Deep Learning in Financial Time Series: A Comparative Analysis of RNN, GRU, LSTM, and Hybrid Models

Yasin Büyükkör
Karamanoğlu Mehmetbey University,
Department of Business Administration, Karaman, Turkey
yasinbuyukkor@kmu.edu.tr

CroEconSur
Vol. 28
No. 1
June 2026
pp. 5-38

Received: February 21, 2025
Accepted: January 16, 2026
Research Article

doi:10.15179/ces.28.1.1

Abstract

Accurate forecasting of financial time series plays a critical role in understanding market dynamics and informing investment decisions. However, the inherent volatility, nonlinearity, and noise of financial data make accurate prediction highly challenging. This study compares five deep learning architectures, including recurrent neural network (RNN), gated recurrent unit (GRU), long short-term memory (LSTM), and two hybrid models, RNN-LSTM and GRU-LSTM, to evaluate their forecasting performance on the S&P 500 Index between January 2022 and January 2025. Daily open, high, low, and volume values are used as input features, while the closing price serves as the target variable. The hyperparameters of all models are optimized using the random search procedure, and their predictive accuracy is assessed based on RMSE, MAE, and MAPE. The Diebold–Mariano test is further applied to examine whether differences

in predictive accuracy among models are statistically significant. The empirical results indicate that hybrid architectures, particularly the GRU-LSTM model, outperform the standalone models by effectively capturing both short-term fluctuations and long-term dependencies in financial time series. The GRU-LSTM model achieves an approximately ten percent lower prediction error compared to the LSTM model, demonstrating the effectiveness of hybridization in improving deep learning forecasting performance. These findings confirm the robustness of hybrid deep learning architectures for financial time series forecasting and provide valuable insights for researchers and practitioners in quantitative finance.

Keywords: financial time series, recurrent neural network, gated recurrent unit, long short-term memory, hybrid deep learning models

JEL classification: C22, C45, C53, G17

1 Introduction

Financial time series forecasting is a multidisciplinary field situated at the intersection of economics, finance, statistics, and econometrics (Santos Júnior et al., 2019). Stock market data often contain substantial noise and reflect complex patterns influenced by political events, investor sentiment, and market instability (Zhong & Enke, 2017). Because these series include both linear and nonlinear components, they exhibit strong volatility and complicated temporal behavior. Therefore, producing accurate forecasts is essential for investors and financial analysts, as it supports informed decision-making and helps reduce financial risk (Ren et al., 2023).

In the 1970s, statistical methods such as autoregressive integrated moving average (ARIMA) were widely employed to analyze time series data (Box et al., 2015). Although these models are generally simple and offer flexibility for most data, their linear structure makes them insufficient for forecasting problems (Firmino et al., 2015). Studies conducted over the last few decades have shown that machine

learning (ML) methods outperform traditional statistical methods. In particular, the emergence of deep learning (DL), a subfield of ML that utilizes multi-layered neural architectures such as artificial neural networks (ANNs), has been a turning point for the analysis of time series. ANN-driven models, which do not require the assumption of independence and stationarity, have been shown to be more advantageous than linear models (Foster et al., 1992; Khashei & Bijari, 2011; Xu et al., 2019).

Selecting appropriate input variables and optimizing network parameters are essential for the effective deployment of ANNs. However, this selection process is often a complex and challenging task (Hussain et al., 2008). This challenge is partially addressed by recurrent neural networks (RNNs). In an RNN, the outputs are related to the previous inputs; thus, it can handle past dependencies in time series analysis. However, an RNN is insufficient to capture the long-term past dependencies (Hochreiter, 1998). GRU and LSTM, as advanced forms of RNN, effectively address this challenge. These architectures utilize gating mechanisms that enable the models to capture long-term dependencies, a crucial capability for accurate financial time series forecasting (Gao et al., 2021).

This paper investigates the effectiveness of RNN, GRU, LSTM, and hybrid architectures in forecasting financial time series and compares their predictive capabilities. The proposed hybrid models, RNN-LSTM and GRU-LSTM, are designed to integrate the strengths of traditional recurrent structures and advanced gated mechanisms to capture both short-term and long-term dependencies in financial data. The empirical analysis uses the daily closing prices of the S&P 500 Index, a key benchmark representing the overall performance of the U.S. stock market, covering the period from January 1, 2022, to January 27, 2025. To ensure consistency and comparability, all models are trained using a 30-day window, where each model uses the previous 30 observations to forecast the next day's closing price. Model performance is evaluated using three standard accuracy measures (RMSE, MAE, and MAPE), while the Diebold–Mariano test is employed to statistically assess whether the differences in predictive accuracy

among the models are significant. The findings reveal that hybridization based on deep learning methods provides a powerful alternative for financial forecasting, offering improved accuracy, statistical robustness, and reliability. These results demonstrate that the proposed hybrid frameworks can serve as valuable decision-support tools for investors and researchers in financial markets.

The subsequent sections of this paper are organized as follows. Section 2 reviews the existing literature and highlights key developments in the application of deep learning methods to financial forecasting. Section 3 presents the theoretical background and architectural structure of the RNN, GRU, and LSTM models, as well as the proposed hybrid configurations. Section 4 describes the empirical methodology, introduces the dataset, explains the data preprocessing and hyperparameter optimization procedures, and presents the empirical results, including the statistical evaluation based on the Diebold–Mariano test. Finally, Section 5 discusses the main findings, interprets their broader implications for financial modeling and forecasting, and outlines potential directions for future research.

2 Literature Review

Over time, the analysis of financial time series has transformed from classical statistical methods to more advanced DL-based approaches capable of analyzing complex structures without relying on strong assumptions. The development of DL models for forecasting financial time series can be traced back to early hybrid approaches that combined statistical and ML models. Zhang (2003) introduced a hybrid model that combines ANN and ARIMA to capture both linear and nonlinear dynamics in time series forecasting. In the study, ARIMA is utilized to forecast the linear element of the data, whereas ANN handles the prediction of the nonlinear elements. The study demonstrated that the proposed model provided higher accuracy in forecasting financial time series compared to standalone models. Ince and Trafalis (2006) used ARIMA and vector autoregressive (VAR)

models to extract the linear components, while support vector regression (SVR) and ANN models were used to extract the nonlinear components. The findings indicated that SVR achieved better predictive effectiveness compared to ANN, and the use of inputs derived from parametric methods enhanced SVR's forecasting performance. Kim and Shin (2007) developed a hybrid model combining adaptive time delay neural networks (ATDNN) and genetic algorithms (GA). The hybrid model outperformed the traditional ATDNN, TDNN, and RNN models. Khashei and Bijari (2012) proposed a hybrid model that included a base model and a classifier. The model analyzed the errors of traditional methods and optimized the forecasts using a probabilistic neural network (PNN). The findings revealed that the proposed model delivered better performance than conventional approaches. Collectively, these early studies established the foundation for hybridization between statistical and neural models, highlighting that combining linear and nonlinear components improves predictive power in financial time series forecasting.

Following these early developments, research attention shifted toward incorporating optimization algorithms and deeper architectures to enhance learning capacity. Rather et al. (2015) used ARIMA and exponential smoothing methods to forecast linear components and RNN for nonlinear components in forecasting stock returns. The proposed model improved accuracy by using weights optimized by GA, which outperformed conventional RNN models. Wei (2016) developed an innovative model integrating an adaptive network-based fuzzy inference system (ANFIS) and empirical mode decomposition (EMD). By examining datasets from the Taiwan (TAIEX) and Hang Seng stock indices, the study concluded that the proposed model surpassed alternative methods, demonstrating lower RMSE values. The hybrid model improved forecasting accuracy by reducing noise and effectively modeling nonlinear structures in financial data. Bao et al. (2017) proposed a DL system, named WSAEs-LSTM, which combines wavelet transforms (WT) and stacked autoencoders (SAE) to enhance feature extraction and forecasting performance. The study demonstrated

that the WSAEs-LSTM improved forecasting accuracy across six different stock indices and provided higher profitability compared to traditional methods. The model's operational framework involves a series of steps: denoising the data using WT, feature extraction by SAE, and finally feeding these features into the LSTM layer. These advances signified a turning point, showing that hybrid deep learning models could simultaneously manage noise reduction, feature extraction, and nonlinear dependency modeling, which traditional approaches struggled to address.

Kim and Won (2018) further extended this stream by combining LSTM and GARCH-derived models for index volatility. Their proposed GEW-LSTM model achieved lower error rates in terms of MSE, heteroscedasticity adjusted MAE, and heteroscedasticity adjusted MSE, as empirically evidenced by tests on the KOSPI 200 Index. Santos Júnior et al. (2019) developed an innovative hybrid model by combining multilayer perceptron (MLP) and SVR. The proposed model was evaluated using six different time series (e.g., Canadian lynx population, sunspot data), and it outperformed existing hybrid methods. Jianwei et al. (2019) integrated independent component analysis (ICA) and GRU for forecasting gold prices. Their findings showed that the proposed model effectively captures long-term trends and cyclical structures in gold prices, offering faster computation and higher forecasting accuracy compared to conventional approaches. By the end of this period, hybrid neural structures had matured beyond statistical integration, focusing instead on advanced optimization and component decomposition to capture both temporal and structural dependencies in financial markets.

Since 2020, hybrid architectures have been increasingly explored to further enhance the forecasting capability of deep learning models. Barra et al. (2020) utilized a series of convolutional neural network (CNN) based models, proposing an innovative method that transforms time series into Gramian angular field (GAF) images for stock market forecasting. The proposed system achieved higher profits and increased market forecast accuracy in the analysis of the S&P 500 Index. Nabipour et al. (2020) examined various ML and DL methods to forecast

the historical future values of the Iranian Stock Market and found that the LSTM model achieved the highest level of accuracy and best fit among competing methods. Liu and Long (2020) developed a hybrid model (EWT-dpLSTM-PSO-ORELM) for financial time series forecasting, which decomposed data into sublayers using empirical wavelet transform (EWT), while the LSTM network was optimized using particle swarm optimization (PSO). Their model consistently outperformed conventional forecasting techniques across major U.S. and Chinese stock indices. Similarly, Hu et al. (2020) proposed combining GARCH and DL methods to predict copper price volatility. Hybrid models created by integrating volatility estimates derived from GARCH with memory networks such as LSTM and BLSTM provided the highest prediction performance.

Qiu et al. (2020) introduced an RNN-based hybrid model that combines LSTM, GRU, and rectified linear unit (ReLU) layers to enhance stock market prediction performance. Sezer et al. (2020) carried out an extensive literature review emphasizing the superior performance of DL models over traditional ML techniques. Islam and Hossain (2021) introduced a hybrid GRU-LSTM approach for FOREX forecasting, demonstrating complementary capabilities of both models in reducing error rates. Their study showed and analyzed the hybrid model's performance over 10-minute and 30-minute intervals using four key currency pairs, including combinations of the euro, British pound, U.S. dollar, and Canadian dollar. Luo et al. (2021) introduced a hybrid approach based on ensemble empirical mode decomposition (EEMD) for forecasting financial data, which improved both precision and directional accuracy.

Between 2021 and 2025, a growing number of studies have focused on developing and comparing hybrid DL architectures across various financial domains. As summarized in Table 1, this period marks a methodological shift from single-model frameworks toward multi-layer and cross-domain hybrids integrating CNN, GRU, and LSTM components. The listed studies collectively demonstrate how hybrid deep learning approaches have become the dominant paradigm in recent financial forecasting research, particularly due to their superior ability to model complex temporal-spatial dependencies.

Recent studies have also expanded toward multi-domain applications. Zhao and Chen (2022) proposed a hybrid CNN-LSTM model capturing spatial and temporal dependencies in stock prices. Song and Choi (2023) combined CNN and GRU layers within an ensemble framework, improving directional prediction accuracy. Das et al. (2024) applied encoder–decoder hybrid architectures (AE-LSTM and AE-GRU) for stock and cryptocurrency forecasting, while Ge (2025) introduced a wavelet-based hybrid MEMD-AO-LSTM model for the S&P 500 and CSI 300 indices, achieving superior predictive performance. These recent contributions underline the continuing evolution and relevance of hybrid DL frameworks, particularly those integrating memory-based and convolutional components, in capturing the complex dynamics of financial markets.

Table 1: *Previous Works*

Reference	Dataset	Method	Best model
Yang (2021)	MDLZ, HRL	LSTM, GRU	LSTM and GRU
Zhao and Chen (2022)	S&P 500 Index	RCSNet, CNN, LSTM, ARIMA	RCSNet
Freeborough and van Zyl (2022)	S&P 500 Index	RNN, LSTM, GRU	GRU
Pirani et al. (2022)	IBM, HDFC, SNOWMAN, BHEL	ARIMA, GRU, LSTM, BiLSTM	GRU
Sako et al. (2022)	8 stock indices and 6 foreign exchange rates	RNN, LSTM, GRU	GRU
Song and Choi (2023)	DAX, DOW, and S&P 500 Index	RNN, CNN-LSTM, GRU-CNN, ensemble models and ML methods	DL methods
Ampountolas (2023)	European financial markets and BTC	ARIMA, ETS-ANN, kNN	ETS-ANN and ARIMA
Al-hnaity and Abbod (2016)	FTSE 100, S&P 500 Index, and Nikkei 225	EMD-SA, EEMD-SVR, EEMD-RNN, EEMD-BPNN, EEMD-GA-WA, AR	EEMD-GA-WA
Kumar et al. (2023)	BTC, Sunspot data, and experimental data	ARIMA, MLP, βSARMA, LSTM, ARIMA-ANN, βSARMA-LSTM	βSARMA-LSTM
Seabe et al. (2023)	BTC, ETC, and LTC	LSTM, GRU, BiLSTM	BiLSTM
Gür (2024)	Silver price	CNN, LSTM, GRU, CNN-LSTM-GRU	CNN-LSTM-GRU

Das et al. (2024)	Apple stock price, JP Morgan Chase, BTC, S&P 500 Index	LSTM, GRU, AE-LSTM, AE-GRU	AE-GRU
Pokou et al. (2024)	S&P 500 Index, TOTALENERGIES, EURO/USD	ARIMA, LSTM, GRU, TCN, MLP, GPR	TCN
Dao (2024)	Apple stock price	LSTM-GRU, transformer-GRU	LSTM-GRU
Raman et al. (2024)	BTC	ARIMA, LSTM, GRU, LSTM-GRU	LSTM-GRU
Zhu et al. (2024)	CSI 300 Index	LSTM, EMD-LSTM, CEEMDAN-LSTM, C-LSTM, EMD-C-LSTM, CEEMDAN-C-LSTM	CEEMDAN-C-LSTM
Kabir et al. (2025)	BTC, Amazon stock price, Google stock price, CSI 300 Index	RW, ARIMA, MLP, LSTM, CNN, ARMA-CNN-LSTM, LSTM-mtrans-MLP	LSTM-mtrans-MLP
Ge (2025)	S&P 500 Index, CSI 300 Index	MLP, ELM, BPNN, RBF, transformer, GRU, BiLSTM, LSTM, WT-LSTM, VMD-LSTM, EMD-LSTM, EEMD-LSTM, AO-LSTM, MEMD-AO-LSTM	MEMD-AO-LSTM

Notes: MDLZ: Mondelez International, HRL: Hormel Food Corp, RCSNet: Residual-CNN-Seq2Seq, IBM: International Business Machines, HDFC: The Housing Development Finance Corporation, SNOWMAN: Snowman Logistics, BHEL: Bharat Heavy Electricals Limited, DAX: Deutscher Aktienindex, ETS-ANN: exponential smoothing-ANN, DOW: Dow Jones Industrial Average, SA: simple average, BPNN: back propagation NN, GA: genetic algorithm, AM: attention mechanism, BTC: Bitcoin, ETH: Ethereum, LTC: Litecoin, MLP: multilayer perception, TCN: temporal convolutional network, GPR: Gaussian process regression, RDG: reduced density gradient, LAS: listen, attend, and spell, CEEMDAN: complete ensemble EEMD with adaptive noise, RW: random walk, MEMD: multivariate empirical mode decomposition, ELM: elaboration likelihood model, RBF: radial basis function, VMD: variational mode decomposition, AO: Aquila optimizer, EEMD: ensemble empirical mode decomposition.

Source: Author's own compilation.

Table 1 provides an overview of previous research utilizing combined methods in empirical analyses. As can be seen from Table 1, analyzing financial time series with hybrid methods increases the forecasting performance of DL models.

Taken together, the reviewed literature reveals that while significant progress has been made in hybridizing neural architectures, most studies emphasize LSTM/GRU or CNN-based models. However, there remains limited comparative analysis of RNN-LSTM and GRU-LSTM structures under identical hyperparameter optimization and evaluation conditions, a gap that this study aims to address.

3 Methodology

The methodological framework of this study follows a structured DL workflow consisting of four main stages: data collection, data preprocessing, model development, and performance evaluation. The financial data were first collected and organized, followed by preprocessing procedures to ensure consistency and stability. Next, the recurrent and hybrid deep learning models were developed and optimized using the random search method for hyperparameter tuning. Finally, model performance was assessed using multiple error metrics to evaluate forecasting accuracy and robustness. The theoretical background and architecture of each model employed in this study are summarized in the following subsections.

3.1 Recurrent Neural Network (RNN)

RNN, specifically the Elman-type recurrent neural network (simple RNN), is used to process sequential information through recurrent connections between hidden layers at different time intervals (Rumelhart et al., 1986). In an RNN, each recurrent module, also referred to as a recurrent unit, is fed by the previous hidden state (h_{t-1}), allowing information from prior time steps to influence the current output. Thus, long-term dependencies can be stored in memory through the network's recurrent connections. In practical applications, RNNs can struggle to capture long-term dependencies, especially when dealing with large datasets. In the empirical analysis, the Elman-type RNN architecture was employed to model sequential dependencies and temporal dynamics within the financial time series data. The formulation of a simple RNN is given by Equation (1).

$$h_t = tanh(W[h_{t-1}, x_t] + b) \tag{1}$$

In Equation (1), b is the bias vector, x_t is the input at time t, h_t is the hidden state, W is the weight matrix, and *tanh* (the hyperbolic tangent activation function) is used to ensure that the input values remain within the range of –1 and +1. Figure 1 illustrates the overall architecture of the RNN model.

3.2 Gated Recurrent Unit (GRU)

To address the vanishing gradient problem observed in recurrent neural networks, the GRU was first developed by Cho et al. (2014). In an RNN, information is stored in memory through a specialized network that incorporates recurrent processing. The basic recurrent process in RNN is carried out by the recurrent unit. To overcome the limitations of RNN in capturing long-term dependencies, GRU uses gating structures that integrate mechanisms to capture past dependencies within the data, allowing the model to retain and update relevant information over time.

A GRU consists of two gates: update and reset gates. The update gate facilitates the transfer of the current input (x_t) and the output from the previous cell (h_{t-1}) to the next cell, ensuring the concurrent processing of current and past data. The reset gate is responsible for weighting (W) the information that was previously learned and ensuring that the required information is transferred to the following iteration.

The formulations of the update gate, reset gate, and hidden state (h_t) are provided in Equation (2).

$$z_t = \sigma(W_z[h_{t-1}, x_t])$$
$$r_t = \sigma(W_r[h_{t-1}, x_t]) \quad (2)$$
$$h_t = (1 - z_t)h_{t-1} + z_t h_t$$

In Equation (2), z_t is the update gate, r_t is the forget gate, and σ is the sigmoid activation function. The general structure of the GRU model is given in Figure 1.

3.3 Long Short-Term Memory (LSTM)

Hochreiter and Schmidhuber (1997) initially introduced LSTM to overcome the vanishing gradient issue in RNNs and to efficiently model long-term dependencies within datasets. LSTM is capable of capturing long-term dependencies using memory cells. A simple RNN contains one layer; however, the LSTM model consists of four layers, including a memory unit along with an input gate, forget gate, and output gate.

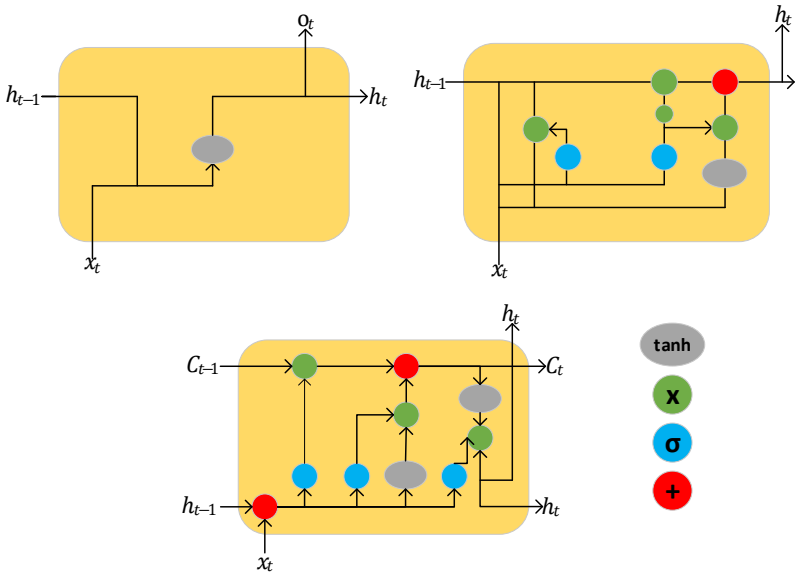
In LSTM, updated information is continuously transferred between the memory cell and the gate structures. Thus, information obtained from past data can easily represent future information, and long-term dependencies can be captured more effectively than in RNNs and GRUs. Equation (3) presents the mathematical formulation of the LSTM model.

$$
\begin{aligned}
X &= \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix} \\
f_t &= \delta\left(W_f \cdot X + b_f\right) \\
i_t &= \delta(W_i \cdot X + b_i) \\
o_t &= \delta(W_o \cdot X + b_o) \\
\widetilde{C_t} &= tanh(W_c \cdot [h_t, x_t] + b_c) \\
C_t &= f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \\
h_t &= o_t \odot tanh(C_t)
\end{aligned}
\tag{3}
$$

In Equation (3), W_f, W_i, W_o, and W_c represent weight matrices of the LSTM; b_f, b_i, b_o, and b_c are the bias vectors; δ is the activation function, and $\odot$ represents the Hadamard (element-wise) multiplication operator.

Figure 1 illustrates the core structural differences between RNN, GRU, and LSTM architectures in a compact form, focusing on information flow and gating mechanisms rather than full computational details.

Figure 1: *RNN Structure (top-left), GRU Structure (top-right), and LSTM Structure (bottom)*



Source: Author's own illustration.

3.4 RNN-LSTM Hybrid Model

Based on the basic RNN and LSTM structures introduced in Sections 3.1–3.3, the first hybrid architecture integrates a simple RNN layer and an LSTM layer within a single end-to-end network. The main idea is to combine the ability of the RNN to capture short-term temporal dependencies with the ability of the LSTM to preserve long-term information through its memory cells. Similar hybrid recurrent designs have been shown to improve forecasting performance in stock index and cryptocurrency markets (Qiu et al., 2020; Koduru et al., 2025; Xiao, 2025; Yunita et al., 2025).

In the first stage, the simple RNN layer computes a sequence of hidden states $h_t^{(RNN)}$ using the recurrence in Equation (1). This sequence summarizes the recent dynamics of the input series and acts as an intermediate representation.

In the second stage, these RNN hidden representations are passed to the LSTM layer. The LSTM cell receives $h_t^{(RNN)}$ as its input, and the gates and cell state are updated according to Equation (3), with x_t replaced by $h_t^{(RNN)}$. Denoting the LSTM hidden state by $h_t^{(LSTM)}$, the final prediction of the hybrid model is obtained through a dense output layer:

$$\hat{y}_t = W_0 h_t^{LSTM(RNN)} + b_0 \tag{4}$$

with notation consistent with Sections 3.1–3.3. Here, Equation (4) is specific to the RNN-LSTM hybrid architecture and explicitly reflects that the LSTM hidden state is obtained from representations generated by the preceding simple RNN layer. In this way, the hybrid RNN-LSTM model jointly exploits short- and long-term temporal information within a unified architecture, and recent evidence indicates that such hybrids can outperform single-architecture RNN or LSTM models in financial time series applications (Koduru et al., 2025; Qiu et al., 2020; Xiao, 2025; Yunita et al., 2025).

3.5 GRU-LSTM Hybrid Model

The second hybrid architecture combines a GRU layer with an LSTM layer in a sequential configuration. The GRU is used to model recent dynamics through its update and reset gates, while the subsequent LSTM layer focuses on capturing long-horizon and higher-order temporal dependencies. This type of GRU-LSTM hybrid has been reported to yield competitive or superior forecasting accuracy in foreign exchange and other financial time series settings (Islam & Hossain, 2021; Qiu et al., 2020; Yunita et al., 2025).

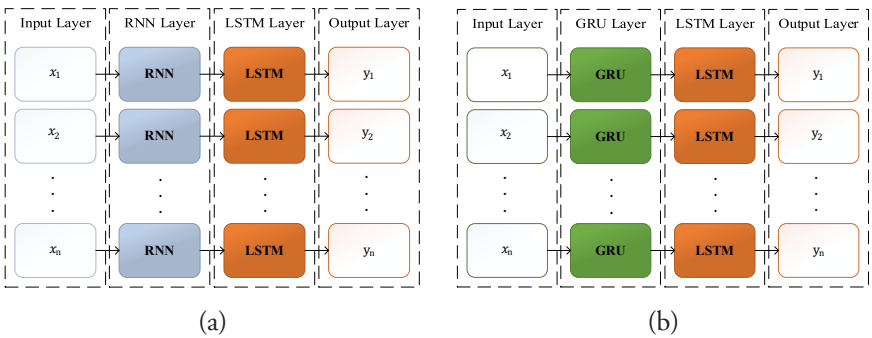
In the first stage, the GRU layer produces a sequence of hidden states $h_t^{(GRU)}$ according to Equation (2). These GRU hidden representations, which already embed information about recent observations, are then used as inputs to the LSTM layer.

In the second stage, the LSTM cell updates follow the same formulation as in Equation (3), with x_t replaced by $h_t^{(GRU)}$. Denoting the LSTM hidden state again by $h_t^{(LSTM)}$, the final output of the hybrid model is obtained through a dense layer:

$$\hat{y}_t = W_0 h_t^{LSTM(GRU)} + b_0 \tag{5}$$

Using the same notation as in the previous subsections, Equation (5) corresponds specifically to the GRU-LSTM hybrid architecture, in which the LSTM hidden state is derived from GRU-based representations in the first stage. GRU-LSTM hybrid architecture leverages the complementary memory mechanisms of the two recurrent units within a single model: the GRU efficiently filters and transforms recent information, while the LSTM provides a richer long-term memory. This design is in line with previous work documenting the benefits of GRU-LSTM hybrids in currency and general time series forecasting (Islam & Hossain, 2021; Qiu et al., 2020; Yunita et al., 2025).

Figure 2: *Architecture of the Hybrid RNN-LSTM and GRU-LSTM Models*



Note: (a) RNN-LSTM hybrid architecture; (b) GRU-LSTM hybrid architecture.

Source: Author's own illustration.

In this study, RNN-LSTM and GRU-LSTM hybrid models were adopted instead of the commonly used LSTM-GRU architecture. The main motivation was to provide a broader comparative perspective by combining recurrent structures with

different memory behaviors. RNN components capture short-term sequential dependencies, whereas LSTM and GRU units effectively learn long-term nonlinear relationships. This complementary design enhances interpretability and contributes to the literature by offering new empirical insights into the interaction of distinct recurrent architectures in financial forecasting. The overall workflow and layer-by-layer structure of both hybrid models are illustrated in Figure 2 for clarity.

4 Data and Data Preprocessing

4.1 Data

For the purpose of forecasting the closing prices of the S&P 500 Index, historical data between January 1, 2022, and January 27, 2025, were obtained from Yahoo Finance. The dataset includes daily open, close, high, and low prices as well as the volumes. The closing prices and the train–test split are illustrated in Figure 3, and the descriptive statistics are summarized in Table 2. After excluding weekends and public holidays during which the market was closed, a total of 768 observations were included in the analysis. In this study, the variables of opening, high, low, and volume (OHLV) are utilized as feature inputs. This approach enables the model to capture dependencies embedded in the daily dataset (Zhang et al., 2024).

Table 2: *Summary Statistics of S&P 500 Index*

Price	Close	High	Low	Open	Volume
Count	768	768	768	768	768
Mean	4631.42	4657.80	4602.33	4631.06	4,187,477,382
Min.	3577.03	3608.34	3491.58	3520.37	1,639,500,000
25%	4091.07	4124.53	4069.79	4087.28	3,691,362,500
50%	4441.60	4462.68	4408.41	4443.58	4,019,965,000
75%	5187.67	5209.19	5153.21	5173.90	4,494,580,000
Max.	6118.70	6128.185	6088.74	6121.43	9,354,280,000
Std. dev.	681.75	678.88	684.53	682.14	858,720,873

Source: Author's own calculation.

4.2 Data Preprocessing

In this study, the optimal hyperparameters of the DL methods were determined using the random search technique to improve the forecasting results. When splitting the data into training and test sets, the temporal order of observations was preserved so that past information was used to forecast future values, consistent with the sequential nature of time series data. To effectively capture these dependencies in the forecasting process, the sliding window method is employed. In this approach, the $(n + 1)^{th}$ observation is forecast based on the previous n^{th} observations. This process is repeated until the last observation is predicted.

To identify the optimal hyperparameters for each deep learning model, the random search technique was employed. Unlike grid search, which exhaustively evaluates every possible combination of parameters, random search samples random combinations within a defined search space (e.g., number of neurons, dropout rate, learning rate, optimizer). This approach substantially reduces computational time while maintaining a high likelihood of finding near-optimal solutions (Bergstra & Bengio, 2012).

4.2.1 Data Scaling

Due to the volatile and non-stationary structure of financial time series, forecasting with DL methods requires substantial computational time and resources. In order to both reduce the computational process and to prevent the data sizes from affecting each other, all data are scaled to the range of 0–1 by applying the min-max method to the data. The min-max method used in the scaling is given in Equation (6).

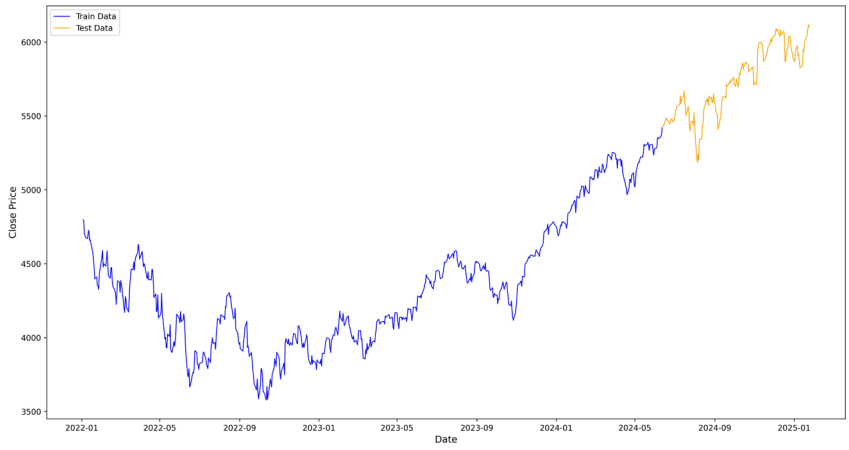
$$x_s = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{6}$$

In Equation (6), x_s is the scaled data, x_i is the raw data, x_{max} and x_{min} are maximum and minimum data, respectively. Min-max scaling maintains the proportional relationships among variables and prevents gradient saturation, providing more stable convergence in neural network training, particularly when using sigmoid or tanh activation functions.

4.2.2 Data Partitioning

The S&P 500 closing price used in this study is split into training and test sets, taking into account the characteristics of the time series data. The first 80 percent of the data, from March 1, 2022, to June 12, 2024, is the training set, while the remaining 20 percent, from June 13, 2024, to January 24, 2025, is the test set. The dataset includes a total of 614 observations for training and 154 observations for testing. Table 3 outlines the distribution of observations along with the respective date ranges for both the training and test sets.

Figure 3: *Train–Test Split*



Source: Author's own illustration.

Table 3: *Train–Test Split*

Dataset	Train data	Test data
S&P 500	2022.03.01 – 2024.06.12	2024.06.13 – 2025.01.24
	614 observations	154 observations

Source: Author's own compilation.

4.2.3 RNN, GRU, LSTM, and Hybrid Models Data Preprocessing

The hyperparameters of the RNN, GRU, LSTM, and hybrid methods, and their respective ranges, are shown in Table 4. The optimal parameters are identified using the random search technique. To capture the nonlinear relationships and volatility in the financial time series, all models are constructed with a minimum of two layers. In addition, the window size for all models is fixed at 30 days in order to account for past dependencies in the time series during the analyses. The window size refers to the number of past observations included in the model to predict future values. The window size was set to 30 days, corresponding to approximately one calendar month of trading activity (about 21–22 trading days). This choice is consistent with prior empirical studies (Das et al., 2018; Gao et al., 2021; Li et al., 2024), which demonstrated that a 30-day horizon effectively captures monthly cycles, momentum persistence, and volatility clustering while preserving model stability despite weekend market closures.

The number of hidden neurons in each model is a critical hyperparameter, representing the core processing units that perform nonlinear transformations and extract complex features from the input data. In this study, this number was systematically determined by searching the discrete range {32, 64, 96, 128, 256, 512} using the random search optimization approach. This method was specifically chosen over grid search for its proven efficiency in finding near-optimal solutions across high-dimensional hyperparameter spaces (Bergstra & Bengio, 2012).

To enhance the generalization ability of neural networks and prevent overfitting, a common issue in deep learning models, the dropout regularization technique

was applied. Dropout temporarily removes each neuron (or hidden unit) from the network with a defined probability (dropout rate) during training. This process prevents neurons from becoming overly dependent on one another and enables the model to learn more robust and independent representations. In this study, dropout rates within the range of 0.0–0.5 were tested, as this interval is widely accepted in the literature and suitable for financial time series tasks. The optimal dropout rate was determined together with other hyperparameters using the random search optimization method. Employing dropout within this range and as a regularization mechanism has been proven to be highly effective in controlling overfitting (Srivastava et al., 2014; Hinton et al., 2012).

The learning rate is one of the most critical hyperparameters that controls the magnitude of weight updates during training (Bengio, 2012). A smaller learning rate ensures stable convergence but increases training time, while a larger value accelerates learning but may cause divergence or oscillation. In this study, learning rates within the range of 0.0001 to 0.1 were tested using the random search optimization method to identify the most suitable configuration for each model.

Table 4: *Model's Parameters*

		Selected best parameters				
	Parameter range	**RNN**	**GRU**	**LSTM**	**RNN-LSTM**	**GRU-LSTM**
Layer 1 neuron	32, 64, 96, 128, 256, 512	96	256	512	256	256
Dropout rate	0, 0.1, 0.2, 0.3, 0.4, 0.5	0.1	0.1	0.2	0.2	0.3
Layer 2 neuron	32, 64, 96, 128, 256, 512	64	512	96	64	128
Learning rate	0.0001, 0.001, 0.01, 0.1	0.001	0.001	0.001	0.001	0.001
Optimizer	Adam, SGD, Adamax, Nadam, RMSprop	RMSprop	Adam	Adam	Adam	Adam

Source: Author's own compilation.

Regarding optimization algorithms, adaptive gradient-based optimizers such as Adam, RMSprop, and Nadam were evaluated, as they dynamically adjust the learning rate based on first- and second-order moments of the gradients. These optimizers have shown superior performance in handling non-stationary and noisy time series data compared to traditional methods like SGD (Kingma & Ba, 2015; Tieleman & Hinton, 2012). Based on empirical results, Adam and RMSprop yielded the best convergence stability and lowest error across models.

4.3 Evaluation Metrics

To measure and compare the predictive accuracy of DL models, basic statistical error metrics are used in this study. The mathematical representations of these error metrics are provided in Equation (7).

$$Root\ mean\ squared\ error = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

$$Mean\ absolute\ error = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{7}$$

$$Mean\ absolute\ percentage\ error = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i}$$

In Equation (7), y_i and $\hat{y}_i$ are the actual and predicted values, respectively, and n refers to the number of data points collected. All model evaluation metrics, including RMSE, MAE, and MAPE, were computed based on the test dataset to ensure unbiased assessment of predictive accuracy and to avoid any potential data leakage from the training process. The evaluation metrics presented in Equation (7) were employed as they are widely used in financial time series forecasting. RMSE and MAE quantify prediction errors, whereas MAPE expresses them in percentage terms, allowing straightforward interpretation and comparison across models.

4.4 Analysis of the Results

The results of the empirical analysis indicate that the model with the highest error rates, considering all performance criteria, is the RNN, as shown in Table 5. The RMSE for the RNN is calculated as 126.63, the MAE as 115.21, and the MAPE as 1.99 percent. This indicates that the older RNN architecture faces challenges in capturing long-term dependencies and handling the complex patterns present in the dataset. Among the individual models, both GRU and LSTM show more effective control over the historical information because of their gate structures. However, among the standalone models, the LSTM outperforms the GRU and shows a greater ability to capture long-term dependencies.

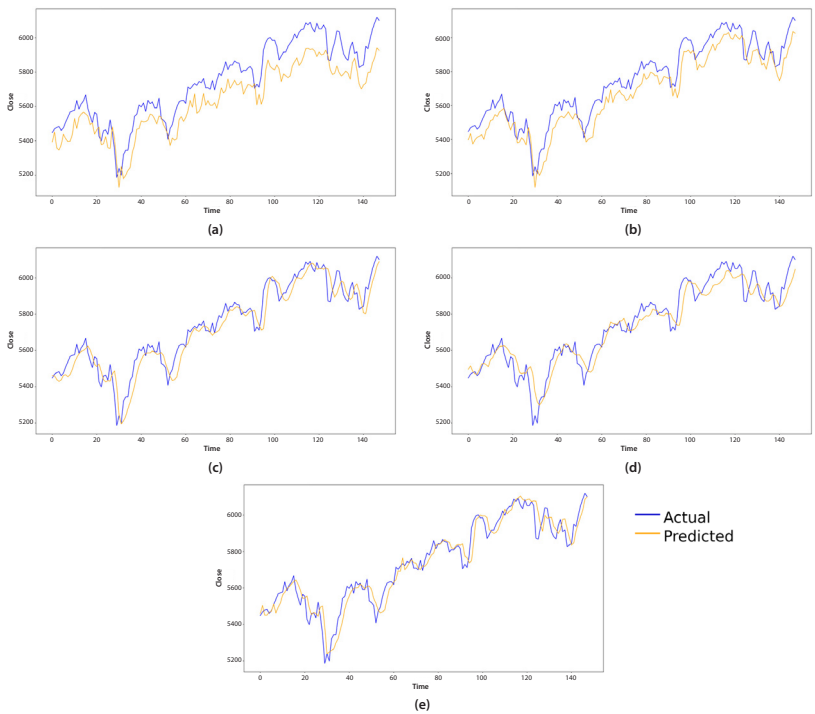
Table 5: *Forecasting Performance Results of Deep Learning Models*

	RNN	GRU	LSTM	RNN-LSTM	GRU-LSTM
RMSE	126.63	87.14	66.39	69.07	**59.82**
MAE	115.21	77.6	51.23	53.27	**44.53**
MAPE	1.99	1.35	0.90	0.93	**0.78**

Source: Author's own calculation.

In the case of the hybrid models, the RNN-LSTM model obtained RMSE, MAE, and MAPE values of 69.07, 53.27, and 0.93 percent, respectively, indicating a slightly lower performance compared to the standalone LSTM model. In contrast, the GRU-LSTM model achieved an RMSE of 59.82, an MAE of 44.53, and a MAPE of 0.78 percent, making it the best-performing model of all the considered models. Figure 4 illustrates the predictive performance of all models on the test dataset.

Figure 4: *Actual and Predicted Values of S&P 500 Closing Index, (a) RNN, (b) GRU, (c) LSTM, (d) RNN-LSTM, (e) GRU-LSTM*

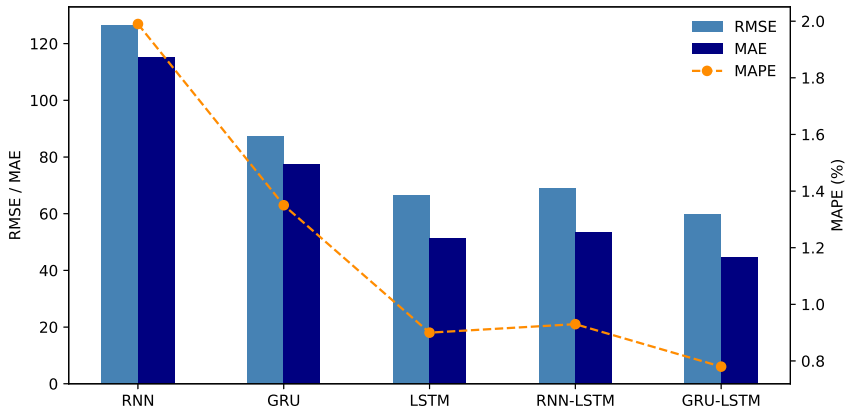


Source: Author's own illustration.

When all empirical results are considered collectively, the GRU and LSTM models, which feature gate mechanisms in their architectures, and the hybrid GRU-LSTM models built using these components, consistently demonstrated superior performance in capturing the long-term dependencies and nonlinear patterns characteristic of financial time series such as the S&P 500. In particular, the GRU-LSTM model provides the best results when all performance criteria are taken into account. The GRU-LSTM model outperformed the RNN model by approximately 53 percent, 61 percent, and 61 percent, respectively, when the RMSE, MAE, and MAPE criteria are taken into account. The same model provided 9.9 percent, 13.1 percent, and 13.3 percent improvements, respectively,

in performance criteria compared to its closest competing LSTM model. Figure 5 presents a comparative illustration of RMSE, MAE, and MAPE values across all models.

Figure 5: *RMSE, MAE, and MAPE Comparison of Models*

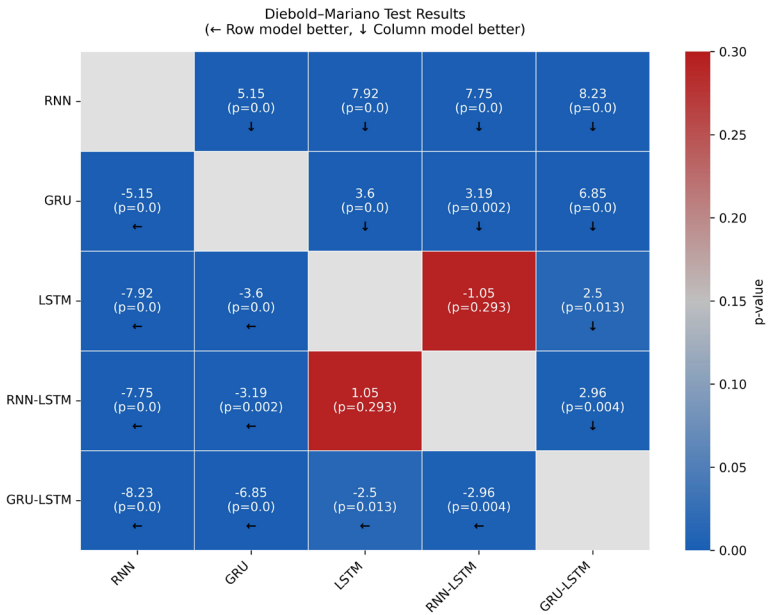


Source: Author's own illustration.

4.5 Diebold–Mariano Test for Forecast Accuracy Comparison

To assess whether the observed differences in forecasting performance across the proposed models are statistically significant, the Diebold–Mariano (DM) test was employed. The DM test evaluates the null hypothesis that two competing forecasting models exhibit equal predictive accuracy, based on the loss differential of their forecast errors. Following the standard methodology introduced by Diebold and Mariano (1995), pairwise comparisons were conducted among all models using squared forecast errors as the loss function.

Figure 6: *Pairwise Diebold–Mariano (DM) Test Results for Model Comparisons*



Source: Author's own illustration.

The results of the DM test are summarized in Figure 6, which provides a comprehensive visual comparison of predictive accuracy across model pairs. The figure presents a heatmap representation of the pairwise DM statistics along with their corresponding *p*-values, allowing for an intuitive assessment of statistical significance. Blue shades indicate statistically significant differences in forecast accuracy at the 5 percent level ($p < 0.05$), while red shades denote non-significant differences. Directional arrows are used to identify the dominant model in each comparison, with leftward arrows indicating superior performance of the row model and downward arrows indicating superior performance of the column model.

The overall pattern displayed in Figure 6 clearly demonstrates that the hybrid GRU-LSTM model achieves the most consistent and statistically significant

improvements in forecasting accuracy when compared with both standalone architectures (RNN, GRU, and LSTM) and the alternative hybrid configuration (RNN-LSTM). In particular, the concentration of significant test outcomes favoring the GRU-LSTM model provides strong statistical evidence that its superior predictive performance is not driven by random variation but reflects a genuine improvement in forecast accuracy. These findings reinforce the effectiveness of hybrid deep learning architectures, especially those combining complementary recurrent memory mechanisms, in financial time series forecasting.

The empirical findings of this study are largely consistent with previous research documenting the superior performance of gated recurrent architectures, such as GRU and LSTM, over simple RNN models in financial time series forecasting (Gao et al., 2021). In line with earlier studies, the results confirm that hybrid deep learning architectures tend to outperform standalone models by jointly capturing short-term dynamics and long-term dependencies (Qiu et al., 2020; Islam & Hossain, 2021). However, this study also provides new insights by offering a controlled and systematic comparison of RNN-LSTM and GRU-LSTM hybrid structures under identical data, optimization, and evaluation settings. The results demonstrate that the GRU-LSTM architecture consistently achieves superior forecasting accuracy, and the Diebold–Mariano test further confirms that these improvements are statistically significant. In this respect, the study not only confirms existing conclusions in the literature but also extends them by highlighting the relative effectiveness of alternative hybrid recurrent configurations.

5 Conclusion

Forecasting financial time series is of great importance to investors and financial institutions due to their volatile and complex nature. For such a critical task, DL methods have recently gained considerable attention and are strengthening their place in academic research. In this study, DL techniques and hybrid models are applied to predict the S&P 500 Index closing values for the period between 2022 and early 2025. The results of the analysis indicate that the RNN architecture struggles to capture long-term dependencies in time series data. However, architectures such as GRU and LSTM show a better ability to capture long-term dependencies due to their gating mechanisms. In particular, the LSTM architecture achieved the second-best overall performance and ranked first among standalone models. Examining hybrid models, the RNN-LSTM model produced results very close to those of the LSTM model. This finding underscores the potential of hybrid architectures, especially when compared to the standalone RNN. The top-performing model, GRU-LSTM, achieved an approximately 10 percent improvement in prediction performance over the LSTM model in terms of the performance criteria. These results suggest that both the GRU and LSTM models are effective in capturing the volatility and long-term dependencies in financial time series data.

Investors, financial institutions, investment strategists, and risk managers should consider advanced DL methods and their hybrid derivatives as alternatives to traditional econometric approaches. The performance of these methods can be further enhanced by integrating features derived from additional technical indicators and macroeconomic variables and by applying data preprocessing techniques (e.g., smoothing). In addition, the complex and volatile nature of financial time series requires the use of more recent approaches, such as attention mechanisms or alternative hybrid models. Improving the performance of different DL methods contributes to more accurate risk assessment and identification of potential opportunities for investors. In conclusion, the use of hybrid methods such as GRU-LSTM, as applied in this study, holds considerable potential for achieving superior long-term forecasting performance in financial time series and improving the accuracy of decision-making.

Literature

Al-hnaity, B., & Abbod, M. (2016). Predicting financial time series data using hybrid model. In Y. Bi, S. Kapoor, & R. Bhatia (Eds.), *Intelligent systems and applications: Extended and selected results from the SAI Intelligent Systems Conference (IntelliSys) 2015* (pp. 19–41). Springer International Publishing.

Ampountolas, A. (2023). Comparative analysis of machine learning, hybrid, and deep learning forecasting models: Evidence from European financial markets and bitcoins. *Forecasting, 5*(2), 472–486. https://doi.org/10.3390/forecast5020026

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

Barra, S., Carta, S. M., Corriga, A., Podda, A. S., & Recupero, D. R. (2020). Deep learning and time series-to-image encoding for financial forecasting. *IEEE/CAA Journal of Automatica Sinica, 7*(3), 683–692. https://doi.org/10.1109/JAS.2020.1003132

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In G. Montavon, G. B. Orr, & K.-R. Müller (Eds.), *Neural networks: Tricks of the trade* (pp. 437–478). Springer. https://doi.org/10.1007/978-3-642-35289-8_26

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research, 13*, 281–305. https://www.jmlr.org/papers/v13/bergstra12a.html

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control*. John Wiley & Sons.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1724–1734). https://doi.org/10.3115/v1/D14-1179

Dao, K. V. (2024). *Evaluating hybrid deep learning models for stock price forecasting: A comparison of LSTM-GRU and transformer-GRU architectures*. SSRN. https://ssrn.com/abstract=5023739

Das, J. D., Thulasiram, R. K., Henry, C., & Thavaneswaran, A. (2024). Encoder–decoder based LSTM and GRU architectures for stocks and cryptocurrency prediction. *Journal of Risk and Financial Management*, *17*(5), 200. https://doi.org/10.3390/jrfm17050200

Das, S. R., Mokashi, K., & Culkin, R. (2018). Are markets truly efficient? Experiments using deep learning algorithms for market movement prediction. *Algorithms, 11*(9), 138. https://doi.org/10.3390/a11090138

Diebold, F. X., & Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics, 13*(3), 253–263. https://doi.org/10.1080/07350015.1995.10524599

Firmino, P. R. A., de Mattos Neto, P. S., & Ferreira, T. A. (2015). Error modeling approach to improve time series forecasters. *Neurocomputing, 153*, 242–254. https://doi.org/10.1016/j.neucom.2014.11.030

Foster, W. R., Collopy, F., & Ungar, L. H. (1992). Neural network forecasting of short, noisy time series. *Computers & Chemical Engineering, 16*(4), 293–297. https://doi.org/10.1016/0098-1354(92)80049-F

Freeborough, W., & van Zyl, T. (2022). Investigating explainability methods in recurrent neural network architectures for financial time series data. *Applied Sciences, 12*(3), 1427. https://doi.org/10.3390/app12031427

Gao, Y., Wang, R., & Zhou, E. (2021). Stock prediction based on optimized LSTM and GRU models. *Scientific Programming*, *2021*, 4055281. https://doi.org/10.1155/2021/4055281

Ge, Q. (2025). Enhancing stock market forecasting: A hybrid model for accurate prediction of S&P 500 and CSI 300 future prices. *Expert Systems with Applications, 260*, 125380. https://doi.org/10.1016/j.eswa.2024.125380

Gür, Y. E. (2024). Comparative analysis of deep learning models for silver price prediction: CNN, LSTM, GRU and hybrid approach. *Akdeniz İİBF Dergisi, 24*(1), 1–13. https://doi.org/10.25294/auiibfd.1404173

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). *Improving neural networks by preventing co-adaptation of feature detectors*. arXiv:1207.0580. https://doi.org/10.48550/arXiv.1207.0580

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

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(2), 107–116. https://doi.org/10.1142/S0218488598000094

Hu, Y., Ni, J., & Wen, L. (2020). A hybrid deep learning approach by integrating LSTM-ANN networks with GARCH model for copper price volatility prediction. *Physica A: Statistical Mechanics and Its Applications, 557*, 124907. https://doi.org/10.1016/j.physa.2020.124907

Hussain, A. J., Knowles, A., Lisboa, P. J. G., & El-Deredy, W. (2008). Financial time series prediction using polynomial pipelined neural networks. *Expert Systems with Applications*, *35*(3), 1186–1199. https://doi.org/10.1016/j.eswa.2007.08.038

Ince, H., & Trafalis, T. B. (2006). A hybrid model for exchange rate prediction. *Decision Support Systems, 42*(2), 1054–1062. https://doi.org/10.1016/j.dss.2005.09.001

Islam, M. S., & Hossain, E. (2021). Foreign exchange currency rate prediction using a GRU-LSTM hybrid network. *Soft Computing Letters, 3*, 100009. https://doi.org/10.1016/j.socl.2020.100009

Jianwei, E., Ye, J., & Jin, H. (2019). A novel hybrid model on the prediction of time series and its application for the gold price analysis and forecasting. *Physica A: Statistical Mechanics and Its Applications, 527*, 121454. https://doi.org/10.1016/j.physa.2019.121454

Kabir, M. R., Bhadra, D., Ridoy, M., & Milanova, M. (2025). LSTM–transformer-based robust hybrid deep learning model for financial time series forecasting. *Sci, 7*(1), 7. https://doi.org/10.3390/sci7010007

Khashei, M., & Bijari, M. (2011). A novel hybridization of artificial neural networks and ARIMA models for time series forecasting. *Applied Soft Computing, 11*(2), 2664–2675. https://doi.org/10.1016/j.asoc.2010.10.015

Khashei, M., & Bijari, M. (2012). A new class of hybrid models for time series forecasting. *Expert Systems with Applications, 39*(4), 4344–4357. https://doi.org/10.1016/j.eswa.2011.09.157

Kim, H.-J., & Shin, K.-S. (2007). A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets. *Applied Soft Computing, 7*(2), 569–576. https://doi.org/10.1016/j.asoc.2006.03.004

Kim, H. Y., & Won, C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications, 103*, 25–37. https://doi.org/10.1016/j.eswa.2018.03.002

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1412.6980

Koduru, G. K., Kumar, T. S., Reddy, B. M., Sai, A. C., & Reddy, S. R. N. (2025). Bitcoin price trends: A neural network approach with RNN & LSTM. *2025 International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1207–1213). IEEE. https://doi.org/10.1109/ICICCS65191.2025.10984756

Kumar, B., Sunil, & Yadav, N. (2023). A novel hybrid model combining βSARMA and LSTM for time series forecasting. *Applied Soft Computing, 134*, 110019. https://doi.org/10.1016/j.asoc.2023.110019

Li, Y., Chen, L., Sun, C., Liu, G., Chen, C., & Zhang, Y. (2024). Accurate stock price forecasting based on deep learning and hierarchical frequency decomposition. *IEEE Access, 12*, 49878–49894. https://doi.org/10.1109/ACCESS.2024.3384430

Liu, H., & Long, Z. (2020). An improved deep learning model for predicting stock market price time series. *Digital Signal Processing, 102*, 102741. https://doi.org/10.1016/j.dsp.2020.102741

Luo, Z., Guo, W., Liu, Q., & Zhang, Z. (2021). A hybrid model for financial time-series forecasting based on mixed methodologies. *Expert Systems, 38*(2), e12633. https://doi.org/10.1111/exsy.12633

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., & Salwana, E. (2020). Deep learning for stock market prediction. *Entropy, 22*(8), 840. https://doi.org/10.3390/e22080840

Pirani, M., Thakkar, P., Jivrani, P., Bohara, M. H., & Garg, D. (2022). A comparative analysis of ARIMA, GRU, LSTM and BiLSTM on financial time series forecasting. *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*. https://doi.org/10.1109/ICDCECE53908.2022.9793213

Pokou, F., Sadefo Kamdem, J., & Benhmad, F. (2024). Hybridization of ARIMA with learning models for forecasting of stock market time series. *Computational Economics, 63*(3), 1349–1399. https://doi.org/10.1007/s10614-023-10499-9

Qiu, Y., Yang, H.-Y., Lu, S., & Chen, W. (2020). A novel hybrid model based on recurrent neural networks for stock market timing. *Soft Computing, 24*(20), 15273–15290. https://doi.org/10.1007/s00500-020-04862-3

Raman, R., Kumar, V., Pillai, B. G., Rabadiya, D., Divekar, R., & Vachharajani, H. (2024). Forecasting bitcoin value with hybrid LSTM-GRU neural networks. *2024 Second International Conference on Data Science and Information System (ICDSIS)*. https://doi.org/10.1109/ICDSIS61070.2024.10594710

Rather, A. M., Agarwal, A., & Sastry, V. N. (2015). Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications, 42*(6), 3234–3241. https://doi.org/10.1016/j.eswa.2014.12.003

Ren, S., Wang, X., Zhou, X., & Zhou, Y. (2023). A novel hybrid model for stock price forecasting integrating Encoder Forest and Informer. *Expert Systems with Applications, 234*, 121080. https://doi.org/10.1016/j.eswa.2023.121080

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Sako, K., Mpinda, B. N., & Rodrigues, P. C. (2022). Neural networks for financial time series forecasting. *Entropy, 24*(5), 657. https://doi.org/10.3390/e24050657

Santos Júnior, D. S. de O., de Oliveira, J. F. L., & de Mattos Neto, P. S. G. (2019). An intelligent hybridization of ARIMA with machine learning models for time series forecasting. *Knowledge-Based Systems, 175*, 72–86. https://doi.org/10.1016/j.knosys.2019.03.011

Seabe, P. L., Moutsinga, C. R. B., & Pindza, E. (2023). Forecasting cryptocurrency prices using LSTM, GRU, and bi-directional LSTM: A deep learning approach. *Fractal and Fractional, 7*(2), 203. https://doi.org/10.3390/fractalfract7020203

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing, 90*, 106181. https://doi.org/10.1016/j.asoc.2020.106181

Song, H., & Choi, H. (2023). Forecasting stock market indices using the recurrent neural network-based hybrid models: CNN-LSTM, GRU-CNN, and ensemble models. *Applied Sciences, 13*(7), 4644. https://doi.org/10.3390/app13074644

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*(1), 1929–1958. https://jmlr.org/papers/v15/srivastava14a.html

Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera: Neural Networks for Machine Learning, 4*(2), 26–31.

Wei, L.-Y. (2016). A hybrid ANFIS model based on empirical mode decomposition for stock time series forecasting. *Applied Soft Computing, 42*, 368–376. https://doi.org/10.1016/j.asoc.2016.01.027

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

Xu, Q., Zhuo, X., Jiang, C., & Liu, Y. (2019). An artificial neural network for mixed frequency data. *Expert Systems with Applications*, *118*, 127–139. https://doi.org/10.1016/j.eswa.2018.10.013

Yang, S. (2021). A novel study on deep learning framework to predict and analyze the financial time series information. *Future Generation Computer Systems, 125*, 812–819. https://doi.org/10.1016/j.future.2021.07.017

Yunita, A., Pratama, M. I., Almuzakki, M. Z., Ramadhan, H., Akhir, E. A. P., Mansur, A. B. F., & Basori, A. H. (2025). Performance analysis of neural network architectures for time series forecasting: A comparative study of RNN, LSTM, GRU, and hybrid models. *MethodsX, 15*, 103462. https://doi.org/10.1016/j.mex.2025.103462

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing, 50*, 159–175. https://doi.org/10.1016/S0925-2312(01)00702-0

Zhang, J., Liu, H., Bai, W., & Li, X. (2024). A hybrid approach of wavelet transform, ARIMA and LSTM model for the share price index futures forecasting. *The North American Journal of Economics and Finance, 69*, 102022. https://doi.org/10.1016/j.najef.2023.102022

Zhao, Y., & Chen, Z. (2022). Forecasting stock price movement: New evidence from a novel hybrid deep learning model. *Journal of Asian Business and Economic Studies, 29*(2), 91–104. https://doi.org/10.1108/JABES-05-2021-0061

Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*, *67*, 126–139. https://doi.org/10.1016/j.eswa.2016.09.027

Zhu, R., Zhong, G.-Y., & Li, J.-C. (2024). Forecasting price in a new hybrid neural network model with machine learning. *Expert Systems with Applications, 249*, 123697. https://doi.org/10.1016/j.eswa.2024.123697