Journal of Engineering and Technology for Industrial Applications

ISSN ONLINE: 2447-0228

ITEGAM-JETIA

Manaus, v.11 n.55, p. 212-221. September/October, 2025.

DOI: https://doi.org/10.5935/jetia. v.11 n.55.2658

RESEARCH ARTICLE OPEN ACCESS

NOVEL INSIGHTS ON THE COMPARATIVE STUDY BETWEEN LSTM AND TRANSFORMER MODELS FOR FINANCIAL TIME SERIES PREDICTION

Rabia Nichani[1], Laid Gasmi[2], Salheddine Kabou[3] and Nabil Laiche[4]

[1, 2] Laboratory of Mathematics Modeling and Applications, University of Adrar, Algeria

[3] Department of Mathematics and Computer Science, Higher Normal School of Bechar.

[4] Dynamical Systems and Control Laboratory. Oum El Bouaghi University, Algeria

[1]https://orcid.org/0009-0004-4755-7946, [2]https://orcid.org/0000-0001-8925-0089

[3]http://orcid.org/ 0000-0002-1423-7215, [4]http://orcid.org/0000-0003-3392-1300

Email : nicha.rabia@univ-adrar.edu.dz, lai.gasmi@univ-adrar.edu.dz, kabou.salaheddine@yahoo.fr, nlaiche2020@gmail.com

ARTICLE INFO

Article History

Received: September 3, 2025

Revised: September 30, 2025

Accepted: October 5, 2025

Published: October 31, 2025

Keywords:

Financial Time Series,
Forecasting,
Deep Learning,
LSTM, Transformer,
Optimizers.

ABSTRACT

This study investigates the impact of deep learning model architectures and optimization techniques on financial time series forecasting, with a focus on Transformer-based models and Long Short-Term Memory networks. Optimization methods such as Adaptive Moment Estimation and Adaptive Moment Estimation with Infinity Norm play a critical role in enhancing training efficiency and prediction accuracy, while architectural decisions determine each model's ability to capture both short-term and long-term dependencies in sequential data. Using datasets including stock prices from Microsoft and NVIDIA, the models are evaluated based on metrics such as Mean Absolute Percentage Error, Root Mean Squared Error, prediction variance, and training speed. The results demonstrate the complementary strengths of Transformers and Long Short-Term Memory Networks, underscoring the importance of tailored architectures and optimization strategies in deep learning-based financial forecasting.

I. INTRODUCTION

Time series forecasting plays a pivotal role in diverse domains, including finance, supply chain management, and environmental science. In finance, predicting future trends in stock prices and market indices is particularly challenging due to the complex statistical properties of financial time series. These properties include non-stationarity, volatility clustering, and heavy tails, which defy the assumptions of traditional time series models and necessitate more advanced approaches [**1**,**2**].

Traditional methods such as ARIMA (Autoregressive Integrated Moving Average) and exponential smoothing have proven effective for stationary data, but they struggle with the dynamic, high-dimensional nature of financial data. Deep learning models, notably Long Short-Term Memory (LSTM) networks and Transformers, have emerged as powerful tools in addressing these challenges. LSTMs excel at capturing temporal dependencies and long-term relationships in sequential data, thanks to their gated architecture [**3**, **4**]. On the other hand, Transformers leverage self-attention mechanisms to model both local and global dependencies effectively, making them particularly suited for complex and irregular time series like those in financial markets [**5, 6**].

Despite their promise, applying LSTMs and Transformers to financial forecasting remains underexplored compared to their widespread adoption in fields like image recognition [**7**] and natural language processing [**8, 9**]. Existing studies have shown mixed results, highlighting gaps in understanding the suitability of these architectures for financial time series forecasting. For example, LSTMs have demonstrated robust performance in capturing sequential patterns, but their reliance on sequential processing can limit computational efficiency [**10,11**]. Transformers, while efficient for long sequences, face challenges such as higher computational complexity due to their self-attention mechanism [**12**].

This study aims to address these gaps by conducting a systematic comparison of LSTMs and Transformers in the context of stock price prediction for Microsoft (MSFT) and Nvidia Corporation (NVDA). Key contributions include exploring the impact of diverse optimizers (e.g., Adam, RMSprop, Adadelta) and evaluating performance based on metrics such as the Mean Absolute Percentage Error

(MAPE) and Root Mean Squared Error (RMSE). The results highlight the transformative potential of deep learning architectures in financial forecasting, offering insights into their strengths and limitations while paving the way for future innovations in hybrid modeling.

The structure of the remainder of this paper is as follows. After this brief introduction, Section 2 will discuss existing surveys that focus on deep learning models such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Transformer models for financial time series forecasting. Section 3 presents the study of the most important statistical properties of the data. In Section 4, we compare the effects of optimizers on the prediction quality of the LSTM model and the Transformer model for two financial time series, MSFT and NVDA, using diverse metrics. Finally, we conclude in Section 5 by summarizing our findings.

II. BACKGROUND AND RELATED WORKS

The ability of Deep Learning (DL) models to analyze intricate patterns in financial time series data has gained significant attention in recent years [**13**]. Numerous studies and surveys have investigated the application of advanced DL architectures such as Convolutional Neural Networks, Multi-Layer Perceptrons, Long Short-Term Memory networks, and Transformer models in financial forecasting tasks. CNNs, known for their strength in feature extraction, have been adapted to handle time series data by effectively capturing local temporal dependencies. For instance, studies such as [**14**,**15**] have demonstrated the utility of CNNs in forecasting financial trends, particularly in tasks requiring multi-dimensional input representations. LSTMs, a variant of RNNs, have been widely adopted due to their ability to capture long-term dependencies and sequential patterns in financial datasets. These capabilities make LSTMs particularly effective for time series forecasting, where past data points significantly influence future predictions. In the following sections, we review the literature on time series prediction, with a specific focus on the application of LSTM and Transformer models for financial forecasting.

II.1 LSTM-BASED TIME SERIES PREDICTION

Long Short-Term Memory (LSTM) networks have been extensively utilized in financial time series prediction, with numerous studies [**16**] demonstrating their success in forecasting stock prices. To further enhance prediction performance, Bidirectional LSTM (BiLSTM) has been employed [**17**]. The application of Natural Language Processing (NLP) techniques has also led to the introduction of Sequence-to-Sequence (Seq2Seq) models [**18**], initially developed for machine translation tasks. These models, when integrated with LSTM structures and attention mechanisms, have shown significant improvements in financial time series forecasting [**19, 20**]. Moreover, [**11**] innovatively combined CNNs and LSTM to create a novel architecture known as DeepLOB.

This architecture outperformed LSTM-only in predicting future stock price movements from Limit Order Book (LOB) data. In a subsequent study, [**10**] further enhanced DeepLOB by incorporating Seq2Seq and attention models, enabling multi-horizon price prediction in a single forward pass. This approach reduced training time and enhanced performance, especially for long-term predictions, outperforming the original DeepLOB model. According to [**21**], the combination of CNN and LSTM demonstrated superior stability and performance in predicting LOB (Limit Order Book) price movements. Similarly, [22] utilized the Order Flow Imbalance (OFI) feature from the LOB to forecast minimum-price returns, where the CNN-LSTM architecture again exhibited superior results. Recently, hybrid models such as Transformer-CNN [**23**], ARIMA-LSTM, and ARIMA-XGBoost have gained attention for improving forecasting accuracy in financial time series prediction [**24**].

II.2 TRANSFORMER-BASED TIME SERIES PREDICTION

The Transformer model, initially introduced for Natural Language Processing (NLP) tasks, has since inspired the development of various advanced models such as SOTA, BERT, and GPT-3. These models have gained significant attention due to their effectiveness, and their application has increasingly extended to time series prediction. However, the canonical self-attention mechanism in Transformer models has a time and memory complexity of $O(L^2)$, which can be computationally prohibitive for long sequences. To address this, several adaptations of the original Transformer architecture have been proposed to enhance its efficiency, particularly in processing long time series for forecasting tasks [**6**], with the most recent advancement being the FEDformer model [**12**]. In the financial domain, the Transformer has been applied to various prediction tasks. For example, [**25**] combined a Temporal Fusion Transformer with Support Vector Regression (SVR) and LSTM for stock price forecasting, showing improved performance. Similarly, [**26**] employed a Transformer to predict Dogecoin prices, demonstrating its superior predictive ability over LSTM models.

These studies primarily focus on using Transformer-based methods for predicting stock prices based on OHLC (Open, High, Low, Close) data. However, there is still limited research on the application of Transformer models to Limit Order Book (LOB) data for prediction. While LSTM models have been widely used in financial time series forecasting and have been extensively tested across different datasets, the use of Transformers remains more restricted. This study aims to extend the application of Transformer-based models to a broader range of financial time series prediction tasks, with a particular focus on comparing their performance against LSTM models [2**7, 28**]. Using financial time series data from companies like Microsoft (MSFT) and NVIDIA (NVDA), we evaluated the models based on MAPE, RMSE, prediction variance, and training speed. The results reveal that Transformer and LSTM models possess complementary strengths, with Transformers excelling at capturing long-term dependencies, while LSTMs, though effective at managing short-term sequential patterns, are more limited when dealing with complex long-term relationships.

These findings align with previous studies, such as those by [**16**], which demonstrated the success of LSTMs in financial time series forecasting. However, our approach, which combines Transformers with specific architectural optimizations, outperforms traditional LSTM models in terms of MAPE and RMSE, while also reducing training time, which is critical for large-scale applications. One of the major contributions of this study lies in exploring the impact of architectural choices and optimization strategies. We show that customizing model architectures based on the specific characteristics of financial time series, along with adapting optimizers like Adam and Adamax, can significantly improve forecasting accuracy while optimizing training efficiency. This approach surpasses the limitations observed in previous studies, where models based solely on LSTMs or CNN-LSTM [**29,30**] architectures did not always achieve optimal performance on extended time horizons or complex datasets. This study advances knowledge in financial time series

forecasting by highlighting the crucial role of architecture and optimization choices in improving the performance of deep learning models. Our results contribute to expanding the capabilities of both Transformers and LSTMs, paving the way for more robust and efficient hybrid approaches in financial forecasting.

III. METHODOLOGY

We will first describe two commonly used recurrent network architectures (LSTM and Transformer), then explain their use in financial time series forecasting.

III. 1 LONG SHORT-TERM MEMORY (LSTM)

An LSTM unit is composed of cells containing an input gate, an output gate, and a forget gate. These three gates regulate the flow of information. With these features, each cell can remember desired values over arbitrary time intervals [**4**]. The structure of the LSTM is shown in Figure .1. The structure of the LSTM consists of three main gates: a forget gate (**3.1**), an input gate (**3.2**), and an output gate (**3.3**). The forget gate, which controls the retention of information, can be expressed as follows:

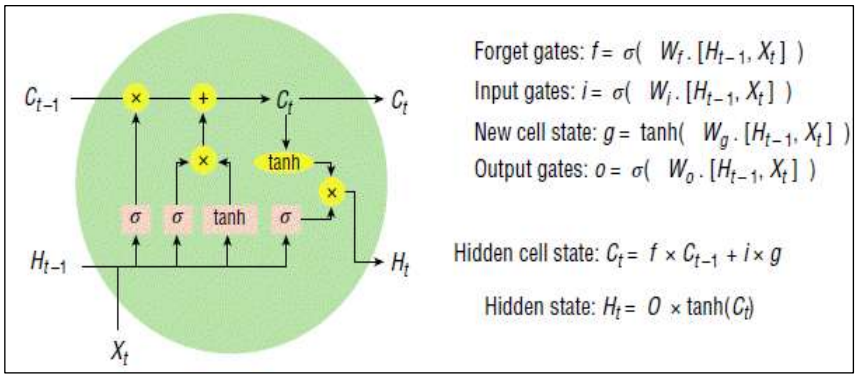

Figure 1: LSTM Diagram.
Source: Authors, (2025).

$$f_t = \sigma\big(W_t.[h_{t-1}, x_t] + b_f\big), \quad (Forgot\ Gate) \tag{3.1}$$
$$i_t = (W_i.[h_{t-1}, x_t] + b_i), \quad (Input\ Gate) \tag{3.2}$$
$$\tilde{c}_t = tanh(W_c.[h_{t-1}, x_t] + b_c), \quad (Cell\ State\ \ update) \tag{3.3}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \quad (New\ Cell\ State) \tag{3.4}$$
$$o_t = \sigma(W_0.[h_{t-1}, x_t] + b_0), \quad (Output\ \ Gate) \tag{3.5}$$
$$h_t = o_t \odot tanh(c_t). \quad (New\ Hidden\ \ State) \tag{3.6}$$

The output from the LSTM, as described in (3.6), is passed through fully connected (dense) layers for prediction:

$$z_1 = ReLU(W_1.ht + b_1), \quad (First\ Dense\ Layer) \tag{3.7}$$
$$y = W_2 . z_1 + b_2. \quad (Output\ Layer) \tag{3.8}$$

The model is trained by minimizing the Mean Squared Error (MSE), which is defined in (3.9):

$$Loss = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2. \tag{3.9}$$

Metrics such as Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) are used to evaluate performance. The MAE, given by (3.10), and the MAPE, given by (3.11), are calculated as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|. \tag{3.10}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100. \tag{3.11}$$

To prevent overfitting, dropout and early stopping are employed. Our baseline LSTM model consists of one LSTM layer with 200 units, followed by two dense layers with 50 and 1 neuron, respectively.

III. 2 TRANSFORMERS ARCHITECTURE

The architecture of the Transformer for financial time series prediction is shown in Figure. 2.

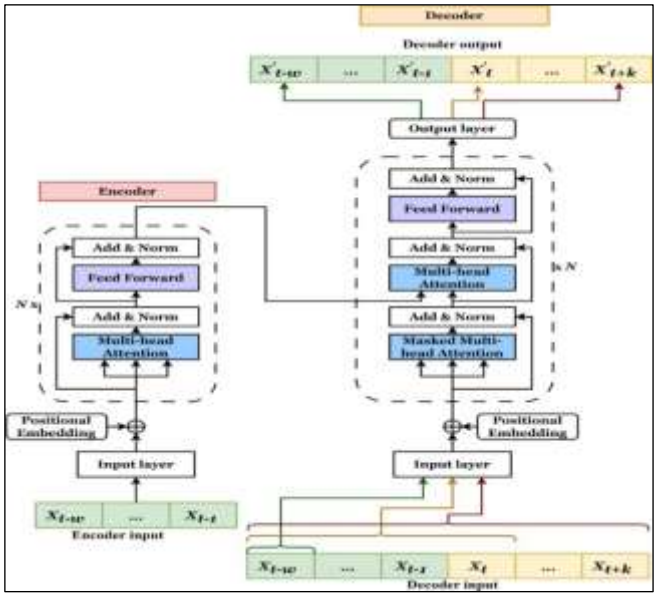

Figure 2: Architecture of the Transformer model for time series forecasting.
Source: Authors, (2025).

Thanks to attention mechanisms, distant elements can be connected directly, minimizing problems with gradients. Furthermore, Transformers solve the problem of network depth [**31**]. On the other hand, training, in practice, proves to require fewer epochs and, for each of them, the parallel treatment of the computation is much simpler to implement see [**32**]. The Transformer model leverages self-attention mechanisms to directly connect distant elements in sequential data, addressing the limitations of traditional RNNs. The self-attention mechanism is defined as:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3.12}$$

Where Q, K and V are the query, key, and value matrices, and d_k is the dimension of the key. Each Transformer block comprises:

- Layer Normalization:

$$LayerNorm(x) = \frac{x-\mu}{\sqrt{\sigma^2+\epsilon}}.\gamma + \beta. \tag{3.13}$$

- Feed-Forward Network:

$$FFN(x) = ReLU(xW_1 + b_1)W_2 + b_2. \tag{3.14}$$

- Residual Connections:

$$Output = LayerOutput + Input. \tag{3.15}$$

The architecture includes global average pooling and dense layers for output prediction. Dropout is applied throughout to enhance generalization. This model efficiently captures temporal dependencies and exhibits strong performance in financial time series forecasting. Building an LSTM for prediction requires using the Keras method; First, it performs a series of steps to initialize the neural network. Second, a dense layer is used to load the densely connected neural network, and an LSTM layer is added to incorporate long short-term memory. Finally, dropout is applied to prevent overfitting. Predicting future inventories when loading a test set requires a series of very important procedures [**33**]. Ray et al **[31]** introduced an innovative hybrid ARIMA-LSTM model. This model leverages a random forest algorithm to determine the appropriate input lags for the LSTM component.

IV. STATISTICAL ANALYSIS OF THE DATASET

The datasets analyzed include the closing prices of Microsoft (MSFT) and Nvidia (NVDA) over distinct time periods. Understanding the statistical properties of these financial time series provides context to the modelling challenges posed by their inherent characteristics. Below are the key statistical summaries for each dataset (see Table 1):

Table 1: Statistical Properties of Microsoft (MSFT) and Nvidia (NVDA) Closing Prices.

Statistic	Microsoft (MSFT)	Nvidia (NVDA)
Mean Closing Price	52.77	4.19
Standard Deviation	78.67	8.78
Skewness	2.32	3.03
Kurtoris	7.53	12.66

Source: Authors, (2025).

These statistics highlight the highly volatile nature of financial markets, characterized by non-normal distributions with significant skewness and kurtosis. The high kurtosis further suggests the potential for extreme values, which must be considered in predictive modelling.

The analysis of Bollinger Bands reveals significant volatility in Microsoft's stock prices over time. The lower band, with an average of 49.64, indicates support levels, while the upper band, averaging 55.30, represents resistance levels. The moving average, which averages 52.47, reflects the central tendency of the stock prices. Extreme values in the percentage B (pctB), ranging from -0.54 to 1.49, highlight occasional deviations beyond the bands, suggesting anomalies or speculative activities. Overall, the dataset exhibits a general upward trend punctuated by periods of high variability, illustrating the dynamic nature of financial time series (see Figure 3).

Figure 3: MSFT closing prices with Bollinger Bands, showing resistance (upper band), support (lower band), and moving averag.
Source: Authors, (2025).

Similarly, the Bollinger Bands analysis for Nvidia Corporation (NVDA) stock demonstrates notable volatility. The lower band averages 3.79, marking support levels, while the upper band averages 12.97, indicating resistance levels. The moving average of NVDA stock prices is 4.19, representing its central tendency. Extreme pctB (percentage Bandwidth) values range from -0.72 to 4.96, indicating occasional deviations beyond the bands that may signal anomalies or significant market events. The dataset shows an overall upward trend interspersed with periods of high volatility, underscoring the dynamic and unpredictable nature of NVDA's stock price (see Figure **4**).

Figure 4. NVDA closing prices with Bollinger Bands, showing resistance (upper band), support (lower band), and moving averag.
Source: Authors, (2025).

In this study, we first establish classes that make it easier to convert unstructured financial time series data into a format suitable for training machine learning models, particularly those that require sequential inputs like LSTMs. This approach transforms our problem into a supervised learning problem. These classes provide consistency, efficiency, and scalability in dataset preparation by automating extraction, transformation, and windowing. Our proposed model is a sequential architecture consisting of an LSTM layer with 200 units, followed by two Dense layers with 50 and 5 neurons, respectively. The LSTM layer captures temporal dependencies in sequential data, while the dense layers reduce dimensionality to produce the final output. With 515,717 parameters (171,905 trainable), our model is trained for 25 epochs with a batch size of 32, minimizing MSE and using MAE and MAPE as performance metrics. Early stopping is employed to prevent overfitting, making the model well-suited for tasks such as financial time series prediction or multi-class classification (see Table 2).

Table 2: Summary of the Transformer model architecture with parameter details.

Layer (Type)	Output Shape	Parameters
lstm(LSTM)	(None, 200)	161,6
dense (Dense)	(None, 50)	10,05
dense_1 (Dense)	(None, 5)	255
Total parameters:		515,717
Trainable parameters:		171,905
Non-trainable parameters:		0
Optimizer parameters:		343,812

Source: Authors, (2025).

The Transformer Encoder uses layer normalization, multi-head attention, residual connections, and feed-forward networks with 1D convolutional layers and dropout for regularization. The build_transformer function stacks Transformer blocks, applies global average pooling, and ends with dense layers for output prediction. Key parameters include a head size of 128, four attention heads, a feed-forward dimension of 2, four Transformer blocks, and a dropout rate of 0.10. The model is trained with MSE loss using the Adam optimizer (learning rate: 0.001) and evaluated with MAE and MAPE.

Training runs for 25 epochs with early stopping to prevent overfitting. The hybrid architecture combines multi-head attention and 1D convolutional layers to process sequential data. It starts with an input layer, followed by layer normalization and multi-head attention to capture long-range dependencies. Convolutional layers and dropout are applied for feature extraction and regularization, while residual connections improve gradient flow. Global average pooling reduces dimensionality before dense layers produces the final output. With 51,617 parameters (17,205 trainable), the model is ideal for tasks like time series forecasting, balancing local and global dependencies (see Table **3**).

Table 3: Summary of the Transformer model architecture with parameter details.

Layer (Type)	Output Shape	Para.	Connected to
Input_layer_1	(None, 5, 1)	0	-
multi_head_attention	(None, 5, 1)	3,585	Input_layer_1
conv1d	(None, 5, 2)	4	multi_head_attention
conv1d_1	(None, 5, 1)	3	conv1d
conv1d_2	(None, 5, 2)	4	conv1d_1
conv1d_3	(None, 5, 1)	3	conv1d_2
dense_3	(None, 5)	1,285	conv1d_3
Total parameters:		51,617	
Trainable parameters:		17,205	
Non-trainable parameters:		0	
Optimizer pparameters:		34,412	

Source: Authors, (2025).

For our Transformer architecture, we base the design on the configuration recommended in the Keras documentation, which is originally tailored for classification tasks. However, we made a key modification: the activation function of the final output layer was changed from softmax to ReLU, and the loss function was switched to MSE to better align with the nature of the problem. In addition to this adjustment, we finetuned the hyperparameters based on experimental results to optimize model performance for this specific task.

V. EXPERIMENTATION RESULTS AN D EVALUATION

Various optimizers were utilized to train the neural network models. Adam combines Adagrad (Adaptive Gradient) and RMSProp, dynamically adjusting learning rates for each parameter [**28**]. Adamax, a variant of Adam, is effective for sparse gradients. RMSProp (Root Mean Square Propagation) is designed for non-stationary data, addressing diminishing learning rates. Adadelta dynamically adapts learning rates, eliminating the need for a fixed rate [**35**]. Adagrad (Adaptive Delta) adjusts learning rates based on the historical sum of squared gradients. Nadam (Nesterov-accelerated Adaptive Moment Estimation) combines Nesterov's accelerated gradient with Adam for faster convergence [**36**]. The performance of Long Short-Term Memory (LSTM) and Transformer models was evaluated using Microsoft (MSFT) and NVIDIA (NVDA) stock datasets. The choice of optimizer significantly impacted the models' performance. Three key metrics were employed to evaluate the models: RMSE, MAPE. The variance ratio was included as an additional evaluation metric.

V. 1. COMPARISON USING VARIOUS OPTIMIZERS FOR THE FIRST SERIES (MSFT)

This subsection presents the results of evaluating LSTM and Transformer models on the MSFT dataset using various optimizers. Lower MAPE values indicate better forecasting accuracy, while variance and RMSE provide additional insights into model performance and stability. The performance metrics for different optimizers, including MAPE, variance, RMSE, and inference times, are summarized in Tables 4,5 ,6, and 7, respectively.

Table 4: MAPE values for various optimizers on the MSFT training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	0.03382	0.03398	0.048708	0.031500	0.028346	0.026310
TRANSF.	0.02309	0.025436	0.030663	0.382570	0.021955	0.023650

Source: Authors, (2025).

Table 4 compares the LSTM and Transformer models for MSFT stock data forecasting, assessed using the MAPE. The results highlight the overall superior performance of the Transformer model. The Adagrad optimizer achieved the best MAPE for both models, with values of 0.028346 for LSTM and 0.021955 for the Transformer.

Table 5: Var values for various optimizers on the MSFT training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	0.111730	0.095078	0.133927	0.004376	0.026858	0.012855
TRANSF.	0.026778	0.035896	0.048417	0.668279	0.004472	0.027254

Source: Authors, (2025).

Table **5** illustrates variance values for different optimizers on the MSFT training dataset using LSTM and Transformer models. For LSTM, Adagrad yields a low variance (0.026858) and performs consistently well, while Adadelta achieves the lowest variance (0.004376) but fails for the Transformer with the highest variance (0.668279). Adagrad proves to be the best optimizer overall, achieving the lowest variance for the Transformer (0.004472) and maintaining stability across both models. Other optimizers show varying degrees of variance, with RMSprop having the highest for LSTM.

Table 6: RMSE values for various optimizers on the MSFT training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	6.92941	7.99667	6.77863	8.83112	15.01159	8.68687
TRANSF.	6.68984	6.82673	6.54173	129.54425	7.18775	6.95662

Source: Authors, (2025).

Table **6** compares the RMSE values of LSTM and Transformer models across various optimizers for the MSFT training dataset. The Transformer consistently outperforms LSTM, achieving lower RMSE values with Adam (6.68984), Adamax (6.82673), RMSprop (6.54173), Adagrad (7.18775), and Nadam (6.95662). However, LSTM performs better with Adadelta, achieving 8.83112 compared to the Transformer's significantly higher RMSE of 129.54425. This underscores the importance of optimizer selection, with Transformers showing superior and more stable performance overall.

Table 7: Inference times for different optimizers on the MSFT training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	20.74728	18.58163	22.88478	23.25408	22.49023	22.24550
TRANSF.	20.05767	21.41631	24.01012	25.90485	23.71948	23.52633

Source: Authors, (2025).

Table **7** shows that the LSTM model trains fastest with Adamax (18.58 seconds), followed by Nadam and Adam. For the Transformer, Adam is the optimal optimizer with the shortest training time (20.06 seconds), followed by Adagrad and Nadam. These results highlight that LSTM achieves faster training with specific optimizers, while the Transformer shows slightly longer but consistent training times.

V. 2. COMPARISON USING VARIOUS OPTIMIZERS FOR THE SECOND SERIES (NVDA)

Before analyzing the detailed performance of LSTM and Transformer models with various optimizers on the NVDA dataset, it is important to provide a brief overview of the evaluation approach. This subsection examines metrics such as MAPE, RMSE, variance, and inference times to assess and compare model performance. The results aim to highlight the impact of optimizer selection on predictive accuracy and computational efficiency in the context of time series forecasting applied to NVDA data. We begin by comparing the performance of each model using the metric, MAPE.

Table 8: MAPE values for various optimizers on the NVDA training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	0.098944	0.141417	0.8429064	0.845593	0.322159	0.059463
TRANSF.	0.045220	0.041582	0.033147	0.695001	0.047531	0.049984

Source: Authors, (2025).

Table **8** shows that the Transformer achieves the lowest MAPE (0.033147) with RMSprop, while LSTM performs best with Nadam (0.059463). For Nadam, the Transformer has a slightly higher MAPE (0.049984) than LSTM. Overall, the Transformer demonstrates greater robustness across optimizers.

Table 9**:** Var values for various optimizers on the NVDA training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	0.374856	0.141196	0.017813	0.961713	0.122696	0.027150
TRANSF.	0.049727	0.020455	0.056631	0.957959	0.027713	0.069556

Source: Authors, (2025).

Table **9** shows that for LSTM, RMSprop achieves the lowest variance (0.017813), while Adadelta displays the highest variance (0.961713). Adagrad and Nadam also perform well, with relatively low variance values. For the Transformer, Adamax (0.020455) and

Adagrad (0.027713) yield the lowest variance, while Adadelta (0.957959) again exhibits the highest variance. Both models show consistent performance with Adamax and Adagrad but struggle with Adadelta.

Table 10: RMSE values for various optimizers on the NVDA training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	11.372042	8.604392	3.768150	44.840153	34.692550	14.44011
TRANSF.	3.173453	2.974158	3.447670	38.953478	3.168120	2.851640

Source: Authors, (2025).

Table **10** highlights that the Transformer performs better than LSTM for the RMSE metric on the NVDA series. The Transformer achieves its lowest RMSE with Nadam (2.8516) and Adamax (2.974158), while LSTM's best RMSE is higher at 3.76815 with RMSprop. Overall, the Transformer consistently delivers lower RMSE values across most optimizers, demonstrating its superiority.

Table 11: Inference times for different optimizers on the NVDA training dataset.

MODEL	ADAM	ADAMAX	RMSPROP	ADADELTA	ADAGRAD	NADAM
LSTM	15.63792	16.28799	23.23479	20.133928	16.62776	18.66561
TRANSF.	16.15776	15.414918	25.02268	20.795882	16.29275	14.93296

Source: Authors, (2025).

Table 11 highlights the inference times for LSTM and Transformer models across various optimizers on the NVDA training dataset. The LSTM achieves the lowest inference time with Adam (15.63792), while the Transformer is fastest with Nadam (14.93296). Despite some variation, both models show comparable inference times across most optimizers, except for RMSprop, where both exhibit higher inference times, with the Transformer slightly slower (25.02268).

V.3. RESULTS SUMMARY AND EVALUATION

The comparative analysis of our proposed models, LSTM, and Transformer, across various optimizers demonstrates distinct advantages of the Transformer regarding accuracy and robustness, particularly with metrics like MAPE, variance, and RMSE. The Transformer consistently achieved lower MAPE and RMSE values in most scenarios, affirming its superiority in capturing complex temporal patterns in stock price forecasting. Among the optimizers, Adagrad and Adamax emerged as the most effective, ensuring stable performance and faster convergence. However, certain limitations were observed, such as the Transformer's occasional instability with optimizers like Adadelta.

On the other hand, LSTM models showed adaptability with optimizers such as Nadam and Adadelta in specific contexts, achieving competitive results. These findings highlight the importance of selecting the right optimizer to balance model complexity, computational efficiency, and prediction accuracy. Overall, the experimentation confirms that the Transformer model, when paired with optimizers like Adamax or Adagrad, offers a promising financial time series forecasting approach, with the potential to outperform traditional LSTM models in accuracy, stability, and scalability. Future optimizations in hyperparameter tuning and the integration of additional market features could further enhance the predictive capabilities of these models.

VI.CONCLUSION AND FUTURE WORK

This study addresses the challenges of forecasting financial time series, specifically focusing on stock price prediction using Transformer-based and LSTM-based models. The findings highlight the critical impact of optimizer choice on model performance. For instance, optimizers such as Adam and Adamax significantly enhance training efficiency and prediction accuracy, underscoring the need to align model architecture and optimizer selection with the dataset's characteristics and the specific requirements of the forecasting task. Moreover, Transformer-based models excel in capturing complex, long-term temporal dependencies, while LSTM models perform effectively in scenarios requiring adaptation to specific optimization techniques. These results emphasize the importance of a tailored approach to model selection and hyperparameter tuning to achieve optimal forecasting accuracy. From a practical perspective, these models hold substantial potential for improving decision-making in financial contexts, such as algorithmic trading, risk management, and investment strategy optimization. Their ability to identify both short- and long-term patterns in market data provides valuable tools for navigating volatile financial markets.

Future research should aim to deepen these findings by exploring more advanced model architectures, including hybrid approaches that combine the strengths of deep learning models (LSTM, GRU, and Transformer) with traditional models such as ARIMA, GARCH, or other statistical techniques. Such combinations could leverage the structural modelling capabilities of traditional methods while benefiting from the ability of deep learning models to capture non-linear and complex relationships. Additionally, expanding training datasets to include macroeconomic variables, such as inflation rates, interest rates, and GDP growth, could enhance model robustness and forecasting accuracy. Applying these models in real-world scenarios, considering factors like market volatility, transaction costs, and dynamic trading strategies, would provide valuable insights into their practical effectiveness. Finally, integrating explainable AI (XAI) techniques could improve the interpretability and reliability of these models, making them more suitable for operational use in financial forecasting.

VII.AUTHOR'S CONTRIBUTION

Conceptualization: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Methodology: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.

Investigation: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Discussion of results: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Writing – Original Draft: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Writing – Review and Editing: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Resources: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Supervision: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.
Approval of the final text: Rabia Nichani, Laid Gasmi, Salheddine Kabou and Nabil Laiche.

VIII.ACKNOWLEDGMENTS

The authors would like to thank the editor and the anonymous referees for their careful comments and valuable suggestions that led to a substantial improvement of the presentation of the paper.

IX.REFERENCES

[1] H. Wasserbacher and M. Spindler, “Machine learning for financial forecasting, planning, and analysis: Recent developments and pitfalls,” Digit. Finance, vol. 4, no. 1, pp. 63–88, 2022.

[2] C. Zhang, N. N. A. Sjarif, and R. Ibrahim, “Deep learning models for price forecasting of financial time series: A review of recent advancements: 2020–2022,”Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 14, no. 1, p. e1519, 2024.

[3] T. Fischer and C. Krauss, “Deep learning with long short-term memory networks for financial market predictions,” Eur. J. Oper. Res., vol. 270, no. 2, pp. 654–669, 2018.

[4] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, …, and I. Polosukhin, “Attention is all you need,” Adv. Neural Inf. Process. Syst., vol. 30, 2017.

[6] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, “Transformers in time series: A survey,” arXiv preprint arXiv:2202.07125, 2022.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Adv. Neural Inf. Process. Syst., vol. 25, 2012.

[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, …, and M. Murray, “Language models are few-shot learners,” Adv. Neural Inf. Process. Syst., vol. 33, pp. 1877–1901, 2020.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.

[10] Z. Zhang and S. Zohren, “Multi-horizon forecasting for limit order books: Novel deep learning approaches and hardware acceleration using intelligent processing units,” arXiv preprint arXiv:2105.10430, 2021.

[11] Z. Zhang, S. Zohren, and S. Roberts, “Deeplob: Deep convolutional neural networks for limit order books,” IEEE Trans. Signal Process., vol. 67, no. 11, pp. 3001–3012, 2019.

[12] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting,” in Proc. Int. Conf. Mach. Learn., PMLR, pp. 27268–27286, 2022.

[13] G. Avinash, V. Ramasubramanian, M. Ray, R. K. Paul, …, and M. A. Iquebal, “Hidden Markov guided deep learning models for forecasting highly volatile agricultural commodity prices,” Appl. Soft Comput., vol. 158, p. 111557, 2024.

[14] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A survey of convolutional neural networks: Analysis, applications, and prospects,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 12, pp. 6999–7019, 2021.

[15] S. Meng, A. Chen, C. Wang, M. Zheng, F. Wu, X. Chen, H. Ni, and P. Li, “Enhancing exchange rate forecasting with explainable deep learning models,” arXiv preprint arXiv:2410.19241, 2024.

[16] J. Cao, Z. Li, and J. Li, “Financial time series forecasting model based on CEEMDAN and LSTM,” Physica A, vol. 519, pp. 127–139, 2019.

[17] S. Siami-Namini, N. Tavakoli, and A. S. Namin, “A comparison of ARIMA and LSTM in forecasting time series,” in Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA), pp. 1394–1401, 2018.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” Adv. Neural Inf. Process. Syst., vol. 27, 2014.

[19] X. Wen and W. Li, “Time series prediction based on LSTM-attention-LSTM model,” IEEE Access, 2023.

[20] X. Zhang, X. Liang, A. Zhiyuli, S. Zhang, R. Xu, and B. Wu, “At-LSTM: An attention-based LSTM model for financial time series prediction,” in IOP Conf. Ser.: Mater. Sci. Eng., vol. 569, no. 5, p. 052037, 2019.

[21] A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, “Using deep learning to detect price change indications in financial markets,” in Proc. 25th Eur. Signal Process. Conf. (EUSIPCO), pp. 2511–2515, 2017.

[22] P. N. Kolm, J. Turiel, and N. Westray, “Deep order flow imbalance: Extracting alpha at multiple horizons from the limit order book,” Math. Finance, vol. 33, no. 4, pp. 1044–1081, 2023.

[23] M. Ranjbar and M. Rahimzadeh, “Advancing gasoline consumption forecasting: A novel hybrid model integrating transformers, LSTM, and CNN,” arXiv preprint arXiv:2410.16336, 2024.

[24] R. Nichani, L. Gasmi, N. Laiche, and S. Kabou, “Optimizing financial time series predictions with hybrid ARIMA, LSTM, and XGBoost models,” Stud. Eng. Exact Sci., vol. 5, no. 2, p. e11188, 2024.

[25] X. Hu, “Stock price prediction based on temporal fusion transformer,” in Proc. Int. Conf. Mach. Learn. Big Data Bus. Intell. (MLBDBI), pp. 60–66, 2021.

[26] S. Sridhar and S. Sanagavarapu, “Multi-head self-attention transformer for Dogecoin price prediction,” in Proc. 14th Int. Conf. Hum. Syst. Interact. (HSI), pp. 1–6, 2021.

[27] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

[28] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[29] C. Han, H. Park, Y. Kim, and G. Gim, “Hybrid CNN-LSTM based time series data prediction model study,” in Proc. IEEE/ACIS Int. Conf. Big Data Cloud Comput. Data Sci. Eng., pp. 43–54, 2022.

[30] D. Patnaik, N. J. Rao, B. Padhiari, and S. Patnaik, “Optimised hybrid CNN-LSTM model for stock price prediction,” Int. J. Manag. Decis. Mak., vol. 23, no. 4, pp. 438–460, 2024.

[31] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp. 770–778, 2016.

[32] G. H. Nayak, M. W. Alam, G. Avinash, R. R. Kumar, M. Ray, S. Barman, …, and J. Bisen, “Transformer based deep learning architecture for time series forecasting,” Software Impacts, vol. 22, 2024.

[33] F. Altché and A. de La Fortelle, “An LSTM network for highway trajectory prediction,” in Proc. 20th IEEE Int. Conf. Intell. Transp. Syst. (ITSC), pp. 353–359, 2017.

[34] S. Ray, A. Lama, P. Mishra, T. Biswas, S. S. Das, and B. Gurung, “An ARIMA-LSTM model for predicting volatile agricultural price series with random forest technique,” Appl. Soft Comput., vol. 149, p. 110939, 2023.

[35] N. Zhang, D. Lei, and J. F. Zhao, “An improved Adagrad gradient descent optimization algorithm,” in Proc. Chinese Autom. Congr. (CAC), pp. 2359–2362, 2018.

[36] J. Sharma, S. Soni, P. Paliwal, S. Saboor, P. K. Chaurasiya, …, and A. Afzal, “A novel long term solar photovoltaic power forecasting approach using LSTM with Nadam optimizer: A case study of India,” Energy Sci. Eng., vol. 10, no. 8, pp. 2909–2929, 2022.