Environmental and Climate Technologies
2026, vol. 30, no. 1, pp. 526–541
https://doi.org/10.2478/rtuect-2026-0036
https://content.sciendo.com

Deep Learning for PM2.5 Forecasting: An Empirical Comparison of LSTM, GRU, and N-BEATS

Murad A. YAGHI*[1], Huthaifa Al-OMARI[2], Salem AlEMAISHAT[3], Raghad Abu WADI[4]

[1]*Department of Data Science and Artificial Intelligence, Al Hussein Technical University, Amman, Jordan*
[2,3]*Department of Computer Science, Al Hussein Technical University, Amman, Jordan*
[4]*Department of Business Intelligence, The Hashemite University, Zarqa, Jordan*

Received 16.03.2026; accepted 20.04.2026

***Abstract* – Accurate predictions of fine particulate matter (PM2.5) levels will significantly contribute toward enhancing public health and provide valuable information to policymakers regarding how to create and implement policies that protect the environment. The application of deep-learning techniques has produced promising results in the area of PM2.5 forecasting; however, few comparative studies using the same dataset and evaluation criteria have been conducted. This study evaluates the performance of three of the most commonly used sequence-oriented architectures (i.e., Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Neural Basis Expansion Analysis for Time Series (N-BEATS)) to predict short-term PM2.5 levels. Each model was trained and tested using the same 70/15/15 data-split ratio and the same four input window sizes (i.e., 5, 10, 15, and 20 hours). Generally, the use of longer input windows improves accuracy; however, the rate of improvement is much reduced beyond 15–20 time steps. N-BEATS produced higher accuracy than both recurrent models over each of the evaluation metrics and input window sizes; specifically, at the 20-hour time step, N-BEATS achieved a root mean squared error (RMSE) of 55.28 µg/m³ and an R^2 value of 0.5116, which represents a reduction in RMSE of approximately 14 % compared to the GRU while providing improved tracking of hazardous pollution events. N-BEATS also trained at a significantly faster pace and experienced very little increase in computational costs as input window lengths increased; whereas the training times of both LSTM and GRU were essentially tripled. N-BEATS' feed-forward architecture and block-based design lend themselves to be executed in parallel, and the inherent basis-function decomposition within N-BEATS is more conducive to modelling the sharp, non-linear fluctuations typically present in pollution datasets. Overall, the findings presented in this study demonstrate that N-BEATS is a viable and effective tool for near-real-time air-quality forecasting applications and emphasize the need for standardization of benchmarking in the field of atmospheric time-series analysis.**

***Keywords* – LSTM; GRU; N-BEATS; PM2.5; particulate matter; weather forecasting**.

* Corresponding author.
E-mail address: murad.yaghi@htu.edu.jo

Murad A. YAGHI https://orcid.org/0000-0002-9859-4075
Huthaifa Al-OMARI https://orcid.org/0009-0005-5611-8395
Salem AlEMAISHAT https://orcid.org/0009-0001-3383-1949
Raghad Abu WADI https://orcid.org/0009-0008-6690-6215

©2026 Author(s). This is an open access article licensed under the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0).

1. INTRODUCTION

Air Pollution is one of the biggest dangers facing both human health and the Environment. Haze Episodes occur so frequently they can hinder transportation, lower visibility, and overwhelm healthcare Systems [1]. Fine Particulates (i.e., particulate matter ≤ 2 μm in diameter) pose significant health risks due to their ability to deeply penetrate lung tissue and cause a wide variety of adverse health effects after extended periods of exposure. Additionally, fine particulates can travel long distances and settle on water and soil surfaces and participate in atmospheric chemical reactions contributing to acid rain and ecological deterioration [2]. Fossil fuel combustion and industrial activities are the main contributors to primary PM2.5 emissions. However, secondary PM2.5 formation occurs through chemical transformations of gaseous precursors such as SO_2 and No_x. Epidemiologic studies have identified numerous health effects of PM2.5 exposures. Short-term exposure to PM2.5 has been linked to an increase in hospital admissions for respiratory and cardiovascular diseases. Chronic exposure to PM2.5 has been shown to result in decreased lung function, increased risk of stroke, and elevated mortality rates. The World Health Organization estimates that globally, ambient air pollution results in approximately 4.2 million premature deaths annually and that PM2.5 is the largest single environmental risk factor for those deaths. These data have prompted many government and environmental agencies to commit large amounts of resources to develop, implement, and maintain monitoring and early warning systems to notify the public when pollutant concentrations reach hazardous levels.

The development of accurate forecasting systems for predicting PM2.5 concentrations is far from being simple. A large number of factors (emission sources, atmospheric chemistry, weather conditions, and geographical characteristics) interact to produce the particulate matter (PM) levels. In addition, traditional statistical techniques (such as ARIMA and regression models) have limitations in the ability to detect abrupt nonlinear changes in time series (which are characteristic of actual pollution data). This has led the scientific community to consider machine learning and, even more recently, deep learning algorithms that can identify arbitrary nonlinear relationships between inputs and outputs from the data itself.

In order to predict PM2.5 values over short- and medium-term periods, authorities can provide citizens with timely warnings about potential pollution episodes and citizens can also reduce their own exposures during those episodes. Therefore, deep learning has been gaining momentum as an approach to forecasting PM2.5 values, since it is capable of extracting both the nonlinear and high-dimensional temporal structures present in air quality data. Research in this area includes a variety of topics: the use of hybrid CNN-LSTM architectures that first extract spatial features and then process them sequentially [3]–[6]; signal decomposition and/or feature engineering strategies combined with recurrent neural networks [7]–[9]; encoder-decoder and optimization based architectures [10], [11]; interpolation based methods for regions where there are few monitoring stations [12]–[14]; and those that utilize reinforcement learning to train the models [15], [16]. Although these works report higher accuracy than the baselines established using classical statistical techniques, many open questions still exist concerning how well the models can generalize when the amount of sensor data is reduced; the need to compare models trained under similar conditions (and possibly on the same data); and the lack of understanding regarding which temporal patterns the models are identifying [5], [13].

Based on the background above, we propose an experiment comparing three of the most popular sequence predictors – LSTM, GRU, and N-BEATS – using PM2.5 as the target time series for forecasting. Our study (i) holds all preprocessing and training settings constant so that performance differences can be attributed mainly to model architecture, (ii) reports MAE, RMSE, and R^2 across four input look-back windows under a fixed one-hour-ahead forecast

horizon, and (iii) compares both predictive accuracy and training efficiency under the same multivariate input setup. The rest of the paper is structured as follows: Section II provides an overview of the three model architectures; Section III summarizes the experimental findings and examines their implications; Section IV concludes with suggestions for further research. As compared to many other studies which have compared only two different models or varied the preprocessing pipeline among different architectures, our approach allows us to ensure that all differences in the model's performance are due to the model itself and not to other aspects of the model's design. Additionally, we move beyond simply reporting average accuracy numbers and we also consider training efficiency, convergence behaviour, and the model's ability to accurately predict extreme pollution events, which is a critical issue for public health applications but has received little attention in the literature on PM2.5 forecasting.

2. METHODOLOGY

Due to their complexity in terms of being nonlinear and varying over time, researchers have difficulty using PM2.5 time series data. To resolve these issues, the researchers adopted the use of neural network (NN) models for handling time series data [15]. We used three models based on NN to analyse our time series data in this research: Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), and Neural Basis Expansion Analysis for Time Series (N-BEATS).

2.1. Long Short-Term Memory (LSTM) Model

Hochreiter and Schmidhuber introduced Long Short-Term Memory (LSTM) networks to address the vanishing-gradient problem that hinders traditional recurrent neural networks when learning long-range dependencies [17]. The ability of the LSTM to capture short- and long-term dependencies in time-series data is provided by the gated memory cell, which determines what stored information should be retained, updated, or erased [18]. Internally, each LSTM unit relies on three gating mechanisms. The input gate regulates how much of the new observation is written into the cell state. The forget gate decides which parts of the existing memory to keep and which to discard. The output gate then controls what portion of the cell state is exposed as the unit's output [3].

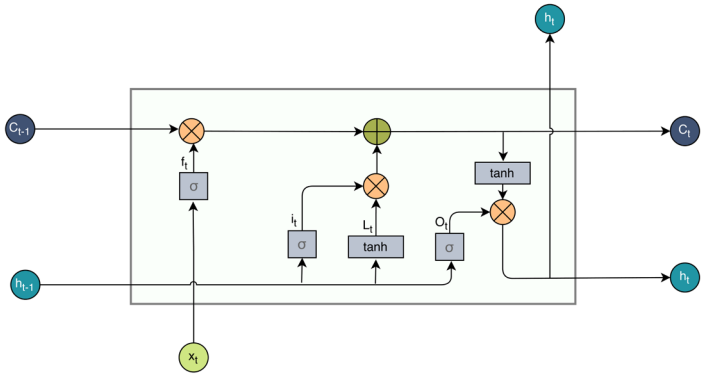


Fig. 1. Architecture of an LSTM cell.

The LSTM cell shown in Fig. 1 computes its gate activations, cell state, and hidden state at time step t according to the following equations:

$$\begin{aligned} f_t &= \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \\ i_t &= \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right) \\ C_t &= f_t \times C_{t-1} + i_t \times L_t \\ L_t &= \tanh\left(W_c\left[h_{t-1}, x_t\right] + b_c\right) \\ o_t &= \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \\ h_t &= o_t \times \tanh\left(C_t\right) \end{aligned} \tag{1}$$

In the equations above, the three gates are denoted by i_t (input), f_t (forget), and o_t (output). The symbol x_t stands for the current input, h_{t-1} is the hidden state produced at the previous step, and C_{t-1} is the prior cell state. All three gates receive both x_t and h_{t-1} as their inputs.

When f_t equals 0 the gate discards the stored information entirely; when f_t equals 1 it passes everything through. The function σ is the sigmoid activation, W_f is the forget-gate weight matrix, and b_f is the associated bias term.

The input gate controls the extent to which new information L_t enters the memory cell at each step. W_i, W_c, and W_o are the weight matrices associated with the input gate, cell state, and output gate, while b_i, b_c, and b_o are their respective bias vectors.

The output gate takes the current input x_t together with the previous hidden state h_{t-1} and computes the activation o_t. The hidden output h_t is then obtained by element-wise multiplication of o_t with the cell state after it has been passed through the tanh non-linearity [19]–[21].

2.2. Gated Recurrent Unit (GRU)

The GRU, proposed by Cho and colleagues in 2014, simplifies the LSTM design by collapsing the separate forget and input gates into a single update gate [22]. This leaves the GRU with just two gating signals, a reset gate and an update gate, which reduces the parameter count while still allowing the network to model long-range dependencies [23].

The key idea behind GRU is that a single update gate can simultaneously handle what LSTM achieves with separate forget and input gates. The Update Gate z_t determines the amount of the prior hidden state to keep in the prior hidden state (z_t values close to 1 indicate 'keep almost the same' whereas values close to 0 indicate 'replace the prior hidden state with the candidate'). The Reset Gate r_t determines the amount of prior state to feed into computing the candidate activation; when rt is very close to zero, the unit will act as if the sequence had just started, effectively letting the network "forget" its past which is irrelevant.

Since GRUs merge the two gates (z_t & r_t) into one, they typically have about 2/3 of the number of parameters as an equivalent sized LSTM cell, thus making them easier to train and less likely to over fit to small or moderately sized data sets. Studies on many different sequential tasks show GRUs match or exceed LSTMs performance, however, due to the limitations in size of PM2.5 training sets, where the largest limitation is the size of training set from available continuous record monitors, the efficiency in the number of parameters can be a useful benefit for the GRU.

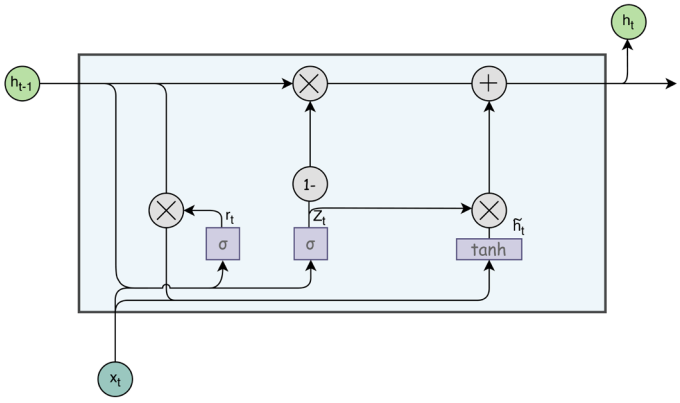


Fig. 2. Architecture of a GRU cell.

The GRU cell shown in Fig. 2 computes its update and reset gates, candidate activation, and new hidden state at time step t according to the following equations.

$$\begin{aligned} h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \\ z_t &= \sigma\left(W_z x_t + U_z h_{t-1} + b_z\right) \\ \tilde{h}_t &= \tanh\left(W_h x_t + U_h\left(r_t \odot h_{t-1}\right) + b_h\right) \\ r_t &= \sigma\left(W_r x_t + U_r h_{t-1} + b_r\right) \end{aligned} \tag{2}$$

In these expressions, $\odot$ denotes element-wise (Hadamard) multiplication, σ is the sigmoid activation, and tanh is the hyperbolic tangent. Both the reset gate r_t and the update gate z_t are driven by two quantities: h_{t-1}, the hidden state from the preceding step, and x_t, the current input vector. The learnable parameters W_z, W_r, and W_h are the corresponding weight matrices [24], [25].

2.3. Neural Basis Expansion Analysis for Time Series (N-BEATS)

N-BEATS is a purely feed-forward deep neural network designed from the ground up for time-series forecasting. It was introduced by Oreshkin and co-authors in 2019 and differs from recurrent models in that it processes fixed-length input windows through a stack of fully connected blocks rather than maintaining hidden states across time steps [28].

The N-BEATS network consists of two sub-networks: the Backcast network, which projects historical data, and the Forecast network, which generates future predictions [27]. Each basic block is composed of a shared fully connected (FC) stack followed by two linear layers that use expansion coefficients θ_t^f and θ_t^b to produce the forecast $O_{t,r}^{k,f}$ and backcast $O_{t,r}^{k,b}$ outputs, respectively.

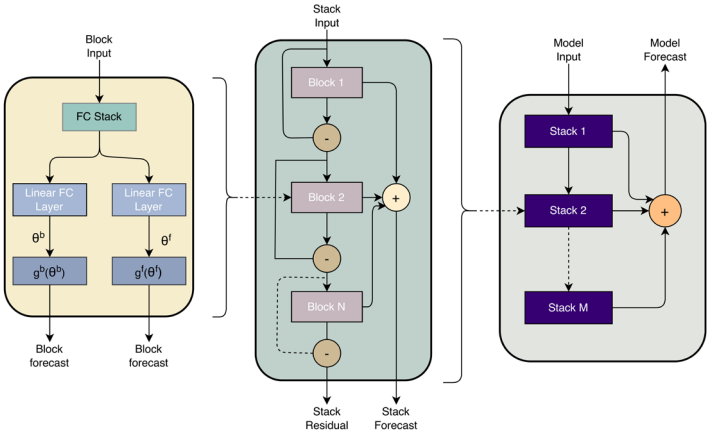


Fig. 3. Basic block architecture of the N-BEATS network.

The operations in the basic building block are presented as (3):

$$\begin{aligned} h^k_{(t,r,1)} &= FC_{(t,1)(X_t)}, h^k_{(t,r,2)} = FC_{(t,2)(h^k_{(t,r,1)})}, h^k_{(t,r,3)} = FC_{(t,3)(h^k_{(t,r,2)})} \\ h^k_{(t,r,4)} &= FC_{(t,4)(h^k_{(t,r,3)})}, h^k_{(t,r,5)} = FC_{(t,5)(h^k_{(t,r,4)})}, h^k_{(t,r,6)} = FC_{(t,6)(h^k_{(t,r,5)})} \\ h^k_{(t,r,7)} &= FC_{(t,7)(h^k_{(t,r,6)})}, h^k_{(t,r,8)} = FC_{(t,8)(h^k_{(t,r,7)})}, O^{(k,f)}_{(t,r)} = \text{Linear}^f_t(h^k_{(t,r,8)}), \\ O^{(k,b)}_{(t,r)} &= \text{Linear}^b_t(h^k_{(t,r,8)}) \end{aligned} \tag{3}$$

For example, the first FC layer operation is

$$h^k_{t,r,1} = \text{ReLU}\left(W_{t,1}X_t + b_{t,1}\right), \tag{4}$$

where ReLU(·) is the rectified linear unit activation, $W_{t,1}$ is the weight matrix, and $b_{t,1}$ is the bias. The linear layer for the forecast output is

$$O^{k,f}_{t,r} = W^{k,f}_{t,r} h^k_{t,r,8}, \tag{5}$$

where $W^{k,f}_{t,r}$ is the forecast weight matrix.

To generate the block's backcast and forecast contributions, the expansion coefficients are combined with basis vectors $\upsilon^{k,b}_{r,i}$ and $\upsilon^{k,f}_{r,i}$:

$$\hat{y}^k_{t,r} = \sum_{i=1}^{\dim\left(\theta^{k,f}_{t,r}\right)} \theta^{k,f}_{t,r,i}\,\upsilon^{k,f}_{r,i}, \hat{x}^k_{t,r} = \sum_{i=1}^{\dim\left(\theta^{k,b}_{t,r}\right)} \theta^{k,b}_{t,r,i}\,\upsilon^{k,b}_{r,i}. \tag{6}$$

The residual input to the next block is calculated as Eq. (7) and the aggregated forecast at block r as Eq. (8), see next page:

$$x^k_{t,r} = x^k_{t-1,r} - \hat{x}^k_{t,q,r}, \tag{7}$$

$$\hat{Y}_{t,r} = \sum_{k=1}^{K} \hat{y}_{t,r}^{k}. \tag{8}$$

Finally, the overall network prediction is

$$\hat{Y}_{t+1} = \sum_{r=1}^{R} \hat{Y}_{t,r}, \tag{9}$$

representing the predicted value [29].

3. EXPERIMENTAL RESULTS

This section benchmarks three deep-learning architectures, N-BEATS, LSTM, and GRU, for one-hour-ahead PM2.5 concentration forecasting. All models were trained and evaluated using the same chronological 70/15/15 % train-validation-test split and the same four labelled input look-back windows of 5, 10, 15, and 20 hours.

3.1. Dataset Description

The dataset used in this study is the Beijing PM2.5 dataset from the UCI Machine Learning Repository [31]. It contains 43 824 hourly observations from 1 January 2010 to 31 December 2014. The PM2.5 target measurements were collected at the United States Embassy monitoring site in Beijing, China, while the meteorological variables were obtained from Beijing Capital International Airport. The raw fields include date and hour, PM2.5 concentration (μg/m³), dew point, temperature, pressure, combined wind direction, cumulated wind speed, cumulated hours of snow, and cumulated hours of rain.

Missing PM2.5 values are marked as "NA" in the source dataset. In preprocessing, these entries were converted to missing values and rows without a valid PM2.5 target were removed; no PM2.5 imputation was applied. This removed 2067 rows and left 41 757 rows with valid PM2.5 targets. Wind direction was one-hot encoded, date and row-number fields were removed from the model input, and the numeric predictors and PM2.5 target were min-max scaled to the range 0–1. Sliding-window samples containing missing values were skipped.

The forecasting setup is multivariate: each input window includes historical PM2.5 together with dew point, temperature, pressure, cumulated wind speed, cumulated snow hours, cumulated rain hours, and one-hot wind-direction indicators. For each labelled look-back window (ws = 5, 10, 15, and 20 hours), the target is the PM2.5 concentration at the next hourly step. Thus, the look-back window controls how much past information is provided to the model, whereas the forecast horizon is fixed at one hour ahead for all experiments.

3.2. Model Configuration and Training Settings

All models were implemented in PyTorch and trained for 200 epochs using mean squared error (MSE) loss, the Adagrad optimizer with a learning rate of 0.001, and a batch size of 1024. Training batches were not shuffled in order to preserve the chronological ordering of the time-series samples. The best model checkpoint for each configuration was selected according to the lowest validation loss. The LSTM and GRU models used five recurrent layers with 64 hidden units and a final fully connected output layer. The N-BEATS model used five blocks, each with five fully connected layers, a hidden size of 128, ReLU activations, and a single-output forecast head.

3.3. Training Time Comparison Across Sequence Lengths

Fig. 4 compares the training time of the LSTM, GRU, and N-BEATS models for each of the four input look-back windows (*ws* = 5, 10, 15, and 20 hours), with all models trained using the settings described in Section 3.2.

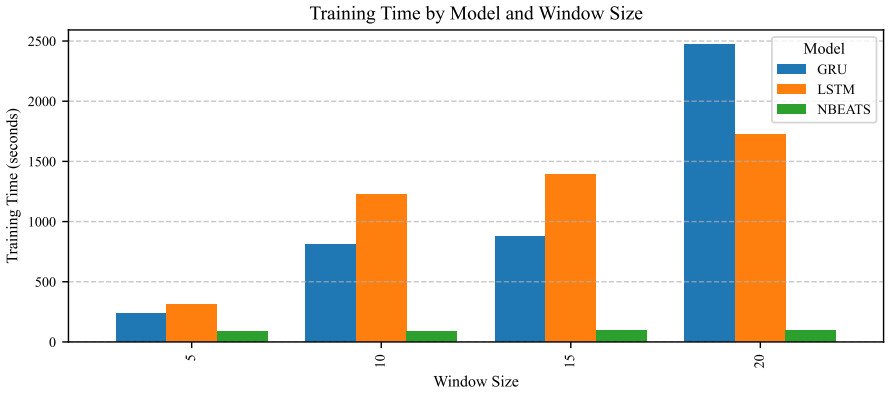


Fig. 4. Training time in seconds for LSTM, GRU, and N-BEATS across the four input look-back windows used for one-hour-ahead PM2.5 forecasting.

The data clearly show how much faster the three architecture types are when processing data at the smallest window (window size, *ws* = 5). GRU was about 200 seconds, LSTMs were about 350 seconds, but N-BEATS completed in less than 50 seconds. The difference widened even further at a window size of 20 where GRUs required roughly 2500 seconds and LSTMs took roughly 1800 seconds, an increase of roughly 10 × and 5 × from their respective base line of 5 sec, while N-BEATS increased only slightly to about 70 seconds.

The reason for these results is simple. The two models LSTMs and GRU both take the input sequence step-by-step; the hidden states depend upon the hidden state of the previous step, therefore, doubling the window length roughly doubles the number of steps in the sequential processing chain, therefore, doubling the wall clock time. The model N-BEATS, however, is fully feedforward. It treats all the data in the look-back window as a single fixed length vector and feeds it into multiple fully connected parallel layers. Therefore, because there is no recursion, the forward and backward pass can both be done in a very parallel manner on current GPU hardware, and increasing the amount of input steps only adds to the width of the first layer, and does not add to the depth of the computation graph.

3.4. Convergence Behaviour

The convergence behaviour of the three architectures was examined through their learning curves. Fig. 5, Fig. 6, and Fig. 7 present the training and validation MSE losses of the LSTM, GRU, and N-BEATS models, respectively, over the 200 training epochs. The losses are computed on the min–max-scaled data, and each figure contains four panels, one per input look-back window.

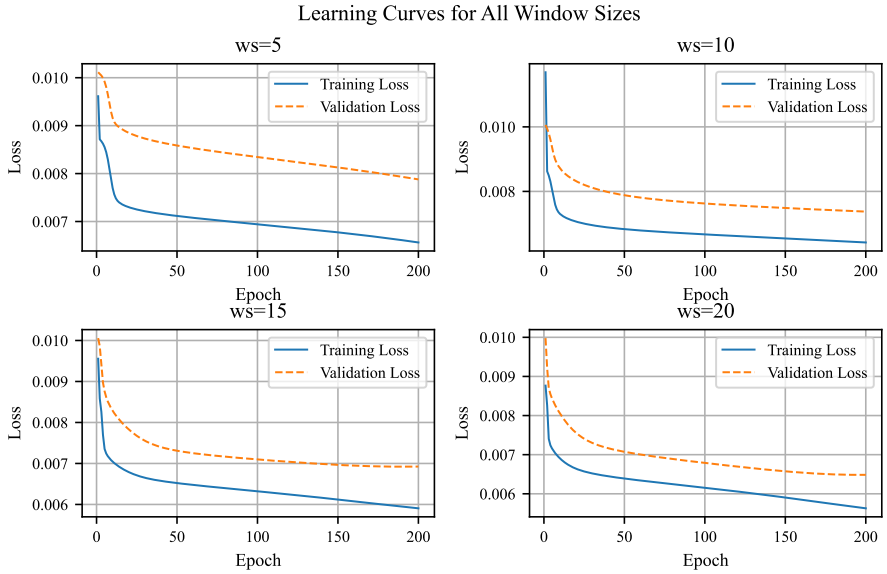


Fig. 5. LSTM training and validation MSE loss curves across input look-back windows of 5, 10, 15, and 20 hours.

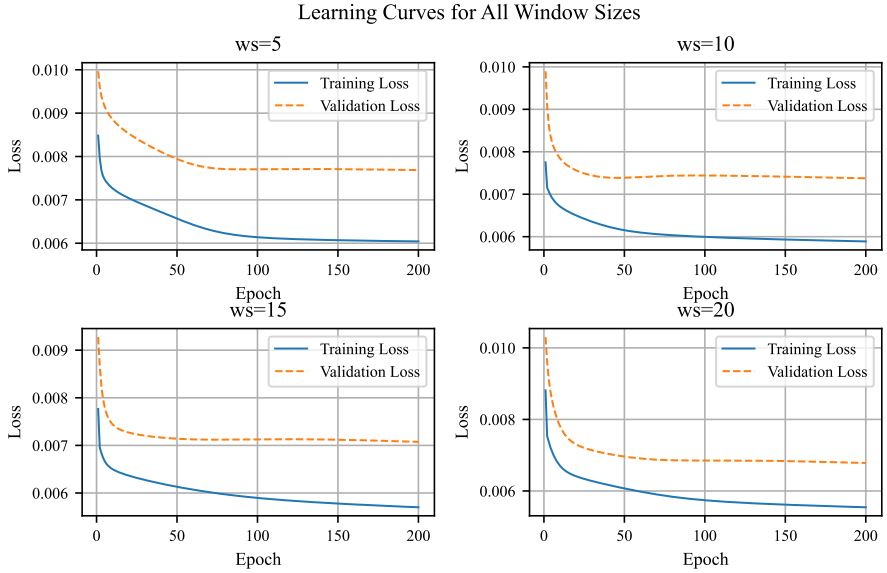


Fig. 6. GRU training and validation MSE loss curves across input look-back windows of 5, 10, 15, and 20 hours.

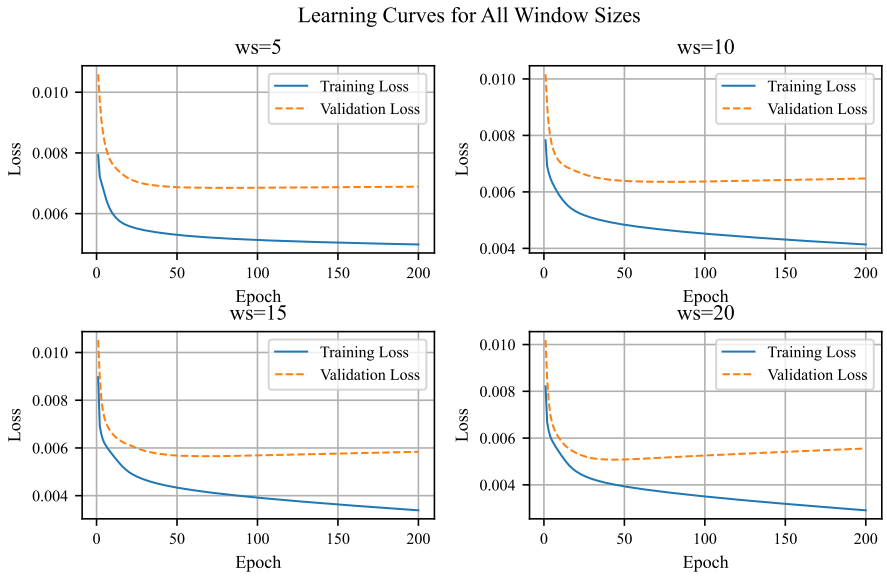


Fig. 7. N-BEATS training and validation MSE loss curves across input look-back windows of 5, 10, 15, and 20 hours.

3.5. Comparative Performance Analysis

Table 1, Table 2, and Table 3 report the test-set performance of the LSTM, GRU, and N-BEATS models, respectively, for the four input look-back windows. Five evaluation metrics are reported: MSE, RMSE, MAE, MAPE, and R^2. The error metrics are computed after inverse scaling, so RMSE and MAE are expressed in µg/m³; lower error values and higher R^2 values indicate better forecasting performance.

TABLE 1. TEST-SET PERFORMANCE OF THE LSTM MODEL FOR ONE-HOUR-AHEAD PM2.5 FORECASTING ACROSS INPUT LOOK-BACK WINDOWS. ERROR METRICS ARE COMPUTED AFTER INVERSE SCALING

Metric	**5**	**10**	**15**	**20**
MSE	5105.48	4948.84	4482.87	4116.90
RMSE	71.45	70.35	66.95	64.16
MAE	53.57	52.34	48.64	45.63
MAPE	148.03 %	143.09 %	130.33 %	117.87 %
R^2 Score	0.1839	0.2091	0.2836	0.3421

TABLE 2. TEST-SET PERFORMANCE OF THE GRU MODEL FOR ONE-HOUR-AHEAD PM2.5 FORECASTING ACROSS INPUT LOOK-BACK WINDOWS. ERROR METRICS ARE COMPUTED AFTER INVERSE SCALING

Metric	**5**	**10**	**15**	**20**
MSE	4512.22	4364.94	4180.85	4028.94
RMSE	67.17	66.07	64.66	63.47
MAE	48.87	47.91	46.38	45.75
MAPE	127.82 %	122.99 %	117.53 %	118.04 %
R^2 Score	0.2788	0.3024	0.3319	0.3562

TABLE 3. TEST-SET PERFORMANCE OF THE N-BEATS MODEL FOR ONE-HOUR-AHEAD PM2.5 FORECASTING ACROSS INPUT LOOK-BACK WINDOWS. ERROR METRICS ARE COMPUTED AFTER INVERSE SCALING

Metric	**5**	**10**	**15**	**20**
MSE	3981.10	3644.41	3159.71	3056.28
RMSE	63.10	60.37	56.21	55.28
MAE	44.87	42.60	39.57	38.77
MAPE	102.68 %	92.25 %	84.46 %	82.32 %
R^2 Score	0.3637	0.4176	0.4951	0.5116

As can be seen when comparing tables, there are several common trends. For all three models – LSTM, GRU and N-BEATS – all three error metrics – MSE, MAE, and MAPE – improve as the window size is increased from 5 to 20 hours, and all three R^2 metrics increase with time. When we specifically examine how much the RMSE improved for each of the three models, we see that the RMSE decreased from 71.45 at *ws* = 5 to 64.16 at *ws* = 20 for the LSTM model, which represents an approximate 10 % improvement. A similar pattern also exists for both the GRU (RMSE decreases from 67.17 to 63.47 – a 5.5 % improvement) and N-BEATS (RMSE decreases from 63.10 to 55.28 – a 12.4 % improvement) models. One interesting aspect of this is that the N-BEATS model seems to benefit the most from larger windows; it is clear that the basis function decomposition utilized by N-BEATS utilizes the added temporal data very effectively – as opposed to the RNNs, where it appears they have already saturated at some point. While the performance of all models does improve with window size, the speed of improvement slows down. Specifically, for the N-BEATS model, the amount of RMSE improvement per hour of added data was 2.73 μg/m³ when going from *ws* = 5 to *ws* = 10, then 4.16 μg/m³ when going from *ws* = 10 to *ws* = 15, and then only 0.93 μg/m³ when going from *ws* = 15 to *ws* = 20. It appears that nearly all of the useful signal for predicting is contained within a 15–20 hour look back window and adding additional hours will result in little or no additional gain in terms of accuracy while requiring much more memory. N-BEATS has the lowest RMSE at *ws* = 20 with an RMSE of 55.28 μg/m³, an MAE of 38.77 μg/m³, and an R^2 of 0.5116. The second best performing model is GRU, with RMSE = 63.47, MAE = 45.75, and R^2 = 0.3562. The worst performing model at *ws* = 20 is the LSTM model, with RMSE = 64.16, MAE = 45.63, and R^2 = 0.3421. Therefore, N-BEATS provides a 12.9 % reduction in RMSE relative to GRU and a 13.8 % reduction in RMSE relative to LSTM. Furthermore, the R^2 improvement is significant as well: N-BEATS explains approximately 51 % of the variance in PM2.5 concentration levels versus approximately 36 % for GRU and 34 % for LSTM. This difference clearly indicates the advantages of using the N-BEATS architecture to capture the complexities in the nonlinear dynamics of particulate matter.

Finally, the R^2 score across all models and window sizes remains under 0.52, indicating that the tested configurations explain only about half of the test-set variation in PM2.5 concentration. This should be interpreted in light of the study design. The experiments are multivariate because the inputs include PM2.5 history together with meteorological and wind-related variables; however, they are still limited to a single monitoring location and a fixed one-hour-ahead forecasting task. The dataset does not include several important external drivers, such as traffic intensity, emission inventories, boundary-layer height, land-use information, or chemical precursor concentrations. These missing drivers, together with the rarity of extreme pollution episodes, likely explain part of the remaining unexplained

variance. The present setup should therefore be viewed as a controlled multivariate benchmark rather than a complete operational air-quality forecasting system.

3.6. Time Series Analysis

To complement the aggregate metrics in Tables 1–3, Fig. 8, Fig. 9, and Fig. 10 compare the observed PM2.5 concentrations on the test set with the one-hour-ahead forecasts generated by the LSTM, GRU, and N-BEATS models, respectively. Each figure contains four panels, one per input look-back window, allowing a visual assessment of how closely each model tracks both typical fluctuations and extreme pollution episodes.

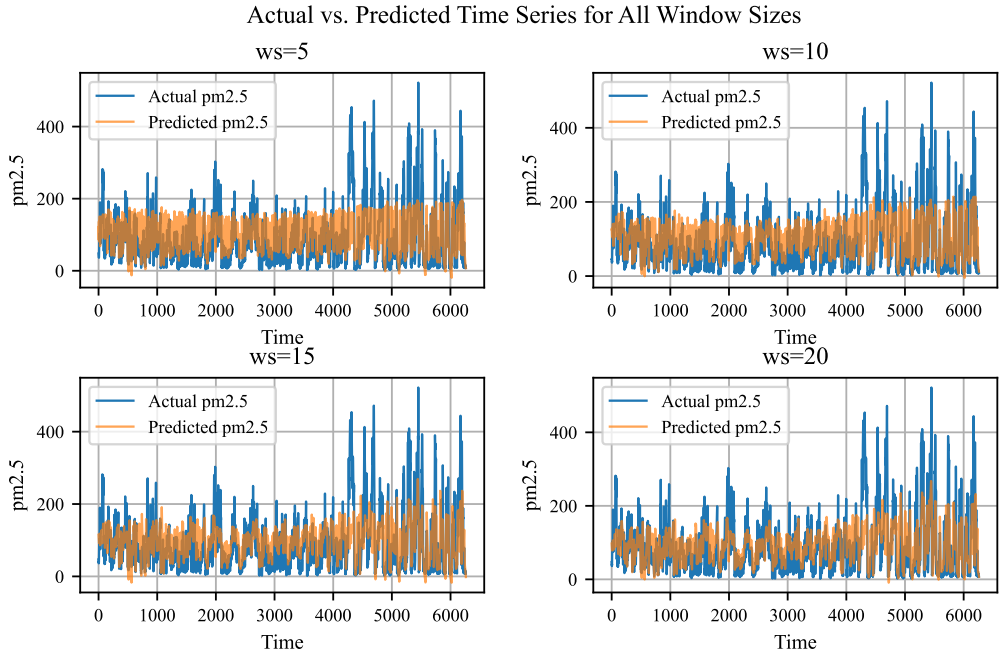


Fig. 8. Observed and one-hour-ahead predicted PM2.5 concentrations on the test set generated by the LSTM model across the four input look-back windows.

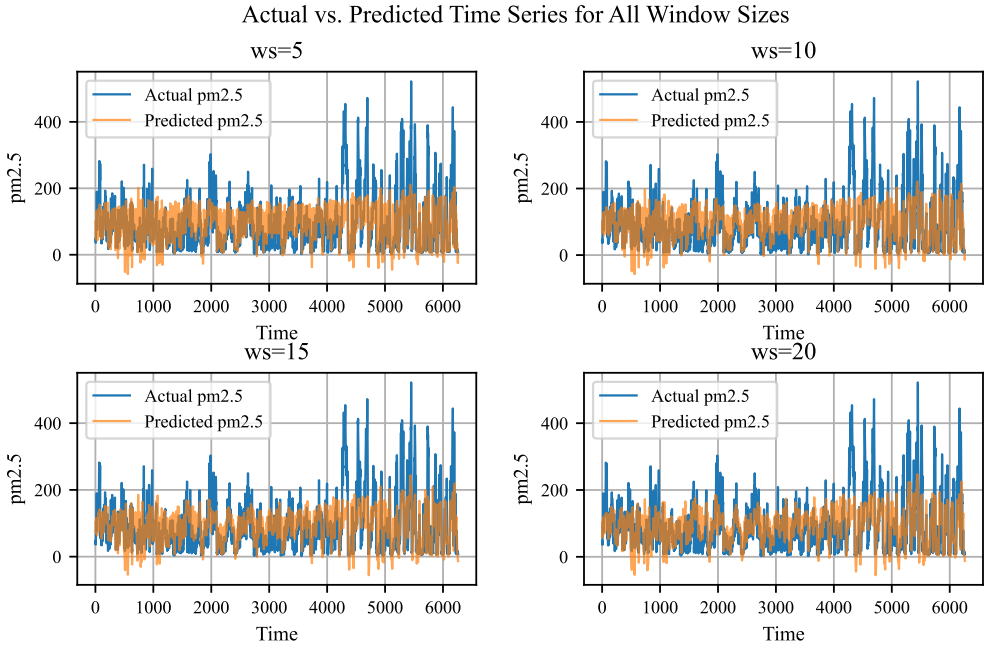


Fig. 9. Observed and one-hour-ahead predicted PM2.5 concentrations on the test set generated by the GRU model across the four input look-back windows.

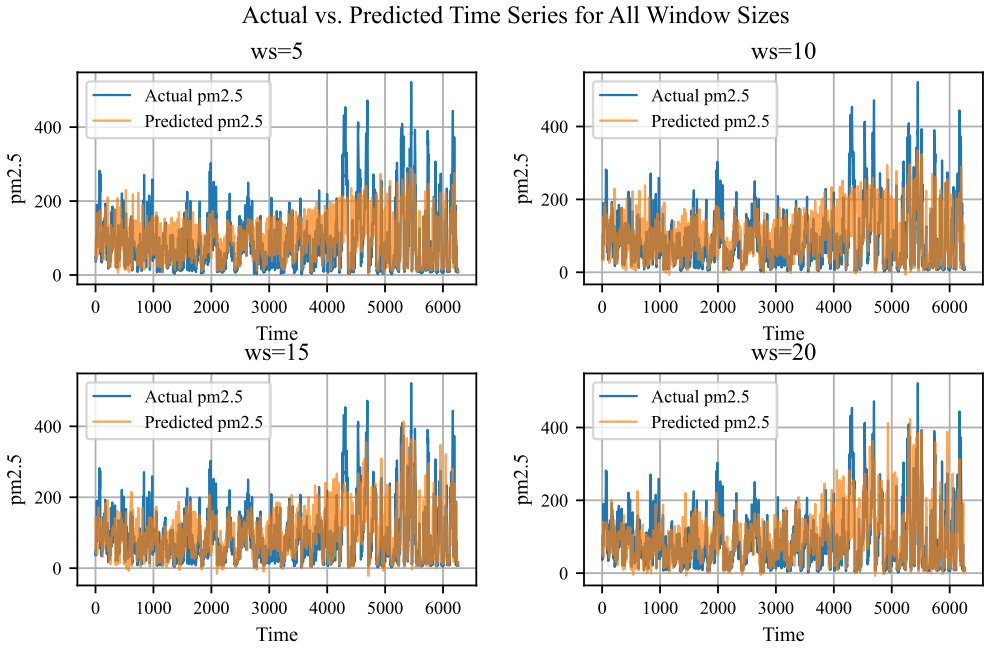


Fig. 10. Observed and one-hour-ahead predicted PM2.5 concentrations on the test set generated by the N-BEATS model across the four input look-back windows.

The plots show that N-BEATS follows high-concentration episodes more closely than the recurrent models, while all models improve as the look-back window increases from 5 to 20 hours. The largest errors still occur during rare extreme pollution spikes, indicating that additional explanatory variables are needed for more reliable peak-event forecasting.

3.7. Discussion

The results show that N-BEATS outperforms LSTM and GRU on the tested accuracy metrics while requiring substantially less training time. This performance advantage is consistent with the feed-forward block structure of N-BEATS, which processes fixed-length input windows without recurrent state updates [28].

The maximum R^2 value for N-BEATS is about 0.51, so a considerable amount of PM2.5 variation remains unexplained. This limitation is not due to a univariate setup; the experiment is multivariate, but the dataset covers one station and excludes important external drivers such as traffic, emission inventories, boundary-layer height, and chemical precursor variables. N-BEATS provides one possible path for incorporating additional exogenous variables in future model development [30]. Recent studies in Environmental and Climate Technologies also emphasize the importance of clear air-quality assessment criteria, forecasting methodology, and data quality in AI-driven environmental and building-energy applications [32]–[36].

4. Conclusion and Future Work

Two overarching conclusions emerge from our experiments. First, extending the look-back window improves prediction quality, but the marginal gains diminish as the window grows beyond 15–20 steps. Second, N-BEATS not only yields lower aggregate errors than LSTM and GRU, but also tracks the timing and severity of dangerous pollution episodes more faithfully—a property that is particularly valuable for real-time air-quality alert systems.

In this study, we carried out a direct, controlled comparison of LSTM, GRU, and N-BEATS for short-horizon PM2.5 forecasting. All three architectures were evaluated under the same one-hour-ahead forecast horizon and four input look-back windows. In general, adding additional historical context resulted in lower error; however, the improvement levelled off when the input window reached approximately 15–20 hours. Across the tested input-window configurations, N-BEATS consistently produced the lowest error and the largest R^2 values compared to both LSTM and GRU. At the 20-hour look-back window (the longest), N-BEATS generated an RMSE of 55.28 μg/m³ and an R^2 value of 0.5116, resulting in a 13–14 % reduction in RMSE compared to both GRU (63.47 μg/m³) and LSTM (64.16 μg/m³). GRU was able to slightly outperform LSTM at every window-size. While N-BEATS was the most accurate of the three architectures, it also trained much faster than the recurrent architectures (N-BEATS < 70 sec vs. GRU/LSTM < 2500 sec) regardless of window size, providing a significant advantage for conducting iterative experiments or for repeatedly retraining the model in operational use.

Future work should evaluate longer multi-step forecast horizons, include additional external drivers such as traffic, emissions, boundary-layer height, and chemical precursor variables, and test transferability across cities.

REFERENCES

[1] Calzada I. Democratising Smart Cities? Penta-Helix Multistakeholder Social Innovation Framework. *Smart Cities* 2020:3(4):1145–1172. https://doi.org/10.3390/smartcities3040057

[2] Zou Z., Cheng C., Shen S. The Complex Nonlinear Coupling Causal Patterns Between PM2.5 and Meteorological Factors in Tibetan Plateau: A Case Study in Xining. *IEEE Access* 2021:9:150373–150382. https://doi.org/10.1109/ACCESS.2021.3123455

[3] Li T., Hua M., Wu X. A Hybrid CNN-LSTM Model for Forecasting Particulate Matter (PM2.5). *IEEE Access* 2020:8:26933–26940. https://doi.org/10.1109/ACCESS.2020.2971348

[4] Qin D., Yu J., Zou G., Yong R., Zhao Q., Zhang B. A Novel Combined Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration. *IEEE Access* 2019:7:20050–20059. https://doi.org/10.1109/ACCESS.2019.2897028

[5] Yang J., Ismail A. W., Li Y., Zhang L., Fadzli F. E. Transfer Learning-Driven Hourly PM2.5 Prediction Based on a Modified Hybrid Deep Learning. *IEEE Access* 2023:11:99614–99627. https://doi.org/10.1109/ACCESS.2023.3314490

[6] Putri T. H., Caraka R. E., Toharudin T., Kim Y., Chen R.-C., Gio P. U., Sakti A. D., Pontoh R. S., Pratiwi I. R., Nugraha F. A. L., Azzahra T. S., Cerelia J. J., Darmawan G., Faidah D. Y., Pardamean B. Fine-Tuning of Predictive Models CNN-LSTM and CONV-LSTM for Nowcasting PM2.5 Level. *IEEE Access* 2024:12:28988–29003. https://doi.org/10.1109/ACCESS.2024.3368034

[7] Wang L., Jin X., Huang Z., Zhu H., Chen Z., Liu Y., Feng H. Short-Term PM2.5 Prediction Based on Multi-Modal Meteorological Data for Consumer-Grade Meteorological Electronic Systems. *IEEE Transactions on Consumer Electronics* 2024:70(1):3464–3474. https://doi.org/10.1109/TCE.2024.3354073

[8] Qiao W., Tian W., Tian Y., Yang Q., Wang Y., Zhang J. The Forecasting of PM2.5 Using a Hybrid Model Based on Wavelet Transform and an Improved Deep Learning Algorithm. *IEEE Access* 2019:7:142814–142825. https://doi.org/10.1109/ACCESS.2019.2944755

[9] Kristiani E., Kuo T.-Y., Yang C.-T., Pai K.-C., Huang C.-Y., Nguyen K. L. P. PM2.5 Forecasting Model Using a Combination of Deep Learning and Statistical Feature Selection. *IEEE Access* 2021:9:68573–68582. https://doi.org/10.1109/ACCESS.2021.3077574

[10] Nguyen M. H., Le Nguyen P., Nguyen K., Le V. A., Nguyen T.-H., Ji Y. PM2.5 Prediction Using Genetic Algorithm-Based Feature Selection and Encoder-Decoder Model. *IEEE Access* 2021:9:57338–57350. https://doi.org/10.1109/ACCESS.2021.3072280

[11] Yang M.-C., Chen M. C. Composite Neural Network: Theory and Application to PM2.5 Prediction. *IEEE Transactions on Knowledge and Data Engineering* 2023:35(2):1311–1323. https://doi.org/10.1109/TKDE.2021.3099135

[12] Sirisumpun N., Wongwailikhit K., Painmanakul P., Vateekul P. Spatio-Temporal PM2.5 Forecasting in Thailand Using Encoder-Decoder Networks. *IEEE Access* 2023:11:69601–69613. https://doi.org/10.1109/ACCESS.2023.3293398

[13] Ma J., Ding Y., Gan V. J. L., Lin C., Wan Z. Spatiotemporal Prediction of PM2.5 Concentrations at Different Time Granularities Using IDW-BLSTM. *IEEE Access* 2019:7:107897–107907. https://doi.org/10.1109/ACCESS.2019.2932445

[14] Chiang Y.-L., Wang J.-C., Lee M.-H., Liu A.-C., Jiang J.-A. Deep-Learning-Based Multi-Timestamp Multi-Location $PM_{2.5}$ Prediction: Verification by Using a Mobile Monitoring System With an IoT Framework Deployed in the Urban Zone of a Metropolitan Area. *IEEE Internet of Things Journal* 2024:11(5):8815–8837. https://doi.org/10.1109/JIOT.2023.3322862

[15] Chang S.-W., Chang C.-L., Li L.-T., Liao S.-W. Reinforcement Learning for Improving the Accuracy of PM2.5 Pollution Forecast Under the Neural Network Framework. *IEEE Access* 2020:8:9864–9874. https://doi.org/10.1109/ACCESS.2019.2932413

[16] Wang H., Chen H. A Novel Particulate Matter 2.5 Concentration Prediction Model Based on Double-Layer Decomposition and Feedback of Model Learning Effect. *IEEE Access* 2022:10:12164–12178. https://doi.org/10.1109/ACCESS.2022.3146170

[17] Hochreiter S., Schmidhuber J. Long Short-Term Memory. *Neural Computation* 1997:9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[18] Wang C., Han D., Liu Q., Luo S. A Deep Learning Approach for Credit Scoring of Peer-to-Peer Lending Using Attention Mechanism LSTM. *IEEE Access* 2019:7:2161–2168. https://doi.org/10.1109/ACCESS.2018.2887138

[19] Li X., Xu X., Wang J., Li J., Qin S., Yuan J. Study on Prediction Model of HIV Incidence Based on GRU Neural Network Optimized by MHPSO. *IEEE Access* 2020:8:49574–49583. https://doi.org/10.1109/ACCESS.2020.2979859

[20] Jin Y.-C., Cao Q., Sun Q., Lin Y., Liu D.-M., Shan-Yu, Wang C.-X., Wang X.-L., Wang X.-Y. Models for COVID-19 Data Prediction Based on Improved LSTM-ARIMA Algorithms. *IEEE Access* 2024:12:3981–3991. https://doi.org/10.1109/ACCESS.2023.3347403

[21] Ma L., Tian S. A Hybrid CNN-LSTM Model for Aircraft 4D Trajectory Prediction. *IEEE Access* 2020:8:134668–134680. https://doi.org/10.1109/ACCESS.2020.3010963

[22] Cho K., van Merriënboer B., Gulcehre C., Bahdanau D., Bougares F., Schwenk H., Bengio Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (EMNLP) 2014:1724–1734. https://doi.org/10.3115/v1/D14-1179

[23] Liu F., Zhang Z., Zhou R. Automatic modulation recognition based on CNN and GRU. *Tsinghua Science and Technology* 2022:27(2):422–431. https://doi.org/10.26599/TST.2020.9010057

[24] Cong Y., Zhao X., Tang K., Wang G., Hu Y., Jiao Y. FA-LSTM: A Novel Toxic Gas Concentration Prediction Model in Pollutant Environment. *IEEE Access* 2022:10:1591–1602. https://doi.org/10.1109/ACCESS.2021.3133497

[25] Sheng A., Xie L., Zhou Y., Wang Z., Liu Y. A Hybrid Model Based on Complete Ensemble Empirical Mode Decomposition With Adaptive Noise, GRU Network and Whale Optimization Algorithm for Wind Power Prediction. *IEEE Access* 2023:11:62840–62854. https://doi.org/10.1109/ACCESS.2023.3287319

[26] Zhang Y., Suzuki G., Shioya H. Prediction and Detection of Sewage Treatment Process Using N-BEATS Autoencoder Network. *IEEE Access* 2022:10:112594–112608. https://doi.org/10.1109/ACCESS.2022.3216924

[27] Motavali A., Yow K.-C., Hansmeier N., Chao T.-C. DSA-BEATS: Dual Self-Attention N-BEATS Model for Forecasting COVID-19 Hospitalization. *IEEE Access* 2023:11:137352–137365. https://doi.org/10.1109/ACCESS.2023.3318931

[28] Oreshkin B.N., Carpov D., Chapados N., Bengio Y. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *Proceedings of the International Conference on Learning Representations* (ICLR) 2020. https://doi.org/10.48550/arXiv.1905.10437

[29] Ji W., Zhu D. ECG Classification Exercise Health Analysis Algorithm Based on GRU and Convolutional Neural Network. *IEEE Access* 2024:12:59842–59850. https://doi.org/10.1109/ACCESS.2024.3392965

[30] Olivares K.G., Challu C., Marcjasz G., Weron R., Dubrawski A. Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with NBEATSx. *International Journal of Forecasting* 2023:39(2):884-900. https://doi.org/10.1016/j.ijforecast.2022.03.001

[31] Chen S. Beijing PM2.5 [Dataset]. UCI Machine Learning Repository, 2015. https://doi.org/10.24432/C5JS49

[32] Sprudza K. L., Klavina A., Berzina B., Kauce R., Martinsone Z. Indoor Air Quality Guidelines Connection to IAQ Certification and Labelling Process. *Environmental and Climate Technologies* 2023:27(1):28–39. https://doi.org/10.2478/rtuect-2023-0003

[33] Ali H., Dedov A., Volkova A. Exploring Heat Demand Forecasting in District Heating Networks Using Random Parameter Linear Regression Model. *Environmental and Climate Technologies* 2024:28(1):670–685. https://doi.org/10.2478/rtuect-2024-0052

[34] Ali D. M. T. E., Motuzienė V. Data Challenges in AI-Driven HVAC Systems: A Critical Analysis and Future Directions. *Environmental and Climate Technologies* 2025:29(1):527–539. https://doi.org/10.2478/rtuect-2025-0036

[35] Chiteka K., Enweremadu C. An Empirical Approach to Solar Photovoltaic Cell Temperature Prediction. *Environmental and Climate Technologies* 2024:28(1):422–436. https://doi.org/10.2478/rtuect-2024-0033

[36] Beltrán-Velamazán C., Gómez-Gil M., Monzón-Chavarrías M., Espinosa-Fernández A., López-Mesa B. Harnessing Open European Data for a Data-Driven Approach to Enhancing Decarbonization Measurement in the Built Environment. *Environmental and Climate Technologies* 2024:28(1):776–793. https://doi.org/10.2478/rtuect-2024-0060