Original scientific paper

UDK: 616-089.17: 621.317.38: 303.733.3
BIBLID:0350-8528(2025),25.p.01-20
doi: 10.5937/zeint0-58547

Electricity Consumption Forecasting using ARIMA and LSTM

Živko Sokolović[1], Saša Milić[1]

[1]University of Belgrade, Institute of Electrical Engineering Nikola Tesla, Koste Glavinića 8a, 11000 Belgrade, Serbia

zivko.sokolovic@ieent.org
s-milic@ieent.org

Abstract: Accurate load forecasting is essential for the reliable and efficient operation of modern power systems. This study presents a comparative analysis of two prominent forecasting models—Autoregressive Integrated Moving Average (ARIMA) and Long Short-Term Memory (LSTM)—to assess their effectiveness in predicting electricity consumption. Both models were developed and fine-tuned through hyperparameter optimization to ensure fair and optimal performance. The evaluation considered predictive accuracy, computational efficiency, and resource usage. While ARIMA demonstrated advantages in inference speed and model simplicity, the LSTM model consistently outperformed it in terms of forecasting accuracy and its ability to capture complex temporal dependencies. These findings underscore the importance of selecting appropriate models and tuning strategies for specific forecasting scenarios. The study highlights LSTM as a more suitable approach for applications that demand high accuracy and adaptability, and it provides a foundation for future research involving advanced or hybrid methods.

Keywords: ARIMA, LSTM, load forecasting, hyperparameter optimization

1 Introduction

Modern power systems require a continuous and reliable electricity supply to meet the demands of consumers. Achieving this goal necessitates accurate predictions of future electricity demand with minimal error. To address this challenge, researchers and experts have been dedicated to developing highly efficient and optimal methods for forecasting electricity consumption. This process, known as load forecasting, plays a crucial role in ensuring that power generation and distribution systems can meet fluctuating demand while maintaining system stability and efficiency. Many decisions, such as unit commitment, off-line network, dispatch and fuel allocation and several operations are controlled by load forecasting [1].

Traditional statistical models, such as the Autoregressive Integrated Moving Average (ARIMA), have been commonly used for load forecasting due to their simplicity and ease of interpretation [2, 3, 4]. However, ARIMA has certain limitations. It requires a substantial amount of data to achieve accurate predictions, which may not always be feasible in real-world business environments. Furthermore, ARIMA operates under the assumption of linearity and stationarity in the data, which can disrupt its performance when dealing with datasets that exhibit irregular or non-stationary patterns [5].

With the development of artificial intelligence (AI), load forecasting methods based on deep learning (DL) have been widely adopted in recent years. Artificial neural networks (ANNs) have been used for this purpose for decades [6, 7]. More recently, the application of recurrent neural networks (RNNs) for load forecasting has increased significantly. Among these, LSTM [8, 9] and GRU [10] networks , as variants of RNNs, effectively capture temporal characteristics while mitigating the vanishing and exploding gradient problems encountered by the traditional RNNs.

Models such as convolutional neural networks (CNNs) are widely used to improve load forecasting due to their excellent feature extraction capabilities. Combining CNNs with LSTM and GRU in hybrid models enhances forecasting accuracy by leveraging their complementary strengths [11, 12].

The remainder of this paper is organized as follows: Section 2 provides the theoretical background, covering key concepts related to ARIMA and LSTM. Section 3 outlines the methodology, including dataset preprocessing, model development, and evaluation metrics. Section 4 presents exploratory data analysis. Section 5 discusses the results, comparing the performance of ARIMA and LSTM models and analyzing their forecasting effectiveness. Section 6 concludes the paper.

2 Theoretical background

2.1 Auto-regressive integrated moving average

ARIMA is a widely used statistical method for time series forecasting [13]. This is a generalized autoregressive moving average (ARMA) model, which combines autoregressive (AR) and moving average (MA) processes, along with differencing to make the time series stationary. The general form of the ARIMA model is denoted as ARIMA (p, d, q), where *p* represents the order of the autoregressive part, *d* is the degree of differencing, and *q* is the order of the moving average part.

2.1.1 AR model (p)

The autoregressive term AR in ARIMA refers to a model where the current value of the time series Y_t depends linearly on its previous values. The AR model is characterized by the parameter *p*, which indicates the number of lagged observations included in the model. Hence, AR model of order *p*, can be written as:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \epsilon_t \tag{1}$$

where Y_t is the value of the time series at time *t*. The coefficients of the AR terms are denoted as $\phi_1, \phi_2, \ldots, \phi_p$, while *c* is the constant term (intercept), and ϵ_t is the white noise error term at time *t*.

2.1.2 Integrated process (d)

The integrated part involves differencing the original time series Y_t d times to achieve stationarity. Stationarity is a property of a time series where its statistical properties, such as mean, variance, and covariance, are constant over time. Many time series exhibit trends or seasonal patterns, which means their mean and variance change over time. Differencing helps to remove these trends and seasonality, making the time series stationary, which leads to better model fit and more accurate predictions. Many time series forecasting methods, including ARIMA, assume that the data is stationary.

If the original series is Y_t, the differenced time series $Y_t{}'$ is obtained by:

$$Y_t' = Y_t - Y_{t-1} \tag{2}$$

The formula for *d*-th order differencing in a time series Y_t is given by:

$$Y_t^{(d)} = (1 - B)^d Y_t \tag{3}$$

where *B* is the backward shift operator such that $BY_t = Y_{t-1}$

2.1.3 MA model (q)

The MA component of an ARIMA model captures the relationship between the current observation and past forecast errors. It expresses the observation as a function of past error terms. The MA model is characterized by the parameter *q*, which indicates the number of lagged forecast errors. Hence, MA model of order *q*, can be written as:

$$Y_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t \tag{4}$$

Where $\theta_1, \theta_2, \dots, \theta_q$ are the coefficients of the MA terms, μ is the constant term or the mean of the time series, and $\epsilon_{t-1}, \epsilon_{t-2}, \dots, \epsilon_{t-q}$ are the past error terms.

Incorporating all three parts, we will express ARIMA (p, d, q) model as follows:

$$Y_t' = c + \phi_1 Y_{t-1}' + \phi_2 Y_{t-2}' + \cdots + \phi_p Y_{t-p}' + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t \quad (5)$$

2.2 Long short-term memory

The LSTM is a type of recurrent neural network (RNN) appropriate for modelling temporal dependencies in sequential data. It was designed to overcome the vanishing gradient problem that limits the performance of traditional RNNs. The LSTM architecture is composed of recurrently connected sub-networks, known as memory blocks. These memory blocks are designed to maintain their state over time and regulate the flow of information using non-linear gating units [14].

The structure of the LSTM is presented in Fig. 1, where $\boldsymbol{x}^{(t)}$, $\boldsymbol{C}^{(t-1)}$, $\boldsymbol{h}^{(t-1)}$ represent the inputs to LSTM cell, while $\boldsymbol{C}^{(t)}$, $\boldsymbol{h}^{(t)}$ are the outputs. Symbol $\odot$ refers to the element-wise product and $\oplus$ means element-wise summation. Four boxes are indicated with an activation function, either the sigmoid function (σ) or tanh, and a set of weights. These boxes apply a linear combination by performing matrix-vector multiplications on their inputs. These units of computation with sigmoid activation functions, whose output units are passed through $\odot$, are referred to as gates [15].

In an LSTM cell, there are three different types of gates, which are known as the forget gate ($\boldsymbol{f}_t$), the input gate ($\boldsymbol{i}_t$), and the output gate ($\boldsymbol{o}_t$). The computations within LSTM can be clarified using the following equations:

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hf}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f\right) \quad (6)$$

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{xi}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hi}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i\right) \quad (7)$$

$$\widetilde{\boldsymbol{C}}_t = \tanh\left(\boldsymbol{W}_{xc}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hc}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_c\right) \quad (8)$$

$$\boldsymbol{C}^{(t)} = \left(\boldsymbol{C}^{(t-1)} \odot \boldsymbol{f}_t\right) \oplus \left(\boldsymbol{i}_t \odot \widetilde{\boldsymbol{C}}_t\right) \quad (9)$$

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{xo}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{ho}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o\right) \quad (10)$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{C}^{(t)}\right) \quad (11)$$

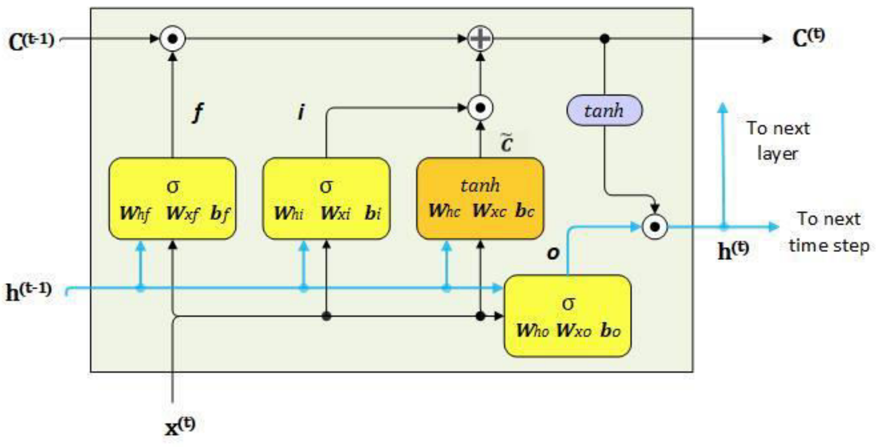

Fig. 1 The structure of the LSTM

3 Methodology

3.1 Dataset

The electricity demand data are provided by the Mixed Holding Power Utility of Republic of Srpska and are not publicly available. Dataset consists of electricity consumption data with a frequency of 15 minutes, spanning a period of 14 months, from September 2022 to November 2023. For the purposes of this analysis, the data was resampled to an hourly frequency, resulting in a total of 10,200 data points. This resampling was performed to facilitate analysis and to enhance forecasting accuracy for electricity usage patterns.

The data for this study were divided using an 80:20 split ratio, separating them into training and testing sets, as depicted in Fig. 2.

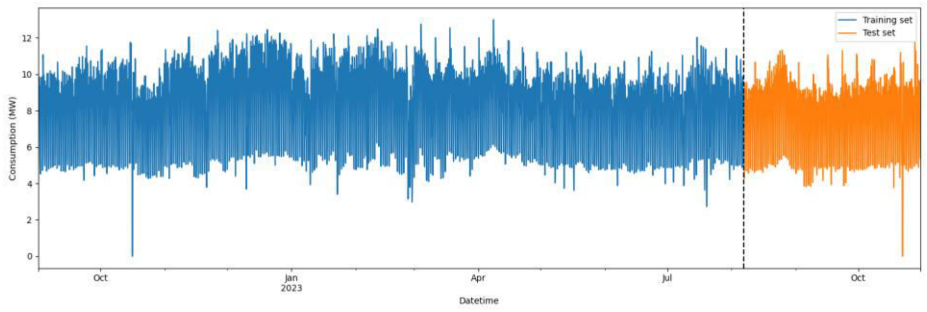

Fig. 2 Electricity consumption data

Under this division, roughly 11 months of data are allocated for model training, while the remaining three months reserved for model testing. Table 1 provides a brief description of the input data.

Table 1 Data statistics

Count	Min (MW)	Max (MW)	Mean (MW)	Std
10200	1.07	13	8.08	2

3.2 Data cleaning and preprocessing

By examining the consumption data, it was found that 22 missing values were present. To address this issue, the backward-fill technique was employed. This technique fills each missing value with the most recent non-missing value preceding it in the data. Subsequently, outliers in the dataset were identified, and visualized using a boxplot, as shown in Fig. 3. The boxplot clearly illustrates the presence of these outliers, marked as circles. To address the outliers, they were imputed using the lower bound method. In this approach, outlier values that deviated significantly from the expected range were replaced with the lower bound of the normal data distribution.

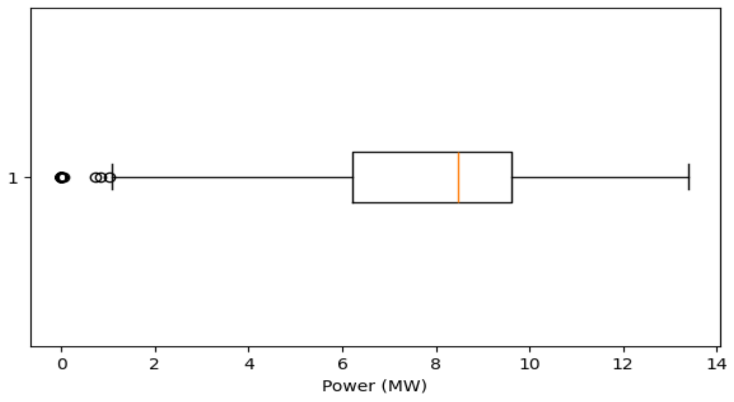

Fig. 3 Boxplot

3.3 Model implementation

A flowchart of the methodology is shown in Fig. 4. Firstly, dataset was preprocessed; all missing values and outliers were imputed (as described in Section 3.2). Next, the data was resampled to an hourly frequency to facilitate analysis (as mentioned in Section 3.1). After preprocessing, the dataset was split into training (80%) and testing (20%) sets. The ARIMA and LSTM model were then trained and their performance was evaluated using RMSE, MAE, and MAPE metrics. Finally, a comparison was conducted to assess the strengths and weaknesses of each model.

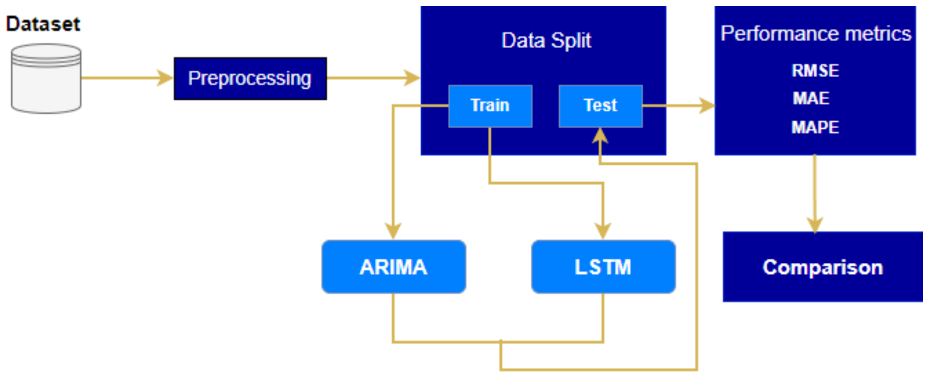

Fig. 4 Methodology flowchart

3.4 Performance metrics

It is often useful to use multiple metrics to evaluate the model's performance. In this study, three key metrics are used: root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE). The expressions for calculating these metrics are defined by the following equations:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{12}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{13}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{14}$$

where y_i, $\hat{y}_i$, and n are the measured value, the predicted value and the total number of points, respectively.

4 Exploratory data analysis

4.1 Time series decomposition

Time series data can exhibit a variety of patterns, and it is often helpful to split a time series into several components, each representing an underlying pattern category [16]. The additive decomposition model assumes that the time series can be expressed as the sum of three components:

$$Y_t = T_t + S_t + R_t \tag{15}$$

Where T_t represents the trend component, capturing long-term movement in the series. S_t denotes the seasonal component, which accounts for recurring patterns at fixed intervals. The remaining part of the series after removing trend and seasonal components denoted as R_t, represents random fluctuations or irregularities. These components were extracted using the *statsmodels.tsa.seasonal* package in Python and visualized in Fig. 5.

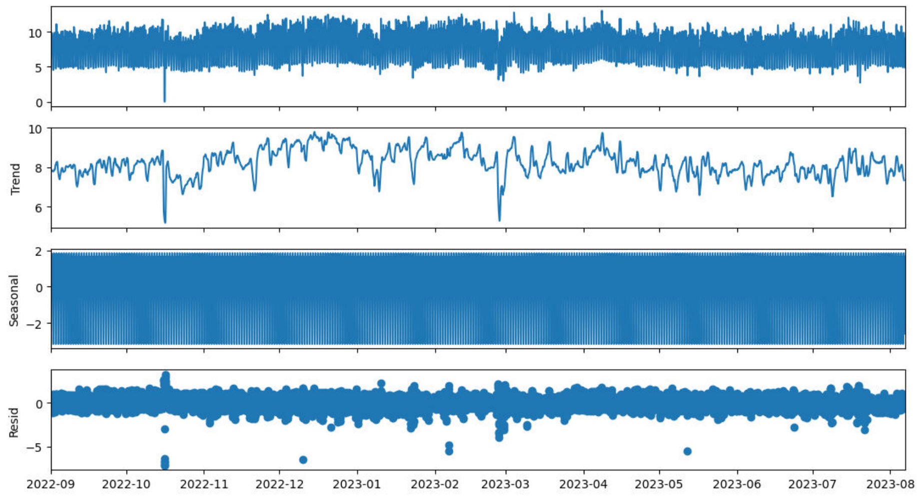

Fig. 5 Time series components: trend, seasonal, residual

The trend part closely resembles the observed time series because it captures the long-term movement and underlying direction of the data, smoothing out short-term fluctuations and seasonal variations. It is somewhat wavy but generally stable, indicating that while there are variations, there is no clear upward or downward long-term trend in the data over the observed period. The residuals are the errors in our model's estimates, representing the difference between the actual values and the predicted values. In the residuals plot, some values vary greatly from zero, indicating significant deviations between the actual and predicted values. These deviations, referred to as shocks, may be attributed to abnormal states in the power system.

The seasonal component is shown more clearly in Fig. 6. The plot displays repeating patterns, highlighting the daily seasonality in electricity consumption data.

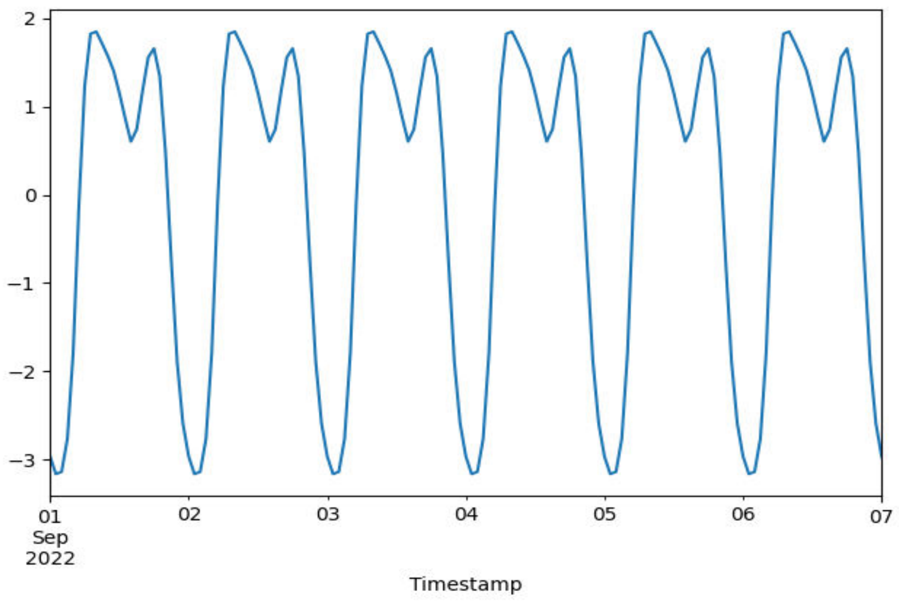

Fig. 6 Seasonal component

4.2 Stationarity analysis

As stated in Section 2.1.2, many time series forecasting methods, including ARIMA, assume that the data is stationary. To assess the stationarity of our time series data, we applied the Augmented Dickey-Fuller (ADF) test [17], which is a widely used statistical test for stationarity.

The ADF test examines the null hypothesis that a unit root is present in a time series, indicating that the series is non-stationary. The alternative hypothesis is that the series is stationary. The test returns a test statistic, a p-value, and critical values at different significance levels (1%, 5%, and 10%). If the p-value is below a predefined significance level (typically 0.05), we reject the null hypothesis, concluding that the time series is stationary.

The ADF test was applied to our data, and the results are summarized in Table 2. Since the p-value is less than the significance level of 0.05, we reject the null hypothesis. This indicates that our time series is stationary.

Table 2 ADF Test

Test Statistic	p-value	1% Critical Value	5% Critical Value	10% Critical Value
-9.191	2.11e-15	-3.43	-2.861	-2.566

4.3 Autocorrelation analysis

The autocorrelation function (ACF) and partial autocorrelation function (PACF) plots provide critical insights into the underlying structure of the time series. ACF measures the correlation between a time series and its lagged values. ACF helps in identifying which lags of the time series have significant correlations. This is useful for determining the MA terms (q) in ARIMA model. On the other hand, PACF helps in determining the direct relationship between an observation and its lagged observations, removing the influence of the intermediate lags. PACF suggest the appropriate number of AR terms (*p)* to include in ARIMA model [18].

The ACF and PACF plots are shown in Fig. 7. The ACF plot shows a sinusoidal pattern with high positive autocorrelation at lag 1. This cyclical pattern repeats approximately every 24 lags, indicating a strong seasonal component in the data. The PACF plot shows a strong spike at lag 1, indicating that the time series exhibits an autoregressive structure. Additional smaller spikes at various lags suggest the presence of multiple autoregressive terms in the data. The shaded blue area around the autocorrelation values represents the confidence intervals. Spikes outside this region are statistically significant.

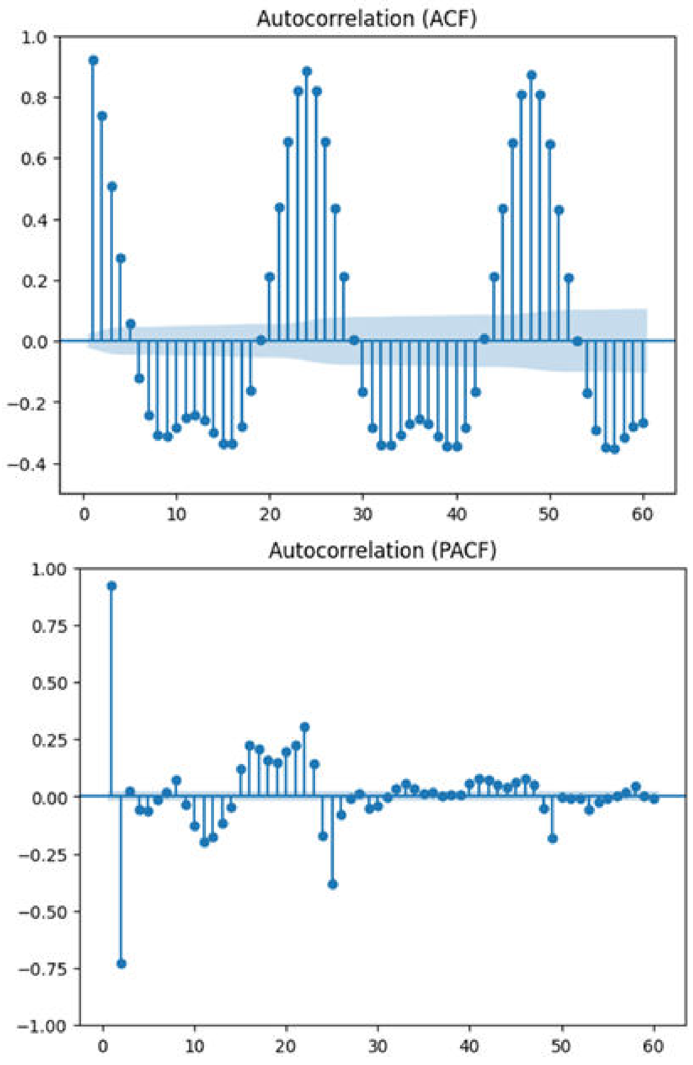

Fig. 7 ACF and PACF

5 Results and discussion

In this section, the results of both models are presented and discussed. The complete *Python* code responsible for the results is available upon request.

5.1 ARIMA results

To determine the optimal ARIMA model parameters, a custom *auto_arima* function was developed in python. This function selects the best non-seasonal (*p, d, q*) and seasonal (*P, D, Q*)[*S*] parameters for the model. The starting and ending values for these parameters were determined based on the autocorrelation analysis (Section 4.3). The Akaike information criterion (AIC) was used as the model selection criterion to determine the best-fitting ARIMA model. As shown in Table 3, the ARIMA (1, 0, 5)(2, 0, 3)[24] model achieved the lowest AIC value, indicating its suitability for the training data.

Fig. 8 visualizes the ARIMA predictions compared to the actual consumption data. Shaded regions around the forecasted values indicate the confidence intervals, which represent the uncertainty around the forecasts. These intervals show the range within which future values are expected to fall with a certain probability (e.g., 95%).

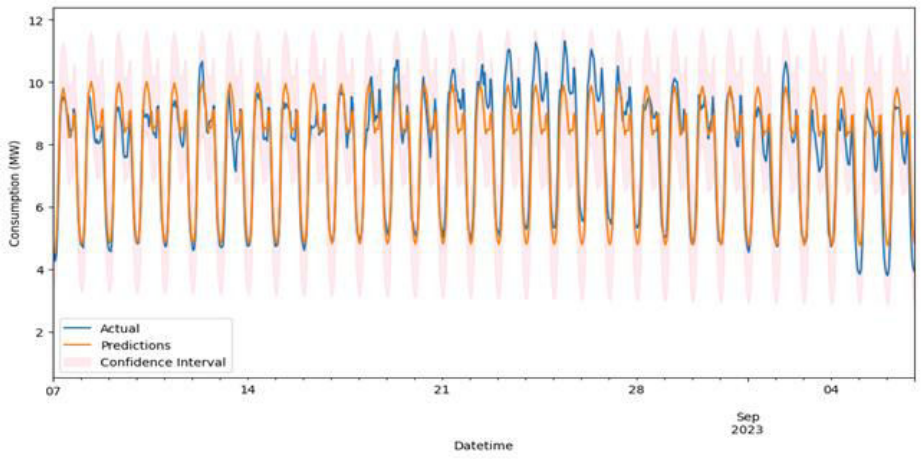

Fig. 8 ARIMA predictions

Table 3 ARIMA model selection

Model	AIC	LogL	Model	AIC	LogL
(1, 0, 1)(1, 0, 1)[24]	6283.455	-3136.728	**(1, 0, 5)(2, 0, 3)[24]**	**6217.002**	**-3096.501**
(1, 0, 1)(1, 0, 2)[24]	6269.441	-3128.721	(2, 0, 1)(1, 0, 1)[24]	6284.525	-3136.263
(1, 0, 1)(1, 0, 3)[24]	6245.567	-3115.784	(2, 0, 1)(1, 0, 2)[24]	6247.056	-3116.528
(1, 0, 1)(2, 0, 1)[24]	6247.249	-3117.624	(2, 0, 1)(2, 0, 1)[24]	6247.95	-3116.975
(1, 0, 1)(2, 0, 2)[24]	6250.796	-3118.398	(2, 0, 1)(2, 0, 2)[24]	6254.432	-3119.216
(1, 0, 1)(2, 0, 3)[24]	6242.053	-3113.026	(2, 0, 2)(1, 0, 1)[24]	6253.188	-3119.594
(1, 0, 2)(1, 0, 1)[24]	6283.982	-3135.991	(2, 0, 2)(1, 0, 2)[24]	6230.014	-3107.007
(1, 0, 2)(1, 0, 2)[24]	6245.399	-3115.699	(2, 0, 2)(1, 0, 3)[24]	6228.015	-3105.008
(1, 0, 2)(1, 0, 3)[24]	6245.355	-3114.678	(2, 0, 2)(2, 0, 1)[24]	6227.894	-3105.947
(1, 0, 2)(2, 0, 1)[24]	6247.207	-3116.604	(2, 0, 2)(2, 0, 2)[24]	6229.132	-3105.566
(1, 0, 2)(2, 0, 2)[24]	6251.174	-3117.587	(2, 0, 2)(2, 0, 3)[24]	6225.291	-3102.646
(1, 0, 2)(2, 0, 3)[24]	6242.188	-3112.094	(2, 0, 3)(1, 0, 1)[24]	6286.971	-3135.485
(1, 0, 3)(1, 0, 1)[24]	6284.372	-3135.186	(2, 0, 3)(1, 0, 2)[24]	6248.757	-3115.379
(1, 0, 3)(1, 0, 2)[24]	6246.544	-3115.272	(2, 0, 3)(1, 0, 3)[24]	6248.849	-3114.425
(1, 0, 3)(1, 0, 3)[24]	6246.462	-3114.231	(2, 0, 3)(2, 0, 1)[24]	6250.389	-3116.195
(1, 0, 3)(2, 0, 1)[24]	6248.592	-3116.296	(2, 0, 3)(2, 0, 2)[24]	6254.502	-3117.251
(1, 0, 3)(2, 0, 2)[24]	6251.882	-3116.941	(2, 0, 3)(2, 0, 3)[24]	6244.841	-3111.421
(1, 0, 3)(2, 0, 3)[24]	6243.228	-3111.614	(2, 0, 4)(1, 0, 1)[24]	6255.041	-3118.52
(1, 0, 4)(1, 0, 1)[24]	6276.734	-3130.367	(2, 0, 4)(1, 0, 2)[24]	6224.837	-3102.418
(1, 0, 4)(1, 0, 2)[24]	6243.957	-3112.979	(2, 0, 4)(1, 0, 3)[24]	6223.952	-3100.976
(1, 0, 4)(1, 0, 3)[24]	6240.61	-3110.305	(2, 0, 4)(2, 0, 1)[24]	6229.613	-3104.806
(1, 0, 4)(2, 0, 1)[24]	6242.697	-3112.348	(2, 0, 4)(2, 0, 2)[24]	6224.829	-3101.414
(1, 0, 4)(2, 0, 2)[24]	6245.189	-3112.594	(2, 0, 4)(2, 0, 3)[24]	6222.287	-3099.143
(1, 0, 4)(2, 0, 3)[24]	6237.647	-3107.823	(2, 0, 5)(1, 0, 1)[24]	6254.948	-3117.474
(1, 0, 5)(1, 0, 1)[24]	6252.181	-3117.09	(2, 0, 5)(1, 0, 2)[24]	6224.832	-3101.416
(1, 0, 5)(1, 0, 2)[24]	6220.459	-3100.229	(2, 0, 5)(1, 0, 3)[24]	6224.063	-3100.032
(1, 0, 5)(1, 0, 3)[24]	6220.203	-3099.102	(2, 0, 5)(2, 0, 1)[24]	6233.771	-3105.886
(1, 0, 5)(2, 0, 1)[24]	6236.507	-3108.253	(2, 0, 5)(2, 0, 2)[24]	6225.913	-3100.957
(1, 0, 5)(2, 0, 2)[24]	6222.798	-3100.399	(2, 0, 5)(2, 0, 3)[24]	6222.978	-3098.489

5.2 LSTM results

To obtain the best LSTM model, Keras Tuner [19] with random search strategy was employed to explore various hyperparameter configurations. The key hyperparameters used in the tuning process are presented in Table 4. After the tuning process is completed, the best combination of hyperparameters that minimizes the validation loss is shown in Table 5.

Parameters set before training are listed in Table 6, while the model training process is illustrated in Fig. 9. Although the maximum number of epochs is set to 100, the training process stops after 57 epochs due to the early stopping criterion, which was employed to prevent overfitting.

Fig. 10 visualizes the LSTM predictions compared to the actual consumption data.

Table 4 Hyperparameter exploration

Hyperparameter	Values explored
Number of LSTM layers	1, 2, 3
Units per layer	32, 64, ..., 256
Dropout rate	0.0, 0.1, ..., 0.5
Dense layer activation function	ReLU, sigmoid
Learning rate	0.01, 0.001, 0.0001

Table 5 Optimal model configuration

Hyperparameter	Value
Number of LSTM layers	3
LSTM layer 1 units	256
LSTM layer 2 units	192
LSTM layer 3 units	128
Dropout rate	0.1
Dense layer activation function	sigmoid
Learning rate	0.001

Table 6 Training parameters

Parameter	Value
Loss function	MSE
Optimizer	Adam
Maximum Epochs	100
Batch size	32
Validation split	10%

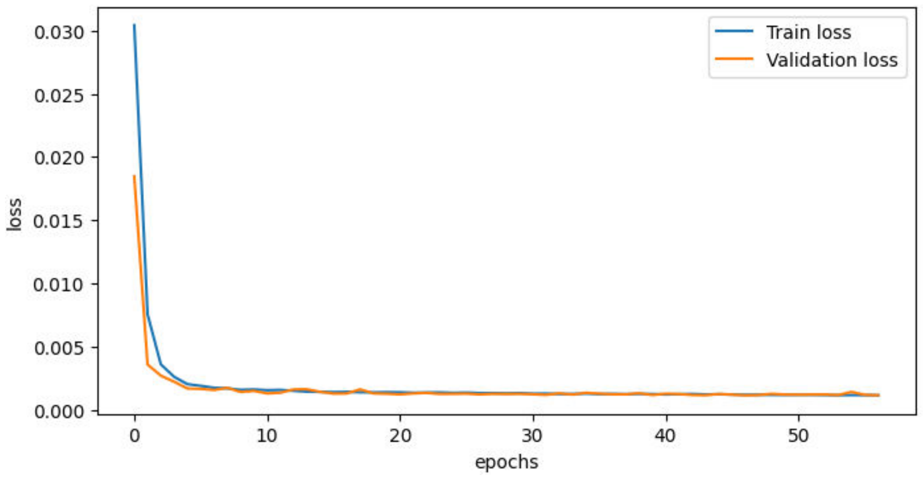

Fig. 9 Loss progression

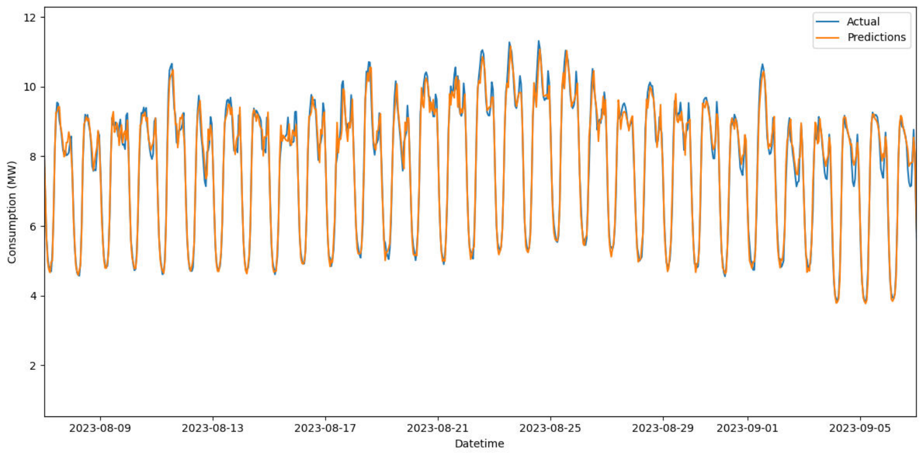

Fig. 10 LSTM predictions

5.3 Performance comparison

Observing Fig. 8 and Fig. 10, it is noticeable that LSTM model provides better predictions compared to the ARIMA model. To validate this observation and confirm the model's effectiveness, the performance of both models were evaluated using key metrics. The evaluation results are shown in Table 7. In addition to predictive accuracy, the evaluation also includes the analysis of training time, testing time, and memory usage during training, providing a comprehensive assessment of both models. The test environment is detailed in Table 8.

In terms of training time, the LSTM model exhibits a significant advantage, requiring only 53 seconds compared to the 90 seconds needed by the ARIMA model. This suggests that the LSTM model is more efficient in terms of training speed, possibly due to its capacity to learn complex patterns more quickly. When considering test time, the ARIMA model processes data at a rate of 10,850 samples per second, far surpassing the LSTM's rate of 3,180 samples per second. This indicates that, during inference, ARIMA operates at a much faster rate, potentially making it more suitable for real-time applications where speed is critical. Memory usage also reveals a stark contrast.

The ARIMA model consumes 4182.96 MB, significantly higher than the 24.62 MB required by the LSTM model. This highlights the relative memory efficiency of the LSTM, which is especially advantageous when scaling up to larger datasets or deploying the model in resource-constrained environments. In terms of forecasting accuracy, the LSTM model outperforms the ARIMA model across all key metrics. The RMSE of the LSTM (0.36) is substantially lower than the ARIMA model's RMSE (0.88), indicating more accurate predictions. Similarly, the MAE for LSTM is 0.25, compared to 0.61 for ARIMA, further emphasizing its superior predictive ability. The MAPE is also notably lower for LSTM (3.64%) compared to ARIMA (9.81%), confirming that the LSTM model offers more reliable and precise load forecasting.

Table 7 Evaluation results

Model	Train time (s)	Test time (samples/s)	Memory usage (MB)	RMSE	MAE	MAPE (%)
ARIMA	90	10850	4182.96	0.88	0.61	9.81
LSTM	53	3180	24.62	0.36	0.25	3.64

Table 8 Test environment

Item	Parameter
System version	Windows 11 x64
Processor model (CPU)	AMD Ryzen 7 5800X 8-Core
Processing speed	3.80 GHz
Memory (RAM)	32 GB
GPU	NVIDIA GeForce GT 1030

6 Conclusion

This paper presented a comparative analysis of ARIMA and LSTM models for load forecasting, evaluating their performance in terms of predictive accuracy, computational efficiency, and practical applicability. The comparison between the LSTM and ARIMA models shows that LSTM delivers better prediction performance. This conclusion is supported by evaluation metrics that assess not only forecasting accuracy but also training time, inference speed, and memory usage. While LSTM trains faster and uses significantly less memory, ARIMA processes data faster during inference. However, LSTM consistently achieves superior forecasting accuracy across all evaluated metrics, making it a more reliable choice for precise predictions, especially in data-intensive or resource-limited scenarios.

In practical applications within the power and energy sector, accurate load forecasting is essential for effective grid management, energy trading, and balancing supply with demand. LSTM models, due to their ability to capture nonlinear temporal dependencies, are particularly well-suited for modelling complex consumption patterns influenced by various external factors such as weather, holidays, or socio-economic events.

Future research will explore the integration of other advanced methods such as Gated Recurrent Units (GRU), Prophet, and XGBoost to further enhance forecasting performance. Additionally, the development of hybrid models may offer a practical compromise between speed and accuracy, adaptable to various load forecasting requirements. A particular focus will also be placed on the experimental and practical implementation of transformer-based models, adapted for time series analysis and forecasting within the electric power sector.

7 Acknowledgment

This work was supported in part by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia under the Contract on the realization and financing of the scientific research work of Reseach and Innovation Organizations in 2025.

References

[1] N. Ahmad, Y. Ghadi, M. Adnan and M. Ali, "Load Forecasting Techniques for Power System: Research Challenges and Survey," *IEEE Access,* 2022.

[2] S. Ozturk and F. Ozturk, "Forecasting Energy Consumption of Turkey by Arima Model," *Journal of Asian Scientific Research*, 2018.

[3] A. Pierre, S. Akim, A. Semenyo and B. Babiga, "Peak Electrical Energy Consumption Prediction by ARIMA, LSTM, GRU, ARIMA-LSTM and ARIMA-GRU Approaches," Energies, 2023.

[4] S. Shariff, "Autoregressive Integrated Moving Average and Long Short-Term Memory Network Models for Forecasting Energy Consumptions," *European Journal of Electrical Engineering and Computer Science*, 2022.

[5] D. Taslim and I. Murwantara, "Comparative analysis of ARIMA and LSTM for predicting fluctuating time series data," *Bulletin of Electrical Engineering and Informatics*, 2024.

[6] L. Zec, J. Mikulović and M. Žarković, "Application of artificial neural network to power consumption forecasting for the Sarajevo region," *Electrical Engineering*, 2024.

[7] K. A. Fakhryza, E. N. Budisusila and A. A. Nugroho, "Application of artificial neural network for peak load forecasting in 150 kV Semarang power system," *Journal of Electrical Systems and Information Technology*, vol. 11, no. 1, 2024.

[8] J. S. Caicedo-Vivas and W. Alfonso-Morales, "Short-Term Load Forecasting Using an LSTM Neural Network for a Grid Operator," *Energies,* vol. 16, no. 23, 2023.

[9] A. A. d. M. Meneses, "Comparing Long Short-Term Memory (LSTM) and bidirectional LSTM deep neural networks for power consumption prediction," *Energy Reports*, vol. 10, 2023.

[10] J. Zheng, X. Chen, K. Yu, L. Gan, Y. Wang and K. Wang, "Short-term Power Load Forecasting of Residential Community Based on GRU Neural Network," in *2018 International Conference on Power System Technology (POWERCON)*, 2018.

[11] A. Agga, A. Abbou, M. Labbadi and Y. e. Houm, "Short-Term Load Forecasting: Based on Hybrid CNN-LSTM Neural Network," in *2021 6th International Conference on Power and Renewable Energy (ICPRE),* 2021.

[12] S. M. Hasanat, K. Ullah, H. Yousaf and K. Munir, "Enhancing Short-Term Load Forecasting with a CNN-GRU Hybrid Model: A Comparative Analysis," *IEEE Access*, 2024.

[13] G. E. P. Box and G. M. Jenkins, Time Series Analysis: Forecasting and Control. Holden-Day., San Francisco: Holden-Day, 1970.

[14] C. B. Vennerod, A. Kjærran and E. S. Bugge, "Long Short-term Memory RNN," 2021.

[15] S. Raschka and V. Mirjalili, Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2, Packt, 2019.

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

[17] D. Dickey and W. Fuller, "Distribution of the estimators for autoregressive time series with a unit root," Journal of the American Statistical Association, 1979.

[18] K. Andras, "Understanding Autocorrelation and Partial Autocorrelation Functions (ACF and PACF)," 2024. [Online]. Available: https://medium.com/@kis.andras.nandor/understanding-autocorrelation-and-partial-autocorrelation-functions-acf-and-pacf-2998e7e1bcb5 [Accessed 19 March 2025].

[19] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi and A. Liu, "Keras Tuner," 2019. [Online]. Available: https://github.com/keras-team/keras-tuner

Kratak sadržaj: Precizno prognoziranje potrošnje električne energije ključno je za pouzdan i efikasan rad savremenih elektroenergetskih sistema. Ova studija prikazuje uporednu analizu dva istaknuta modela za prognozu — Autoregressive Integrated Moving Average (ARIMA) i Long Short-Term Memory (LSTM) — sa ciljem procene njihove efikasnosti u predviđanju potrošnje električne energije. Oba modela su razvijena i optimizovana radi pravičnog poređenja i najboljih mogućih performansi. Procena modela je obuhvatila tačnost predviđanja, računarsku efikasnost i upotrebu memorijskih resursa. Iako je ARIMA pokazala prednosti u brzini predviđanja i jednostavnosti modela, LSTM je dosledno davao preciznije prognoze i bolje prepoznavao složene vremenske obrasce. Rezultati ukazuju na značaj pažljivog izbora modela i strategije podešavanja za konkretne scenarije prognoziranja. Studija ističe LSTM kao pogodniji pristup za primene koje zahtevaju visoku tačnost i prilagodljivost, te pruža osnovu za buduća istraživanja naprednih ili hibridnih metoda.

Ključne reči: ARIMA, LSTM, prognoza opterecenja, optimizacija hiperparametara

Prognoza potrošnje električne energije korišćenjem ARIMA i LSTM modela

Živko Sokolović[1], Saša Milić[1]

Submission date: 2025-04-29.
Acceptance date: 2025-05-12