FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 10 No. 3, February, 2026, pp 220 – 231
DOI: https://doi.org/10.33003/fjs-2026-1003-4723

RUNGE–KUTTA DUAL ATTENTION OPTIMIZATION FOR LSTM-BASED FINANCIAL TIME-SERIES FORECASTING: STABILITY, EFFICIENCY, AND ROBUSTNESS

***David Opeoluwa Oyewola, Sulaiman Awwal Akinwunmi and Joel John Taura**

Department of Mathematics and Statistics, Federal University of Kashere, Gombe State, Nigeria.

*Corresponding authors' email: davidakaprof01@yahoo.com

ABSTRACT

This study evaluates the performance of the Runge–Kutta Dual Attention (RUN-DA) optimization framework for hyperparameter tuning in a dual-attention Long Short-Term Memory (LSTM) model for financial time-series forecasting. The experiment was conducted using historical stock price data of MRS Oil Plc covering the period 2012–2024, representing a Nigerian financial market dataset. The proposed optimizer was compared with the Genetic Algorithm (GA) and Brown Bear Optimization Algorithm (BBOA) under consistent experimental conditions. Model performance was assessed using validation Mean Squared Error (MSE) and computational efficiency. Results show that RUN-DA achieved the lowest mean fitness value of 0.1369, compared with 0.4054 for GA and 0.1924 for BBOA. Sensitivity analysis indicated that moderate learning rates and time-step values produced more stable generalization performance. The evaluation under different market regimes further showed that RUN-DA maintained relatively lower fitness values across volatility conditions, decreasing from 0.2343 in low-volatility periods to 0.0473 in high-volatility periods, while GA and BBOA recorded higher corresponding values. In terms of computational efficiency, RUN-DA converged in 1015.23 seconds, slightly faster than GA (1073.25 seconds) and substantially faster than BBOA (2147.05 seconds). These results suggest that RUN-DA provides an effective optimization approach for improving LSTM-based financial forecasting models, although further validation on additional financial assets and evaluation metrics is recommended.

Keywords: Runge–Kutta Optimization, Dual-Attention LSTM, Hyperparameter Tuning, Genetic Algorithm (GA), Brown Bear Optimization Algorithm (BBOA), Hyperparameter optimization, Metaheuristic Optimization, Financial Volatility Forecasting

INTRODUCTION

Financial time-series forecasting (Oyewola, *et al.,* 2025) is one of the many obstacles in quantitative finance because of the inherent nonstationarity, noise and volatility of the financial markets; accurate modelling of the data is important for portfolio optimisation (Salo *et al*., 2024), risk management (Ghazieh & Chebana, 2021) and algorithmic trading (Bhuiyan *et al.*, 2025). Traditional statistical models that use linearity and stationarity as restrictive assumptions have limited effectiveness in forecasting due to the complexity of the temporal dependencies, abrupt shifts in regime and nonlinear dynamics that can be observed in financial time series (Fu, 2025). On the other hand, LSTM networks (Fischer & Krauss, 2018) provide a very strong ability to capture long-range temporal dependencies in sequential data, therefore LSTM based models have received widespread acceptance for financial forecasting. Additionally, to help improve predictive performance of LSTM Networks, researchers have introduced attention mechanisms into the architecture of LSTMs (Xiao, 2025) which enable the model to focus on specific, and the most relevant, time-series segments and/or input features. Although significant improvements have been realised with respect to LSTM networks, parameter selection of LSTM networks continues to play a critical role in how effective they will be on a particular forecasting task. Many parameters, including learning rate, time-step length and configuration of the network, all have a significant impact on the speed of convergence, stability of training, and generalisation of performance of the LSTM network. Recently, (Oyewola *et al*., 2026) introduced a Runge–Kutta Optimization–based Dual-Attention LSTM (RUN-DA-LSTM) framework, demonstrating that numerically inspired optimization strategies can effectively guide hyperparameter tuning in deep learning models (Oziegbe, *et al.*, 2026). Recent studies have increasingly applied metaheuristic optimization techniques (Zhao, *et al*., 2025) for deep learning hyperparameter tuning, demonstrating improved search efficiency compared with traditional grid or manual methods. Approaches such as Particle Swarm Optimization (PSO) (Imran, *et al*, 2013) and Bayesian Optimization (Paulson & Tsay, 2025) have been successfully applied to optimize neural network configurations and improve predictive performance in complex forecasting tasks.

By leveraging principles derived from Runge–Kutta numerical integration, the RUN algorithm provides a structured and stable search mechanism capable of balancing exploration and exploitation. Their study demonstrated that the RUN-DA-LSTM framework achieves improved forecasting accuracy and stable training behavior compared with baseline LSTM and GRU models, underscoring its effectiveness for modeling complex financial time-series data. Building upon this prior work, the present study does not propose a new optimization algorithm or model architecture. Instead, it provides an extended and in-depth empirical evaluation of the RUN-DA-LSTM framework, with particular emphasis on convergence dynamics, robustness, sensitivity, and computational efficiency. While the original study established the feasibility and effectiveness of RUN-DA-LSTM, a detailed comparative analysis of its optimization behavior under varying conditions—such as different market volatility regimes and repeated stochastic initializations—remains limited. Accordingly, this work expands the analysis of RUN-DA-LSTM by systematically comparing its convergence characteristics and stability against widely used metaheuristic optimizers, namely the Genetic Algorithm (GA) and the Brown Bear Optimization Algorithm (BBOA). In addition, comprehensive sensitivity analyses are conducted to examine the influence of critical hyperparameters, including learning rate and time-step length, on model generalization. The study further investigates the adaptability of the framework under low- and high-volatility market conditions, providing insights into its robustness in realistic, non-stationary financial environments. Computational efficiency is also assessed to evaluate the practical deployability of the approach.

The key contributions of this study are therefore empirical and analytical rather than methodological. Specifically, this work:

i. provides a detailed convergence and stability analysis of the RUN-DA-LSTM framework;
ii. evaluates its robustness across multiple independent runs and volatility regimes;
iii. examines hyperparameter sensitivity to support informed configuration choices; and
iv. assesses the trade-off between optimization performance and computational cost relative to established metaheuristic methods.

Related Works

Due to its ability to learn how different types of financial data interact over time, deep learning techniques are becoming popular for predicting future trends in a variety of areas related to finance. Within the family of deep learning methods, one of the best is the use of recurrent neural networks (RNNs) and their variations (such as LSTM and GRU) that are tailored to learn more complex patterns than typical RNNs. Compared to traditional methods for predicting future stock prices, estimating volatility, and estimating returns on investments, LSTMs and other gated recurrent networks outperform all previous methods (Oyewola *et al*., 2024). The reason that LSTMs work so well is that they are designed to manage the vanishing gradient problem common to conventional RNN architectures; this allows them to maintain long-term relationships when working with data over time (Hochreiter and Schmidhuber, 1997).

In a study conducted by (Yan & Ouyang, 2018), a hybrid approach utilizing deep learning techniques to predict future financial time series was analyzed that used wavelet analysis and Long Short-Term Memory networks (LSTMs). According to the study's authors, by using wavelet decomposition to pre-process financial price time series, they were able to identify important trends and cycles from the nonlinear and non-stationary characteristics of financial price series. After the signals were processed, they were modeled using LSTMs, enabling them to capture both temporal dependencies and sequential correlations. The authors tested their hybrid framework on daily closing prices of the Shanghai Composite Index (SSE), comparing their results with traditional ML methods (i.e., multilayer perceptrons, support vector machines and k-nearest neighbors). The results showed that LSTM-based model had greater predictive ability than all other models, especially in terms of both short-term and long-term price movements. In addition, their results demonstrated that the use of wavelet decomposition and reconstruction significantly improved the LSTM model generalization and increased the model's forecasting accuracy over long horizons. Therefore, the study shows that by incorporating both the techniques of signal processing and deep learning, better models for financial time series prediction can be created.

A comprehensive review by (Sezer *et al*., 2020)) examined the evolution of financial time-series forecasting from traditional machine learning approaches to modern deep learning methodologies. Their study highlighted that financial forecasting has become a central application of computational intelligence in both academic research and industry practice due to its wide-ranging use cases and economic significance. While earlier studies predominantly focused on conventional machine learning models, the review emphasized the growing dominance of deep learning techniques, which have consistently demonstrated superior predictive performance. The authors systematically categorized existing deep learning–based forecasting studies according to application domains, including stock indices, foreign exchange, and commodity markets, as well as by model architecture, such as convolutional neural networks, deep belief networks, and long short-term memory networks. In addition to synthesizing prior work, the review identified key challenges and open research directions, outlining potential limitations and future opportunities for advancing deep learning–driven financial forecasting research.

A recent study (He *et al*., 2023) examined forecasting of financial time series used a combination of: convolutional neural networks (CNN) and long short-term memory networks (LSTM) combined in an ensemble framework to build on the predictive capabilities of both NN's (CNN + LSTM) to learn complex spatial and temporal patterns, and Autoregressive Moving Average (ARMA) models which allowed an account of the linear autocorrelation structures of the time series data. The use of both complementary models through an ensemble architecture allowed for improved representation of heterogeneous characteristics of the financial time series by using the combined outputs of the three component model types as the input for the ensemble architecture compared to using only the outputs of each separate model type. Additional empirical evaluations performed on various financial time-series dataset have consistently shown that the ensemble model produces superior results in terms of predictive accuracy and robustness when compared to individual component models of CNN, LSTM, and ARMA and highlight the various advantages of hybrid deep learning methods in performing financial forecasting tasks.

In a recent study conducted by (Zhang & Sjarif *et al*., 2024), the increasing popularity of using deep-learning methods for financial time series price prediction relative to more traditional statistical and machine learning methods was looked at. With an unprecedented rise in the number of new methodologies developed, the authors reviewed the literature from 2020 - 2022 to provide a synthesis of current methods and trends for both researchers and practitioners. The authors also analyzed several of the key components of deep learning forecasting architecture and provided examples of where they have been used and how they compare with older methods in terms of strengths and weaknesses. Particular attention was given to advanced modeling paradigms, including Transformers, generative adversarial networks, graph neural networks, and deep quantum neural networks, reflecting the diversification of deep learning strategies in financial forecasting. Additionally, the study outlined several promising research directions, such as moving beyond point forecasts toward interval prediction, assessing the robustness of decomposition-based ensemble models, evaluating the impact of data scale on predictive performance, and exploring increasingly complex model structures. These insights highlight both the progress and the open challenges in deep learning–based financial price prediction.

A study by (Yang, 2021) introduced a deep learning–based framework aimed at forecasting financial indicators under conditions of potential internal and external risks faced by financial organizations. The framework adopted long short-term memory networks as a baseline predictive model to estimate key financial indicators, including return on tangible assets, price-to-sales ratio, and price–earnings ratio. Empirical validation was conducted using stock data from Mondelez International and Hormel Foods, demonstrating the model's ability to capture fluctuations associated with varying economic conditions. By linking indicator forecasts to risk assessment, the framework provided early warnings of financial instability, highlighting the applicability of deep learning methods for proactive financial risk monitoring across organizations of different scales.

A study by (Bao *et al.*, 2017) proposed an integrated deep learning framework for stock price prediction that combines signal decomposition and hierarchical feature learning. Their approach employed wavelet transforms to reduce noise in raw price series, followed by stacked autoencoders to extract high-level representations from the denoised data. These learned features were subsequently passed to a long short-term memory network to model temporal dependencies and forecast next-day closing prices. The framework was evaluated using multiple stock market indices and their corresponding futures contracts, where it demonstrated superior predictive accuracy and trading performance compared with several benchmark models. The findings underscored the effectiveness of coupling feature extraction, noise reduction, and sequence modeling in financial time-series forecasting.

A study by (Zhang, et al., 2020) introduced a hybrid deep learning framework for stock market forecasting that integrates signal decomposition, dimensionality reduction, and sequence learning. Their proposed CEEMD–PCA–LSTM model first applied complementary ensemble empirical mode decomposition to separate financial time series into multiple intrinsic mode functions representing trends and fluctuations at different temporal scales. Principal component analysis was then employed to reduce redundancy among the decomposed components while preserving the dominant information content, thereby enhancing computational efficiency. Each reduced component was subsequently modeled using long short-term memory networks to capture temporal dependencies, and the individual predictions were aggregated to generate the final forecast. Empirical evaluations across six representative stock indices from diverse market environments demonstrated that the hybrid model achieved lower forecasting errors, higher directional accuracy, and superior trading performance compared with benchmark approaches. Robustness analysis further confirmed the stability of the proposed framework under varying market conditions, underscoring the effectiveness of decomposition-based deep learning models for financial time-series prediction.

In the context of deep learning hyperparameter optimization, Runge–Kutta–based strategies have shown promising results in efficiently navigating high-dimensional, non-convex search spaces. Recent studies have proposed numerically inspired optimization strategies for tuning deep learning models in financial forecasting, including the integration of Runge–Kutta–based optimization with dual-attention LSTM architectures. These approaches have been shown to deliver smoother convergence and superior validation performance relative to conventional metaheuristic algorithms, owing to their stable and efficient search mechanisms. Motivated by these findings, the present study extends this line of research by conducting a comprehensive comparative analysis with evolutionary and nature-inspired optimizers and by assessing robustness across varying market environments.

MATERIALS AND METHODS

This chapter describes the methodological framework used to evaluate the effectiveness of numerically inspired optimization techniques for deep learning–based financial time series forecasting. The study focuses on forecasting stock price movements using historical market data from MRS Oil Nigeria Plc, a publicly traded company listed on the Nigerian Exchange Limited (NGX). The methodological framework adopted in this study is derived from the RUN-DA-LSTM model previously introduced by Oyewola et al. (2026). The earlier work proposed a hybrid deep learning architecture that integrates Runge–Kutta Optimization with a Dual-Attention Long Short-Term Memory network for financial time-series prediction. While the core architecture of the model remains unchanged, the present study extends the previous research through a more comprehensive empirical investigation. Specifically, this work expands the experimental framework by incorporating additional comparative optimization algorithms, including Genetic Algorithm (GA) and Brown Bear Optimization Algorithm (BBOA). Furthermore, the study introduces volatility-based performance analysis to evaluate model behavior under different market conditions. To ensure statistical reliability and robustness, repeated experimental runs with multiple random seeds are conducted, enabling detailed analysis of convergence stability and sensitivity to stochastic initialization. Through these methodological extensions, the study provides deeper insights into the practical effectiveness and robustness of the RUN-DA-LSTM framework when applied to financial forecasting tasks.

Dataset Description

The dataset used in this study consists of historical stock market data obtained for MRS Oil Nigeria Plc, a publicly traded energy marketing company listed on the Nigerian Exchange Limited (NGX). The data provides a reliable financial time-series suitable for evaluating forecasting models and optimization strategies under different market conditions.

Data Source

The historical stock price data were obtained from publicly accessible financial market repositories and the official trading records of the Nigerian Exchange. These sources provide verified daily trading information for listed companies.

Selected Asset

The study focuses specifically on the stock of MRS Oil Nigeria Plc, which was selected due to its observable price dynamics and market fluctuations. These characteristics make it an appropriate case study for assessing the effectiveness of deep learning–based financial forecasting models.

Time Period

The dataset covers a 13-year period from 2012 to 2024, providing a sufficiently long time horizon for capturing both long-term market trends and short-term fluctuations. The extended temporal coverage allows the model to learn various market patterns, including stable periods and episodes of heightened volatility.

Data Preprocessing and Preparation

To ensure numerical stability and improve model convergence, several preprocessing steps were applied.

Data Cleaning

Missing values and irregular records were inspected and removed. Duplicate entries were also eliminated to maintain consistency.

Data Normalization

All input variables were normalized using Min-Max scaling, which transforms each variable into the range [0,1]:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where

i. x = original value
ii. x_{min} = minimum value in the dataset
iii. x_{max} = maximum value in the dataset.

Normalization ensures that features with larger magnitudes do not dominate the training process.

Dual-Attention Long Short-Term Memory (DA-LSTM) Architecture

To effectively capture nonlinear temporal dependencies and varying feature importance in financial time-series data, this study employs a Dual-Attention Long Short-Term Memory (DA-LSTM) architecture. The model integrates two complementary attention mechanisms: feature-level attention and temporal attention, allowing the network to dynamically emphasize the most relevant input variables and time steps during prediction.

The DA-LSTM architecture is built upon the Long Short-Term Memory (LSTM) neural network, a specialized form of Recurrent Neural Network (RNN) designed to address long-term dependency problems in sequential data.

The proposed architecture consists of three major components:

i. Input feature attention layer
ii. LSTM temporal modeling layer
iii. Temporal attention layer

These components collectively enable the model to learn both cross-feature relationships and long-range temporal dependencies.

Long Short-Term Memory Network

The LSTM network is used to model sequential relationships in financial data. Unlike traditional recurrent neural networks, LSTM incorporates gating mechanisms that regulate information flow through the network.

At time step t, given an input vector x_t and previous hidden state h_{t-1}, the LSTM computes the following operations.

Forget Gate

The forget gate determines which information from the previous cell state should be discarded.

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \quad (2)$$

where:

i. f_t = forget gate vector
ii. W_f = weight matrix
iii. b_f = bias term
iv. σ = sigmoid activation function

Input Gate

The input gate controls how much new information should be stored in the cell state.

$$i_t = \sigma(W_i\left[h_{t-1}, x_t\right] + b_i) \quad (3)$$

The candidate memory content is computed as:

$$\widetilde{C_t} = \tanh[W_c[h_{t-1}, x_t] + b_c] \quad (4)$$

Cell State Update

The new cell state is obtained by combining the previous state and the candidate memory:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \quad (5)$$

Where $\odot$ represents element-wise multiplication.

Output Gate

The output gate determines the hidden state passed to the next time step.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (6)$$

The hidden state is then computed as:

$$h_t = o_t \odot \tanh(C_t) \quad (7)$$

The sequence of hidden states h_t captures the temporal representation of the financial time series.

Feature Attention Mechanism

Financial datasets often contain multiple explanatory variables whose importance varies over time. To address this challenge, a feature-level attention mechanism is applied to dynamically assign weights to input variables.

Let the input sequence be:

$$X = [x_1, x_2, \ldots, x_t] \quad (8)$$

where each $x_t \in \mathbb{R}^n$ represents the vector of n features at time step t.

The attention score for each feature is calculated as:

$$e_t^{(f)} = v_f^T \tan(W_f x_t + b_f) \quad (9)$$

The attention weights are then obtained using the softmax function:

$$\alpha_t = \frac{\exp(e_t^{(f)})}{\sum_{k=1}^{T} \exp(e_t^{(f)})} \quad (10)$$

The weighted input representation becomes:

$$\widetilde{x_t} = \alpha_t . x_t \quad (11)$$

This mechanism enables the model to focus more strongly on features that contribute significantly to stock price prediction.

Temporal Attention Mechanism

While the feature attention mechanism identifies the importance of variables, the temporal attention mechanism determines which historical time steps are most relevant for predicting future stock prices.

Let the hidden states produced by the LSTM layer be:

$$H = [h_1, h_2, \ldots, h_t] \quad (12)$$

The temporal attention score for each time step is computed as:

$$e_t^{(t)} = v_t^T \tanh(W_t h_t + b_t) \quad (13)$$

The corresponding temporal attention weights are obtained via softmax normalization:

$$\beta_t = \frac{\exp(e_t^{(t)})}{\sum_{k=1}^{T} \exp(e_k^{(t)})} \quad (14)$$

The final context vector is computed as a weighted combination of hidden states:

$$c = \sum_{t=1}^{T} \beta_t h_t \quad (15)$$

This context vector summarizes the most informative temporal patterns in the sequence.

Output Prediction Layer

The context vector produced by the temporal attention layer is passed through a fully connected layer to generate the final prediction.

$$\hat{y} = W_y c + b_y \quad (16)$$

where:

i. $\hat{y}$ represents the predicted stock price
ii. W_y is the output weight matrix
iii. b_y is the bias term.

Runge–Kutta Optimization Algorithm for Hyperparameter Tuning

To improve the predictive performance of the Dual-Attention LSTM model, this study employs the Runge–Kutta Optimization Algorithm (RUN) for hyperparameter tuning. The RUN algorithm is a population-based metaheuristic inspired by the classical Runge–Kutta Method, a well-known numerical technique originally developed for solving ordinary differential equations.

In optimization problems, RUN mimics the step-wise estimation strategy of the Runge–Kutta numerical integration process to guide candidate solutions toward optimal regions of the search space.

In this study, the RUN algorithm is used to optimize key hyperparameters of the Long Short-Term Memory-based dual-attention model, including:

i. number of LSTM units
ii. dropout rate
iii. learning rate
iv. batch size
v. time-step window length

The objective is to minimize the prediction error of the model on the validation dataset.

Optimization Problem Formulation

Let the hyperparameter vector be represented as:

$$X = [x_1, x_2, x_3, \dots, x_d] \tag{17}$$

where:

i. d represents the number of hyperparameters
ii. x_i denotes the value of the i^{th} hyperparameter.

The optimization objective is defined as:

$$\min f(X) \tag{18}$$

where $f(X)$ represents the prediction error of the DA-LSTM model measured using a chosen loss metric such as Mean Squared Error (MSE):

$$f(X) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{19}$$

where:

i. y_i is the actual stock price
ii. $\hat{y}_i$ is the predicted value
iii. *N* is the number of validation samples.

Initialization of Candidate Solutions

The RUN algorithm begins by generating a population of candidate solutions randomly within predefined parameter bounds.

For each candidate *i*, the initial position is defined as:

$$X_t = X_{min} + rand \times (X_{max} - X_{min}) \tag{20}$$

where:

i. X_{min} and X_{max} represent lower and upper bounds of the search space
ii. *rand* is a uniformly distributed random number in [0,1].

Each candidate solution corresponds to a specific configuration of the DA-LSTM hyperparameters.

Runge–Kutta Search Mechanism

The core idea of the RUN algorithm is based on the fourth-order Runge–Kutta approximation, which estimates the next state of a dynamic system using multiple intermediate slopes.

For a candidate solution X_i, the update process uses four intermediate vectors defined as:

$$k_1 = f(X_i) \tag{21}$$

$$k_2 = f(X_i + \tfrac{1}{2}k_1) \tag{22}$$

$$k_3 = f(X_i + \tfrac{1}{2}k_2) \tag{23}$$

$$k_4 = f(X_i + k_3) \tag{24}$$

The candidate solution is then updated using the weighted combination:

$$X_i^{new} = X_i + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{25}$$

Comparative Optimization Algorithms

Based on the results of the RUN-DA, two of the most common metaheuristic optimization algorithms were selected to act as baseline comparisons:

Genetic Algorithm (GA)

A commonly used method for evolutionary optimization based on the use of selection, crossover, and mutation operators; the GA (Whitley, 1994) keeps track of the progress of a population through its evolution process, which enables the GA to conduct a global search. However, the convergence time of the GA may take more time when using larger dimensional spaces.

Brown Bear Optimization Algorithm (BBOA)

The Brown Bear Optimization Algorithm (BBOA) (Prakash *et al.*, 2023)has been inspired by nature, representing an optimization algorithm based on the behaviour of brown bears when hunting for food and when moving throughout their habitats. The main focus of the BBOA is on exploiting resources; thus, it has very fast convergence times when compared to the other baseline comparisons. The main disadvantage of the BBOA is that it may not be able to explore all the possibilities in a complex search space.

In order to provide the most accurate representation of the performance between the RUN-DA and the baseline algorithms, all three algorithms were set up to have the same parameters for population size, search space boundaries, stopping conditions, and fitness evaluations.

Experimental Design and Data Handling

Normalization is applied to the financial time series data before any training process to produce numerically stable data for training purposes; the data set was split into separate training sets, validation sets, and test sets, with the validation set being used exclusively for optimization of hyperparameters. Generalization is evaluated through the model's performance on previously unseen data sets. The assessment of robustness throughout multiple runs of each optimization algorithm using various random seeds allows for a statistical assessment of convergence stability and sensitivity to stochastic initialization, while remaining within identical experimental settings. Evaluation of the optimization algorithm's ability to adapt to varying market conditions is achieved by segmenting the data into low volatility and high volatility regimes based on market variability metrics.

Sensitivity Analysis

Sensitivity analyses are conducted to examine the influence of key hyperparameters on model performance:

i. Learning Rate: Multiple learning rate values are evaluated to assess their impact on convergence stability and validation MSE.
ii. Time-Step Length: Tested configurations of different time-steps are evaluated to find the best temporal window for capturing sequential dependencies.

Various trials of each parameter are conducted, with the outputs evaluated for distributions of validation mean squared errors (MSE) and summarized through box plots that illustrate variability and robustness.

Evaluation Metrics

In this study, the primary evaluation metric is the Mean Squared Error (MSE), which can be expressed mathematically as follows:

$$MSE = \frac{1}{N}\sum_{I=1}^{N}(y_i - \widehat{y_i})^2 \quad (26)$$

where y_i is the actual values, $\widehat{y_i}$ is the predicted values, respectively, and N is the number of observations.

Mean Squared Error is not the only metric evaluated. The following additional metrics will also be used in order to allow for comparison:

i. Mean/Standard Deviation of Fitness Values for each run (for robustness of results),
ii. Convergence Speed (number of iterations to reach stable optimum) and
iii. Computational Time (seconds - measures how fast and how easily a solution can be implemented).

Analysis Procedure

The analysis proceeds in three stages:

i. Convergence Analysis: Evaluation of fitness trajectories across iterations to assess optimization dynamics.
ii. Stability and Robustness Analysis: Statistical comparison of final fitness distributions across multiple runs.
iii. Regime-Specific Evaluation: Performance assessment under low- and high-volatility market conditions.

RESULTS AND DISCUSSION

Figure 1 shows how the Runge-Kutta Optimisation Algorithm's route when applied to hyper-parameter tuning on the dual-attention long short-term memory (DA-LSTM) model. The horizontal axis illustrates the number of optimisation iterations, and the vertical axis represents the best fitness value (validation mean squared error) obtained at each of these iterations. At iteration zero, the Runge-Kutta Optimisation Algorithm has a high initial fitness value due to the random initialisation of the population of hyperparameters being explored. In iteration two, there is a large drop in fitness value, which demonstrates that the Runge-Kutta Algorithm is rapidly exploring and efficiently exploiting some regions of the solution space. The RUN-DA's ability to quickly escape from sub-optimal configurations is indicative of RUN-DA's strong ability to search globally. The convergence curve between iteration 2 & iteration 4 shows a slow slope meaning the algorithm is changing from global exploration of design space to local refinement of design space. Also, after I4, the fitness value has been relatively constant to the end of optimization indicating optimality of the algorithm and that it is unable to find further improvements in terms of fitness in the given design space. The smooth and monotonically decreasing nature of the Convergence Pattern without bouncing around or divergent points indicates the numerical stability of the RUN-DA combined with the characteristics of the Non-Convex Deep Learning Objective Function. Furthermore, there were no significant sudden changes in Fitness levels throughout the Optimization Process, which indicates that RUN-DA does not suffer from the premature Convergence Issue while continuing to exploit good quality solutions in a consistent manner.

In the context of hyperparameter optimization for the dual-attention LSTM framework, Figure 2 shows the convergence behaviour of the Genetic Algorithm (GA). As shown in the plot's x-axis, as time goes on the GA finds better and better fitness values (or validation mean squared errors), while at the y-axis the GA has become more and more consistent in finding better and better candidates. In the beginning of the GA optimising process, the GA begins with a very high level of fitness (i.e., validation MSE), because this was a randomised population over the search space (which includes the best candidates). Over time, the fitness value starts to decrease, which can be attributed to the application of GA operators selecting the best candidates from the population during selection and crossover, as well as by mutation. Rapid initial convergence towards optimum values has been seen in some optimisations methods, such as gradient based and numerical ones, but this has not been the case in this instance, where the GA's initial convergence has been relatively slow. The precipitous dips and subsequent level segmentations of the fitness curve from the time of Iteration 3 through Iteration 6 can also be attributed to the nature of genetic algorithms and their ability to generate significant improvements in population convergence/generation (meaningful improvements in validation MSE) only when favourable genetic combinations are present in the population; the fact that there are long segments of flat lines also indicates that GA is in a state of stagnation, during which either the genetic information being passed on to future generations has not improved over time or the genetic information produced by the GA is completely unsatisfactory for creating the fitness generating candidates needed for the optimised state. After Iteration 6, the fitness curve rapidly stabilises with no further significant changes for the remaining iterations of the optimised process. This plateau suggests that the GA has converged to a local optimum and exhibits limited exploitation capability in the later stages of the search. Such behavior is commonly associated with premature convergence in high-dimensional, non-convex optimization problems, particularly when applied to deep learning hyperparameter tuning.

The Brown Bear Optimization Algorithm (BBOA) shown in Figure 3 shows some interesting behaviour when performing hyperparameter optimization on the Long Short-Term Memory (LSTM) model. The number of iterations conducted along with their respective best fitness value has been tracked throughout the process and gives an idea on how well the BBOA will perform to optimise the hyperparameters of the LSTM model. Initially, the BBOA starts with relatively high fitness values (iteration 0) due to the random nature of the generations within the search area; consequently, high decreases in fitness values occur in the first few iterations of BBOA's implementation, indicating the algorithm is quickly identifying "good" sections of the solution space via exploration strategies. In comparison to other population-based metaheuristics, which typically exhibit a gradual/coarse stepwise convergence trajectory, the BBOA has a steep downwards descent of fitness near the initial start of optimisation, since the exploitation operators contained within the algorithm's implementation emphasise exploitation over exploration. The fitness curve is beginning to be in the vicinity of optimum by iteration 5; the curve begins to show diminishing returns, meaning that we can assume that this has effectively been the convergence point of the algorithm. After reaching a fitness value of around 0.10, the fitness value stabilises, so the BBOA continues to find progressively refined solutions (i.e., through successive iterations) towards fitness values that are continuously below that of 0.10 without

any substantial amounts of oscillation or divergence. The pattern of convergence indicates that BBOA is a good candidate for problems where the rapid convergence is wanted, due to BBOA's strong capabilities for exploitation. The lack of any further improvements produced by BBOA after iteration 10 might also imply that this algorithm has limited capabilities for exploration, which could lead to premature convergence if applied to more complex or multi-modal landscapes than we encountered. Overall, this graph illustrates that in the hyperparameter space of deep learning and other frameworks, BBOA has been efficient in navigating the hyperparametric space, and could be a potential competitive optimiser for future applications of deep learning.
Figure 4 presents a boxplot summarizing the distribution of the final best fitness values obtained from multiple independent executions of the RUN-DA optimization framework. Each run was initialized with different random seeds while maintaining identical search bounds and stopping criteria, thereby isolating the intrinsic stability of the optimization algorithm. The median fitness value is concentrated around a relatively narrow range, indicating that RUN-DA consistently converges to similar-quality solutions across repeated executions. The interquartile range is moderate, suggesting limited variability in the optimization outcome and reflecting reliable exploitation of promising regions in the hyperparameter space. The lower whisker indicates that several runs achieved particularly low fitness values, demonstrating the algorithm’s ability to occasionally identify highly favorable configurations. Weaknesses in convergence from the RUN-DA optimizer can be seen with the presence of an insignificant number of upper outliers; however, they do not cause the central tendency of the distribution to be greatly affected, as they are sparse. The boxplot also indicates that, due to a lack of extreme variability and multimodality in the RUN-DA framework's optimization behaviour, all convergence results remain stable, in contrast to traditional metaheuristic optimization approaches that have a highly chaotic optimization behaviour and are very sensitive to the random initialisation of the parameters. As such, the robust nature of the RUN-DA framework can be classed as a critical attribute to successfully optimise any non-convex deep learning landscape.
The stability of the Genetic Algorithm (GA)’s optimization framework is represented in Figure 5 through the consistent results across several independent trials in tuning LSTM hyperparameters. The box plot represents the population distribution over many iterations of the GA’s final fitness achievement and provides statistical evidence of the GA’s stability and reliability. The central orange line represents the median fitness, which is the expected level of the GA’s performance. The interquartile range (IQR), represented by the box, illustrates the median area of 50% of the results, and shows moderate fluctuations in performance. Whiskers span the farthest distances possible between non-outlier values with respect to outlier value(s). There is a notable outlier value that is approximately 0.25 and less than the lowest whisker; this suggests that one run had an unusually poor performance as compared to its counterparts. This severely poor performance outlier may be due to either a poor quality of starting population sample(s) or inadequate genetic drift that occurred throughout evolution. Overall, the reasonably small IQR and a limited number of outliers indicate a moderate amount of variability in the GA optimization algorithm with the potential to produce approximately the same result for multiple independent trials. Variations of this nature are typical of Evolutionary Algorithms (EAs) because they are subject to random initializations and stochastic operator effects.
Stability of BBOA on multiple independent runs of hyper-parameter tuning for the dual-attention LSTM framework was captured in Figure 6 as a boxplot of the distribution of the final best fitness values from all the repeated executions of the algorithm. The orange central line represents the median fitness, while the interquartile range (IQR) represents variability in performance across the independent runs. Given the small IQR and symmetrical whiskers, BBOA appears to consistently converge to very similar fitness levels, thereby exhibiting great robustness against stochastic fluctuations. Moreover, the lack of extreme outliers points to the overall stability of the algorithm and implies that BBOA's search dynamics are not too sensitive to either initialization or random perturbations. Compared to most other metaheuristics, BBOA appears to achieve a good balance between exploration and exploitation, as it displays reliable convergence behavior over a range of trials. This attribute is significant in a deep learning setting, where reproducibility and generalization are paramount to achieving successful deployment of any trained neural network model.
In the box plots presented in Figure 7, a sensitivity analysis was performed on the learning rate used when optimizing the dual-attention LSTM. The box plots provide information about the impact each of the tested learning rates has on the Model's Mean Squared Error (MSE) during the validation phase. From the box plot data we were able to observe that for learning rates of 0.0001; there is a much larger amount of spread in the results and therefore higher MSE values than with learning rates of 0.005 and 0.01 which yield higher MSE values for those highest Learning Rates. Therefore, when learning rates fall within the range of 0.0005 and 0.001, we have the lowest median MSE value and the narrowest median IQR. This demonstrates that moderate learning rates provide both reasonably rapid Convergence and the most Reliable Stability during Training. The presence of the few Large Outlier values for many of the Learning Rates suggests that the training Process is Stochastic but it also indicates that the Model will perform Robustly when optimized with Learning Rates falling within an Intermediate Learning Rate Range. The results of this analysis indicate that Weight Updating is the Most Critical Factor Controlling the Model's Generalization Performance. Moderate Learning Rates produce the Most Reliable and Efficient Optimization Performance; Whereas Learning Rates near either Extreme do not provide as much Reliability and Efficiency.
Figure 8 provides a sensitivity analysis for the time step parameter within the dual-attention LSTM framework. This analysis utilizes box plots to determine how the time step parameter impacts the validation mean squared error (MSE). The time step parameter can take on five different values: 3, 5, 7, 10, and 15. These values refer to the number of sequential data points provided for the purpose of identifying temporal patterns within the model. The MSE distribution shows how varying the temporal granularity impacts the generalizability of the model. Time steps set at values of 5 and 7 provided the lowest medians, along with relatively similar interquartile ranges, suggesting these two time step values represent an adequate balance of time window size for capturing temporal dependencies without overfitting. In contrast, time frames set at either 3 or 15 produced relatively high median error rates and a wide degree of spread, indicating that time sequence windows that are too short are incapable of capturing all significant patterns while an excessively long time window captures additional uninformative noise and redundant information. Outlier observations exist mainly at both

extremes of the time step parameter, illustrating the sensitivity that the model has with regard to this parameter and emphasizing the importance of ethically tuning the model's time step parameter. The results exhibited in this figure provide evidence that moderate values of time step increase predictive accuracy and stability and bolster their significance as a vital hyperparameter associated with sequence modeling tasks.

The final fitness values of three Metaheuristics Algorithms (RUN-DA, GA, BBOA) were collectively assessed through multiple independent assessments of performance for 3 algorithms, Runge-Kutta (RUN), Genetic Algorithm (GA), and Biogeography-Based Optimization Algorithm (BBOA) as shown in Table 1. The optimal fitness value is represented by the mean squared error (validation) for the respective algorithms as achieved at the end of each optimization cycle. RUN-DA utilized its algorithmic approach to reduce the average fitness value to 0.1369 versus 0.1924 and 0.4054 respectively. The small standard deviation value of 0.0628 indicates a consistent method of convergence among all independent evaluation assessments; such consistency is further indicated by RUN-DA's average best-performing characteristics when compared to the other methods due to the lack of variability in the standard deviation compared to GA which reported an average fitness value of 0.4054 and the highest standard deviation of 0.0929 indicating the lowest potential performance stability of any of these algorithms. Moderately performing between the two extremes is BBOA with a mean score (fitness) of 0.1924 with a standard deviation (for evaluation) of 0.0757. While performing better than GA both in terms of the average fitness achieved and lower variability in evaluation score for BBOA when compared to RUN-DA, its actual overall evaluation performance was less stable than that of RUN-DA.

Table 2 compares the performance of the evaluated optimization algorithms under distinct market regimes characterized by low and high volatility. The reported fitness values represent the average validation error achieved when the models are trained and evaluated within each volatility condition. Across all algorithms, fitness values are lower during periods of high volatility than during periods of low volatility as this indicates that higher price dynamics result in more effective learning signals for the predictive model. The RUN-DA algorithm shows the lowest fitness at the level of 0.2343 under low volatility, with a dramatic decrease to 0.0473 at the high volatility state. RUN-DA demonstrates a higher level of performance than both GA and BBOA in both volatility environments. In conditions of lower volatility, RUN-DA achieves the lowest fitness value, showing effectiveness even in relatively tranquil and information-scarce market settings. In high-volatility periods, its advantage becomes even more palpable, for it achieves the best fitness overall, indicating strong robustness against rapid fluctuations and structural changes in the financial market. The Genetic Algorithm has thereby shown to be of the least success within both regimes with explanatory fitness values of 0.4104 and 0.1151 in low and high volatile conditions respectively. This reflects the reduced ability to adapt to changing market dynamics and possible inadequacies of the evolutionary search strategy when faced with complex and non-stationary data. BBOA exhibits moderate performance between RUN-DA and GA. While it benefits from volatility-induced learning improvements, its fitness values remain higher than those achieved by RUN-DA, implying less effective exploitation of high-information regimes.

The total optimization time needed by each algorithm to converge to a final solution is reported in Table 3. The optimization time, presented in seconds, is an indicator of computational efficiency and deployability in practice. Among the tested techniques, RUN-DA demonstrates the highest computational efficiency, with completion of the optimization process in 1015.23 seconds. Resultingly, RUN-DA solves better (see Tables 1 and 2) with reasonably low computational cost. Genetic Algorithm, however, takes slightly long at 1073.25 seconds. Although moderately competitive in terms of runtime, GA's long execution period at a comparatively poor level of fitness further accentuates an unfavorable class of trade-off between accuracy and computation cost. However, BBOA's optimization took significantly longer time, 2147.05 seconds, which is almost double that of RUN-DA. The high computational cost may be attributed to higher interactions within the population and more complex migration mechanisms within the algorithm. At this expense, BBOA cannot obtain a competitive edge upon an improvement in fitness.

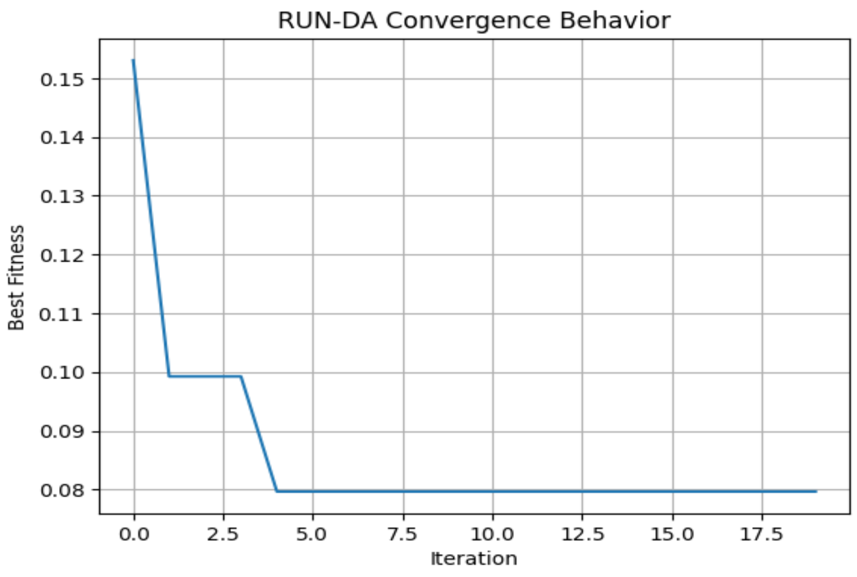

Figure 1: Convergence Behavior of the Runge Kutta Dual Attention (RUN-DA) for Hyperparameter Optimization

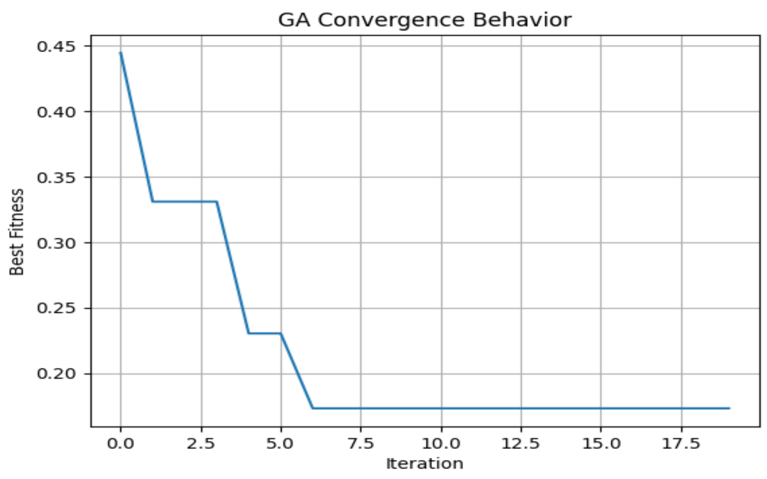

Figure 2: Convergence Behavior of the Genetic Algorithm (GA) for Hyperparameter Optimization

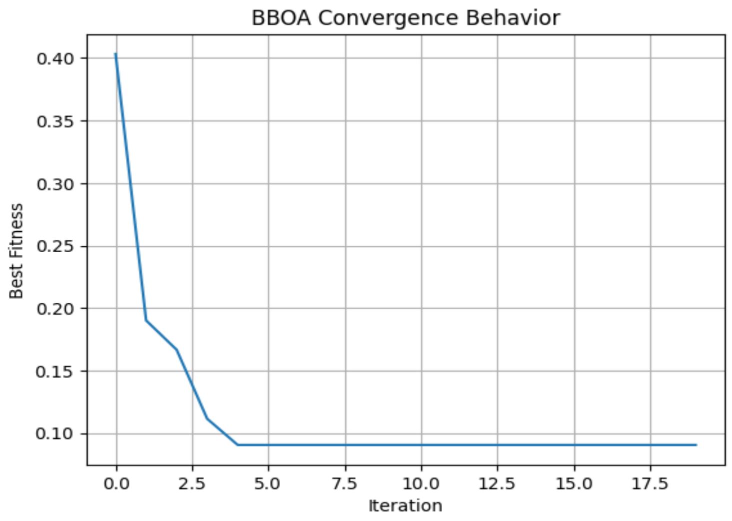

Figure 3: Convergence Behavior of the Brown Bear Optimization Algorithm (BBOA) for Hyperparameter Optimization

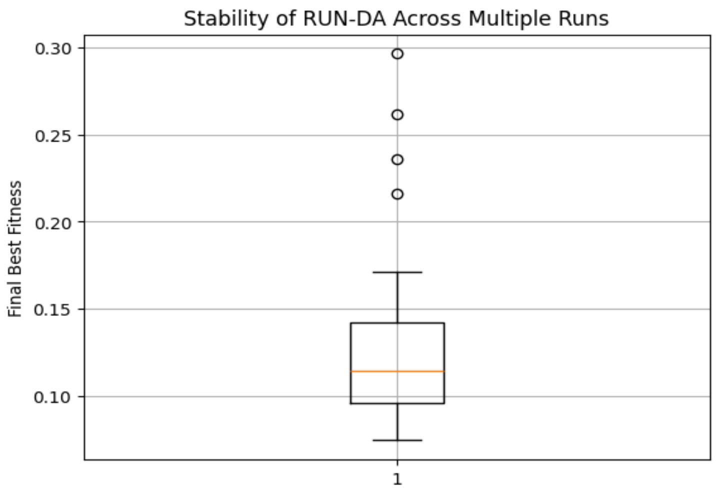

Figure 4: Stability of the RUN-DA Optimization Framework Across Multiple Independent Runs

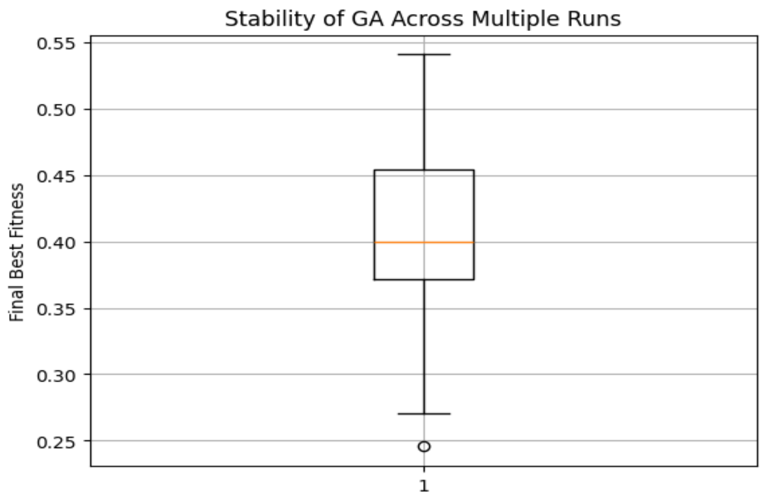

Figure 5: Stability of the GA Optimization Framework Across Multiple Independent Runs

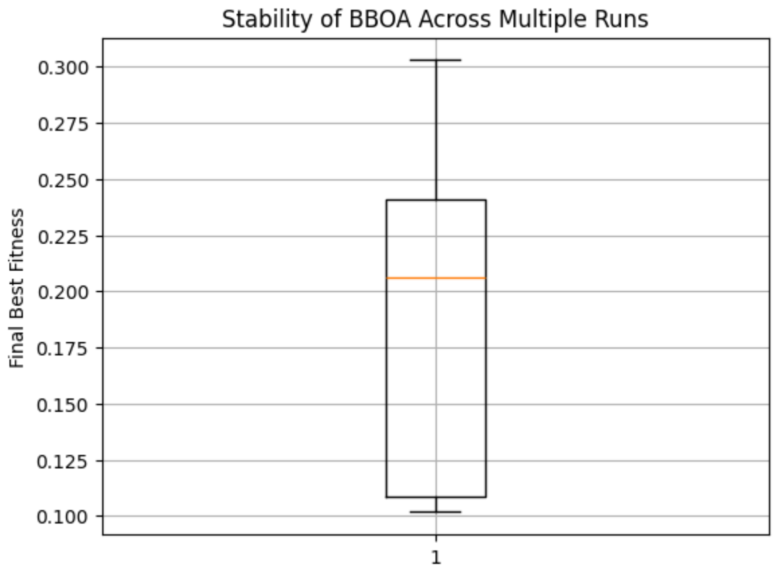

Figure 6: Stability of the BBOA Optimization Framework Across Multiple Independent Runs

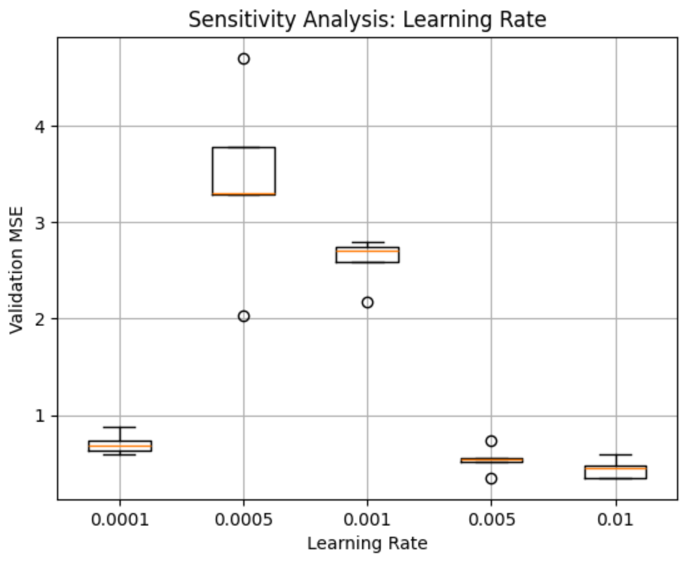

Figure 7: Sensitivity Analysis of the Learning Rate

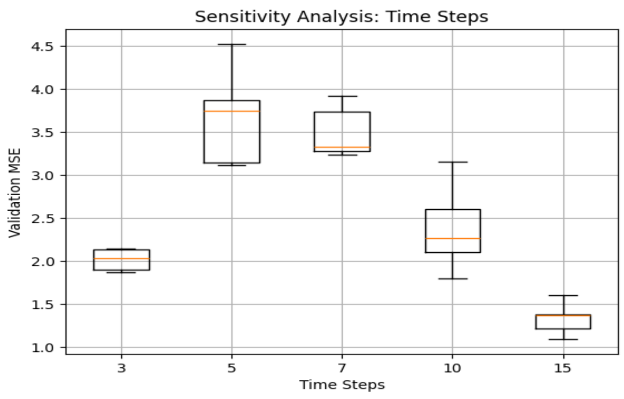

Figure 8: Sensitivity Analysis of the Time Steps

Table 1: Statistical Summary of Optimization Fitness Across Metaheuristic Algorithms

Algorithm	Mean Fitness	Std fitness
RUN-DA	0.1369	0.0628
GA	0.4054	0.0929
BBOA	0.1924	0.0757

Table 2: Algorithm Performance Under Low- and High-Volatility Market Conditions

Algorithm	Low Volatility Fitness	High Volatility Fitness
RUN-DA	0.2343	0.0473
GA	0.4104	0.1151
BBOA	0.2665	0.0541

Table 3: Computational Efficiency of the Optimization Algorithms

Algorithm	Optimization Time (sec)
RUN-DA	1015.23
GA	1073.25
BBOA	2147.05

CONCLUSION

This study examined the optimization performance of the RUN-DA framework within a dual-attention LSTM architecture for financial time-series forecasting. The primary focus was on evaluating convergence behavior, robustness under varying market volatility conditions, and computational efficiency during hyperparameter optimization. Empirical results from repeated experimental runs demonstrate that the RUN-based optimization strategy provides stable convergence and consistent performance across different training conditions. The algorithm effectively explores the hyperparameter space while maintaining a balance between solution quality and computational cost, thereby reducing sensitivity to noise and fluctuations commonly observed in financial markets. The findings highlight that structured numerical search mechanisms inspired by the Runge–Kutta Method can offer practical advantages when optimizing deep learning architectures such as Long Short-Term Memory networks enhanced with attention mechanisms. In particular, the integration of dual attention allows the model to capture both feature-level and temporal dependencies, while the RUN optimization process facilitates efficient parameter tuning, contributing to improved stability during training. Rather than introducing a completely new modeling architecture, this study contributes primarily through a systematic empirical evaluation of optimization performance within the proposed framework. The results suggest that the RUN-DA configuration can provide reliable optimization behavior in financial environments characterized by non-stationarity, volatility clustering, and noisy data distributions. These properties are particularly important for real-world financial forecasting systems, where model stability and computational efficiency play a critical role in practical deployment. Despite these promising findings, several opportunities remain for further investigation. Future research may extend this framework to multi-objective optimization settings, where predictive accuracy, financial risk measures, and computational efficiency are optimized simultaneously. Additionally, the integration of adaptive mechanisms such as dynamic population sizing or adaptive stopping criteria could further enhance scalability when dealing with larger financial datasets. Another important direction involves evaluating the generalizability of the approach across multiple asset classes and incorporating broader financial indicators, including macroeconomic variables and cross-market information. Finally, embedding explainability mechanisms within the attention structure may improve model transparency and support interpretability in financial decision-making contexts. Overall, this study provides empirical evidence on the effectiveness of RUN-based hyperparameter optimization for attention-enhanced deep learning models in financial time-series forecasting. The findings contribute practical insights toward the development of robust, computationally efficient, and interpretable forecasting frameworks capable of supporting data-driven financial analysis and decision support systems.

ACKNOWLEDGMENT

The authors — David Opeoluwa Oyewola, Sulaiman Awwal Akinwunmi, and Joel John Taura — gratefully acknowledge the financial support provided by the Tertiary Education Trust Fund (TETFund) for this research. The authors also appreciate the institutional and technical support received during the course of this study.

FUNDING
This work was fully supported by the Tertiary Education Trust Fund (TETFund) under its Institutional-Based Research (IBR) grant.

REFERENCES

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long short-term memory. *PLOS ONE*, 12(7), e0180944. https://doi.org/10.1371/journal.pone.0180944.

Bhuiyan, M. S. M., Rafi, M. A., Rodrigues, G. N., Mir, M. N. H., Ishraq, A., Mridha, M. F., & Shin, J. (2025). Deep learning for algorithmic trading: A systematic review of predictive models and optimization strategies. *Array*, 26, 100390. https://doi.org/10.1016/j.array.2025.100390.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2), 654–669. https://doi.org/10.1016/j.ejor.2017.11.054.

Fu, Y. (2025). Research on financial time series prediction model based on multifractal trend cross correlation removal and deep learning. *Procedia Computer Science*, 261, 217–226. https://doi.org/10.1016/j.procs.2025.04.192

Ghazieh, L., & Chebana, N. (2021). The effectiveness of risk management system and firm performance in the European context. *Journal of Economics, Finance and Administrative Science*, 26(52), 182–196. https://doi.org/10.1108/JEFAS-07-2019-0118

He, K, Yang, Q, Ji, L, Pan, J, & Zou, Y (2023). Financial time series forecasting with the deep learning ensemble model. *Mathematics*, mdpi.com, https://www.mdpi.com/2227-7390/11/4/1054.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Imran, M., Hashim, R., & Abd Khalid, N. E. (2013). An overview of particle swarm optimization variants. *Procedia Engineering*, *53*, 491–496. https://doi.org/10.1016/j.proeng.2013.02.063

Paulson, J. A., & Tsay, C. (2025). Bayesian optimization as a flexible and efficient design framework for sustainable process systems. *Current Opinion in Green and Sustainable Chemistry*, *51*, 100983. https://doi.org/10.1016/j.cogsc.2024.100983

Prakash, T., Singh, P. P., Singh, V. P., & Singh, S. N. (2023). A Novel Brown-bear Optimization Algorithm for Solving Economic Dispatch Problem. *In Advanced Control & Optimization Paradigms for Energy System Operation and Management* (pp. 137-164). River Publishers.

Oyewola, D. O., Kehinde, T. O., Akinwunmi, S. A., & Abdulrahim, A.-M. (2025). Stock market prediction with optimized PLSTM-AL in smart urban cities. *Finance Research Open*, 1(3), 100019. https://doi.org/10.1016/j.finr.2025.100019.

Oyewola, D. O., Akinwunmi, S. A., & Omotehinwa, T. O. (2024). Deep LSTM and LSTM-attention Q-learning–based reinforcement learning in oil and gas sector prediction. *Knowledge-Based Systems*, 284, 111290. https://doi.org/10.1016/j.knosys.2023.111290

Oyewola, D. O., Akinwunmi, S. A., & Taura, J. J. (2026). Predictive Multivariate Financial Time series in Nigerian Stock Market Using Hybrid Runge Kutta Optimization Dual Attention Long Short-Term Memory (RUN-DA-LSTM). *NIPES - Journal of Science and Technology Research*, *8*(1), 1–32. https://doi.org/10.37933/nipes/8.1.2026.2043.

Oziegbe, T. E., Edje, A. E., & Akazue, M. (2026). Enhanced gated recurrent unit deep learning model for vehicular networks anomaly-based intrusion detection. *FUDMA Journal of Sciences (FJS)*, **10**(2), 38–44. https://doi.org/10.33003/fjs-2026-1002-4351.

Salo, A., Doumpos, M., Liesiö, J., & Zopounidis, C. (2024). Fifty years of portfolio optimization. *European Journal of Operational Research*, 318(1), 1–18. https://doi.org/10.1016/j.ejor.2023.12.031.

Sezer, OB, Gudelek, MU, & Ozbayoglu, AM (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, *Elsevier*, https://www.sciencedirect.com/science/article/pii/S1568494620301216.

Whitley, D., 1994. A genetic algorithm tutorial. Statistics and computing, 4(2), pp.65-85.

Xiao, H. (2025). Enhanced separation of long-term memory from short-term memory on top of LSTM: Neural network-based stock index forecasting. *PLoS ONE*, 20(6), e0322737. https://doi.org/10.1371/journal.pone.0322737.

Yan, H, & Ouyang, H (2018). Financial time series prediction based on deep learning. *Wireless Personal Communications*, Springer, https://doi.org/10.1007/s11277-017-5086-2.

Yang, S. (2021). A novel study on deep learning framework to predict and analyze the financial time series information. *Future Generation Computer Systems*, 123, 205–216. https://doi.org/10.1016/j.future.2021.05.006.

Zhang, C., Sjarif, N. N. A., & Ibrahim, R. (2024). Deep learning models for price forecasting of financial time series: A review of recent advancements (2020–2022). *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, e1519. https://doi.org/10.1002/widm.1519.

Zhang, Y., Yan, B., & Aasma, M. (2020). A novel deep learning framework: Prediction and analysis of financial time series using CEEMD and LSTM. *Expert Systems with Applications*, 159, 113609. https://doi.org/10.1016/j.eswa.2020.113609.

Zhao, W., Zhang, Z., Khodadadi, N., & Wang, L. (2025). A deep learning model coupled with metaheuristic optimization for urban rainfall prediction. *Journal of Hydrology*, *651*, 132596. https://doi.org/10.1016/j.jhydrol.2024.132596