

A Hybrid Wavelet-LSTM Model for Forecasting NSPO Stock Prices: Structure, Methodology, and Empirical Application

Nooruldeen A. Noori^{1*}, Hiba H. Abdullah²

¹Anbar Education Directorate, Anbar, Iraq

²Department of Mathematics, College of Education for Women, Tikrit University, Tikrit, Iraq

Email: *nooruldeen.a.noori35508@st.tu.edu.iq

How to cite this paper: Noori, N.A. and Abdullah, H.H. (2026) A Hybrid Wavelet-LSTM Model for Forecasting NSPO Stock Prices: Structure, Methodology, and Empirical Application. *Journal of Applied Mathematics and Physics*, **14**, 3890-3913.
<https://doi.org/10.4236/jamp.2026.1410190>

Received: September 2, 2026

Accepted: October 11, 2026

Published: October 14, 2026

Copyright © 2026 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
<http://creativecommons.org/licenses/by/4.0/>



Open Access

Abstract

Financial time series forecasting is sensitive to nonlinear dynamics, abrupt shocks, and preprocessing choices that can contaminate out-of-sample evaluation. This study evaluates a hybrid Wavelet-Long Short-Term Memory (Wavelet-LSTM) model for weekly stock prices of Northern Upper Egypt Development Company (NSPO) under a leakage-safe chronological protocol. The dataset contains 319 weekly prices from 5 January 2020 to 8 March 2026, producing 318 logarithmic returns. The final 10 returns and corresponding prices were reserved exclusively for rolling one-step-ahead testing, while the preceding 20 returns formed a chronological validation segment and the first 288 returns formed the base training segment. At every test origin, the db4 level-3 discrete wavelet decomposition and min-max normalization were recomputed using only information available before the target observation. Hyperparameters were selected without using the final test period. The selected Wavelet-LSTM used 12 lags, 40 hidden units per component, dropout 0.20, 150 epochs, and an initial learning rate of 0.005. Across the 10 rolling test origins, the Wavelet-LSTM achieved return RMSE = 0.1757, return MAE = 0.0946, and directional accuracy = 50%. On the reconstructed price scale, RMSE = 0.1163, MAE = 0.0714, and MAPE = 8.40%. A matched standard LSTM achieved price RMSE = 0.1086 and MAPE = 8.11%. Thus, the leakage-safe evaluation does not support a forecasting-accuracy advantage for wavelet decomposition on this short terminal holdout, although the framework remains useful for multi-scale representation. The revised protocol provides a reproducible assessment that prevents future observations from entering decomposition, scaling, model selection, or forecasting.

Keywords

Wavelet Decomposition, LSTM, Stock Price Forecasting, Logarithmic Returns, Rolling-Origin Evaluation, Information Leakage, NSPO Stock Prices

1. Introduction

Financial markets are complex systems which are often sensitive to macroeconomic and structural changes, changes in the country's government, political events, investor expectations, uncertainty in the stock market, and new, external events. It is this kind of complexity that renders stock price forecasting a fundamental yet complex challenge for scholars, investors, analysts and decision-makers. Predicting the future accurately provides assets and liabilities of portfolios, helps in risk management and investment risk mitigation and investment management, and is important in market strategy. Financial time series exhibit nonlinear movement, irregularities, clustering and time structure. These characters result in that, the prediction of future price fluctuations is hard at best, since it's not only linearity that determines the future trend of stock prices as much as it is not deterministic trends of the past.

We have been employing classical statistical models widely in time series forecasting because they have straightforward mathematical form and can be easily interpreted. Models, including Autoregressive Integrated Moving Average (ARIMA), still help to represent linear dependence and short term temporal patterns. For some, however, such models are based on assumptions such as stationarity, linearity and regularity of error behavior. If the series includes nonlinear structures not visible to the model, abrupt price changes, and multifaceted temporal dependence, classical models also lose their predictive power. Zhang [1] demonstrated that the hybrid structure of ARIMA combined with neural networks achieves superior predictive performance over the non-linear-linear ARIMA hybrid structure. Hyndman and Athanasopoulos [2] also pointed by stressing that effective forecasting is in terms of pre-processed processing, model choice, model validation, accuracy validation, and not of choosing a structured model for all data sources.

AI and deep learning methods have gained acceptance in financial forecasting in recent years because they can learn nonlinear structures in the data directly. Long Short-Term Memory (LSTM) networks are one of these architectures and have attracted considerable interest in stock price predictions. The LSTM networks are a specific form of recurrent neural network created to model sequential dependency and to solve the vanishing gradient problem of classical recurrent networks. By utilizing memory cells and gating mechanisms, LSTM manages to store useful past observations while ignoring irrelevant signals. The latter feature renders LSTM applicable to financial time series, where short-term movements as

well as long-term trends can be responsible for current prices and returns.

Although the forecasting efficiency of LSTM is very high, the application of LSTM directly on raw financial data does not necessarily result in the best scenarios. In the financial time series the data are usually dominated with various mix of long-run movement, medium-frequency variations, local shocks, and high-frequency noise. In a case where a specific neural network is trained on the original series, this would need to be learned simultaneously. This can lead to poor prediction accuracy, particularly with an erratic volatility and abrupt changes in the series. Thus, decomposition-based hybrid models have emerged as a key research direction in time series forecasting. The key takeaway is to re-construct the final forecast from the predicted components, and learn a good model to use for that recordings from the new original series.

It has been reported in multiple recent works that decomposition approaches with neural networks add value to prediction. Chen et al. [3] used seasonal and trend decomposition under Loess based LSTM approach for short-term prediction of metro riderships and the results demonstrated that using decomposition can enhance learning before neural modeling. Aljboori [4] presented the STDR-MNN hybrid model which combines structural decomposition with modular neural-network. Sultana and Aljboori [5] proposed STL-FNN with seasonal decomposition and trend extraction incorporated into feedforward neural networks. The STR-ENN model by Othman and Aljboori [6] combines Seasonal-Trend Regression with an Elman neural network to model temporal memory. In recent works, Abdullah, Noori, and Hamza [7] introduced an improved hybrid STD-RBFNN model and implemented it on the prediction of Tesla stock price by presenting that adding dispersion component improves forecasting performance, when non-linear financial behavior is detected. Another viable method to update neural forecasting models is wavelet decomposition, as this method takes into account time and frequency dimensions of the time series. To the contrary to standard smoothing methodology, wavelet decomposition, does not treat the series in indivisible manner.

It does not, but instead breaks down the series into a low-frequency approximation element and multiple high-frequency detail elements. The approximation element is related to the net movement of the series; the detail element is the local variations, shocks and short term anomaly. This property makes wavelet decomposition an especially strong fitting method for financial data, where price movements can have both smooth long-term behavior and rapid short-term change. To this end, wavelet decomposition with LSTM combines wavelet decomposition, every decomposed constituent of the structure is taken through separate (independent) processing, where the neural network can then learn more uniform yet simpler patterns from the information. This study aims at building a cost-effective and reproducible hybrid framework that can be fully functional: both the wavelet decomposition and forecasting is performed directly on the stock prices of Northern Upper Egypt Development Company (NSPO). The hybrid decomposition-

neural network model has been well investigated, yet the practical applications still pay a significant focus on the global forecasting accuracy, and little attention to how the initial series is transformed, decomposed, component modeled and reconstructed. Additionally, the volatility of Egyptian stock market has its own features, and a Wavelet-LSTM based model for NSPO price prediction is an applied valuable feature in financial time series modeling.

This study evaluates whether wavelet decomposition adds forecasting value to an LSTM model when the empirical protocol strictly respects the chronological information set. The analysis transforms 319 weekly NSPO prices into 318 logarithmic returns, reserves the final 10 returns for untouched out-of-sample evaluation, and uses the immediately preceding 20 returns for chronological validation. The key methodological contribution is a leakage-safe expanding-window design: at each forecast origin, wavelet decomposition, normalization, lag construction, and model fitting use only observations available before the target week. The final test consists of 10 rolling one-step-ahead forecasts. After each forecast is evaluated, the newly realized return becomes available for the next origin. A standard LSTM is evaluated under the same lag order, hidden size, dropout, epochs, learning rate, normalization rule, and deterministic seed structure. This design allows the empirical comparison to focus on the effect of the decomposition-and-componentwise modeling structure rather than on unequal preprocessing or tuning. Accuracy is reported on both return and reconstructed-price scales, together with directional accuracy.

2. Theoretical Background of Forecasting and Hybrid Wavelet-LSTM Models

2.1. Classical Forecasting Models

Classical forecasting models provide an important foundation for time series analysis because they describe temporal dependence through explicit mathematical structures. Linear regression models and autoregressive models have been widely used in economic and financial forecasting because they are simple, interpretable, and computationally efficient. In a basic linear regression forecasting structure, the observed value of a financial series at time t can be represented as

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_p x_{p,t} + \varepsilon_t \quad (1)$$

where y_t is the response variable, $x_{1,t}, x_{2,t}, \dots, x_{p,t}$ are explanatory variables, $\beta_0, \beta_1, \dots, \beta_p$ are model parameters, and ε_t is the random error term.

This formulation assumes that the relation between the predictors and the response is linear. Therefore, it may become less effective when the financial series contains nonlinear dynamics, abrupt structural changes, volatility clustering, or hidden temporal interactions. Autoregressive models extend linear modeling by using previous values of the series as predictors. An autoregressive model of order p , denoted by $AR(p)$, is written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t \quad (2)$$

where c is a constant term, $\phi_1, \phi_2, \dots, \phi_p$ are autoregressive coefficients, and ε_t is a white-noise error term.

The model assumes the present value of the series is linearly related to the previous values. This assumption is feasible for stable and weakly nonlinear series. However, this assumption is restrictive for stock price series where local perturbations or nonlinear response from the market may affect the future directions. ARIMA(p, d, q), which stands for Autoregressive Integrated Moving Average model, is one of the most popular classical models for non-stationary time series. It integrates autoregressive, differencing, and moving-average error correction. The general form is

$$\phi(B)(1-B)^d y_t = \theta(B)\varepsilon_t \quad (3)$$

where B is the backshift operator such that $By_t = y_{t-1}$, d is the order of differencing, $\phi(B)$ is the autoregressive polynomial, and $\theta(B)$ is the moving-average polynomial. These polynomials are defined by

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p,$$

and

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \dots + \theta_q B^q.$$

While ARIMA models are very useful for most linear forecasting problems, they are also constrained when the series has complex nonlinear relations. Moreover, they need proper stationarity treatment and fine-grained ordering. Price movements within financial markets are influenced by nonlinear investor behavior, abrupt information disruptions, and volatility in market regimes. Therefore, the classical models might make poor predictions if the underlying structure cannot be captured by linear dependence alone. This constraint allowed to hybridize predictive models that combine the statistical preprocessing with neural learning, in the form of ARIMA-neural framework, hybridization architectures [1] and current model validation and data-driven prediction theories [2].

2.2. Deep Learning Models for Financial Time Series Forecasting

Deep learning algorithms have become widely adopted for financial forecasting because they can learn nonlinear temporal interactions dynamically and directly from data. While classical forms are rooted in an ordered framework between inputs and outputs, neural networks rely upon nonlinearity from the inputs to their outputs. A neural network estimates an indeterminate nonlinear $f(\cdot)$, meaning

$$\hat{y}_t = f(\mathbf{x}_t; \boldsymbol{\theta}) \quad (4)$$

where $\mathbf{x}_t$ is the input vector, $\boldsymbol{\theta}$ the trained parameters, and $\hat{y}_t$ represents the predicted value. For forecasting time series the input vector is typically built from lagged observations:

For time series forecasting, the input vector is usually constructed from lagged observations:

$$\mathbf{x}_t = (y_{t-1}, y_{t-2}, \dots, y_{t-p})^T \quad (5)$$

Recurrent Neural Networks (RNNs) was a general algorithm for modelling sequential data, which was able to update a hidden state through time. A standard RNN can be written as:

$$\mathbf{h}_t = g(W_x \mathbf{x}_t + W_h \mathbf{h}_{t-1} + \mathbf{b}_h) \quad (6)$$

and

$$\hat{y}_t = W_y \mathbf{h}_t + b_y \quad (7)$$

where $\mathbf{h}_t$ is the hidden state at the time t , W_x is the input weight matrix, W_h is the recurrent weight matrix, W_y is the output weight matrix, $\mathbf{b}_h$ the hidden bias vector, b_y is output bias, and $g(\cdot)$ is a nonlinear activation function.

But, standard RNNs tend to encounter the vanishing gradient issue during training and this is even more prevalent when the model needs to learn long-distance dependence. This problem negatively affects the ability of the network to be able to recall useful information from previous observations. To solve this issue, long short-term memory (LSTM) networks were proposed by employing memory cells and gates. An LSTM module takes care of information transfer through three gates: the forget gate, the input gate, and the output gate. For an input vector $\mathbf{x}_t$, previous hidden state $\mathbf{h}_{t-1}$, previous cell state $\mathbf{c}_{t-1}$; the LSTM equations are formulated by [8] [9]:

$$\begin{aligned} \mathbf{f}_t &= \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f), \\ \mathbf{i}_t &= \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i), \\ \tilde{\mathbf{c}}_t &= \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1} + \mathbf{b}_c), \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t, \\ \mathbf{o}_t &= \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o), \end{aligned} \quad (8)$$

and

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t).$$

Here, the forget gate, input gate, output gate, candidate cell state, and updated cell state control information flow through the LSTM. The logistic sigmoid and hyperbolic tangent functions provide the nonlinear gate transformations, and element-wise multiplication combines gate and state information. The final prediction follows the standard LSTM output mapping described in [8] [9]:

$$\hat{y}_t = W_y \mathbf{h}_t + b_y \quad (9)$$

These gates provide an index for LSTM network to control which data and memory or how it is passed to the output. This type of structure makes LSTM appropriate for financial prediction as stock prices movements can be influenced not only by short-run changes over the short-run, but also long-run patterns that are long-run. Hence, LSTM is much better suited to model temporal dependence than RNN and classical linear forecasting methods based on the RNN and classical linear forecasting approaches.

2.3. The Proposed Hybrid Wavelet-LSTM Forecasting Framework

LSTM networks can model nonlinear temporal dependence; however, a direct model must learn low-frequency movement, local oscillation, and short-term shocks simultaneously. Wavelet decomposition can separate these scales before neural learning. In forecasting applications, the decomposition must respect the information set available at each origin. A decomposition calculated once on the complete series can use values from the holdout period when forming coefficients near the training boundary and can therefore create information leakage.

The hybrid Wavelet-LSTM model integrates wavelet decomposition with a Long Short-Term Memory neural network. Wavelet decomposition separates the original time series into a smooth low-frequency approximation component and several high-frequency detail components. The LSTM network then learns the temporal dynamics of each component separately. After that, all predicted components are recombined to reconstruct the final forecasted series. This structure combines the ability of wavelet decomposition to extract multi-scale information with the ability of LSTM to learn nonlinear sequential dependence. In the revised empirical protocol, the wavelet operation follows the multi-resolution principles summarized in [10] [11], but it is recalculated separately at every forecast origin so that future observations cannot affect boundary coefficients. Neural-network learning principles are discussed in [12], while related applications include nonlinear time-series and RBFNN comparisons [13], statistical distribution modeling [14], conditional-variance stability forecasting [15], and seasonal autoregressive versus feed-forward neural-network forecasting [16].

Let y_t denote the original time series at time t . Under a wavelet decomposition of level k , the series can be represented as

$$y_t = A_t + D_{1,t} + D_{2,t} + \cdots + D_{k,t} \quad (10)$$

where A_t is the approximation component at time t , and $D_{j,t}$ is the j -th detail component at time t , for $j = 1, 2, \dots, k$.

The approximation component A_t describes the large-scale structure of the series and captures the smooth low-frequency movement. The detail components $D_{1,t}, D_{2,t}, \dots, D_{k,t}$ describe fine-scale oscillations, local shocks, and short-term irregularities. Each detail component captures variations in a specific frequency band determined by the selected decomposition level and mother wavelet.

In the empirical implementation, the model may use the Daubechies wavelet of order four and a third-level decomposition. In that case, the original series or transformed return series can be written as

$$y_t = A_{3,t} + D_{3,t} + D_{2,t} + D_{1,t} \quad (11)$$

where $A_{3,t}$ is the approximation component at level three, while $D_{3,t}$, $D_{2,t}$, and $D_{1,t}$ are the detail components at levels three, two, and one, respectively. The component $A_{3,t}$ captures the smooth low-frequency behavior, while $D_{1,t}$ usually contains the highest-frequency fluctuations.

For each decomposed component, lagged input-output pairs are constructed

before training. Let

$$C_{j,t} \in \{A_{3,t}, D_{3,t}, D_{2,t}, D_{1,t}\} \quad (12)$$

For a lag order p , the input vector for the j -th component is

$$\mathbf{x}_{j,t} = (C_{j,t-1}, C_{j,t-2}, \dots, C_{j,t-p})^T \quad (13)$$

and the target output is

$$y_{j,t} = C_{j,t}.$$

An independent LSTM model is then trained for each component:

$$\hat{C}_{j,t} = f_j(\mathbf{x}_{j,t}; \boldsymbol{\theta}_j) \quad (14)$$

where $f_j(\cdot)$ denotes the LSTM mapping for the j -th component and $\boldsymbol{\theta}_j$ is the corresponding parameter vector. Training is performed by minimizing the mean squared error for each component:

$$L_j(\boldsymbol{\theta}_j) = \frac{1}{N_j} \sum_{t=1}^{N_j} (C_{j,t} - \hat{C}_{j,t})^2 \quad (15)$$

After all components are forecasted, the final predicted series is reconstructed by summing the predicted components:

$$\hat{y}_t = \hat{A}_t + \hat{D}_{1,t} + \hat{D}_{2,t} + \dots + \hat{D}_{k,t} \quad (16)$$

For the third-level decomposition, this becomes

$$\hat{y}_t = \hat{A}_{3,t} + \hat{D}_{3,t} + \hat{D}_{2,t} + \hat{D}_{1,t} \quad (17)$$

If the model is applied to logarithmic returns, the predicted price can be recovered from the predicted return by

$$\hat{P}_t = P_{t-1} \exp(\hat{r}_t) \quad (18)$$

where P_{t-1} is the observed price at time $t-1$ and $\hat{r}_t$ is the predicted logarithmic return at time t .

The application procedure of the Wavelet-LSTM hybrid model can be summarized as follows.

Step 1: Convert the observed price series to logarithmic returns and define the chronological information set available at each forecast origin.

Step 2: Reserve the final 10 returns and their corresponding prices exclusively for final testing; do not use them for decomposition, normalization, tuning, or early model selection.

Step 3: Reserve the 20 returns immediately preceding the final test as a chronological validation segment. The earlier 288 returns form the base training segment used for hyperparameter selection.

$$y_t = A_t + D_{1,t} + D_{2,t} + \dots + D_{k,t}$$

At a generic origin t , let the available return history contain only observations up to $t-1$. The db4 level-3 decomposition is applied to that history, yielding one approximation component and three detail components. Because the target return

and all later observations are excluded, the component values at the training boundary cannot contain information from the future.

The approximation component represents the low-frequency structure, while the detail components describe progressively higher-frequency fluctuations. Their sum reconstructs the available return history under the selected decomposition rule.

Step 4: At every forecast origin, recompute the db4 decomposition at level 3 using only the expanding historical return window. Periodic boundary handling is used, with last-value extension when an intermediate decomposition length is odd.

Step 5: Fit the min-max normalization parameters separately to each component using the current historical component values only. Apply no scaling parameter estimated from validation or future test observations.

Step 6: Construct supervised sequences with lag order 12 from the normalized historical components.

Step 7: Train one LSTM for each wavelet component using the hyperparameters selected from the pre-test validation segment.

Step 8: Generate one-step-ahead forecasts for A3, D3, D2, and D1 at the current origin.

Step 9: Reconstruct the next logarithmic-return forecast by summing the four component forecasts:

$$\hat{A}_t, \hat{D}_{1,t}, \hat{D}_{2,t}, \dots, \hat{D}_{k,t}$$

Step 10: Convert the predicted return to a one-step-ahead price forecast using the observed price available immediately before the target week:

$$\hat{y}_t = \hat{A}_t + \hat{D}_{1,t} + \hat{D}_{2,t} + \dots + \hat{D}_{k,t}$$

Step 11: Score the forecast. After the target week is observed, append its realized return to the information set before moving to the next origin. Therefore, the ten holdout predictions are rolling one-step-ahead forecasts, not a ten-step recursive forecast, and no unobserved holdout return is used as an input.

Step 12: Select hyperparameters only from the chronological validation segment, then evaluate the frozen design on the final 10 rolling origins using return-scale MSE, RMSE, MAE, directional accuracy, and reconstructed-price MSE, RMSE, MAE, MAPE, and R^2 .

3. Materials and Data Description

This empirical application uses weekly stock prices of Northern Upper Egypt Development Company (NSPO) listed on the Egyptian Exchange. The observations were obtained from Investing.com for the period 5 January 2020 to 8 March 2026. The supplied series contains 319 positive price observations. Taking first differences of log prices produces 318 logarithmic returns. The final 10 returns and their corresponding 10 prices are reserved for the final rolling-origin test. Before that test, 308 returns are available: the first 288 returns form the base training block

and the next 20 returns form the chronological validation block. With lag order 12, the base training block yields 276 supervised sequences. At the first final test origin, the expanding history contains 308 returns and therefore 296 lagged training sequences; this increases to 305 sequences by the tenth origin as realized test returns become available one at a time. **Table 1** summarizes the dataset structure and chronological split used in the analysis.

Table 1. Description of the empirical dataset.

Item	Description
Company	Northern Upper Egypt Development Company
Symbol	NSPO
Market	Egyptian Exchange
Data source	Investing.com
Frequency	Weekly
Period	5 January 2020 to 8 March 2026
Price observations	319
Logarithmic returns	318
Base training block	288 returns; 276 sequences with lag 12
Chronological validation	20 returns immediately before final test
Final test	10 returns and 10 corresponding prices
Forecast design	10 rolling one-step-ahead expanding-window origins
Training sequences at test origins	296 at origin 1, increasing to 305 at origin 10

4. Application to Northern Upper Egypt Development Company (NSPO) Stock Prices

The empirical analysis begins with the original weekly NSPO price series. **Figure 1** is descriptive only and displays the complete observed sample after the forecasting experiment has been defined. It shows strong level changes, abrupt peaks, and heterogeneous volatility, which motivate the use of logarithmic returns and a non-linear forecasting framework. The full-sample descriptive plot is not used to estimate any decomposition or scaling parameter for the holdout forecasts.

Figure 1 depicts the behavior of the data without any transformations, decomposition, or modeling being carried out. This figure is also significant from a methodological point of view because it represents the series in terms of whether it is stationary or not and whether it indicates clear trends, price jumps, or distinct volatility phases. There were multiple price phases in the series; it doesn't ride a constant mean—**Figure 1**. Price movements are observed at the start of the sample, where prices range roughly between 2.2 and 2.5, until they enter a continuous upward phase that exceeds 3.0, followed by a downward movement that returns the series to lower levels. The regularization of this behavior occurs more than once and suggests that the series does not conform to a simple linear trend; in-

stead, it is accompanied by structural variation and shifts in the ascending or descending regimes. It also indicates the striking sharp jumps in prices in the middle and later parts of the series, especially around the weeks of the series where prices are higher than 5.0. They are peaks of strong activity, or of positive price shocks, followed by generally rapid decreases. This behavior suggests that the data show heterogeneous volatility, as the amount of variation varies across different periods.

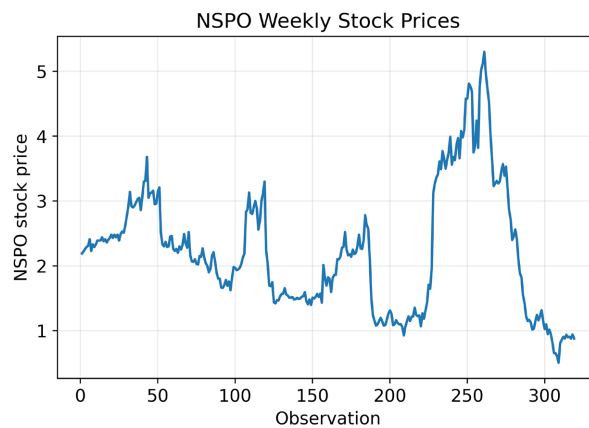


Figure 1. Time series plot of NSPO stock prices.

The price series is converted to logarithmic returns before forecasting. For consecutive prices P_t and P_{t-1} , the return is $r_t = \log(P_t/P_{t-1})$. This transformation produces 318 returns from 319 prices. **Figure 2** is again descriptive; the leakage-safe forecasting calculations use only the chronological subset available at each origin.

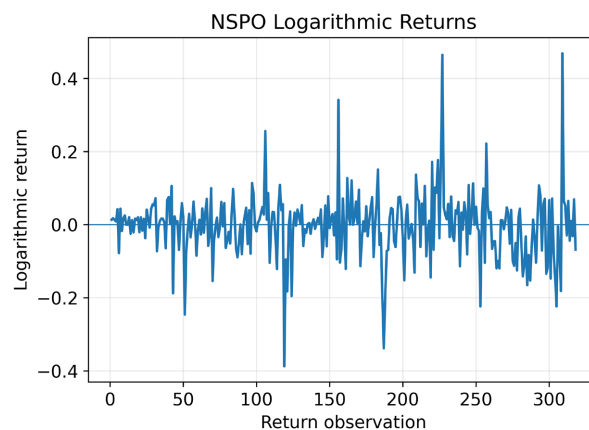


Figure 2. Time series plot of NSPO logarithmic returns.

Figure 2, which represents the time series of NSPO logarithmic returns, shows that the series fluctuates around zero, with positive and negative variations of different magnitudes. Sharp peaks also appear in some periods, indicating the pres-

ence of time shocks or sudden changes in returns. This feature is common in financial time series, since returns do not follow a smooth and stable pattern, but are instead affected by rapid and strong movements that appear as local jumps. Such behavior justifies the use of wavelet analysis, because it allows short-term fluctuations to be separated from the general low-frequency movement.

For model development, the Discrete Wavelet Transform uses the Daubechies wavelet of order four (db4) at level 3. The crucial revision is that the transform is not calculated once on the full return series. During validation and final testing, the DWT is recalculated from the return history available at each forecast origin.

Figure 3 shows the decomposition of the 288-return base training block only; it is therefore separated from the final validation and test observations.

$$r_t = A_{3,t} + D_{3,t} + D_{2,t} + D_{1,t} \quad (19)$$

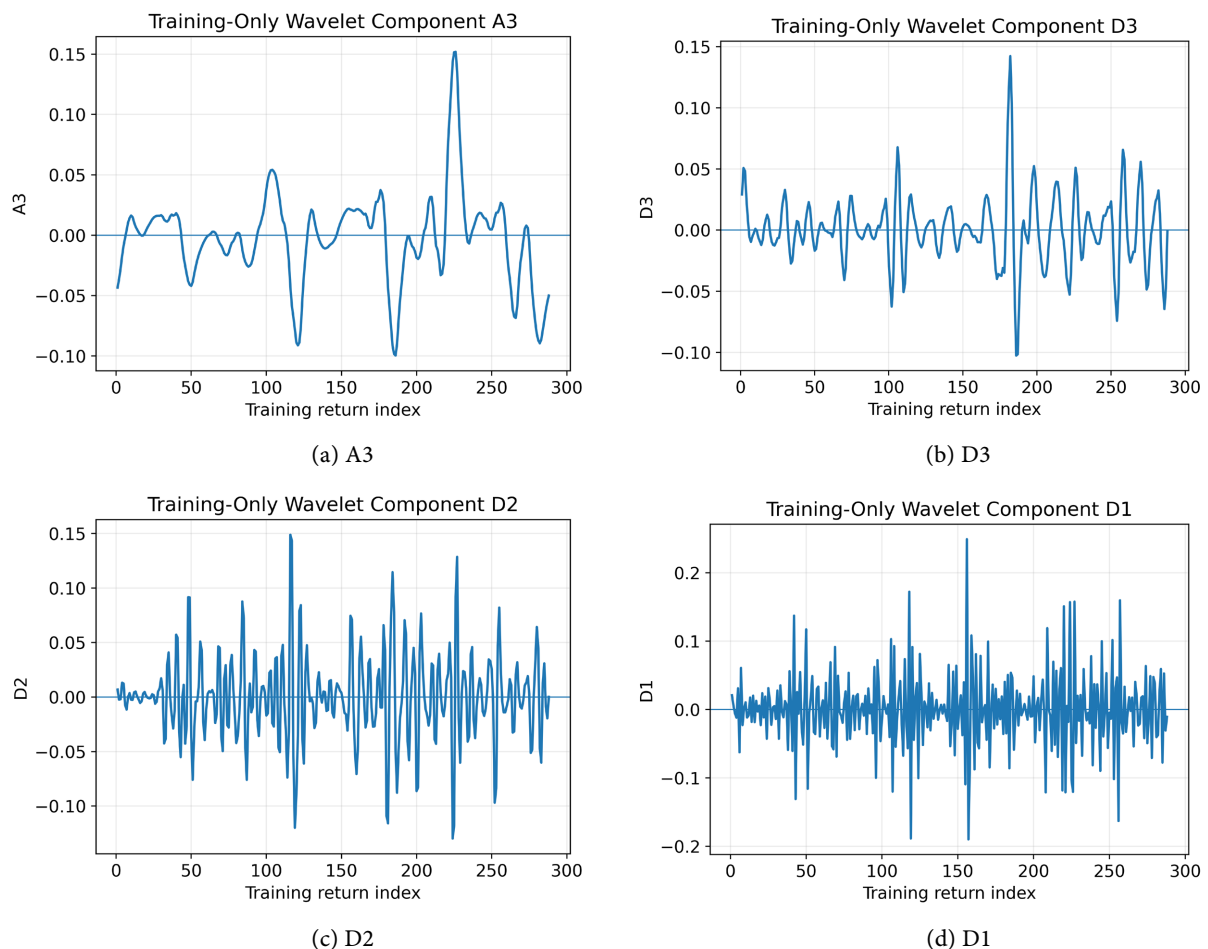


Figure 3. Training-only level-3 db4 decomposition: (a) approximation A3; (b) detail D3; (c) detail D2; and (d) detail D1.

The level-3 decomposition produces A3 together with D3, D2, and D1. A3 represents the smoother low-frequency structure; D3 and D2 represent intermediate fluctuations; and D1 contains the highest-frequency local variation under the selected level. The exact same decomposition rule is reapplied to the expanding his-

tory at each final forecast origin.

Leakage-safe forecasting pseudocode:

for $k = 1, \dots, 10$

H_k = returns available before test origin k

$[A3, D3, D2, D1] = \text{db4_level3_DWT}(H_k)$

fit component min-max scalers using H_k only

fit the component LSTMs and forecast one step ahead

$\hat{r}(k) = \hat{A3} + \hat{D3} + \hat{D2} + \hat{D1}$

$\hat{P}(k) = P_{\text{previous_observed}} \times \exp(\hat{r}(k))$

after scoring, append the realized return before origin $k+1$

end

This origin-specific decomposition prevents holdout values from influencing wavelet coefficients near the training boundary. It also makes the information set explicit: each test prediction is generated only from observations that would have been known at that time.

Figure 3 is based exclusively on the first 288 returns used for base training. The four panels demonstrate the distinct frequency behavior available to the componentwise model without using the validation or final test periods. This figure is a training-data diagnostic and does not constitute out-of-sample evidence.

At each forecast origin, one LSTM is fitted to each available wavelet component. Lagged input-output pairs are generated only after the component series has been decomposed and normalized from the current historical window. This ensures that both target construction and scaling respect chronology.

$$\mathbf{x}_{j,t} = (C_{j,t-1}, C_{j,t-2}, \dots, C_{j,t-12})^T.$$

The target is the next value of the corresponding component. Each component network minimizes mean squared error over the historical lagged sequences. Component forecasts are then transformed back to their original component scales before reconstruction.

$$L_j(\theta_j) = \frac{1}{N_j} \sum_{t=1}^{N_j} (C_{j,t} - \hat{C}_{j,t})^2.$$

Hyperparameters were selected on a chronological 20-return validation segment located immediately before the final 10-return test. Seven predefined candidates varied the decomposition level (2 or 3), lag order (8 or 12), hidden units (40 or 80), dropout (0.10 or 0.20), epochs (150 or 250), and initial learning rate (0.001 or 0.005). The selected configuration—level 3, lag 12, 40 hidden units, dropout 0.20, 150 epochs, and learning rate 0.005—achieved the lowest validation RMSE of 0.09945, with validation MAE 0.08636 and directional accuracy 65%. The alternative candidates produced validation RMSE values from 0.10052 to 0.10957. The final 10 test observations were not used in this selection. **Table 2** presents the reproducible configuration of the revised Wavelet-LSTM and the matched standard LSTM.

Table 2. Reproducible configuration of the revised Wavelet-LSTM and matched standard LSTM.

Parameter	Wavelet-LSTM	Standard LSTM
Lag order	12	12
Testing horizon	1 week at each of 10 origins	1 week at each of 10 origins
Hidden units	40 per component	40
Maximum epochs	150	150
Dropout rate	0.20	0.20
Initial learning rate	0.005	0.005
Training algorithm	Adam	Adam
Gradient threshold	1	1
Decomposition level	3	Not applied
Mother wavelet	db4	Not applied
Normalization	Min-max fitted at each origin from available history only	Same origin-specific rule
Validation	20 returns before final test	Same chronological validation period
Random seed	2026; deterministic component/origin offsets	2026; deterministic origin offsets
Implementation	Python 3/PyTorch; leakage-safe rerun	Python 3/PyTorch; leakage-safe rerun

Both models use the same one-dimensional sequence input, a single LSTM layer, dropout, a fully connected scalar output, and mean-squared-error regression training. The hybrid model fits this architecture separately to A3, D3, D2, and D1, whereas the standard LSTM fits one network directly to the undecomposed logarithmic-return history. Apart from the decomposition-and-componentwise structure, the lag order, hidden size, dropout, epochs, learning rate, optimizer, gradient threshold, validation period, normalization timing, and base random seed are matched.

Reproducibility specification: sequence input dimension = 1; LSTM hidden units = 40; dropout = 0.20; fully connected output dimension = 1; loss = MSE; optimizer = Adam; initial learning rate = 0.005; maximum epochs = 150; gradient clipping threshold = 1; base random seed = 2026. All preprocessing parameters are re-estimated from the historical window available at the relevant origin.

Figures 4-7 present in-sample diagnostics for the four LSTM component models fitted only to the 288-return base training block. They are not used to report final forecast accuracy.

Figure 4 shows that the LSTM follows the smoother A3 component within the base training block. Because this is an in-sample diagnostic, the closeness of the fitted and observed curves is interpreted only as evidence that the selected network can represent the low-frequency training pattern; it is not used as evidence of generalization.

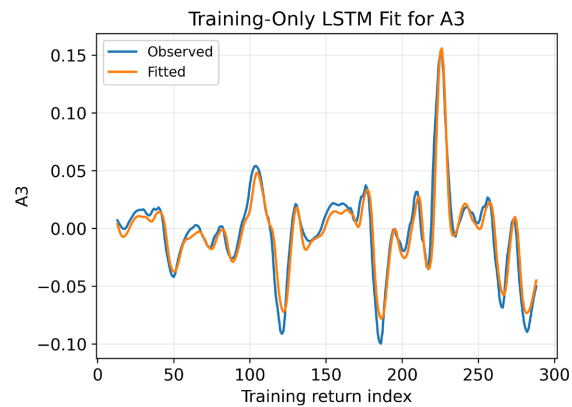


Figure 4. Training-only LSTM fit for approximation component A3.

Figure 5 and **Figure 6** show the corresponding training-only fits for D3 and D2. These detail components contain faster local variation, so larger discrepancies appear around abrupt oscillations. The plots are retained to document component behavior, while model selection and final claims are based on chronological validation and out-of-sample rolling forecasts rather than training fit.

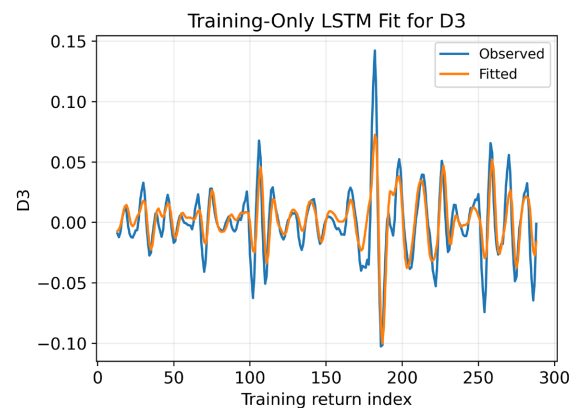


Figure 5. Training-only LSTM fit for detail component D3.

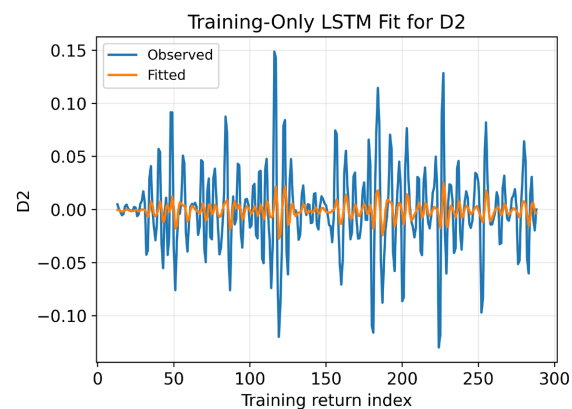


Figure 6. Training-only LSTM fit for detail component D2.

Figure 7 reports the training-only D1 fit. D1 contains the highest-frequency variation among the retained components, and its local fluctuations are more difficult to reproduce. This provides a descriptive explanation for why decomposition does not automatically guarantee improved forecast accuracy once future information is excluded.

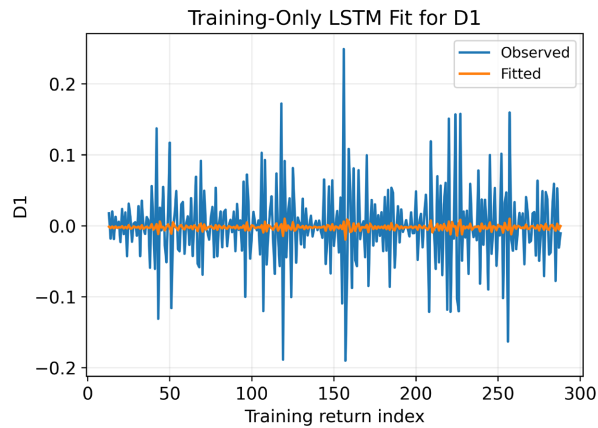


Figure 7. Training-only LSTM fit for detail component D1.

At each forecast origin, the four one-step component forecasts are summed to reconstruct the predicted logarithmic return:

$$\hat{r}_t = \hat{A}_{3,t} + \hat{D}_{3,t} + \hat{D}_{2,t} + \hat{D}_{1,t}.$$

The reconstructed return is converted to a one-step-ahead price forecast using the previous observed price:

$$\hat{P}_t = P_{t-1} \exp(\hat{r}_t).$$

The final evaluation does not use a ten-step recursive price path. Instead, the one-step reconstruction above is repeated at each of the ten rolling origins. Once the actual price and return for origin k are observed, they become available to construct origin $k + 1$. This distinction prevents the later holdout forecasts from being confused with a purely recursive multi-step experiment.

The cumulative recursive reconstruction shown in the earlier version is therefore not used in the revised final test.

Before opening the final 10-observation test, the selected hyperparameters were assessed on the chronological 20-return validation segment. **Figure 8** shows the recursive validation path for the Wavelet-LSTM and matched standard LSTM, while **Figure 9** displays the Wavelet-LSTM validation scatter.

The 20-return validation experiment was completed before the final holdout was evaluated. The selected Wavelet-LSTM produced validation RMSE = 0.09945, MAE = 0.08636, and directional accuracy = 65%. Under the same selected architecture, the standard LSTM produced validation RMSE = 0.10213, MAE = 0.08839, and directional accuracy = 60%. These values were used only to document model selection and were not combined with the final test metrics.

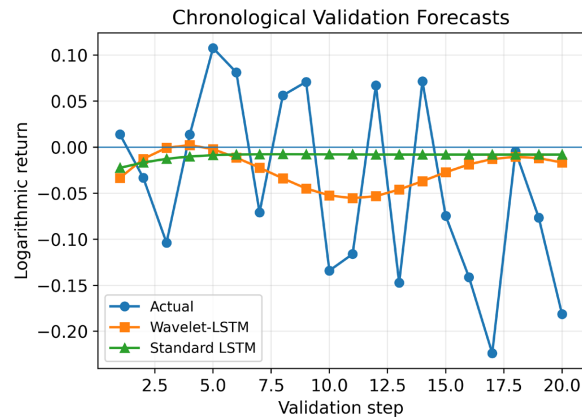


Figure 8. Chronological pre-test validation forecasts for Wavelet-LSTM and standard LSTM.

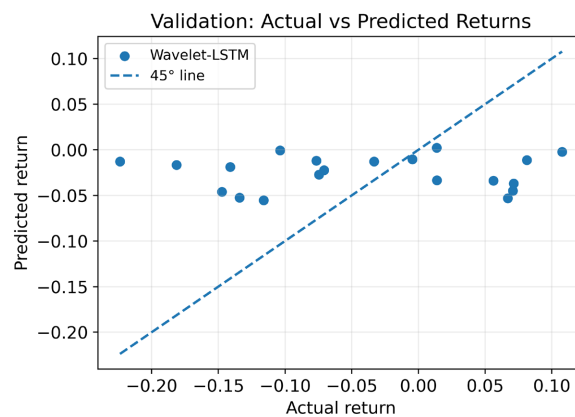


Figure 9. Validation actual versus Wavelet-LSTM predicted logarithmic returns.

Figure 9 shows that validation errors remain substantial around larger positive and negative returns. This prevents the in-sample component fits from being interpreted as near-perfect forecasting evidence and provides a more realistic basis for evaluating the terminal holdout.

The final 10 logarithmic returns and corresponding prices were kept untouched during hyperparameter selection. They were then evaluated at ten expanding rolling origins. At origin 1 the model uses the 308 returns available before the final test. After the first realized test return is observed, it is appended before origin 2, and this process continues through origin 10. **Table 3** reports the leakage-safe Wavelet-LSTM return forecasts.

The leakage-safe results are materially different from the earlier full-series implementation. The Wavelet-LSTM correctly predicts the sign of 5 of the 10 returns, giving directional accuracy of 50%. The largest error occurs at the first test origin, where the actual return is 0.469009 but the model forecasts -0.057161 . The remaining origins contain smaller changes, and the forecast magnitudes are correspondingly closer, but the model still does not reproduce every sign or shock.

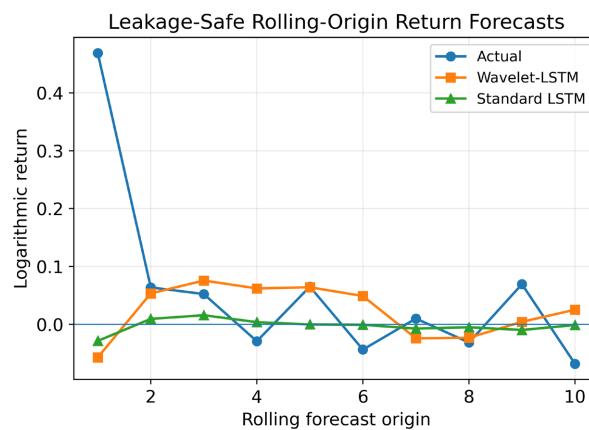
Table 3. Actual and leakage-safe Wavelet-LSTM logarithmic-return forecasts for the final 10 rolling origins.

Rolling origin	Actual log-return	Wavelet-LSTM forecast
1	0.469009	-0.057161
2	0.063839	0.053408
3	0.052285	0.075714
4	-0.029216	0.062047
5	0.065108	0.064071
6	-0.043675	0.048943
7	0.009995	-0.024215
8	-0.031428	-0.023117
9	0.069373	0.004308
10	-0.068233	0.025236

Figure 10 and **Figure 11** summarize the final return-scale evaluation. **Figure 10** compares actual returns with both leakage-safe models at all ten rolling origins. **Figure 11** focuses on the Wavelet-LSTM actual-versus-predicted relationship.

Figure 10 makes the first-origin shock explicit. Neither model anticipates the jump from the last pre-test price to the first test price, and this single event has a strong effect on aggregate error. Across all ten origins, the Wavelet-LSTM return RMSE is 0.17568 and MAE is 0.09460, while the standard LSTM return RMSE is 0.16459 and MAE is 0.09197.

Figure 11 confirms that the Wavelet-LSTM predictions are concentrated closer to zero than the most extreme realized returns. This behavior explains the limited directional accuracy and the negative return-scale R^2 value. The final assessment therefore relies on the complete set of reported errors rather than on visual training fit.

**Figure 10.** Leakage-safe rolling-origin logarithmic-return forecasts for the final 10 observations.

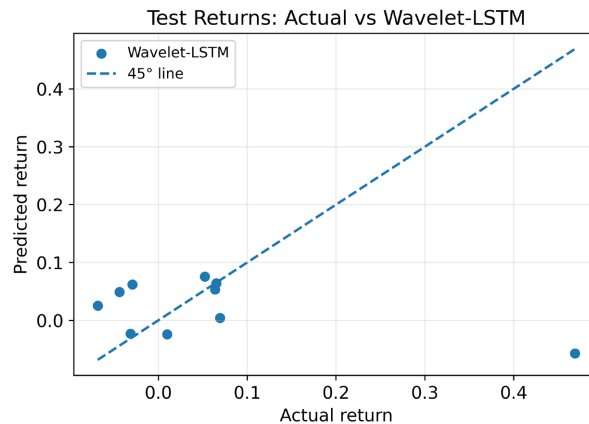


Figure 11. Actual versus Wavelet-LSTM predicted logarithmic returns in the final test.

On the price scale, the first test origin is again the dominant error source: the actual price is 0.804, while the Wavelet-LSTM forecast is 0.47505. This first-origin error contributes approximately 80.0% of the total squared price error over the ten-origin test. Over origins 2 - 10, the hybrid-model price RMSE is 0.0548, but the primary reported metrics retain all ten origins. **Figure 12** and **Figure 13** therefore display the complete final test without excluding the shock. **Table 4** reports the corresponding rolling-origin Wavelet-LSTM price forecasts.

Figure 12 shows that both models fail to anticipate the first large upward price shock but track the later price range more closely once the shock becomes part of the observed history. **Figure 13** shows substantial dispersion around the 45-degree line, particularly because of the first origin. These plots support a cautious interpretation: the revised experiment measures genuine one-step forecasting performance, but the short terminal test is strongly affected by one exceptional movement.

Table 4. Actual and leakage-safe Wavelet-LSTM NSPO price forecasts for the final 10 rolling origins.

Rolling origin	Actual price	Wavelet-LSTM forecast
1	0.804	0.47505
2	0.857	0.84811
3	0.903	0.92441
4	0.877	0.96080
5	0.936	0.93503
6	0.896	0.98295
7	0.905	0.87456
8	0.877	0.88432
9	0.940	0.88079
10	0.878	0.96402

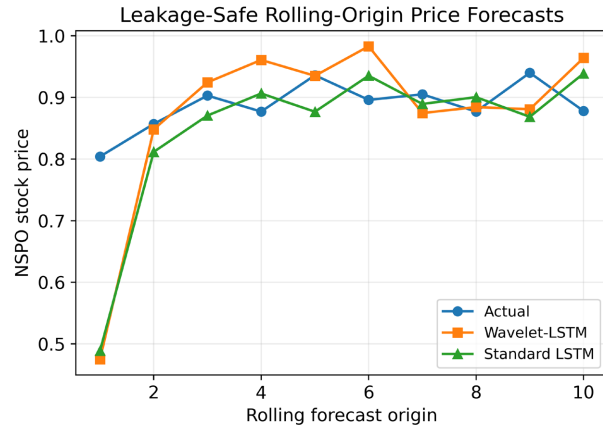


Figure 12. Leakage-safe rolling-origin NSPO price forecasts for Wavelet-LSTM and standard LSTM.

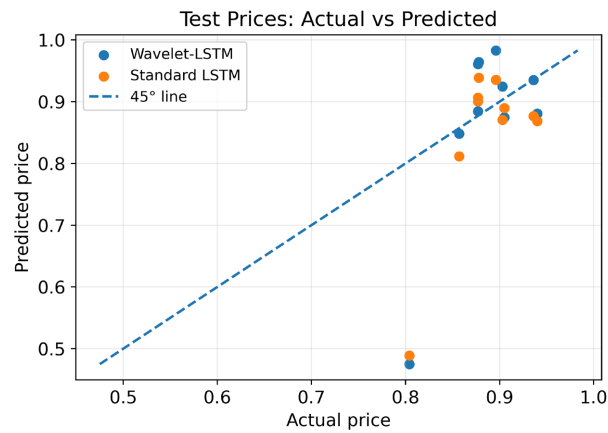


Figure 13. Actual versus predicted NSPO prices for both leakage-safe models.

Forecast accuracy is reported on both the logarithmic-return and reconstructed-price scales. Return MAPE is not emphasized because several realized returns are close to zero, which makes percentage errors unstable. Directional accuracy is reported directly on the return scale. **Table 5** summarizes the leakage-safe Wavelet-LSTM results.

Table 5. Leakage-safe Wavelet-LSTM forecast accuracy at return and price scales.

Metric	Return scale	Price scale
MSE	0.030863	0.013521
RMSE	0.175679	0.116281
MAE	0.094600	0.071396
MAPE	Not emphasized	8.3980%
R^2	-0.446904	-8.699560
Directional accuracy	50.0%	—

For the Wavelet-LSTM, return RMSE = 0.17568, return MAE = 0.09460, and directional accuracy = 50%. On the price scale, RMSE = 0.11628, MAE = 0.07140, and MAPE = 8.3980%. The negative R^2 values show that, over this short and shock-dominated terminal period, the forecasts do not outperform a constant-mean reference in squared-error terms. These results are less favorable than the earlier report, but they are based on a stricter chronological information set and therefore provide a more defensible out-of-sample assessment.

The matched comparison does not reproduce the previously reported 91.51% RMSE reduction. Under the leakage-safe rolling-origin design, the standard LSTM achieves return RMSE = 0.16459 and price RMSE = 0.10862, compared with 0.17568 and 0.11628 for the Wavelet-LSTM. Price MAPE is 8.108% for the standard LSTM and 8.398% for the Wavelet-LSTM, while both models obtain 50% directional accuracy. The relative Wavelet-LSTM RMSE improvement, defined with the standard LSTM as the reference, is therefore negative: **Table 6** reports the standard-LSTM price forecasts, and **Table 7** summarizes the matched rolling-origin accuracy comparison.

Table 6. Actual and leakage-safe standard-LSTM NSPO price forecasts for the final 10 rolling origins.

Rolling origin	Actual price	Standard LSTM forecast
1	0.804	0.48879
2	0.857	0.81161
3	0.903	0.87056
4	0.877	0.90641
5	0.936	0.87677
6	0.896	0.93521
7	0.905	0.88939
8	0.877	0.90043
9	0.940	0.86848
10	0.878	0.93883

Table 7. Leakage-safe rolling-origin accuracy comparison between Wavelet-LSTM and matched standard LSTM.

Model	Return RMSE	Return MAE	Direction	Price RMSE	Price MAE	Price MAPE	Price R^2
Wavelet-LSTM	0.17568	0.09460	50.0%	0.11628	0.07140	8.398%	-8.6996
Standard LSTM	0.16459	0.09197	50.0%	0.10862	0.06923	8.108%	-7.4638

$$\text{Improvement}(\%) = \frac{\text{RMSE}_{\text{Standard LSTM}} - \text{RMSE}_{\text{Wavelet-LSTM}}}{\text{RMSE}_{\text{Standard LSTM}}} \times 100.$$

Using the leakage-safe price RMSE values:

$$\text{Improvement}(\%) = \frac{0.10862 - 0.11628}{0.10862} \times 100 = -7.05\%.$$

The relative value is -7.05% , meaning that the Wavelet-LSTM price RMSE is approximately 7.05% higher than the matched standard-LSTM RMSE in this final test. Consequently, the revised evidence does not attribute an accuracy gain to wavelet decomposition for this particular terminal holdout.

The wavelet stage still provides an interpretable multi-scale representation by separating the return history into lower- and higher-frequency components, but representational separation alone does not guarantee superior forecasting. The first holdout observation contains a very large positive return that neither model anticipates, and the hybrid model also produces larger errors at several later origins. The empirical contribution should therefore be interpreted as a leakage-safe comparison of modeling structures rather than as proof that decomposition universally improves LSTM forecasts.

The revised empirical procedure has four chronological stages. First, 319 prices are converted to 318 logarithmic returns. Second, the first 288 returns are used as base training data and the next 20 returns are reserved for chronological validation. Third, the selected configuration uses db4 level 3, lag 12, 40 hidden units per component, dropout 0.20, 150 epochs, and learning rate 0.005, with all decomposition and normalization recalculated from available history only. Fourth, the final 10 returns are evaluated through expanding rolling one-step-ahead origins and compared with a matched standard LSTM. The final test is reported at both return and price scales, with directional accuracy included to assess sign prediction.

5. Conclusion

This study presents a hybrid Wavelet-LSTM framework for forecasting NSPO stock prices and, in the revised analysis, evaluates it under a leakage-safe chronological protocol. The supplied 319 weekly prices generate 318 logarithmic returns. The final 10 observations are kept exclusively for rolling one-step-ahead testing, while a preceding 20-return block is used for validation. At every forecast origin, db4 level-3 decomposition and min-max normalization are recalculated using only the return history available before the target week. The selected Wavelet-LSTM uses lag 12, 40 hidden units per component, dropout 0.20, 150 epochs, and learning rate 0.005. In the final test, it achieves return RMSE = 0.1757, return MAE = 0.0946, directional accuracy = 50%, price RMSE = 0.1163, price MAE = 0.0714, and price MAPE = 8.40%. A matched standard LSTM achieves lower error in the same experiment, with price RMSE = 0.1086 and MAPE = 8.11%. The earlier claim of a 91.51% RMSE reduction is therefore not retained. The main empirical lesson is that wavelet decomposition can organize multi-scale information, but its forecasting value must be established under causal preprocessing and repeated chronological evaluation rather than inferred from full-series decomposition or in-sample fit. Future research should evaluate longer rolling-origin sequences, alternative wavelet boundary rules, broader hyperparameter grids, and additional assets before drawing general conclusions about relative forecasting performance.

Author Contributions

Nooruldeen A. Noori: Methodology, software development and programming, formal analysis, interpretation of results, and writing of the original manuscript. Hiba H. Abdullah: Supervision, critical review of the manuscript, and editing. Both authors have reviewed and approved the final version of the manuscript and agree to be accountable for their respective contributions.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

- [1] Zhang, G.P. (2003) Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model. *Neurocomputing*, **50**, 159-175.
[https://doi.org/10.1016/s0925-2312\(01\)00702-0](https://doi.org/10.1016/s0925-2312(01)00702-0)
- [2] Hyndman, R.J. and Athanasopoulos, G. (2018) Forecasting: Principles and Practice. OTexts.
- [3] Chen, D., Zhang, J. and Jiang, S. (2020) Forecasting the Short-Term Metro Ridership with Seasonal and Trend Decomposition Using Loess and LSTM Neural Networks. *IEEE Access*, **8**, 91181-91187. <https://doi.org/10.1109/access.2020.2995044>
- [4] Aljboori, N.S.K. (2023) Structural Structure of the STDR-MNN Hybrid Model with Application. *Advances in the Theory of Nonlinear Analysis and Its Application*, **7**, 89-98.
- [5] Sultana, Z.A. and Aljbooria, N.S.K. (2024) Seasonal Decomposition and Trend Using Hybrid STL-FNN with Application. *Advances in the Theory of Nonlinear Analysis and Its Application*, **7**, 35-46.
- [6] Othman, M.L. and Aljboori, N.S.K. (2025) The Hybrid STR-ENN Model: A Study with Application. In: Pal, S. and Rocha, Á., Eds., *Proceedings of 4th International Conference on Mathematical Modeling and Computational Science*, Springer, 104-117. https://doi.org/10.1007/978-3-031-91008-1_10
- [7] Abdullah, H.H., Noori, N.A. and Hamza, T.S. (2026) The Improved Hybrid STD-Radial Basis Function Neural Network Approach for Time Series Forecasting Application to Tesla Stock Price Prediction. *F1000Research*, **15**, Article 286.
<https://doi.org/10.12688/f1000research.172498.2>
- [8] Yu, Y., Si, X., Hu, C. and Zhang, J. (2019) A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, **31**, 1235-1270.
https://doi.org/10.1162/neco_a_01199
- [9] Yunita, A., Pratama, M.I., Almuzakki, M.Z., Ramadhan, H., Akhir, E.A.P., Firdausiah Mansur, A.B., *et al.* (2025) Performance Analysis of Neural Network Architectures for Time Series Forecasting: A Comparative Study of RNN, LSTM, GRU, and Hybrid Models. *MethodsX*, **15**, Article ID: 103462.
<https://doi.org/10.1016/j.mex.2025.103462>
- [10] Elias, I.I. and Ali, T.H. (2025) Optimal Level and Order of the Coiflets Wavelet in the VAR Time Series Denoise Analysis. *Frontiers in Applied Mathematics and Statistics*, **11**, Article 1526540. <https://doi.org/10.3389/fams.2025.1526540>
- [11] Shuvo, S.B., Alam, S.S., Ayman, S.U., Chakma, A., Salvi, M., Seoni, S., *et al.* (2025) Application of Wavelet Transformation and Artificial Intelligence Techniques in

- Healthcare: A Systemic Review. *WIREs Data Mining and Knowledge Discovery*, **15**, e70007. <https://doi.org/10.1002/widm.70007>
- [12] Haykin, S. (2009) *Neural Networks and Learning Machines*, 3/E. Pearson Education India.
 - [13] Abdullah, H.H., Khalaf, N.S. and Noori, N.A. (2024) Comparison of Non-Linear Time Series Models (β -T-EGARCH and NARMAX Models) with Radial Basis Function Neural Network Using real Data. *Iraqi Journal for Computer Science and Mathematics*, **5**, Article No. 38. <https://doi.org/10.52866/ijcsm.2024.05.03.003>
 - [14] Noori, N.A. (2023) Exploring the Properties, Simulation, and Applications of the Odd Burr XII Gompertz Distribution. *Advances in the Theory of Nonlinear Analysis and Its Application*, **7**, 60-75.
 - [15] Abdullah, H.H., Aljboori, N.S.K. and Noori, N.A. (2025) Stability of Conditional Variance and Stability Forecasts for Natural Gas Contracts Using NAGARCH Models. *Mathematical Modelling of Engineering Problems*, **12**, 1551-1561. <https://doi.org/10.18280/mmep.120509>
 - [16] Ibrahim, M.Q., Mohammed, A.A., Khalaf, A.A. and Noori, N.A. (2025) Comparative Analysis of Seasonal Mixed Integrated Autoregressive Moving Average and Feed-Forward Neural Networks Models in Predicting United State Natural Hazard Casualties. *Baghdad Science Journal*, **22**, 2360-2374. <https://doi.org/10.21123/2411-7986.5002>