JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 11 Number 1 | June 2026: 126-141
DOI: 10.15575/join.v11i1.1689

CPSO-LSTM: Chaotic Particle Swarm Optimization improved LSTM Hyperparameters for Air Pollution Prediction

Tri Andi[1,3], Andri Pranolo[2], Amelia Ritahani Ismail[3], Candra Juni Cahyo Kusuma[4]
[1]Information Technology, Faculty of Engineering, University of Muhammadiyah Yogyakarta, Indonesia
[2]Department of Informatics, Universitas Ahmad Dahlan, Indonesia
[3]Department of Computer Science, International Islamic University Malaysia, Malaysia
[4]Universitas PGRI Yogyakarta, Indonesia

Article Info

Article history:

Received June 12, 2025
Revised October 20, 2025
Accepted October 30, 2025
Published April 25, 2026

Keywords:

Air Quality Prediction
Chaotic Optimization
Hyperparameter Tuning
Long Short-Term Memory Networks
Particle Swarm Optimization
Time Series Forecasting

ABSTRACT

Accurate air pollution predictions are crucial for public health and environmental management, but achieving high prediction accuracy remains a challenge due to the complexity of temporal patterns in pollution data. This study aims to improve performance of Long Short-Term Memory (LSTM) by optimizing hyperparameters tuning based Chaotic Particle Swarm Optimization (CPSO) for air pollution predictions. Hyperparameter optimization included the number of hidden layers, neurons, activation functions, loss functions, optimizers, batch sizes, and epochs. The proposed model LSTM-CPSO compared to other models, baseline LSTM and PSO-LSTM, to predict the concentrations of PM2.5, PM10, NO2, SO2, CO, and O3 in Jakarta based on Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) metrics. The experimental results show that CPSO-LSTM achieves superior performance with MSE 0.012105, MAE 0.086356, RMSE 0.110022, and MAPE 32.31%, outperforming the baseline LSTM by 38.1% on the MSE metric and 11.9% on MAPE. Interestingly, LSTM-CPSO produces better architecture with 2 hidden layers and 91 neurons than LSTM-PSO that requires 7 hidden layers with 51 neurons. Similarly, LSTM-CPSO has shortest 5 training epochs better than LSTM-PSO with 16 epochs. This research demonstrates that chaos-based metaheuristic optimization can select the best hyperparameters to improve the performance of LSTM for air quality forecasting.

Corresponding Author:

Andri Pranolo,
Department of Informatics, Universitas Ahmad Dahlan
Ki Ageng Pemanahan Street, Kragilan, Tamanan, Banguntapan District, Bantul Regency, Special Region of Yogyakarta 55191.
Email: andri.pranolo@tif.uad.ac.id

1. INTRODUCTION

Jakarta, as the Special Capital Region (DKI), has the highest population density in Indonesia. Based on the latest data, the population of Jakarta reached almost 11 million people, with a density of 16,000 people per square kilometer [1]. On weekdays, the population increased due to commuter traffic and vehicles from buffer areas such as Bogor, Depok, Tangerang, and Bekasi. This high density is inseparable from Jakarta's role as the center of national government, economy, and culture. Further, Jakarta has primary sources of pollution from motor vehicle emissions, industrial activities, and coal-fired power plants. Even more concerning, PM2.5 levels in Jakarta are recorded as 23 times above the World Health Organization (WHO) safe threshold [2]. These conditions trigger some problems,

especially air pollution, which has reached alarming levels. In 2023, Jakarta was listed among the cities with the worst air quality globally. The health impacts include increased cases of asthma, ARI(Acute Respiratory Tract Infection), and various other respiratory diseases. Centre for Research on Energy and Clean Air (CREA) analysis estimates that air pollution in Jakarta causes around 2,000 premature deaths per year, with economic losses reaching 1.1 billion USD [3].

To address this problem, an accurate air quality prediction system is needed to support mitigation efforts. However, air quality prediction is a complex challenge as it involves many interdependent variables. Machine learning-based approaches, especially for time series data, offer a more precise solution. Various machine learning algorithms, such as Long Short-Term Memory (LSTM) [4]–[8], CRANE [9]–[11] , ARIMA, Artificial Neural Network (ANN) [12], Support Vector Machine [13], [14], and CNN [10], [15]–[18] have proven effective for time series prediction. In this study, LSTM is chosen as the main model due to its ability to overcome the vanishing gradient problem via backpropagation through time [19]. Hochreiter and Schmidhuber [20] developed LSTM as a refinement of the recurrent neural network (RNN), which has limitations in storing long-term information. The advantages of LSTM, including its computational efficiency and gradient handling, make it widely applied in various fields of time series prediction.

Several empirical studies prove the effectiveness of LSTM. Dhakal et al. [21] reported satisfactory accuracy in PM2.5 prediction, while Mussumeci and Coelho [22] reported that LSTM outperformed other methods in predicting dengue fever cases in Brazil. LSTMs have also proven successful in predicting some subjects, such as agriculture [23], energy [24], weather [25], public health[26], power grid [27], air pollution[28], and stock prediction [29], [30]. The findings of Das et al. [31] reinforce that LSTM achieves the best accuracy in the RMSE metric compared to regression models, Hidden Markov models, and other conventional machine learning methods. Nevertheless, LSTM implementation still has room for improvement. Some studies, such as [32], noted that in certain cases, LSTM may be less accurate than a basic CNN. To optimize performance, some researchers developed hybrid models by combining LSTM and optimization algorithms[33]–[35] . One of the main challenges in LSTM implementation is determining optimal hyperparameters, where conventional randomized approaches often yield suboptimal accuracy. Although LSTM has a powerful architecture, its optimal performance depends heavily on proper hyperparameter configuration. Hyperparameters such as the number of hidden layers[36], the number of neurons[37], the activation function[38], the optimizer[39], the batch size[40], and epochs[41] can affect the model's ability to learn complex patterns in air pollution data. The manual hyperparameter tuning process is not only time-consuming but also often does not yield an optimal configuration.

Particle Swarm Optimization (PSO) has been known as one of the effective metaheuristic algorithms for hyperparameter optimization in various machine learning applications [42]. PSO simulates the swarm intelligence behavior of a group of particles moving through the search space to find the optimal solution [43]. However, conventional PSO may prematurely converge to a local optimum, especially in complex search spaces [44]. To overcome this limitation, Chaotic PSO (CPSO) has been proposed as a variant of PSO that integrates chaos mapping to increase search diversity and avoid local optima [45]. CPSO uses chaos factors to introduce controlled variability in the search process, thus exploring the search space more effectively.

This study aims to evaluate the performance of the proposed LSTM-CPSO air pollution prediction and compare it with the Baseline LSTM without hyperparameter optimization and the LSTM with hyperparameter optimization using PSO (PSO-LSTM). The main contribution of this research is a comprehensive evaluation of the effectiveness of both metaheuristic optimization methods in improving LSTM prediction accuracy for air pollution applications.

2. METHOD

This study uses an experimental approach to compare the effectiveness of three air pollution prediction models: the proposed CPSO-LSTM, the baseline LSTM, and the PSO-LSTM [46]. The research methodology (Figure 1) consists of five main phases that are interrelated: (1) data collection and preprocessing, (2) LSTM architecture, (3) hyperparameter optimization using PSO and CPSO, and (4) performance evaluation. Each phase is designed to address specific challenges in air pollution time series prediction.

p-ISSN: 2528-1682
e-ISSN: 2527-9165

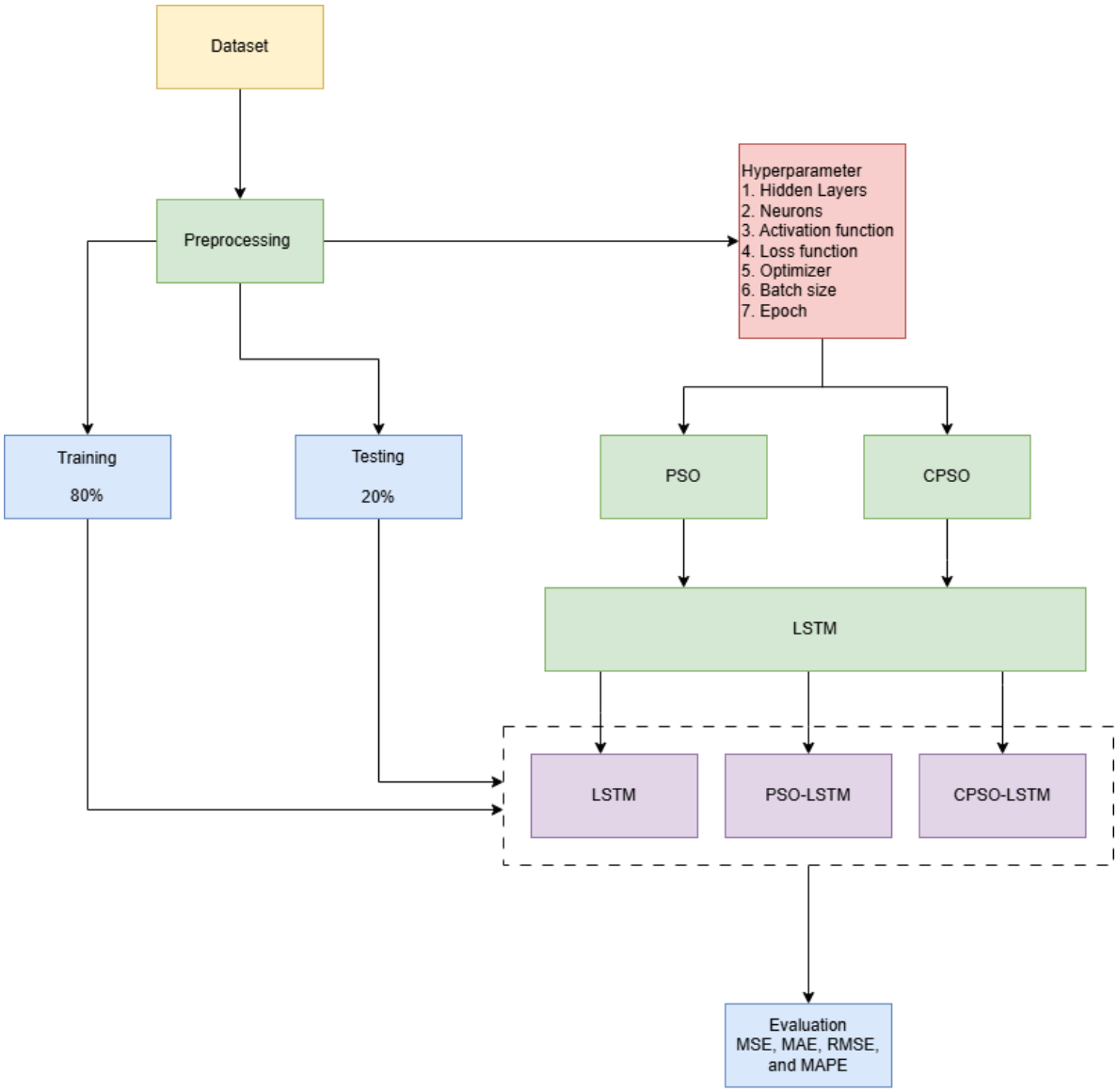

Figure 1. Research methodology

2.1. Data Collection and Preprocessing

This study utilizes a dataset (Table 1) containing Air Quality Index (AQI) or Air Pollution Standard Index (ISPU) records collected from air quality monitoring stations in the Jakarta Special Capital Region from 2010 to 2025. Figure 2 shows a feature visualization of the dataset, which covers key air pollution parameters, including PM2.5, PM10, NO2, SO2, CO, and O3.

Table 1. Variable Table

Variable Name	Role	Type	Description	Unit	Missing value
Date	Features	Date	Date when AQI was measured		No
Station	Features	Characters	Monitoring station name		No
PM10	Features	Integer	Concentration of particles with a diameter of 10 micrometers or less (PM10)	$\mu g/m^3$	No
PM25	Features	Integer	Concentration of particles with a diameter of 2.5 micrometers or less (PM2.5)	$\mu g/m^3$	Available

Variable Name	Role	Type	Description	Unit	Missing value
SO2	Features	Integer	Sulfur dioxide (SO2) concentration	ppm	No
CO	Features	Integer	Carbon monoxide (CO) concentration	ppm	No
O3	Features	Integer	Ozone concentration (O3)	Ppm	No
NO2	Features	Integer	Nitrogen dioxide (NO2) concentration	ppm	No
Max	Features	Integer	Maximum value recorded among pollutants for a specific date and station		No
Critical	Features	Category	The pollutant that has the highest concentration on that date and station		No
Category	Features	Categories	Air quality category based on the 'max' value. This category describes the level of air quality		No

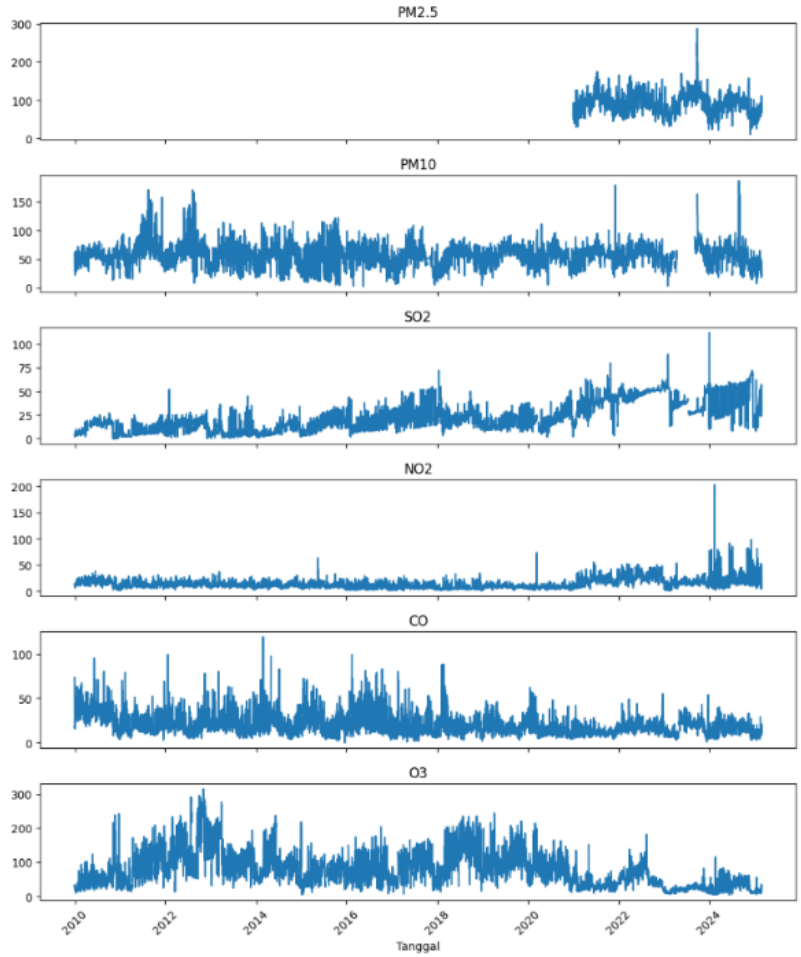

Figure 2. Dataset Visualization

Before modeling, a data preprocessing stage was conducted, comprising several main steps. First, data cleaning is performed to handle missing values. Next, the data were normalized using Min-Max scaling to a uniform range. The data is then transformed into a supervised learning format by applying a sliding window approach to prepare it as input for the time series prediction model. For model training and evaluation, the dataset was divided into two subsets: 80% for training and 20% for validation. This division is done sequentially to ensure an effective model development process.

2.2. LSTM Architecture

Long Short-Term Memory (LSTM) is a neural network architecture specifically designed to overcome the vanishing gradient problem in traditional RNNs. LSTM has a cell-state structure that allows the model to remember long-term information and forget irrelevant information via three main

p-ISSN: 2528-1682
e-ISSN: 2527-9165

gates: the forget gate, the input gate, and the output gate. The following table contains the LSTM baseline parameters (Table 2).

Table 2. LSTM baseline parameters

Parameter	Value
LSTM Units	50
Dropout Rate	0.2
Learning Rate	0.001
Batch Size	32
Optimizer	Adam
Loss Function	MSE

2.3. Hyperparameter optimization using PSO and CPSO

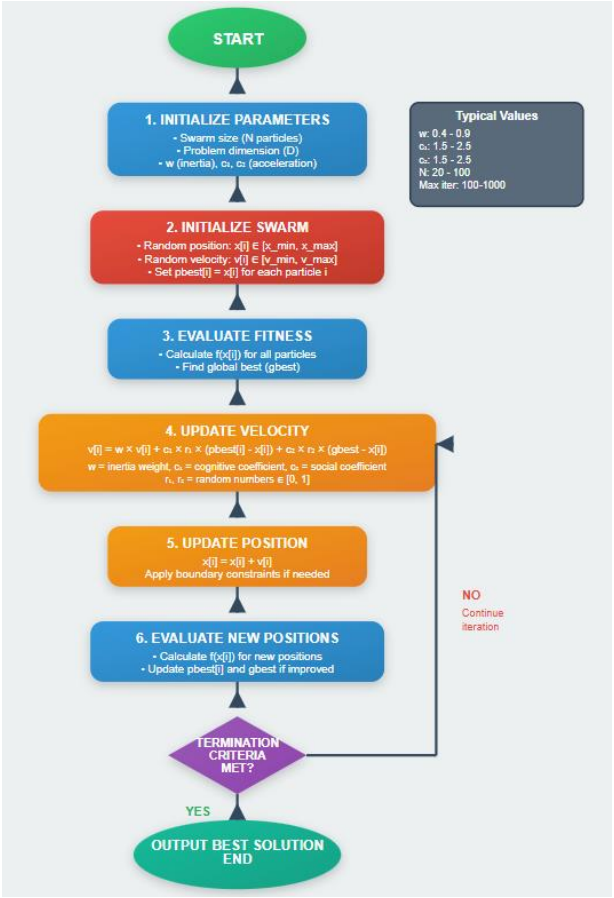

Figure 3. PSO Algorithm

Figure 4. CPSO algorithm

In this study, the search space for LSTM hyperparameter tuning using PSO and CPSO is shown in Table 3. The hyperparameter search for the LSTM model was conducted over a search space of 7 parameters, including the number of hidden layers and neurons, ranging from 2 to 10 layers and 1 to 100 neurons, respectively, to capture both shallow and deeper network configurations. Activation functions (Tanh and Sigmoid), loss functions (MSE and MAE), and optimizers (Adam and RMSprop)

represent widely adopted choices in LSTM-based time-series forecasting. The batch size was tested using two discrete values (32 and 64). The number of training epochs was varied continuously from 5 to 100 to observe the model's convergence behavior under short and extended training regimes.

Table 3. Hyperparameter search of the LSTM method

No	Hyperparameters	Search Space	Type
1.	Hidden Layers [36]	[2,10]	Continuous
2.	Neurons [37]	[1,100]	Continuous
3.	Activation function [38]	Tanh, Sigmoid	Discrete with step=1
4.	Loss function [19]	MSE, MAE	Discrete with step=1
5.	Optimizer [39]	Adam, RMSprop	Discrete with step=1
6.	Batch size [40]	[32,64]	Discrete with step=1
7.	Epoch [41]	[5,100]	Continuous

Particle Swarm Optimization (PSO)

PSO (Figure 3) is a metaheuristic optimization algorithm inspired by the social behavior of a group of birds or fish [47]. Each particle in the swarm has an updated position and velocity based on its own personal best and the best experience of the entire swarm (global best).

Main PSO Formula:

1. Update Velocity:

$$v_i^{t+1} = w.v_i^t + c_1.r_1.\left(p_{best_i} - x_i^t\right) + c_2.r_2.(g_{best} - x_i^t) \quad (1)$$

Description:
v_i^{t+1} = particle velocity i at iteration t+1
w= inertia weight, typically 0.4–0.9
c_1= cognitive parameter (usually 2)
c_2= social parameter (usually 2)
$r_{1,}, r_2$= random number in [0,1]
p_{best_i}= personal best position
g_{best}= global best position
x_i^t= position of particle i at iteration t

Component Explanation:

a. Inertia Term ($w \cdot v_i^t$)

This component represents the particle momentum from previous iterations. Inertial weights w (usually range from 0.4-0.9) serve to:

- Maintains the particle velocity of previous iterations
- Balancing global exploration (broad search) and local exploitation (deep search)
- Higher values encourage global exploration, while lower values encourage local convergence w
- By maintaining momentum, the particles will not change direction too drastically, maintaining continuity of motion

b. Cognitive Term ($c_1 \cdot r_1 \cdot (p_{best_i} - x_i^t)$)

This cognitive term reflects the personal experience of the particles. These components:

- Pushing the particle back to the best position it has ever found (p_{best_i})
- $c_1$is a cognitive parameter (usually worth 2), controlling the force of attraction against a person's best position
- $r_1$is a random number in the range [0,1], giving stochasticity to the movement
- $(p_{best_i} - x_i^t)$is the vector of the distance between the current position and the personal best position
- The greater this difference, the stronger the drive to return to the best position

c. Social Term ($c_2 \cdot r_2 \cdot (g_{best} - x_i^t)$)

This social term captures the collective influence of swarms. These components:

- Pushes the particles towards the best position found by the entire swarm (g_{best})
- $c_2$is a social parameter (usually worth 2), controlling the force of attraction against the global best position

- $r_2$is an independent random number in the range [0,1], giving randomness to the direction of movement
- $(g_{best} - x_i^t)$is the distance vector between the current position and the best global position
- This component facilitates the "exchange of information" between particles indirectly

2. Update Position:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

This equation is a simple kinematic rule that updates the position of a particle based on its velocity. More specifically:

- $x_i^{t+1}$is the position of the particles i on the iteration $t+1$
- $x_i^t$is the position of the particle i in the iteration (previous position) it
- $v_i^{t+1}$is the rate that has been updated on the iteration $t+1$
- Position updates are done by adding a velocity vector to the current position

This equation is consistent with the basic physics principle, where the change in position is equal to the velocity multiplied by time (in this case, time is one iteration). The new position of the particle determines the value of the objective function to be evaluated in the next iteration.

Chaotic Particle Swarm Optimization (CPSO)

CPSO (Figure 4) is a variant of PSO that integrates chaos mapping to enhance exploration[48]. The chaos factor (CF) introduces controlled variability into the search process, helping the algorithm avoid local optima.

Main CPSO Formula:

1. Addition of Chaotic Map:
A commonly used chaotic map is the Logistic Map:

$$z_{k+1} = \mu \,.\, z_k \,.\, (1 - z_k) \quad (3)$$

Description:
z_k= chaotic value at iteration k
μ= chaos control parameters, generally between 3.57–4
z_0= initial chaotic conditions, between (0,1)

Component Explanation:

a. Logistic Map as a Chaos Generator

Logistic maps are one of the simplest yet most useful chaos maps in the literature of nonlinear dynamics. Its functions in CPSO are:

- Generates chaotic value sequences that are highly sensitive to initial conditions
- The chaos value (z_{k+1}) is calculated based on the previous value (z_k) in a nonlinear way
- Generate values that are unpredictable but remain limited in the range [0,1]
- Gives each particle a unique "chaos perturbation" based on its own chaos map history

Characteristics of Logistic Map:

- Deterministic (calculated with certainty from previous iterations) but produces a seemingly random output
- Sensitive dependence on initial conditions: small differences in initial values z_0 result in very different chaos sequences
- Chaotic behavior occurs in a certain parameter regime (especially when) $\mu \in [3.57,4]$

b. Chaos Control Parameter (μ)

These chaos control parameters set the level of chaos in the system:

- Optimal range: Typically between 3.57 to 4
  - $\mu < 3{,}57$: The system is periodic, not chaotic (lack of exploration)
  - $\mu \in [3{,}57,4]$: The system is chaotic with intermittency (maximum exploration)
  - $\mu > 4$: Chaos value out of range [0,1] (invalid)

- General setting: Frequently used values $\mu = 3{,}99$ or $\mu = 4$ to get strong chaos
- Function: Control the "intensity" of chaotic explorations added to the algorithm
- Trade-off: Higher values μ provide wider exploration but may delay convergence

c. Early Chaos Conditions (z_0)

The initial value of chaos is the starting point of the logistic map iteration:

- Range: Usually randomly selected in intervals [0.1]
- Independence per particle: Each particle can have a different z_0, or all particles share the same chaos sequence
- Importance: The initial conditions determine the order of chaos that will be generated in each iteration
- Best practice: Often use a value of $z_0 = 0{,}5$ or select randomly to increase diversity.

d. Chaotic Value (z_k)

The chaos value generated on the iteration to k:

- It is the output of the logistic map that will be integrated into the velocity update equation
- These values have a chaotic nature: non-repetitive, unpredictable, yet deterministic
- Used as a perturbation factor to "shake" particles out of the local convergence area
- Contribute to global exploration in a systematic yet "random-like" way

2. Speed Update with Chaos:

$$v_i^{t+1} = w.v_i^t + c_1.r_1.\left(p_{best_i} - x_i^t\right) + c_2.r_2.(g_{best} - x_i^t) + \alpha\,.z_k \quad (4)$$

Description:
α= chaos scaling factor (chaos influence weight)
z_k= value of chaotic map
Other parameters are the same as PSO.

Component Explanation:

a. Three Initial Components (Inertia, Cognitive, Social)

The first three components are identical to standard PSO:

- Inertia term ($w \cdot v_i^t$): Maintains the momentum of the previous particle
- Cognitive term ($c_1 \cdot r_1 \cdot (p_{best_i} - x_i^t)$): Pulling particles to their personal best position
- Social term ($c_2 \cdot r_2 \cdot (g_{best} - x_i^t)$): Attracting particles to global best position

These three components remain the basis of particle movement in CPSO.

b. Chaos Scaling Factor (α)

This parameter controls the effect of chaos on speed updates:

- Definition: The weight factor that regulates the intensity of chaos contribution in the update velocity
- General range: Usually within [0, 1] or [0, 2] depending on the researcher's preference
  - α small (0.01-0.1): Chaos has minimal influence, PSO almost standard
  - α moderate (0.1-0.5): Chaos makes a significant contribution
  - α large (0.5-1): Chaos dominates exploration
- Main function: To set the balance between exploration (chaos) and exploitation (PSO standard)
- Algorithm influences:
  - αtoo small: The benefits of chaos are not maximum, convergence like regular PSOs
  - αtoo big: Algorithms become too random, difficult to achieve global optimal
  - αAdaptive: Some studies use that decline with iteration α

c. Chaotic Value (z_k)

The chaos value generated from the logistic map in the iteration of k:

- Added directly to the velocity update equation as a perturbation term
- Gives a systematic random "boost" to particles
- Helps particles cross the "valley" in the objective function landscape
- It is more structured than the addition of pure random noise because it follows the dynamics of chaos

3. Position Update:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (5)$$

Explanation:
The position update equation in CPSO is identical to standard PSO:

- $x_i^{t+1}$is the position of the particles i on the iteration $t+1$
- $x_i^t$is the position of the particles i on the iteration t
- $v_i^{t+1}$is an updated speed (this time with a Chaos contribution)
- The new position is the result of summing the old position with the new velocity vector

2.5. Performance Evaluation

To assess the accuracy and reliability of the prediction model, this study uses four key evaluation metrics that comprehensively evaluate prediction error and model fit to the actual data. These four metrics complement each other: MSE and RMSE help identify large errors, MAE provides a stable picture of accuracy, and MAPE. This combination ensures a comprehensive evaluation of air quality prediction model performance. The experiments were carried out using a consistent configuration across all optimization methods. Both PSO and CPSO used 6 particles and were run for 8 iterations to ensure a fair comparison of their optimization capabilities. The dataset was split into 80% training and 20% validation to evaluate model performance during training and prevent overfitting. Four evaluation metrics, MSE, MAE, RMSE, and MAPE, were used to provide a comprehensive assessment of predictive accuracy, capturing both absolute and relative error characteristics.

Mean Squared Error (MSE)

MSE measures the average squared difference between the predicted and actual values[49]–[51]. This metric is very sensitive to large errors due to squared effects, so it is useful for identifying outliers. The smaller the MSE value, the better the model performance.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (6)$$

Description:
n= amount of data/samples
y_i= actual/observed value i
$\hat{y}_i$= i-th predicted value
$(y_i - \hat{y}_i)$= error/mistake number i

Mean Absolute Error (MAE)

MAE calculates the average absolute difference between the prediction and the true value[52], [53]. Unlike MSE, MAE gives equal weight to all errors, making it more robust to noise. A low MAE value indicates high prediction accuracy.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (7)$$

Description:
n= amount of data/samples
y_i= actual/observed value i
$\hat{y}_i$= i-th predicted value
$|y_i - \hat{y}_i|$= absolute value of the i-th error

Root Mean Squared Error (RMSE)

RMSE is the square root of MSE, which returns the error scale to the original units of data[54]–[56]. This metric is often used because it is easy to interpret and emphasizes large errors more than MAE. Like MSE, the lower the RMSE, the better the model is at predicting.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (8)$$

Description:
n= amount of data/samples
y_i= actual/observed value i
$\hat{y}_i$= i-th predicted value
$(y_i - \hat{y}_i)$= error/mistake number i

Mean Absolute Percentage Error (MAPE)

MAPE measures the average absolute percentage error between actual values and predicted values[57]–[59]. This metric is very useful because it provides an interpretation of error in percentage form, making it easy to understand and compare between datasets with different scales.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\% \quad (9)$$

Description:
n= amount of data/samples
y_i= actual/observed value i
$\hat{y}_i$= i-th predicted value

3. RESULT AND DISCUSSION

3.1. Hyperparameter Optimization Results

The hyperparameter optimization process yielded distinct optimal configurations for each model, demonstrating the effectiveness of the metaheuristic approach in refining the LSTM architecture.

Table 4. Hyperparameter search results

No	Hyperparameters	PSO	CPSO
1.	Hidden Layers	2	2
2.	Neurons	51	91
3.	Activation function	Sigmoid	Sigmoid
4.	Loss function	MSE	MSE
5.	Optimizer	RMSprop	RMSprop
6.	Batch size	32	64
7.	Epoch	16	5

The PSO algorithm produces a relatively complex architecture with 7 hidden layers and 51 neurons per layer, using a sigmoid activation function. This model uses an RMSprop optimizer with a batch size of 32 and is trained for 16 epochs. This configuration shows that PSO tends to select deeper networks with sufficient training epochs to capture temporal patterns in air pollution data.

In contrast, the CPSO approach produces a much more efficient architecture with only 2 hidden layers, yet a significantly higher number of neurons (91 per layer). The activation function remains sigmoid, with the RMSprop optimizer, a larger batch size (64), and only 5 training epochs. This striking difference indicates that CPSO can find a more straightforward yet highly effective model structure, offering a significant advantage in computational efficiency and training time. The dramatic reduction from 16 epochs to only 5 epochs shows that CPSO successfully finds a more optimal configuration with much faster convergence.

3.2. Convergence Analysis

Convergence analysis of optimization algorithms shows very significant differences in characteristics between conventional PSO and CPSO. In PSO, the optimization process requires 16

epochs to converge, suggesting a more extensive but potentially less efficient hyperparameter search. The resulting model has a high complexity with 7 hidden layers, suggesting that PSO tends to explore a more complex solution space.

In contrast, CPSO exhibits far superior convergence behavior, requiring only 5 training epochs (a 68.75% reduction from PSO), while producing a simpler architecture with 2 hidden layers. This remarkable efficiency is primarily due to the chaos factor, which effectively balances the exploration phase (searching for new areas) and the exploitation phase (optimizing already discovered areas) within the hyperparameter search space.

This faster, more efficient convergence is one of the key factors explaining why CPSO can produce models with better final performance than conventional PSO while also drastically reducing computation time. CPSO's ability to find optimal solutions with a minimal number of iterations demonstrates the superiority of this algorithm in navigating complex search spaces.

3.3. Model Performance Comparison

The performance evaluation results of the three models show significant differences:

Table 5. Model Performance Comparison

Model	MSE	IT IS	RMSE	MAP
Baseline LSTM	0.019124	0.111838	0.138289	36.742071
PSO-LSTM	0.018284	0.108809	0.135218	36.186673
Proposed CPSO-LSTM	0.012105	0.086356	0.110022	32.308927

A comparison of the three models shows very significant differences in prediction accuracy. The baseline LSTM model recorded an MSE of 0.019124, MAE of 0.111838, RMSE of 0.138289, and MAPE of 36.74%, indicating relatively low performance as a base model without optimization. The application of PSO optimization successfully improved the model's performance by reducing the MSE value to 0.018284 (a 4.4% improvement), MAE to 0.108809 (2.7% better), RMSE to 0.135218 (2.2% better), and MAPE to 36.19% (1.5% error reduction).

However, the CPSO-LSTM model showed a much greater advantage, achieving the best values across all evaluation metrics. Compared to the baseline LSTM, CPSO-LSTM successfully improved performance by 38.1% on MSE (0.012105), 22.8% on MAE (0.086356), 20.4% on RMSE (0.110022), and 11.9% on MAPE (32.31%). More impressively, compared to PSO-LSTM, CPSO-LSTM showed improvements of 33.8% in MSE, 20.6% in MAE, 18.6% in RMSE, and 10.7% in MAPE.

The striking difference between PSO-LSTM and CPSO-LSTM shows that integrating chaos factors into optimization algorithms not only results in faster convergence, as discussed earlier, but also dramatically improves model prediction accuracy. Although the MAPE of 32.31% still indicates room for improvement, the substantial improvement from the baseline (36.74%) demonstrates the effectiveness of the CPSO approach in optimizing LSTM model hyperparameters for air quality prediction.

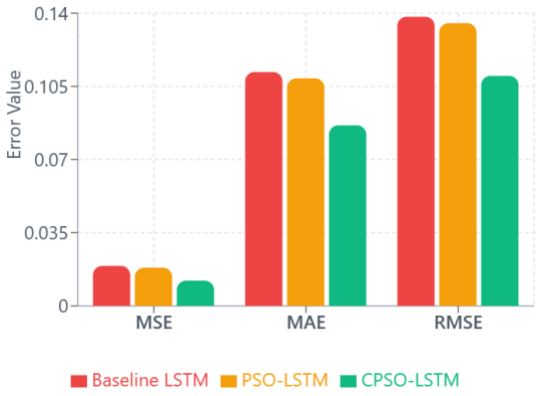

Figure 4. Error Metrics Comparison

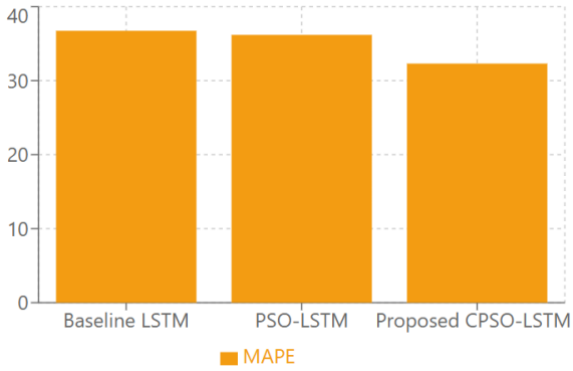

Figure 5. Mean Absolute Percentage Error (MAPE) Comparison

3.4. Discussion

Main Contributions of the Research

This study makes three significant contributions to the field of air quality prediction using deep learning. First, we propose the application of Chaotic Particle Swarm Optimization (CPSO) for LSTM hyperparameter optimization in the context of air pollution prediction, which, to our knowledge, has not been widely explored in the literature. Second, this study empirically proves that the integration of chaos factors in PSO can improve prediction performance by up to 38.1% (MSE) compared to the baseline model and 33.8% compared to conventional PSO, while also producing a much more efficient architecture with only 2 hidden layers versus 7 hidden layers in conventional PSO and a reduction in training time of 68.75% (5 epochs vs. 16 epochs). Third, our findings show that CPSO can achieve an optimal balance between model complexity and prediction accuracy, resulting in a model that is not only more accurate (11.9% improvement in MAPE) but also highly computationally efficient.

Optimization Effectiveness Analysis

An in-depth analysis of the optimization process reveals significant advantages of CPSO over conventional PSO in optimizing LSTM hyperparameters for air pollution prediction. Three key interrelated factors can explain the success of CPSO. Superior Search Space Exploration. The chaos factor embedded in CPSO provides a much more effective search-space exploration capability, allowing the algorithm to avoid local-optima traps that are often a weakness of conventional PSO. This is evident from CPSO's ability to find optimal configurations with far fewer epochs (5 vs. 16), demonstrating that the chaos factor facilitates a more focused, efficient search for solutions. The variability generated by the chaos factor facilitates solution diversification in the early stages of optimization, while maintaining intensification capabilities in the final stages.

CPSO exhibits significantly more efficient convergence, reaching the optimal solution in only 5 epochs compared to 16 for PSO (a 68.75% reduction). This not only dramatically speeds up the training process but also increases the reliability of the optimization results. This efficiency is crucial for practical applications where computation time and resources are important considerations. Most interestingly, CPSO successfully found a hyperparameter configuration that intelligently balances model complexity and prediction performance. The CPSO-LSTM architecture with 2 hidden layers and 91 neurons proved much more effective than the PSO-LSTM architecture with 7 hidden layers and 51 neurons. This unique combination of structural simplicity (a 71.4% reduction in the number of layers) and adequate processing capacity (a 78.4% increase in the number of neurons per layer) is the key to the superiority of CPSO-LSTM in generating more accurate predictions for air quality problems, while significantly maintaining the computational efficiency of the model.

Practical Implications

The findings of this study have very significant practical implications for the implementation of air quality prediction systems. The much simpler architecture (2 hidden layers) of CPSO-LSTM allows for highly efficient deployment on resource-constrained systems, such as edge computing or embedded systems for real-time air quality monitoring. The 68.75% reduction in training time (from 16 to 5

epochs) is particularly advantageous for scenarios where models need to be retrained periodically with the latest data, such as in adaptive monitoring systems. The 11.9% improvement in prediction accuracy (from 36.74% to 32.31%) on MAPE indicates significant progress towards the standards required for practical applications in public health and environmental management decision-making. Although an MAPE of 32.31% still indicates room for further improvement, the CPSO-LSTM model has demonstrated promising capabilities for implementation in air pollution early warning systems.

Limitations and Future Research Directions

Although the study's results show the significant superiority of CPSO-LSTM, several limitations should be considered. First, the MAPE of 32.31% still indicates significant prediction error, suggesting the need further to explore the model architecture or data preprocessing methods. Second, this study focused on a specific air quality dataset, so generalizing the results to other geographical contexts or pollutant parameters requires further validation. Third, the chaos factor used in CPSO was selected based on initial experiments, and further research is needed to determine the optimal value adaptively. Fourth, this study has not explored combining CPSO with other deep learning architectures, such as GRU, Bidirectional LSTM, Transformer, or hybrid models, which may further improve prediction accuracy.

These findings not only demonstrate the effectiveness of CPSO in this specific case but also open the door to its application to a range of time-series forecasting problems, including weather prediction, energy forecasting, and financial data analysis.

4. CONCLUSION

This study successfully demonstrates the effectiveness of metaheuristic techniques for optimizing LSTM model hyperparameters in air pollution prediction, addressing the challenges of optimal hyperparameter configuration complexity and computational efficiency. The main contributions of this research are: (1) empirically proving that CPSO significantly outperforms conventional PSO with a 38.1% increase in accuracy on MSE and 11.9% on MAPE, while reducing architecture complexity by 71.4% (from 7 to 2 hidden layers) and training time by 68.75% (from 16 to 5 epochs); (2) identifying that the integration of the chaos factor produces an optimal balance between exploration and exploitation, as evidenced by much faster and more stable convergence; and (3) demonstrated that a simple architecture with larger neuron capacity (2 layers with 91 neurons) can be far more effective than a complex architecture with smaller neuron capacity (7 layers with 51 neurons), offering superior computational efficiency without sacrificing accuracy.

The CPSO-LSTM model achieved the best performance across all evaluation metrics with an MSE of 0.012105, MAE of 0.086356, RMSE of 0.110022, and MAPE of 32.31%, outperforming the baseline LSTM by 38.1% (MSE) and 11.9% (MAPE), and PSO-LSTM by 33.8% (MSE) and 10.7% (MAPE). This superiority lies not only in its numerical accuracy but also in its remarkable convergence efficiency and the simplicity of its model architecture. These findings provide a significant methodological contribution to hyperparameter optimization in time series prediction applications, particularly for real-time air quality monitoring systems that require a balance between accuracy and computational efficiency.

REFERENCES

[1] P. Arifin, D. Muhafidin, and R. Pancasilawan, "City growth and its impact on residential problems: A case study in the city of Jakarta," *J. Community Serv. Empower.*, vol. 5, no. 2, pp. 272–281, 2024.

[2] S. A. Aulia, "Air pollution threatens health and climate change in Jakarta," *Hum. Error Saf.*, vol. 1, no. 1, pp. 36–46, 2024.

[3] B. P. Resosudarmo and L. Napitupulu, "Health and economic impact of air pollution in Jakarta," *Econ. Rec.*, vol. 80, pp. S65–S75, 2004.

[4] J. Muñoz and K. Castaño, "Enhancing Stock Price Predictions: Leveraging LSTM for Accurate Forecasting of Ecopetrol's Stock Performance," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 2, pp. 47–55, Jul. 2024, doi: 10.33292/ijarlit.v2i2.36.

[5] C. B. Pop, V. R. Chifu, I. S. A. Cozac, M. Antal, and C. Pop, "Optimizing the Data Center Energy Consumption Using a Particle

Swarm Optimization-Based Approach," 2015, pp. 176–189. doi: 10.1007/978-3-319-43177-2_12.

[6] D. Bohovic, "Comparison of LSTM and GRU Methods for Predicting Gold Exchange Rate against US Dollar," *Int. J. Artif. Intell. Informatics*, vol. 3, no. 1, pp. 24–29, Jan. 2025, doi: 10.33292/ijarlit.v3i1.43.

[7] D. Dzhumashev and M. Aybek, "Prediction of the Exchange Rate of the Russian Ruble (RUB) against the United States Dollar (USD): Performance Comparison of LSTM and CNN Models," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 1, pp. 25–32, Jan. 2024, doi: 10.33292/ijarlit.v2i1.33.

[8] C. J. C. Kusuma and K. Khairunnisa, "Optimizing Bidirectional LSTM for Energy Consumption Prediction Using Chaotic Particle Swarm Optimization and Hyperparameter Tuning," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 2, pp. 57–60, Jul. 2024, doi: 10.33292/ijarlit.v2i2.37.

[9] T. L. Aníbal and R. Okanlawon, "Stock Price Prediction of ReconAfrica (RECAF) Using Gated Recurrent Unit (GRU): Analysis and Implications for Investment Decisions," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 2, pp. 41–46, 2025, doi: 10.33292/ijarlit.v2i2.35.

[10] F. Agustin and P. De Melin, "Comparison of GRU and CNN Methods for Predicting the Exchange Rate of Argentine Peso (ARS) against US Dollar (USD)," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 1, pp. 9–16, 2024, doi: 10.33292/ijarlit.v2i1.31.

[11] R. Kepo and D. Okokpujie, "Performance Comparison of GRU and LSTM Methods for Predicting Bitcoin Exchange Rate against US Dollar," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 1, pp. 17–24, 2024, doi: 10.33292/ijarlit.v2i1.32.

[12] U. Pujianto, D. P. P. Setyadi, and M. I. Akbar, "Prediction of stock purchase decisions using artificial neural network method," *Appl. Eng. Technol.*, vol. 1, no. 2, pp. 23–33, Apr. 2022, doi: 10.31763/aet.v2i1.686.

[13] M. A. A. Abdalla *et al.*, "Machine learning-based residential load demand forecasting: Evaluating ELM, XGBoost, RF, and SVM for enhanced energy system and sustainability," *Sci. Inf. Technol. Lett.*, vol. 6, no. 1, pp. 1–15, May 2025, doi: 10.31763/sitech.v6i1.1866.

[14] V. O. Geteloma *et al.*, "Enhanced data augmentation for predicting consumer churn rate with monetization and retention strategies: a pilot study," *Appl. Eng. Technol.*, vol. 3, no. 1, pp. 35–51, Apr. 2024, doi: 10.31763/aet.v3i1.1408.

[15] G. Tzoulis, "Harnessing Convolutional Neural Networks for Accurate Stock Price Prediction: A Case Study of Hellenic Telecommunications Organization (HTO.AT)," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 2, pp. 66–72, 2024, doi: 10.33292/ijarlit.v2i2.39.

[16] K. Rai and A. Vijayan, "Performance Comparison of Long Short-Term Memory and Convolutional Neural Network for Prediction of Exchange Rate of Indian Rupee against US Dollar," *Int. J. Artif. Intell. Informatics*, vol. 3, no. 1, pp. 9–15, Jan. 2025, doi: 10.33292/ijarlit.v3i1.41.

[17] F. Al Anshori and S. Pidgeon, "Prediction of Euro to US Dollar Exchange Rate Using CNN Method with Grid Optimization," *Int. J. Artif. Intell. Informatics*, vol. 3, no. 2, pp. 27–44, Jul. 2025, doi: 10.33292/ijarlit.v3i2.45.

[18] D. P. Ismi and N. Khoirunnisa, "Enhancing the performance of heart arrhythmia prediction model using Convolutional Neural Network based architectures," *Sci. Inf. Technol. Lett.*, vol. 5, no. 2, pp. 1–12, Nov. 2024, doi: 10.31763/sitech.v5i2.1794.

[19] A. Pranolo, Y. Mao, A. P. Wibawa, A. B. P. Utama, and F. A. Dwiyanto, "Robust LSTM With Tuned-PSO and Bifold-Attention Mechanism for Analyzing Multivariate Time-Series," *IEEE Access*, vol. 10, no. July, pp. 78423–78434, 2022, doi: 10.1109/ACCESS.2022.3193643.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] S. Dhakal, Y. Gautam, and A. Bhattarai, "Exploring a deep LSTM neural network to forecast daily PM2.5 concentration using meteorological parameters in Kathmandu Valley, Nepal," *Air Qual. Atmos. Heal.*, vol. 14, no. 1, pp. 83–96, 2021, doi: 10.1007/s11869-020-00915-6.

[22] E. Mussumeci and F. Codeço Coelho, "Large-scale multivariate forecasting models for Dengue - LSTM versus random forest regression," *Spat. Spatiotemporal. Epidemiol.*, vol. 35, p. 100372, 2020, doi: 10.1016/j.sste.2020.100372.

[23] J. Zhang, Y. Zhu, X. Zhang, M. Ye, and J. Yang, "Developing a Long Short-Term Memory (LSTM) based model for predicting water table depth in agricultural areas," *J. Hydrol.*, vol. 561, pp. 918–929, 2018, doi: 10.1016/j.jhydrol.2018.04.065.

[24] S. Srivastava and S. Lessmann, "A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data," *Sol. Energy*, vol. 162, no. December 2017, pp. 232–247, 2018, doi: 10.1016/j.solener.2018.01.005.

[25] A. G. Salman, Y. Heryadi, E. Abdurahman, and W. Suparta, "Single Layer & Multi-layer Long Short-Term Memory (LSTM) Model with Intermediate Variables for Weather Forecasting," *Procedia Comput. Sci.*, vol. 135, pp. 89–98, 2018, doi: 10.1016/j.procs.2018.08.153.

[26] V. K. R. Chimmula and L. Zhang, "Time series forecasting of COVID-19 transmission in Canada using LSTM networks," *Chaos, Solitons and Fractals*, vol. 135, 2020, doi: 10.1016/j.chaos.2020.109864.

[27] J. Tulensalo, J. Seppänen, and A. Ilin, "An LSTM model for power grid loss prediction," *Electr. Power Syst. Res.*, vol. 189, no. March, p. 106823, 2020, doi: 10.1016/j.epsr.2020.106823.

[28] Y. S. Chang, H. T. Chiao, S. Abimannan, Y. P. Huang, Y. T. Tsai, and K. M. Lin, "An LSTM-based aggregated model for air pollution forecasting," *Atmos. Pollut. Res.*, vol. 11, no. 8, pp. 1451–1463, 2020, doi: 10.1016/j.apr.2020.05.015.

[29] B. Mokona and N. Shipo, "Comparative Analysis of LSTM and Grid Search Optimized LSTM for Stock Prediction: Case Study of Africa Energy Corp. (AFE.V)," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 1, pp. 1–8, 2025, doi: https://doi.org/10.33292/ijarlit.v2i1.30.

[30] S. Sanhatham, "Stock Price Prediction of Thai Oil Public Company Limited (TOP.BK) Using LSTM Model with Grid Search Hyperparameter optimization," *Int. J. Artif. Intell. Informatics*, vol. 2, no. 1, pp. 33–40, 2024, doi: 10.33292/ijarlit.v2i1.34.

[31] R. Das, A. I. Middya, and S. Roy, *High granular and short term time series forecasting of PM 2.5 air pollutant - a comparative*

review, vol. 55, no. 2. Springer Netherlands, 2022. doi: 10.1007/s10462-021-09991-1.

[32] I. V Pustokhina, D. A. Pustokhin, E. L. Lydia, P. Garg, A. Kadian, and K. Shankar, "Hyperparameter search based convolution neural network with Bi-LSTM model for intrusion detection system in multimedia big data environment," *Multimed. Tools Appl.*, vol. 81, no. 24, pp. 34951–34968, 2022.

[33] Y. Li, Z. Tong, S. Tong, and D. Westerdahl, "A data-driven interval forecasting model for building energy prediction using attention-based LSTM and fuzzy information granulation," *Sustain. Cities Soc.*, vol. 76, no. August 2021, p. 103481, 2022, doi: 10.1016/j.scs.2021.103481.

[34] E. Yilmaz and E. Tcrol, "Comparison of LSTM and LSTM with Grid Search Optimization for Stock Price Prediction of Saudi Arabian Oil Company (Aramco)," *Int. J. Artif. Intell. Informatics*, vol. 3, no. 2, pp. 52–59, Jul. 2025, doi: 10.33292/ijarlit.v3i2.47.

[35] A. Flavia and C. Mio, "Performance Comparison of Standard LSTM and LSTM with Random Search Optimization for Spark New Zealand Limited Stock Price Prediction," *Int. J. Artif. Intell. Informatics*, vol. 3, no. 2, pp. 60–66, Jul. 2025, doi: 10.33292/ijarlit.v3i2.48.

[36] H. Abbasimehr, M. Shabani, and M. Yousefi, "An optimized model using LSTM network for demand forecasting," *Comput. Ind. Eng.*, vol. 143, no. March, p. 106435, 2020, doi: 10.1016/j.cie.2020.106435.

[37] W. Elmasry, A. Akbulut, and A. H. Zaim, "Evolving deep learning architectures for network intrusion detection using a double PSO metaheuristic," *Comput. Networks*, vol. 168, p. 107042, 2020, doi: 10.1016/j.comnet.2019.107042.

[38] M. Munem, T. M. Rubaith Bashar, M. H. Roni, M. Shahriar, T. B. Shawkat, and H. Rahaman, "Electric power load forecasting based on multivariate LSTM neural network using bayesian optimization," *2020 IEEE Electr. Power Energy Conf. EPEC 2020*, vol. 3, 2020, doi: 10.1109/EPEC48502.2020.9320123.

[39] N. Buslim, I. L. Rahmatullah, B. A. Setyawan, and A. Alamsyah, "Comparing bitcoin's prediction model using GRU, RNN, and LSTM by hyperparameter optimization grid search and random search," in *2021 9th International Conference on Cyber and IT Service Management (CITSM)*, 2021, pp. 1–6.

[40] S. Zhou, L. Zhou, M. Mao, H.-M. Tai, and Y. Wan, "An optimized heterogeneous structure LSTM network for electricity price forecasting," *Ieee Access*, vol. 7, pp. 108161–108173, 2019.

[41] N. Gorgolis, I. Hatzilygeroudis, Z. Istenes, and L. N. G. Gyenne, "Hyperparameter Optimization of LSTM Network Models through Genetic Algorithm," *10th Int. Conf. Information, Intell. Syst. Appl. IISA 2019*, vol. 00, no. c, pp. 1–4, 2019, doi: 10.1109/IISA.2019.8900675.

[42] A. Indrawati and I. N. Wahyuni, "Enhancing machine learning models through hyperparameter optimization with particle swarm optimization," in *2023 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, 2023, pp. 244–249.

[43] D. Freitas, L. G. Lopes, and F. Morgado-Dias, "Particle swarm optimisation: a historical review up to the current developments," *Entropy*, vol. 22, no. 3, p. 362, 2020.

[44] S. Chourasia, H. Sharma, M. Singh, and J. C. Bansal, "Global and local neighborhood based particle swarm optimization," in *Harmony Search and Nature Inspired Optimization Algorithms: Theory and Applications, ICHSA 2018*, 2019, pp. 449–460.

[45] D. Tian, X. Zhao, and Z. Shi, "Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization," *Swarm Evol. Comput.*, vol. 51, p. 100573, 2019.

[46] A. B. Putra Utama, A. P. Wibawa, M. Muladi, L. Hernandez, H. Haviluddin, and M. F. Teng, "E-Journal visitor analysis using particle swarm optimization-long short-term memory (PSO LSTM)," *Sci. Inf. Technol. Lett.*, vol. 6, no. 1, pp. 55–65, May 2025, doi: 10.31763/sitech.v5i2.2347.

[47] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, 1995, vol. 4, pp. 1942–1948.

[48] J. Wang, "CPSO: chaotic particle swarm optimization for cluster analysis," *J. Artif. Intell. Technol.*, vol. 3, no. 2, pp. 46–52, 2023.

[49] J.-D. Wang and C. Oktomy Noto Susanto, "Traffic Flow Prediction with Heterogenous Data Using a Hybrid CNN-LSTM Model," *Comput. Mater. Contin.*, vol. 76, no. 3, pp. 3097–3112, 2023, doi: 10.32604/cmc.2023.040914.

[50] S. Riyadi, C. Damarjati, S. Khotimah, and A. J. Ishak, "Optimization of the VGG Deep Learning Model Performance for Covid-19 Detection Using CT-Scan Images," *Regist. J. Ilm. Teknol. Sist. Inf.*, vol. 10, no. 1, pp. 91–101, Jun. 2024, doi: 10.26594/register.v10i1.3598.

[51] S. Riyadi, A. Divayu Andriyani, and S. Noraini Sulaiman, "Improving Hate Speech Detection Using Double-Layers Hybrid CNN-RNN Model on Imbalanced Dataset," *IEEE Access*, vol. 12, pp. 159660–159668, 2024, doi: 10.1109/ACCESS.2024.3487433.

[52] S. Riyadi, M. A. P. Suradi, C. Damarjati, H.-C. Chen, R. Al-Hamdi, and A. M. Masyhur, "Classification of Political Party Conflicts and Their Mediation Using Modified Recurrent Convolutional Neural Network," *J. Appl. Data Sci.*, vol. 6, no. 1, pp. 416–425, 2025.

[53] S. Riyadi, N. Audita, and F. A. Abidin, "Movie Recommendation Using Collaborative Filtering Method and K-Nearest Neighbours: A Case Study on Netflix," in *Lecture Notes in Networks and Systems*, 2024, vol. 1120 LNNS, pp. 428–436. doi: 10.1007/978-3-031-70518-2_38.

[54] N. A. Utama, W. I. Triyani, S. Riyadi, and C. Damarjati, "Discrete Curvelet Transform Feature Extraction for Mangosteen Fruit Surface Damage Detection," *Emerg. Inf. Sci. Technol.*, vol. 5, no. 1, pp. 46–51, 2024, doi: 10.18196/eist.v5i1.22602.

[55] A. Kurnianti, P. Pahlevi, and I. Mufidah, "Recommendation System for Prospective Bride and Groom Using Cosine Similarity Algorithm," *Emerg. Inf. Sci. Technol.*, vol. 4, no. 1, pp. 8–15, 2023, doi: 10.18196/eist.v4i1.18683.

[56] I. Prabasari, A. Zuhri, S. Riyadi, T. K. Hariadi, and N. A. Utama, "Analysis of Cross Validation on Classification of Mangosteen Maturity Stages using Support Vector Machine," *Emerg. Inf. Sci. Technol.*, vol. 5, no. 1, pp. 24–29, 2024, doi: 10.18196/eist.v5i1.22359.

[57] L. I. Berlina, U. S. Aesyi, and K. Kharisma, "Community perspective analysis of Yogyakarta special region using K-means algorithm," *Emerg. Inf. Sci. Technol.*, vol. 5, no. 2, pp. 66–73, 2024, doi: 10.18196/eist.v5i2.24729.

[58] A. Rakhmadi and N. D. Rahmawati, "Implementation of Simple Additive Weighting to Decide a Fund Proposal," *Emerg. Inf. Sci. Technol.*, vol. 4, no. 2, pp. 67–74, 2023, doi: 10.18196/eist.v4i2.20741.

[59] R. Y. Hartanta, A. Asroni, S. Riyadi, and J. Jeckson, "Analysis and Visualization of High School Student Achievement Data Using Decision Tree and Cross-Validation in Rapidminer," *Emerg. Inf. Sci. Technol.*, vol. 4, no. 2, pp. 62–66, 2023, doi: 10.18196/eist.v4i2.20731.