International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 21, Number 1 (2026) pp. 81-109

https://dx.doi.org/10.37622/IJAER/21.1.2026.81-109

Hybrid CNN–BiLSTM–Attention Model Optimized with HHO–GA for Accurate Short-Term Weather Forecasting

Rahul Agarwal
Department of ECE
Sharda University Agra, Agra
UP, India
rahulagarwal@agra.sharda.ac.in

Choppelli Rajasekhar
Department of ECE
NIILM University, Kaithal
Haryana, India
rajasekhar@niilmuniversity.ac.in

Abstract

In order to manage climate risk, prepare for renewable energy, and keep an eye on the environment, accurate multivariate weather forecasting is required. Optimal hybrid deep learning architecture for weather data capture of both localised spatial patterns and long-range temporal correlations is shown in this paper. The design integrates Networks that use Convolutional Neural Networks (CNNs), Bidirectional Long Short-Term Memory (BiLSTM), and a method for self-attention. For both training and testing, we used a publicly available multivariate meteorological dataset from Kaggle, which has over 96,000 observations and spans a whole decade. Atmospheric pressure, temperature, humidity, and wind speed are the input variables employed by the model. A type of hybrid Harris Hawks Optimization-Genetic Algorithm (HHO-GA) is used to change critical learning hyperparameters, further improving prediction dependability. Compared to baseline architectures, the optimized CNN-BiLSTM-Attention model clearly performs better. Over both high and low variability variables, the model has strong explanatory power with R^2 values of 0.9777 for temperature, 0.9339 for humidity, 0.7320 for wind speed, and 0.6713 for pressure. Corresponding error measures further support its predictive power with RMSE values of 1.3336°C, 0.0479, 3.3745 km/h, and 51.2578 millibars, and MSE values of 1.7784, 0.0023, 11.3872, and 2627.367, respectively. Compared to CNN-LSTM and other reference models, our results demonstrate consistent improvements of 10–25%. When these

factors are addressed, a weather forecasting system that combines evolutionary optimisation using deep spatial-temporal learning is both powerful and widely applicable. Future work may extend this architecture to larger regional or satellite-derived datasets, incorporate physics-informed constraints for improved interpretability, and deploy the model in real-time decision-support systems.

Keywords: Deep Learning, Weather Forecasting, Hybrid Optimization, Spatial–Temporal Modeling, Hyperparameter Tuning, Real-Time Forecasting Systems.

I. Introduction

Weather forecasting is crucial in today's world as it affects everything from transportation and agriculture to public safety and electricity[1];[2];[3].Industries may benefit from improved operating efficiency, reduced risk, and optimized resource utilization thanks to accurate weather predictions, particularly those for the short term [4]Power system security and stability, for example, are advantages of accurate weather predictions in the renewable energy sector, which in turn improves electricity market transactions, solar and wind electricity forecasting, etc.[5]. Additionally, timely warnings of storms, floods, or heatwaves allow authorities to minimize economic damage and perhaps save lives, making accurate weather forecasting crucial for disaster planning and response [6]. Because of the impending drastic changes in the climate [7], accurate and timely weather forecasts are becoming more important, underscoring the necessity for sophisticated prediction techniques to meet these difficulties [8].

The numerical weather prediction (NWP) methods, which provide an approximate solution to the equations explaining atmospheric dynamics, provide the basis of the conventional approach for weather prediction. We compare the current weather forecast with a weather time series forecast using deterministic NWP algorithms. The use of ensemble forecasts to estimate the probability distribution for various future weather scenarios is becoming more common among meteorological institutions[9]There are a lot of NWP-based predictions in these projections, and each one indicates a different possible outcome [10]..Among the many obstacles that NWP models must overcome are fundamental gaps in our knowledge of physical processes, the difficulty of drawing useful conclusions from massive volumes of observational data, and the enormous computing resources that are sometimes required[11]. As a result, creating ensemble weather forecasts using more computationally effective techniques continues to be an important research topic.

New developments within machine learning (ML) have increased the precision and efficiency of computerized weather prediction systems. The use of machine learning to replace antiquated NWP models in weather forecasting has shown encouraging results [12];[13]. Given these rapid developments and the growing need for trustworthy prediction systems, in-depth evaluations are required prior to ML-based forecasting frameworks being operationalized [14]. Accurate modeling is intrinsically

challenging because meteorological systems often exhibit complex variations, primarily represented in time and space [15]. Basic weather predicting tasks have been successfully executed using SVM, LR, RF, and other conventional machine learning approaches. To classify weather conditions and predict continuous variables like temperature or precipitation, for instance, SVM and LR have been trained using historical data[16];[17]. Large-scale, high-dimensional inputs make it difficult for these models to capture more intricate connections in meteorological data, even when they perform well for certain tasks like regression or classification with less datasets [18].

Due to its ability to process massive amounts of data with many dimensions and to accurately depict complex spatiotemporal and nonlinear dynamics, DL and hybrid models have garnered considerable attention for application in weather forecasting, particularly in urban settings. CNNs are excellent at extracting short-term temporal and localized spatial information [19], but Gated Recurrent Units and LSTM networks represent long-range relationships with less complexity [20], [21]. By using self-attention mechanisms, attention-based architectures like Transformers improve feature prioritization [22]. Localized success has been shown by recent research such as Multi-view Stacked CNN-BiLSTM[23], as well as enhanced hybrid DL frameworks[24], however they run into problems with severe event prediction, computing efficiency, and coastal validation.

The complete complexity of atmospheric interactions remains a difficulty for deep learning models, even with advances in weather forecasting. This is particularly true in coastal microclimates and during severe weather events, furthermore, the models aren't very good at transferring results to other types of weather. The hyperparameter sensitivity of the models and the computational efficiency constraints both highlight these limitations. This research proposes a CNN-BiLSTM architecture that is hybrid using a self-attention mechanism as well as an HHO-GA optimizer to overcome these obstacles. Constant and automatic hyperparameter tuning is ensured by the hybrid optimizer, which adaptively highlights the most relevant temporal trends. When you bring all of these things together, you get a short-term weather forecast that is accurate, easy to comprehend, and computationally efficient.

These are the primary findings from this study:

- A CNN-BiLSTM forecasting model is created to adequately capture bidirectional temporal patterns and spatial relationships in meteorological data from several sources.
- A Self-Attention mechanism is incorporated to adaptively weight critical time steps, improving interpretability and enabling the model to focus on influential atmospheric variations.
- A Hybrid HHO–GA optimization algorithm is introduced to automate hyperparameter tuning, improving model convergence, generalization capability, and overall robustness.
- A unified spatiotemporal prediction framework is designed to handle nonlinear and dynamic atmospheric variables, enhancing stability across diverse weather conditions.

- Systematic evaluation demonstrates performance improvements over conventional ML models and standalone DL architectures, showing higher accuracy, adaptability, and reliability.
- An efficient, real-time–ready forecasting system is established to manage practical challenges such as chaotic weather dynamics, high dimensionality, and model sensitivity.

The following sections are organized in a certain way: Section 2 delves into pertinent research and data gaps. In Section 3, the proposed models are detailed. Section 4 explains the method in depth. In Section 5, we describe the experiments, analyses the outcomes, and compare the findings. Section 6 concludes the inquiry and lays out the following steps.

II. Literature Review

Deep learning models, including RNNs and CNNs, have greatly improved weather forecasting because they understand spatial and temporal relationships. A strong tool for weather forecasting, CNNs have emerged for their extraordinary speed within extracting spatiotemporal patterns throughout many applications. Just like every other application, weather forecasting requires the capacity to learn complicated spatiotemporal elements. Scher et al. [25]studied data-driven weather prediction systems, with a focus on CNNs, to determine their performance. Researchers looked at spherical convolution as an alternative to traditional convolution methods and included hemisphere-specific topological features into the network architecture to account for Earth's curvature. When tested using the Weather bench dataset, they discovered that both approaches enhanced forecasting ability, with the greatest results coming from their combination.

Wein et al. [26] used historical data without physical processes to create deep CNN models that forecast basic meteorological fields throughout the Northern Hemisphere. At lead durations of up to three days, the CNNs were able to forecast changes in weather system strength and outperformed simple models. Despite not surpassing operational models, they shown potential for effective ensemble forecasting. However, there are also issues with CNN-based approaches, such as poor performance over longer forecast periods and trouble managing teleconnections between remote grid locations[27]; [18].These constraints prompted the development of hybrid models integrating CNNs with other approaches, new methodologies based on RNNs and transformers, and improved long-distance dependence and long-term prediction performance.

Sequential prediction tasks are a strong suit for RNN, GRU, and LSTM, two improved RNN architectures [28]. In order to establish a standard for seasonal precipitation prediction, Shen Haojun et al. [29]investigated summer precipitation in China using LSTM. When compared to the BCC, The findings demonstrate that the LSTM network has greater prediction skills compared to its back propagation neural network and stepwise regression models. In order to provide a new strategy for research on precipitation prediction, Han Ying et al. [30]proposed an enhanced LSTM model, the LSTM-Weighted Broad Learning System, which integrates the benefits of deep learning and width learning. In order to forecast blizzards, Samy et al[31]suggested an LSTM-based deep learning method that prioritises the detection of

weather patterns via sequential data processing. It has been tested on real-time datasets gathered from the Polar Region and uses bivariate parameters like wind speed along with air pressure to increase the accuracy of extreme weather predictions. [32]used weather station data, radar, lightning, and GPS to train the LSTM encoder-decoder model for forecasting heavy rain and wind. The results showed improved capacity in recognizing severe wind with an accuracy of over 90% and better capability to recognize heavy rain within 30 minutes when utilising four distinct dataset configurations for 1-hour nowcasting employing 10-minute intervals. The preceding twelve hours, [33] used two TensorFlow-implemented Bi-LSTM models to forecast the next 24 hours and 72 hours. Two models were developed; one was to anticipate Kew Gardens temperatures and the other Heathrow air temperatures and relative humidity. However, the recurrent structure still limits these RNN-based techniques; that is, the intrinsic sequential calculation process prevents training from being parallelized, leading to comparatively high temporal complexity and slow training efficiency.

Hybrid deep learning approaches integrate CNNs for spatial feature extraction with LSTM, GRU, or Transformer architectures for temporal sequencing [34]; [35]. Complementing these models, metaheuristic optimizers further enhance forecasting accuracy by efficiently capturing complex spatiotemporal weather dynamics through effective parameter tuning [36]. The datasets, major contributions, and limitations of contemporary hybrid DL and optimization-based approaches used in weather forecasting are compiled in Table 1.

TABLE 1.RECENT HYBRID DL & OPTIMIZATION METHODS FOR WEATHER FORECASTING

Study	**Model / Approach**	**Dataset**	**Key Contributions**	**Limitations**
Houssein et al. [37]	MGO (Mountain Gazelle Optimizer)–GA– Deep LSTM; MGO–GA–Deep GRU.	Climate data from 9 Egyptian cities	Hybrid evolutionary optimization; lowest RMSE (1.75); improved robustness	Region-specific; high computational load
Band et al. [38]	Time-Varying Filter Empirical Mode Decomposition (TVFEMD) + Gradient Boosting + LSTM; TVFEMD–LSTM	Wind speed data (Carbondale, Champaign, DeKalb)	Two-stage hybrid; RMSE as low as 0.056; performance improves with lag	Multi-stage model increases complexity

Zhang et al.[39]	CNN–BiLSTM–Attention Mechanism	ERA5 + observational data	Improves severe-weather forecasts: TS ↑ 15.46% (rainfall), 10.28% (thunderstorms), 12.47% (hail), 14.76% (gusts) over human forecasts	Requires significant resources, limiting scalability
Bashir et al. [40]	Correlation Analysis: Enhanced Empirical Mode Decomposition of the Whole Ensemble with Adaptive Noise-HHO–Spatiotemporal Attention + Sequence-to-Sequence	Wind speed data	Lowest RMSE (0.639); strong feature decomposition + optimization	Very computationally intensive
Gong et al. [35]	CNN–LSTM	Global temperature dataset	MAE improved to 0.901; stable prediction across regions	Data imbalance (North America-heavy)
Bahmani et al. [41]	Artificial Neural Network (ANN)+ Reptile Search Algorithm; ANN + Particle Swarm Optimization; ANN + Whale Optimization Algorithm	Streamflow datasets from rivers in Urmia	RSA boosted ANN accuracy; RMSE 1.65	PSO outperforms RSA in some stations
Sen et al. [42]	GA, DE (Differential Evolution), PSO applied to	Short-term weather datasets	DE achieved best MAPE; metaheuristics outperform	No new forecasting architecture introduced

	ANN, ARIMA (Autoregressive Integrated Moving Average), GRU, LSTM.		manual tuning	

A. ***Research gap:***

Although there have been great strides in weather forecasting using CNNs and deep learning, RNNs, LSTMs, GRUs, and hybrid models, existing approaches still face critical limitations. CNNs struggle with long-range dependencies and teleconnections, while RNN-based models suffer from sequential-processing bottlenecks and high computational cost. While CNN-LSTM and CNN-BiLSTM hybrid designs enhance spatiotemporal learning, their flexibility in a variety of climates is diminished since they often depend on simple or single optimizers with little hyperparameter adjustment. Additionally, there aren't many studies that integrate global optimization and attention processes, and none that we are aware of provide a cohesive, computationally effective framework that simultaneously enhances interpretability, feature extraction, temporal modeling, and robust hyperparameter tuning. These gaps underscore the need of a comprehensive system that integrates CNN–BiLSTM, Self-Attention, and a hybrid HHO–GA optimizer to provide short-term weather forecasting that is more accurate, adaptable, and scalable.

III. Background

The study used DL and optimisation models such as CNN, Bi-LSTM networks, HHO, and GA, which are detailed in this part. We can optimize model parameters to increase prediction performance and find spatial-temporal trends in weather data by combining these strategies.

A. ***Convolutional Neural Network***

Typical convolutional neural network (CNN) models include of convolutional, pooling, flattening, along with fully connected layers, as stated in reference[43] To see these layers, look at Figure 1. The convolutional layer implements weight sharing and sliding windows to decrease computational complexity. It is the primary component from a convolutional neural network. With the input data, we use the kernel approach to extract different attributes at this layer. Now we move on to the pooling layer. By running each feature map separately and reducing the number of links between them, this layer helps keep the feature map size down. Successful model training relies on reducing dimensionality and extracting dominating characteristics, both of which pooling aims to accomplish[44]. Both maximum and average pooling are among the several pooling strategies that may be used. It is necessary to employ the flattening layer to produce a one-dimensional vector before moving on to the fully connected linked (FC) layer, which consists of neurons, biases, and weights to connect the neurons among the next layers. A frequent approach in CNN networks is to add the FC layer to be the final layer before the output layer.

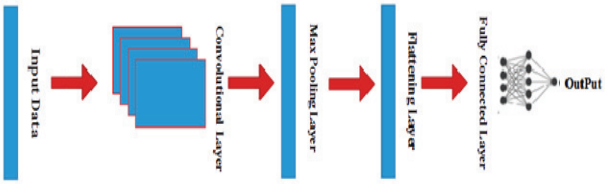

Fig 1 Typical CNN Model

The convolutional neural network (CNN) model implemented in this study includes convolutional, pooling, flattening, along with completely connected layers. The operations have the following mathematical definitions:

Input Layer:

The meteorological feature map that was entered is shown as:

$$X \in \mathbb{R}^{H \times W \times C} \quad (1)$$

In this case, Cstands for the sum of all feature channels, whereas H and W denote the spatial dimensions of the input.

Convolutional Layer:

The extraction of spatial information is done using a set of learnable kernels, K. The output of the convolution is provided by:

$$F = \text{ReLU}(X * K + b_1) \quad (2)$$

where * denotes convolution, b_1 is the bias term, while ReLU is the activation function.

Max Pooling Layer:

In order to preserve important properties while reducing spatial dimensions, a pooling kernel with stride size s of size $n \times n$ is used:

$$P = \max(F) \quad (3)$$

Layer of Flattening:

A one-dimensional feature vector is created from the pooled feature maps:

$$v = \text{Flatten}(P) \quad (4)$$

Dense Layer:

The compressed vector is mapped to a dense hidden representation using weight matrix $W_1$and bias b_2:

$$h = \text{ReLU}(v \cdot W_1 + b_2) \quad (5)$$

Output Layer:

A Softmax function converts hidden representations into class probabilities:

$$\hat{Y} - \text{Softmax}(W_2 \cdot h + b_3) \quad (6)$$

Here, $\hat{Y}$denotes the probability distribution over the output classes.

In this study, the CNN highlights the essential spatial patterns from the meteorological feature maps, yielding a compact representation that encompasses the localized atmospheric behavior. The Bi-LSTM then takes these spatial features for temporal modeling.

*B. **Bidirectional Long Short-Term Memory***

Two versions of the present RNN—the forward LSTM and the reverse LSTM—are combined in a Bi-LSTM model. One solution to the issue of gradient vanishing in classic RNNs is the gating unit, which is an integral part of long short-term memory (LSTM). Because of this, the LSTM improves the performance of sequential learning models by detecting and capitalizing on long-distance relationships.

A storage unit (c_t) an input gate (i_t), an output gate (o_t) a forget gate (f_t), and so on.are the four main components of the new structure that each cell unit in LSTM contains. One LSTM module's internals are shown in Figure 2.

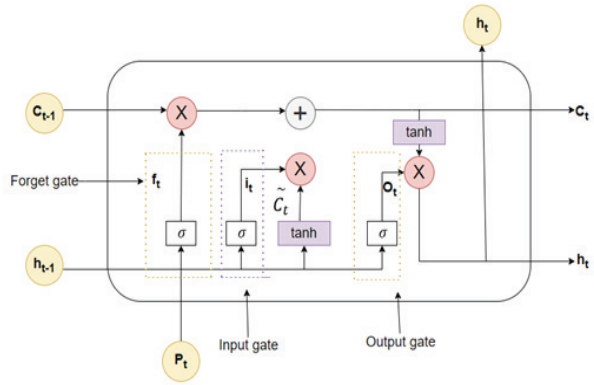

Fig 2 LSTM cell unit

The following is the formula for the LSTM update:

Forget the gate mechanism:

$$f_t = \sigma\left(W_{(f)}x_t + U_{(f)}h_{t-1} + b_{(f)}\right) \quad (7)$$

The forget gate resets a section of the memory cellf_t. A symbolσ stands for the Sigmoid activation function. The input at time t, represented byx_t in the weight matrix $W_{(f)}$,is located in$U$.This is the hidden layer's output weight matrix, h_{t-1} is the hidden state, as well as stands for short-term memory. A vector containing offsets, denoted as $b_{(f)}$.

Mechanism of input gate:

$$i_t = \sigma(W_{(i)}x_t + U_{(i)}h_{t-1} + b_{(i)} \quad (8)$$

The $i_t$input gate It stands for the memory cell's input gates. The symbolσ stands for the Sigmoid activation function. The input at a given time is represented by x_t in the weight matrix $W_{(i)}.U$ Is the buried layer's weight matrix h_{t-1} Does the concealed state stand for working memory, and is the offset vector$b_{(i)}$?

Candidate state:

$$\tilde{c} = f_t \odot c_{t-1} \text{ (3)} \quad (9)$$

$\tilde{c}$This cell might be used as a memory. The long-term memory is represented by the cell state of$t-1$which is the forget gate f_t which acts as the reset memory cell c_{t-1}.

Cell state update:

$$c_t = \tilde{c} + i_t \odot tanh(W_{(c)}x_t + U_{(c)}h_{t-1} + b_{(c)}) \quad (10)$$

In this case, a candidate's memory cell is denoted as $\tilde{c}$, and $c_t$store data for the long term. The input gates regulate the input to the memory cell and are represented by asi_t.The input at a given time is represented by x_t in the weight matrix $W_{(c)}$. UCan we view the weight matrix of the buried layer? The condition of short-term memory that is veiled is h_{t-1}). The vector denoting offset is $b_{(c)}$.

mechanism for the output gate:

$$o_t = \sigma(W_{(o)}x_t + U_{(o)}h_{t-1} + b_{(o)}) \quad (11)$$

Each memory cell has its own set of output gates, denoted by o_t. σ Stands for the sigmoid activation function Inputs are represented by weight matrices $W_{(o)}$, where x_t is the current time UIs. Its weight matrix, h_{t-1} for hidden layer output, stands for short-term memory $b_{(o)}$, and it reflects the hidden state andis the vector of offset.

Hidden state update:

$$h_t = o_t \odot tanh(c_t) \quad (12)$$

$h_t$is the concealed state, The cell state representing long-term memory, represented by c_t, is the output gate that controls the output as the memory cell, and anh stands for the Sigmoid activation function. [45] proposed that Bi-LSTM networks capture both backward and forward temporal features more effectively than single-direction LSTM architectures. For weather forecasting, where atmospheric variables exhibit strong bidirectional dependencies across time (e.g., prior-day humidity influencing future rainfall while future pressure trends reflect emerging systems), such dual-direction learning is essential.

Therefore, this study employs a Bi-LSTM network to more effectively learn long-range temporal patterns in meteorological variables, enabling improved representation of sequential atmospheric behaviours. By integrating Bi-LSTM with a CNN feature extractor, the model jointly learns spatial–temporal relationships crucial for short-term weather prediction.

The implementation of backward and forward contextual interactions in the Bi-LSTM network employed in this study is shown in Figure 3. The Bi-LSTM works by mimicking the CNN's deep spatial properties' temporal dynamics; this makes the network more resilient to changes in weather and diverse areas, and it also improves its fitting capabilities.

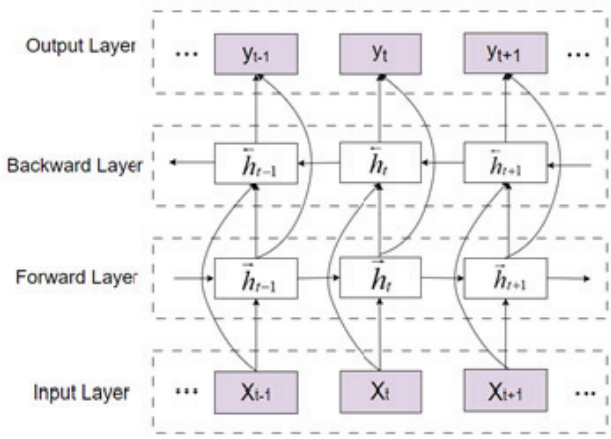

Fig 3 Bi-LSTM network model

The hidden states, both forward and backward, are calculated as:

$$\vec{h}_t = \text{LSTM}(x_t, \vec{h}_{t-1}) \quad (13)$$

$$\overleftarrow{h}_t = \text{LSTM}(x_t, \overleftarrow{h}_{t+1}) \quad (14)$$

The final Bi-LSTM output at time t is obtained by concatenation:

$$h_t = [\vec{h}_t \parallel \overleftarrow{h}_t] \quad (15)$$

The final output state is determined by concatenating forward and backward hidden states, enabling richer temporal representation.

The Bi-LSTM learns the forward and backward dependencies in weather sequences, thereby modeling the CNN-extracted features temporal trends, which consequently results in the together spatial-temporal learning boosting the short-term meteorological forecasting accuracy.

C. *Attention mechanism*

This attention mechanism relies on the fact that our brains are good at prioritizing and processing information depending on its relative importance[46].This study led to the development of an attention mechanism, which allows for the assignment of different weight coefficients for the information recovered by the LSTM. This mechanism was established in order to further assign these features. The ability of the model to generate predictions was thus enhanced as a result of this situation. Figure 4 provides a pictorial picture of the Attention structure that may be seen. The LSTM learns t time-related feature vectors, and the Self-Attention module is in charge of applying t shared biases and weights to them. Following that, the computer is trained with the help of these feature vectors. In order to carry out deep-level feature mining, the weighted summation method was decided upon as the most appropriate strategy. Following the completion of the independent calculation of the weight of each characteristic, this was carried out. For the purpose of calculation, the formula that is as follows is utilized:

$$\beta_i = \sigma(W_i h_t + b_i) \quad (16)$$

$$a_i = softmax(\beta_i) = \frac{exp(\beta_i)}{\sum i\, exp(\beta_i)} \quad (17)$$

$$O = H \otimes a_i \quad (18)$$

The activation function is represented by β_i, the feature's relevance is represented by a_i, the attention weight is represented by O, and the output prediction result is represented by W_i, b_i,Additionally, the bias vector and weight matrix that connect the neuronal nodes σ, respectively.

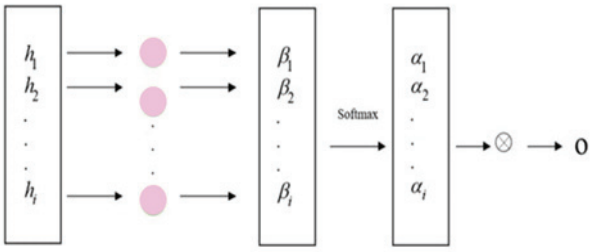

Fig4 Attention network structure diagram

The Attention mechanism in this study refines the temporal representations learned by the Bi-LSTM by selectively emphasizing the most influential atmospheric patterns. By weighting critical time-dependent features, it ensures that only the most relevant meteorological signals are forwarded for subsequent optimization and forecasting.

D. *The algorithm of Harris Hawks*

The population-based metaheuristic algorithm known as Harris Hawks Optimisation (HHO) draws its inspiration upon the cooperative hunting techniques used by the stunning and accomplished ravens [47]. HHO also, as many metaheuristic (MH) algorithms, alternates during the process of defining the global maxima and minima between two opposite kinds of movements that of exploring and that of exploiting. In this case, every hawk among the population symbolizes a solution candidate for the optimization problem.

Exploration Phase

The hawks are placed uniformly over the search surface at the outset of optimization in order to explore more. The new location for each hawk is determined as shows:

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 \mid X_{rand}(t) - 2r_2 X(t) \mid, & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3 (LB + r_4 - (UB - LB)), & q < 0.5 \end{cases} \quad (19)$$

Here, $X(t)$is the hawk's position at time t, as well as$X_{rabbit}(t)$ is the solution the group has identified so far, which is the location of the prey. The position of one hawk at random is denoted by $X_{rand}(t)$, whereas an average position among all the hawks within the group is denoted by $X_m(t)$ is The parameters r_1, r_2, r_3, r_4, and $q$are uniformly distributed random integers between $(0,1)$, and they introduce a random

element into the search process. The penultimate step in optimisation is for UB and LB to establish the lower along with upper bounds of the decision variables. This keeps the hawk locations within the acceptable search zone.

The average hawk position is computed as:

$$X_m(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \quad (20)$$

Transition to Utilization.

Based upon the prey's escape energy (E), an algorithm transitions with exploration to exploitation:

- $|E| \geq 1$: exploration
- $|E| < 0.5$: exploitation

Exploitation Phase

When exploitation begins, hawks perform either hard besiege or soft besiege depending on E and random parameters.

- **Hard Besiege**

$$X(t+1) = X_{\text{rabbit}}(t) - E\,|\Delta X(t)| \quad (21)$$

- **Soft Besiege**

A more adaptive strategy employing a trial vector Y and Levy-flight–based abrupt dives:

$$Y = X_{\text{rabbit}}(t) - E\,|JX_{\text{rabbit}}(t) - X(t)| \quad (22)$$

$$Z = Y + S\,LF(D) \quad (10)$$

where S is a random vector that adds random variation to the search, and $LF(D)$ is the Lévy flight function which employs an heavy-tailed Lévy distribution to create step sizes. and D indicates the problem dimension, which determines the length of the search vector.

Lévy flight is defined as:

$$LF(x) = 0.01\,\frac{u\sigma}{|v|^{1/\beta}} \quad (23)$$

$$\sigma = \left(\frac{\Gamma(1+\beta)\sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2})\beta 2^{\frac{\beta-1}{2}}}\right)^{1/\beta} \quad (24)$$

where $u, v \in (0,1)$ and $\beta = 1.5$.

The hawk's final updated position is:

$$X(t+1) = \begin{cases} Y, & \text{if } F(Y) < F(X(t)) \\ Z, & \text{if } F(Z) < F(X(t)) \end{cases} \quad (25)$$

The HHO algorithm plays the role of the primary optimizer in the tuning of the CNN–Bi-LSTM model through the adaptive balancing of exploration and exploitation. The smooth transition from soft to hard besiege phases guarantees a fast convergence to high-quality forecasting parameters and at the same time prevents getting stuck in local minima.

E. Genetic Algorithm

Metaheuristic algorithms like Genetic Algorithm (GA) draw inspiration from natural selection, the driving force behind biological evolution as proposed by Charles Darwin. Using an evolutionary process analogous to that of a population's most successful members, GA repeatedly generates potential solutions to achieve optimum or near-optimal outcomes for optimisation problems with constraints and without[48].

The evolution in a GA begins with a random collection of people (solutions) that make up the population. During each generation, the fitness of every member in the population is assessed, and a subset of the current population is selected for the following generation based on their fitness levels. Upon reaching an acceptable fitness level for the population or the maximum number of generations, the program terminates. What follows is an explanation of the two main operations—mutation and crossover—that a GA does to generate a new population:

- Mutation: A chromosome is a series of bits that encodes a whole population. Mutation is the process of altering the value of one bit on one chromosome at random.
- Crossover: Picking two people at random from a population, finding a random spot, and then crossing their tails.

Algorithm 1 presents the Genetic Algorithm's summarized pseudocode., and the overall workflow is illustrated in Fig. 5**.**

Algorithm 1:AlgorithmGenetic

Input: Size of Population N,Chromosome Length$L$, Termination Standard

Output:

Best Variable 1 Use N random individuals to start the population;
2 When the Termination Criteria is not fulfilled
3 For each individual $x_i$in the population, calculate the fitness $f(x_i)$.
4 Choose parents p_1, p_2among the population by using a suitable selection technique;
5 Apply consistent crossover and mutation procedures to the chosen parents to create a new population. c_i;
6substitute the new population for the existing o ne;
7**end**
8return the population's finest individual;

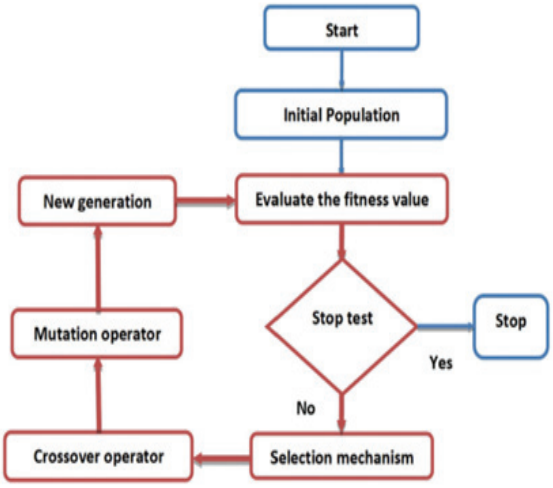

Fig 5 The GA's flowchart

In an effort to further improve optimization stability, the GA is used in conjunction with HHO, this permits population-level evolution and, by extension, search space variety. Combining GA's global recombination with HHO's adaptive hunting, the framework gets stronger and more general CNN-Bi-LSTM parameters, thus enabling the same reliable performance over different climate scenarios and forecasting conditions.

IV. Methodology

The methodological framework applied in this study involves a structured sequence of operations, which includes data gathering, preprocessing, feature normalization, and the construction of temporal sequences. Based on this, a DL architecture was established, which comprised CNNs for local feature extraction, Bi-LSTM layers for longer-distance temporal dependency capture, and a self-attention mechanism that enhanced the model's ability to identify the most critical phases of a meteorological sequence. Using HHO-GA, hyperparameters were fine-tuned to enhance the model's performance. Overfitting may be avoided, early-stop regularization was also utilized. Training, validating, and comparing the model to a collection of baseline designs constitute the final element of the technique. The second algorithm officially summarizes the whole operational process of the hybrid forecasting system that is being suggested. In Figure 6, we can see the whole workflow of the hybrid forecasting framework that has been suggested.

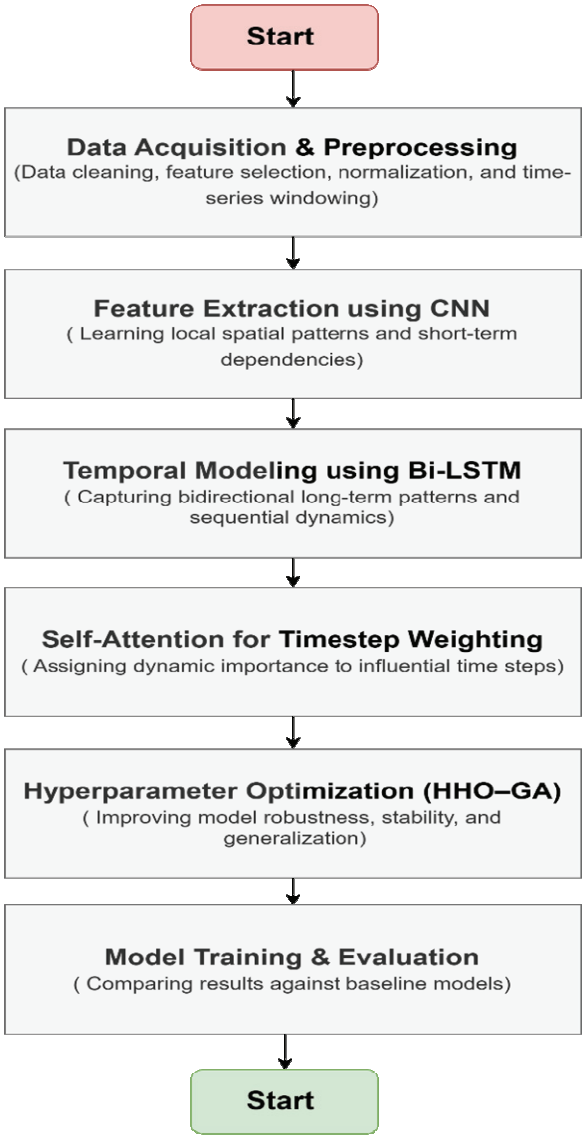

Fig 6The Proposed Framework's Overview

A. Data Gathering and Dataset Overview

This open-source Weather Dataset upon Kaggle, which is available online, provided the weather data utilised in this study:https://www.kaggle.com/datasets/muthuj7/weather-dataset).

There is a high temporal resolution in this dataset that is appropriate for both long-term climate pattern study and short-term forecasting; it comprises hourly weather observations covering a period of ten years, between 1 April 2006 to 9 September 2016.

The dataset is appropriate for data-driven meteorological modelling because to its large amount of information, which includes 96,453 entries with timestamps. The four most significant meteorological variables in the whole dataset are temperature (°C), relative humidity (%), wind speed (km/h), and air pressure (millibars). These

variables include a wide variety of temporal behaviours, including slow-changing ones like pressure to fast-changing ones like wind speed.

The dataset is provided in CSV format, allowing seamless integration with preprocessing workflows and deep learning pipelines. Distributed under Kaggle's standard license, it supports academic and research usage with appropriate citation. This dataset's multivariate structure, persistent long-term trends, and abundant temporal information make it perfect for training and testing the proposed hybrid deep learning forecasting methodology.

B. Data Preprocessing

The raw meteorological data required systematic preprocessing in order to establish trustworthiness, facilitate model learning, and maintain the time series aspect that is necessary for DL-based forecasting. The preprocessing pipeline involved data cleaning, feature normalization, and supervised sequence construction.

- **Data Cleaning:** The dataset taken in its raw form had a few cases of unreadable data, noisy sensors, and inconsistencies with time stamps. To preserve temporal structure, the following operations were applied:
  - **Missing Values:** Filled using linear interpolation

$$x_t = x_{t-1} + \frac{x_{t+1} - x_{t-1}}{2} \qquad (26)$$

Here, x represents the value of any meteorological variable (e.g., temperature, humidity, pressure) at time t. This approach preserves smooth transitions between consecutive measurements.

  - **Outliers:** The rolling median filter technique was applied to smooth values outside the range of ±3σ, which helped to reduce noise while keeping the natural variability intact.
  - **Duplicate Timestamps:** Removed in order to keep the strict chronological ordering necessary for sequence modeling.
- **Feature Scaling:** To keep the gradient updates the same across CNN, Bi-LSTM, and attention layers, Min-Max scaling was applied to the numerical features in the whole dataset:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \qquad (27)$$

The scaling guaranteed standardized feature ranges, which helped in the convergence and had the side effect of requiring less time for the hyperparameter optimization process.

- **Time-Series Windowing**

The continuous data stream was turned into supervised learning samples by using a sliding-window technique:

$$X = [x_{t-n}, \ldots, x_t], y = x_{t+\Delta} \qquad (28)$$

In this case, n represents the input sequence length and Δ stands for the anticipated time. The model was able to acquire knowledge of both the short as well as long-range temporal dependencies—crucial for accurate weather forecasting—through this method.

C. *Proposed Hybrid CNN–BiLSTM–Self-Attention Model*

The proposed forecasting system combines a series of deep learning components in one structure such as CNN, inputs that may represent the temporal and spatial dynamics of meteorological time-series data, as well as BiLSTM units along with a scaled dot-product self-attention mechanism. The method starts with the use of a CNN, which is able to extract short-range spatial dependencies from the normalized input sequences. This stage applies a convolutional transformation of the type

$$F^{(l)} = \sigma(W^{(l)} * X + b^{(l)}), \qquad (29)$$

Using the l-th convolution layer's learnable biases and weights, represented by $W^{(l)}$ and $b^{(l)}$, the operationalconvolution denoted by $*$, a input tensor X, along with the activation function of ReLU denoted by σ.CNNs through this process promote noise suppression, localized variation and the extraction of informative representations which reflect the most important short-term meteorological changes. These maps of features serve as the purified spatial input for the next phase of temporal modeling.

In order to identify the long-range sequential relationships, the model employs a Bidirectional LSTM once the spatial characteristics have been extracted. Both the forward and backward LSTMs read the sequence in sequence, but only forward LSTM reads it from beginning to finish, thus they complement each other in their temporal modelling efforts. The transitions of backward and forward concealed states can be expressed as

$$\vec{h}_t = f_{\rightarrow}(x_t, \vec{h}_{t-1}), \overleftarrow{h}_t = f_{\leftarrow}(x_t, \overleftarrow{h}_{t+1}), \qquad (30)$$

where $f_{\rightarrow}(\cdot)$ and $f_{\leftarrow}(\cdot)$ denote the forward and backward LSTM transformation functions. The combined temporal encoding at each timestep is then defined as

$$h_t = \vec{h}_t \oplus \overleftarrow{h}_t, \qquad (31)$$

with $\oplus$ indicating concatenation along the feature dimension.

This bidirectional arrangement gives the model the capability to pick up contextual signals from both the preceding and succeeding places in the sequence leading to its capacity to grasp non-linear temporal patterns that are typical of atmospheric processes. To augment the model's strength in detecting the prime moments in the sequence, the Bi-LSTM outputs are fed into a scaled dot-product self-attention mechanism. The said mechanism builds query, key, and value projections that are amenable to learning and calculates attention weights according to

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (32)$$

layer Attention boosts or diminishes certain timesteps according to their importance relative to each other, thus allowing the model to hone in on sudden changes or key transitions that have a major impact on the precision of weather prediction. By means of spatial extraction, bidirectional temporal learning, and selective temporal emphasis working together, the hybrid CNN–BiLSTM–Attention architecture offers a complete representation of meteorological sequences prior to the optimization stage.

D. Hybrid HHO–GA Hyperparameter Optimization

The success of DL models is largely determined by the hyperparameter settings, and the traditional manual tuning is in most cases inefficient and likely to yield suboptimal results. To overcome this drawback, the proposed approach combines the global search and the local refinement by using a hybrid HHO–GA method. The optimization process is guided by a composite objective function that aims to reduce the total prediction error, which is represented as

$$\mathcal{J} = \alpha\,\mathrm{MAE} + \beta\,\mathrm{RMSE} \qquad (33)$$

Here, the two weighting coefficients dictate the trade-off between the size of the error reduction and the extent of the punishment for highly deviated values. Hence, this formulation guarantees that the optimization procedure enhances not only the overall performance but also the resistance to large errors.

The optimization starts with the HHO's exploration phase, where the potential solutions are updating their locations depending on the hawk's metaphor of trying to encircle and catch the prey unexpectedly. The position update is carried out according to the formula

$$X_{t+1} = X_{\text{rabbit}} - E\ |\ JX_{\text{rabbit}} - X_t\ | \qquad (34)$$

where X_t represents the solution at the present moment, $X_{\text{rabbit}}$is the optimal solution found until iteration t, E stands for the escaping energy of the animal, and J represents the strength of the jump. This kind of behavior allows the optimizer to go through the whole hypterparameter domain and thus prevents it from getting trapped in local minima early on.

When the escaping energy is lower and the algorithm has reached the exploitation phase, the Genetic Algorithm is applied to enhance the local search. During this phase, GA applies crossover by mixing two parent solutions through a weighted sum, which is mathematically expressed as

$$X_{\text{child}} - \lambda X_{p1} + (1-\lambda)X_{p2} \qquad (35)$$

resulting in an offspring solution that carries traits from both parents. To keep the genetic pool varied and avoid being stuck, a mutation operator introduces random perturbations to the offspring in a controlled manner through

$$X_{\text{mut}} = X_{\text{child}} + \delta\mathcal{N}(0,1) \qquad (36)$$

where δ controls the extent of mutation as well as the normally distributed noise term N (0,1). Optimizers may refine their best solutions more effectively using GA within the exploitation phase, while keeping the required variation for strong convergence.

HHO's exploration dynamics and GA's exploitation capabilities work together in a way that the hybrid optimization algorithm continuously makes its way through the hyperparameter landscape. This approach guarantees that the forecasting model is trained under the best-tuned configuration, thus, enhancing predictive performance and allowing the architecture to adapt well to new meteorological patterns.

E. Training Procedure

The model was trained in a temporally consistent manner to allow realistic forecasting behavior. The data was divided as follows: 70% for training, 20% for validation, and 10% for testing. In this approach, the model might store knowledge

gained from past observations and use it to analyses fresh data. A loss function called Mean Squared Error (MSE) was used to update the network weights, and the Adam optimizer was used to fine-tune the training. Hyperparameters like as learning rate and batch size may be externally modified with the help of the hybrid HHO-GA architecture. Training became more consistent as a result of our ability to methodically sample the search area. When the validation loss levelled off, training was terminated using an early-stopping criterion. By doing so, we ensured that the model could avoid overfitting while maintaining good generalizability.

F. Performance Evaluation Metric

The proposed forecasting model was tested for its predictive accuracy using three standard regression measures: coefficient of determination, root mean squared error (RMSE), and mean absolute error (MAE)(R^2).The measures that measure relative prediction accuracy, quantify average deviation, and penalize significant mistakes, respectively, provide further insights into the performance of the model. Their mathematical equations are expressed as

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} | y_i - \hat{y}_i |, \quad (37)$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad (38)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \quad (39)$$

In this context, $y_i$stands for the real value, $\hat{y}_i$ for the expected value, as well as N for an overall count of assessments. Together, these evaluation metrics enable a robust assessment of model accuracy, error distribution, and overall forecasting reliability.

Algorithm 2 Hybrid CNN-BiLSTM-Attention Weather Forecasting

Require: Raw meteorological dataset D

Ensure: Trained forecasting model with optimized parameters

Step1: Data Preprocessing

1: Load dataset $D$with timestamped records

2: Handle missing values: $x_t = \frac{x_{t-1}+x_{t+1}}{2}$

3: Remove outliers using rolling median filter (±3σ)

4: Apply Min-Max scaling: $x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$

5: Generate sequences: $X = \{x_{t-L+1}, \ldots, x_t\} \Rightarrow y_t = x_{t+\Delta}$

Step 2: Model Architecture Setup

6: Initialize CNN layers with ReLU activation

7: Compute: $f^{(i)} = \sigma(W^{(i)} * x + b^{(i)})$

8: Initialize Bi-LSTM layers

9:Compute forward: $h_f = f_f(x_t, h_{t-1})$

10:Compute backward: $h_b = f_b(x_t, h_{t+1})$

11:Concatenate: $h_t = h_f \oplus h_b$

12: Apply self-attention: $\text{Attention}(Q,K,V)\text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$

Step 3: Hyperparameter Optimization

13: Define objective $J = \alpha \cdot \text{MAE} + \beta \cdot \text{RMSE}$

14: Initialize HHO-GA hybrid optimizer

15: While not converged

16:HHO exploration: $X_{t+1} = X_{\text{rabbit}} - E \mid X_{\text{rabbit}} - X_t \mid$

17:GA crossover: $X_{\text{child}} = \lambda X_p + (1-\lambda)X_q$

18:GA mutation: $X_{\text{mut}} = X_{\text{child}} + \delta N(0,1)$

19:Evaluate fitness using objective function J

20:Update best solution X_{rabbit}

21: End while

Step 4: Model Training

22: Split data: 70% train, 20% validation, 10% test

23: Initialize model with optimized hyperparameters

24: Loss function: $\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2$

25: Configure Adam optimizer with tuned learning rate

26: For each epoch do

27:Forward pass through CNN-BiLSTM-Attention

28:Compute gradients and update weights

29:Monitor validation loss for early stopping

30: End for

Step 5: Performance Evaluation

31:Compute MAE: $\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} \mid y_i - \hat{y}_i \mid$

32:Compute MSE: $\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2$

33:Compute RMSE: $\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}$

34: Compute $R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2}$

35: Compare against baseline models

36: return Trained model and performance metrics

V. Result and Discussion

The proposed hybrid CNN–BiLSTM–Attention model which has been optimized using the HHO–GA algorithm was tested on the test dataset for performance evaluation and then compared to a number of established deep learning architectures. The evaluation was based on a set of quantitative performance metrics, which comprised the R^2, RMSE, and MSE, as well as visual comparisons of real and forecasted meteorological parameters. The results have shown that the optimized

hybrid model is superior in predictive accuracy with respect to all the studied variables.

A. ***Proposed Model's Prediction Performance.***

Initially, the model's prediction capabilities were tested by comparing the actual and anticipated weather conditions, including humidity, temperature, wind speed, along with atmospheric pressure. Figures 7–10 show the results of a few test samples comparing the predicted and ground-truth values. Figure 7 shows that the model can understand complicated thermal dynamics since it is quite accurate at capturing both sudden changes in temperature and more steady trends. Predictions for humidity (Figure 8) are similarly quite comparable; in fact, the only instance of notable variation happens whenever the moisture content undergoes a rapid shift. The model also does a good job of predicting wind speeds, which exhibit the quick oscillations seen in short-term wind variability (Figure 9). Figure 10 shows that the pressure predictions are constant and trustworthy, even when the dataset has naturally low fluctuation and sometimes unusual drops.

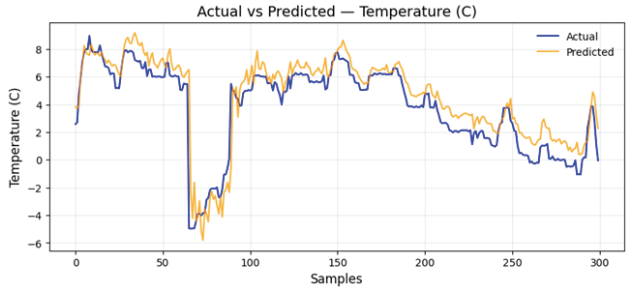

Fig 7Current and Forecasted Trends in Temperature

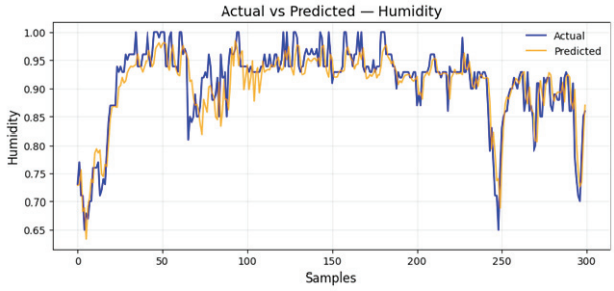

Fig 8Patterns of actual and anticipated humidity

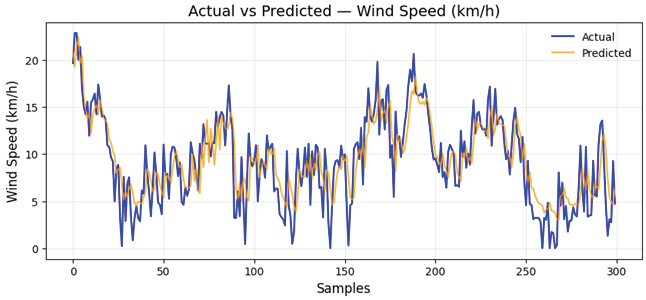

Fig 9Variations in wind speed, both actual and expected

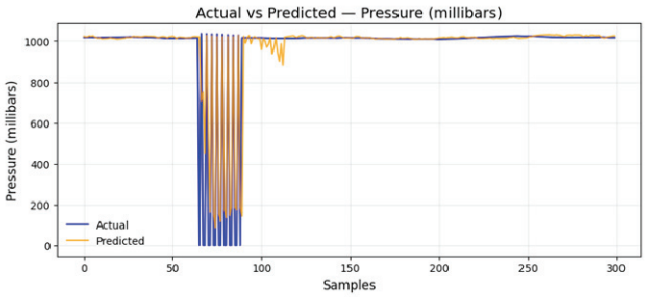

Fig 10Profiles of atmospheric pressure, both actual and anticipated

The quantitative results matched these visual patterns. R^2 values of 0.9777 for temperature, 0.9339 for humidity, 0.7320 for wind speed, and 0.6713 for pressure were obtained using the suggested model. In both high-variance and low-variance atmospheric circumstances, the model's resilience was shown by the low RMSE and MSE values that persisted across all variables. Table 2 summarizes all of the measurements that were collected.

TABLE 2 FORECASTING PERFORMANCE OF THE PROPOSED MODEL

Parameter	**R^2**		**RMSE**	**MSE**
Temperature (°C)	0.9777		1.3336	1.7784
Humidity	0.9339		0.0479	0.0023
Wind Speed (km/h)	0.732		3.3745	11.3872
Pressure (millibars)	0.6713		51.2578	2627.367

B. *Comparative Evaluation with Baseline Models*

The productivity of the proposed hybrid model was evaluated by comparing it to three well-established deep learning models in the industry: CNN-LSTM, CNN-BiLSTM-Attention, along with the GA-Deep GRU model. According to the comparison result shown in Table 3, the HHO-GA optimisation, when integrated into the CNN-BiLSTM-Attention framework, uniformly improves all of the meteorological parameters.

During temperature forecasting, the CNN-BiLSTM-Attention model improved over the CNN-LSTM baseline model, bringing the R^2 value up to 0.9689. A small gap separated the preceding models from GA-Deep GRU, even though the latter had a R^2 of 0.9724. In spite of this, the suggested optimised hybrid model beat the competitors, boasting the best R^{2_2} value of 0.9777 along with the lowest RMSE as well as MSE values. With R^2rising from 0.9127 in the CNN-LSTM model to 0.9339 within the optimised hybrid one, the trend of humidity prediction was also comparable.

Parameter variability in wind speed forecasting is high, which makes the differences across baseline models even more apparent. The CNN-LSTM model gets an $R^2$of 0.7041, then gradually gets to 0.7256 with GA-Deep GRU. The suggested model once more manages to get the best result of R^2equal to 0.7320. Atmospheric pressure, which has the least variability and also has some anomalous segments, is the most difficult predictor variable in the entire modeling process. On the other hand, out of all the architectures tested, the suggested hybrid model still managed to get the best R^2value (0.6713) and the fewest errors.

TABLE 3. COMPARATIVE ANALYSIS OF FORECASTING MODELS' PERFORMANCE

Parameter	Model	R²	RMSE	MSE
Temperature (°C)	CNN–LSTM	0.9612	1.8424	3.3924
	CNN–BiLSTM–Attention	0.9689	1.6212	2.6282
	GA–Deep GRU.	0.9724	1.5096	2.279
	CNN–BiLSTM-Attention with HHO–GA Optimization	**0.9777**	**1.3336**	**1.7784**
Humidity	CNN–LSTM	0.9127	0.0593	0.0035
	CNN–BiLSTM–Attention	0.9224	0.0536	0.0028
	GA–Deep GRU.	0.9281	0.0508	0.0025
	CNN–BiLSTM-Attention with HHO–GA Optimization	**0.9339**	**0.0479**	**0.0023**
Wind Speed (km/h)	CNN–LSTM	0.7041	3.6892	13.6102
	CNN–BiLSTM–Attention	0.7188	3.5121	12.3351
	GA–Deep GRU.	0.7256	3.4267	11.7503

	CNN–BiLSTM-Attention with HHO–GA Optimization	**0.732**	**3.3745**	**11.3872**
Pressure (millibars)	CNN–LSTM	0.6124	59.2213	3507.142
	CNN–BiLSTM–Attention	0.6358	55.8045	3113.256
	GA–Deep GRU.	0.6532	53.9044	2906.631
	CNN–BiLSTM-Attention with HHO–GA Optimization	**0.6713**	**51.2578**	**2627.367**

Discussion

The findings validate that the combination of attention-driven timestep weighting, bidirectional temporal encoding, and convolutional spatial extraction significantly improves forecasting accuracy. Through the methodical adjustment of hyperparameters that control learning dynamics, the incorporation of HHO–GA optimization further enhances model stability. The steady improvement in every meteorological variable shows how the suggested framework can be adjusted to both characteristics with naturally low variation (like pressure) and those with fast fluctuations (like wind speed). Furthermore, the comparison findings show that the suggested model retains resistance against noise and aberrant environmental patterns often seen in real-world datasets, in addition to capturing fine-grained temporal alterations. The importance of integrating DL with evolutionary optimization techniques is further shown by the better generalization performance across baseline comparisons. For multivariate meteorological forecasting, the optimized CNN–BiLSTM–Attention model demonstrates a powerful, scalable, and highly generalizable architecture, laying a solid basis for next developments in data-driven atmospheric prediction.

VI. Conclusion

This study presents a novel hybrid DL framework for multivariate weather forecasting; it combines a self-attention mechanism, convolutional neural networks (CNNs), and long short-term memory (BiLSTM) units with a hybrid HHO-GA optimisation strategy. The model was able to capture context-sensitive timestep relevance, long-range temporal interdependence, and localized spatial aspects within complicated atmospheric sequences according to the methodological approach. In order to improve convergence stability and forecast reliability across all weather

variables, the HHO–GA optimizer was essential in helping to refine the hyperparameter space.

According to the empirical findings, the suggested CNN–BiLSTM–Attention model with HHO–GA optimization continuously performed better than baseline models like CNN–LSTM, CNN–BiLSTM–Attention, and GA–Deep GRU. Considerable improvements in R^2, Measures of root-mean-square (RMSE) along with mean-squared error (MSE) over humidity, temperature, wind speed, along with pressure are signs of the architecture's resilience and usefulness. The ability of the model to handle both highly dynamic as well as generally changing variables demonstrated its strong adaptability to different climate patterns.Overall, the study establishes that a strong and scalable method for precise weather forecasting is provided by combining deep feature extraction with optimization strategies inspired by biology. The results provide opportunities to expand the framework to real-time monitoring applications, domain-specific decision-support systems, and more general climate analytics.

References

[1] M. R. Bendre, R. C. Thool, and V. R. Thool, "Big data in precision agriculture: Weather forecasting for future farming," in *2015 1st international conference on next generation computing technologies (NGCT)*, IEEE, 2015, pp. 744–750.

[2] R. Meenal *et al.*, "Weather forecasting for renewable energy system: a review," *Arch. Comput. Methods Eng.*, vol. 29, no. 5, pp. 2875–2891, 2022.

[3] D. Xue, L. Wu, T. Xu, C. Wu, Z. Wang, and Z. He, "Space weather effects on transportation systems: A review of current understanding and future outlook," *Sp. Weather*, vol. 22, no. 12, p. e2024SW004055, 2024.

[4] H. Saima, J. Jaafar, S. Belhaouari, and T. A. Jillani, "Intelligent methods for weather forecasting: A review," in *2011 national postgraduate conference*, IEEE, 2011, pp. 1–6.

[5] H. Zhang, M. Zhang, R. Yi, Y. Liu, Q. H. Wen, and X. Meng, "Growing Importance of Micro-Meteorology in the New Power System: Review, Analysis and Case Study," *Energies*, vol. 17, no. 6, p. 1365, 2024.

[6] M. K. Linnenluecke, A. Griffiths, and M. Winn, "Extreme weather events and the critical importance of anticipatory adaptation and organizational resilience in responding to impacts," *Bus. Strateg. Environ.*, vol. 21, no. 1, pp. 17–32, 2012.

[7] J. W. Day and C. Hall, "Global climate change: a warmer and more unpredictable future," in *America's Most Sustainable Cities and Regions: Surviving the 21st Century Megatrends*, Springer, 2016, pp. 137–166.

[8] H. Zhang, Y. Liu, C. Zhang, and N. Li, "Machine learning methods for weather forecasting: A survey," *Atmosphere (Basel).*, vol. 16, no. 1, p. 82, 2025.

[9] T. Palmer and R. Hagedorn, *Predictability of weather and climate*. Cambridge University Press, 2006.

[10] [N. Roberts *et al.*, "IMPROVER: the new probabilistic postprocessing system at the Met Office," *Bull. Am. Meteorol. Soc.*, vol. 104, no. 3, pp. E680–E697, 2023.

[11] X. Ren *et al.*, "Deep learning-based weather prediction: a survey. Big Data Res 23: 100178," 2020.
[12] N. Singh, S. Chaturvedi, and S. Akhter, "Weather forecasting using machine learning algorithm," in *2019 International Conference on Signal Processing and Communication (ICSC)*, IEEE, 2019, pp. 171–174.
[13] L. Chen *et al.*, "FuXi: a cascade machine learning forecasting system for 15-day global weather forecast," *npj Clim. Atmos. Sci.*, vol. 6, no. 1, p. 190, 2023.
[14] O. C. Pasche, J. Wider, Z. Zhang, J. Zscheischler, and S. Engelke, "Validating deep learning weather forecast models on recent High-Impact extreme events," *Artif. Intell. Earth Syst.*, vol. 4, no. 1, p. e240033, 2025.
[15] F. Zhang, "Weather Forecasting and Analysis with LSTM Based on Deep learning," in *Proceedings of the 2024 2nd International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2024)*, Springer Nature, 2024, p. 165.
[16] D. Dhilipkumar, P. S. Y. Bala, T. Yogeswaran, and K. Vanitha, "Real time weather prediction system using ensemble machine learning," in *2023 Second international conference on augmented intelligence and sustainable systems (ICAISS)*, IEEE, 2023, pp. 477–484.
[17] M. Holmstrom, D. Liu, and C. Vo, "Machine learning applied to weather forecasting," *Meteorol. Appl*, vol. 10, no. 1, pp. 1–5, 2016.
[18] J. Zhang, B. Wang, M. Hua, Z. Chen, S. Liang, and X. Kang, "Spatiotemporal meteorological prediction based on fully convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, 2024.
[19] J. Wang, L. Jin, X. Li, S. He, M. Huang, and H. Wang, "A hybrid air quality index prediction model based on CNN and attention gate unit," *IEEE Access*, vol. 10, pp. 113343–113354, 2022.
[20] A. Mishra and Y. Gupta, "Comparative analysis of Air Quality Index prediction using deep learning algorithms," *Spat. Inf. Res.*, vol. 32, no. 1, pp. 63–72, 2024.
[21] Z. He *et al.*, "Gated recurrent unit models outperform other Machine learning models in prediction of minimum temperature in greenhouse Based on local weather data," *Comput. Electron. Agric.*, vol. 202, p. 107416, 2022.
[22] Z. Zhang and S. Zhang, "Modeling air quality PM2. 5 forecasting using deep sparse attention-based transformer networks," *Int. J. Environ. Sci. Technol.*, vol. 20, no. 12, pp. 13535–13550, 2023.
[23] S. Kumar and V. Kumar, "Multi-view Stacked CNN-BiLSTM (MvS CNN-BiLSTM) for urban PM2. 5 concentration prediction of India's polluted cities," *J. Clean. Prod.*, vol. 444, p. 141259, 2024.
[24] Y. Han *et al.*, "A short-term wind speed prediction method utilizing novel hybrid deep learning algorithms to correct numerical weather forecasting," *Appl. Energy*, vol. 312, p. 118777, 2022.
[25] S. Scher and G. Messori, "Spherical convolution and other forms of informed machine learning for deep neural network based weather forecasts," *arXiv Prepr. arXiv2008.13524*, p. 9, 2020.
[26] J. A. Weyn, D. R. Durran, and R. Caruana, "Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from

historical weather data," *J. Adv. Model. Earth Syst.*, vol. 11, no. 8, pp. 2680–2693, 2019.
[27] A. Araujo, W. Norris, and J. Sim, "Computing receptive fields of convolutional neural networks," *Distill*, vol. 4, no. 11, p. e21, 2019.
[28] Z. Karevan and J. A. K. Suykens, "Spatio-temporal stacked LSTM for temperature prediction in weather forecasting," *arXiv Prepr. arXiv1811.06341*, 2018.
[29] H. Shen, Y. Luo, Z. Zhao, and H. Wang, "Prediction of summer precipitation in China based on LSTM network," *Clim. Chang. Res.*, vol. 16, no. 3, pp. 263–275, 2020.
[30] Y. Han, J. Guan, Y. Cao, and J. Luo, "Application of improved LSTM-WBLS model in daily precipitation forecast," *Nanjing Xinxi Gongcheng Daxue Xuebao*, vol. 15, no. 2, pp. 180–186, 2023.
[31] V. S. Samy and V. Thenkanidiyoor, "Blizzard prediction in antarctic meteorological data using deep learning techniques," *J. Intell. Fuzzy Syst.*, vol. 45, no. 4, pp. 6797–6812, 2023.
[32] S. Chkeir, A. Anesiadou, A. Mascitelli, and R. Biondi, "Nowcasting extreme rain and extreme wind speed with machine learning techniques applied to different input datasets," *Atmos. Res.*, vol. 282, p. 106548, 2023.
[33] G. Zenkner and S. Navarro-Martinez, "A flexible and lightweight deep learning weather forecasting model," *Appl. Intell.*, vol. 53, no. 21, pp. 24991–25002, 2023.
[34] M. Fan, O. Imran, A. Singh, and S. A. Ajila, "Using cnn-lstm model for weather forecasting," in *2022 IEEE International Conference on Big Data (Big Data)*, IEEE, 2022, pp. 4120–4125.
[35] Y. Gong, Y. Zhang, F. Wang, and C.-H. Lee, "Deep learning for weather forecasting: A cnn-lstm hybrid model for predicting historical temperature data," *arXiv Prepr. arXiv2410.14963*, 2024.
[36] [S. Mittal and O. P. Sangwan, "Metaheuristic Techniques Optimised LSTM Network for Improved Weather Prediction," in *2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, IEEE, 2024, pp. 1–6.
[37] E. H. Houssein, M. Dirar, A. A. Khalil, A. A. Ali, and W. M. Mohamed, "Optimized deep learning architecture for predicting maximum temperatures in key Egyptian regions using hybrid genetic algorithm and mountain Gazelle optimizer," *Artif. Intell. Rev.*, vol. 58, no. 9, p. 258, 2025.
[38] S. S. Band *et al.*, "A two-stage deep learning-based hybrid model for daily wind speed forecasting," *Heliyon*, vol. 11, no. 1, 2025.
[39] J. Zhang, M. Yin, P. Wang, and Z. Gao, "A method based on deep learning for severe convective weather forecast: CNN-BILSTM-AM (version 1.0)," *Atmosphere (Basel).*, vol. 15, no. 10, p. 1229, 2024.
[40] H. Bashir, M. Sibtain, Ö. Hanay, M. I. Azam, and S. Saleem, "Decomposition and Harris hawks optimized multivariate wind speed forecasting utilizing sequence2sequence-based spatiotemporal attention," *Energy*, vol. 278, p. 127933, 2023.

[41] M. J. Bahmani, Z. Kayhomayoon, S. G. Milan, F. Hassani, M. Malekpoor, and R. Berndtsson, "Development of a new hybrid model to enhance streamflow estimation using artificial neural network and reptile search algorithm," *Sci. Rep.*, vol. 15, no. 1, p. 6098, 2025.
[42] A. Sen, A. R. Mazumder, D. Dutta, U. Sen, P. Syam, and S. Dhar, "Comparative evaluation of metaheuristic algorithms for hyperparameter selection in short-term weather forecasting," *arXiv Prepr. arXiv2309.02600*, 2023.
[43] V. Suresh, P. Janik, J. Rezmer, and Z. Leonowicz, "Forecasting solar PV output using convolutional neural networks with a sliding window algorithm," *Energies*, vol. 13, no. 3, p. 723, 2020.
[44] S. Aslam, H. Herodotou, S. M. Mohsin, N. Javaid, N. Ashraf, and S. Aslam, "A survey on deep learning methods for power load and renewable energy forecasting in smart microgrids," *Renew. Sustain. Energy Rev.*, vol. 144, no. March, 2021, doi: 10.1016/j.rser.2021.110992.
[45] T. Xu, Y. Du, C. Fu, and C. Xie, "Incorporating forward and backward instances in a bi-lstm-cnn model for relation classification," in *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, IEEE, 2018, pp. 2133–2137.
[46] K. Mohiuddin *et al.*, "Retention Is All You Need," *Int. Conf. Inf. Knowl. Manag. Proc.*, no. Nips, pp. 4752–4758, 2023, doi: 10.1145/3583780.3615497.
[47] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Futur. Gener. Comput. Syst.*, vol. 97, pp. 849–872, 2019.
[48] J. H. Holland and C. X. X. C. X. Vx, "Genetic Algorithm," vol. 267, no. 1, pp. 66–73, 1992.