Advancing Temporal Forecasting: A Comparative Analysis of Conventional Paradigms and Deep Learning Architectures on Publicly Accessible Datasets

Liang Gao[1], Reza Jafari[2], Amir H. Jafari[1*]

[1]Data Science Department, George Washington University, 725 21st St. NW, Washington, 20052, DC, USA.
[2]Computer Science Department, Virginia Tech University, Blacksburg, Falls church, 22043, VA, USA.

*Corresponding author(s). E-mail(s): ajafari@gwu.edu;
Contributing authors: liang.gao@gwu.edu; rjafari@vt.edu;

Abstract

In this study, we compare the performance of classical time series models (AR, MA, ARMA, ARIMA) and modern deep learning techniques (LSTM, Bi-LSTM, Seq2Seq) alongside a state-of-the-art Transformers model. These models have been individually studied and applied to specific problems, but the lack of a standardized benchmark for comparing their performance across diverse datasets creates a significant gap in the literature. This research aims to benchmark the performance of representative time-series models across different datasets. We evaluate the models using three publicly available datasets and compare their prediction accuracy using metrics i.e. MSE, RMSE, and MAE. The results show that modern deep learning techniques outperform classical models, with the simplest architecture, LSTM, achieving the lowest MSE across all datasets. Although the Transformers model demonstrated superior performance to classical models, it did not outperform modern techniques. This suggests that for the datasets used in this study, complex Transformers architectures may not be necessary to achieve optimal results. This study provides some insights into time series studies.

Keywords: Time series modeling, ARIMA, LSTM, Transformers

1 Introduction

Time series data refers to a sequence of data points collected or recorded at successive, equally spaced time intervals [1]. Time series modeling, a rapidly advancing field, has garnered significant attention from the research community in recent decades. The primary objective of time series modeling is to systematically collect and analyze historical data to develop a model that accurately represents the underlying structure of the series. This model is then used to predict future values or make forecasts based on the history of past observations.

Time series analysis is a critical area of research with applications spanning diverse fields, such as finance [2], social science [3], climate science [4], energy management [5], and healthcare [6]. Over the years, numerous models have been developed to address specific time series challenges, ranging from classical approaches like Autoregressive (AR), Moving Average (MA), and Autoregressive Integrated Moving Average (ARIMA) models to advanced deep learning architectures such as Long Short-Term Memory (LSTM) networks and Transformers.

Classical models have laid the groundwork for time series analysis. For instance, the Autoregressive(AR) model has been employed to predict the instantaneous phase and frequency of neural oscillations in EEG data, demonstrating its effectiveness in real-time forward prediction for shorter intervals [7]. Researchers have utilized the Autoregressive Moving Average (ARMA) model in combination with the Levenberg-Marquardt algorithm for speech recognition applications, demonstrating its effectiveness in handling time series data in this domain [8]. In recent years, advancements in machine learning have introduced deep learning techniques, which have shown superior performance in capturing complex temporal dependencies. For example, the LSTM network was successfully applied to forecast the COVID-19 outbreak in Canada, predicting trends, potential stopping points, and multi-day case counts [9]. Similarly, the Time Series Transformers (TST) model has been utilized to recognize fault modes in rotating machinery by combining its architecture with a time series tokenizer, significantly outperforming traditional CNN and RNN models [10].

However, there is a noticeable lack of benchmarking studies comparing these models' performances on standardized datasets. The primary objective of this study is to bridge this gap by evaluating and comparing the performance of classical time series models [11]—Autoregressive(AR), Moving Average (MA), Autoregressive Moving Average (ARMA), and Autoregressive Integrated Moving Average (ARIMA), advanced modern techniques, including Long Short-Term Memory (LSTM) [12], Bidirectional LSTM (BiLSTM) [13], sequence-to-sequence architectures [14], and the state-of-the-art Transformers model [15]. To achieve this, we applied these models to three publicly available datasets and evaluated their performance using multiple metrics, such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). Our study provides a quantitative foundation for assessing the accuracy and reliability of each model across the datasets. Through this analysis, we have identified the strengths and limitations of the models in three different datasets, which can offer insights into their applicability for diverse time series forecasting challenges. Our findings aim to inform future research, guide model selection, and enhance the development of effective time series forecasting solutions.

2 Methodology

2.1 Data

We utilized three public and realistic datasets in this study to evaluate the performance of various time series models:

- **Weather Station Beutenberg Dataset** [16]: This dataset contains meteorological measurements, such as temperature, humidity, and wind speed, recorded every 10 minutes, capturing fine-grained temporal variations in environmental conditions. The data was accessed from Kaggle, and the target variable is the temperature in Celsius.
- **Power Consumption of Tetouan City** [17]: This dataset comprises the energy consumption data of Tetouan City, recorded at 10-minute intervals, offering detailed observations of energy usage patterns. It offers insights into power usage patterns and facilitates the evaluation of energy forecasting models. The dataset was retrieved from the UCI Machine Learning Repository and used to forecast the power consumption in Zone 1.
- **Air Pollution Forecasting Dataset** [18]: This dataset includes hourly measurements of air pollutants along with meteorological features, providing insights into atmospheric quality over time. It serves as a multivariate time series dataset to assess forecasting models in pollution monitoring. The data was sourced from Kaggle, and the target variable is the pollution called PM2.5 concentration.

2.2 Models

Classical models. We implemented four classical models: Autoregressive (AR), Moving Average (MA), Autoregressive Moving Average (ARMA), and Autoregressive Integrated Moving Average (ARIMA) models.

In an AR model, we forecast the variable of interest using a linear combination of past values of the variable. An AR process of order n_a AR(n_a) can be written as:

$$y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) = \epsilon(t)$$

Rather than using past values of the forecast variable in a regression, an MA model uses past forecast errors. A MA process of order order n_b, MA(n_b) can be represented as:

$$y(t) = \epsilon(t) + b_1 \epsilon(t-1) + b_2 \epsilon(t-2) + \cdots + b_{n_b} \epsilon(t-n_b)$$

The ARMA model is the combination of AR and MA models, which can be represented as:

$$y(t)+a_1 y(t-1)+a_2 y(t-2)+\cdots+a_{n_a} y(t-n_a) = \epsilon(t)+b_1 \epsilon(t-1)+b_2 \epsilon(t-2)+\cdots+b_{n_b} \epsilon(t-n_b)$$

Compared to pure AR and MA models, the ARMA models offer an efficient linear approach for modeling stationary time series. The ARIMA model is a generalization of the ARMA model with differencing, designed to handle non-stationary time series

sequences. In the general form, the ARIMA(n_a, d, n_b) model can be written as:

$$\left(1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}\right) \left(1 - q^{-1}\right)^d y(t) = \left(1 + b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}\right) \epsilon(t)$$

Notation:

- $y(t)$ is the value of the time series at time t,
- $a_1, a_2, \ldots, a_{n_a}$ are the coefficients of the AR model,
- n_a is the order of the autoregressive model,
- $b_1, b_2, \ldots, b_{n_b}$ are the coefficients of the MA model,
- n_b is the order of the moving average model,
- $\epsilon(t)$ is a white noise normally distributed (WN $\sim (0, \sigma_\epsilon^2)$),
- d is the number of non-seasonal order differencing.

Modern techniques. Three modern techniques are implemented: Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), and Sequence-to-sequence (Seq2Seq) models. LSTM, introduced by Hochreiter and Schmidhuber in 1997 [12], is a specialized Recurrent Neural Network (RNN) designed to capture long-term dependencies in sequential data while mitigating the vanishing gradient problem [19]. Its architecture incorporates a cell state and three gates—input, forget, and output—to regulate information flow, making it well-suited for time series forecasting and natural language processing tasks. BiLSTM extends LSTM by processing data bidirectionally, utilizing forward and backward LSTMs to capture past and future context, which enhances its performance in tasks like time series forecasting, NLP, and speech recognition [13]. Seq2Seq models transform input sequences into output sequences of varying lengths and are widely used in tasks such as language translation and time series forecasting [20]. The implemented Seq2Seq architecture adopts an encoder-decoder LSTM framework enhanced by an attention mechanism.

State-of-the-Art Models. The Transformers model, introduced by Vaswani et al. [15], revolutionized deep learning with its attention mechanism and parallelized architecture. Unlike sequential recurrent models, Transformers process entire sequences simultaneously, efficiently capturing long-term dependencies. The model comprises an encoder, which applies self-attention and feed-forward layers to process input sequences, and a decoder, which generates outputs by attending to the encoder's representation and prior outputs.

2.3 Analysis

This study used time series domain knowledge, including ACF/PACF plots and generalized partial autocorrelation (GPAC), to estimate the time series order for ARMA models. Besides domain knowledge, we also used a hyperparameter optimization framework - Optuna, in the experiment. These tools are introduced below.

The Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) are essential tools in time series analysis [11, 21, 22]. The ACF quantifies the correlation between a time series and its lagged values, aiding in identifying trends, seasonality, and the order of Moving Average (MA) models. Significant autocorrelations at specific lags suggest predictability at those intervals. The PACF extends

Hyperparameter	Value
Sequence length	6
Batch Size	128
Optimizer	Adam
Training Epochs	100
Learning Rate	0.001

Table 1 Hyperparameters for LSTM, Bi-LSTM, Seq2Seq, and Transformers.

this by isolating the correlation between a series and its lagged values, removing the influence of intermediate lags, and making it particularly effective for determining the order of Autoregressive (AR) models.

The Generalized Partial Autocorrelation (GPAC) is used in time series analysis, particularly in Auto-Regressive Moving-Average (ARMA) modeling. It is used to estimate the order of the ARMA model when the auto-regressive (AR) and moving-average (MA) orders (n_a and n_b, respectively) are not known prior. The pattern observed in the GPAC table, where a column has constant values and a row consists entirely of zeros, identifies the orders of n_a for the AR component and n_b for the MA component. Specifically, the row number represents the MA order, while the column number indicates the AR order. This structure provides a systematic approach to estimating the order of ARMA model.

In addition to using time series domain knowledge (ACF/PACF plots and the GPAC table), we applied Optuna, an open-source hyperparameter optimization framework [23], to determine the order of classical linear time series models. Optuna employs Bayesian optimization and pruning techniques to efficiently search for optimal hyperparameters, making it a flexible and computationally efficient tool for model tuning.

3 Experiment results

3.1 Experiment settings

Our experiments utilized AWS cloud resources to meet the computational demands of the models. Classical time series models (AR, MA, ARMA, ARIMA) were executed on AWS CPU instances due to their lightweight nature, while deep learning models (LSTM, BiLSTM, Seq2Seq, Transformers) leveraged GPU instances for efficient training and faster convergence. Standardized hyperparameters (Table 1) were applied consistently across all modern and state-of-the-art models to ensure comparability.

For the modern techniques (LSTM, BiLSTM, Seq2Seq), we standardized the hyperparameters using a single layer and a hidden size 2 across all models. This uniform configuration was chosen to maintain comparability and isolate the model architecture's effect on performance. In the Optuna setting, we capped the maximum order for the AR process at 20, the MA process at 10, and conducted 30 optimization trials. We used the drift method in the temperature dataset to fill the 9 missing values. There are no missing values in the other two datasets. Each dataset was split into 80% for training and 20% for validation, ensuring a balanced trade-off between model

training and validation. Additionally, data was normalized to ensure fair comparison across datasets before calculating evaluation metrics using min-max scaling as below:

$$X_{\text{normalized}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}. \tag{1}$$

3.2 Results and discussion

In this study, we analyzed the characteristics of three datasets and systematically investigated the performance of various models on these datasets. In the following sections, we comprehensively discuss our findings and insights.

Exploratory Data Analysis (EDA). We conducted EDA to examine dataset characteristics and identify underlying patterns. Visualizations included sequence plots for trend observation, rolling mean and variance plots for stationarity assessment, ACF/PACF plots for model order identification and time series decomposition to analyze seasonal, trend, and residual components. Figure 1 presents these visualizations. All three datasets exhibited stationarity, confirmed by stable rolling means and variances. ACF tail-offs and PACF cut-offs indicated AR patterns. Table 2 summarizes the trend and seasonality strengths: the Weather dataset showed a strong trend (94.33%) and moderate seasonality (74.79%), the Air Pollution dataset revealed a notable trend (86.01%) but weak seasonality (40.67%), and the Power Consumption dataset demonstrated both strong trend (92.41%) and seasonality (98.64%).

	Strength of Trend (%)	**Strength of Seasonality (%)**
Weather	94.33	74.79
Air Pollution	86.01	40.67
Power Consumption	92.41	98.64

Table 2 Strength of Trend and Seasonality for Each Dataset.

Classical models results. We implemented four classical linear time series models: AR, MA, ARMA, and ARIMA, each addressing different temporal dependencies in time series data. To determine the appropriate model orders n_a for the AR component and n_b for the MA component, we used ACF and PACF plots to identify lag relationships and the GPAC table to assess lag patterns systematically.

For the AR model, the order was set based on the cutoff in the PACF plot, while for the MA model, since there was no MA pattern from ACF and PACF plots, we used Optuna to optimize the best order, with the maximum order set to 10. The GPAC table in Figure 2 was used to determine the AR and MA orders for the ARMA model. We applied first-order differencing to our data for the ARIMA model, analyzing the differenced data using the GPAC in Figure 3 to identify the AR and MA orders.

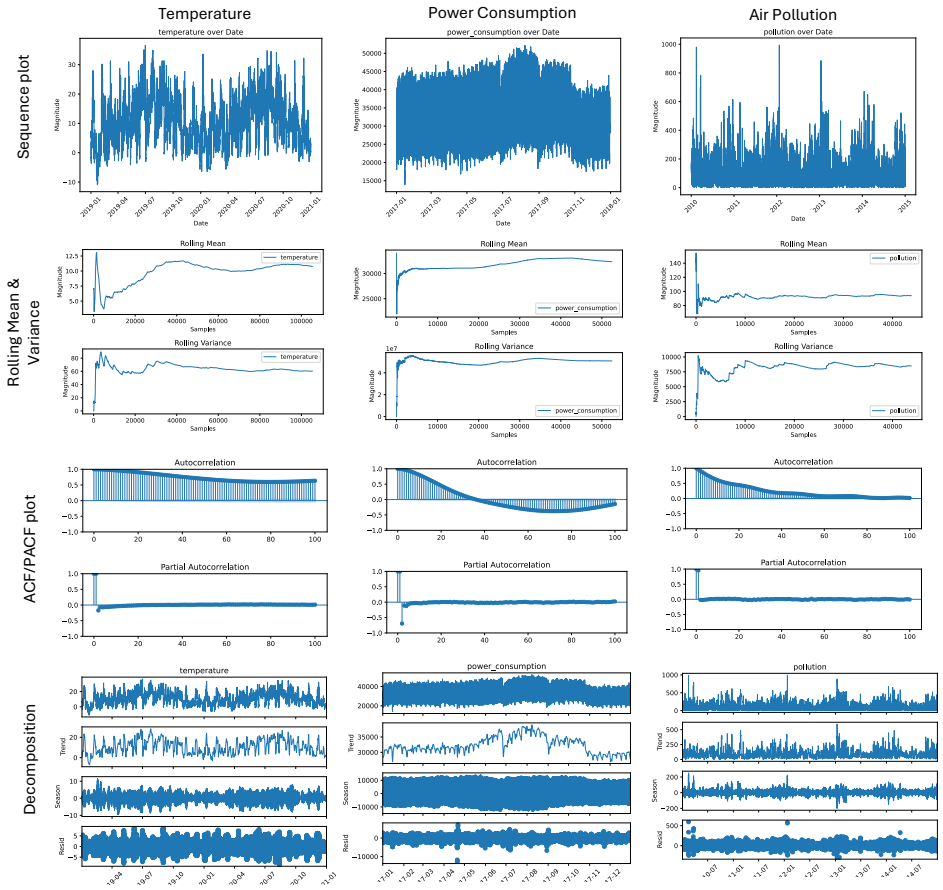

Fig. 1 EDA plots: time series sequences over time, ACF&PACF, and time series decomposition plots for three datasets.

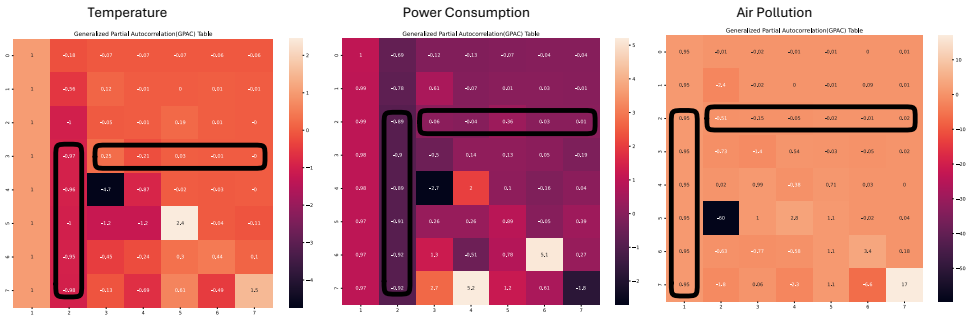

Fig. 2 GPAC for three stationary raw datasets to find potential order for ARMA process.

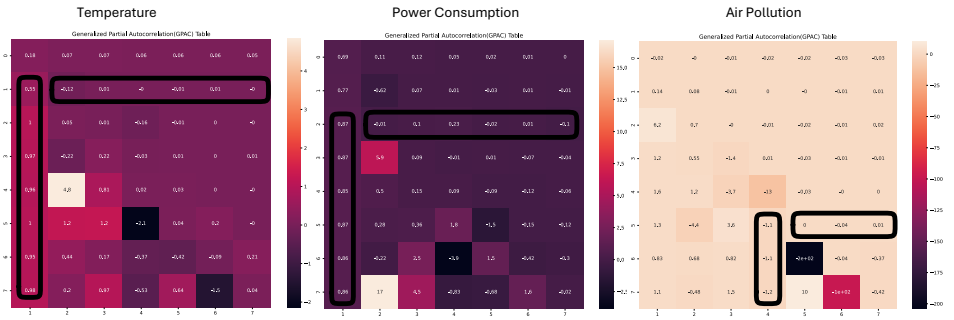

Fig. 3 GPAC for three non-seasonal first-order differencing data to find potential order for ARMA process.

In addition to traditional methods like domain expertise and manual plot interpretation, we utilized Optuna, an advanced machine learning optimization algorithm, for hyperparameter tuning. Optuna's efficient search strategies helped minimize errors and improve model performance. For the AR component, we set the maximum potential order to 20 and 10 for the MA component, as no clear MA pattern was found in the ACF and PACF plots. We conducted 20 trials for the AR model, 10 for the MA model, and 30 for the ARMA and ARIMA models.

Table 3 compares the model orders obtained through domain knowledge and Optuna and their respective MSE values. The minor differences in MSE suggest that while domain knowledge is valuable, Optuna offers a systematic and automated alternative, particularly when prior knowledge is insufficient or when determining an appropriate order is challenging. We selected the order with the lowest MSE from

	Domain Order	**MSE**	**Optuna Order**	**MSE**
AR				
Temperature	(2)	0.038986	(4)	0.038950
Power Consumption	(2)	0.059108	(10)	0.059213
Air Pollution	(1)	0.019484	(5)	0.019482
MA				
Temperature	–	–	(10)	0.038870
Power Consumption	–	–	(10)	0.059218
Air Pollution	–	–	(10)	0.019472
ARMA				
Temperature	(2,3)	0.038971	(8,2)	0.038892
Power Consumption	(2,2)	0.059206	(4,5)	0.059281
Air Pollution	(1,2)	0.019483	(2,6)	0.019458
ARIMA				
Temperature	(1,1,1)	0.049111	(7,1,6)	0.044493
Power Consumption	(1,1,2)	0.055292	(11,1,6)	0.055068
Air Pollution	(4,1,5)	0.019482	(13,1,10)	0.019479

Table 3 MSE comparison between models which orders determined by domain knowledge and Optuna. Predictions and ground truth are normalized using the equation in 1.

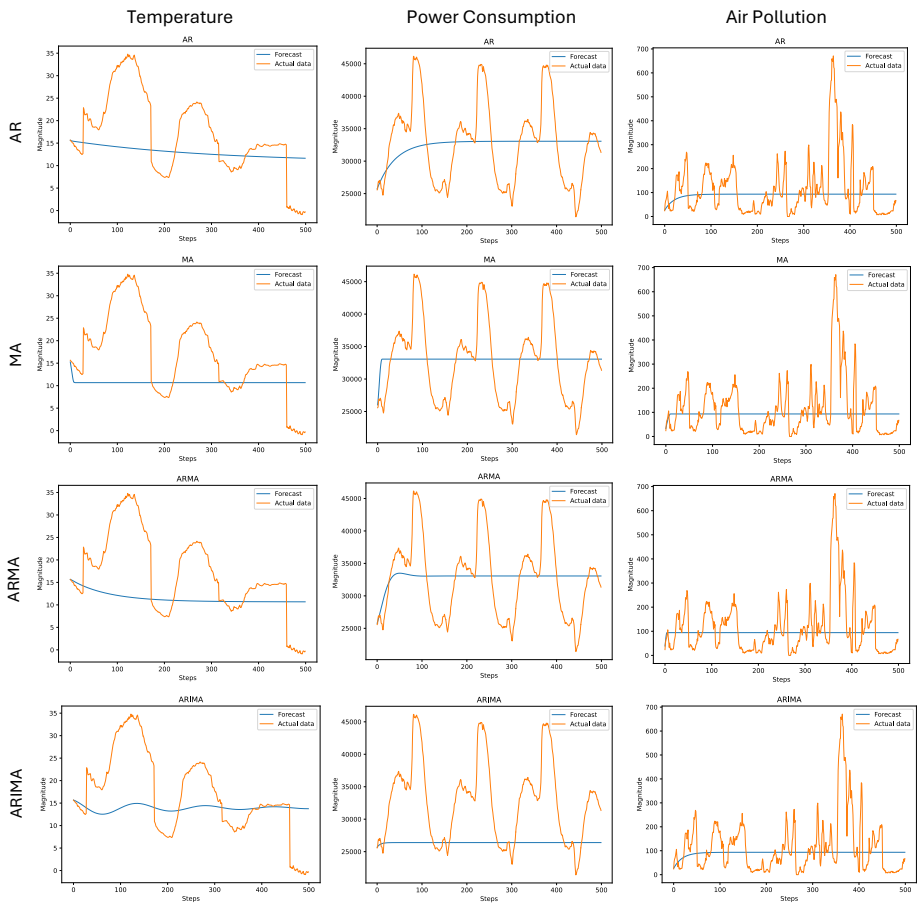

Fig. 4 Plots: comparison between predicted values vs. ground truth for four classical models (AR, MA, ARMA, ARIMA) across three datasets (First 500 samples).

those determined using domain knowledge and Optuna as the final order for each model. In cases where the GPAC showed multiple patterns, we chose the order corresponding to the lowest MSE. To evaluate the performance of the four classical models, we computed MSE, RMSE, and MAE, with results presented in Table 4. The comparison between predicted and actual values is visualized in Figure 4. The performance of the classical models was not very satisfactory, which is expected given their simplicity and reliance on basic approaches, primarily serving as baseline references.

Modern techniques results. We implemented three modern techniques, including LSTM, Bi-LSTM, and Seq2Seq models. The three modern techniques were trained with identical hyperparameters, as shown in Table 1, ensuring direct comparability across models and isolating the impact of each model's architecture. MSE, RMSE, and MAE were calculated to evaluate performance, as summarized in Table 5. Figure 5 compares the predicted and actual values for LSTM, BiLSTM, and Seq2Seq models. The modern techniques significantly outperformed the classical models, with lower

	MSE	RMSE	MAE
AR			
Temperature	0.038950	0.197359	0.161282
Power consumption	0.059108	0.243121	0.204268
Air pollution	0.019482	0.139578	0.101504
MA			
Temperature	0.038870	0.197156	0.160967
Power consumption	0.059218	0.243347	0.204480
Air pollution	0.019472	0.139542	0.101515
ARMA			
Temperature	0.038892	0.197210	0.161051
Power consumption	0.059206	0.243322	0.204456
Air pollution	0.019458	0.139493	0.101845
ARIMA			
Temperature	0.044493	0.210933	0.178464
Power consumption	0.055068	0.234666	0.193346
Air pollution	0.019479	0.139569	0.101566

Table 4 Performance metrics comparison for AR, MA, ARMA, ARIMA. Both predictions and ground truth are normalized using the Equation 1.

MSE values and superior accuracy. Among them, the LSTM model, despite its simplicity, achieved the lowest MSE, indicating its effectiveness for the datasets in this study.

	MSE	RMSE	MAE
LSTM			
Temperature	0.000127	0.011277	0.004017
Power consumption	0.000175	0.013225	0.008923
Air pollution	0.001260	0.035502	0.018459
Bi-LSTM			
Temperature	0.000131	0.011437	0.004310
Power consumption	0.000179	0.013362	0.009071
Air pollution	0.000261	0.035516	0.019128
Seq2Seq			
Temperature	0.000128	0.011307	0.004099
Power consumption	0.000226	0.015046	0.009872
Air pollution	0.000247	0.035312	0.018662

Table 5 Performance metrics comparison for LSTM, Bi-LSTM, and Seq2Seq. Both prediction and ground truth are normalized using the Equation 1.

State-of-the-Art results. We also implemented the Transformers model, a state-of-the-art architecture with a self-attention mechanism. To ensure consistency, we used the same hyperparameters as the modern techniques, as shown in Table 1. Based on the prediction vs. actual plots in Figure 6 and the metrics in Table 6, the Transformers model outperforms the classical models but does not exceed the performance

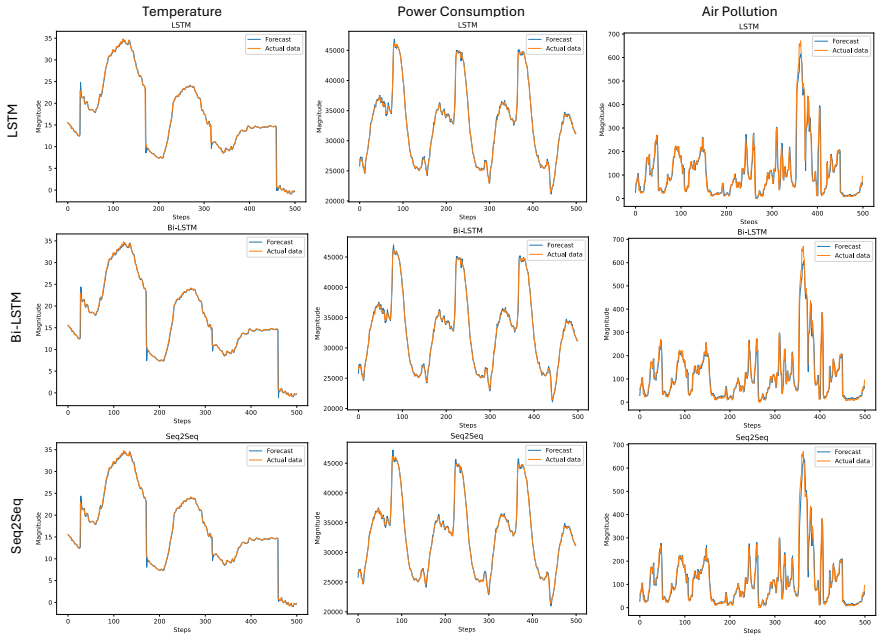

Fig. 5 Plots: comparison between predicted values vs. ground truth for three modern techniques (LSTM, Bi-LSTM, and Seq2Seq) across three datasets (First 500 samples).

of the three modern techniques. This suggests that for these datasets, the complex Transformers architecture may not be necessary for optimal performance.

	MSE	**RMSE**	**MAE**
Temperature	0.000599	0.024466	0.016530
Power consumption	0.000467	0.021602	0.015685
Air pollution	0.001370	0.037011	0.019526

Table 6 Performance metrics for Transformers model. Both prediction and ground truth are normalized using the Equation 1.

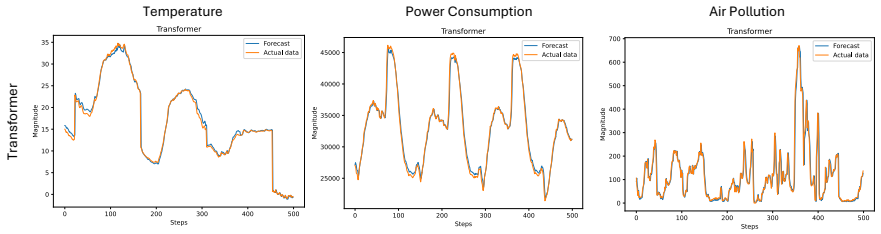

Fig. 6 Plots for predicted values vs. ground truth for Transformers across three datasets (First 500 samples).

4 Conclusion

In this study, we evaluated the performance of four classical models (AR, MA, ARMA, ARIMA), three modern techniques (LSTM, BiLSTM, Seq2Seq), and a state-of-the-art Transformers model on three publicly available datasets. Overall, our findings indicate that while we implemented relatively simple classical models, both modern techniques and the Transformers model consistently outperformed the classical approaches. Among the deep learning-based models, LSTM, with its simplest architecture, achieved the lowest MSE. This suggests that even the simplest deep learning architecture is sufficient to handle the data effectively for these datasets.

5 Limitation and Future Work

There are some limitations to this study. For classical models, we selected relatively simple configurations based on available information, which may not represent their optimal performance. It is possible to identify better-performing classical models with more extensive exploration. When using GPAC, we limited the number of rows and columns, which may have prevented us from identifying the correct patterns. Similarly, when using Optuna for hyperparameter tuning, we constrained the maximum potential order and the number of trials. Increasing these limits could lead to the discovery of more suitable models. For the modern techniques and Transformers, all models were trained with a fixed sequence length and a common set of hyperparameters, which may not be optimal for every model or dataset.

Future work could involve implementing Seasonal Autoregressive Integrated Moving Average (SARIMA) models [11] to better capture the seasonality present in the datasets. Additionally, applying advanced techniques such as the Box-Jenkins methodology [11] and exploring other model variations could provide a more comprehensive performance comparison. While the current work focuses on univariate time series analysis, future efforts could extend to multivariate time series, allowing us to explore the relationships between multiple variables over time. By addressing these limitations and exploring alternative approaches, we aim to refine our findings and broaden the scope of this research.

References

[1] Adhikari, R., Agrawal, R.K.: An introductory study on time series modeling and forecasting. arXiv preprint arXiv:1302.6613 (2013)

[2] Kim, Y.S., Rachev, S.T., Bianchi, M.L., Mitov, I., Fabozzi, F.J.: Time series analysis for financial market meltdowns. Journal of Banking & Finance **35**(8), 1879–1891 (2011)

[3] Box-Steffensmeier, J.M., Freeman, J.R., Hitt, M.P., Pevehouse, J.C.: Time series analysis for the social sciences (2014)

[4] Mudelsee, M.: Climate time series analysis. Atmospheric and **397** (2010)

[5] Chou, J.-S., Tran, D.-S.: Forecasting energy consumption time series using machine learning techniques based on usage patterns of residential householders. Energy **165**, 709–726 (2018)

[6] Gao, R., Tang, Y., Xu, K., Huo, Y., Bao, S., Antic, S.L., Epstein, E.S., Deppen, S., Paulson, A.B., Sandler, K.L., *et al.*: Time-distanced gates in long short-term memory networks. Medical image analysis **65**, 101785 (2020)

[7] Shakeel, A., Tanaka, T., Kitajo, K.: Time-series prediction of the oscillatory phase of eeg signals using the least mean square algorithm-based ar model. Applied Sciences **10**(10), 3616 (2020)

[8] Jafari, R., Jafari, A.H.: Speech recognition using arma model and levenberg-marquardt algorithm. In: Intelligent Systems Conference, pp. 351–367 (2024). Springer

[9] Chimmula, V.K.R., Zhang, L.: Time series forecasting of covid-19 transmission in canada using lstm networks. Chaos, solitons & fractals **135**, 109864 (2020)

[10] Jin, Y., Hou, L., Chen, Y.: A time series transformer based method for the rotating machinery fault diagnosis. Neurocomputing **494**, 379–395 (2022)

[11] Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time series analysis: forecasting and control (2015)

[12] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)

[13] Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural networks **18**(5-6), 602–610 (2005)

[14] Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, vol. 27, pp. 3104–3112 (2014)

[15] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems (2017)

[16] Mnassri, B.: Weather Station Beutenberg Dataset. Accessed: 2023-12-11 (2020). https://www.kaggle.com/datasets/mnassrib/jena-weather-dataset/data

[17] Salam, A., El Hibaoui, A.: Power Consumption of Tetouan City. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5B034 (2018)

[18] Roy, R.: Air Pollution Forecasting - LSTM Multivariate. Accessed: 2024-12-02 (2020). https://www.kaggle.com/datasets/rupakroy/

lstm-datasets-multivariate-univariate

[19] Medsker, L.R., Jain, L., *et al.*: Recurrent neural networks. Design and Applications **5**(64-67), 2 (2001)

[20] Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)

[21] Hyndman, R., Athanasopoulos, G.: Forecasting: principles and practice (2018)

[22] Chatfield, C., Xing, H.: The analysis of time series: an introduction with r (2019)

[23] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2623–2631 (2019)