Temperature Forecasting with LSTM: A Case Study on Kyiv Weather Data*

Oleksandr Makoveichuk[1,†], Oleksandr Golubenko[1,*,†], Stanislav Kukhtyk[1,†], Artem Antonenko[2,†], Viktoriia Bereznychenko[3,†], Andrii Iatsyshyn[4]

[1] *Yuriy Bugay International Scientific and Technical University, Volodymyrska 7, 04025 Kyiv, Ukraine*

[2] *National University of Life and Environmental Sciences of Ukraine, Heroiv oborony 15, 03041 Kyiv, Ukraine*

[3] *Institute of Electrodynamics of NAS of Ukraine, Beresteyskyi 56, 03057 Kyiv, Ukraine*

[4] *Center for Information-Analytical and Technical Support of Nuclear Power Facilities Monitoring, National Academy of Sciences of Ukraine, Akademika Palladina 34a, 03142 Kyiv, Ukraine*

Abstract

Accurate temperature forecasting is essential for urban planning, energy management, and environmental monitoring in smart cities. This study evaluates the use of long short-term memory (LSTM) networks for weekly average temperature (TAVG) prediction in Kyiv, using ARIMA as a baseline model. Historical temperature time series from 1960 onward were employed, and multiple look-back windows were tested to capture seasonal and long-term dependencies. Forecast performance was assessed using RMSE and MAE metrics, showing that LSTM provides more accurate predictions, while ARIMA effectively captures trends and seasonal components. Forecast errors were further analyzed via normal distribution fitting to compare model characteristics. The study emphasizes the importance of rigorous model comparison, including alternatives such as Prophet, and highlights opportunities for long-term analysis to investigate climate trends, global warming effects, or anomalies linked to astronomical, climatic, or anthropogenic factors. These findings demonstrate the potential of deep learning approaches to support data-driven decision-making and sustainable urban management.

Keywords

time series forecasting, LSTM, ARIMA, seasonal trends, climate data analysis

1. Introduction

Time series analysis is a statistical methodology that involves the systematic collection of observations at fixed time intervals to detect underlying patterns and long-term trends. This approach is widely employed to support informed decision-making and to construct accurate forecasts based on historical data. Time series forecasting, in turn, refers to the process of predicting future values by analyzing past trends and latent regularities. It encompasses a wide spectrum of approaches, ranging from classical statistical models to contemporary deep learning techniques. Each class of models is grounded in a distinct mathematical framework, reflecting different assumptions about the structure of temporal data.

In the broader context of smart cities, time series forecasting underpins numerous domains where continuous data collection allows proactive and adaptive urban management. Forecasting models support transportation and mobility optimization by anticipating traffic congestion and public transit demand; enable efficient energy and utility distribution through load prediction; and contribute to environmental monitoring by modeling air quality, temperature, and noise variations. In

* *Applied Information Systems and Technologies in the Digital Society (AISTDS-2025), October 01, 2025, Kyiv, Ukraine*

[1,*] Corresponding author.

[†] These authors contributed equally.

o.makoveychuk@istu.edu.ua (O. Makoveichuk); o.golubenko@istu.edu.ua (O. Golubenko); stan.kukhtyk@istu.edu.ua (S. Kukhtyk); artem.v.antonenko@gmail.com (A. Antonenko); vika.bereznichenko@i.ua (V. Bereznychenko); iatsyshyn.andriy@gmail.com (A. Iatsyshyn)

0000-0003-4425-016X (O. Makoveichuk); 0000-0002-1776-5160 (O. Golubenko); 0000-0002-2738-5866 (S. Kukhtyk); 0000-0001-9397-1209 (A. Antonenko); 0000-0002-9961-1703 (V. Bereznychenko): 0000-0001-5508-7017 (A. Iatsyshyn);

infrastructure management, predictive analytics facilitates maintenance scheduling and enhances the reliability of critical systems. Moreover, forecasting aids public safety by anticipating demand for emergency services and supports evidence-based decision-making in economic planning and healthcare management. Thus, time series forecasting serves as a cornerstone of smart city intelligence, ensuring sustainable, resilient, and responsive urban operations [1–5].

Air temperature forecasting is particularly essential for smart city applications, as it directly influences urban planning, energy management, and public safety. Predictive insights enable the optimized operation of heating, ventilation, and air conditioning systems, support the efficient allocation of energy resources, and guide real-time responses to extreme weather events. Furthermore, accurate temperature forecasts facilitate environmental monitoring, traffic management, and health-related interventions, making them a critical component of data-driven urban infrastructure [6].

2. Classical and Modern Forecasting Methods

Forecasting time series encompasses a broad spectrum of approaches, ranging from classical statistical models to modern deep learning techniques. Each class of models is grounded in a distinct mathematical framework, reflecting different assumptions about the structure and dynamics of temporal data.

The classical linear family of models begins with the *Autoregressive Integrated Moving Average (ARIMA)* model [7]. ARIMA seeks to capture three key phenomena in time series: autoregression (dependence on previous values), differencing (to remove non-stationarity), and moving averages (dependence on past forecast errors). Formally, an ARIMA(p, d, q) model is expressed as

$$\phi_p(B)(1 - B)^d y_t = \theta_q(B)\varepsilon_t, \tag{1}$$

where y_t denotes the observed value of the time series at time t, $\phi_p(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ is the autoregressive polynomial, $\theta_q(B) = 1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q$ is the moving-average polynomial, B is the backshift operator ($By_t = y_{t-1}$, i.e., $B^k y_t = y_{t-k}$), d is the order of differencing, and ε_t is white noise. A diagram illustrating the ARIMA model is shown in Figure 1.

Figure 1: Diagram illustrating the ARIMA model (reproduced from [7]).

This compact structure makes ARIMA particularly powerful for capturing linear correlations across time.

An extension of this framework, the *Seasonal ARIMA* (*SARIMA*) model [8], introduces seasonal components into the ARIMA structure. A SARIMA$(p, d, q)(P, D, Q)^S$ process is given by

$$\Phi_P(B^S)\phi_p(B)(1 - B)^d(1 - B^S)^D y_t = \Theta_Q(B^S)\theta_q(B)\varepsilon_t, \tag{2}$$

where S denotes the seasonal period (e.g., 12 for monthly data with annual seasonality). Here, $\Phi_P(B^S)$ and $\Theta_Q(B^S)$ represent seasonal autoregressive and moving-average polynomials, while $(1 - B^S)^D$ models seasonal differencing. This allows SARIMA to explicitly encode recurrent seasonal fluctuations, a central feature of many economic and meteorological datasets [9].

The advantage of SARIMA models over traditional ARIMA models lies in their inherent ability to explicitly account for seasonality. This feature is crucial when the time series exhibits recurring patterns over fixed intervals, such as monthly sales, quarterly financial data, or annual climatic cycles. While ARIMA models effectively handle various time series through their autoregressive, differencing, and moving average components, they may struggle to capture periodic seasonal patterns, potentially reducing forecast accuracy [10].

While ARIMA-type models are linear, more recent frameworks adopt nonlinear and additive formulations. A widely used example is *Prophet*, developed by Facebook (Meta) [11]. Prophet assumes that a time series can be decomposed into interpretable components — trend, seasonality, holiday effects, and noise — via an additive model:

$$y_t = g_t + s_t + h_t + \varepsilon_t, \tag{3}$$

where g_t captures long-term growth, s_t represents periodic seasonal fluctuations, h_t accounts for holiday or event-driven shocks, and ε_t is the error term.

The trend component g_t can take two primary forms. The piecewise linear model is Prophet's default, defined as

$$g_t = \{kt + m, \quad t \leq \tau \; kt + m + \delta(t - c), \quad t > \tau \,, \tag{4}$$

where k is the slope, m the offset, τ a change point, and δ the change in slope.

Alternatively, the logistic growth model describes nonlinear growth saturating at a carrying capacity C:

$$g_t = \frac{C}{1 + e^{-k(t-m)}}, \tag{5}$$

with growth rate k and offset m. This is suitable for scenarios where growth slows as a limit is approached, such as product adoption or population dynamics.

The seasonality component s_t is modeled using a truncated Fourier series:

$$s_t = \sum_{n=1}^{N} \left[a_n \, cos \, cos \, \frac{2\pi nt}{P} + b_n \, sin \, sin \, \frac{2\pi nt}{P} \right], \tag{6}$$

where P denotes the seasonal period (e.g., 365.25 for yearly effects) and N is the number of harmonics. This flexible basis expansion allows Prophet to capture complex periodic patterns beyond simple sinusoids. Weekly seasonality, by contrast, is often represented by dummy variables indicating weekdays, while holidays h_t are introduced as binary regressors marking known calendar events such as public holidays or sales campaigns.

Despite its simplicity of form, Prophet relies on Bayesian estimation to identify optimal trend change points, regularize parameters, and quantify forecast uncertainty. As such, it provides a balance between interpretability and automation, making it suitable for business and policy applications [12].

Beyond additive statistical models, the field has increasingly turned to deep learning architectures, which can approximate highly nonlinear dynamics. A *Recurrent Neural Network (RNN)* is a class of artificial neural networks designed to process sequential data by maintaining a hidden state that captures information from previous time steps. Unlike feedforward networks, which assume independence between inputs, RNNs incorporate temporal dependencies by recurrently

updating their hidden state based on both the current input and the prior hidden state. This structure makes RNNs particularly suitable for modeling time series, natural language, and other sequential data, as they can, in principle, learn long-term dependencies within sequences [13, 14].

RNN is the foundational structure, defined recursively as

$$H_t = \sigma(W_H H_{t-1} + W_x x_t + b), \tag{7}$$

$$y_t = W_y H_t + c, \tag{8}$$

where H_t is the hidden state, x_t the input at time t, σ a nonlinear activation function, b, c biases, and W_H, W_x, W_y weight matrices. A schematic representation of the RNN model is presented in Figure 2.

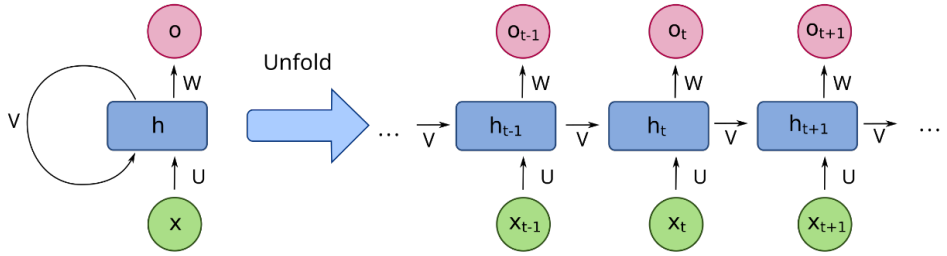

Figure 2: Diagram illustrating the RNN model information [Public domain], via Wikimedia Commons. (By fdeloche - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=60109157).

Long Short-Term Memory (LSTM) networks are an extension of Recurrent Neural Networks (RNNs) specifically designed to overcome the vanishing and exploding gradient problems that limit the ability of standard RNNs to capture long-term dependencies. LSTMs introduce a memory cell along with gating mechanisms—input, forget, and output gates—that regulate the flow of information. This architecture enables the network to selectively retain, update, or discard information across long sequences, making LSTMs particularly effective for complex time series forecasting, speech recognition, and natural language processing tasks [15, 16].

LSTM networks can be described by the following set of equations:

$$i_t = \sigma(W_i x_t + U_i H_{t-1} + b_i), \tag{9}$$

$$f_t = \sigma(W_f x_t + U_f H_{t-1} + b_f), \tag{10}$$

$$o_t = \sigma(W_o x_t + U_o H_{t-1} + b_o), \tag{11}$$

$$\tilde{c}_t = \sigma(W_c x_t + U_c H_{t-1} + b_c), \tag{12}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_{t-1}, \tag{13}$$

$$h_t = o_t \odot \ tanh\ tanh\ c_t\ , \tag{14}$$

where the symbols denote:
x_t — the input vector at time step t;
H_{t-1} — the hidden state from the previous time step $t-1$;

i_t — the input gate, controlling how much new information is added to the memory;
f_t — the forget gate, regulating which portion of the previous memory is retained;
o_t — the output gate, determining which part of the memory contributes to the output;
$\tilde{c}_t$ — the candidate memory cell, representing new information to be potentially incorporated;
c_t — the memory cell state at time step t;
H_t — the hidden state at time step t, also serving as the output and input to the next time step;
W_i, W_f, W_o, W_c — weight matrices applied to the input x_t;
U_i, U_f, U_o, U_c — weight matrices applied to the previous hidden state H_{t-1};
b_i, b_f, b_o, b_c — bias vectors;
$\sigma()$ — the sigmoid activation function, constraining values to the range [0,1];
$tanh\ tanh\ ()$ — the hyperbolic tangent function, mapping values to [-1,1];
$\odot$— element-wise (Hadamard) multiplication, applied to vectors of equal dimension, e.g.,

$$a \odot b = [a_1 b_1, a_2 b_2, a_3 b_3, \cdots, a_n b_n]. \tag{15}$$

This element-wise multiplication allows the gating vectors i_t, f_t, and o_t to selectively modulate the memory and hidden states, enabling LSTMs to retain, update, or output information in a controlled manner. A diagram illustrating the LSTM model is shown in Figure 3.

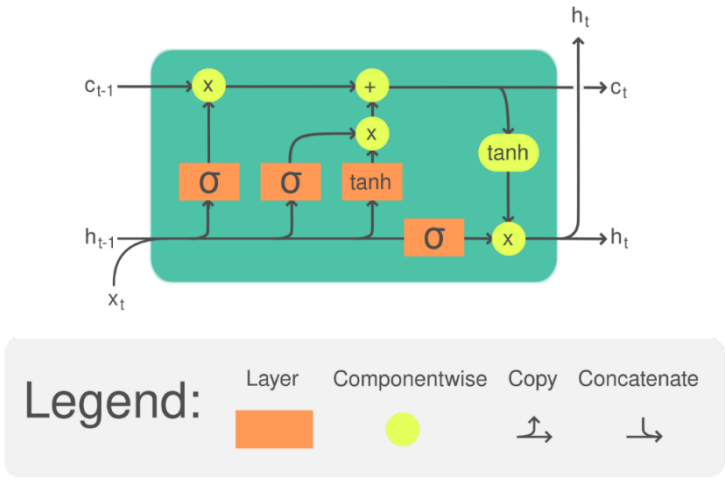

Figure 3: Diagram illustrating the LSTM cell with all gates and the flow of information [Public domain], via Wikimedia Commons (By Guillaume Chevalier - File:The_LSTM_Cell.svg, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=109362147).

Variants such as *Bidirectional LSTMs (BLSTM)* process sequences in both forward and backward directions, while *Gated Recurrent Units (GRUs)* simplify the gating mechanism yet maintain competitive performance in modeling sequential dependencies [17].

3. Experimental validation

The dataset comprises historical daily temperature observations for Kyiv, which is part of the Kaggle dataset covering the period 1881–2017 [18]. For demonstration purposes, only the subset starting from January 1, 2001, is considered. Although the original dataset includes minimum (TMIN), maximum (TMAX), and average (TAVG) temperatures, this study focuses exclusively on the weekly average temperature (TAVG). The daily data were aggregated to a weekly frequency, resulting in consecutive weekly averages spanning several decades. For model evaluation, the training period includes data prior to January 1, 2010, while all subsequent observations constitute the test period.

Forecasting experiments were conducted using multiple look-back windows, which determine the number of preceding weekly observations used as input for the model. Specifically, four look-back windows were considered: 4 weeks (approximately one month), 13 weeks (one quarter), 52 weeks (one year), and 104 weeks (two years). These windows were chosen to capture short-term, seasonal, and long-term temporal dependencies, allowing the models to learn patterns ranging from monthly variability to multi-year trends.

The dataset provides sufficient temporal coverage and variability, enabling robust evaluation of short- and medium-term temperature forecasting performance. Representative weekly TAVG profiles during the test period illustrate both seasonal fluctuations and interannual variability, demonstrating the suitability of the dataset for developing predictive models. Figure 4 shows the Kyiv temperature time series divided into training and testing subsets. This structured data serves as the input for the LSTM-based forecasting model, which leverages the sequential nature of the observations to capture temporal dependencies and generate accurate future temperature predictions.

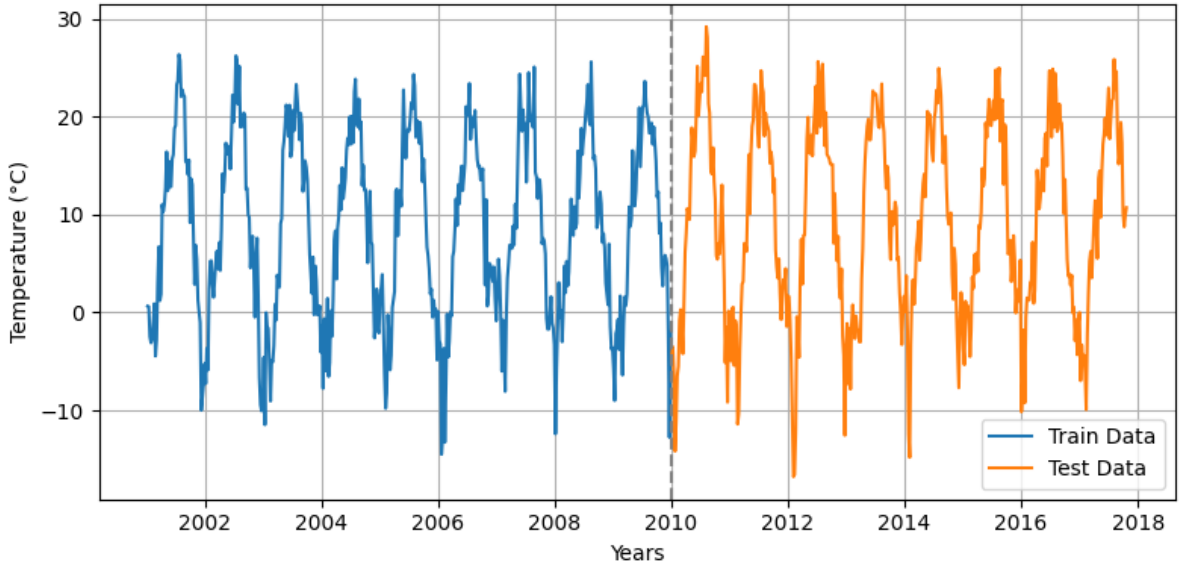

Figure 4: Train–test data split for the temperature time series. The dashed vertical line marks the boundary between the training and testing subsets.

The predictive model is based on a LSTM neural network, designed to capture temporal dependencies in TAVG series. The input layer receives sequences of length equal to the chosen look-back window, with a single feature per time step corresponding to the temperature values. This is followed by an LSTM layer comprising 64 units, which incorporates a recurrent dropout rate of 0.2 to mitigate overfitting by randomly deactivating recurrent connections during training. A subsequent Dropout layer with a rate of 0.2 further regularizes the network by reducing co-adaptation between neurons. Finally, a fully connected Dense layer with a single neuron produces the forecasted temperature for the subsequent week. The model is compiled using the mean squared error (MSE) loss function and optimized with the *Adam* algorithm. Training is performed over 50 epochs with a batch size of 16, and 20% of the training data is reserved for validation. The network learns to map past weekly temperature patterns to future values, capturing both seasonal fluctuations and long-term trends.

Training was performed over 50 epochs with a batch size of 16, using 20% of the training data for validation to monitor generalization performance. The resulting training history, including both training and validation losses, was retained for subsequent analysis and visualization, allowing evaluation of convergence and overfitting. This architecture and training protocol facilitate accurate one-step-ahead temperature forecasting by leveraging the temporal structure of the input sequences.

As a baseline, the ARIMA model was implemented to provide a comparative reference against the deep learning approach. To determine the optimal model configuration, a systematic grid search was performed over all parameter combinations (p, d, q)ranging from $(0,0,1)$to $(4,1,4)$, excluding the trivial case $(0,0,0)$. For each candidate model, forecasts were generated using the training subset, and the corresponding predictive performance was evaluated on the test set. The evaluation metric employed was the root mean squared error (RMSE), which quantifies relative deviations between predicted and observed values and facilitates consistent comparison across datasets. The ARIMA model yielding the minimum RMSE was selected as the optimal baseline configuration for subsequent analysis [19].

As shown in Figure 5, LSTM forecasts more closely match the observed TAVG values than ARIMA forecasts (RMSE = 3.74 vs. 4.15), whereas ARIMA exhibits a smoother and slightly shifted profile. Given the pronounced seasonal patterns, ARIMA could alternatively capture the underlying trend and seasonal components. Forecast errors over the test period (Figure 6) yield MAE = 2.93 for LSTM and MAE = 3.40 for ARIMA.

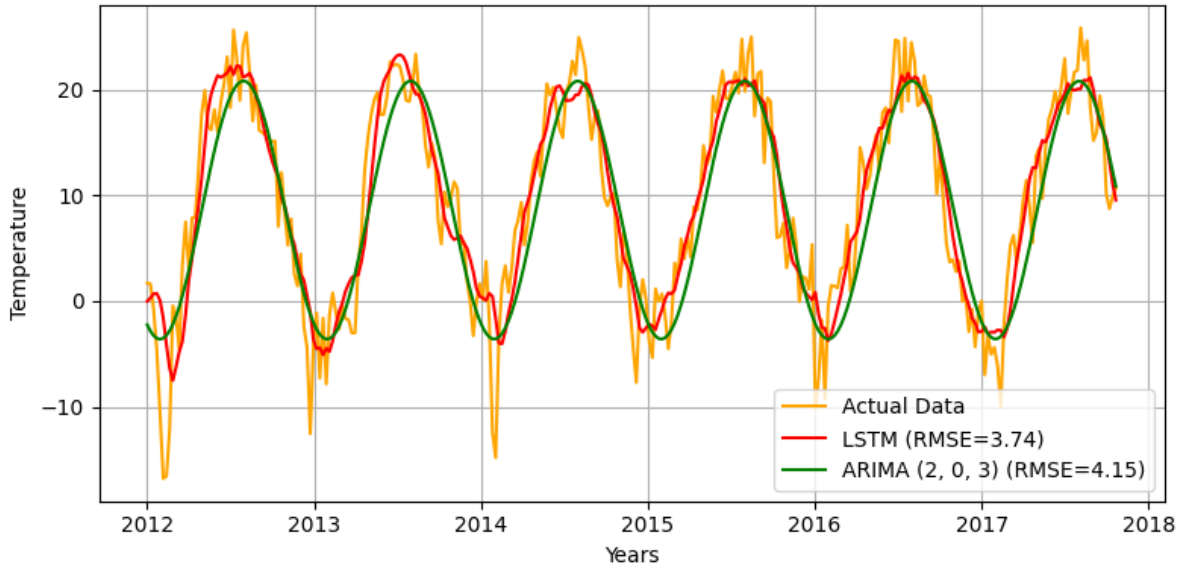

Figure 5: Comparison of LSTM and ARIMA forecasts with actual TAVG values for the test period.

Figure 6: Forecast errors for TAVG over the test period for LSTM and ARIMA models.

To further investigate the forecast performance, the distribution of forecast errors for TAVG was analyzed for both LSTM and ARIMA models. Histograms of the errors were constructed, spanning the interval from −15 °C to 15 °C, and overlaid with fitted normal distributions to approximate the underlying error patterns. For each model, the mean (μ), standard deviation (σ), and the location of the distribution peak were estimated, providing insights into the bias and dispersion of the predictions. Figure 7 presents the error distributions for LSTM and ARIMA forecasts, enabling a direct comparison of their predictive characteristics.

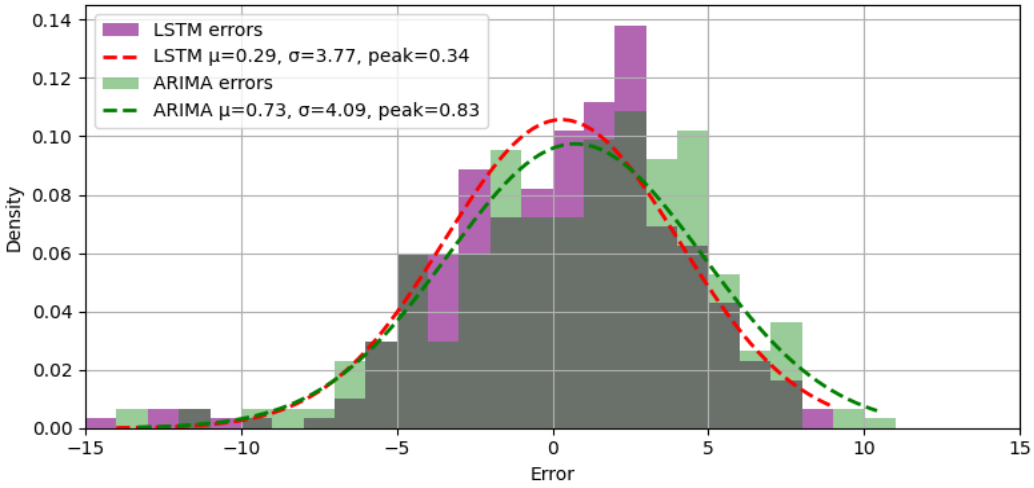

Figure 7: Histogram of forecast errors for TAVG over the test period with fitted normal distributions for LSTM and ARIMA models.

The LSTM errors are centered closer to zero (μ = 0.29) with a narrower spread (σ = 3.77) and a peak near zero (peak = 0.34), indicating minimal bias and more consistent predictions. In contrast, the ARIMA errors exhibit a larger mean deviation from zero (μ = 0.73), a wider spread (σ = 4.09), and a peak shifted further from zero (peak = 0.83), reflecting a smoother and phase-shifted forecast profile. These statistics confirm that LSTM more accurately captures short-term fluctuations and seasonal patterns in the temperature time series, providing more reliable and less biased forecasts compared to the ARIMA baseline.

4. Conclusion

This study has investigated the feasibility and performance of deep learning–based forecasting for weekly average temperatures (TAVG) in Kyiv, using LSTM networks as the primary predictive model and ARIMA as a baseline. The analysis demonstrated that the LSTM model outperforms ARIMA in terms of both RMSE and MAE, providing forecasts that more closely align with actual observations, while ARIMA offers smoother predictions that can capture underlying trends and seasonal patterns. Forecast errors were further analyzed using normal approximation, allowing characterization of the error distributions and highlighting differences between the two modeling approaches.

The results indicate that weekly temperature series exhibit pronounced seasonal behavior, and multi-week look-back windows enable the LSTM model to effectively learn both short- and medium-term dependencies. These findings highlight the potential of LSTM-based forecasting to support climate-related decision-making, urban planning, and environmental monitoring in a smart city context. To ensure robust evaluation, it is essential to compare LSTM forecasts with other models, such as ARIMA and Prophet, to contextualize performance and assess the advantages of deep learning relative to classical statistical approaches. Future work may explore hybrid modeling approaches that combine LSTM and statistical models to improve trend and seasonal component estimation, extend forecasts to longer horizons, and incorporate additional meteorological or environmental variables to enhance prediction accuracy. Overall, the study provides a robust

framework for accurate, data-driven temperature forecasting and demonstrates the practical advantages of deep learning methods over traditional time series models. Additionally, long-term analysis of historical temperature data could reveal effects of global warming or other anomalies. Such analyses could further explore correlations with astronomical factors (e.g., solar activity), climatic phenomena (e.g., El Niño and La Niña), and anthropogenic influences (e.g., urbanization, reservoir water levels, agricultural engineering), providing deeper insight into the drivers of temperature variability.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Z. Zhang, S. Ren, X. Qian, N. Duffield. Towards Invariant Time Series Forecasting in Smart Cities. arXiv preprint arXiv:2405.05430. (2024). https://arxiv.org/abs/2405.05430

[2] Q. Shao, X. Piao, X. Yao, et al. An adaptive composite time series forecasting model for short-term traffic flow. Journal of Big Data, 11, 102 (2024). https://doi.org/10.1186/s40537-024-00967-w

[3] S. Liang. Predicting short-term urban bike sharing demand in a smart city. Computers, Environment and Urban Systems, 88, 101703 (2026). https://doi.org/10.1016/j.compenvurbsys.2021.101703

[4] T. González Grandón, J. Schwenzer, T. Steens, J. Breuing. Electricity demand forecasting with hybrid classical statistical and machine learning algorithms: Case study of Ukraine. Applied Energy, 355(C), 122249 (2024). https://doi.org/10.1016/j.apenergy.2023.122249

[5] I. Didych, A. Mykytyshyn, A. Stanko, M. Mytnyk. Application of machine learning methods to the prediction of NO_2 concentration in the air environment. CEUR Workshop Proceedings, 3896, 15–18 (2023). Available at: https://ceur-ws.org/Vol-3896/short15.pdf

[6] Real-Time Temperature Prediction Models for Smart City Applications. https://www.researchgate.net/publication/388194104_Real-Time_Temperature_Prediction_Models_for_Smart_City_Applications

[7] G. E. P. Box, G. M. Jenkins. Time Series Analysis: Forecasting and Control. Prentice Hall (1970)

[8] R. J. Hyndman, G. Athanasopoulos. Forecasting: Principles and Practice. OTexts (2021)

[9] V. S. Yavuz. Forecasting monthly rainfall and temperature patterns in Van Province, Türkiye using ARIMA and SARIMA models. Journal of Water and Climate Change (2025)

[10] GeeksforGeeks. ARIMA vs. SARIMA model. Available at: https://www.geeksforgeeks.org/machine-learning/arima-vs-sarima-model/

[11] S. J. Taylor, B. Letham. Prophet: forecasting at scale. PeerJ Preprints, 6 (2018). https://doi.org/10.7287/peerj.preprints.3190v2

[12] L. Menculini, A. Marini, M. Proietti, C. Moretti. Comparing Prophet and Deep Learning to ARIMA in Forecasting Wholesale Food Prices. arXiv:2107.12770 (2021)

[13] J. L. Elman. Finding structure in time. Cognitive Science, 14(2), 179–211 (1990)

[14] I. Goodfellow, Y. Bengio, A. Courville. Deep Learning. MIT Press (2016)

[15] S. Hochreiter, J. Schmidhuber. Long short-term memory. Neural Computation, 9(8), 1735–1780 (1997)

[16] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, J. Schmidhuber. LSTM: A search space odyssey. IEEE Transactions on Neural Networks and Learning Systems, 28(10), 2222–2232 (2017)

[17] GeeksforGeeks. Bidirectional LSTM in NLP. Available at: https://www.geeksforgeeks.org/nlp/bidirectional-lstm-in-nl

[18] Kyiv, Ukraine Weather Statistics Dataset. Available at: https://www.kaggle.com/datasets/thedevastator/kyiv-ukraine-weather-statistics

[19] H. Khudov, O. Makoveychuk, I. Butko, I. Khizhnyak. A model for prediction of geospatial data in systems for processing geospatial information. Systems of Arms and Military Equipment, 2(66), 123–128 (2021). https://doi.org/10.30748/soivt.2021.66.16