Research Article GU J Sci 39(1): 499-522 (2026) DOI: 10.35378/gujs.1709484

Gazi University

Journal of Science

http://dergipark.org.tr/gujs

Forecasting Performance Comparison: LSTM Model vs. Dynamic and Static Forecasting of ARIMA and Random Walk Models - Evidence from S&P500

Levent GUL[1, *], Ebru YUKSEL HALILOGLU[2], N. Hande SEVGI[3], Nukhet DOGAN[1], M. Hakan BERUMENT[4]

[1]*Ankara Haci Bayram Veli University, Department of Econometrics, 06500, Ankara, Türkiye*
[2]*Gazi University, Department of Industrial Engineering, 06570, Ankara, Türkiye*
[3]*Ufuk University, Department of International Trade and Finance, 06805, Ankara, Türkiye*
[4]*Bilkent University, Department of Economics, 06800, Ankara, Türkiye*

Highlights

- The forecasting performance of econometric models and machine learning models was compared.
- Distinction between static and dynamic forecasting in ARIMA and RW.
- Evaluating model stability via three cross-validation methods.

Article Info

Received: 30 May 2025
Accepted: 30 Dec 2025

Keywords

Autoregressive Integrated Moving Average, Cross-validation, Long Short-Term Memory, Machine Learning, Random Walk

Abstract

This paper re-evaluates the forecasting performance of Long Short-Term Memory (LSTM) models against traditional Autoregressive Integrated Moving Average (ARIMA) and Random Walk (RW) models using S&P 500 index data. The LSTM models in the literature, which utilize observed lagged values for their forecasting (static forecasting), are compared to traditional models that use recursive, self-generated predictions for forecasting (dynamic forecasting). To provide comparable statistics, we gather relevant statistics for the static forecasting of ARIMA and RW models. We repeat the exercise for three different cross-validation schemes. The empirical evidence presented here suggests that the static forecasting power of static ARIMA and RW models consistently matches or outperforms that of LSTMs. Moreover, we find that while traditional models remain robust across varying sample sizes, the performance of LSTMs decreases when the training datasets are reduced. The empirical evidence presented here suggests that the reported superiority of LSTM in financial time series forecasting might be due to the forecasting method that is employed rather than a genuine predictive advantage.

1. INTRODUCTION

Forecasting financial data based on its historical data is crucial for investors, portfolio managers, and companies listed on the market. In this context, traditional stochastic models have been widely used for time series forecasting, among which the most popular are the Autoregressive Integrated Moving Average (ARIMA), Vector Autoregression (VAR), and Generalized Autoregressive Conditional Heteroscedasticity (GARCH) models. Even if these econometric models have well-founded statistical properties, they require linear models and stationary data for their estimations [1, 2]. Thus, the complex and non-stationary structure of financial data presents significant challenges for forecasting series [3-6]. Thus, elimination of structural problems was a priority. Recently, the Long Short-Term Memory (LSTM) has become one of the most popular deep learning methods, often used as an alternative to traditional econometric models for forecasting. It incorporates both long- and short-term memories, handling nonlinear patterns without requiring a specific model structure [7]. In forecasting financial markets and economic indicators, LSTM is one of the most widely used methods. For instance, LSTM models have been used in stock price prediction, cryptocurrency forecasting, commodity price prediction, and foreign exchange rate forecasting [6, 8-13]. Additionally, LSTM has also been used in other domains such as image recognition, disease diagnosis, and speech recognition [14-26].

*Corresponding author, e-mail: lgul0809@gmail.com

This study aims to compare the forecasting performances of two traditional econometric models, specifically ARIMA and Random Walk (RW), with that of the LSTM model by forecasting future values of the S&P 500 index. Depending on the type of dataset, there is no unanimous relative superiority of any method over others. Some studies argue that the ARIMA model outperforms better than LSTM models [27-34], on the other hand, others suggest that the LSTM model performs better than the ARIMA models [35-43]. Thus, it can be stated that there is no consensus on the superiority of one technique over another in general. At this point, we aim to present a different perspective on the forecast performance comparison between ARIMA and LSTM, thereby enriching the method of comparison with static versus dynamic forecast approaches across various cross-validation methods. Thus, we try to add more evidence to the discussion about the dominance of one technique over the other.

Importantly, most earlier comparison studies use dynamic forecasting for ARIMA, where forecasted values are fed back in as inputs over time. This method leads to the accumulation of forecast errors and puts econometric models at a disadvantage compared to LSTM, which utilizes real (observed) data during forecasting (static forecasting). The former can amplify early errors and cause forecasts to drift toward the deterministic component of a model. On the other hand, LSTM forecasts typically generate the statistics by feeding observed lagged values from the test segment through a network with fixed, pre-estimated parameters (static forecasting). When the performance measures of ARIMA and RW are generated only under dynamic forecasting, while LSTM uses observed lag values, the linear models, such as ARIMA and RW, face a systematically harsher information setup, and their relative performance is biased downward. This asymmetry might be the reason why some papers cited above report that LSTM dominates ARIMA and RW models. The present study explicitly employs both static and dynamic forecasting methods for ARIMA and RW. It compares them with the static forecasts of LSTM. This generates comparable statistics, in which differences in forecast accuracy can be attributed to model structure rather than to differences in the information available at the time of forecasting. Our estimates suggest that static ARIMA and static RW forecasts can match or surpass LSTM across multiple cross-validation schemes and error measures for S&P 500 log-prices. Thus, these results suggest that the difference in performance between LSTM and traditional econometric models may be due to the selection of static versus dynamic forecasting, rather than their technical specifications. The main contribution of this study lies in demonstrating the differences between the algorithms considered, distinguishing it from previous works that use similar forecasting approaches for ARIMA and RW models.

Understanding the static and dynamic forecast methodologies in ARIMA and RW models requires examining their forecast generation beyond their in-sample forecast comparison. The generating forecasts using ARIMA and RW models involves estimating parameters from the dataset, followed by either the dynamic method, which produces out-of-sample forecasts, or the static method, which generates in-sample forecasts. The dynamic method forecasts future values by utilizing previously forecasted values in the recursive forecasting process. In contrast, the static method uses the observed or actual values as inputs, though both share the initial step of parameter estimation using the complete dataset [2, 44, 45][1]. However, the LSTM's static forecast process differs; the dataset is initially partitioned into training and test sets, then parameters are optimized on the training set, and last, forecasts are subsequently generated for the test set using the observed values and the optimized parameters. Consistent with comparative studies involving LSTM, ARIMA, and RW, this research employs a clear separation of the dataset into training and test sets. To enhance methodological comparability, the study proposes that forecasts for the ARIMA and RW models should be generated using the actual values from the test set, thus aligning the procedure with the static forecasting method and the standard LSTM forecast method. Furthermore, the results from the dynamic forecasting method will be included in the final comparisons to elucidate the procedural differences between these two forecasting techniques.

To compare the models under consideration, we employ three different cross-validations, unlike the literature, which generally uses only one cross-validation. Thus, the forecasting performance of the models can be compared using not only the entire data series but also their subsamples. In the first cross-validation

[1] Reference [44], pp. 49-50 and reference [45], pp. 234-237

method (TS, Time-series Split), 80% of the dataset is gathered for training, and 20% is reserved for testing the models' forecasts. The second cross-validation employs a time-series split method adapted from the K-fold technique (FTS, 5-Fold Time-series Split), dividing the dataset into five folds. In the FTS method, the initial training set and test set are determined by dividing the data set into five folds. In each following fold, each fold adds the test set of the previous fold to the training set. The iterative procedure continues until the fifth fold. An alternative methodological approach to cross-validation, designated as the Blocked 5-Fold Time-series Split (BFTS), involves segmenting the dataset into five separate divisions. The BFTS method differs fundamentally from the standard FTS approach in its handling of temporal leakage. Temporal leakage occurs when information from the future inadvertently influences model training. The BFTS method stops temporal leakage. It does this by ensuring that each new training part starts immediately after the last one finishes. The test parts are placed in the right spots, too. The FTS and BFTS both divided the data into different time-based groups. This segmentation enables researchers to evaluate the model's performance under different time-based conditions. By testing the model across multiple temporal arrangements, a more comprehensive understanding of its reliability emerges. We use Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) to compare models. In the literature, although only one or two forecast accuracy measures are used for evaluation, our study employs four commonly used performance criteria to assess the robustness of the findings.

Section 2 explains the LSTM, ARIMA, and RW models employed, along with the performance criteria used to compare the forecasts. Section 3 outlines the parameter selection process for the LSTM and ARIMA models, as well as the selection of the model specifications used in the comparison. Section 4 describes the dataset and provides details regarding the cross-validation methodologies employed. The empirical evidence is reported in section 5. Finally, section 6 concludes the paper.

2. METHODOLOGY

2.1. LSTM Model

Conventional machine learning techniques have been successfully employed in areas such as image recognition, audio recognition, and natural language processing, which do not need memory utilization [46-49]. However, forecasting time series data requires a memory mechanism in which previous observations do affect the current value. To address this, Recurrent Neural Networks (RNNs), which incorporate memory, were developed [50]. Despite their advantages, RNNs face several problems, including exploding or vanishing gradients, due to their recursive structures and deep layers [7]. In other words, the effect of previous observations increases too rapidly over time, and the forecast value exploits, or the effects of post-observation disappear from the current value, too quickly. To overcome these problems, the LSTM was introduced by Hochreiter and Schmidhuber (1997) [7]. The LSTM model explicitly incorporates the long-term trend for forecasting, and it is extensively used in the literature for forecasting time series data [51].

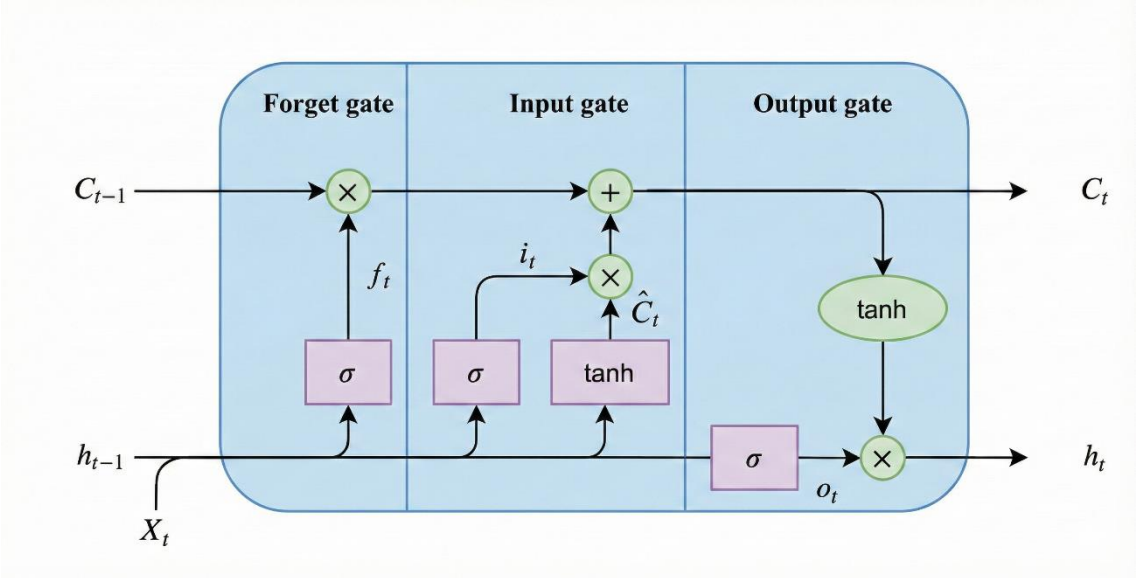

Figure 1. *LSTM Mechanism [52]*

Figure 1 shows the structure of an LSTM neuron. Here, the LSTM architecture consists of a memory cell and three gates. In Figure 1, ⊕ and ⊗ denote addition and multiplication in the model, σ denotes the sigmoid activation function, hyperbolic tangent(tanh) denotes the tanh activation function, and the black arrows in the picture show the direction of information flow. The mathematical equations of the LSTM are as follows:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{1}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2}$$

$$\hat{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{3}$$

$$C_t = f_t C_{t-1} + i_t \hat{C}_t \tag{4}$$

$$O_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{5}$$

$$h_t = O_t \, tanh(C_t) \tag{6}$$

where f_t is the forget gate, i_t is the input gate, $\hat{C}_t$ is the tanh layer. C_t is the new cell state vector at time step t, O_t is the output gate, x_t is the input vector at the time slot. h_t is current cell output, h_{t-1} is the previous cell output. W_f, W_i, W_c,and W_o are the input weight matrices for the forget gate, input gate, candidate cell state vector, and output gate, respectively. U_f, U_i, U_c and U_o are the recurrent weight matrices for the forget gate, input gate, candidate cell state vector, and output gate, respectively. b_f, b_i, b_c and b_o ae the bias vectors for the forget gate, input gate, candidate memory content, and output gate, respectively.

The LSTM models address the gradient problems that RNN produces. The LSTM architecture consists of three distinct gates, each serving a specific purpose. The forget gate, as given in Equation (1), generates a coefficient between 0 and 1; a value of 1 indicates the complete retention of information, while 0 indicates the total erasure. Equations (2)-(4) are for the memory gate. They determine which new data points are stored within each cell. This process begins with a sigmoid layer, referred to as the 'input gate layer,' which determines the values to be modified. Later, a tanh layer generates a vector of new candidate values to be integrated into the state. Equation (5) is used for the output gate, which overwrites the information extracted from each cell. The specific output is contingent upon the filtered and newly added data, as well as the cell state. h_t in Equation (6) is for the final estimate. Importantly, for the forecast value, the observed variable x_t, rather than the forecasted value of x_t, is included. As we will see later, this is not the case for the dynamic forecasts for the ARIMA and RW models.

2.2. ARIMA Model

ARIMA is a combination of the Autoregressive (AR) processes, which involve regressing a variable on its own lagged values, and the Moving Average (MA) processes, which incorporate past random shocks. The Integrated (I) part refers to the process of differencing non-stationary data a certain number of times (d) to achieve stationarity. The basic elements of the ARIMA (*p, d, q*) model are as follows [1, 53].

p: represents the order of AR(*p*) process. An autoregressive model refers to a regression model that uses the dependencies between the dependent variable and its lagged values. The AR(*p*) process is as follows

$$Y_t = \beta_0 + \sum_{j=1}^{p} \beta_j Y_{t-j} + \varepsilon_t \tag{7}$$

where Y_t is the dependent variable, β_0 is constant term, β_j are autoregressive coefficients, and ε_t is a Gaussian white noise residual series with zero mean and variance σ_ε^2.

q: represents the order of the MA(*q*) process. The moving average model is a regression model that uses the lagged error terms. The process MA(*q*) is as follows

$$Y_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j \varepsilon_{t-j} + \varepsilon_t \tag{8}$$

where α_0 is the constant term, α_j are the coefficients of the lagged error terms. We assume that the ε_t is the residual term.

d: represents integration order. It shows the degree of difference needed to make the series stationary. Since the ARIMA model is based on a regression model, the series must be stationary. If the series is not stationary, the model is converted to stationary by taking the difference according to the number of unit roots.

By combining these two models, an ARIMA model of order (*p, d, q*) is formed:

$$\Delta^d Y_t = \beta_0 + \sum_{i=1}^{p} \beta_i \Delta^d Y_{t-i} + \sum_{j=1}^{q} \alpha_j \varepsilon_{t-j} + \varepsilon_t \tag{9}$$

where Δ is the differencing operator and *d* is the degree of differencing.

In the ARIMA model, usually dynamic forecasting is used. Here, the dynamic ARIMA forecasting can be written as follows:

$$\Delta^d \hat{Y}_{t+1} = \hat{\beta}_0 + \sum_{i=0}^{p} \hat{\beta}_i \Delta^d \hat{Y}_{t-i} + \sum_{j=1}^{q} \hat{\alpha}_j \hat{\varepsilon}_{t-j} \, . \tag{10}$$

In which the forecasted value of the lagged dependent variable is used as a regressor. However, in the static ARIMA forecasting model, observed lagged values of the dependent variable are used, as in the LSTM model:

$$\Delta^d \hat{Y}_{t+1} = \hat{\beta}_0 + \sum_{i=0}^{p} \hat{\beta}_i \Delta^d Y_{t-i} + \sum_{j=1}^{q} \hat{\alpha}_j \hat{\varepsilon}_{t-j} \, . \tag{11}$$

As seen in equations (10) and (11), while the dynamic forecasting method uses previously forecasted values for the lagged values of Y_t, ($\widehat{Y_t}$), the static forecasting method uses the actual values for the lagged values of Y_t. In our study, we employ the static forecasting method in ARIMA models to make them comparable to LSTM models, as LSTM utilizes actual lagged observations to forecast future values. Another important point to mention here is the estimated parameters of the ARIMA models. Since the LSTM model fixes the parameters after training and forecasts the test set with fixed parameters and real values, a similar approach is followed with the ARIMA models for comparability. After training the ARIMA models, the estimated parameters are kept unchanged, and the test set forecasts are calculated using these fixed parameters. The

procedures for forecasting the test set data in both LSTM and ARIMA models are similar. In both cases, the models use fixed parameters estimated with the training set, and real data values are fed into the model to forecast future values.

2.3. Random Walk Model

The Efficient Market Hypothesis (EMH) suggests that the returns of an asset (prices) are inherently unpredictable [54, 55]. The hypothesis assumes that because current asset prices reflect all publicly available information, systematic forecasts cannot consistently generate abnormal returns over the long term. Thus, the RW model is considered the fundamental benchmark for stock price forecasting. Parallel to this, Meese and Rogoff (1983) argue that a simple RW process often outperforms complex structural models in exchange rate forecasting [56].

The RW models assume that price changes in capital markets do not follow any known direction or pattern and are therefore unpredictable. RW implies that, at a given point in time, the size and direction of the price change are chosen randomly with respect to the stock of knowledge available at that time. RW is associated with the efficient market hypothesis, which states that current market prices reflect all publicly available information, including past prices and current market conditions.

The RW model with drift is a special case of the ARIMA model, where p = q = 0 and d = 1, specifically an ARIMA(0,1,0) model. In other words:

$$Y_t = \beta_0 + Y_{t-1} + \varepsilon_t \tag{12}$$
$$Y_t - Y_{t-1} = \beta_0 + \varepsilon_t \tag{13}$$
$$\Delta Y_t = \beta_0 + \varepsilon_t \, . \tag{14}$$

The forecasting methods of the ARIMA model: $\hat{Y}_{t+1} = \hat{\beta}_0 + \hat{Y}_{t-1}$ is used for the dynamic forecasting, and, similar to the LSTM model, $\hat{Y}_{t+1} = \hat{\beta}_0 + Y_{t-1}$ is used for the static forecasting.

2.4. Performance Criteria

The employed performance criteria are used twice. These are model parameter selection and inter-model performance comparison. While a single criterion is often sufficient for parameter selection, the heterogeneity of model structures in comparative analyses requires the consideration of multiple criteria. To be particular, in comparing different models, both error-term-based criteria and multiple criteria are commonly employed. In this study, various performance criteria are used for both model parameter selection and model comparison. RMSE is employed for both LSTM hyperparameter tuning and ARIMA parameter selection. Furthermore, to ensure the robustness of the comparative analysis across inter-model performance, four performance criteria are used. Their mathematical formulations of these metrics are as follows:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t| \tag{15}$$
$$MSE = \frac{1}{n}\sum_{t=1}^{n} e_t^2 \tag{16}$$
$$RMSE = \sqrt{MSE} \tag{17}$$
$$MAPE = \frac{1}{n}\sum_{t=1}^{n} \frac{|e_t|}{|Y_t|} \tag{18}$$

where e_t refers to forecast error, Y_t is observed value of the dependent variable and n is the number of observations in the forecasting period.

Each of these performance criteria has its own advantages and disadvantages. Even if MAE is easy to compute and interpret. MSE is more sensitive to outliers [57-59]. Although the RMSE shares the same scale as the data with MSE, it remains vulnerable to outliers [60]. The MAPE is often favored as a unitless

metric. However, in instances where errors are zero or near-zero, MAPE can yield infinite or excessively large values. Another disadvantage of MAPE is that it penalizes estimates of outliers more severely than those that are closer to the sample means [61].

The present paper uses the static forecasting method as the main benchmark against LSTM. Note that LSTM forecasts employ a parallel method to produce forecasts, as in the neural network. In both cases, model forecasts are generated after using the observed lagged variable rather than recursively feeding back previously forecasted values.

3. MODEL SELECTION

Selecting the appropriate parameters is crucial for a model's performance. First, one needs to choose the predetermination of specific parameters termed hyperparameters in the LSTM model, while they are referred to as model parameters in the ARIMA model. While the Keras-Tuner procedure is applied for LSTM to determine these parameters, the autocorrelation function (ACF) and partial autocorrelation function (PACF) graphs are used for the ARIMA model selection.

The LSTM model architecture consists of one input layer, two hidden layers, a single dropout layer, and two dense layers. Thus, in order to eliminate the overfitting problem, we employed the Keras-Tuner methodology for hyperparameter optimization. This procedure is done through a nested cross-validation framework wherein the dataset is partitioned into three distinct segments: 60% allocated for training, 20% designated for validation, and 20% reserved for testing. Importantly, the hyperparameter search process utilizes the 20% validation subset derived exclusively from the training partition for model refinement and evaluation. At the same time, the employed test set remains and is used solely for assessing the overall performance of the ultimate model.

The hyperparameters used in the experiment are as follows:

- To evaluate the performance on different architectures for hidden layers and dense layers, experiments were conducted using four activation functions: Linear, Rectified Linear Unit (ReLU), Sigmoid and tanh.
- A range of 8 to 512 units was implemented for the hidden and dense layers. Experiments were executed with an incremental step of 16 units to identify the optimal unit density.
- To prevent overfitting, dropout was applied to the hidden layers. To determine the optimum dropout rate, experiments were conducted in the range of 0.1-0.5 with a fixed step of 0.1.
- To determine the optimum learning rate, four alternative rates frequently used in the literature, namely 0.1, 0.01, 0.001, and 0.0001, were included in the experiments.

Keras-Tuner is set with 200 epochs, a batch size of 32, and 500 models are used. Additionally, stop_early (tf.keras.callbacks.EarlyStopping) is employed to prevent overfitting during training. After trying 500 models, the model that gives the lowest RMSE according to the validation data is selected as the best model. Then, 200 epochs are run with the best model, and the epoch number yielding the lowest RMSE value is selected as the best epoch. Finally, using the best model and the best epoch, a set of batch sizes (16, 32, 48, 64, 80) is tried, and the batch size that results in the minimum RMSE is chosen as the best batch size. The hyperparameters of the LSTM model that yielded the optimal values, as indicated by the results, are presented in Table 1.

Table 1. *Hyperparameters Selected for LSTM*

Hyperparameters	
Units for Layer 1	40
Units for Layer 2	168
Units for Dense 1	48
Units for Dense 2	1
Activation Function for Layer 1	Linear
Activation Function for Layer 2	tanh
Activation Function for Dense 1	Linear
Activation Function for Dense 2	ReLU
Dropout	0.4
Learning Rate	0.001
Epoch	125
Batch Size	48

Regarding the ARIMA model selection, the stationarity of the time series data is first assessed. For this reason, the Augmented Dickey–Fuller (ADF) test, developed by Dickey and Fuller (1981), is used [62]. The results indicate that the series achieved stationarity after first-order differencing, denoted as I(1). Following the establishment of data stationarity, the ACF and PACF plots are analyzed to decide which models to try. Similar to the nested cross-validation methodology employed for LSTM model selection, PACF and ACF plots are generated for all partitions of the data set.

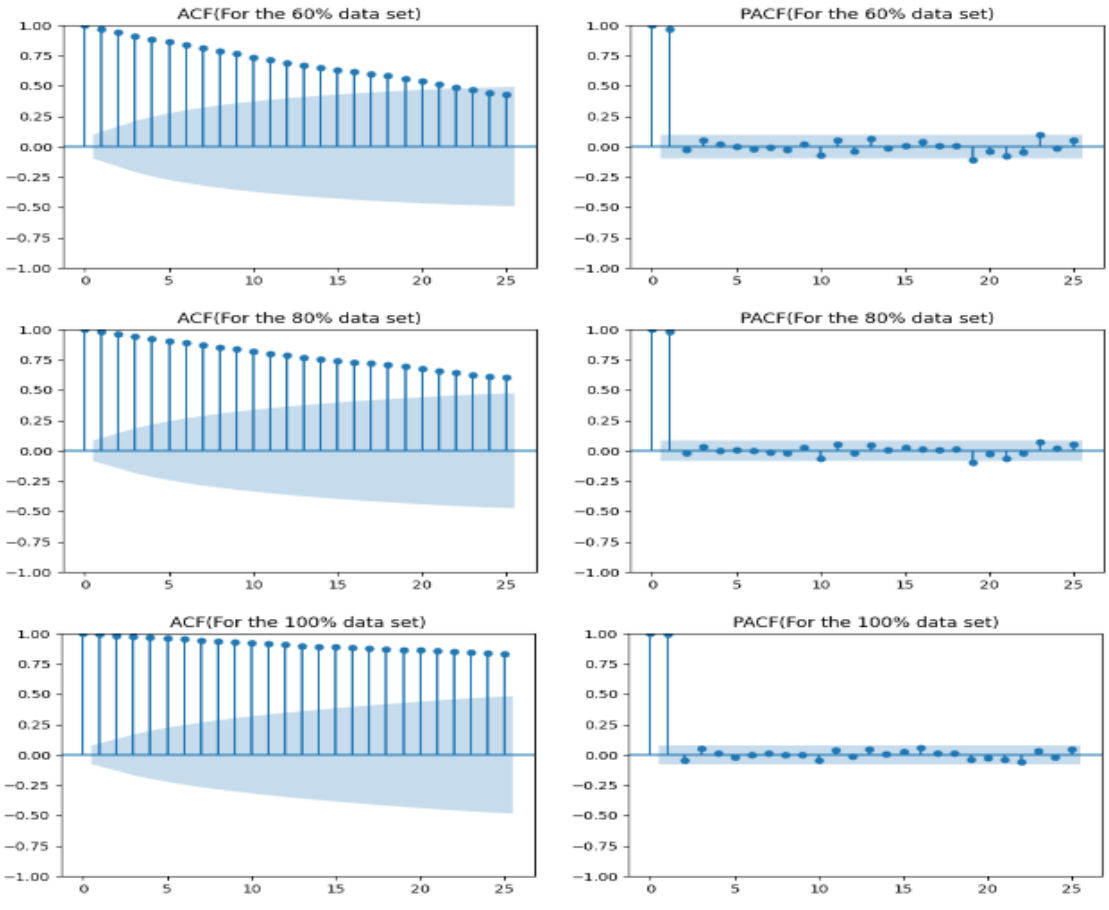

Figure 2. *ACF and PACF Plots*

ACF and PACF plots, as reported in Figure 2, reveal that the series exhibits non-stationarity, and its first difference appears to follow an AR(1) process. However, to determine the optimal model specification, a range of ARIMA models were evaluated based on the validation set, including (0, 1, 1), (0, 1, 2), (1, 1, 0), (1, 1, 1), (1, 1, 2), (2, 1, 0), (2, 1, 1) and (2, 1, 2). The relevant statistics of these specifications are reported in Table 2. The table reports both static and dynamic ARIMA estimations. Analysis of these results indicates that the ARIMA(2, 1, 0) model exhibits the best performance across all criteria for the static ARIMA. For the dynamic ARIMA, the (2, 1, 1) model demonstrates optimal performance across all evaluated criteria.

Table 2. *ARIMA Model Selection Results*

	MSE	**RMSE**	**MAE**	**MAPE**
Order		**Static ARIMA**		
(0, 1, 1)	0.00007019	0.00837798	0.00626232	0.00073012
(0, 1, 2)	0.00007012	0.00837382	0.00625832	0.00072965
(1, 1, 0)	0.00007014	0.00837516	0.00625257	0.00072898
(1, 1, 1)	0.00007223	0.00849876	0.00633502	0.00073855
(1, 1, 2)	0.00008165	0.00903595	0.00695967	0.00081134
(2, 1, 0)	0.00007009***	0.00837209***	0.00625029***	0.00072868***
(2, 1, 1)	0.00009130	0.00955493	0.00740887	0.00086365
(2, 1, 2)	0.00008673	0.00931304	0.00712562	0.00083051
		Dynamic ARIMA		
(0, 1, 1)	0.00466369	0.06829121	0.05981354	0.00695741
(0, 1, 2)	0.00462439	0.06800288	0.05951207	0.00692229
(1, 1, 0)	0.00465098	0.06819808	0.05972952	0.00694763
(1, 1, 1)	0.00465445	0.06822349	0.05975671	0.00695080
(1, 1, 2)	0.00458146	0.06768648	0.05918602	0.00688431
(2, 1, 0)	0.00461968	0.06796824	0.05948001	0.00691856
(2, 1, 1)	0.00456445***	0.06756075***	0.05905641***	0.00686921***
(2, 1, 2)	0.00458217	0.06769175	0.05918521	0.00688421

Notes: *** is for marking the lowest value

4. DATA AND CROSS VALIDATION

This study estimates and compares the performance of three different forecasting models (LSTM, ARIMA, and RW) under various cross-validation testing methods. We conducted all of our analyses using the Python programming language within the Visual Studio Code environment. The "tensorflow.keras" library is used for the estimation of the LSTM model, while the "statsmodels.api" library is used for the ARIMA and RW models' estimations. The Timeseries-Split method from the "sklearn.model_selection" library is applied for cross-validation, and the "sklearn.metrics" library is used to calculate performance criteria. The data set employed here consists of the logarithms of the S&P 500 stock index's closing prices. To eliminate the effect of the COVID-19 pandemic, the data set starts in 2022. The study used a total of 670 data points from the daily S&P 500 index, obtained from Yahoo Finance, between March 1, 2022, and September 3, 2024.

To evaluate the models' performances, we used the cross-validation technique. Thus, we split the available data into multiple folds for training and validation purposes, with each fold used for training and validation. Then, the forecasting methods are compared under three different cross-validation settings. The results from each validation step are then used to evaluate the model's performance, and this average is used in model comparison [63,64]. Three different cross-validation methods are used in the study. The first of these is the TS method, which splits the data series into two parts: 80% for training and 20% for testing, as shown in Figure 3. This method is considered the standard data division method, frequently used in the literature.

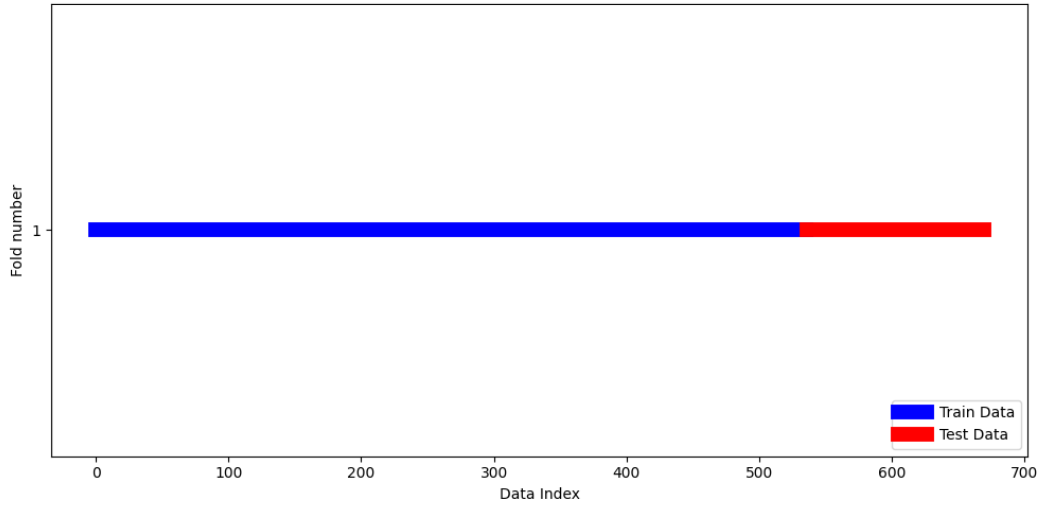

Figure 3. *Structure of 5-Fold Time-series Split*

The second cross-validation method used is the FTS method, which is a multi-fold time series partitioning process similar to K-fold cross-validation.

In FTS, the dataset is partitioned into folds according to a predetermined parameter k. Initially, the training and test sets for the first fold are defined according to the parameter k, as shown in Figure 4. Then, for each subsequent fold, the test set of the previous partition is incorporated into the training set of the current fold, creating an expanding training set. This process goes on step by step. Each new part includes a bigger share of the data. It ends when the last fold takes in the whole dataset. Fold 5 in Figure 4 shows this. The dataset gets split into five parts in this study. Figure 4 presents that setup.

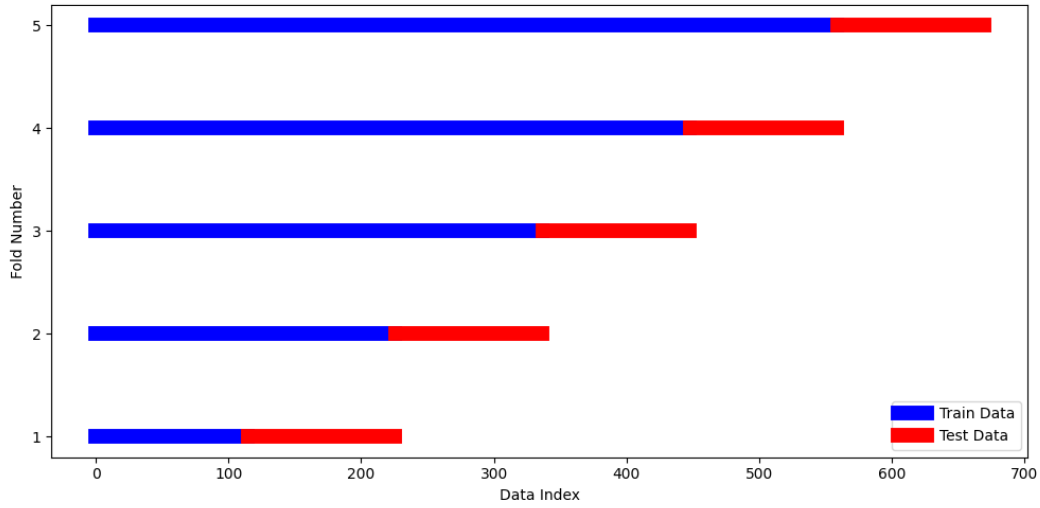

Figure 4. *Structure of Blocked 5-Fold Time-series Split*

The third cross-validation method, BFTS, prevents the connection between folds, unlike FTS, to prevent back leakage in the dataset. Here, the training set of each fold starts from where the test set of the previous fold ends. Within the scope of the study, the dataset is divided into five folds using the BFTS approach. The arrangement of data series according to BFTS is shown in Figure 5.

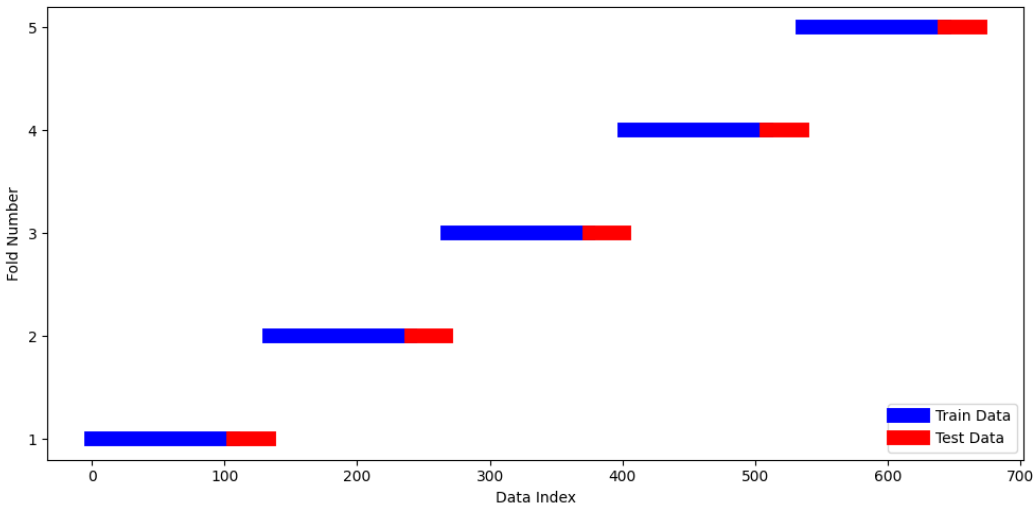

Figure 5. *Structure of Blocked 5-Fold Time-series Split*

The graphical depictions of how the S&P 500 index data is split with respect to TS, FTS, and BFTS are provided in Figures A.1, A.2, and A.3 of the appendix, respectively.

5. EMPIRICAL EVIDENCE

In this study, the performances of the ARIMA, RW, and LSTM models are compared using three different cross-validation techniques (TS, FTS and BFTS) and four different performance criteria (MSE, RMSE, MAE, and MAPE). Additionally, since both the static and dynamic forecasting methods are used for forecasting the test set in ARIMA and RW models, there are a total of 60 different comparisons. The relevant statistics, organized by cross-validation method, are reported in Table 3.

Table 3. *Forecast Metrics*

	LSTM	ARIMA(S)	ARIMA(D)	RW(S)	RW(D)
			Time-series Split		
MSE	0.00007245	*0.00007009	0.00456445	0.00007016	0.00467945
RMSE	0.00851204	*0.00837209	0.06756075	0.00837636	0.06840648
MAE	0.00640883	*0.00625029	0.05905641	0.00626193	0.05992149
MAPE	0.00074743	*0.00072868	0.00686921	0.00073007	0.00696997
			5-Fold Time-series Split		
MSE	0.00880150	*0.00010519	0.01441618	0.00010546	0.01445417
RMSE	0.08771578	*0.00979985	0.10384603	0.00981131	0.10373040
MAE	0.07694178	*0.00754839	0.09262481	0.00755054	0.09252848
MAPE	0.00823196	*0.00090180	0.01106450	0.00090207	0.01105272
			Blocked 5-Fold Time-series Split		
MSE	0.01374079	0.00013091	0.00098824	*0.00012429	0.00096275
RMSE	0.10067937	0.01075839	0.02217435	*0.01046916	0.02661079
MAE	0.09687155	0.00844926	0.02364309	*0.00820555	0.02357785
MAPE	0.01129902	0.00100717	0.00282837	*0.00097851	0.00281817

*Notes: (S) indicates forecasts using the static method; (D) indicates forecasts using the dynamic method. * is for marking the best performance*

Table 3 reveals that the model's performance metrics are consistent with each other, and the results are parallel. Although different metrics can be sensitive to outliers in various ways, the statistics reveal that all four metrics agree and are not significantly affected by extreme values in the data. For example, where the MSE criterion is at its minimum, the other three criteria also give their minimum values. This consistency

allows us to assess model comparisons more easily. In the TS cross-validation, the forecasting performances of the LSTM, ARIMA(S), and RW(S) models are parallel. When using dynamic forecasting with the ARIMA and RW models, their ability to predict accurately drops a lot. However, the ARIMA model shows the best forecasting results in this cross-validation test when it uses static forecasting. The FTS cross-validation approach reveals a considerable decrease in model performance across all forecasting methods, with the LSTM model exhibiting the most significant decline in predictive power. However, the relative performance ranking of the models remains consistent with the TS cross-validation. According to the results of BFTS cross-validation, however, the performance ranking of the models changes, with RW(S) showing the best performance and LSTM showing the worst. Moreover, while there is a performance increase in ARIMA(D) and RW(D) models in BFTS, performance decreases in the other models.

Table 4. *Performance Comparison of LSTM with ARIMA and RW Models (Percentage)*

	ARIMA(S)	**RW(S)**	**ARIMA(D)**	**RW(D)**
			Time-series Split	
MSE	96.74	96.84	6299.73	6458.43
RMSE	98.36	98.41	793.71	803.64
MAE	97.53	97.71	921.48	934.98
MAPE	97.49	97.68	919.05	932.53
			5-Fold Time-series Split	
MSE	1.20	1.20	163.79	164.22
RMSE	11.17	11.19	118.39	118.26
MAE	9.81	9.81	120.38	120.26
MAPE	10.95	10.96	134.41	134.27
			Blocked 5-Fold Time-series Split	
MSE	0.95	0.90	7.19	7.01
RMSE	10.69	10.40	22.02	26.43
MAE	8.72	8.47	24.41	24.34
MAPE	8.91	8.66	25.03	24.94

*Notes: (S) indicates forecasts using the static method; (D) indicates forecasts using the dynamic method. * is for marking the best performance*

The four performance criteria for the ARIMA(S), RW(S), ARIMA(S), and RW(D) relative to LSTM are reported in Table 4, which provides a better understanding of the magnitude of the differences between the models. A value of less than 100 indicates that the model in comparison performs better than the LSTM, as its MSE (RMSE, MAE, or MAPE) value is lower than that of the LSTM. In the TS cross-validation, ARIMA(S) and RW(S) models outperform the LSTM model across all four performance criteria, and the forecast errors of ARIMA(S) are slightly lower than those of RW(S). However, both the ARIMA(D) and RW(D) models fall behind the LSTM model across all criteria in the TS scenario. In FTS and BFTS, ARIMA(S) and RW(S) show a remarkable improvement in performance compared to the LSTM for all forecast accuracy measures. Under FTS, ARIMA(S) is still the best model, while under BFTS, RW(S) becomes the best model, although the forecasting performances of these two models are very close. These indicate that both ARIMA(S) and RW(S) models are much more efficient in FTS and BFTS than in TS. Interestingly, while ARIMA(D) and RW(D) models perform poorly in both TS and FTS, they outperform the LSTM model in BFTS.

Figures A.4, A.5 and A.6 in the appendix show the observed values alongside the predicted values generated by each model across the three cross-validation approaches—TS, FTS, and BFTS—for all corresponding validation folds. These figures suggest that the model performance comparison by presenting the forecasts of all models under different validation schemes provides a detailed visualization of model behavior. ARIMA(S), RW(S), and LSTM models reveal a strong ability to track observed values in the TS cross-validation. Regarding the ARIMA(S) and RW(S), the static forecasts exhibit stability in TS and generate forecasts closely aligned with the actual values in FTS and BFTS cross-validations. On the other hand, ARIMA and RW models employing the dynamic forecasting method exhibit limitations in capturing the

exact values. The dynamic forecasting specifications employed in ARIMA(D) and RW(D) methodologies reveal a convergence pattern. Over time, their predictions asymptotically approach a trajectory that is mathematically determined by the constant coefficient within the model specification. This is due to the nature of dynamic forecasting, where current forecasts rely on past forecasted values. Thus, after a certain point, the estimates stabilize around the model's estimated parameters. A similar pattern is consistently observed across other cross-validations for dynamic forecasts. As for the LSTM, although it performs well in TS, performance degradation is observed in FTS and BFTS cross-validations. This lower performance is likely due to the decreased availability of data in these cross-validations.

The Diebold-Mariano (DM) test is also used to assess if there is a statistically significant difference in the predictive power between two forecasting models. Here, the null hypothesis (H_0) is that the two models have equal predictive power, meaning no difference exists between their corresponding error series. Conversely, the alternative hypothesis (H_A) asserts that a difference does exist, indicating one model is superior. In this study, the DM test played a crucial role in comparing the performance of five models. DM Test statistics are summarized in Table 5.

Table 5. *Diebold-Mariano Test Statistics*

	ARIMA(D) vs LSTM	**RW(D) vs LSTM**	**ARIMA(S) vs LSTM**	**RW(S) vs LSTM**	**ARIMA(S) vs ARIMA(D)**
			Time-series Split		
DM-Statistic	12.499 ***	12.580 ***	-0.598	-0.590	-12.525 ***
			5-Fold Time-series Split		
DM-Statistic (1.Fold)	14.800 ***	14.811 ***	-17.264 ***	-17.253 ***	-15.700 ***
DM-Statistic (2.Fold)	6.300 ***	6.030 ***	-18.373 ***	-18.401 ***	-9.172 ***
DM-Statistic (3.Fold)	-10.765 ***	-11.012 ***	-36.091 ***	-36.087 ***	-16.889 ***
DM-Statistic (4.Fold)	-18.071 ***	-16.868 ***	-14.531 ***	-14.532 ***	-11.840 ***
DM-Statistic (5.Fold)	0.553	0.648	-20.455 ***	-20.429 ***	-9.605 ***
			Blocked 5-Fold Time-series Split		
DM-Statistic (1.Fold)	1.378	1.334	-6.535 ***	-6.535 ***	-5.455 ***
DM-Statistic (2.Fold)	-47.764 ***	-46.473 ***	-47.875 ***	-47.875 ***	-3.464 ***
DM-Statistic (3.Fold)	-31.518 ***	-30.049 ***	-30.261 ***	-30.261 ***	-3.903 ***
DM-Statistic (4.Fold)	-1.452	-0.935	-8.284 ***	-8.284 ***	-5.544 ***
DM-Statistic (5.Fold)	-23.866 ***	-25.230 ***	-25.974 ***	-25.974 ***	-3.114 ***

*Note: *** indicates that the test statistics exceed the critical value at the 1% significance level.*

The statistics reported in Table 5 show that, for our main conclusions, the most relevant comparisons are those involving LSTM versus ARIMA(S) and RW(S), except in the TS cross-validation fold, where the LSTM model's error terms are significantly different from those of ARIMA(S) and RW(S). The dynamic forecasting specifications employed in ARIMA(D) and RW(D) methodologies reveal a convergence pattern whereby predictions asymptotically approach a trajectory that is mathematically determined by the constant coefficient within the model specification. This implies that ARIMA(S) and RW(S) significantly outperform LSTM. Parallel with the literature, ARIMA(D) and RW(D) reveal the systematic disadvantage of dynamic forecasting relative to static methods, as well as to the LSTM model. Overall, these DM test statistics support our conclusion that static ARIMA and RW provide at least comparable, and often superior, accuracy to LSTM under comparable information sets.

There might be various reasons for the observed superior forecasting power of ARIMA(S) and RW(S) over LSTM. First, if the true model is linear, then the bias-variance tradeoff favors parsimonious models. Note that with the constant term ARIMA(2,1,0) requires only four parameters to be estimated, while the LSTM requires tens of thousands of trainable weights. This parsimony might be the reason it reduces estimation

variance without introducing substantial bias for the S&P 500 returns. Second, LSTM networks require substantially larger training samples; on the other hand, ARIMA and RW models can be reliably estimated with considerably fewer observations. These two might be the reason why LSTM performance worsens considerably in FTS and BFTS. Third, the RW model's strong performance might be due to the Efficient Market Hypothesis. The theory suggests that if stock price changes are largely unpredictable, then the best forecast is simply the last observed value, and more complex models cannot systematically improve upon this benchmark.

Overall, considering all three cross-validation schemes and error metrics, the empirical evidence suggests that static forecasting for ARIMA and RW systematically outperforms their dynamic counterparts. In other words, the static forecasts track the actual values more closely than dynamic forecasts. This is parallel with the superiority of static over dynamic forecasting in the S&P 500 forecasts, suggesting that the static specification provides a more reliable benchmark when comparing traditional models with LSTM.

Makridakis and Hibon's (2000) M-Competitions consistently demonstrate that statistically more complex methods do not necessarily provide better forecasts than simpler ones [65]. A comprehensive meta-analysis of 97 comparisons found that complexity increases forecast error by an average of 27% [66]. Our results, where LSTM produces errors that are up to 111 times larger than those of traditional models in specific cross-validation scenarios, represent an extreme manifestation of this complexity penalty when applied to efficiently priced financial series.

The Efficient Market Hypothesis suggests that financial markets account for all available information to set prices, thereby making sophisticated forecasting models, such as LSTM, unnecessary for predicting future prices. Therefore, this theoretical prediction parallels our empirical evidence. This explains why LSTM's capacity to model complex nonlinear patterns provides no benefit when applied to near-random-walk financial data.

Our empirical conclusions have strong implications for financial data forecasting. First, ARIMA and RM models are computationally more efficient relative to LSTM models; therefore, they facilitate meaningful reductions in operational expenditures and resource allocation requirements. Second, ARIMA and RW provide robust performance across different training set sizes. However, the accuracy of LSTM decreases considerably when the training data is reduced, which is a critical consideration for applications requiring robust performance across diverse market conditions. Third, the static forecasting methodology introduced in this study eliminates the cumulative error propagation that artificially disadvantages traditional models in dynamic forecasting comparisons. Financial practitioners implementing rolling horizon forecasts with access to actual lagged observations should consider static ARIMA forecasting as a computationally efficient alternative to deep learning that delivers superior or comparable accuracy.

The findings presented herein should not be interpreted as suggesting that deep learning methodologies are inferior for time series forecasting applications. The existing literature indicates that LSTM architectures may offer notable advantages when dealing with very large datasets exceeding 50,000 observations, strong nonlinear dynamics, or long multi-step forecasting horizons. In contrast, daily stock index forecasts often involve only a few hundred observations that display near-random-walk behavior. This pattern implies that the comparative strength of traditional econometric models depends on specific data characteristics rather than an inherent methodological drawback. As the M3 Competition of Makridakis and Hibon (2000) demonstrates, hybrid approaches that combine statistical structure with machine learning flexibility may represent a productive direction for future research [65].

Extensions, Limitations and Further Studies:

The dataset we employ begins with the post-COVID-19 pandemic era. This might be considered too short a period. We repeated the analyses starting in 1954, immediately after the end of the Korean War, and extended the sample through 2025. The empirical evidence provided on a single series (S&P500) for the Korean War Era is robust to the main conclusion of the paper. The corresponding LSTM hyperparameter settings and forecast metrics are reported in Tables A.1–A.4 in the appendix. They confirm

that our main conclusion—namely, that static ARIMA and RW match or outperform LSTM under comparable information sets—remains robust over this longer sample. On the limitation side, we consider just one variable with a limited time span. Even if the data set extended starting with the end of the Korean War. Alternative variables with a longer time span, such as different financial variables and non-financial series, including national income, international trade data, and other variables from other countries, can be considered. Alternative deep learning structures can also be considered for the LSTM architecture. Hybrid approaches combining statistical structure with machine learning flexibility may represent an alternative. Lastly, rather than a single step (Daily) ahead forecasts. Multiple-step forecasting can be considered, such as two-day, seven-day and monthly forecasts. Lastly, the model does not include exogenous variables. The day-of-the-week effect could be incorporated using four daily dummy variables.

6. CONCLUSION

This study provides empirical evidence on the forecasting power of LSTM models alongside traditional econometric approaches, including the ARIMA and RW models. These three models are applied to the logarithms of daily S&P 500 index closing values from January 3, 2022, to March 9, 2024. Particular emphasis is placed on the critical distinction between static and dynamic forecasting methodologies for ARIMA and RW.

The basic conclusion of our empirical exercise is twofold: First, static forecasting demonstrates superior performance relative to dynamic forecasting across both ARIMA and RW methodologies. Second, ARIMA with static forecasting (ARIMA(S)) and RW with static forecasting (RW(S)) consistently outperform LSTM across the entirety of cross-validation schemes examined, namely TS, FTS, and BFTS. However, the performance of the LSTM model worsens substantially as the training data size is reduced in FTS and BFTS, which favors simpler models such as RW. Overall, this study demonstrates that the perceived superiority of LSTM over ARIMA and RW in prior literature may stem from biased comparisons using dynamic forecasting for traditional models. When static forecasting is employed to ensure comparability, the superiority of LSTM disappears.

The analyses were conducted using Python, with libraries such as tensorflow.keras for LSTM, statsmodels.api for ARIMA and RW, sklearn.metrics for performance evaluation, and Keras-Tuner for hyperparameter optimization. The source code is available on GitHub for reproducibility.

The empirical evidence presented herein demonstrates the importance of distinguishing between static and dynamic forecasting. The future research may include (1) incorporation of a set of exogenous variables—such as the day-of-the-week effect; (2) the extension of forecasting windows to accommodate multi-temporal horizons—such as 2, 7, and 31 day-ahead forecasting; (3) the evaluation of heterogeneous deep learning architectures across expanded and heterogeneous datasets; and (4) the exploration of hybrid modeling approaches that synthesize complementary methodological strengths of each.

CONFLICTS OF INTEREST

No conflict of interest was declared by the authors.

REFERENCES

[1] Box, G.E.P., Jenkins, G.M., Reinsel, G.C., Ljung, G.M., “Time Series Analysis: Forecasting and Control”, John Wiley and Sons, (2015).

[2] Hamilton, J. D., “Time Series Analysis”, Princeton University Press, (1994).

[3] Deboeck, G.J., “Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets”, (39), John Wiley and Sons, (1994).

[4] Abu-Mostafa, Y.S., Atiya, A.F., "Introduction to financial forecasting", Applied Intelligence, 6: 205-213, (1996).

[5] Schmidhuber, J., "Deep learning in neural networks: An overview", Neural Networks, 61: 85-117, (2015).

[6] Jothi, L.S., "Time Series Analysis Using LSTM Networks and Its Application to Financial Forecasting", In Artificial Intelligence for Financial Risk Management and Analysis, 19-58, IGI Global Scientific Publishing, (2025).

[7] Hochreiter S., Schmidhuber J., "Long Short-Term Memory", Neural Computation MIT-Press, 9(8), 1735-1780, (1997).

[8] Roondiwala, M., Patel, H., Varma, S., "Predicting stock prices using LSTM", International Journal of Science and Research (IJSR), 6(4), 1754-1756, (2017).

[9] Mendes, J.F.B., "Forecasting bitcoin prices: ARIMA vs LSTM", Master's thesis, (2019).

[10] Han, Z., Zhu, X., Su, Z., "Forecasting maritime and financial market trends: Leveraging CNN-LSTM models for sustainable shipping and China's financial market integration", Sustainability, 16(22), 9853, (2024).

[11] You, Y., "Forecasting stock price: A deep learning approach with LSTM and hyperparameter optimization", Highlights in Science, Engineering and Technology, 85: 328-338, (2024).

[12] Wang, J., Hong, S., Dong, Y., Li, Z., Hu, J., "Predicting stock market trends using LSTM networks: overcoming RNN limitations for improved financial forecasting", Journal of Computer Science and Software Applications, 4(3): 1-7, (2024).

[13] Wang, Y., Zhang, H., Huang, B., Lin, Z., Pang, C., "LSTM stock prediction model based on blockchain", High-Confidence Computing, 100316, (2025).

[14] Xiao D., Ketcha M.D., "Deep Speech 2: End-to-end speech recognition in English and mandarin", Journal of Heterocyclic Chemistry, 32(2): 499–503, (1995).

[15] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., Andrew Y.N., "Deep Speech: Scaling up end-to-end speech recognition", ArXiv Preprint ArXiv:1412.5567, (2014).

[16] Huang, Y., Wang, W., Wang, L., "Instance-aware image and sentence matching with selective multimodal LSTM", In Proceedings of the IEEE conference on computer vision and pattern recognition, 2310-2318, (2017).

[17] Zhao, A., Qi, L., Dong, J., Yu, H., "Dual channel LSTM based multi-feature extraction in gait for diagnosis of neurodegenerative diseases.", Knowledge-Based Systems, 145: 91-97, (2018).

[18] Shewalkar, A., "Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU", Journal of Artificial Intelligence and Soft Computing Research, 9, (2019).

[19] Tatsunami, Y., Taki, M., "Sequencer: Deep LSTM for image classification", Advances in Neural Information Processing Systems, 35: 38204-38217, (2022).

[20] Vidya, B., Sasikumar, P., "Parkinson's disease diagnosis and stage prediction based on gait signal analysis using EMD and CNN–LSTM network", Engineering Applications of Artificial Intelligence, 114, 105099, (2022).

[21] Hassanzadeh, H., Qadir, J.A., Omer, S.M., Ahmed, M.H., Khezri, E., "Deep learning for speaker recognition: A comparative analysis of 1D-CNN and LSTM models using diverse datasets", In 2024 4th Interdisciplinary Conference on Electrics and Computer (INTCEC), 1-8, IEEE, (2024).

[22] Khartheesvar, G., Kumar, M., Yadav, A.K., Yadav, D., "Automatic Indian sign language recognition using Media Pipe holistic and LSTM network", Multimedia Tools and Applications, 83(20): 58329-58348, (2024).

[23] Wang, X., Huang, J., Chatzakou, M., Medijainen, K., Toomela, A., Nõmm, S., Ruzhansky, M., "LSTM-CNN: An efficient diagnostic network for Parkinson's disease utilizing dynamic handwriting analysis", Computer Methods and Programs in Biomedicine, 247, 108066, (2024).

[24] Basha, S.A.K., Vincent, P.D.R., Mohammad, S.I., Vasudevan, A., Soon, E.E.H., Shambour, Q., Alshurideh, M.T., "Exploring deep learning methods for audio speech emotion detection: An ensemble MFCCs, CNNs and LSTM", Applied Mathematics & Information Sciences,19(1): 75-85, (2025).

[25] Haq, E.U., Yong, Q., Yuan, Z., Huarong, X., Haq, R.U., "Multimodal fusion diagnosis of the Alzheimer's disease via lightweight CNN-LSTM model using magnetic resonance imaging (MRI)", Biomedical Signal Processing and Control, 104: 107545, (2025).

[26] Liu, Y., Brezzi, L., Liang, Z., Gabrieli, F., Zhou, Z., Cola, S., "Image analysis and LSTM methods for forecasting surficial displacements of a landslide triggered by snowfall and rainfall", Landslides, 22(3): 619-635, (2025).

[27] Han, J.H., "Comparing models for time series analysis", Wharton Research Scholars, Philadelphia, (2018).

[28] Yamak, P.T., Yujian, L., Gadosey, P.K., "A Comparison Between ARIMA, LSTM, and GRU for Time Series Forecasting", In Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, 49-55, (2019).

[29] Andréasson, D., Mortensen, B.J., "Forecasting the OMXS30 - a comparison between ARIMA and LSTM", bachelor's Thesis, Uppsala University, (2020).

[30] Liu, X., Lin, Z., Feng, Z., "Short-term offshore wind speed forecast by seasonal ARIMA-A comparison against GRU and LSTM", Energy, 227, 120492, (2021).

[31] Zhang, R., Song, H., Chen, Q., Wang, Y., Wang, S., Li, Y., "Comparison of ARIMA and LSTM for prediction of hemorrhagic fever at different time scales in China", Plos One, 17(1): e0262009, (2022).

[32] Kobiela, D., Krefta, D., Król, W., Weichbroth, P., "ARIMA vs LSTM on NASDAQ stock exchange data", Procedia Computer Science, 207, 3836-3845, (2022).

[33] Albeladi, K., Zafar, B., Mueen, A., "Time series forecasting using LSTM and ARIMA", International Journal of Advanced Computer Science and Applications, 14(1): 313-320, (2023).

[34] Azani, N.W., Trisya, C.P., Sari, L.M., Handayani, H., Alhamid, M.R.M., "Performance Comparison of ARIMA, LSTM and SVM Models for Electric Energy Consumption Analysis", Public Research Journal of Engineering, Data Technology and Computer Science, 1(2), (2024).

[35] Siami-Namini, S., Tavakoli, N., Namin, A.S., "A Comparison of ARIMA and LSTM in Forecasting Time Series", 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, (2018).

[36] Masum, S., Liu, Y., Chiverton, J., "Multi-step Time Series Forecasting of Electric Load Using Machine Learning Models", In Artificial Intelligence and Soft Computing: 17th International Conference, Zakopane, Poland, (2018).

[37] Karakoyun, E.S., Cibikdiken, A.O., "Comparison of ARIMA Time Series Model and LSTM Deep Learning Algorithm for Bitcoin Price Forecasting", In The 13th Multidisciplinary Academic Conference, Prague, 171-180, (2018).

[38] Alzheev, A.V., Kochkarov, R.A., "Comparative analysis of ARIMA and LSTM predictive models: Evidence from Russian stocks", Finance: Theory and Practice, 24(1): 14-23, (2020).

[39] Elsaraiti, M., Merabet, A., "A comparative analysis of the ARIMA and LSTM predictive models and their effectiveness for predicting wind speed", Energies, 14(20): 6782, (2021).

[40] Wu, H., Chen, S., Ding, Y., "Comparison of ARIMA and LSTM for Stock Price Prediction", Financial Engineering and Risk Management, 6(1): 1-7, (2023).

[41] Hamiane, S., Ghanou, Y., Khalifi, H., Telmem, M., "Comparative analysis of LSTM, ARIMA, and hybrid models for forecasting future GDP", Journal Homepage: http://iieta. org/journals/isi, 29(3): 853-861, (2024).

[42] Nensi, A.I.E., Al Maida, M., Notodiputro, K.A., Angraini, Y., Mualifah, L.N.A., "Performance Analysis of ARIMA, LSTM, and Hybrid ARIMA-LSTM in Forecasting the Composite Stock Price Index", CAUCHY: Jurnal Matematika Murni dan Aplikasi, 10(2): 588-604, (2025).

[43] Köse, N., Gür, Y.E., Ünal, E., "Deep Learning and Machine Learning Insights into the Global Economic Drivers of the Bitcoin Price", Journal of Forecasting, (2025).

[44] Ouliaris, S., Pagan, A., Restrepo, J., "Quantitative macroeconomic modeling with structural vector autoregressions–an EViews implementation", IHS Global, 13, (2016).

[45] Startz, R., "EViews Illustrated", University of California: Santa Barbara, CA, USA, (2019).

[46] Dahl, G., Ranzato, M.A., Mohamed, A.R., Hinton, G.E., "Phone recognition with the mean-covariance restricted Boltzmann machine", Advances in Neural Information Processing Systems, 23, (2010).

[47] Bordes, A., Glorot, X., Weston, J., Bengio, Y., "Joint learning of words and meaning representations for open-text semantic parsing", Artificial Intelligence and Statistics, 127-135, (2012).

[48] Krizhevsky, A., Sutskever, I., Hinton, G.E., "ImageNet classification with deep convolutional neural networks", Communications of the ACM, 60(6), 84-90, (2017).

[49] Zhang, Q., Yang, L.T., Chen, Z., Li, P., "A survey on deep learning for big data", Information Fusion, 42: 146-157, (2018).

[50] Sherstinsky, A., "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network", Physica D: Nonlinear Phenomena, 404, 132306, (2020).

[51] Yang, S., Yu, X., Zhou, Y., "LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example", International Workshop on Electronic Communication and Artificial Intelligence, Shanghai, China, (2020).

[52] https://thorirmar.com/post/insight_into_lstm. Access date: 30.12.2025

[53] Dinardo, J., Johnston, J., "Econometric Methods", McGraw-Hill, (1996).

[54] Fama, E.F., "Random walks in stock market prices", Financial Analysts Journal, 51(1): 75-80, (1965).

[55] Fama, E.F., "Efficient capital markets: A review of theory and empirical work", The Journal of Finance, 25(2): 383-417, (1970).

[56] Meese, R.A., Rogoff, K., "Empirical exchange rate models of the seventies: Do they fit out of sample?", Journal of International Economics, 14(1-2): 3-24, (1983).

[57] Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Winkler, R., "The accuracy of extrapolation (time series) methods: Results of a forecasting competition", Journal of Forecasting, 1(2): 111-153, (1982).

[58] Chatfield, C., "Apples, oranges and mean square error", International Journal of Forecasting, 4(4): 515-518, (1988).

[59] Armstrong, J.S., Collopy, F., "Error measures for generalizing about forecasting methods: Empirical comparisons", International Journal of Forecasting, 8(1): 69-80, (1992).

[60] Armstrong, J.S., "Principles of Forecasting: A Handbook for Researchers and Practitioners", 443-472, (2001).

[61] Fildes, R., "The evaluation of extrapolative forecasting methods", International Journal of Forecasting, 8(1): 81-98, (1993).

[62] Dickey, D.A., Fuller, W.A., "Likelihood ratio statistics for autoregressive time series with a unit root", Econometrica: Journal of the Econometric Society, 1057-1072, (1981).

[63] Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B., "Bayesian Data Analysis", Third Edition, Chapman and Hall/CRC, (2013).

[64] Milà, C., Mateu, J., Pebesma, E., Meyer, H., "Nearest neighbour distance matching Leave-One-Out Cross-Validation for map validation", Methods in Ecology and Evolution, 13(6): 1304-1316, (2022).

[65] Makridakis, S., Hibon, M., "The M3-Competition: results, conclusions and implications", International Journal of Forecasting, 16(4): 451-476, (2000).

[66] Green, K.C., Armstrong, J.S., "Simple versus complex forecasting: The evidence", Journal of Business Research, 68(8): 1678-1685, (2015).

APPENDIX

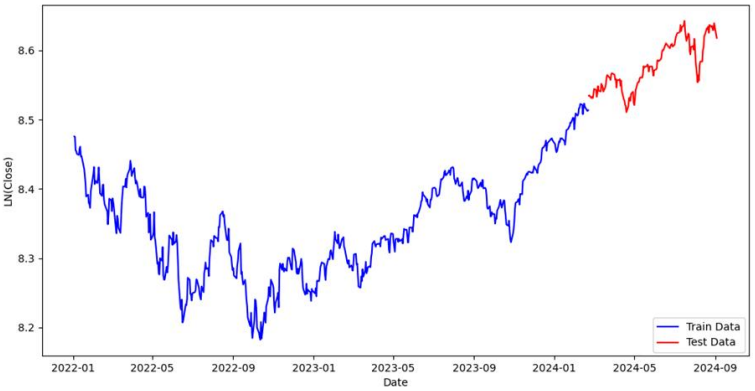

Figure A.1. *Data Split for TS*

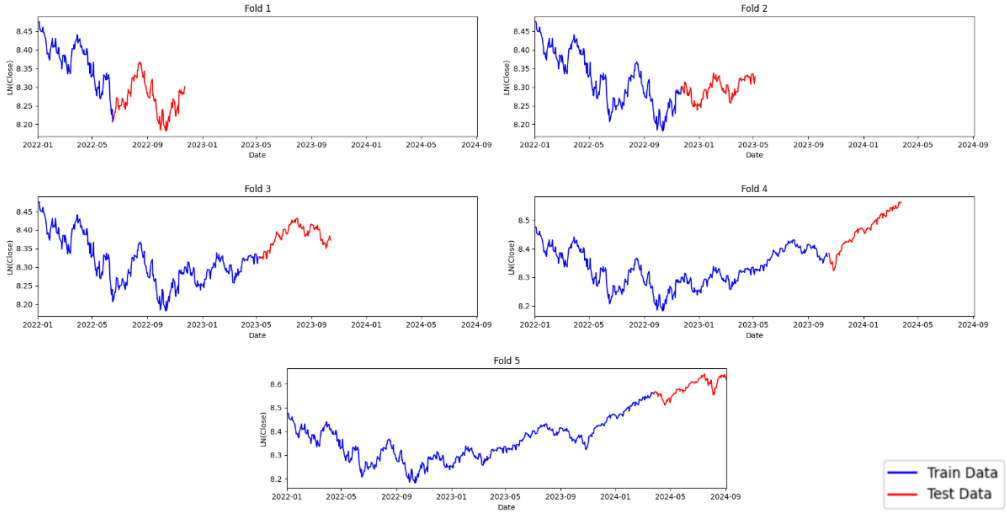

Figure A.2. *Data Split for FTS*

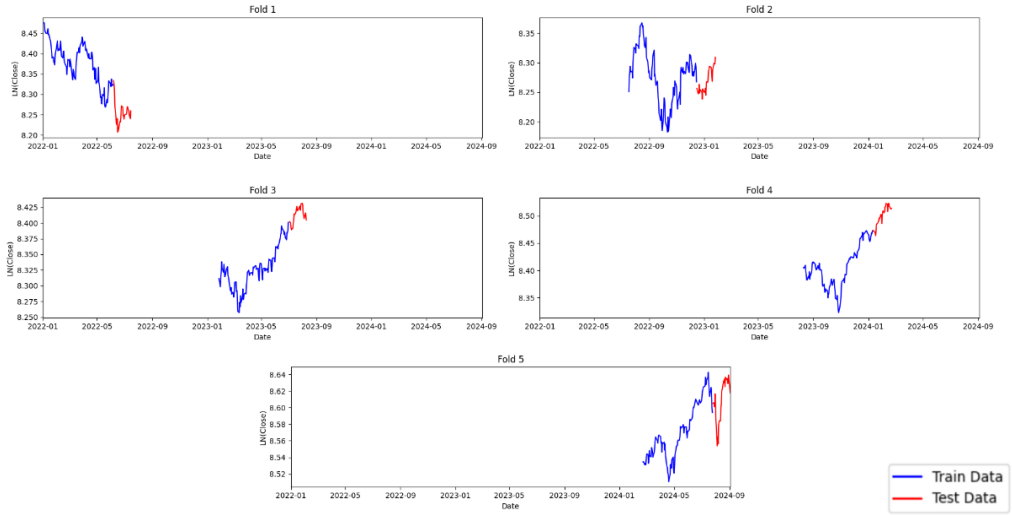

Figure A.3. *Data Split for BFTS*

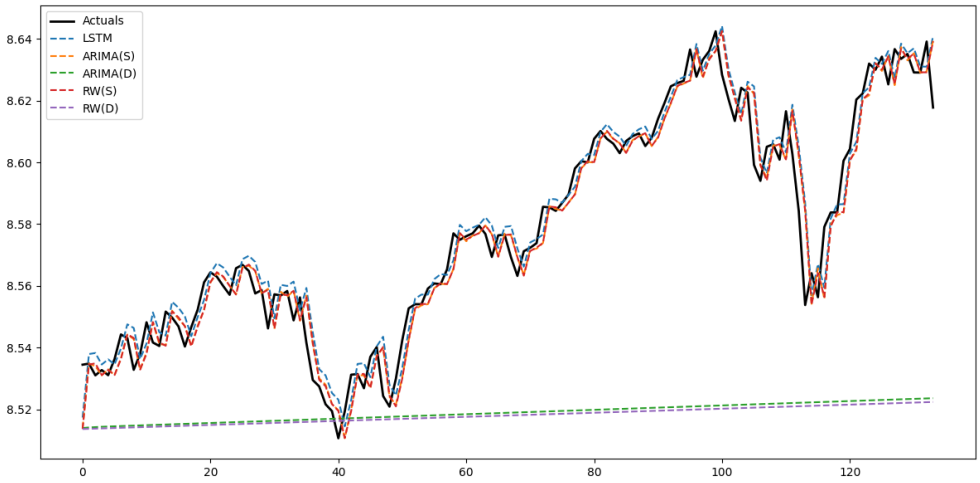

Figure A.4. *Observed and Forecasts for TS*

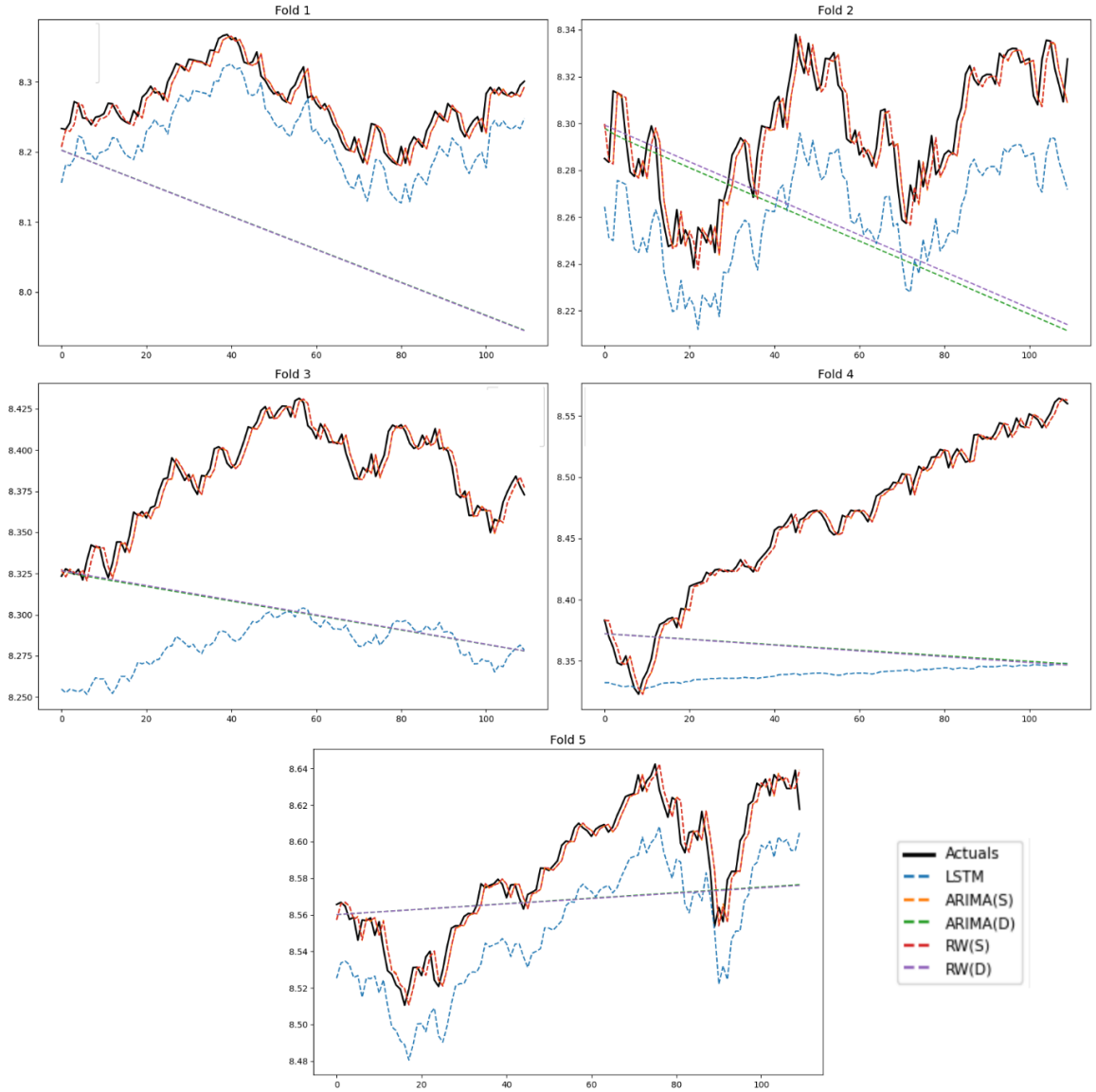

Figure A.5. *Observed and Forecasts for FTS*

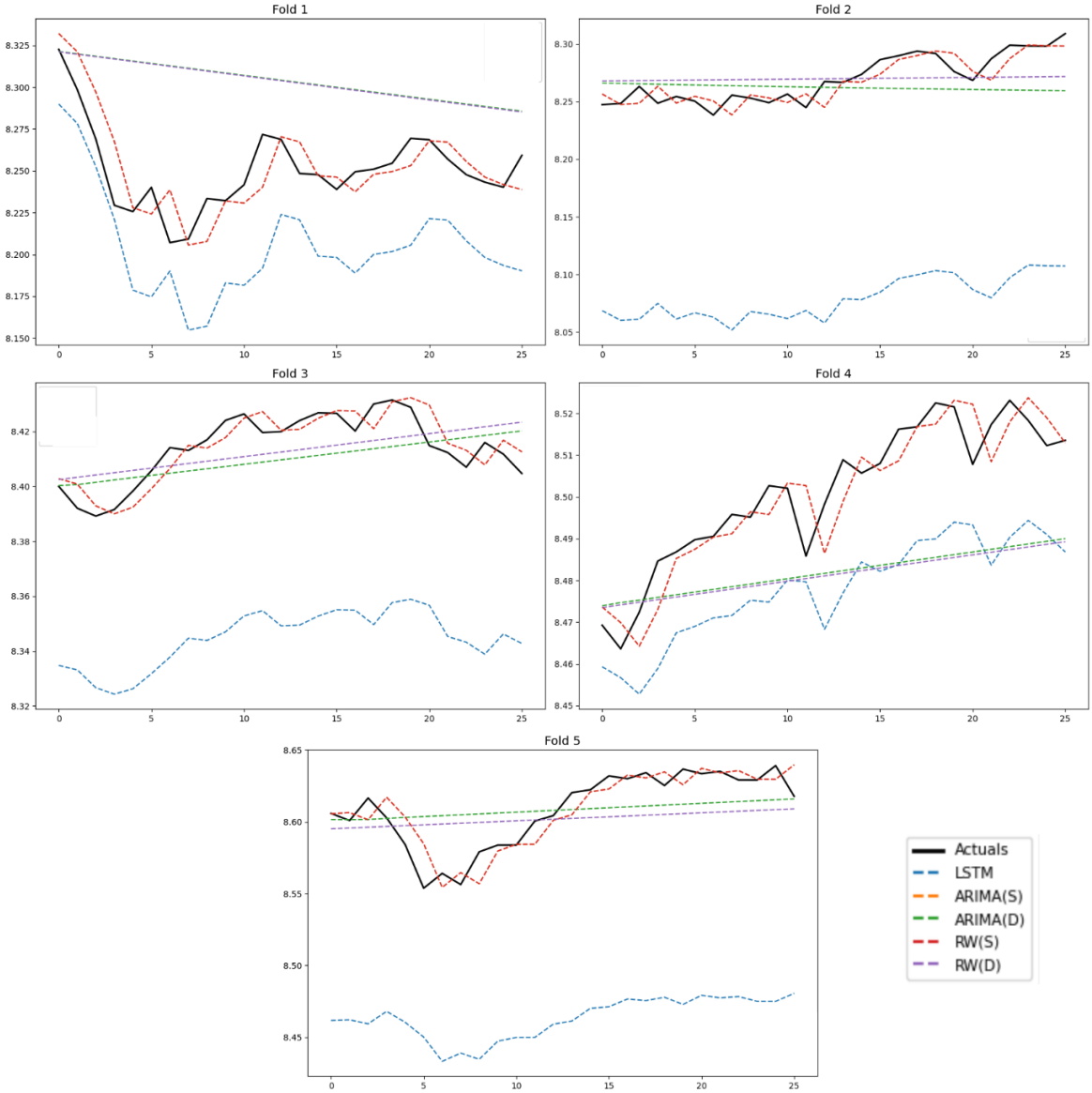

Figure A.6. *Observed and Forecasts for BFTS*

Table A.1. *LSTM Hyperparameter Selection for the 1954–2025 Period*

Hyperparameters	
Units for Layer 1	184
Units for Layer 2	280
Units for Dense 1	40
Units for Dense 2	1
Activation Function for Layer 1	tanh
Activation Function for Layer 2	ReLU
Activation Function for Dense 1	ReLU
Activation Function for Dense 2	Linear
Dropout	0.1
Learning Rate	0.001
Epoch	21
Batch Size	128

Table A.2. *ARIMA Model Selection Metrics for the 1954–2025 Period*

	MSE	RMSE	MAE	MAPE
Order		**Static ARIMA**		
(0, 1, 1)	0.00015208	0.01233211	0.00799743	0.00098050
(0, 1, 2)	0.00015208***	0.01233209***	0.00799744***	0.00098050***
(1, 1, 0)	0.00015338	0.01238472	0.00801280	0.00098245
(1, 1, 1)	0.00016122	0.01269742	0.00871297	0.00106690
(1, 1, 2)	0.00015613	0.01249537	0.00804921	0.00098702
(2, 1, 0)	0.00015502	0.01245063	0.00802557	0.00098410
(2, 1, 1)	0.00015265	0.01235519	0.00800325	0.00098124
(2, 1, 2)	0.00016590	0.01288030	0.00837684	0.00102692
		Dynamic ARIMA		
(0, 1, 1)	0.02470642	0.15718275	0.12369190	0.01485028
(0, 1, 2)	0.02479574	0.15746663	0.12396955	0.01488366
(1, 1, 0)	0.02460548***	0.15686136***	0.12339413***	0.01481462***
(1, 1, 1)	0.02490858	0.15782452	0.12428981	0.01492192
(1, 1, 2)	0.02479635	0.15746856	0.12397157	0.01488391
(2, 1, 0)	0.02470373	0.15717421	0.12369876	0.01485123
(2, 1, 1)	0.02474478	0.15730474	0.12381950	0.01486569
(2, 1, 2)	0.02482977	0.15757466	0.12407170	0.01489591

*Notes: *** is for marking the lowest value*

Table A.3. *Forecast Metrics for the 1954–2025 Period*

	LSTM	ARIMA(S)	ARIMA(D)	RW(S)	RW(D)
			Time-series Split		
MSE	0.00019155	*0.00015208	0.02460548	0.00015208	0.02596782
RMSE	0.01384011	*0.01233209	0.15686136	0.01233217	0.16114534
MAE	0.00969653	*0.00799744	0.12339413	0.00799768	0.12736519
MAPE	0.00119305	*0.00098050	0.01481462	0.00098053	0.01529034
			5-Fold Time-series Split		
MSE	0.08625883	*0.00010667	0.06771373	0.00010667	0.06711891
RMSE	0.27737885	*0.00998574	0.24337986	0.00998574	0.24283788
MAE	0.22211825	*0.00688704	0.20146166	0.00688706	0.20110182
MAPE	0.01050539	*0.00116477	0.03490350	0.00116477	0.03476184
			Blocked 5-Fold Time-series Split		
MSE	0.03264845	*0.00013377	0.06209968	0.00013377	0.06168423
RMSE	0.17078937	*0.01039695	0.21700633	0.01039678	0.21632889
MAE	0.13666492	*0.00745935	0.18797482	0.00745864	0.18741506
MAPE	0.00618227	*0.00120993	0.03048987	0.00120977	0.03041540

*Notes: (S) indicates forecasts using the static method; (D) indicates forecasts using the dynamic method. * is for marking the best performance*

Table A.4. *Performance Comparison of LSTM with ARIMA and RW Models for the 1954–2025 Period (Percentage)*

	ARIMA(S)	**RW(S)**	**ARIMA(D)**	**RW(D)**
			Time-series Split	
MSE	79,40	79,40	12845,55	13556,77
RMSE	89,10	89,10	1133,38	1164,34
MAE	82,48	82,48	1272,56	1313,51
MAPE	82,18	82,19	1241,74	1281,62
			5-Fold Time-series Split	
MSE	0,12	0,12	78,50	77,81
RMSE	3,60	3,60	87,74	87,55
MAE	3,10	3,10	90,70	90,54
MAPE	11,09	11,09	332,24	330,90
			Blocked 5-Fold Time-series Split	
MSE	0,41	0,41	190,21	188,93
RMSE	6,09	6,09	127,06	126,66
MAE	5,46	5,46	137,54	137,13
MAPE	19,57	19,57	493,18	491,98

*Notes: (S) indicates forecasts using the static method; (D) indicates forecasts using the dynamic method. * is for marking the best performance*