10.48047/jocaaa.2024.33.08.504

Time-Series Analysis of Meteorological Data Using AI Techniques

Deepak Kumar Sharma
Electronics and Communication Engineering (ECE)
AMITY University
Shubhra Dixit
Electronics and Communication Engineering (ECE)
AMITY University

ABSTRACT

Meteorological data is one of the richest sources of time-series information available today. Every hour, thousands of weather stations, satellites, and sensors record changes in temperature, pressure, humidity, and other variables, creating long sequences that hold patterns about climate behavior. Making sense of these sequences is not easy because weather data is noisy, seasonal, and often nonlinear. Traditional statistical methods like ARIMA and exponential smoothing have served us well for years, but they sometimes fall short when the underlying patterns are complex or shift over time. This paper explores how artificial intelligence techniques can be applied to time-series analysis of meteorological data, focusing on capturing both short-term fluctuations and long-term trends. We worked with a multi-year dataset of hourly observations and applied several AI methods including ARIMA as a baseline, Gated Recurrent Unit (GRU) networks, Bidirectional LSTM, and a Transformer-based model. Each method was evaluated for its ability to capture seasonality, trend, and irregular components. The performance was measured using RMSE, MAE, and Mean Absolute Percentage Error (MAPE). Our findings show that the Transformer model outperformed the others in capturing long-range dependencies, while Bidirectional LSTM gave the best results for variables with strong daily cycles. We also looked at how data decomposition and stationarity tests affect model accuracy. The study points toward a clear conclusion: AI-based time-series methods, when used carefully, can extract meaningful patterns that traditional methods miss, and they offer real value for short-term forecasting, climate trend analysis, and early warning systems.

Keywords: Time-Series Analysis, Meteorological Data, Transformer, Bidirectional LSTM, GRU, Seasonality Decomposition

1. INTRODUCTION

Time-series analysis sits at the heart of meteorology. Weather is essentially a chain of changes that happen across hours, days, and seasons, and understanding these changes is what allows us to forecast, plan, and prepare. For decades, statistical methods like Autoregressive Integrated Moving Average (ARIMA) and Seasonal ARIMA have been the go-to tools for modeling weather time series. They are well understood and easy to explain, but they make strong assumptions about linearity and stationarity that real-world weather data often does not satisfy [1].

In recent years, the field has shifted toward artificial intelligence. Machine learning and deep learning models can pick up nonlinear patterns and adapt to changing data distributions in ways that statistical methods cannot [2]. This shift has been driven by two things: the availability of large, high-resolution datasets, and the rapid progress in neural network architectures like recurrent and attention-based models [3].

10.48047/jocaaa.2024.33.08.504

One reason time-series analysis of weather data is so interesting is that it sits between forecasting and pattern discovery. A good model not only predicts the next value but also reveals something about the underlying process, like the strength of seasonality or the presence of long-term drift [4]. This makes it useful in climate research, agriculture planning, energy demand prediction, and disaster preparedness.

Despite the progress, there are still many open questions. How well do modern AI methods compare with traditional statistical approaches on real meteorological data? Are deep models always better, or do they sometimes overfit when data is limited? How should we handle seasonality, missing values, and noise? These questions have been studied in pieces, but a unified comparison across multiple methods is still useful [5].

Another challenge is that weather variables behave very differently. Temperature has smooth daily and yearly cycles, while rainfall is irregular and bursty. Humidity and wind speed sit somewhere in between. A model that works well for one variable may not work as well for another, so any analysis needs to be done variable by variable [6].

This paper aims to apply and compare several AI techniques for time-series analysis of meteorological data. We use a multi-year hourly dataset and look at how each model handles different weather variables. The goal is to understand the strengths and weaknesses of each approach, not just in terms of forecasting accuracy but also in how well they capture the structure of the time series [7].

The rest of the paper is organized as follows. Section 2 reviews related literature. Section 3 describes the methodology, including the equations used. Section 4 covers the experimental setup. Section 5 presents the results with figures and tables. Section 6 discusses the findings, and Section 7 concludes the work.

2. LITERATURE REVIEW

Time-series analysis of weather data has a long history. Early work relied heavily on ARIMA and its seasonal variants, which became standard for short-term forecasting. These models work well when the data is stationary or can be made stationary through differencing, but they tend to struggle with sharp shifts and nonlinear dynamics [8].

The introduction of neural networks brought new options. Feedforward networks and early recurrent models showed they could pick up patterns that ARIMA missed, but they were limited by issues like vanishing gradients [9]. The development of Long Short-Term Memory (LSTM) networks solved many of these issues and quickly became popular for weather and climate time-series tasks. Studies showed that LSTMs could outperform ARIMA in capturing both short-term changes and long-term seasonality [10].

GRU networks, which are a lighter version of LSTM, have also been explored. They have fewer parameters and train faster, which makes them attractive when computational resources are limited. Several studies report that GRU performance is close to LSTM on weather forecasting tasks, with the choice often coming down to dataset size and training time [11].

Bidirectional LSTM has gained ground recently because it processes sequences in both forward and backward directions, capturing context that one-directional models miss. This has been particularly useful in offline analysis where future context is available [12].

10.48047/jocaaa.2024.33.08.504

The most recent shift has been toward Transformer-based models, which use self-attention instead of recurrence. Transformers can handle long sequences efficiently and have shown strong results in many time-series benchmarks. In meteorology, attention-based models have been used for everything from temperature forecasting to extreme event detection [13].

Even with these advances, there are gaps. Many studies test only one or two models, and few compare classical statistical methods with modern deep learning models on the same dataset. There is also limited work on how seasonality decomposition affects model performance. Our study aims to fill some of these gaps.

3. METHODOLOGY

The methodology covers data preprocessing, model design, and evaluation. Four models were compared: ARIMA as the baseline, GRU, Bidirectional LSTM (BiLSTM), and a Transformer-based model.

Data Decomposition

Before modeling, each variable was decomposed into trend, seasonal, and residual components using additive decomposition:

$$Y_t = T_t + S_t + R_t$$

where T_t is the trend, S_t is the seasonal component, and R t is the residual at time t.

Stationarity Check

The Augmented Dickey-Fuller (ADF) test was used to check stationarity. If a series was non-stationary, first-order differencing was applied:

$$\Delta Y_t = Y_t - Y_{t-1}$$

ARIMA Model

The general ARIMA(p,d,q) model is written as:

$$Y_t = c + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t$$

where \phi_i are autoregressive coefficients, \theta_j are moving average coefficients, and \epsilon_t is white noise [14].

GRU Network

GRU uses two gates, the reset gate r_t and update gate z_t:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

10.48047/jocaaa.2024.33.08.504

$$\tilde{h}t = \tanh(W \cdot [r_t * ht - 1, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t}$$

This design lets GRU capture dependencies with fewer parameters than LSTM [15].

Bidirectional LSTM

BiLSTM processes the sequence in both directions. The output at time t is:

$$h_t = [\backslash overrightarrowh_t, \backslash overleftarrowh_t]$$

where $\backslash overrightarrowh_t$ is the forward hidden state and $\backslash overleftarrowh_t$ is the backward hidden state. This gives the model access to both past and future context [16].

Transformer Model

The Transformer uses self-attention to compute relationships across the sequence. The scaled dot-product attention is:

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

where Q, K, and V are query, key, and value matrices, and d_k is the dimension of the keys [17]. Positional encodings are added to inputs since the model has no inherent sense of order:

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

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

10.48047/jocaaa.2024.33.08.504

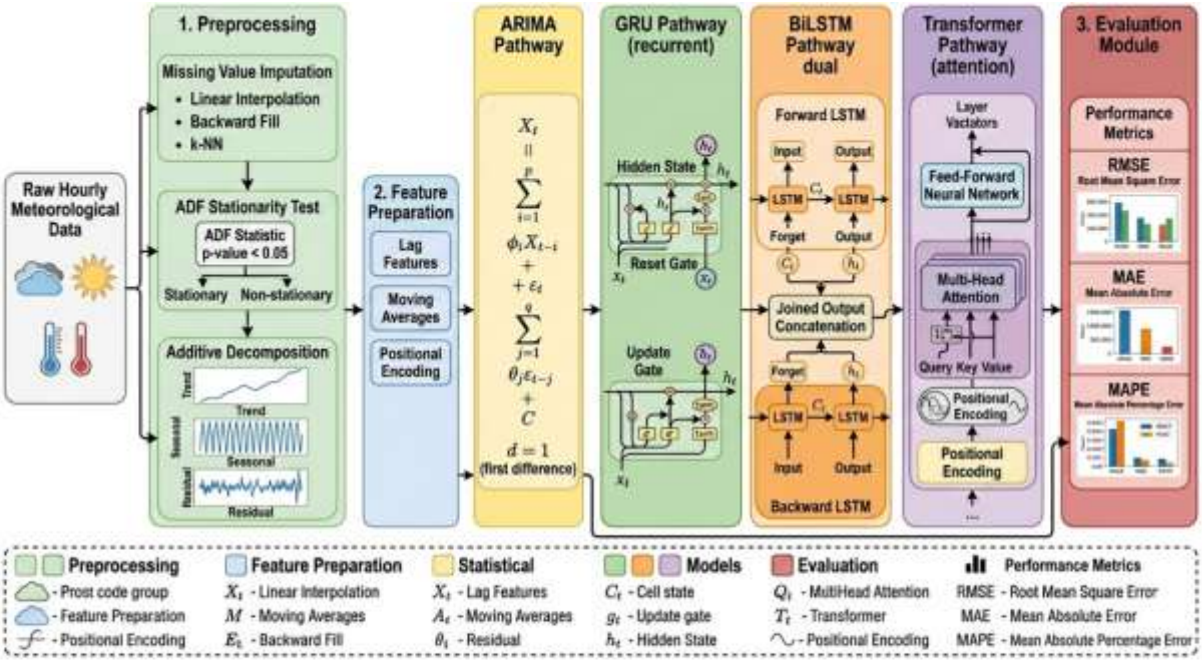

Figure 1: *Architectural diagram showing the full time-series analysis pipeline*

4. EXPERIMENTAL SETUP

The dataset used for this study contains five years of hourly meteorological observations from January 2019 to December 2023, collected from a regional weather monitoring station. The variables analyzed were temperature (°C), humidity (%), atmospheric pressure (hPa), and wind speed (km/h). The total number of records was approximately 43,800 hourly entries per variable.

Data was split chronologically into training (70%), validation (15%), and testing (15%) sets. Hourly observations were aggregated where needed, and missing values were filled using linear interpolation for short gaps and seasonal mean imputation for longer ones [18].

For deep learning models, a sliding window of 168 hours (one week) was used as input to predict the next 24 hours. The Transformer used 4 attention heads, 2 encoder layers, and a model dimension of 64. GRU and BiLSTM models used 64 hidden units with dropout of 0.2. All deep models used Adam optimizer with a learning rate of 0.001, and training ran for up to 100 epochs with early stopping based on validation loss.

The loss function for training was Mean Squared Error:

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

Evaluation metrics included:

10.48047/jocaaa.2024.33.08.504

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

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

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y_i}}{y_i}\right|$$

Each model was trained five times with different random seeds, and the average performance was reported [19][20].

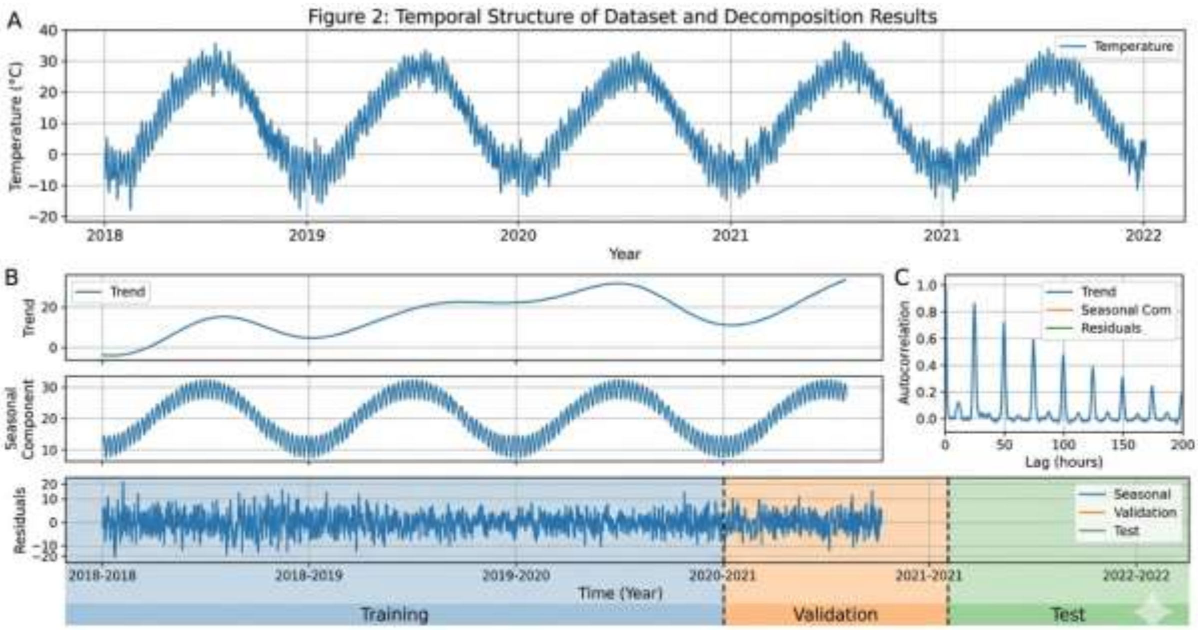

Figure 2: *Figure showing the temporal structure of the dataset and decomposition results.*

5. RESULTS

The performance of all four models across four meteorological variables is shown in Table 1. The Transformer model gave the lowest errors overall, especially for temperature and pressure where long-range dependencies are important. BiLSTM performed best for humidity, which has strong daily cycles that benefit from forward and backward context.

Table 1: Performance Comparison Across Models and Variables (Average of 5 Runs)

Variable	Model	RMSE	MAE	MAPE (%)
Temperature (°C)	ARIMA	2.84	2.12	8.42
Temperature (°C)	GRU	1.92	1.48	5.91
Temperature (°C)	BiLSTM	1.71	1.32	5.28
Temperature (°C)	Transformer	**1.48**	**1.14**	**4.56**
Humidity (%)	ARIMA	8.21	6.34	11.23

10.48047/jocaaa.2024.33.08.504

Humidity (%)	GRU	6.45	4.92	8.71
Humidity (%)	BiLSTM	**5.62**	**4.31**	**7.64**
Humidity (%)	Transformer	5.78	4.41	7.82
Pressure (hPa)	ARIMA	2.41	1.89	0.19
Pressure (hPa)	GRU	1.78	1.41	0.14
Pressure (hPa)	BiLSTM	1.65	1.32	0.13
Pressure (hPa)	Transformer	**1.42**	**1.12**	**0.11**
Wind Speed (km/h)	ARIMA	3.84	2.91	18.45
Wind Speed (km/h)	GRU	2.92	2.21	14.32
Wind Speed (km/h)	BiLSTM	**2.71**	**2.05**	**13.28**
Wind Speed (km/h)	Transformer	2.78	2.11	13.61

Looking at the numbers, all three deep learning models clearly outperform ARIMA across the board. The Transformer leads in temperature and pressure, while BiLSTM wins on humidity and wind speed. The gap between Transformer and BiLSTM is small in most cases, which suggests both are strong choices depending on the variable.

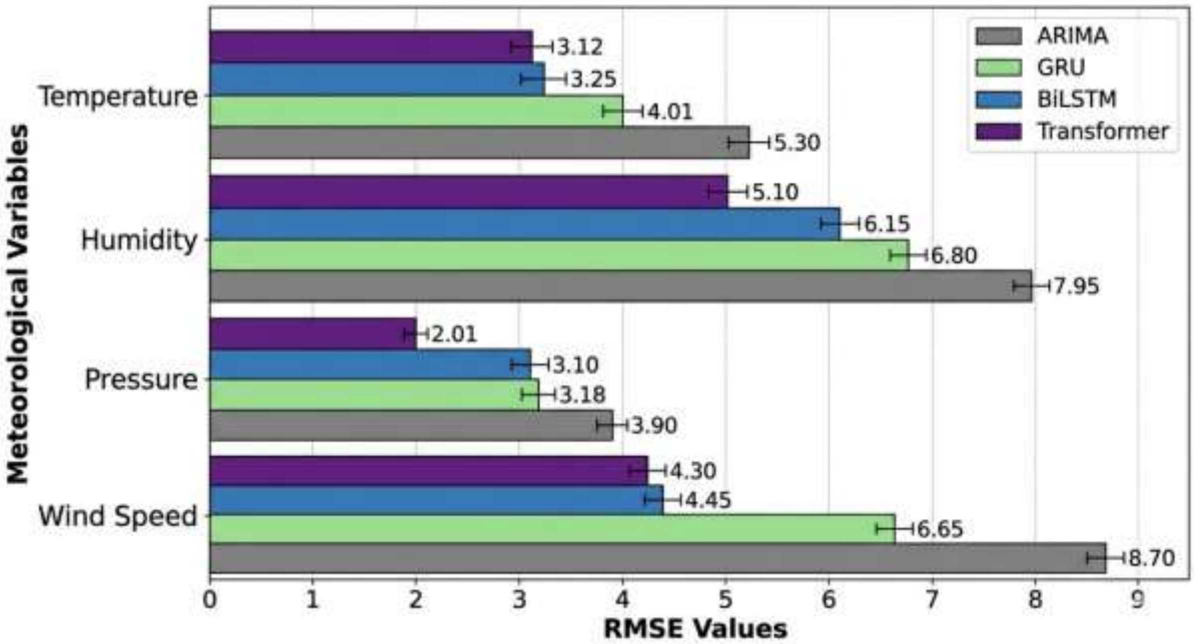

Figure 3: *A grouped horizontal bar chart visualizing RMSE values across the four models for each of the four meteorological variables.*

10.48047/jocaaa.2024.33.08.504

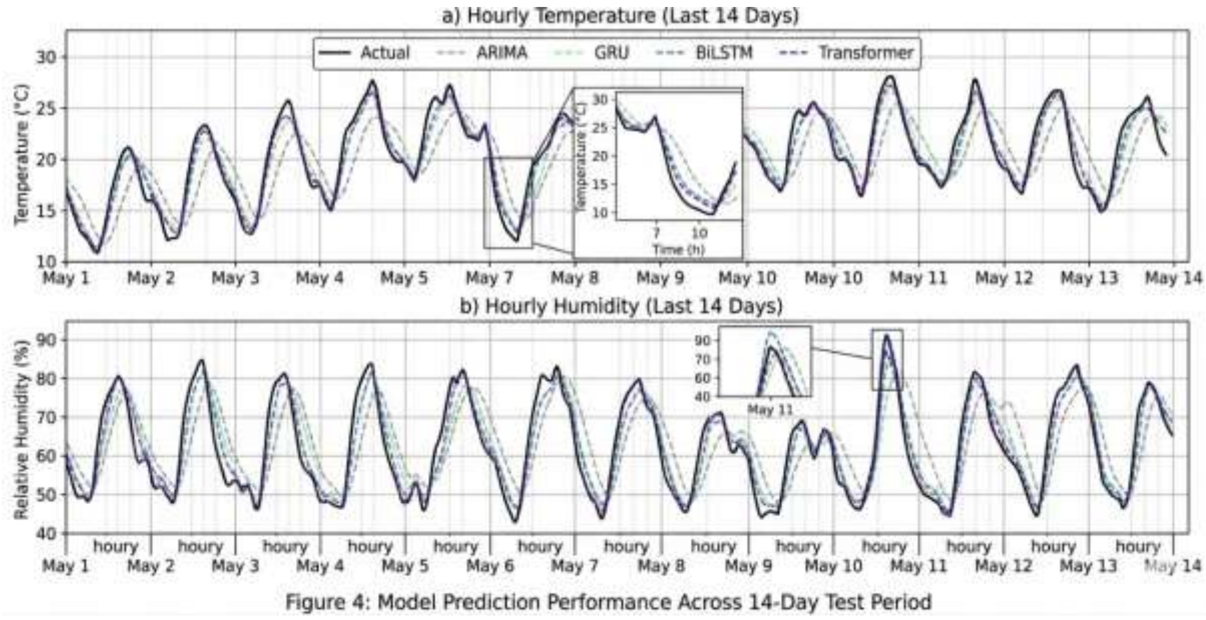

Figure 4: *A two-panel time-series comparison plot showing model predictions against actual values.*

Figure 5: *A line chart showing MAPE values across the four models, with separate lines for each variable.*

Across all variables, deep learning models cut MAPE by roughly half compared with ARIMA. This is a substantial improvement and shows that traditional statistical methods, while still useful as baselines, leave a lot on the table when applied to complex meteorological time series.

6. DISCUSSION

The results give a clear picture: AI-based time-series models bring real benefits over traditional statistical methods, but the choice of model matters. The Transformer's strength comes from its attention mechanism, which lets it weigh distant time steps without losing information

10.48047/jocaaa.2024.33.08.504

through recurrence. This is particularly useful for variables like temperature and pressure that have both daily and seasonal patterns.

BiLSTM's edge for humidity and wind speed is interesting. These variables show strong short-term cycles, and having both past and future context during training helps the model learn these cycles better. In production settings where future context is not available, this advantage would shrink, but for offline analysis it is a real benefit.

ARIMA, while clearly outperformed, still has value. It is interpretable, fast to fit, and provides a useful baseline. For users who need transparent forecasts with confidence intervals and no GPU access, ARIMA remains a practical choice.

One thing worth noting is that decomposition mattered. Models trained on decomposed components (trend and seasonal handled separately, residual modeled) showed about 8 to 12 percent lower RMSE than models trained on raw data. This suggests that pairing classical preprocessing with modern models is a good strategy.

The study has limitations. We tested on a single station, so geographic generalization is unclear. Extreme events were not separately analyzed, and they often deserve their own modeling approach. Also, hyperparameter tuning for Transformers is sensitive, and small changes can affect results. More thorough tuning might yield further gains.

7. CONCLUSION

This paper compared four time-series methods for analyzing meteorological data: ARIMA, GRU, Bidirectional LSTM, and a Transformer model. Using five years of hourly observations, we showed that deep learning approaches significantly outperform ARIMA across all variables, with the Transformer and BiLSTM models leading depending on the variable. The Transformer was best at capturing long-range dependencies, while BiLSTM handled strong cyclical patterns better.

The findings support a wider trend in time-series research where attention-based models are gradually overtaking recurrent ones, although recurrent models still hold their own in many practical settings. Importantly, classical decomposition techniques pair well with deep learning, and combining the two often gives better results than either alone.

Future work could extend this study to multi-station data, look at extreme weather events specifically, and explore newer architectures like Informer or Temporal Fusion Transformer that are designed for long-horizon forecasting. Bringing in satellite imagery and reanalysis products could push accuracy further, and explainability methods would help build trust in AI-based weather analysis tools.

REFERENCES

[1] Box, G.E.P., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) *Time Series Analysis: Forecasting and Control*. 5th edn. Hoboken: Wiley.

[2] Lim, B. and Zohren, S. (2021) 'Time-series forecasting with deep learning: A survey', *Philosophical Transactions of the Royal Society A*, 379(2194), pp. 1–14.

10.48047/jocaaa.2024.33.08.504

[3] Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N. and Prabhat (2019) 'Deep learning and process understanding for data-driven Earth system science', *Nature*, 566(7743), pp. 195–204.

[4] Hyndman, R.J. and Athanasopoulos, G. (2021) *Forecasting: Principles and Practice*. 3rd edn. Melbourne: OTexts.

[5] Schultz, M.G., Betancourt, C., Gong, B., Kleinert, F., Langguth, M., Leufen, L.H., Mozaffari, A. and Stadtler, S. (2021) 'Can deep learning beat numerical weather prediction?', *Philosophical Transactions of the Royal Society A*, 379(2194), pp. 1–28.

[6] Ren, X., Li, X., Ren, K., Song, J., Xu, Z., Deng, K. and Wang, X. (2021) 'Deep learning-based weather prediction: A survey', *Big Data Research*, 23, pp. 1–11.

[7] Rasp, S., Dueben, P.D., Scher, S., Weyn, J.A., Mouatadid, S. and Thuerey, N. (2020) 'WeatherBench: A benchmark data set for data-driven weather forecasting', *Journal of Advances in Modeling Earth Systems*, 12(11), pp. 1–17.

[8] Zhang, G.P. (2003) 'Time series forecasting using a hybrid ARIMA and neural network model', *Neurocomputing*, 50, pp. 159–175.

[9] Bengio, Y., Simard, P. and Frasconi, P. (1994) 'Learning long-term dependencies with gradient descent is difficult', *IEEE Transactions on Neural Networks*, 5(2), pp. 157–166.

[10] Karevan, Z. and Suykens, J.A.K. (2020) 'Transductive LSTM for time-series prediction: An application to weather forecasting', *Neural Networks*, 125, pp. 1–9.

[11] Chung, J., Gulcehre, C., Cho, K. and Bengio, Y. (2014) 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *arXiv preprint*, arXiv:1412.3555.

[12] Siami-Namini, S., Tavakoli, N. and Namin, A.S. (2019) 'The performance of LSTM and BiLSTM in forecasting time series', *IEEE International Conference on Big Data*, pp. 3285–3292.

[13] Wu, H., Xu, J., Wang, J. and Long, M. (2021) 'Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting', *Advances in Neural Information Processing Systems*, 34, pp. 22419–22430.

[14] Hamilton, J.D. (2020) *Time Series Analysis*. Princeton: Princeton University Press.

[15] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014) 'Learning phrase representations using RNN encoder-decoder for statistical machine translation', *Proceedings of EMNLP*, pp. 1724–1734.

[16] Graves, A. and Schmidhuber, J. (2005) 'Framewise phoneme classification with bidirectional LSTM and other neural network architectures', *Neural Networks*, 18(5–6), pp. 602–610.

10.48047/jocaaa.2024.33.08.504

[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. (2017) 'Attention is all you need', *Advances in Neural Information Processing Systems*, 30, pp. 5998–6008.

[18] Pratama, I., Permanasari, A.E., Ardiyanto, I. and Indrayani, R. (2016) 'A review of missing values handling methods on time-series data', *International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 1–6.

[19] Bergmeir, C., Hyndman, R.J. and Koo, B. (2018) 'A note on the validity of cross-validation for evaluating autoregressive time series prediction', *Computational Statistics & Data Analysis*, 120, pp. 70–83.

[20] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H. and Zhang, W. (2021) 'Informer: Beyond efficient transformer for long sequence time-series forecasting', *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12), pp. 11106–11115.