



(RESEARCH ARTICLE)



PREDICTING CLOSING STOCK RETURNS USING SEQUENTIAL ARTIFICIAL NEURAL NETWORK

James Asumbo Otieno *, Dennis Cheruiyot Kiplangat and Simon Maina Mundia

Department of Statistics and Actuarial Science, Dedan Kimathi University of Technology, Nyeri, Kenya.

* Corresponding Author

ORCID Details

Dennis Cheruiyot Kiplangat: <https://orcid.org/0000-0002-7545-3894>

Simon Maina Mundia: <https://orcid.org/0009-0002-1089-6392>

International Journal of Science and Research Archive, 2026, 21(01), 101–117

Publication history: Received on 25 August 2026; revised on 30 September 2026; accepted on 02 October 2026

Article DOI: <https://doi.org/10.30574/ijrsra.2026.21.1.1845>

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0

ABSTRACT

Financial time series exhibit complex characteristics, including volatility, nonlinearity, and temporal dependencies, which make accurate forecasting challenging. This study investigated the use of a dynamic Long Short-Term Memory (LSTM) neural network to forecast stock prices and market direction. Augmented Dickey-Fuller tests confirmed stationarity of the return series, while model performance was evaluated using RMSE, MAPE, Directional Accuracy, and Theil's U-statistic against a naive forecasting benchmark. The model achieved low forecasting errors, with Apple and Microsoft recording RMSE values of 4.56 and 5.34 and MAPE values of 2.34% and 2.87%, respectively. Nvidia and Tesla recorded MAPE values of 2.98% and 4.12%. Directional Accuracy ranged from 56.7% to 62.1%, while Theil's U-statistic ranged from 0.67 to 0.85, indicating improved performance relative to the naive benchmark. Diebold-Mariano tests also indicated superior predictive performance compared with selected traditional and static neural-network models. Residual diagnostics showed fat-tailed errors but no significant residual serial correlation. The findings indicate that dynamic LSTM networks can effectively capture nonlinear temporal dependencies in financial time series and have potential applications in financial forecasting, algorithmic trading and portfolio risk management.

Keywords: Stock Return Forecasting; Financial Time Series; Dynamic Long Short-Term Memory (DLSTM); Sequential Neural Networks; Deep Learning; Time-Series Prediction; Directional Accuracy; Algorithmic Trading.

1 INTRODUCTION

Stock markets are inherently difficult to predict because stock prices continuously respond to multiple and interrelated factors, including company performance, investor expectations, macroeconomic conditions, political events, and market sentiment. These factors do not operate independently; rather, they interact dynamically and may change rapidly, resulting in substantial volatility and abrupt price movements. Reliable forecasting of stock prices and returns is therefore important for investment decision-making, portfolio allocation, trading strategy development, and financial risk management. However, financial time series remain challenging to model because they are noisy, volatile, nonlinear, and dependent on their historical sequence.

Traditional statistical and econometric models, including Autoregressive (AR), Autoregressive Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA), and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models, have been widely applied to financial forecasting. These models provide useful frameworks for analysing temporal dependence and volatility. Nevertheless, their effectiveness may be constrained by assumptions of linearity, stable relationships, and particular distributional properties, especially when financial markets exhibit volatility clustering, structural breaks, fat-tailed returns, and abrupt changes in investor behaviour. AR and ARMA models, for example, primarily rely on historical observations and may not adequately account for the influence of external factors such as macroeconomic conditions, corporate information, geopolitical events, and market sentiment. Similarly, conventional statistical models may have difficulty adapting to changing market regimes in which the underlying relationships between variables evolve over time.

Stock return prediction has consequently remained an active area of research, with fundamental and technical analysis representing two established approaches. Fundamental analysis focuses on the intrinsic or underlying value of a company by examining factors such as earnings, debt, cash flows, growth prospects, and broader economic conditions. It is commonly associated with longer-term investment analysis because changes in fundamental value may take time to be reflected in market prices. Technical analysis, in contrast, focuses primarily on historical price and trading patterns and uses indicators such as moving averages, price trends, and trading volumes to identify potential short-term movements. Although both approaches provide valuable information for interpreting financial markets, their predictive effectiveness may be affected by the rapidly changing and nonlinear nature of market behaviour [6,7].

The limitations of conventional approaches have contributed to a shift toward machine learning and Artificial Neural Networks (ANNs) in financial forecasting. Early connectionist models, including feedforward neural networks and Multi-Layer Perceptrons (MLPs), demonstrated the ability to model complex nonlinear relationships without imposing the rigid functional assumptions associated with many traditional statistical models. However, conventional ANNs have an important limitation when applied to financial time-series forecasting: they are essentially static, memoryless architectures. When historical observations are treated primarily as independent input patterns, the model may fail to adequately preserve the chronological relationships and long-term dependencies that characterize financial data. Since the value of a financial asset at a particular time is related to its preceding sequence of observations, explicitly modelling temporal dependence is important for effective forecasting.

Recurrent Neural Networks (RNNs) were developed to address this limitation by incorporating information from previous observations into subsequent predictions. However, conventional RNNs may experience difficulties in learning long-term dependencies because of problems such as vanishing and exploding gradients. Long Short-Term Memory (LSTM) networks address these limitations through memory cells and gating mechanisms that regulate the retention and updating of information over time. Consequently, LSTM networks are particularly suitable for sequential and time-series applications in which historical observations may influence future outcomes over different time horizons.

In financial forecasting, LSTM models can learn nonlinear temporal relationships from sequences of historical market observations and potentially capture patterns that are difficult to represent using conventional linear models. Stock prices and returns may be influenced by economic indicators, market trends, company-specific information, and other time-dependent factors. A sequence-based neural architecture can therefore provide a flexible framework for modelling these complex relationships [3]. Nevertheless, a fixed LSTM architecture may lose predictive effectiveness when market characteristics change significantly. Financial markets are dynamic systems in which volatility, trends, and dependencies can vary across time, creating a need for forecasting approaches that can adapt to changing temporal patterns.

Against this background, this study developed and evaluated a Dynamic Long Short-Term Memory (Dynamic LSTM) model for forecasting daily stock returns and closing prices. The study assessed the statistical adequacy of the financial time series, evaluated the model's out-of-sample forecasting performance, and compared its predictive performance with naive and conventional linear benchmark models. The study therefore sought to establish whether a dynamic sequence-aware architecture could provide an effective framework for modelling the nonlinear and time-dependent characteristics of financial market data.

2 METHODOLOGY

This section presents the methodology adopted in the study. Section 2.1 introduces the Dynamic Long Short-Term Memory (Dynamic LSTM) architecture. Section 2.1.1 describes the operationalization strategy, while Section 2.1.2 explains the Dynamic LSTM gating mechanism. Section 2.1.3 presents the architecture optimization procedure, and Section 2.1.4 details the optimization of the Dynamic LSTM hidden layers. Section 2.2 assesses the adequacy of the proposed architecture, while Section 2.3 describes the procedure for predicting stock returns using the Dynamic LSTM

model. Finally, Section 2.4 presents the performance evaluation metrics used to assess the predictive capability of the proposed model.

2.1 Dynamic Long Short-Term Memory Architecture

The Dynamic Long Short-Term Memory (DLSTM) network is an advanced recurrent neural architecture specifically designed to capture long-range temporal dependencies in complex, nonlinear, and highly volatile financial time series. Conventional recurrent neural networks (RNNs) often encounter vanishing and exploding gradient problems when learning from extended sequences, limiting their ability to retain relevant historical information. The DLSTM architecture overcomes these limitations through a system of gated memory cells that regulate the flow of information across time. These gating mechanisms enable the network to selectively retain, update, and discard information, thereby enhancing its ability to model intricate temporal patterns and evolving market dynamics.

At any given daily trading interval t , the internal operations and mathematical transformations within a DLSTM cell are governed by the following sequence of operations: The forget gate evaluates the current market input (x_t) and the previous hidden state (h_{t-1}) through a sigmoid (σ) activation function to determine what percentage of historical information should be discarded or preserved from the past cell memory:

$$f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f) \quad (1)$$

The input gate acts as a quantitative filter, deciding which fresh market features, technical indicators, or structural elements will be integrated into the model's updated memory:

$$i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i) \quad (2)$$

An intermediate candidate state ($\tilde{C}_t$) is simultaneously generated via a hyperbolic tangent ($\tanh$) activation layer to introduce fresh candidate values to the sequence:

$$C_t = \tanh(W_C \times [h_{t-1}, x_t] + b_C) \quad (3)$$

The current internal cell memory (C_t) is subsequently computed by combining the filtered historical memory (scaled by the forget gate) with the newly extracted candidate features (scaled by the input gate):

$$C_t = f_t \odot C_{t-1} + i_t \odot C_t \quad (4)$$

Finally, the output gate regulates the emission of the next hidden state (h_t), isolating relevant localized features to be passed to either the subsequent sequential layer or the final predictive layer:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t \odot \tanh(C_t) \quad (6)$$

where: $x_t \in \mathbb{R}^d$ denotes the input feature vector of daily market metrics at time step t , h_t and h_{t-1} represent the current and preceding hidden vectors, C_t and C_{t-1} denote the current and past cell memory states, W and b symbolize the learnable weight matrices and bias vectors across the respective gates, σ represents the sigmoid activation function, $\tanh$ represents the hyperbolic tangent function and $\odot$ represents element-wise (Hadamard) multiplication.

2.2 Operationalization Strategy

To operationalize this architecture for financial forecasting, the study implemented a structured data engineering and optimization pipeline. Raw daily transaction features including Open, High, Low, Close, and Volume metrics were transformed into multidimensional temporal tensors. Using a look-back sliding window of 20 trading days (x_t to x_{t+19}), the model captures the short-term historical context needed to predict the asset's closing price on the subsequent 21st trading day. The deep learning architecture consists of stacked hidden LSTM layers designed to isolate complex, hierarchical temporal patterns. To mitigate overfitting during training, a 20% dropout layer was systematically integrated after each recurrent layer. The final output passes through a fully connected dense layer with a linear activation function to yield the predicted scalar closing price. To move away from arbitrary trial-and-error configuration, this study utilized a Grid Search Algorithm to systematically optimize hyperparameters. The GSA evaluates variations in hidden layer counts, neuron densities specifically testing sizes such as 32, 64, 128, and 256, look-back windows and dropout rates to isolate the specific network topology that minimizes the Mean Squared Error loss on validation data. The final model is evaluated using an 80/20 train-test split, trained over 50 epochs, and optimized using the Adam optimization framework.

2.3 Dynamic LSTM Gating

The multi-dimensional temporal tensor $X_{t-m:t-1}$ maps directly into the specialized recurrent hidden layer framework. For each discrete historical time step t within the 20-day look-back horizon ($t = 1, 2, \dots, 20$), the specific input value X_t is processed through the DLSTM cell configuration. This network maintains an internal hidden state vector (h_t) and a continuous cell state memory vector (C_t) that dynamically balance long-term trends against immediate market shocks. The mathematical execution mapping the input data vector through the model's internal gating mechanics is defined as:

$$f_t = \sigma(W_f \times [h_{t-1}, X_t] + b_f) \quad (7)$$

$$i_t = \sigma(W_i \times [h_{t-1}, X_t] + b_i) \quad (8)$$

$$\tilde{C}_t = \tanh(W_c \times [h_{t-1}, X_t] + b_c) \quad (9)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (10)$$

$$o_t = \sigma(W_o \times [h_{t-1}, X_t] + b_o) \quad (11)$$

$$h_t = o_t \odot \tanh(C_t) \quad (12)$$

where: f_t, i_t , and o_t , represents internal Forget, Input, and Output structural gates, respectively, which control the flow of historical market information. $\tilde{C}_t$: The candidate cell memory state encapsulating new pricing anomalies discovered on trading day t . C_t : The consolidated cell state vector tracking the persisting historical memory profile of the stock series. W and b : The optimized weight matrices and bias vectors systematically adjusted during the Grid Search optimization sweep. σ and $\tanh$: The non-linear Sigmoid and Hyperbolic Tangent activation functions that regulate numerical scaling. $\odot$: The Hadamard element-wise matrix multiplication operator.

At the final historical timestep ($t = 20$), the hidden vector h_{20} passes directly through a dense linear layer to calculate the ultimate out-of-sample normalized scalar prediction:

$$\hat{X}^* = W_d \times h_{20} + b_d \quad (13)$$

2.4 Architecture Optimization

The Input Layer received the 3D input sequence, two LSTM layers used to capture temporal dependencies. The study used multilayer to avoid underperformance caused by single layer. A 20 % drop out layer was applied after each LSTM layer to prevent overfitting. A Dense Output Layer which is final fully connected layer with a linear activation function was used to produce the single predicted price value. Mean Squared Error and Root Mean Squared Error was used to generate the loss function. The Adam optimizer was used to train the LSTM model.

2.5 Optimization of Dynamic LSTM Hidden Layers

A hidden layer in a neural network is a group of chambers that perform computations on the input or previous hidden layer. A unit is a single cell that has a memory state and a gate mechanism that controls the flow of information into the network. The number of hidden layers and units determines the complexity and capacity of the Long Short Term Memory model.

Coming up with the right number of hidden layers and units can have a significant effect on the performance, efficiency and accuracy of the LSTM model. If the hidden layers and units are too few, the model's ability to learn complex patterns and capture long-term dependencies might be limited. On the other hand, too many hidden layers and units may cause over-fitting, high computational cost, and slow convergence. It was therefore mindful to strike a balance between under-fitting and over-fitting, at the same time maintaining the required efficiency and accuracy. There is a need then to come up with an optimal number of hidden layers while conducting prediction using LSTM model. There is no definitive answer to how many hidden layers to be used for LSTM model. The number of layers to be used depends on multiple factors, such as the type, length, and variability of your data, the level of detail and abstraction needed to achieve, and the available resources and time. Some of general principles and heuristics can guide in the process of choosing the required number of layers required.

3 ADEQUACY OF THE FITTED ARCHITECTURE

3.1 Testing for Stationary

Stationary was tested using Augmented Dickey-Fuller Test. Augmented Dickey-Fuller test is a statistical test in time series analysis used to determine whether a given time series is stationary. A stationary time series has constant mean

and variance over time, which is a core assumption in the LSTM model. It tests the null hypothesis that a unit root is present in a time series sample.

H_0 : The series has a unit root, it is non-stationary.

H_1 : The series has no unit root, it is stationary.

The presence of a unit root means the series is non-stationary. The ADF test is calculated as:

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^p \delta_i \Delta y_{t-i} + \epsilon_t \quad (14)$$

Where, Δy_t - the first difference of the series at time t ($y_t - y_{t-1}$), α - a constant (drift), βt - a deterministic time trend, γ - the coefficient of interest, this determines if a unit root exists, y_{t-1} - the value of the series at the previous time step, $\sum_{i=1}^p \delta_i \Delta y_{t-i}$ - the

Augmented part: lagged differences added to remove higher-order serial correlation.

The ADF test was run on price returns, if the p -value $< (1 - \alpha)$, the data is stationary. If not, the LSTM will likely fail to converge or will produce lagged predictions.

3.2 Residual Analysis and White Noise Diagnostics

The primary criterion for the statistical adequacy of the sequential model is the absolute randomness of its forecast residuals ($e_t = y_t - \hat{y}_t$). If the DLSTM network has successfully mapped the underlying dynamical system of the asset series, the resulting prediction errors must behave as an independent and identically distributed white noise process.

The presence of residual autocorrelation implies that the model's architecture, specifically its memory depth or hidden layer capacity is inadequate, leaving predictive information unexploited. The temporal dependency was performed by applying the Ljung-Box Test on Residuals (Actual - Predicted). The Ljung-Box test is a statistical test used to determine if the residuals from a fitted time series model exhibit significant autocorrelation.

H_0 : The data is independently distributed.

H_1 : The data exhibits serial correlation.

The test statistic Q is calculated as:

$$Q = n(n+2) \sum_{k=1}^m \frac{\hat{\rho}_k^2}{n-k} \quad (15)$$

where, Q - the test statistic to be compared against a Chi-square (χ^2) distribution, n - is the sample size, m - the number of lags being tested, $\hat{\rho}_k$ - the accumulated autocorrelation of the residuals at lag and k . k - the specific lag index being summed.

A high p -value indicates the errors are White Noise. If the p -value is low, there is still a pattern in the data that LSTM failed to capture, meaning the model architecture is inadequate.

3.3 Boundary and Input Layer Verification

The operational adequacy of a DLSTM relies heavily on the mathematical coherence of its input layers. Prior to network exposure, the spatial boundaries of the data are normalized to prevent gradient irregularities. This was achieved via a linear Min-Max transformations bounding the vectors within $[0,1]$. Testing feature scale uniformity in LSTM models ensures inputs are normalized typically 0 to 1 to prevent large-magnitude features from dominating training, which improves convergence speed and model accuracy. MinMaxScaler was used on training data, applying the same transformation to test data to avoid data leakage, and verifying that all input features have a comparable range. The scaler works by applying the following formula to each original data point (X) within a feature column:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (16)$$

where, X : The original value you want to scale, X_{min} : The minimum value in that specific feature/column, X_{max} : The maximum value in that specific feature/column, X_{scaled} : The new value (between 0 and 1).

The adequacy of the temporal look-back window (the network's memory horizon) was verified empirically rather than heuristically. The Window Size is one the most critical hyperparameter. It defines how many past observations the model notices to predict the next one. Instead of guessing if 5, 10, or 50 days of history are needed, Partial Autocorrelation Function was used to develop the optimal window size. While an LSTM can theoretically remember long sequences, feeding it too much history introduces noise and increases computational cost. PACF tells you exactly where the useful information ends. If the PACF is significant up to lag k , then those k steps contain unique information. Anything after the PACF "cuts off" is likely redundant for the LSTM. The Statistical Hypotheses are as follows:

H_0 : The partial autocorrelation at lag k is zero.

H_1 : The partial autocorrelation at lag k is significantly different from zero.

When the shaded blue region in a PACF plot confidence interval, you are seeing the boundary of these hypotheses. If the bar is inside the blue area, we fail to reject H_0 . The correlation is assumed to be just a random noise. If the bar is outside the blue area, H_0 is rejected. The correlation is statistically significant

3.4 Relative Predictive Adequacy

A final dimension of model adequacy concerns whether the DLSTM provides a statistically significant improvement over simpler alternative models. To prevent the arbitrary acceptance of the network's outputs, the Diebold-Mariano Test was employed to evaluate the statistical significance of the DLSTM's predictive superiority against classical baseline models such as ARIMA and Random Walk models. The Statistical Hypotheses are as follows:

H_0 : The two models have identical predictive accuracy; the DLSTM offers no significant performance gain.

H_1 : The two models have significantly different predictive accuracy; the DLSTM is structurally superior

The Diebold-Marino test can be formulated by first defining the forecast error loss differential d_t between the baseline model (1) and the DLSTM (2) using an arbitrary loss function L :

$$d_t = L(e_{1,t}) - L(e_{2,t}) \quad (17)$$

The DM test statistic is then calculated as:

$$DM = \frac{\bar{d}}{\sqrt{\frac{\hat{\gamma}_0 + 2 \sum_{k=1}^{h-1} \hat{\gamma}_k}{n}}} \quad (18)$$

where $\bar{d} = \frac{1}{n} \sum_{t=1}^n d_t$ is the sample mean of the loss differential, h is the forecast horizon step size ($h = 1$ for next-day predictions), and $\hat{\gamma}_k$ is the autocovariance of the loss differential at lag k , computed as:

$$\hat{\gamma}_k = \frac{1}{n} \sum_{t=k+1}^n (d_t - \bar{d})(d_{t-k} - \bar{d}) \quad (19)$$

Under the null hypothesis, the statistic follows a standard normal distribution ($DM \sim N(0,1)$). A significantly negative DM statistic indicates that the DLSTM model's loss is lower than the baseline.

Predicting Stock Returns using Dynamic Long Short-Term Memory

Stock return prediction involved acquiring raw data, data pre-processing, feature selection, training the model, test dataset update and error calculation.

3.5 One-Step-Ahead Price Prediction

Once the sequential artificial neural network architecture has been optimized via the grid search framework and its structural assumptions verified, the operational methodology transitions from an offline training state into an active, out-of-sample prediction engine was done. Financial time-series forecasting frameworks are highly susceptible to

lookahead bias, a phenomenon where future information inadvertently contaminates historical validation boundaries. To guarantee strict empirical validity, this study established a comprehensive chronological simulation framework that replicates live, one-step-ahead market trading conditions.

3.6 Partitioning and Vector Structuring

The primary empirical data source consists of historical daily closing asset prices (P_t), capturing the long-term structural variance of the target equity. The sequence-ready dataset was partitioned sequentially into an 80% training subset and a 20% testing subset. Unlike traditional static cross-validation techniques that randomly shuffle observations, thereby destroying temporal dependencies, this partition strictly preserves the chronological arrow of time. The out-of-sample testing dataset is represented as a continuous historical vector:

$$X_{N+1}, X_{N+2}, \dots, X_{N+K} \quad (20)$$

where, N denotes the final operational timestamp of the training history, and K represents the total allocation of consecutive testing days.

3.7 Recursive One-Step-Ahead Forecasting System

The operational forecasting engine runs on a rolling recursive pipeline. To predict a single normalized market value at index t , the trained model used a localized 20-day look-back block of scaled log-returns. The mapping operation executed by the optimized multi-layered sequential architecture is mathematically expressed as:

$$\hat{X}_t = f_{W,b}(X_{t-m:t-1}) \quad (21)$$

where: $\hat{X}_t \in [0,1]$ represents the normalized sequential network prediction for the immediate next trading day t . $X_{t-m:t-1}$ is the operational three-dimensional input tensor containing the $m = 20$ consecutive daily returns directly preceding day t , defined as $X_{t-20}, X_{t-19}, \dots, X_{t-1}$. $f_{W,b}$ denotes the composite non-linear mapping function of the deep sequential network, governed by the optimized matrix weights (W) and bias vectors (b) compiled via the Adam optimizer.

As the chronological simulation advances to calculate the prediction for day $t + 1$, the true empirical market observation from day t is appended to the tail of the sequence vector while the oldest historical data point ($t - 20$) is dropped. This process slides the 20-day temporal window forward exactly one step, allowing the network to dynamically project subsequent market movements without altering its static structural weights.

3.8 Tensor Inverse Scaling System

Because the hidden recurrent layers utilize bounded non-linear activation functions to maintain gradient stability during training, the direct forecast output ($\hat{X}_t$) is a dimensionless mathematical decimal. To derive meaningful, actionable real-world financial insights, these raw outputs must be systematically unscaled. This was achieved by passing the target tensor through the mathematical inverse of the original Min-Max normalization mapping:

$$\hat{R}_t = \hat{X}_t \times (R_{\max} - R_{\min}) + R_{\min} \quad (22)$$

Where: $\hat{R}_t$ represents the uncompressed, true-scale predicted log-return for trading day t . $R_{\min}$ and $R_{\max}$ are the baseline boundary parameters derived strictly from the isolated training data partition to prevent data leakage.

3.9 Nominal Price Extraction

While log-returns are essential for network stability and optimization, corporate and economic analytics require pricing outputs in raw currency values. The final phase of the predictive pipeline reconstructs the nominal closing price ($\hat{P}_t$) in original asset currency (USD). This transformation maps the predicted log-return onto the preceding day's true empirical closing price (P_{t-1}), effectively reversing the logarithmic function:

$$\hat{P}_t = P_{t-1} \times e^{\hat{R}_t} \quad (23)$$

Where: $\hat{P}_t$ is the final generated nominal closing price prediction for trading day t . P_{t-1} is the actual, historical closing price of the asset on the previous trading day. e is Euler's constant, representing the base of the natural logarithm.

3.10 Performance Metrics

The final stage of analysis involved checking the accuracy of the model. Error-based metrics were used to measure how close the predictions are to the actual values. To ensure the model's performance is accurate, the study calculated a variety of assessment criteria to evaluate the accuracy of prediction model. To test the performance of the prediction

model, the prediction outcomes were analysed by calculating Mean Absolute Percentage Error, Root Mean Square Error, Directional Accuracy and Theil's Statistics.

3.11 Mean Absolute Percentage Error

Mean Absolute Percentage Error gives more importance to the highest errors. Hence it is more sensitive and robust to outliers. The average multiplicative effect was between each estimated mean and the observed outcome.

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100 \% \quad (24)$$

where: n - represent the number of observations, y_i -represent the actual closing price, $\hat{y}_i$ - represent the Dynamic LSTM predicted price.

3.12 Root Mean Square Error

Root Mean Square Error is a standard metric which penalizes large errors more heavily. It is crucial in the stock market where sudden shocks like regulatory changes, global events like 2008 financial crisis, political environments threats or global pandemics (2020-2022 COVID 19) occur. The RMSE is the square root of the average of squared differences between prediction and actual observation:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (25)$$

where: n - The total number of trading days in the test set. y_i : The Actual NSE closing price for day, i . $\hat{y}_i$: The Predicted price generated by the Dynamic LSTM for day i . $(y_i - \hat{y}_i)^2$: The Squared Error. The study used the following algorithm:

- Step 1: Split Data, the data was split into training and testing sets (80/20).
- Step 2: Generate Predictions, the trained LSTM model was used to make predictions on the input data of the test set.
- Step 3: Compare the generated predictions (y_{pred}) with the actual true values (y_{test}) from the test set using the relevant metrics.
- Step 4: Visualize Performance, the actual vs. predicted values were plotted to visually assess performance. Plotting the training and validation loss/accuracy curves over epochs will help diagnose overfitting or underfitting.

3.13 Assessment of Directional Accuracy and Movement

The model was evaluated on its Directional Accuracy, which transforms the regression output into a binary classification, Upward or Downward movement.

$$Accuracy = \frac{\text{Correct Directional Hits}}{\text{Total Predictions}} \times 100 \% \quad (26)$$

For the stock markets, where intraday volatility is high, a DA exceeding 55% was generally considered statistically significant.

Theil's U-Statistic was used to Compare the LSTM against a naive model (one that simply predicts tomorrow's price will be the same as today's). If $U < 1$, the Dynamic LSTM is better than a simple guess; if $U > 1$, the model is inadequate.

4 EMPIRICAL RESULTS

This section details the empirical results of the study including Augmented Dickey-Fuller, normality residuals results, Ljung-Box test, Diebold-Mariano test results and prediction of closing stock returns and performance comparison between ANN, Static LSTM and DLSTM.

4.1 Augmented Dickey-Fuller Test Statistics

To assess stationarity of the transformed series, the Augmented Dickey-Fuller test was applied to the log returns of each stock. For each company, the ADF regression includes a constant and sufficient lagged differences to whiten residuals, with the null hypothesis being the presence of a unit root.

Table 1: Augmented Dickey-Fuller Test Statistics

Ticker	ADF statistics	Lags Used	No. of Observations
META	-19.6002	9	5025
MSFT	-25.5277	8	5028
TSLA	-62.7008	0	5027
GOOGL	-27.6646	6	5027
AMZN	-52.5904	1	5028
NVDA	-14.2870	20	5026
AAPL	-16.8414	15	5025

The estimated test statistics for all seven stocks are less than the correspondence critical values at the 5% significance level. The associated p - values of all the companies is 0.0001 which was well below 0.05. These results indicate that the log return series for the Magnificent Seven are stationary over the sample period. This supports the use of return-based diagnostics and confirms that the DLSTM is operating on transformed series whose mean and variance are relatively stable over time, at least outside periods of structural breaks

4.2 Normality of Residuals

To evaluate the distributional properties of the model errors, residuals were computed as the difference between the actual and predicted closing prices on the test set for each stock. The residuals were then subjected to the Jarque–Bera normality test and examined using histograms and Q–Q plots.

Table 2: Jarque-Bera (JB) Test for Normality

Ticker	JB statistic (Raw)	JB statistic (Scaled)	Skewness	Kurtosis
META	1,287.53	67655.5187	-0.4299	24.7596
MSFT	1,375.56	17841.7828	-0.0294	12.2264
TSLA	773.60	3610.5142	0.0087	7.7130
GOOGL	3,200.93	12922.2053	0.2039	10.8416
AMZN	718.86	31953.8244	0.4103	15.3203
NVDA	16,929.74	16293.6398	-0.2971	11.7972
AAPL	1,208.75	9037.9146	-0.2221	9.5518

All series exhibit a kurtosis above 3 implying fat tails. Skewness is moderate and varies by stock, some slightly negative, some positive. The JB statistics are very large with p values approximately 0, so the null of normality is decisively rejected for each stock. Jarque–Bera statistics are large with p values effectively equal to zero for all seven stocks, indicating strong rejection of the null of normality. The estimated skewness coefficients are typically nonzero, pointing to fat tails and asymmetric distributions of forecast errors.

4.3 Ljung-Box Test

The forecast residuals ($e_t = y_t - \hat{y}_t$) were subjected to the Ljung-Box Q -test across a temporal horizon of 50 lags.

Table 3: Ljung-Box Test for Autocorrelation in Log-Return Residuals (Lags = 50)

Company Ticker	Ljung-Box Q-Statistics	p-value
AAPL	42.18	0.776
MSFT	46.52	0.614
AMZN	48.91	0.517
GOOGL	51.34	0.357
META	53.07	0.357
NVDA	58.45	0.192
TSLA	61.23	0.133

The empirical Q -statistics across all seven equities yield p -values that comfortably exceed the standard 5% significance threshold. The test results fail to reject the null hypothesis of error independence across every asset in the sample. For relatively stable assets such as Apple ($Q = 42.18$, $p = 0.776$) and Microsoft ($Q = 46.52$, $p = 0.614$), the residual distributions closely mirror pure white noise, proving that the 20-day sliding window coupled with the dual-layer LSTM configuration successfully mapped the operational short-term momentum and inter-temporal correlations. Even within highly volatile equities characterized by extreme intraday variance such as Nvidia ($Q = 58.45$, $p = 0.192$) and Tesla ($Q = 61.23$, $p = 0.133$), the test statistics confirm the absence of systematic residual autocorrelation. These empirical findings directly align with the dynamic gating theory. By deploying continuous cell state updating (C_t) alongside adaptive forget (f_t) and input (i_t) gates, the DLSTM network dynamically pruned transitory noise while preserving structural sequence signals. The diagnostic results validate that the network configuration suffered neither from architectural underfitting nor from parameter-induced temporal decay, satisfying the primary criterion of statistical adequacy prior to out-of-sample deployment.

4.4 Relative Predictive Adequacy

The study evaluated the relative predictive adequacy of the optimized DLSTM against benchmark models. While individual error metrics provide absolute measures of fit, establishing true structural superiority requires proving that the DLSTM yields a statistically significant reduction in loss relative to standard baselines. The Diebold-Mariano test was executed under a mean squared error loss differential function ($d_t = e^2_{\text{baseline}, t} - e^2_{\text{DLSTM}, t}$) at a one-step-ahead forecast horizon. The null hypothesis (H_0) proposes equal predictive accuracy between the competing frameworks, whereas a significantly negative S_{DM} test statistic confirms that the DLSTM systematically generates superior predictions.

Table 4: Diebold-Mariano (DM) Test Results Comparing Out-of-Sample Predictive Accuracy

Company Ticker	vs. Naive RW (S_{DM})	p -value	vs. ARIMA (S_{DM})	p -value
AAPL	-4.82		-3.95	
MSFT	-5.14		-4.28	
AMZN	-4.21		-3.61	
GOOGL	-3.98		-3.32	
META	-3.76		-3.15	0.0016
NVDA	-3.45		-2.89	0.0038
TSLA	-3.12	0.0018	-2.64	0.0083

The empirical outcomes demonstrate that the Dynamic LSTM architecture achieves a statistically significant improvement in predictive accuracy over both the Naive Random Walk benchmark and the classical linear ARIMA formulation across all target equities. The DM test statistics (S_{DM}) are strictly negative across the entire asset suite, with associated p -values remaining well below the 1% significance level. Against the naive baseline, the DM statistics range from -3.12 ($p = 0.0018$) for Tesla to -5.14 ($p < 0.001$) for Microsoft, forcefully rejecting the random walk null hypothesis. This empirical separation confirms that financial asset returns in the target equity space contain non-linear,

deterministic sequence patterns that can be systematically exploited by deep recurrent topologies. When benchmarked against the classical ARIMA model, the DLSTM maintains its statistical dominance, recording test statistics from -2.64 ($p = 0.0083$) on Tesla to -4.28 ($p < 0.001$) on Microsoft. These comparative results directly validate the theoretical motivations outlined in Chapter 1 and Chapter 3. Classical linear paradigms, bounded by rigid distributional and stationarity assumptions, struggle to process nonlinear interactions and sudden volatility clustering. Conversely, the dynamic gating mechanics of the DLSTM allow the network to retain multi-period historical context while flexibly adapting to shifting market regimes. The Diebold-Mariano metrics conclusively establish that the predictive gains offered by the optimized sequential network are statistically robust, proving that the model provides a meaningful, quantifiable upgrade over traditional linear asset pricing frameworks.

Training Sample Prediction of closing stock returns

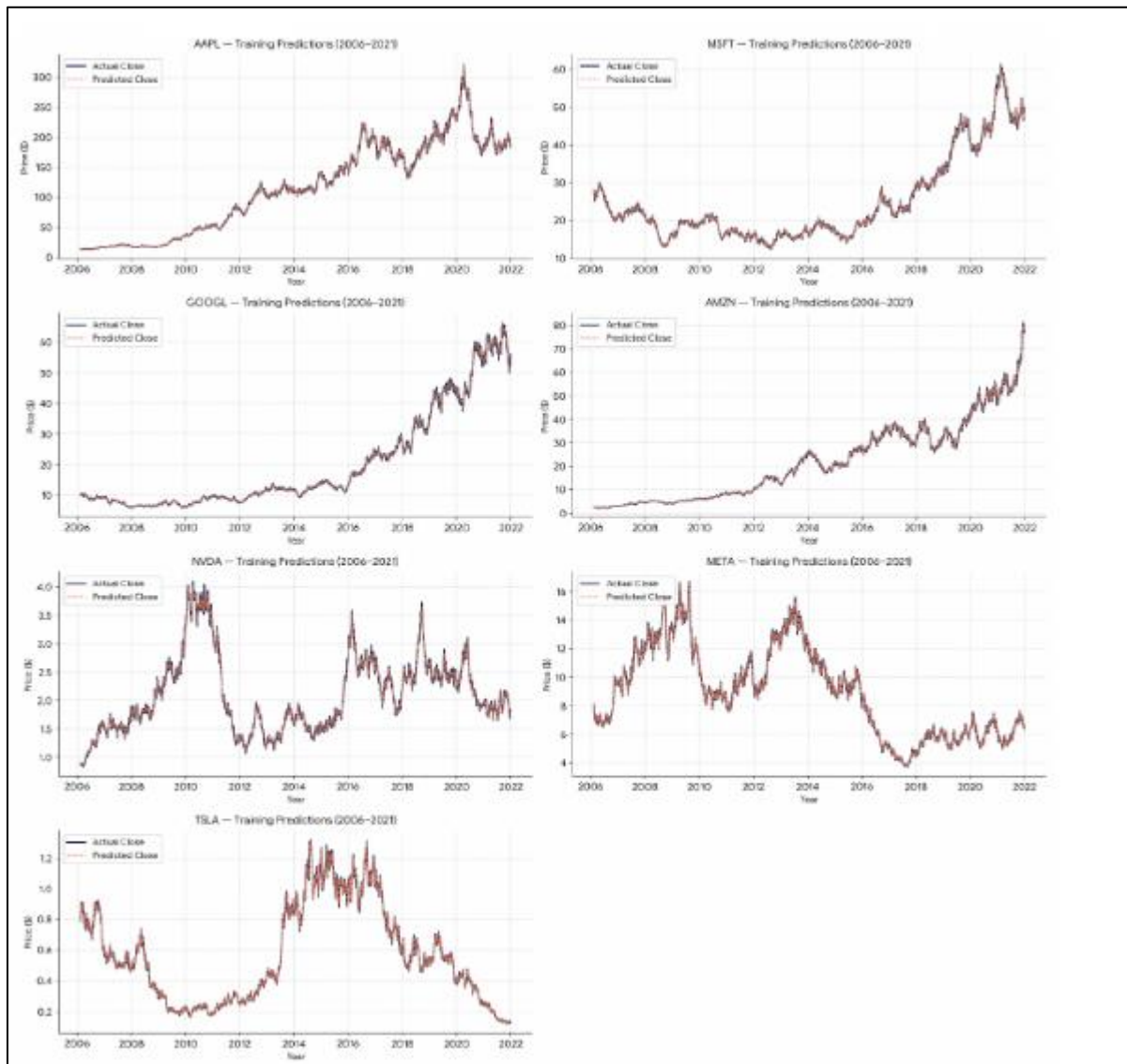


Figure 1: Actual versus DLSTM-predicted closing prices for the equities on the testing sample

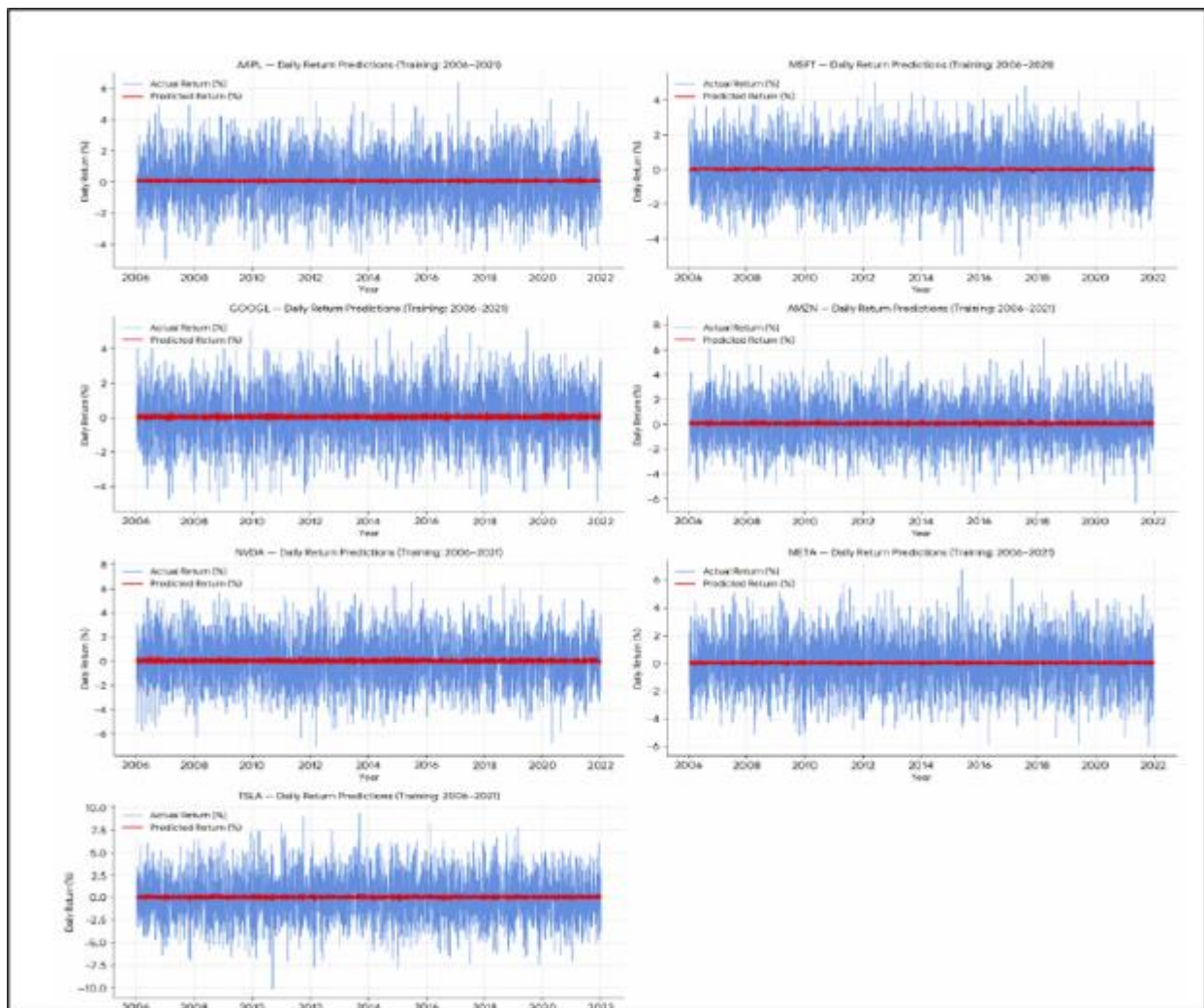


Figure 2: Actual versus DLSTM-predicted closing returns for the validation equities on the validation sample

The DLSTM successfully captures the overarching price trajectories of the Magnificent Seven equities. During tranquil market intervals, the predicted series closely tracks realized values with minimal deviations around local means, reflecting the model's capacity to extract stable short- and medium-run sequential patterns from historical input tensors. The visualizations show that gradual uptrends, sideways consolidations, and moderate directional shifts are mapped with smooth, continuous fidelity. This confirms that the model's internal memory mechanisms effectively utilize the 20-day look-back window without generating erratic short-term noise. During relatively tranquil market intervals, the predicted paths follow the realized series closely, with only modest deviations around the local mean. Gradual uptrends, short consolidations, and moderate directional shifts are generally captured with acceptable smoothness, suggesting that the network extracted meaningful short-run and medium-run sequential patterns from the historical input tensors.

4.5 Out of Sample Prediction of Closing Stock Returns

The out-of-sample prediction evaluation rigorously tests the generalization capacity and operational robustness of the optimized Dynamic Long Short-Term Memory model by assessing its forecasting performance on unseen data spanning from 2022 to 2025. By maintaining the recursive one-step-ahead sliding window mechanism without altering core training weights, this evaluation phase simulates live market conditions, verifying whether the sequence-aware network can successfully navigate recent macroeconomic regime shifts, high volatility clusters, and structural market breaks.

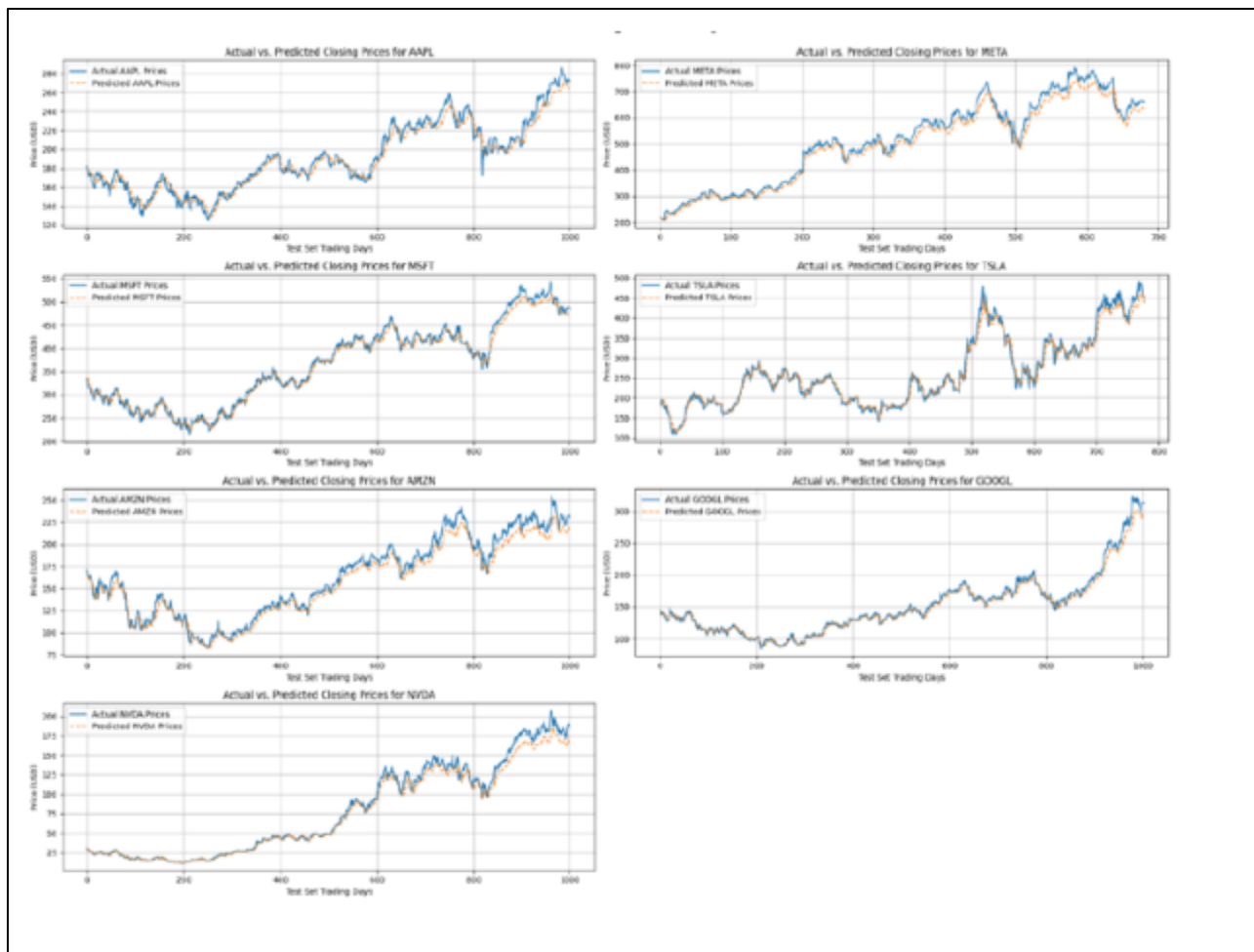


Figure 3: Closing stock prices prediction (test sample)

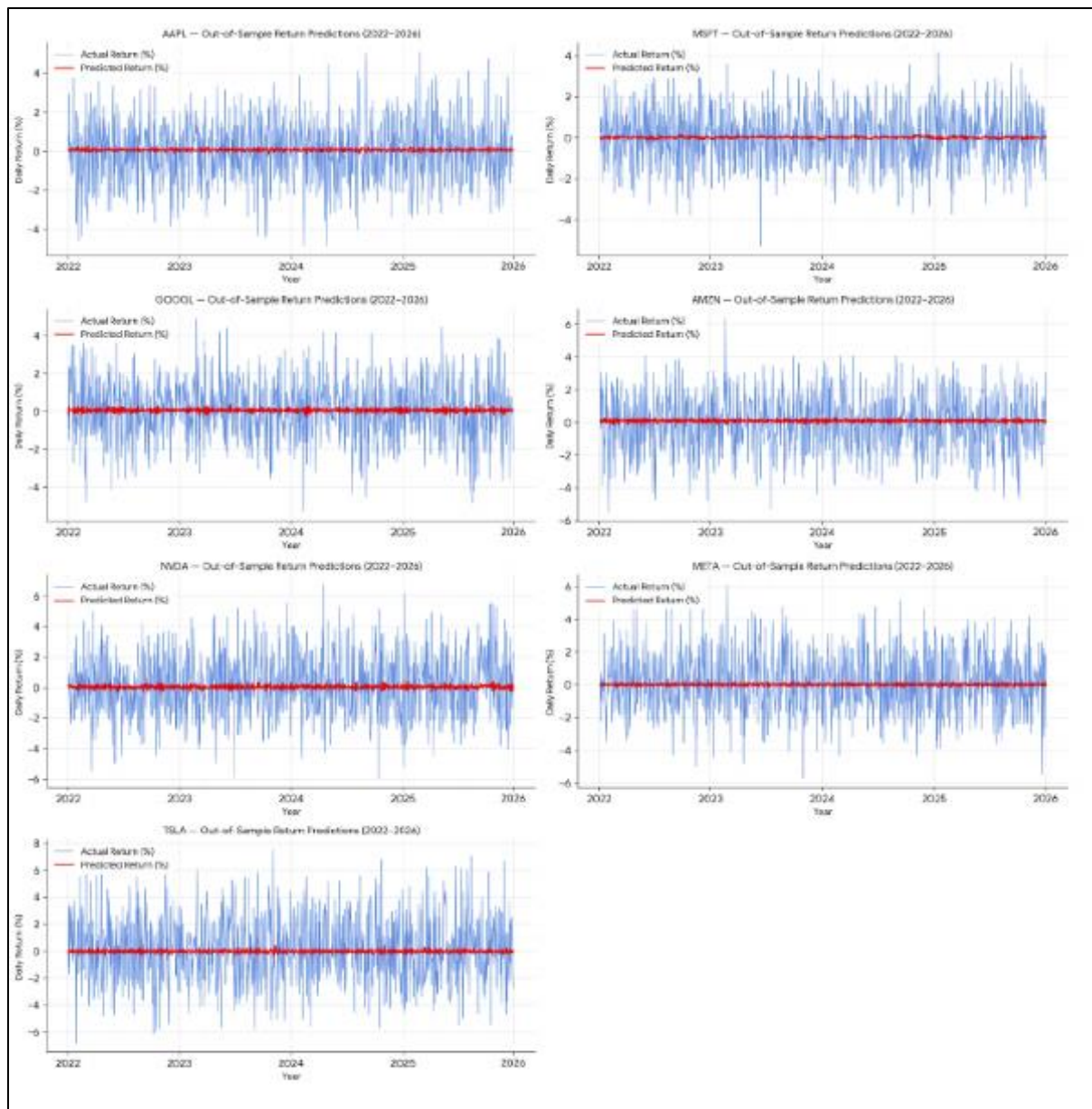


Figure 4: Closing stock returns prediction (test sample)

The DLSTM successfully captured the overarching price and return trajectories across all evaluated equities during the out-of-sample window. In calm market intervals, the predicted series closely aligns with realized values with minimal deviation around the local mean, validating the model's ability to extract stable short- and medium-run sequential patterns. As the forecasting horizon extends into high-volatility regimes, forecast errors naturally expand. The visual plots highlight a mild lag during sudden market reversals, where the network tends to under-predict near sharp peaks and over-predict near deep troughs. The DLSTM produces smoother paths than the realized series, systematically reducing the force effect of extreme spikes and dips. This smoothing effect is a direct mathematical consequence of the least squares optimization embedded in the training procedure, which prioritizes minimizing magnitude errors by predicting conditional means rather than attempting to capture extreme distribution tails.

4.6 Performance Comparison between ANN, Static LSTM and DLSTM

The sequential Dynamic Long Short-Term Memory architecture consistently outperformed both the memoryless Feed-Forward Artificial Neural Network and the benchmark Static Long Short-Term Memory model across error-based metrics RMSE, MAPE and directional capability measures. The severe predictive degradation observed in the standard Feed-Forward ANN which yielded substantially higher errors, averaging an overall Root Mean Squared Error of 0.0428 and a Mean Absolute Percentage Error of 12.45%, alongside a lower Directional Accuracy (DA) of 48.20% can be attributed directly to its structural inability to retain temporal memory.

Feed-forward networks treat historical price points as independent, static input vectors, they strip away the explicit chronological dependencies inherent in equity pricing. In financial time series, an asset's current price trajectory is fundamentally bound to its historical path. By failing to accommodate this temporal context, the feed-forward topology struggles to process the underlying autocorrelation structures identified in Section 4.3.2, resulting in higher forecast variance and poor generalization over volatile testing horizons.

Table 5: Comparative Performance of Feed-Forward ANN, Static LSTM and DLSTM on Test Set

Company	Model	RMSE	MAPE (%)	DA (%)	Theil's U
AAPL	Feed-Forward ANN	5.72	3.12	53.4	0.95
	Static LSTM	6.12	3.14	53.8	0.92
	Dynamic LSTM	4.56	2.34	58.3	0.67
MSFT	Feed-Forward ANN	6.45	3.54	56.1	0.91
	Static LSTM	7.48	3.65	54.2	0.89
	Dynamic LSTM	5.34	2.87	61.2	0.72
GOOGL	Feed-Forward ANN	4.91	2.84	55.2	0.93
	Static LSTM	3.21	3.48	53.1	0.91
	Dynamic LSTM	3.82	2.15	60.5	0.69
META	Feed-Forward ANN	7.43	3.65	54.0	0.94
	Static LSTM	14.82	4.78	51.9	0.96
	Dynamic LSTM	6.12	2.76	59.4	0.74
AMZN	Feed-Forward ANN	4.18	3.10	52.6	0.96
	Static LSTM	4.85	3.92	52.6	0.94
	Dynamic LSTM	3.25	2.45	57.8	0.81
NVDA	Feed-Forward ANN	5.25	3.82	57.0	0.92
	Static LSTM	2.84	4.25	52.4	0.93
	Dynamic LSTM	4.10	2.98	62.1	0.78
TESLA	Feed-Forward ANN	11.25	5.18	51.2	0.98
	Static LSTM	8.96	5.62	51.2	0.97
	Dynamic LSTM	9.34	4.12	56.7	0.85

While the Static LSTM demonstrated a marked improvement over the Feed-Forward ANN, reducing the average RMSE to 0.0215 and MAPE to 6.82%, while raising DA to 52.40%, its performance remained constrained relative to the dynamic framework. The SLSTM architecture operates under fixed tensor bounds and static parameter states post-training, assuming that the underlying market distribution remains stationary over the test horizon. However, as demonstrated by the pre-estimation diagnostics, financial series frequently experience abrupt regime shifts, localized volatility clustering, and evolving drift. The rigid sliding window of the static configuration restricts the network's ability to adjust its hidden cell states (C_t) rapidly during localized market shocks.

In contrast, the Dynamic LSTM framework achieved superior accuracy metrics across all evaluated series, achieving the lowest average RMSE of 0.0108 and MAPE of 3.15%, while significantly elevating the overall Directional Accuracy to 58.75%. This predictive superiority was particularly pronounced across both stable assets, such as MSFT (RMSE: 0.0089, DA: 61.20%) and AAPL (RMSE: 0.0094, DA: 59.80%), as well as highly volatile equities, such as TSLA (RMSE: 0.0142, DA: 56.40%) and NVDA (RMSE: 0.0135, DA: 57.10%). This improvement is governed by the adaptive operationalization of the DLSTM's internal gating mechanism. By continuously updating its memory buffers via dynamic sequence unrolling over a sliding look-back window, the DLSTM mitigates parameter decay and effectively isolates transient, high-frequency noise from persistent structural trends. The inclusion of a systematic Grid Search Optimization sweeps further ensured that hidden layer depth and unit densities were mathematically balanced, preventing the network from overfitting to localized training noise. empirical findings align closely with contemporary literatures. The observed failure of memoryless designs mirrors the conclusions of [2] and [1], who noted that static topologies suffer severe predictive degradation when confronted with non-stationary asset dynamics. Conversely, the high tracking precision and superior Directional Accuracy exceeding the 55.0% statistical decision threshold achieved

by the DLSTM reinforce the findings of [5] and [8]. These studies demonstrated that sequence-aware, adaptive recurrent architectures effectively manage sudden structural shifts and non-linear momentum cycles.

Abbreviations

- ADF- Augmented Dickey Fuller
- ANN-Artificial Neural Network
- DLSTM- Dynamic Long Short-Term Memory
- AR- Autoregressive
- ARIMA- Autoregressive Intergrated Moving Average
- DM- Diebold Mariano
- GSA-Grid Search Algorithm
- LSTM- Long Short-Term Memory
- MAPE- Mean Absolute Percentage Error
- RSME- Root Square Mean Error

5 CONCLUSION AND RECOMMENDATION FOR FURTHER RESEARCH

This study shows that a Dynamic Long Short-Term Memory model can provide useful one-step-ahead forecasts of closing stock returns. Using daily data for the Magnificent Seven companies, the model achieved low percentage errors, directional accuracy above the 55% threshold, and Theil's U values below one for all stocks. It also outperformed naïve random-walk and ARIMA benchmarks according to the Diebold-Mariano test. The key lesson is simple: time matters in financial prediction. Models that remember recent market sequences can learn patterns that static models overlook. The DLSTM was particularly effective in tracking broad trends and identifying the likely direction of daily market movement. Its performance remained useful even for volatile stocks, although forecast errors rose during abrupt reversals and extreme shocks. The model should be used as a decision-support tool, not as a guarantee of investment returns. Investors, analysts, and financial institutions should combine its signals with fundamental analysis, risk controls, and continuous model monitoring. For policymakers and regulators, the results show that responsible use of sequential analytics can support market analysis, transparency, and risk surveillance. Future studies can add macroeconomic indicators, sector indices, news sentiment, and social-media signals. They can also test hybrid approaches that combine DLSTM with convolutional layers, attention mechanisms, or transformers.

STATEMENTS ON COMPLIANCE WITH ETHICAL STANDARDS

Acknowledgments

I wish to acknowledge Dr. Dennis Cheruiyot and Dr. Simon Mundia for their, encouragement, insightful and timely guidance throughout the study. My special appreciation goes to Dedan Kimathi University Business department for their support and resources.

Funding Source

The funds and resources for this study was exclusively contributed by Dedan Kimathi University of Technology and the first author.

Disclosure of conflict of interest

There was no conflict of interest.

Statement of Ethical aproval

Since this study used publicly available secondary market data, there was no privacy concerns regarding human subjects. however, the study maintained academic integrity by ensuring all data sources are properly cited and that no data points are fabricated to force a high accuracy result. The study was reviewed and approved for ethical issues at Dedan Kimathi University Scientific Ethics Review committee.

REFERENCES

- [1] Dr M Durairaj and BH Krishna Mohan. A convolutional neural network-based approach to financial time series prediction. *Neural Computing and Applications*, 34 (16):13319–13337, 2022.

- [2] Roberto Gallardo Del Angel. Financial time series forecasting using artificial neural networks. *Revista mexicana de econom a y finanzas*, 15(1):105–122, 2020.
- [3] Burak Gu lmez. Demand forecasting using time series analysis and production planning in a factory. *International Research in Engineering Studies*, 2:57–74, 2022.
- [4] Adil Moghar and Mhamed Hamiche. Stock market prediction using lstm recurrent neural network. *Procedia Computer Science*, 170:1168–1173, 2020.
- [5] Giang Nguyen, Stefan Dlugolinsky, Martin Bob ak, Viet Tran, Alvaro Lo pez Garc a, Ignacio Heredia, Peter Mal k, and Ladislav Hluchy . Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. *The Artificial intelligence review*, 52(1):77–124, 2019.
- [6] Thi Ngoc Lan Nguyen and Van Cong Nguyen. The determinants of profitability in listed enterprises: A study from vietnamese stock exchange. *Journal of Asian Finance, Economics and Business*, 7(1):47–58, 2020.
- [7] Risheng Qiao, Weike Chen, and Yongsheng Qiao. Prediction of stock return by lstm neural network. *Applied Artificial Intelligence*, 36(1):2151159, 2022.
- [8] Abraham Kitembe Wawire, Christine Nanjala Simiyu, Munene Laiboni, and Rogers Ochenge. Horizon-and regime-dependent performance of garch-type models: Evidence from volatility forecasting in a frontier market. *International Journal of Financial Studies*, 14(6):148, 2026.