary of hyper-parameters and constraints for LSTM, GRU and TCN Models.

Hyper-parameters	LSTM	GRU	TCN
N	$1 \leq N \leq 6$		
U	$32 \leq U \leq 512$		-
D	$0.01 \leq D \leq 0.99$		$0.0 \leq D \leq 0.5$
A	{ReLU, Tanh, Sigmoid}		
K	$1 \leq K \leq 25$		
T	$1 \leq T \leq 12$		
F	-	-	$32 \leq F \leq 128$
K_s	-	-	$2 \leq K_s \leq 8$
S	-	-	$1 \leq S \leq 3$
Δ	-	-	{1, 2, 4, 8, 16, 32}
P	-	-	{causal, same}

runs for 15 epochs with a patience of 3 and a batch size of 128. Table 3 lists the aSAR algorithm parameters.

TABLE 3. aSAR algorithm parameters.

Parameter	Value
Initial Temperature (T_0)	1000
Stopping Temperature (T_{stop})	1
Base Cooling Rate (α_0)	0.99
Max Iterations ($K_{\max}$)	300
Population Size (N)	10
Variance Scaling Factor (β)	0.1
Penalty Strength (λ)	0.2
Max Similarity ($D_{\max}$)	0.5
Learning Rate	1×10^{-4}
Training Split	80%
Validation Split	10%
Test Split	10%
Epochs	15
Patience	3
Batch Size	128

B. SPATIO-TEMPORAL DATA DESCRIPTION

This study utilizes a spatio-temporal dataset capturing environmental observations collected across multiple locations and time periods. The dataset is obtained from an open-source repository provided by the National Renewable Energy Laboratory (NREL) [49]. At each location, hourly measurements are recorded for five key parameters, temperature (°C), relative humidity (%), wind speed (m/s), wind direction (°) and air pressure (hPa). Data collection involves a central station and four additional stations positioned 100 kilometers apart, aligned in the north, south, east and west directions.

The spatial component arises from recording observations at different locations, revealing how meteorological behavior varies geographically. The temporal component emerges from hourly sampling over extended periods, illustrating how these environmental features evolve over time. By combining both spatial and temporal dimensions, the dataset supports more comprehensive modeling compared to single-point or purely time-based series.

In this work, the main goal is to predict the next six-hour state of these environmental variables. The emphasis lies on optimizing hyperparameters of models such as LSTM, GRU, or TCN so that the prediction error remains minimal while retaining a compact architecture. Balancing prediction accuracy and MS is crucial for efficient and scalable deployment, especially where computational resources are constrained. By tuning aspects such as the number of layers, hidden units and regularization parameters, the approach seeks to exploit the spatio-temporal nature of the data without overfitting or incurring unnecessary computational costs.

Table 4 summarizes the locations, time ranges and the total number of hours in the dataset. An 80-10-10 split partitions the data into training, validation and testing sets. This division ensures the models learn from ample historical data while allowing independent validation and testing, thereby upholding a rigorous standard for performance evaluation.

TABLE 4. Summary of the spatio-temporal dataset and its 80-10-10 split.

Location	Period	Total Hours	Train (80%)	Valid (10%)	Test (10%)
Chile	2019–2022	35,070	28,056	3,507	3,507
Kazakhstan	2017–2019	26,298	21,038	2,630	2,630
Mongolia	2011–2015	43,830	35,064	4,383	4,383

Data quality is assured through NREL protocols [49], including sensor calibration and automated anomaly detection. Temporal consistency is confirmed by sequential timestamp validation and zero-gap analysis. Spatial consistency is supported by five-station measurements at a distance of 100 Km from each other. Normalized entropy quantifies feature diversity as:

$$H_n(X_i) = -\frac{1}{\log N} \sum_{k=1}^{N} p(x_k) \log p(x_k) \tag{37}$$

where $p(x_k)$ is the probability of bin k and N is the count of non-empty bins.Table 5 presents the normalized entropy of wind speed and its range per country.

TABLE 5. Normalized entropy and range of wind speed.

Country	Normalized Entropy H_n	Min Speed (m/s)	Max Speed (m/s)
Chile	0.85	0.10	16.10
Kazakhstan	0.72	0.20	18.40
Mongolia	0.76	0.10	12.30

Wind speed entropy is highest in Chile ($H_n = 0.85$), indicating a more uniform distribution of values. Mongolia and Kazakhstan have lower entropy values (0.76 and 0.72 respectively), indicating more clustered distributions.

These spatio-temporal records form the foundation for developing robust predictive models. By accounting for both geographic variation and temporal dynamics, the dataset provides a richer context than traditional time series data alone. This structure ultimately facilitates the design of models that deliver accurate six-hour predictions while maintaining practical levels of complexity for real-world applications.

C. COMPARISON MATRICES

This section presents the evaluation metrics used for model performance assessment. The metrics include, Mean Squared Error (MSE), Mean Absolute Error (MAE), R^2 Score and MS. The formulas for these metrics are provided below.

MSE is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 \tag{38}$$

MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right) \tag{39}$$

R^2 Score is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \tag{40}$$

In the above equations, y_i represents the actual wind speed, $\hat{y}_i$ represents the predicted wind speed, $\bar{y}$ is the mean of the actual wind speeds and n is the number of observations. The metric MS is computed separately. These metrics quantify the predictive accuracy and model fit in forecasting wind speeds, offering a comprehensive assessment of both error magnitude and explanatory power.

D. RESULTS OF PROPOSED SCHEME

This section discusses the results of the aSAR optimization algorithms in detail. The proposed scheme is applied to three datasets collected from Chile, Kazakhstan and Mongolia as discussed previously. In each dataset, the aSAR algorithm is used for hyper-parameter optimization using all nine objective function. Since the aSAR algorithm trains the models for only 15 epochs, there is a possibility that some models with certain hyper-parameters may not perform well initially but may improve with extensive training. Therefore, the top three models for each objective function are selected and then extensively trained for 2000 epochs with a patience of 15. Their MAPE and MS are then evaluated using the AHP procedure to determine the top models.

In each figure, unique models generated by all objective functions are plotted against MAPE and MS. MAPE is represented as θ and MS as radius r, with a circular layout on a 2D plane and the AHP score on the Z-axis. The MAPE, MS and AHP score are normalized before plotting. The best model is marked with a red dot, the second-best with a yellow dot and the third best with a green dot. The rest of the models are shown in cyan.

Figure 3 shows unique models generated by all objective functions when optimizing a TCN model on the Chile dataset using the aSAR algorithm. The best model has a MAPE of 13.03% and an MS of 9.01 MB, generated by the QP objective function (equation 7). The second-best model, generated by the HM objective function (equation 6), has a MAPE of 13.15% and an MS of 10.23 MB. The third best model, generated by the LC objective function (equation 4), has a MAPE of 14.22% and an MS of 12.34 MB. Moreover, figure 4 shows unique models generated by all objective functions when optimizing an LSTM model on the Chile dataset using the aSAR algorithm. The best model has a MAPE of 5.42% and an MS of 3.65 MB, generated by the FLI objective function (equation 10). The second-best model, generated by the QP objective function (equation 7), has a MAPE of 5.65% and an MS of 13.66 MB. The third best model, generated by the LWC objective function (equation 2), has a MAPE of 3.92% and an MS of 28.05 MB.

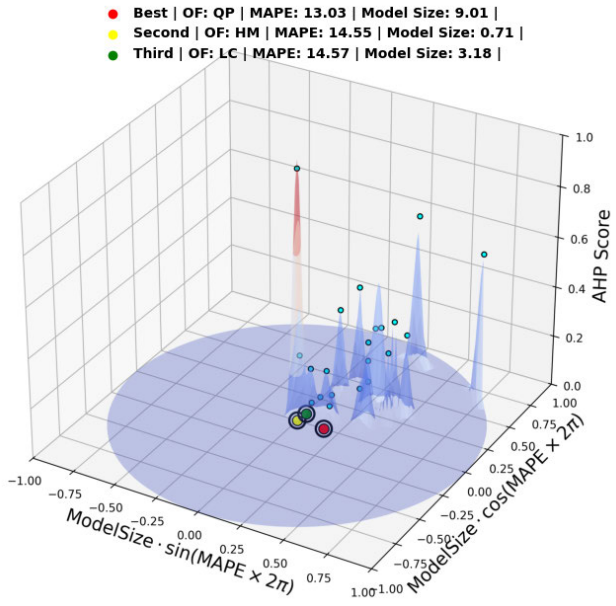

FIGURE 3. AHP score against MS and MAPE for TCN model on Chile dataset.

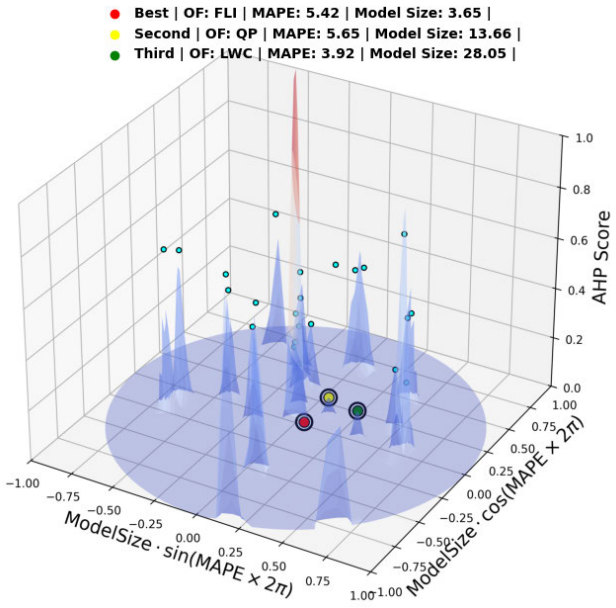

FIGURE 4. AHP score against MS and MAPE for LSTM model on Chile dataset.

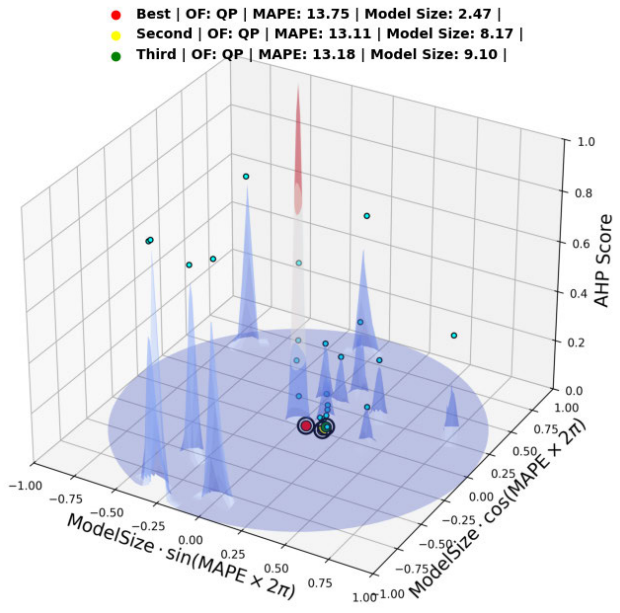

FIGURE 5. AHP score against MS and MAPE for GRU model on Chile dataset.

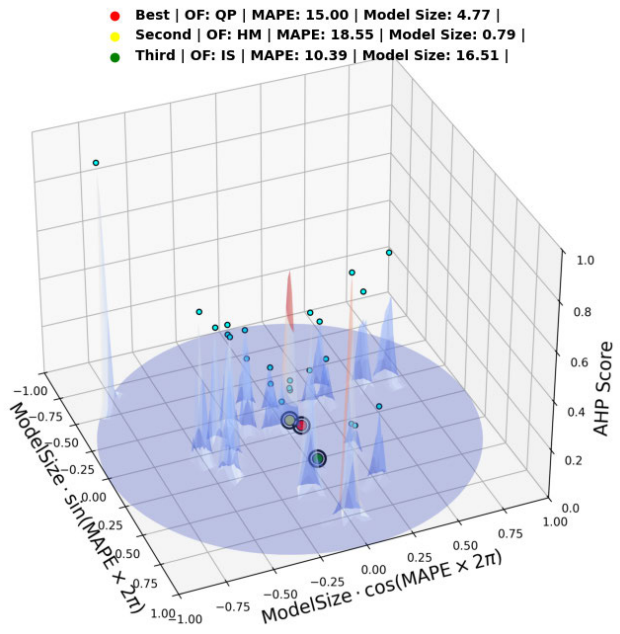

FIGURE 6. AHP score against MS and MAPE for TCN model on Kazakhstan dataset.

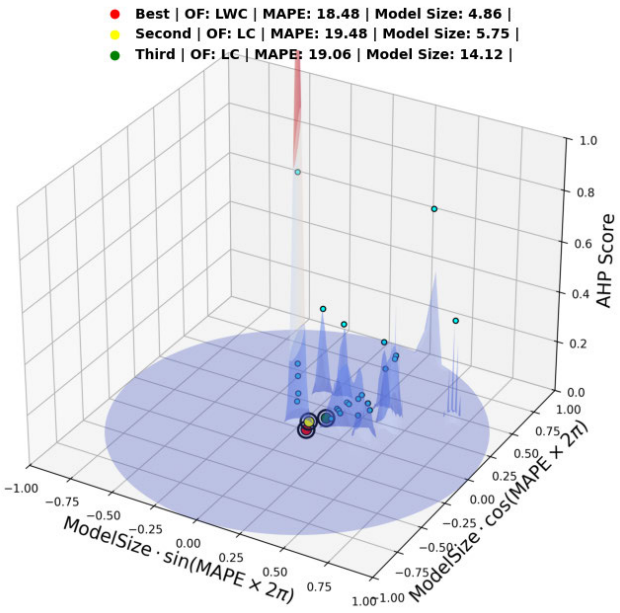

FIGURE 7. AHP score against MS and MAPE for LSTM model on Kazakhstan dataset.

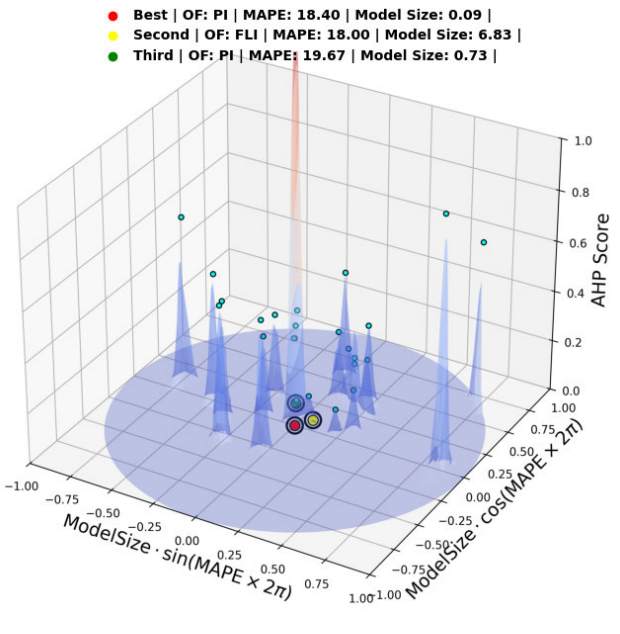

FIGURE 8. AHP score against MS and MAPE for GRU model on Kazakhstan dataset.

Furthermore, figure 5 shows unique models generated by all objective functions when optimizing a GRU model on the Chile dataset using the aSAR algorithm. The best model has a MAPE of 13.75% and an MS of 2.47 MB, generated by the QP objective function (equation 7). The second-best model, also generated by the QP objective function, has a MAPE of 13.11% and an MS of 8.17 MB. The third-best model, again generated by the QP objective function, has a MAPE of 13.18% and an MS of 9.10 MB.

Coming to the next dataset, figure 6 shows unique models generated by all objective functions when optimizing a TCN model on the Kazakhstan dataset using the aSAR algorithm. The best model has a MAPE of 15.0% and an MS of 4.77 MB, generated by the QP objective function (equation 7). The second-best model, generated by the HM objective function (equation 6), has a MAPE of 18.55% and an MS of 0.79 MB. The third best model, generated by the IS objective function (equation 3), has a MAPE of 10.39% and an MS of 16.51 MB.

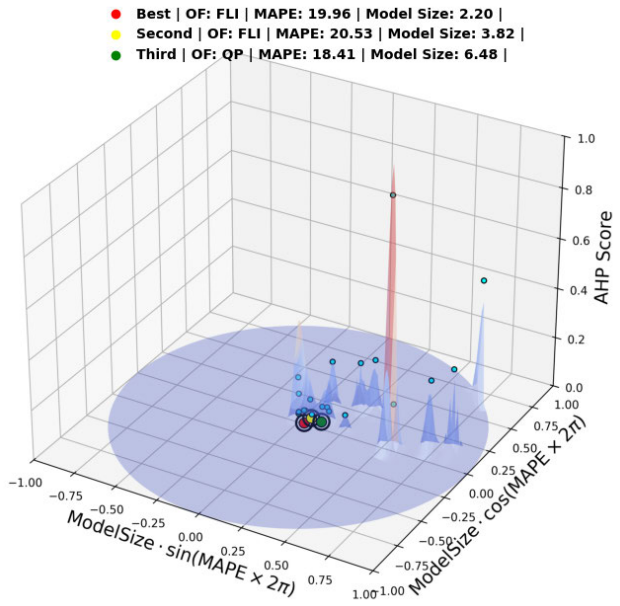

FIGURE 9. AHP score against MS and MAPE for TCN model on Mongolia dataset.

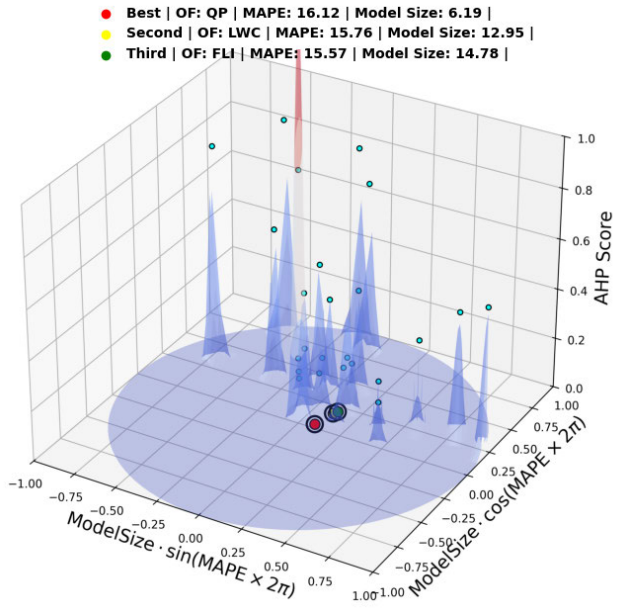

FIGURE 10. AHP score against MS and MAPE for LSTM model on Mongolia dataset.

Another model, figure 7, shows unique models generated by all objective functions when optimizing an LSTM model on the Kazakhstan dataset using the aSAR algorithm. The best model has a MAPE of 18.48% and an MS of 4.86 MB, generated by the LWC objective function (equation 2). The second-best model, generated by the LC objective function (equation 4), has a MAPE of 19.48% and an MS of 5.75 MB. The third best model, also generated by the LC objective function, has a MAPE of 19.06% and an MS of 14.12 MB.

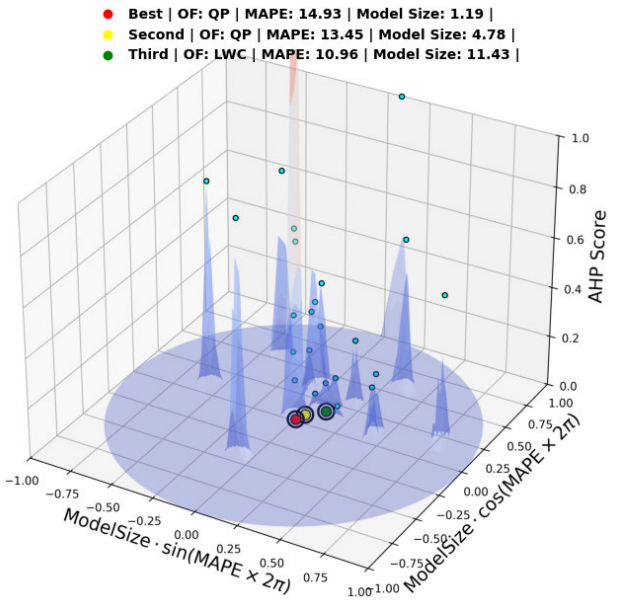

FIGURE 11. AHP score against MS and MAPE for GRU model on Mongolia dataset.

Figure 8 shows unique models generated by all objective functions when optimizing a GRU model on the Kazakhstan dataset using the aSAR algorithm. The best model has an MAPE of 18.40% and an MS of 0.09 MB, generated by the PI objective function (equation 9). The second-best model, generated by the FLI objective function (equation 10), has an MAPE of 18.00% and an MS of 6.83 MB. The third-best model, also generated by the PI objective function, has an MAPE of 19.67% and an MS of 0.73 MB.

Moving to the final dataset, figure 9 shows unique models generated by all objective functions when optimizing a TCN model on the Mongolia dataset using the aSAR algorithm. The best model has an MAPE of 19.96% and an MS of 2.20 MB, generated by the FLI objective function (equation 10). The second-best model, also generated by the FLI objective function, has an MAPE of 20.53% and an MS of 3.82 MB. The third-best model, generated by the QP objective function (equation 7), has an MAPE of 18.41% and an MS of 6.48 MB. Furthermore, figure 10 shows unique models generated by all objective functions when optimizing an LSTM model on the Mongolia dataset using the aSAR algorithm. The best model has an MAPE of 16.12% and an MS of 6.19 MB, generated by the QP objective function (equation 7). The second-best model, generated by the LWC objective function (equation 2), has an MAPE of 15.76% and an MS of 12.95 MB. The third best model, generated by the FLI objective function (equation 10), has an MAPE of 15.57% and an MS of 14.78 MB.

Moreover, figure 11 shows unique models generated by all objective functions when optimizing a GRU model on the Mongolia dataset using the aSAR algorithm. The best model has a MAPE of 14.93% and an MS of 1.19 MB, generated

by the QP objective function (equation 7). The second-best model, also generated by the QP objective function, has a MAPE of 13.45% and an MS of 4.78 MB. The third best model, generated by the LWC objective function (equation 2), has a MAPE of 10.96% and an MS of 11.43 MB.

1) ANALYSIS OF OBJECTIVE FUNCTION PERFORMANCE ACROSS DATASETS AND MODELS

This study evaluates the performance of nine objective functions across selected datasets for each model. Each combination of model and dataset is tested with these objective functions and the top three functions for each combination are recorded that are shown in red yellow and green colors in previous section. However, the frequency and proportion of these functions in the top three results are visually presented in figure 12.

The analysis reveals that the Quadratic Penalty (QP) objective function, defined by 7, is the most frequently selected, appearing in 37.04% of the top three results based on its AHP Score. This shows that QP effectively balances the trade-off between MS and MAPE (MAP) score across the various models and datasets. Moreover, the consistent appearance of QP in the top results suggests that it manages the trade-offs in model complexity and predictive accuracy well, which is essential in achieving robust performance across different scenarios. In addition, the Focal Loss Inspired (FLI) function, as described in 10, is the next most frequent, appearing in 18.52% of the top three results. This is due to its ability to focus on difficult-to-predict instances, which allows the model to improve prediction accuracy by prioritizing challenging data points. This approach is particularly relevant in datasets with high variability or noise. Hence, FLI's ability to emphasize these challenging cases makes it a valuable choice in complex data environments.

Additionally, the Linear Weighted Combination (LWC) function, defined in 2, is observed in 14.81% of the top results. This combines different error components linearly, suggests its effectiveness in environments where a straightforward aggregation of errors enhances model performance. Moreover, Logarithmic Combination (LC), Harmonic Mean (HM) and Product of Inverses (PI) are observed less frequently. Where, LC, defined in 4, appears in 11.11% of the top results, suggesting its utility in reducing the influence of large errors. Furthermore, HM, as described in 6 and PI, defined in 9, each appear in 7.41% of the results, highlighting their roles in managing specific aspects of model performance.

Furthermore, it is noteworthy that ES and CLL consistently fail to appear in the top 3 results.Their absence suggests that they may not be well-suited to the trade-offs required by the models and datasets in this study. In short, figure 12 shows the effectiveness of the QP objective function in achieving a balanced model that does not overly compromise between size and accuracy. Additionally, other functions like FLI and LWC also demonstrate their utility in particular contexts, offering alternative approaches depending on the specific demands of the dataset and model configuration.

E. PERFORMANCE ANALYSIS OF ADAPTIVE SAR-OPTIMIZED MODELS

This section provides a detailed analysis of the aSAR-optimized models (GRU, LSTM, TCN) across three different datasets (Chile, Kazakhstan, Mongolia). Each table presents the MSE, R^2 and MAE for the models over six prediction steps, followed by a discussion of the results.

The MSE performance of the aSAR-optimized models varies across the three datasets. In the Chile dataset, the LSTM model clearly outperforms the GRU and TCN models, achieving an average MSE of 0.11 $(m/s)^2$, which is significantly lower than GRU's 0.66 $(m/s)^2$ and TCN's 0.74 $(m/s)^2$. This indicates that LSTM provides 83.33% and 85.14% lesser MSE than GRU and TCN, respectively. In the Kazakhstan dataset, the TCN model stands out with the lowest average MSE of 0.27 $(m/s)^2$, outperforming the GRU and LSTM models by 41.30% and 40.00%, respectively. For the Mongolia dataset, GRU is the best performer with an average MSE of 0.14 $(m/s)^2$, outperforming LSTM by 26.32% and TCN by 50.00%.

The R^2 performance metric, which indicates the proportion of variance explained by the model, shows that the LSTM model performs best in the Chile dataset with an average R^2 of 0.98. This is significantly higher than the GRU and TCN models, which have average R^2 scores of 0.90 and 0.91, respectively, highlighting LSTM's superior predictive power in this dataset. In the Kazakhstan dataset, the TCN model once again leads with an average R^2 of 0.93, outperforming GRU (0.89) and LSTM (0.88) by 4.49% and 5.68%, respectively. For the Mongolia dataset, GRU is the top performer with an average R^2 of 0.95, slightly higher than LSTM's 0.94 and significantly higher than TCN's 0.91.

The MAE provides additional insight into the performance of the aSAR-optimized models. For the Chile dataset, LSTM again shows its dominance with the lowest average MAE of 0.23 m/s, outperforming GRU and TCN by 59.65% and 56.60%, respectively. In the Kazakhstan dataset, TCN shows the best performance with an average MAE of 0.38 m/s, making it 20.83% less than GRU and 22.45% less than LSTM. For the Mongolia dataset, GRU again leads with an average MAE of 0.27 m/s, outperforming LSTM by 10.00% and TCN by 27.03%.

This comprehensive analysis of the aSAR optimized models' performance across three datasets demonstrates that the choice of the optimal model depends heavily on the specific area from where the datasets are collected and the WS behaviors. The LSTM model excels in the Chile dataset, showing superior performance in MSE, R^2 and MAE. The TCN model is most effective for the Kazakhstan dataset, where it leads in all metrics. For the Mongolia dataset, the GRU model is the top performer, providing the best results in MSE, R^2 and MAE. These findings underscore

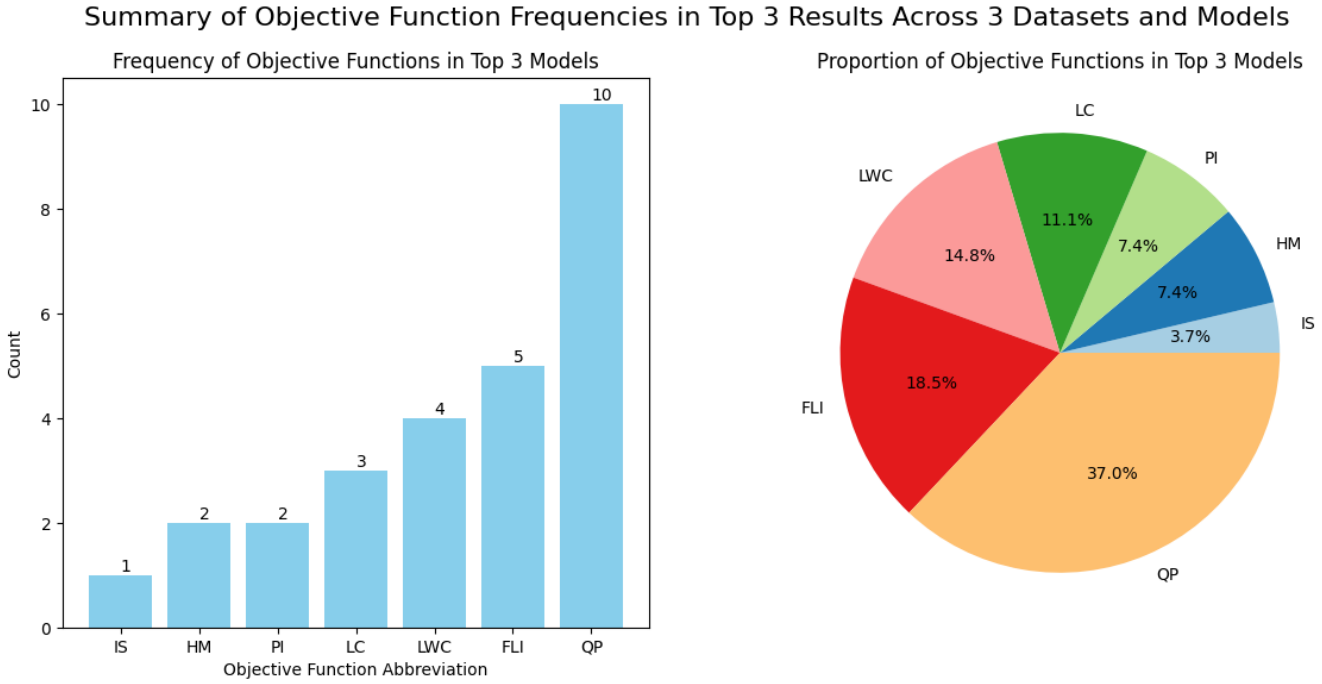

FIGURE 12. Frequency and Proportion of Objective Functions in Top 3 Results Across All Datasets and Models.

TABLE 6. aSAR-optimized models performance - MSE (m/s^2).

Dataset	Model	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Average MSE
Chile	GRU	0.06	0.25	0.49	0.74	1.03	1.39	0.66
	LSTM	**0.04**	**0.08**	**0.10**	**0.11**	**0.13**	**0.19**	**0.11**
	TCN	0.09	0.23	0.43	0.66	0.88	1.13	0.74
Kazakhstan	GRU	**0.06**	0.17	0.34	0.51	0.72	0.94	0.46
	LSTM	0.07	0.17	0.33	0.51	0.70	0.92	0.45
	TCN	0.08	**0.15**	**0.23**	**0.30**	**0.38**	**0.49**	**0.27**
Mongolia	GRU	**0.05**	**0.09**	**0.12**	**0.15**	**0.19**	**0.25**	**0.14**
	LSTM	0.05	0.11	0.16	0.21	0.26	0.34	0.19
	TCN	0.07	0.15	0.24	0.32	0.40	0.51	0.28

the importance of selecting models based on the dataset's characteristics to achieve optimal predictive accuracy.

F. LOCATION-SPECIFIC OPTIMIZATION FOR IMPROVED WIND SPEED PREDICTION ON UNSEEN WIND PATTERNS

The results presented earlier are based on test data representing previously unseen samples within each dataset. However, WSP is highly dependent on local wind patterns, which vary significantly by location. Therefore, optimizing models individually for each site is essential. The proposed approach enables location-specific offline optimization, demonstrating that tailoring the model to local wind patterns substantially enhances prediction performance.

This method allows manufacturers of small-scale domestic wind turbines to utilize publicly available wind data sources, such as NERL, to download site-specific wind patterns and optimize their models accordingly. Users can then download optimized ONNX models from manufacturer websites that are customized for their local conditions. Given the significant variability in wind patterns across locations, developing a single generalized model for small-scale turbines is impractical. Instead, this study emphasizes the importance of determining not only the optimal model architecture but also the location-specific parameters to improve WSP accuracy while keeping MS efficient.

G. UNCERTAINTY QUANTIFICATION IN WIND SPEED PREDICTION

Wind speed prediction uncertainty originates from both internal and external factors. Internal uncertainties arise from the model's structure and parameters, while external uncertainties come from measurement noise, sensor errors and unmodeled atmospheric variability. These uncertainties can be parametric (related to model parameters) or non-parametric (related to model form and architecture). In practice, the exact magnitude and structure of these

TABLE 7. aSAR-optimized models performance - R^2.

Dataset	Model	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	**Average R^2**
Chile	GRU	0.99	0.96	0.93	0.89	0.84	0.78	0.90
	LSTM	**0.99**	**0.99**	**0.98**	**0.98**	**0.98**	**0.97**	**0.98**
	TCN	0.99	0.96	0.93	0.89	0.86	0.82	0.91
Kazakhstan	GRU	0.99	0.96	0.92	0.88	0.83	0.77	0.89
	LSTM	0.98	0.96	0.93	0.88	0.84	0.78	0.88
	TCN	**0.98**	**0.96**	**0.95**	**0.93**	**0.91**	**0.88**	**0.93**
Mongolia	GRU	**0.98**	**0.97**	**0.96**	**0.95**	**0.94**	**0.92**	**0.95**
	LSTM	0.98	0.96	0.95	0.93	0.92	0.89	0.94
	TCN	0.98	0.95	0.92	0.90	0.87	0.84	0.91

TABLE 8. aSAR-optimized models performance - MAE (m/s).

Dataset	Model	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	**Average MAE**
Chile	GRU	0.19	0.35	0.52	0.65	0.78	0.92	0.57
	LSTM	**0.15**	**0.21**	**0.23**	**0.23**	**0.25**	**0.31**	**0.23**
	TCN	0.23	0.36	0.49	0.60	0.70	0.80	0.53
Kazakhstan	GRU	**0.18**	0.32	0.45	0.54	0.65	0.75	0.48
	LSTM	0.21	0.32	0.45	0.56	0.64	0.74	0.49
	TCN	0.22	**0.30**	**0.37**	**0.42**	**0.46**	**0.53**	**0.38**
Mongolia	GRU	0.16	**0.22**	**0.26**	**0.28**	**0.32**	**0.36**	**0.27**
	LSTM	**0.15**	0.24	0.29	0.32	0.36	0.41	0.30
	TCN	0.19	0.29	0.36	0.41	0.46	0.51	0.37

uncertainties are unknown and time-varying. Our approach quantifies overall uncertainty by analyzing prediction residuals and constructing well-calibrated 90% prediction intervals. The consistent PI coverage near the nominal level across different datasets and forecast horizons demonstrates that this method effectively captures the combined uncertainty effects.

Thus, although uncertainty components cannot be fully separated in real-time, the residual-based intervals provide meaningful and actionable estimates of forecast confidence, which is crucial for operational WSP and energy management. To quantify uncertainty, we analyze prediction residuals defined as $\epsilon = Y_{\text{test}} - \hat{Y}_{\text{pred}}$ and construct 90% prediction intervals (PIs) based on the empirical residual percentiles. We use key metrics to characterize uncertainty:

- **Residual bias** (μ_ϵ): Mean residual error per forecast horizon, indicating systematic over- or under-prediction.
- **Residual dispersion** (σ_ϵ): Standard deviation of residuals, representing stochastic uncertainty magnitude.
- **PI coverage**: Percentage of true observations falling within the 90% prediction interval, measuring calibration accuracy.
- **PI width**: Average size of the prediction intervals, balancing precision and reliability.

1) UNCERTAINTY CHARACTERISTICS ACROSS REGIONS

Table 9 summarizes these metrics for the top-performing models across Chile, Kazakhstan and Mongolia. Residual bias remains close to zero at all horizons($|\mu_\epsilon| < 0.04$ m/s), confirming negligible systematic error in the forecasts. Residual dispersion increases with forecast horizon, reflecting growing uncertainty over time due to chaotic atmospheric dynamics. For example, in Mongolia, dispersion rises from approximately 0.22 m/s at 1 hour ahead to 0.49 m/s at 6 hours ahead.

PI coverage consistently aligns closely with the nominal 90% target (all near 89.98–89.99%), indicating well-calibrated uncertainty intervals. Notably, the average PI width expands proportionally with residual dispersion, roughly doubling it, which is expected as wider intervals are required to maintain coverage over increasing uncertainty horizons.

2) IMPLICATIONS FOR EDGE DEPLOYMENT AND OPERATIONAL USE

Quantifying uncertainty is crucial for effective energy management and real-time operational decisions. Wider prediction intervals at longer horizons (for example, exceeding 1.5 m/s at 6 hours) indicate lower forecast confidence. In such cases, the system may switch to safer fallback options, such as using typical average wind speeds for the area or reducing reliance on predicted wind power to avoid unexpected drops in energy supply. Lightweight models like aSAR-GRU (1.19 MB) are well-suited for real-time uncertainty quantification on edge devices such as the NVIDIA Jetson, enabling on-device computation of prediction intervals for autonomous control. Although more complex methods exist to separately model different uncertainty types, they often require significantly more computational and memory resources, making them impractical for resource-constrained devices like the Jetson. Overall, the uncertainty quantification results confirm well-calibrated models with low bias, increasing stochastic uncertainty with forecast horizon and reliable

TABLE 9. Uncertainty metrics for optimized models (90% PIs).

Dataset	Horizon (hr)	Residual μ_ϵ (m/s)	Residual σ_ϵ (m/s)	PI Coverage (%)	PI Width (m/s)
Chile (aSAR-LSTM)	1	-0.007	0.20	89.99	0.62
	2	0.008	0.28	89.99	0.89
	3	0.022	0.32	89.99	0.96
	4	0.019	0.34	89.99	0.99
	5	-0.006	0.36	89.99	1.07
	6	-0.026	0.44	89.99	1.33
Kazakhstan (aSAR-TCN)	1	-0.046	0.28	89.98	0.92
	2	-0.033	0.39	89.98	1.28
	3	-0.024	0.48	89.98	1.53
	4	-0.019	0.55	89.98	1.75
	5	0.012	0.62	89.98	1.93
	6	0.033	0.70	89.98	2.21
Mongolia (aSAR-GRU)	1	-0.013	0.22	89.98	0.70
	2	-0.007	0.30	89.98	0.99
	3	0.002	0.35	89.98	1.13
	4	0.012	0.39	89.98	1.20
	5	0.027	0.43	89.98	1.36
	6	0.031	0.49	89.98	1.56

TABLE 10. Performance comparison with state-of-the-art models.

Dataset	Model	MSE (m/s^2)	MAE (m/s)	R^2 Score	MAPE (%)	MS (MB)
Chile	iTransformer [50]	0.09	0.24	0.97	5.6	13.9
	Flowformer [51]	0.13	0.27	0.95	6.2	27.9
	IDBO-VTGA [34]	0.24	0.56	0.89	14.2	2.4
	PI-LSTM [25]	0.22	0.45	0.91	9.7	3.85
	1D CNN-LSTM [52]	0.15	0.31	0.92	8.8	4.89
	Trans-LSTM [47]	0.17	0.34	0.90	9.8	5.65
	aSAR-LSTM	0.11	**0.23**	**0.98**	**5.4**	**3.65**
Kazakhstan	iTransformer [50]	0.28	0.41	0.90	17.5	13.9
	Flowformer [51]	0.35	0.42	0.87	19.6	27.9
	IDBO-VTGA [34]	0.55	0.89	0.82	26.5	2.4
	PI-LSTM [25]	0.37	0.46	0.84	20.9	3.85
	1D CNN-LSTM [52]	0.29	0.44	0.89	18.4	4.89
	Trans-LSTM [47]	0.45	0.52	0.85	22.2	5.65
	aSAR-TCN	**0.27**	**0.38**	**0.93**	**15.0**	**4.77**
Mongolia	iTransformer [50]	0.20	0.31	0.91	16.8	13.9
	Flowformer [51]	0.18	0.28	0.93	15.9	27.9
	IDBO-VTGA [34]	0.24	0.39	0.86	24.6	2.4
	PI-LSTM [25]	0.28	0.42	0.84	25.9	3.85
	1D CNN-LSTM [52]	0.23	0.34	0.88	19.2	4.89
	Trans-LSTM [47]	0.20	0.34	0.90	17.7	5.65
	aSAR-GRU	**0.14**	**0.27**	**0.95**	**14.9**	**1.19**

prediction interval coverage, providing actionable confidence estimates essential for operational wind energy forecasting.

H. COMPARISON WITH EXISTING METHODS

This section compares the proposed aSAR models with existing methods for WSP. The models considered include iTransformer [50], Flowformer [51], IDBO-VTGA [34], PI-LSTM [25], Trans-LSTM [47] and 1D CNN-LSTM [52]. All these models are optimized using the Chile dataset, and the same hyperparameters are applied to other locations. The hyperparameters of all models are optimized to minimize MSE, except for the Trans-LSTM model, which uses its own proposed objective function. In contrast, the proposed model employs terrain-dependent hyperparameter optimization and selects architectures specifically suited to each location rather than relying on a single universal architecture. Therefore, for the Chile dataset, the model used is aSAR-LSTM, for the Kazakhstan dataset, aSAR-TCN is applied and for the Mongolia dataset, aSAR-GRU is employed. The performance metrics reported are MSE in $(m/s)^2$, MAE in (m/s), R^2 Score, MAPE in (%) and MS in MB. Table 10 shows the numerical values for all models.

For the Chile dataset, iTransformer reports an MSE of 0.09 m/s^2, MAE of 0.24 m/s, R^2 of 0.97 and MAPE of 5.6% with a MS of 13.9 MB, while Flowformer shows a slightly higher MSE of 0.13 m/s^2, MAE of 0.27 m/s, R^2 of 0.95 and MAPE of 6.2% at 27.9 MB. IDBO-VTGA, with a MS of 2.4 MB, exhibits higher errors with an MSE of 0.24 m/s^2 and MAE of 0.56 m/s and a lower R^2 of 0.89 along with a MAPE of 14.2%. PI-LSTM and 1D CNN-LSTM yield intermediate results with MSEs of 0.22 m/s^2 and 0.15 m/s^2, MAEs of

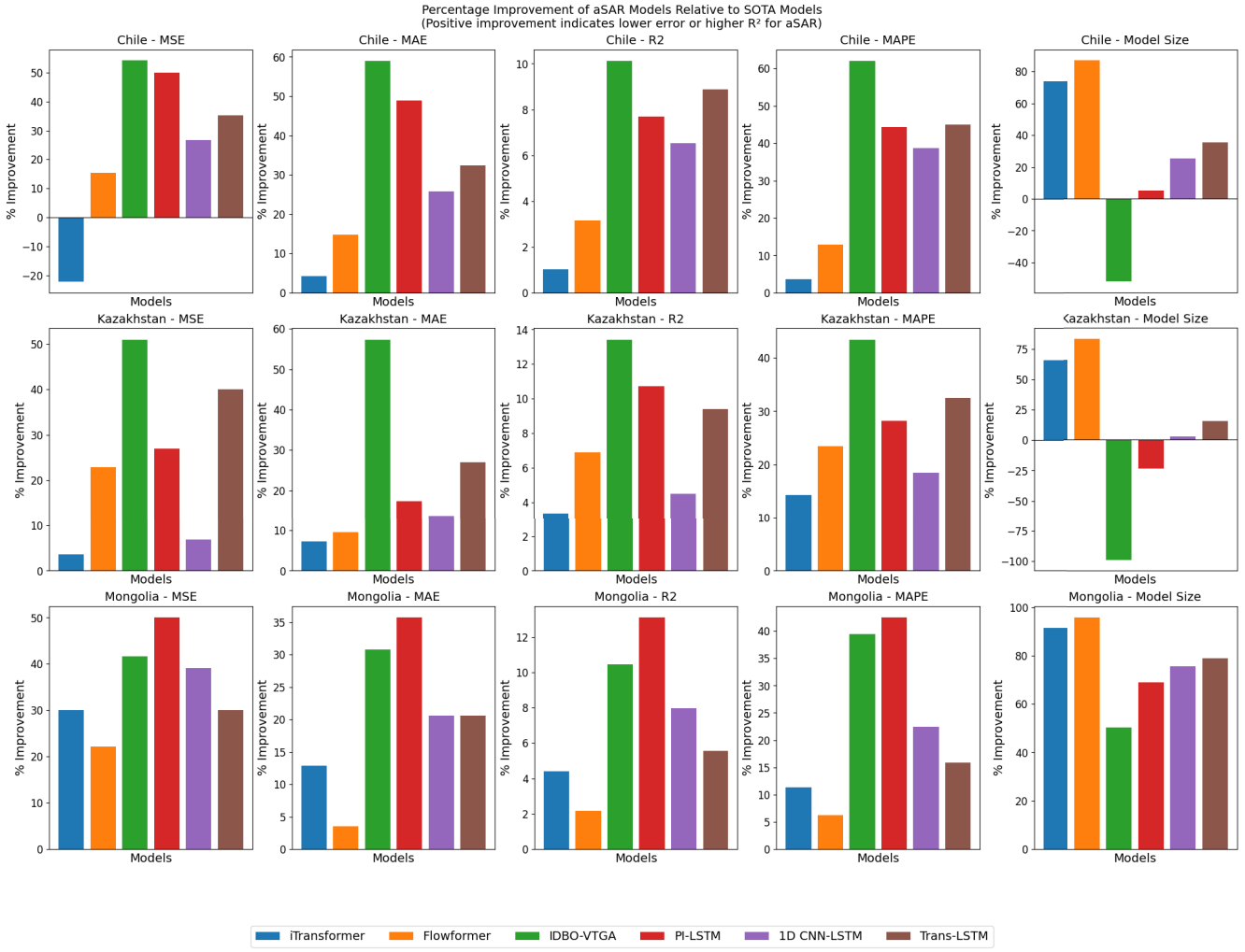

FIGURE 13. Percentage improvements of the aSAR models relative to state-of-the-art methods across the Chile, Kazakhstan and Mongolia datasets. The x-axis is labeled "Models" and the legend, located below the subplots, identifies the competing models.

TABLE 11. Optimized Hyper-parameters and average latency for GRU and LSTM models.

Model	Dataset	N	U	D	A	K	T	$Latency(ms)$
GRU	Chile	1	256	0.10	Tanh	14	11	0.40
	Kazakhstan	1	34	0.14	Tanh	18	10	0.51
	Mongolia	3	[107, 41, 98]	0.08	[ReLU, Tanh, Tanh]	12	12	0.614
LSTM	Chile	1	267	0.04	Tanh	24	11	0.17
	Kazakhstan	1	315	0.01	ReLU	15	9	1.3
	Mongolia	1	356	0.07	ReLU	17	9	0.29

TABLE 12. Optimized hyper-parameters and average latency for TCN models.

Dataset	F	K_s (Kernel Size)	S	Δ	P	Use Skip Connections	D (Dropout Rate)	A	K	T	$Latency(ms)$
Mongolia	62	4	1	[4, 16, 1, 8, 2, 32]	causal	True	0.09	ReLU	25	6	0.32
Kazakhstan	39	7	3	[2, 4, 32, 1, 8, 16]	causal	True	0.07	ReLU	15	6	0.62
Chile	93	8	1	[4, 32, 16, 2, 8, 1]	causal	True	0.03	ReLU	9	10	0.30

0.45 m/s and 0.31 m/s, R^2 scores of 0.91 and 0.92 and MAPEs of 9.7% and 8.8%, with MSs of 3.85 MB and 4.89 MB, respectively. In contrast, aSAR-LSTM attains an MSE of 0.11 m/s^2, MAE of 0.23 m/s, R^2 of 0.98 and MAPE of 5.4%

while having a MS of 3.65 MB. The comparison illustrates that although iTransformer has the lowest MSE, its larger MS and the performance of Flowformer, which is both high in size and error, are drawbacks. Meanwhile, IDBO-VTGA, despite its small size, suffers from high errors.

In the Kazakhstan dataset, iTransformer achieves an MSE of 0.28 m/s^2, MAE of 0.41 m/s, R^2 of 0.90 and MAPE of 17.5% with a size of 13.9 MB. Flowformer performs worse with an MSE of 0.35 m/s^2, MAE of 0.42 m/s, R^2 of 0.87 and MAPE of 19.6% and its MS is 27.9 MB. IDBO-VTGA continues to show higher error metrics with an MSE of 0.55 m/s^2, MAE of 0.89 m/s, R^2 of 0.82 and MAPE of 26.5% at a size of 2.4 MB. PI-LSTM and 1D CNN-LSTM yield MSEs of 0.37 m/s^2 and 0.29 m/s^2, MAEs of 0.46 m/s and 0.44 m/s, R^2 scores of 0.84 and 0.89 and MAPEs of 20.9% and 18.4%, with sizes of 3.85 MB and 4.89 MB, respectively. aSAR-TCN records an MSE of 0.27 m/s^2, MAE of 0.38 m/s, R^2 of 0.93 and MAPE of 15.0% while having a MS of 4.77 MB. This demonstrates that aSAR-TCN provides lower errors compared to the other methods and although iTransformer shows comparable MSE, its overall efficiency is limited by a larger MS, whereas Flowformer suffers both in performance and size.

For the Mongolia dataset, iTransformer achieves an MSE of 0.20 m/s^2, MAE of 0.31 m/s, R^2 of 0.91 and MAPE of 16.8% at a size of 13.9 MB. Flowformer reports an MSE of 0.18 m/s^2, MAE of 0.28 m/s, R^2 of 0.93 and MAPE of 15.9%, but with a MS of 27.9 MB. IDBO-VTGA records an MSE of 0.24 m/s^2, MAE of 0.39 m/s, R^2 of 0.86 and MAPE of 24.6% while being the smallest model at 2.4 MB, which, however, comes with relatively high errors. PI-LSTM and 1D CNN-LSTM provide MSEs of 0.28 m/s^2 and 0.23 m/s^2, MAEs of 0.42 m/s and 0.34 m/s, R^2 scores of 0.84 and 0.88 and MAPEs of 25.9% and 19.2% with model sizes of 3.85 MB and 4.89 MB, respectively. aSAR-GRU attains the lowest errors among the models with an MSE of 0.14 m/s^2, MAE of 0.27 m/s, R^2 of 0.95 and MAPE of 14.9% and its MS is 1.19 MB. This comparison shows that while Flowformer maintains a low error in MSE relative to some models, its high MS is a disadvantage. Similarly, IDBO-VTGA, though small in size, exhibits higher error values. aSAR-GRU, on the other hand, balances error metrics and MS effectively.

Across the three datasets, the proposed aSAR models show a mixed pattern of improvements and drawbacks in different metrics. In the Chile dataset, the iTransformer variant exhibits a negative 22.22% improvement in MSE—indicating worse performance in that metric—while delivering modest gains in MAE (4.17%), R^2 (1.03%) and MAPE (3.57%), along with a substantial reduction in MS (73.74%). In contrast, Flowformer achieves positive improvements in Chile with a 15.38% gain in MSE, 14.81% in MAE, 3.16% in R^2 and 12.90% in MAPE, though its MS increases by 86.92%. The IDBO-VTGA and PI-LSTM variants in Chile record high improvements in error metrics—54.17% and 50.00% in MSE respectively—with IDBO-VTGA also reducing MS by 52.08% and PI-LSTM showing a slight size increase of 5.19%, while the 1D CNN-LSTM delivers moderate gains (26.67% in MSE) with a 25.36% reduction in size. In the Kazakhstan dataset, iTransformer offers small improvements (e.g., 3.57% in MSE) and a 65.68% reduction in MS, whereas Flowformer and IDBO-VTGA improve MSE by 22.86% and 50.91% respectively, with IDBO-VTGA notably reducing size by 98.75%. PI-LSTM and 1D CNN-LSTM in Kazakhstan achieve improvements of 27.03% and 6.90% in MSE, with MS reductions of 23.90% and a slight increase of 2.45% respectively. In the Mongolia dataset, iTransformer shows a 30.00% improvement in MSE with a large size increase of 91.44% and Flowformer improves MSE by 22.22% while increasing size by 95.73%. Additionally, IDBO-VTGA attains a 41.67% improvement in MSE with a 50.42% size increase, PI-LSTM records a 50.00% improvement in MSE with a 69.09% size increase and 1D CNN-LSTM achieves a 39.13% improvement in MSE with a 75.66% size increase. These results demonstrate that while some models achieve high error metric improvements, they may do so at the cost of increased MS and a negative improvement in a metric (as seen with iTransformer's MSE in Chile) clearly indicates poorer performance in that specific aspect. Figure 13 graphically illustrates these percentage improvements across the three datasets.

The overall comparison across datasets indicates that models with lower error metrics tend to have larger model sizes, as seen with iTransformer and Flowformer, whereas models like IDBO-VTGA, despite their small sizes, do not achieve competitive accuracy. The proposed aSAR models, namely aSAR-LSTM, aSAR-TCN and aSAR-GRU, offer a balanced trade-off between performance and MS, resulting in lower errors and reduced computational requirements.

IV. CONCLUSION

This research developed a location-specific hyperparameter optimization framework for WSP models using LSTM, GRU, and TCN architectures. The aSAR algorithm, incorporating a memory-based rejection mechanism, balanced prediction accuracy against model complexity under memory constraints. Nine objective functions were evaluated, with Quadratic Penalty (QP), Focal Loss Inspired (FLI), and Linear Weighted Combination (LWC) demonstrating superior performance in minimizing MAPE and MS. Evaluations across Chile, Kazakhstan, and Mongolia confirmed that universal architectures fail to generalize. Both architecture selection and hyperparameter tuning depend critically on local wind patterns, indicating that small-scale turbines require location-optimized models for efficient edge deployment.

Extensive evaluations conducted on datasets from Chile, Kazakhstan, and Mongolia highlighted that a single set of hyperparameters or a universal model architecture is insufficient for every location. Instead, results emphasized that both the choice of architecture and hyperparameters are significantly dependent on local wind patterns. This indicates that domestic small-scale wind turbines would greatly

benefit from location-specific optimization and architecture selection. However, this method requires a substantial dataset to train the model and find the optimal hyperparameters. Changes in weather patterns due to global warming may affect the performance of the deployed model. In future work, extensive experimentation will be required to determine a robust set of data features and to identify optimal hyperparameters and architectures. Such efforts could facilitate the training of a surrogate model capable of suggesting optimal checkpoints tailored to specific locations based on local wind statistics (variance, ramp frequency). Given the highly variable nature of wind statistics, the development of these adaptive surrogate models could substantially streamline the deployment process, making high-precision, location-specific wind prediction more accessible and efficient.

APPENDIX
OPTIMIZED MODEL HYPER-PARAMETERS AND AVERAGE LATENCY FOR SELECTED DATASETS

This section presents the optimized hyper-parameters for the GRU, LSTM and TCN models on the selected datasets. Table 11 summarizes the optimized hyper-parameters for the GRU and LSTM models. Both models for Chile and Kazakhstan use simpler architectures ($N = 1$), with different numbers of units and activation functions: Tanh for Chile and ReLU for Kazakhstan. The Mongolia dataset required a deeper GRU model ($N = 3$) with different units across layers and a combination of ReLU and Tanh activations. The LSTM model for Mongolia uses ReLU activation and a slightly higher number of units.

Table 12 presents the optimized hyper-parameters for the TCN models. The configurations vary across datasets, with different numbers of filters (F), kernel sizes (K_s) and dilations (Δ) reflecting the complexity of the data. The dropout rates were lowest for Chile ($D = 0.03$) and highest for Mongolia ($D = 0.09$), with Kazakhstan's dropout rate in between. The use of ReLU activation was consistent across all TCN models, suggesting its effectiveness for capturing non-linearities.

ACKNOWLEDGMENT

The authors would like to thank Prince Sultan University for their support. The authors are also grateful to the Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R239), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

REFERENCES

[1] S. M. Valdivia-Bautista, J. A. Domínguez-Navarro, M. Pérez-Cisneros, C. J. Vega-Gómez, and B. Castillo-Téllez, "Artificial intelligence in wind speed forecasting: A review," *Energies*, vol. 16, no. 5, p. 2457, Mar. 2023.

[2] B. S. Bommidi, K. Teeparthi, and V. K. Dulla Mallesham, "ICEEMDAN-Informer-GWO: A hybrid model for accurate wind speed prediction," *Environ. Sci. Pollut. Res.*, vol. 31, no. 23, pp. 34056–34081, May 2024.

[3] Ö. A. Karaman, "Prediction of wind power with machine learning models," *Appl. Sci.*, vol. 13, no. 20, p. 11455, Oct. 2023.

[4] E. Nascimento, T. de Melo, and D. Moreira, "Wind power prediction using hybrid models," *Renew. Energy*, vol. 183, pp. 351–368, Jul. 2023.

[5] M. Verma and H. K. Ghritlahre, "Forecasting of wind speed by using three different techniques of prediction models," *Ann. Data Sci.*, vol. 10, no. 3, pp. 679–711, Jun. 2023.

[6] X. Zhao, J. Liu, D. Yu, and J. Chang, "One-day-ahead probabilistic wind speed forecast based on optimized numerical weather prediction data," *Energy Convers. Manage.*, vol. 164, pp. 560–569, May 2018.

[7] H. Wang, S. Han, Y. Liu, J. Yan, and L. Li, "Sequence transfer correction algorithm for numerical weather prediction wind speed and its application in a wind power forecasting system," *Appl. Energy*, vol. 237, pp. 1–10, Mar. 2019.

[8] F. Cassola and M. Burlando, "Wind speed and wind energy forecast through Kalman filtering of numerical weather prediction model output," *Appl. Energy*, vol. 99, pp. 154–166, Nov. 2012.

[9] X. Liu, L. Zhang, J. Wang, Y. Zhou, and W. Gan, "A unified multi-step wind speed forecasting framework based on numerical weather prediction grids and wind farm monitoring data," *Renew. Energy*, vol. 211, pp. 948–963, Jul. 2023.

[10] A. Gsella, A. de Meij, A. Kerschbaumer, E. Reimer, P. Thunis, and C. Cuvelier, "Evaluation of MM5, WRF and TRAMPER meteorology over the complex Terrain of the Po Valley, Italy," *Atmos. Environ.*, vol. 89, pp. 797–806, Jun. 2014.

[11] Z. Di, J. Ao, Q. Duan, J. Wang, W. Gong, C. Shen, Y. Gan, and Z. Liu, "Improving WRF model turbine-height wind-speed forecasting using a surrogate- based automatic optimization method," *Atmos. Res.*, vol. 226, pp. 1–16, Sep. 2019.

[12] F. M. Ottaviani and A. De Marco, "Multiple linear regression model for improved project cost forecasting," *Proc. Comput. Sci.*, vol. 196, no. 1, pp. 808–815, Jan. 2022.

[13] M. E. Banihabib and P. Mousavi-Mirkalaei, "Extended linear and non-linear auto-regressive models for forecasting the urban water consumption of a fast-growing city in an arid region," *Sustain. Cities Soc.*, vol. 48, Jul. 2019, Art. no. 101585.

[14] J. Palomares-Salas, J. de la Rosa, J. Ramiro, J. Melgar, A. Aguera, and A. Moreno, "ARIMA vs. Neural networks for wind speed forecasting," in *Proc. IEEE Int. Conf. Comput. Intell. Meas. Syst. Appl.*, May 2009, pp. 129–133.

[15] D. Chen, P. Li, X. Li, C. Xu, Y. Xiao, and B. Lei, "Short-term wind speed forecasting considering heteroscedasticity," in *Proc. Int. Conf. Adv. Power Syst. Autom. Protection*, vol. 2, Oct. 2011, pp. 884–888.

[16] W. Sun, B. Tan, and Q. Wang, "Multi-step wind speed forecasting based on secondary decomposition algorithm and optimized back propagation neural network," *Appl. Soft Comput.*, vol. 113, Dec. 2021, Art. no. 107894.

[17] Z. Liu, R. Hara, and H. Kita, "Hybrid forecasting system based on data area division and deep learning neural network for short-term wind speed forecasting," *Energy Convers. Manage.*, vol. 238, Jun. 2021, Art. no. 114136.

[18] D. Bechrakis and P. Sparis, "Wind speed forecasting using feedforward neural networks," *Renew. Energy*, vol. 13, pp. 345–354, Jan. 1998.

[19] M. A. Mohandes, T. O. Halawani, S. Rehman, and A. A. Hussain, "Support vector machines for wind speed prediction," *Renew. Energy*, vol. 29, no. 6, pp. 939–947, May 2004.

[20] S. Moreno, J. Gonzlez, and A. Cuerva, "Wind speed forecasting based on singular spectrum analysis and support vector regression," *Renew. Energy*, vol. 126, pp. 293–305, May 2018.

[21] D. Geng, H. Zhang, and H. Wu, "Short-term wind speed prediction based on principal component analysis and LSTM," *Appl. Sci.*, vol. 10, no. 13, p. 4416, Jun. 2020.

[22] J. Li, Z. Song, X. Wang, Y. Wang, and Y. Jia, "A novel offshore wind farm typhoon wind speed prediction model based on PSO–Bi-LSTM improved by VMD," *Energy*, vol. 251, Jul. 2022, Art. no. 123848.

[23] F. Shahid, A. Zameer, and M. Muneeb, "A novel genetic LSTM model for wind power forecast," *Energy*, vol. 223, May 2021, Art. no. 120069.

[24] H. Cai, Z. Wang, X. Feng, and X. Wang, "Wind speed prediction using extreme gradient boosting," *Energy*, vol. 205, Jan. 2020, Art. no. 118046.

[25] L. Aslam, R. Zou, E. S. Awan, and S. A. Butt, "Integrating physics-informed vectors for improved wind speed forecasting with neural networks," in *Proc. 14th Asian Control Conf. (ASCC)*, Jul. 2024, pp. 1902–1907.

[26] H. Liu, H. Tian, Y. Li, and Y. Zhang, "Comparison of arima and ann models for short-term wind speed forecasting," *Renew. Energy*, vol. 48, pp. 1–6, Apr. 2012.

[27] Z. Zhang, J. Wang, and Z. Yan, "A novel hybrid model for wind speed forecasting based on singular spectrum analysis and wavelet transforms," *Energy Convers. Manage.*, vol. 195, pp. 191–208, Jan. 2019.
[28] Z. Wan, J. Yang, and G. Liu, "Short-term wind speed prediction using transformer models," *Renew. Energy*, vol. 202, pp. 92–101, May 2023.
[29] S. Pan, K. Li, and Z. Wang, "Oil and gas production forecasting using hybrid CNN-LSTM networks," *J. Petroleum Sci. Eng.*, vol. 208, Jun. 2023, Art. no. 109391.
[30] J. Wang, W. Jiang, S. Shu, and X. He, "A multi-factor clustering integration paradigm for wind speed point-interval prediction based on feature selection and optimized inverted transformer," *Energy*, vol. 320, Apr. 2025, Art. no. 135210.
[31] H. Wu, K. Meng, D. Fan, Z. Zhang, and Q. Liu, "Multistep short-term wind speed forecasting using transformer," *Energy*, vol. 261, Dec. 2022, Art. no. 125231.
[32] X. Li, W. Zhang, and Y. Liu, "Enhancing wind speed forecasting accuracy using attention mechanisms," *Renew. Energy*, vol. 210, Jun. 2024, Art. no. 118763.
[33] A. Zhu, Q. Zhao, T. Yang, L. Zhou, and B. Zeng, "Wind speed prediction and reconstruction based on improved grey wolf optimization algorithm and deep learning networks," *Comput. Electr. Eng.*, vol. 114, Mar. 2024, Art. no. 109074.
[34] Y. He, W. Wang, M. Li, and Q. Wang, "A short-term wind power prediction approach based on an improved dung beetle optimizer algorithm, variational modal decomposition, and deep learning," *Comput. Electr. Eng.*, vol. 116, May 2024, Art. no. 109182.
[35] S. Liu, T. Xu, X. Du, Y. Zhang, and J. Wu, "A hybrid deep learning model based on parallel architecture TCN-LSTM with Savitzky–Golay filter for wind power prediction," *Energy Convers. Manage.*, vol. 302, Feb. 2024, Art. no. 118122.
[36] B. Yan, R. Shen, K. Li, Z. Wang, Q. Yang, X. Zhou, and L. Zhang, "Spatio-temporal correlation for simultaneous ultra-short-term wind speed prediction at multiple locations," *Energy*, vol. 284, Dec. 2023, Art. no. 128418.
[37] Y. Lehnardt, J. R. Barber, and O. Berger-Tal, "Effects of wind turbine noise on songbird behavior during nonbreeding season," *Conservation Biol.*, vol. 38, no. 2, p. 14188, Apr. 2024.
[38] Y. Teff-Seker, O. Berger-Tal, Y. Lehnardt, and N. Teschner, "Noise pollution from wind turbines and its effects on wildlife: A cross-national analysis of current policies and planning regulations," *Renew. Sustain. Energy Rev.*, vol. 168, Oct. 2022, Art. no. 112801.
[39] G. E. Barter, L. Sethuraman, P. Bortolotti, J. Keller, and D. A. Torrey, "Beyond 15 MW: A cost of energy perspective on the next generation of drivetrain technologies for offshore wind turbines," *Appl. Energy*, vol. 344, Aug. 2023, Art. no. 121272.
[40] R. Wiser, D. Millstein, M. Bolinger, S. Jeong, and A. Mills, "The hidden value of large-rotor, tall-tower wind turbines in the United States," *Wind Eng.*, vol. 45, no. 4, pp. 857–871, Aug. 2021.
[41] F. X. Turc Castellà, "Operations and maintenance costs for offshore wind farm. analysis and strategies to reduce o&m costs," Master's thesis, Dept. Ind. Eng., Escola Tècnica Superior d'Enginyeria Ind. de Barcelona (ETSEIB), Universitat Politècnica de Catalunya, Barcelona, Spain, 2020.
[42] A. Ata Teneler and H. Hassoy, "Health effects of wind turbines: A review of the literature between 2010–2020," *Int. J. Environ. Health Res.*, vol. 33, no. 2, pp. 143–157, Feb. 2023.
[43] P. Gkeka-Serpetsidaki, S. Papadopoulos, and T. Tsoutsos, "Assessment of the visual impact of offshore wind farms," *Renew. Energy*, vol. 190, pp. 358–370, May 2022.
[44] M. Bilgili and H. Alphan, "Visual impact and potential visibility assessment of wind turbines installed in Turkey," *Gazi Univ. J. Sci.*, vol. 35, no. 1, pp. 198–217, Mar. 2022.
[45] M. A. Mostafa, E. A. El-Hay, and M. M. ELkholy, "Recent trends in wind energy conversion system with grid integration based on soft computing methods: Comprehensive review, comparisons and insights," *Arch. Comput. Methods Eng.*, vol. 30, no. 3, pp. 1439–1478, Apr. 2023.
[46] S. D. Ahmed, F. S. Al-Ismail, M. Shafiullah, F. A. Al-Sulaiman, and I. M. El-Amin, "Grid integration challenges of wind energy: A review," *IEEE Access*, vol. 8, pp. 10857–10878, 2020.
[47] L. Aslam, R. Zou, E. S. Awan, S. S. Hussain, K. A. Shakil, M. A. Wani, and M. Asim, "Hardware-centric exploration of the discrete design space in transformer-LSTM models for wind speed prediction on memory-constrained devices," *Energies*, vol. 18, no. 9, p. 2153, Feb. 2025.
[48] O. Tutsoy and S. Colak, "Adaptive estimator design for unstable output error systems: A test problem and traditional system identification based analysis," *Proc. Inst. Mech. Eng., I, J. Syst. Control Eng.*, vol. 229, no. 10, pp. 902–916, Nov. 2015, doi: 10.1177/0959651815603910.
[49] S. M. Wilcox, "National solar radiation database 1991–2010 update: User's manual," National Renewable Energy Laboratory (NREL), Golden, CO, USA, Tech. Rep. NREL/TP-5500-54824, 2012.
[50] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "ITransformer: Inverted transformers are effective for time series forecasting," 2023, *arXiv:2310.06625*.
[51] F. Biagini, L. Gonon, A. Mazzon, and T. Meyer-Brandis, "Detecting asset price bubbles using deep learning," 2022, *arXiv:2210.01726*.
[52] A. Lawal, S. Rehman, L. M. Alhems, and M. M. Alam, "Wind speed prediction using hybrid 1D CNN and BLSTM network," *IEEE Access*, vol. 9, pp. 156672–156679, 2021.

LAEEQ ASLAM (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from International Islamic University Islamabad, Pakistan, in 2010 and 2017, respectively. He is currently pursuing the Ph.D. degree in control science with Central South University, Changsha, China.

From 2013 to 2019, he was with iUSE-SEMS, where he was a Laboratory Engineer and a Lecturer, teaching courses in microprocessors, digital logic design, and electronic circuit analysis. He supervised numerous student projects related to automation, robotics, and renewable energy systems. From 2019 to 2020, he was a Research Assistant with International Islamic University Islamabad on a Higher Education Commission of Pakistan (HEC) funded project focused on the early detection of breast cancer. Additionally, he was a Machine Learning Engineer with DLision, Wah Cantt, Pakistan, from 2021 to 2022, specializing in training and fine-tuning deep learning models for semantic segmentation, time-series prediction, and model deployment on embedded systems. His research interests include machine learning for edge devices and sustainable systems.

RUNMIN ZOU (Senior Member, IEEE) received the B.Eng. degree in automatic control and the M.Eng. degree in control theory and control engineering from Central South University (CSU), Changsha, China, in 1994 and 1997, respectively, and the Ph.D. degree in automatic control from the Laboratory "Institut de Recherche en Communications et Cybernétique de Nantes," École Centrale de Nantes, France, in 2009.

Since 2005, he has been with the School of Information Science and Engineering and a Predecessor with the School of Automation, CSU, where he has been a Full Professor, since 2012. He has been the Deputy Dean of the School of Automation, since February 2019, and in charge of international affairs. He is currently the Leader of the Institute of Smart Energy and Advance Control, which is mainly engaged in the study of the automation control technology of smart energy systems. He is a reviewer of some top control journals and has finished dozens of industry control projects or products, involving electricity, finance, medical, communications, and other fields. His research interests include the intersection of control theory, artificial intelligence, and power electronics with their applications to renewable energy and complex dynamical systems.

EBRAHIM SHAHZAD AWAN received the Ph.D. degree from International Islamic University, Islamabad.

He is currently a Faculty Member with the School of Engineering, Design and Built Environment, Western Sydney University. He is dedicated to developing advanced control strategies and optimization techniques to improve the reliability, efficiency, and resilience of power systems, especially in the context of integrating renewable energy and smart grid technologies. Throughout his academic career, he has made significant contributions to the field by applying cutting-edge methodologies, such as sliding-mode observers, game theory, and fuzzy logic to power system optimization. His work aims to enhance the performance of power systems by ensuring fault tolerance, improving system stability, and optimizing energy flow in both traditional and renewable energy networks. As a Researcher and an Educator, he is committed to advancing the integration of innovative control techniques into real-world power systems to foster more sustainable and resilient energy solutions. His primary research interests include power system control, optimization, and fault tolerance, with a particular emphasis on microgrid systems and renewable energy sources.

SAYYED SHAHID HUSSAIN received the B.S. degree in telecommunication engineering from Hazara University, Mansehra, in 2016, and the master's degree from the School of Microelectronics, Xi'an Jiaotong University, China, where he gained advanced knowledge in emerging technologies. Currently, he is pursuing the Ph.D. degree with Central South University, focusing on optimization techniques in power systems, with an emphasis on artificial intelligence and machine learning.

His work bridges the fields of telecommunication engineering, AI, and power systems. He is passionate about exploring how machine learning can transform power system operations, fault detection, and overall performance. His research aims to enhance the efficiency, stability, and reliability of electrical grids by integrating AI-driven optimization methods. Actively involved in academic conferences and publications, he strives to advance the role of AI in the energy sector and collaborate with experts to create innovative solutions for the challenges facing modern power infrastructure. His primary research interests include developing intelligent algorithms to optimize power networks, improve system management, and contribute to more sustainable and resilient energy solutions.

MUHAMMAD ASIM received the M.S. degree in mathematics from the University of Peshawar, Peshawar, Pakistan, in 2013, the M.Phil. degree in mathematics from Kohat University of Science and Technology, Kohat, Pakistan, in 2016, and the Ph.D. degree in computer science and technology from Central South University, Changsha, China, in 2022. Currently, he is a Postdoctoral Researcher with EIAS Data Science Laboratory, CCIS, Prince Sultan University, Riyadh, Saudi Arabia. His current research interests include artificial intelligence, computational intelligence techniques, cloud computing, edge computing, 5G/6G communication systems, and autonomous vehicles. He was awarded the Outstanding International Graduate of Central South University, in 2022.

SAMIA ALLAOUA CHELLOUG received the Engineering degree in computer science, the master's degree in computer science, and the Ph.D. degree in networking from the University of Constantine, Algeria, in 2003, 2006, and 2013, respectively. From 2006 to 2013, she was an Assistant Professor with the Faculty of Computer Science, Constantine University. In August 2013, she joined the Department of Networks and Communication Systems, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia, where she is currently an Associate Professor. Her current research interests include wireless sensor networks, networking, the Internet of Things, and artificial intelligence.

MOHAMMED A. ELAFFENDI is currently a Professor of computer science with the Department of Computer Science, Prince Sultan University, where he is also the Director of the Quantum and Intelligent Computing (QIC) Center. He is the former Dean of the College of Computer and Information Sciences (CCIS), the Dean of AIDE, and a Rector with the University. He is the Founder and the Director of the Data Science Laboratory (EIAS) and the Center of Excellence in Cybersecurity. His research interests include data science, intelligent and cognitive systems, machine learning, and natural language processing (NLP).