Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

VTG-TLN: Volatility Trend Grouping-Temporal Learning Network for Live Stock Market Index Closing Price Forecasting

Bhuvaneshwari P V[1], Radhakrishnan Vignesh[2]

[1]Presidency School of Artificial Intelligence and Advance Computing, Presidency University, Bangalore, India,
[2]Presidency School of Artificial Intelligence and Advance Computing, Presidency University, Bangalore, India,

E-mail: [1] BHUVANESHWARI.20233CSE0032@presidencyuniversity.in,[2]vigneshr@presidencyuniversity.in

Abstract
Stock market prediction is a key research area because financial markets are highly dynamic and influenced by corporate performance, economic conditions, market sentiments and global events. However, accurately forecasting live stock market index closing prices is difficult because financial time-series data have short-term and long-term trend dependencies with different temporal characteristics. In this research, Volatility Trend Grouping-Temporal Learning Network (VTG-TLN) is proposed for live stock market index closing price forecasting. In the volatility branch, the proposed VTG-TLN employs a Temporal Convolutional Network (TCN) to capture rapid price fluctuations and short-term dependencies, whereas Long Short-Term Memory (LSTM) learns long-term dependencies and persistent patterns in the trend branch. By integrating these representations, VTG-TLN captures both long-term price trends and short-term market variations, which leads to accurate live stock market index closing price forecasting. Compared with existing External Trend and Internal Components Analysis (ETICA)-LSTM, proposed VTG-TLN method achieves a lower Mean Absolute Error (MAE) of 22.65 and 77.87 on S&P 500 and NASDAQ datasets.

Keywords: Economic conditions, Long Short-Term Memory, Mean Absolute Error, Live stock market index closing price forecasting, Temporal Convolutional Network.

1. Introduction

The stock market is a major indicator of a country's economic health and acts as the backbone of the global financial system. Consequently, investors, governments, and industry stakeholders monitor stock market fluctuations to assess economic conditions, determine investment opportunities, and support informed financial decision-making [1]. Recently, around 60 stock exchanges worldwide have represented nearly $125 trillion [2] in the total market capitalization of global equity markets. The United States (US) stock market is the world's leading equity market, with an overall share of 42% followed by China with a share of nearly 8%. The empirical evaluation of equity markets is a vital research area because billions of dollars are traded in these markets on a daily basis [3]. The stock market is a highly intricate and dynamic system influenced by a wide range of factors, including politics [4], global events, economic [5] indicators, corporate performance, and market sentiment [6]. Understanding the highly intricate and volatile behavior is the primary challenge for investors [7] especially when making informed investment and forecasting decisions. Also, live stock market index closing price forecasting is challenging due to the noisy [8] [9] and unpredictable nature of financial data [10]. Although unpredictability and market volatility [11] introduce risks, it provide opportunities for investors to generate profits via well-timed trading decisions [12]. Artificial Intelligence (AI) has played a vita role in managing diverse tasks in the stock market [13], including risk management [14], robo-advisory services, and stock movements prediction. By analyzing historical stock prices, market trends, and corporate financial statements, AI methods offer valuable insights into future performance [15]. With the development of Machine Learning [16] and Deep Learning (DL) [17] methods, researchers explore flexible and sophisticated forecasting models [18]. ML methods help learn intricate patterns and relationships from historical financial data [19], whereas DL methods extract discriminative features from financial data, enhancing prediction performance and minimizing error [20].

Fatene Dioubi et al. [21] presented an External Trend and Internal Components Analysis (ETICA) decomposition model with LSTM to improve prediction of stock market. ETICA-LSTM provides a better understanding of market dynamics, which enable LSTM to capture intricate temporal patterns. ETICA has the ability to identify the factors influencing stock prices, which provides precise modeling and minimizes error. However, ETICA-LSTM has limited ability to model diverse short-term and long-term temporal patterns because decomposition and a single LSTM do not assign different temporal characteristics to learning branches which leads to high prediction error. Yi Ji et al. [22] established a Transformer-based method with generative decoding and a Mean Square Error (MSE) loss function, called Galformer to forecast stock prices. The generative decoder was used to forecast long time-series sequence in a single forward process, which enhance prediction speed. The MSE loss function integrated trend accuracy and quantitative error of the predicted outcomes which provides feedback and enhances model performance. Nevertheless, Galformer requires high computational complexity because Transformer-based self-attention processes relationships across the input sequence, which increase memory requirements as sequence length increases. Peng Zhu et al. [23] developed a Multi-head Cross-attention mechanism with an Improved Gated Recurrent Unit (MCI-GRU) for forecasting the stock market. The GRU model was improved by replacing the reset gate with an attention mechanism which enhances the model's flexibility. The attention mechanism was used to learn unobservable latent market state representation which was refined through interactions with cross-sectional and temporal features. However, MCI-GRU has limited ability to distinguish short-term volatility from long-term trends because both temporal characteristics were modeled in the same recurrent method.

Xin Wang et al. [24] suggested a Mixed Attention-based Multi-Scale Temporal Network (MA-MSTNet) to predict stock market. Maximal Information Coefficient (MIC) was used for capturing nonlinear dependencies among stock features which minimize noise and complexity. A sliding-window attention-based module was employed to extract local short-term high-frequency information and global long-term low-frequency information that improves the representation of temporal features. An adaptive weighting mechanism dynamically combines the multi-scale features and enhances the model's performance. Adusumalli Balaji et al. [25] introduced a Stock Generative Adversarial Network (StockGAN++) to predict stock price. Initially, z-score normalization was used during pre-processing whereas high-level features were extracted by utilizing stacked autoencoder module. The StockGAN++ was used for prediction by integrating Temporal Convolutional Autoencoder (TCAE) and Gated Graph Convolutional Network (GGCN). To fine-tune the hyperparameters of StockGAN++, Improved Chaotic assisted Grasshopper Optimization (Imp-CGop) was used to enhance the model performance. Kun Huang et al. [26] presented a Multisource information Fusion with Dual-graph Attention Network (MF-DAT) to predict the stock trend. Low-rank Multimodal Fusion (LMF) was used to fuse diverse characteristics extracted from stock data for capturing correlation between market information. The short-term and long-term state characteristics of stocks were learned via graph attention layer. Finally, the node representation of the stock relationship network constructed by determining stock clusters through community detection was updated to capture underlying relationships among stocks. Haein Lee et al. [27] implemented a Bidirectional- Recurrent Neural Network (Bi-RNN) and Bidirectional-LSTM (Bi-LSTM) for forecasting the stock market by incorporating Environmental, Social, and Governance (ESG) sentiment and technical indicators. Stopword removal and lemmatization were applied to news data for removing unwanted text. Then, Financial Bidirectional Encoder Representation from Transformer (FinBERT) was used to calculate the sentiment index. At last, the implemented method was used to predict the future prices. Sanskar Agarwal et al. [28] suggested a Successive Variational Model Decomposition (SVMD) with LSTM to predict stock market. SVMD was used as a data decomposition method to divide the intricate original series into multiple intrinsic model functions which minimize non-stationary and simplify the underlying temporal patterns. Then, LSTM was used to analyze the temporal data for enhancing prediction performance. However, SVMD-LSTM has limited ability to distinguish short-term volatility from long-term trends because decomposed components were modeled without temporal branches.

1.1 Problem Statement and Objective

Accurately forecasting live stock market index closing price remains challenging because financial time-series data have long-term and short-term dependencies with different temporal characteristics. To solve this issue, Volatility Trend Grouping-Temporal Learning Network (VTG-TLN) is proposed for live stock market index closing price forecasting. Multivariate Empirical Mode Decomposition (MEMD) decomposes the Open High Low Close Volume (OHLCV) features into multiple Intrinsic Mode Functions (IMF), while sample entropy scale clustering groups the IMF based on temporal complexity to determine trend and volatility information. Based on this grouping, the volatility branch employs five OHLCV features and four

candlestick numerical features, whereas the trend branch utilizes the five OHLCV features. The Temporal Convolutional Network (TCN) is used to capture rapid short-term dependencies in volatility branch whereas LSTM-based trend branch learns long-term temporal dependencies. The volatility and trend branches are concatenated and mapped to a 5-dimensional output representing the closing prices for the next five trading days, which enables the model to capture both short and long-term market dynamics.

1.2 Contribution

The primary contribution of the proposed method are as follows:

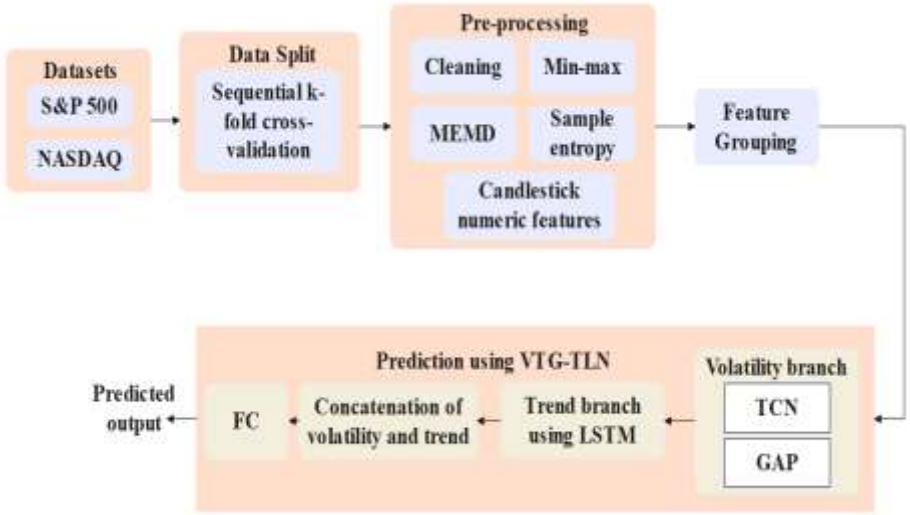
- MEMD is used to decompose the OHLCV features into multiple IMFs, with each IMF representing market variations at diverse temporal scale. This decomposition separates rapidly changing fluctuations from slower market patterns, providing scale-specific information for volatility trend grouping.
- In the volatility branch, TCN method is used to capture local temporal dependencies and short-term price fluctuations in the volatility-related features. The dilated convolutions with increasing dilation rates enhance the temporal receptive field, preserve the chronological order of observations and enable the model to learn local fluctuations and short-term patterns.
- LSTM captures long-term dependencies by retaining relevant information from earlier observations across the 20 day trend sequence. The LSTM layers refine the temporal representations and allow the model to distinguish long-term movements from short-term fluctuations.

The remaining section are presented as follows: Section 2 presents the proposed methodology; Section 3 analyzes the experimental results; and Section 4 presents the conclusion.

2. Proposed Methodology

This research proposes the VTG-TLN method to forecast the live stock market index closing prices using the S&P 500 and NASDAQ datasets. During pre-processing, cleaning, min-max normalization, MEMD, sample entropy scale clustering, and candlestick numeric feature generation are used to prepare the input data for the subsequent process. Then, feature grouping is applied to organize the pre-processed features into separate volatility and trend inputs. Finally, VTG-TLN is used for live stock market index closing price forecasting and Figure 1 depicts the overall workflow of proposed methodology.

Figure 1. Overall workflow of the proposed VTG-TLN method for live stock market index closing price forecasting



2.1 Datasets

This research employs the real-world S&P 500 [29] and NASDAQ [30] stock market datasets sourced from Yahoo Finance to evaluate the VTG-TLN method for live stock closing price forecasting. Both datasets comprise daily OHLCV data spanning from February 25, 2016 to July 13, 2026. For both datasets, sequential k-fold (k=6) cross-validation is used, resulting in 2,014 samples for training, 224 samples for validation, and 372 samples for testing with a total of 2,610 trading day samples. Table 1 shows the dataset distribution, data periods, samples, and prediction tasks considered in this research. The obtained data are passed through the pre-processing stage for further processing.

Table 1. Summary of S&P 500 and NASDAQ datasets used for live stock closing price forecasting

Datasets	Total samples (trading days)	Data period	Features	Prediction task
S&P 500	2,610	February 25, 2016 to July 13, 2026	OHLCV	Regression (No classes)
NASDAQ	2,610			

2.2 Pre-processing

During pre-processing, OHLCV features are cleaned to eliminate duplicates and handle missing values followed by min-max normalization using only training to prevent data leakage. Then, MEMD is used to decompose the normalized data into IMF by preserving inter-variable relationships. For each IMF, sample entropy is computed to measure its complexity and the IMF are divided into volatility and trend components by employing median entropy as the threshold. At last, numerical candlestick features are obtained from the cleaned OHLCV data to capture local price-action characteristics. A detailed explanation of these methods is provided below.

OHLCV cleaning: Initially, the OHLCV features are cleaned by removing duplicate records and handling missing values in trading-day observations. This ensures that the data are chronologically consistent for further processing.

Min-max normalization: The cleaned OHLCV features are normalized by employing min-max scaling in the range of [0,1] because variables have different numerical scales, especially price values and trading volume. Without normalization, data with high magnitudes have a greater influence during model training. Min-max [31] normalization conserves the relative variation and temporal patterns of original data which provides consistent scale for the model. To avoid data leakage, min-max normalization is applied independently within each fold. Typically, the minimum and maximum values are calculated only from the training partition of each fold and then applied to test and validation partitions of same fold. Hence, no information from test and validation partitions are employed to compute the normalization statistics. The mathematical formula for min-max normalization is given in equation (1)

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

Where X_{scaled} represents normalized data, X denotes input, X_{min}, X_{max} illustrates minimum and maximum value calculated from training data.

MEMD: The normalized features are decomposed using MEMD into multiple IMFs to capture intricate and non-stationary financial data. Although min-max normalization changes the amplitude of time-series value to common range, it does not modify the temporal order and variation patterns. Hence, MEMD separates the normalized time series into components characterized by different rates of temporal variation while preserving the relationships among OHLCV variables. The resulting IMFs are grouped into trend and volatility components using sample entropy.

Sample entropy scale clustering: Sample entropy is computed independently for each IMF obtained from MEMD to measure its irregularity and complexity. Based on these entropy values, the IMFs are divided into two groups like trend and volatility. The median sample entropy is employed as the threshold, where higher-entropy IMFs are assigned to the volatility group and lower or equal entropy IMFs are assigned to trend group. This provides separate representations of short-term fluctuations and long-term market movements.

Candlestick numeric features: To capture short-term price action information, four derived candlestick features (upper-wick ratio, body ratio, lower-wick ratio, and binary doji indicator) are computed from OHLC values. These features are synchronized with corresponding trading days of OHLCV and concatenated along the feature dimension. Thus, the four derived features and five OHLCV features form the nine input channels of the volatility branch which characterize short-term price movements and local market fluctuations. Table 2 shows the window length and input channel configuration used for both datasets. The preprocessed data are fed into the feature grouping module to derive informative market characteristics.

Table 2. Configuration of input window length and channel dimensions used for volatility and trend components

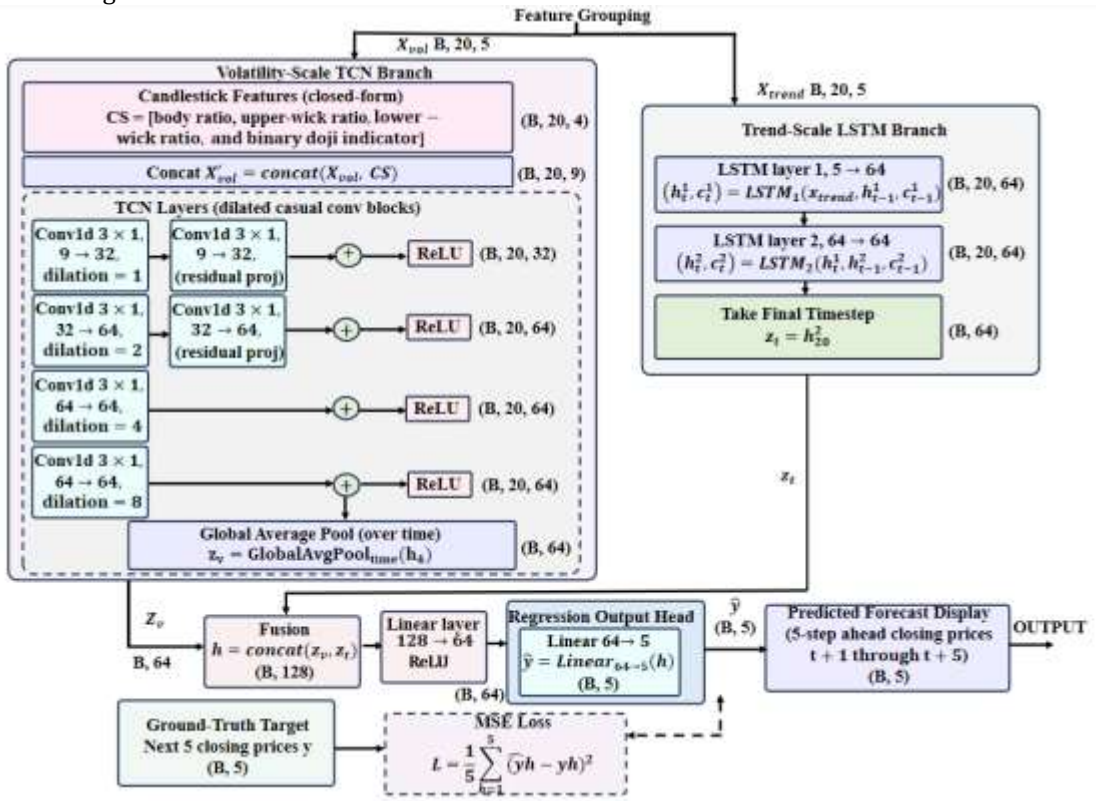
Datasets	Window length (Days)	Input channels
S&P 500	20	9 (volatility branch), 5 (trend branch)
NASDAQ	20	9 (volatility branch), 5 (trend branch)

2.3 Feature Grouping

After pre-processing, the input features are structured into two branches based on temporal characteristics. The volatility branch contains 5 OHLCV features and 4 derived candlestick features, resulting in 9 channels per timestep. The trend branch retains only 5 OHLCV features, resulting in 5 channels per timestep because candlestick derived features primarily characterize short-term price behavior. No manual feature selection stage is performed; instead, MEMD decomposition and sample entropy scale clustering determine relevant information while retaining the grouped input channels. Hence, the 20 day input windows are denoted as (B, 20, 9) for volatility branch and (B, 20, 5) for trend branch which are passed through the temporal branches of VTG-TLN to forecast future stock closing prices.

2.4 Live Stock Market Index Closing Price Forecasting using Volatility Trend Grouping-Temporal Learning Network (VTG-TLN)

Following feature grouping, the proposed VTG-TLN processes the resulting trend and volatility inputs through temporal modeling branches to capture short and long-term stock market dynamics. The volatility input is denoted as (B, 20, 9) and is processed by employing TCN to capture short-term price fluctuations and local temporal dependencies. Then, a Global Average Pooling (GAP) generates a 64-dimensional volatility representation. The trend input is represented as (B, 20, 5) and is processed by an LSTM layer to capture long-term temporal dependencies, producing a 64-dimensional trend representation. These two representations are concatenated to form a 128-dimensional fused representation which is fed into Fully Connected (FC) layers to produce the 5-dimensional closing price forecast. Figure 2 shows an architecture of the proposed VTG-TLN method for live stock closing price forecasting.



2.4.1 Volatility Branch using Temporal Convolutional Network (TCN)

Short-term stock volatility is characterized by local dependencies, rapid fluctuations, and sudden changes in price movement. In the volatility branch, TCN is used instead of CNN because its dilated convolutions offer a larger multi-scale temporal receptive field, preserve temporal ordering and ensure effective processing. The input passes through a convolution layer, where each layer employs a filter to extract relevant temporal patterns. Compared with a convolutional network, TCN utilizes dilated convolutions that introduce fixed gaps between filter components and expand the temporal receptive field. By expanding the dilation rate across layers like d = 1, 2, 4, 8, TCN captures dependencies over time horizons without increasing computational complexity. For a kernel size k=3, left padding is computed as (k-1)d, resulting in left padding values of 2, 4, 8, and 16 with dilation rates of 1, 2, 4, and 8. This ensures that the output at time step t is based only on the present and preceding inputs ($\leq t$), thereby preventing information leakage from future time steps. No output trimming is needed because the left padding conserves the sequence length. A primary element of TCN is the residual module which facilitates the training of deep networks. To enhance and stabilize training, Rectified Linear Unit (ReLU) introduces non-linearity to learn intricate patterns. Furthermore, residual connections are applied in the first two TCN blocks, where a 1×1 convolution is used to match the input and output channel dimension before residual addition. These residual connections allow the transformed input to bypass the convolutional layers and provide gradient flow with stable training.

2.4.2 Global Average Pooling (GAP)

After the final TCN layer, the resulting feature maps have 64 channels across the temporal dimension. The GAP layer averages the activation values of each channel across all time steps by converting the (B, 64, 20) feature map into a (B, 64) representation. This fixed-length volatility representation is passed to the LSTM for learning temporal dependencies.

2.4.3 Trend Branch using Long Short Term Memory (LSTM)

Long-term stock trends contain sequential dependencies, where information from earlier observations affects later market behavior. LSTM is preferred because its gates manage the retention, updating, and removal of temporal information which allow relevant historical information to persist across the sequence. The LSTM layers learn hierarchical temporal representation, with the 1st layer capturing lower-level temporal dependencies and the 2nd layer refining long-term trend representations. Compared with LSTM, GRU is a simpler and computationally efficient gated structure, but its reset and update gates integrate the memory-control process and do not provide separate cell-state mechanism like LSTM which provide less control over the retention of long-term temporal information. Similarly, the transformer model is not chosen because its self-attention mechanism has high computational cost and parameter requirements, while the 20 days input window does not need extensive global attention for the trend branch. Hence, LSTM is chosen because it provide balance between memory control, computational complexity, and temporal dependency modeling for the trend branch.

The forget gate evaluates which previous information is retained, input gate manage the new information, and output gate handles the information passed to hidden state. The forget gate is formulated in equation (2) whereas the candidate state and input gate are expressed in equations (3) and (4). The cell state is represented in equation (5) while output gate and hidden states are provided in equations (6) and (7).

$$f_t = \sigma(x_t W_f + h_{t-1} W_f + b_f) \quad (2)$$

$$g_t = tanh(x_t W_g + h_{t-1} W_g + b_g) \quad (3)$$

$$i_t = \sigma(x_t i + h_{t-1} W_i + b_i) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + g_t \odot i_t \quad (5)$$

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

$$h_t = o \odot tanh c_t \quad (7)$$

Where x_t illustrate input vector at time step t, h_{t-1} and c_{t-1} refer to previous hidden and cell states, f_t represent forget gate, i_t determine input gate, g_t demonstrate candidate state, o_t denote output gate, and W_f, W_g, W_i, W_o represent weight matrices of forget gate, candidate state, input gate, and output gate. The

b_f, b_g, b_i, b_o refer to bias vectors of forget gate, candidate state, input gate, and output gate, c_t and h_t denotes current cell state and hidden state, $\sigma(.)$ and $tanh$ illustrate sigmoid and tangent activation function, and $\odot$ denote element-wise multiplication.

2.4.4 Concatenation and Fully Connected (FC) Layer

Following the TCN-based volatility branch and LSTM-based trend branch, the resulting 64-dimensional volatility and 64-dimensional trend representations are integrated to form a unified representation. The two representations are concatenated to produce a 128-dimensional fused representation. This concatenated representation contains short-term volatility patterns and long-term trend patterns which are fed into FC layer and minimize dimensionality from 128 to 64. Then, ReLU activation is used to introduce non-linearity and improve the learning of fused representation. Finally, the second FC layer maps the 64-dimensional representation to a 5-dimensional output that generates the final live stock market index closing price.

The hyperparameters are chosen based on grid search to ensure optimal performance for proposed, baseline, and existing methods. The MSE loss function is used over a 5-step prediction horizon because it calculates the squared difference between predicted and actual values at each step. The Adam optimizer is used with learning rate of $3e^{-4}$ for the volatility branch and $1e^{-3}$ for the trend branch to provide adaptive parameter updates. A batch size of 64 is employed to balance gradient stability and computational efficiency. Moreover, training is performed for a maximum of 150 epochs with early stopping based on validation MSE using a patience of 12 to minimize overfitting. For the MEMD method, the standard sifting stopping criterion is used to identify the IMFs of each fold. Table 3 presents the layer-wise configuration of the proposed VTG-TLN method. Table 4 lists the mathematical notation and definition used throughout the methodology and Algorithm 1 shows a pseudocode of VTG-TLN to ensure reproducibility.

Table 3. Layer-wise configuration and input output dimensions of VTG-TLN for live stock market index closing price forecasting

Layer	Input	Output	Layer type
vol.tcn1	9	32	Conv1D (k=3, d=1)
vol.tcn1 residual	9	32	Conv1D (k=1)
vol.tcn2	32	64	Conv1D (k=3, d=2)
vol.tcn2 residual	32	64	Conv1D (k=1)
vol.tcn3	64	64	Conv1D (k=3, d=4)
vol.tcn4	64	64	Conv1D (k=3, d=8)
vol.global avg pool	64	64	GAP
trend.lstm1	5	64	LSTM
trend.lstm2	64	64	LSTM
fusion.concat	64+64	128	Concatenation
fusion.fc1	128	64	Linear
fusion.relu	64	64	ReLU
fusion.fc2	64	5	Linear

Table 4. Notation and definitions of symbols used in proposed VTG-TLN method for live stock market index closing price forecasting

Symbols	Definition
X_{scaled}	normalized data
X	input
X_{min}, X_{max}	minimum and maximum value calculated from training data
x_t	input vector at time step t
h_{t-1} and c_{t-1}	previous hidden and cell states
f_t	forget gate
i_t	input gate
g_t	candidate state
o_t	output gate
W_f, W_g, W_i, W_o	weight matrices of forget gate, candidate state, input gate, and output gate
b_f, b_g, b_i, b_o	bias vectors of forget gate, candidate state, input gate, and output gate
c_t and h_t	current cell state and hidden state

$\sigma(.)$ and $tanh$	sigmoid and tangent activation function
$\odot$	element-wise multiplication

Algorithm 1 of proposed VTG-TLN method

Notation: N represent number of trading-day observation, $L = 20$ denote input-window length, $H = 5$ illustrate forecasting horizons, and N_s determine actual number of input-target sequence employed for model training.

Input: Datasets D={S&P 500, NASDAQ}
Output: Predicted 5-step closing price vector

Start

Step 1: Datasets
Load datasets D
Apply sequential k-fold cross validation

Step 2: Pre-processing
For each dataset D in Input X do
Remove duplicate records and handle missing values
Employ min-max normalization using training set statistics
Apply MEMD and obtain IMF components
For each IMF j=1, 2,…k do
Calculate sample entropy
End For
Compute median entropy threshold
Group IMF components:
High entropy IMF → volatility group
Low/equal entropy IMF → trend group
Calculate four candlestick derived features from OHLC
End For

Step 3: Feature Grouping
Form volatility input using 5 OHLCV features + 4 candlestick features → 9 channels
Form trend input using 5 OHLCV features → 5 channels
Calculate $N_s = N - L - H + 1$
For t=1 to N_s do
Construct 20-day volatility window $W_{volatility}(t) \epsilon R^{(20\times9)}$
Construct 20-day trend window $W_{trend}(t) \epsilon R^{(20\times5)}$
Construct 5 step prediction target $y(t) \epsilon R^{(5)}$
End For

Step 4: Prediction
Construct volatility branch using TCN

$$V1 \leftarrow TCN(W_{volatility}; C = 32, k = 3, d = 1)$$
$$V2 \leftarrow TCN(V1; C = 64, k = 3, d = 2)$$
$$V3 \leftarrow TCN(V2; C = 64, k = 3, d = 4)$$
$$V4 \leftarrow TCN(V3; C = 64, k = 3, d = 8)$$
$$V \leftarrow GAP\ (V4)$$

Obtain volatility representation $V \epsilon R^{(64)}$
Construct trend branch using LSTM

$$T1 \leftarrow LSTM(W_{trend}; hidden = 64)$$
$$T2 \leftarrow LSTM(T1; hidden = 64)$$

Consider final timesteps and obtain trend representation $T \epsilon R^{(64)}$
Perform concatenation

$$F \leftarrow Concatenate(V, T)$$
$$F \epsilon R^{(128)}$$

$$F1 \leftarrow Linear(F; 128 \rightarrow 64)$$
$$F2 \leftarrow ReLU(F1)$$
$$\hat{y}_i \leftarrow Linear(F2; 64 \rightarrow 5)$$

Compute MSE between predicted and actual closing price values

Train VTG-TLN using Adam optimizer with learning rate of $3e^{-4}$ for volatility branch and $1e^{-3}$ for trend branch

Apply early stopping based on validation MSE

Generate final 5-step closing price forecast

Return final 5-step closing price vector
End

3. Results

The proposed VTG-TLN method is simulated using Python 3.11 on system equipped with an Intel(R) Core (TM) i7-7700K CPU @ 4.20GHz processor, 32 GB Random Access Memory (RAM), and an NVIDIA GeForce RTX 3050 with 8 GB memory. The experiments are performed on 64-bit x64-based system running on Windows 10 Pro, version 22H2. The Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and R^2 are used to evaluate model performance, as expressed in equations (8) to (11). The baseline and existing methods are reimplemented under same settings as proposed method, including sequential k-fold cross-validation, pre-processing, hyperparameter details, and hardware/software configuration to ensure fair comparison.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|^2} \quad (8)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (9)$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n} \left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \quad (10)$$

$$R^2 = 1 - \frac{MSE(y_i - \hat{y}_i)}{Var(y_i)} \quad (11)$$

Where y_i represent actual values, n denote number of sample, Var refers to variance, and $\hat{y}_i$ determine predicted values.

3.1 Performance Analysis

Table 5 presents an analysis of different DL methods for prediction. Compared with other methods, VTG-TLN achieves lower MAE due to integrating short-term volatility patterns from TCN and long-term trend dependencies from LSTM which provides complementary temporal representation. MEMD and sample-entropy grouping separate intricate market variations into volatility and trend components which minimize non-stationary fluctuations. This ensure accurate estimation of stock movements and minimizes error on both datasets. The mean ± Standard Deviation (SD) is used to measure the variability of model performance across 24 independent runs.

Table 5. Analysis of different DL methods for the first forecast step (t+1) of live stock market index closing price forecasting (Reimplemented)

Datasets	Methods	R^2	RMSE	MAE	MAPE (%)
S&P 500	RNN without MEMD	0.9790 ± 0.0028	66.81 ± 4.77	50.61 ± 3.62	1.050 ± 0.075
	LSTM without MEMD	0.9715 ± 0.0032	77.62 ± 5.47	58.80 ± 4.15	1.220 ± 0.086
	GRU without MEMD	0.9822 ± 0.0022	58.53 ± 4.14	44.34 ± 3.13	0.920 ± 0.065
	Transformer without MEMD	0.9895 ± 0.0014	33.72 ± 2.61	25.55 ± 1.98	0.530 ± 0.041
	VTG-TLN	**0.9915 ± 0.0012**	**29.90 ± 2.29**	**22.65 ± 1.74**	**0.470 ± 0.036**
NASDAQ	RNN without MEMD	0.9765 ± 0.0028	220.92 ± 15.78	173.96 ± 12.43	1.134 ± 0.081
	LSTM without MEMD	0.9690 ± 0.0032	256.69 ± 18.09	202.12 ± 14.25	1.318 ± 0.093
	GRU without	0.9797 ±	193.57 ±	152.42 ±	0.994 ± 0.070

	MEMD	0.0022	13.68	10.77	
	Transformer without MEMD	0.9870 ± 0.0014	111.51 ± 8.63	87.81 ± 6.79	0.572 ± 0.044
	VTG-TLN	**0.9890 ± 0.0012**	**98.89 ± 7.57**	**77.87 ± 5.96**	**0.508 ± 0.039**

Table 6 represents the fold-wise analysis of the VTG-TLN method on different datasets. The small variation between folds shows that VTG-TLN offers stable and consistent predictive performance across diverse data partitions. This consistency is due to model's ability to learn long-term temporal dependencies via LSTM and short-term temporal dependencies by TCN. The mean ±SD values denote the robustness of proposed method as the low SD indicates limited performance fluctuations among the six folds.

Table 6. k-fold wise performance of proposed VTG-TLN method for the first forecast step (t+1) on S&P 500 and NASDAQ datasets (Reimplemented)

Datasets	Folds	R^2	RMSE	MAE	MAPE (%)
S&P 500	Fold 1	0.9933	31.50	23.87	0.495
	Fold 2	0.9913	32.99	24.99	0.519
	Fold 3	0.9908	27.81	21.07	0.437
	Fold 4	0.9926	29.67	22.48	0.466
	Fold 5	0.9904	26.86	20.35	0.422
	Fold 6	0.9905	30.58	23.17	0.481
	Mean± SD	**0.9915 ± 0.0012**	**29.90 ± 2.29**	**22.65 ± 1.74**	**0.470 ± 0.036**
NASDAQ	Fold 1	0.9908	104.18	82.03	0.535
	Fold 2	0.9888	109.10	85.91	0.560
	Fold 3	0.9883	91.98	72.42	0.472
	Fold 4	0.9901	98.12	77.26	0.504
	Fold 5	0.9879	88.83	69.94	0.456
	Fold 6	0.9880	101.13	79.63	0.519
	Mean± SD	**0.9890 ± 0.0012**	**98.89 ± 7.57**	**77.87 ± 5.96**	**0.508 ± 0.039**

Table 7 shows the cross-dataset generalization performance of proposed method when trained on one dataset and tested on another dataset. Initially, the model is trained on S&P 500 and tested on NASDAQ dataset. Then, the model is trained on NASDAQ and tested on S&P 500 dataset. The outcomes shows that that proposed VTG-TLN maintains high predictive performance when evaluated on a different market from the training dataset. This denotes that VTG-TLN learns temporal patterns and generalize across different stock market datasets.

Table 7. Cross-dataset generalization performance of VTG-TLN method for the first forecast step (t+1) (Reimplemented)

Training dataset	Testing dataset	R^2	RMSE	MAE	MAPE (%)
S&P 500	NASDAQ	0.9815±0.0016	123.61±9.84	97.33±7.76	0.634±0.052
NASDAQ	S&P 500	0.9790±0.0015	43.60±3.47	33.03±2.71	0.685±0.057

Table 8 shows the ablation study to evaluate the contribution of each component in the VTG-TLN method. The VTG-TLN obtains lower MAE due to capturing both long-term and short-term market characteristics. The TCN-only variant suffers from limited long-term dependency modeling whereas LSTM-only variant does not capture short-term volatility. Moreover, removing candlestick features minimizes local price-action information while eliminating MEMD limits the separation of intricate temporal variations. Hence, integrating TCN, LSTM, MEMD, and candlestick features offers a comprehensive representation of market dynamics which leads to lower prediction error than individual ablation variants.

Table 8. Ablation study to analyze the contribution of each component in the VTG-TLN method for the first forecast step (t+1) (Reimplemented)

Datasets	Components	R^2	RMSE	MAE	MAPE

					(%)
S&P 500	TCN+GAP+FC (without LSTM)	0.9895 ± 0.0013	33.72 ± 2.40	25.55 ± 1.82	0.530 ± 0.038
	LSTM+FC (without TCN)	0.9911 ± 0.0013	30.54 ± 2.52	23.14 ± 1.91	0.480 ±0.040
	VTG-TLN (without candlestick numeric features)	0.9906 ± 0.0014	31.81 ± 2.63	24.10 ± 2.00	0.500 ± 0.041
	VTG-TLN (without MEMD)	0.9878 ± 0.0014	38.81 ± 2.75	29.40 ± 2.08	0.610 ± 0.043
	VTG-TLN	**0.9915 ± 0.0012**	**29.90 ± 2.29**	**22.65 ± 1.74**	**0.470 ± 0.036**
NASDAQ	TCN+GAP+FC (without LSTM)	0.9851 ± 0.0014	128.36 ± 9.09	101.00 ± 7.16	0.666 ± 0.047
	LSTM+FC (without TCN)	0.9872 ± 0.0013	112.81 ± 7.95	87.83 ± 6.26	0.575 ± 0.041
	VTG-TLN (without candlestick numeric features)	0.9887 ± 0.0013	101.09 ± 8.33	79.55 ± 6.56	0.517 ± 0.043
	VTG-TLN (without MEMD)	0.9883 ± 0.0014	105.22 ± 8.71	82.81 ± 6.86	0.541 ± 0.045
	VTG-TLN	**0.9890 ± 0.0012**	**98.89 ± 7.57**	**77.87 ± 5.96**	**0.508 ± 0.039**

Table 9 shows the performance of the VTG-TLN method across five-step forecasting window from t+1 to t+5. The t+1 refers to the immediate next trading day closing price prediction whereas t+5 denotes fifth future trading day forecasting. The t+1 horizon obtains low MAE because predicting the immediate next time step has greater temporal dependency on recently observed market information. When the horizon increases from t+2 to t+5, the influence of present observations becomes weaker because of unpredictable fluctuations and market volatility.

Table 9. Analysis of the proposed VTG-TLN method across five-step forecasting window from t+1 to t+5 (Reimplemented)

Datasets	Horizons	R^2	RMSE	MAE	MAPE (%)
S&P 500	**t+1**	**0.9915 ± 0.0012**	**29.90 ± 2.29**	**22.65 ± 1.74**	**0.470 ± 0.036**
	t+2	0.9914 ± 0.0013	32.00 ± 2.40	24.24 ± 1.82	0.503 ± 0.038
	t+3	0.9913 ± 0.0013	34.09 ± 2.52	25.83 ± 1.91	0.536 ± 0.040
	t+4	0.9912 ± 0.0014	36.18 ± 2.63	27.41 ± 2.00	0.569 ± 0.041
	t+5	0.9911 ± 0.0014	37.98 ± 2.75	28.77 ± 2.08	0.597 ± 0.043
NASDAQ	**t+1**	**0.9890 ± 0.0012**	**98.89 ± 7.57**	**77.87 ± 5.96**	**0.508 ± 0.039**
	t+2	0.9889 ± 0.0013	105.81 ± 7.95	83.32 ± 6.26	0.543 ± 0.041
	t+3	0.9888 ± 0.0013	112.73 ± 8.33	88.77 ± 6.56	0.579 ± 0.043
	t+4	0.9887 ± 0.0014	119.66 ± 8.71	94.22 ±6.86	0.614 ± 0.045
	t+5	0.9886 ± 0.0014	125.59 ± 9.09	98.89 ± 7.16	0.645 ± 0.047

Tables 10 and 11 shows the analysis of 24 independent runs using random seed range[1-24] for MAPE metric. MAPE computes the prediction error as a percentage relative to actual stock market values, where lower MAPE represents better prediction performance. The variations in MAPE between runs assist to

evaluate the stability of model while smaller errors of VTG-TLN shows reliable predictive performance under repeated runs.

Table 10. Analysis of 24 independent runs using random seed range [1-24] for MAPE (%) metric on S&P 500 dataset

Random seed range	RNN without MEMD	LSTM without MEMD	GRU without MEMD	Transformer without MEMD	VTG-TLN
1	1.054	1.265	0.925	0.550	0.452
2	1.065	1.177	0.968	0.608	0.523
3	1.153	1.237	0.903	0.512	0.442
4	0.987	1.179	0.815	0.460	0.437
5	1.136	0.990	0.860	0.502	0.506
6	0.975	1.233	0.789	0.534	0.411
7	0.987	1.274	0.995	0.565	0.493
8	1.111	1.192	0.828	0.534	0.490
9	1.038	1.275	0.927	0.515	0.481
10	1.052	1.309	0.834	0.513	0.449
11	1.020	1.319	1.005	0.620	0.508
12	1.119	1.334	1.042	0.598	0.509
13	1.153	1.420	1.011	0.537	0.518
14	0.947	1.010	0.895	0.450	0.419
15	1.031	1.168	0.950	0.578	0.423
16	1.139	1.339	0.932	0.601	0.539
17	0.982	1.214	0.971	0.492	0.502
18	1.157	1.220	0.922	0.580	0.487
19	0.996	1.161	0.956	0.548	0.480
20	1.006	1.295	1.021	0.538	0.503
21	1.241	1.319	1.027	0.565	0.509
22	0.981	1.204	0.880	0.581	0.493
23	1.230	1.261	1.022	0.596	0.579
24	1.002	1.245	0.961	0.504	0.487

Table 11. Analysis of 24 independent runs using random seed range [1-24] for MAPE (%) metric on NASDAQ dataset

Random seed range	RNN without MEMD	LSTM without MEMD	GRUwithout MEMD	Transformer without MEMD	VTG-TLN
1	1.085	1.239	0.920	0.521	0.412
2	1.074	1.191	0.976	0.549	0.563
3	1.114	1.550	1.114	0.573	0.591
4	1.185	1.356	1.028	0.707	0.538
5	1.111	1.342	0.877	0.516	0.503
6	1.142	1.105	0.924	0.616	0.517
7	1.162	1.392	1.014	0.558	0.520
8	1.305	1.330	1.099	0.547	0.500
9	1.191	1.273	1.006	0.574	0.529
10	0.911	1.326	0.983	0.527	0.512
11	1.203	1.284	0.941	0.555	0.551
12	1.217	1.376	1.116	0.621	0.590
13	1.232	1.430	1.122	0.648	0.583
14	1.075	1.206	0.933	0.556	0.462
15	1.288	1.478	1.085	0.560	0.550
16	1.182	1.250	0.865	0.648	0.578
17	1.151	1.312	1.006	0.609	0.483
18	1.033	1.276	1.078	0.636	0.488
19	1.090	1.316	1.044	0.661	0.510

20	1.164	1.459	1.028	0.553	0.492
21	1.258	1.447	1.015	0.636	0.560
22	1.198	1.446	1.128	0.595	0.519
23	1.011	1.239	0.945	0.564	0.481
24	1.194	1.359	0.960	0.567	0.511

Table 12 demonstrates the statistical analysis of VTG-TLN against each baseline method over 24 independent runs and random seed range [1-24] using MAPE metric. Cohen's calculates the effect size of the difference in MAPE between each baseline method and proposed VTG-TLN. After multiple comparisons, the Holm threshold denotes the adjusted significance level required for each comparison remains statistically significant whereas Holm p-values determine the significance probability after applying the Holm correction. The p-value from paired t-test is calculated to determine whether the results are statistically significant. The null hypothesis (H0) states there is no significant difference between proposed and each baseline method whereas the alternative hypothesis (H1) represents that a significant difference exists. The obtained p-value is less than 0.05 which shows that H0 is rejected and the results are statistically significant.

Table 12. Statistical analysis of proposed method against each baseline method over 24 independent runs and random seed range [1-24] using MAPE metric

Datasets	Methods (vs VTG-TLN)	Mean difference	SD	t-statistics	p-value from paired t-test	Holm threshold	p-Holm	Cohen's d
S&P 500	RNN without MEMD	0.580	0.070	40.34	7.53×10^{-23}	0.0167	7.53×10^{-23}	8.29
	LSTM without MEMD	0.750	0.090	40.75	5.99×10^{-23}	0.0125	5.99×10^{-23}	8.33
	GRU without MEMD	0.450	0.057	38.39	2.32×10^{-22}	0.0250	2.32×10^{-22}	7.89
	Transformer without MEMD	0.060	0.041	7.12	2.98×10^{-7}	0.0500	2.98×10^{-7}	1.46
NASDAQ	RNN without MEMD	0.626	0.083	36.88	5.75×10^{-22}	0.0167	5.75×10^{-22}	7.54
	LSTM without MEMD	0.810	0.095	41.57	3.81×10^{-23}	0.0125	3.81×10^{-23}	8.53
	GRU without MEMD	0.486	0.075	31.66	1.80×10^{-20}	0.0250	1.80×10^{-20}	6.48
	Transformer without MEMD	0.065	0.051	6.17	3.28×10^{-6}	0.0500	3.28×10^{-6}	1.27

Table 13 presents the analysis of computational complexity for the baseline and proposed method. Since both datasets contain the same features and number of samples, computational complexity remains the same for both datasets. The proposed VTG-TLN obtains a training time of 58.0s because TCN processes temporal patterns through parallel convolutional operations whereas LSTM sequentially processes time

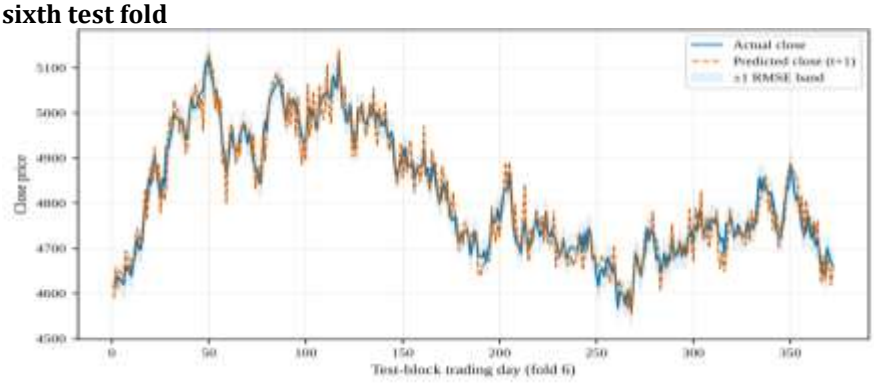
steps by recurrent states. The TCN's parallel computation minimizes processing overhead and enhance training. Also, VTG-TLN employs separate branches for trend and volatility modeling which enables effective learning of temporal patterns.

Table 13. Analysis of computational complexity for live stock market index closing price forecasting on both datasets

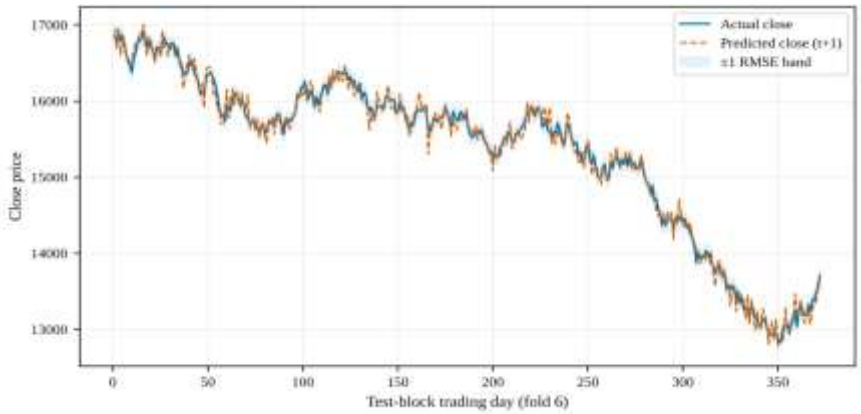
Methods	Training time (s)	Inference time (ms) per sample	Memory consumption (MB)	FLOPs (G)	Parameters (M)
RNN without MEMD	53.3	0.46	0.603	0.005976	0.1508
LSTM without MEMD	71.0	0.61	0.803	0.007968	0.2008
GRU without MEMD	71.0	0.61	0.803	0.005976	0.1508
Transformer without MEMD	12.0	0.10	0.136	0.007968	0.2008
VTG-TLN	**58.0**	**0.47**	**0.375**	**0.003391**	**0.0938**

Figure 3 depicts the actual and predicted closing prices of both datasets over the sixth fold. The predicted curves closely follow the movements of the actual price across declining, rising, and fluctuating market periods which shows that VTG-TLN captures the underlying temporal variations in both datasets. The close alignment results from learning short-term volatility using TCN and long-term trends via LSTM whereas MEMD separates diverse temporal variations before prediction. Around the prediction values, the RMSE bands indicate that the prediction errors remain controlled throughout the test period.

Figure 3. Actual and predicted closing prices for the S&P 500 and NASDAQ datasets during the sixth test fold



S&P 500 dataset



NASDAQ dataset

3.2 Comparative Analysis

Tables 14 and 15 represents the comparative evaluation of existing and proposed method on different datasets. The existing method values are reimplemented based on proposed method settings. Compared with existing methods, VTG-TLN achieves lower MAE of 22.65 and 77.87 due to learning long-term and short-term dependencies separately before fusion which minimize interference between different market behaviors. GAP transforms the feature maps into fixed-length representations which enable effective integration with trend branch. This process enhances the estimation of future closing prices and minimizes prediction errors.

Table 14. Comparative evaluation of existing methods with the proposed method on S&P 500 dataset (Reimplemented)

Methods	R^2	RMSE	MAE	MAPE (%)
ETICA-LSTM [21]	0.9784 ± 0.0029	68.24 ± 4.91	51.72 ± 3.74	1.082 ± 0.078
Galformer [22]	0.9712 ± 0.0033	78.36 ± 5.58	59.34 ± 4.24	1.228 ± 0.088
MCI-GRU [23]	0.9818 ± 0.0023	59.47 ± 4.26	45.08 ± 3.21	0.934 ± 0.067
MA-MSTNet [24]	0.9891 ± 0.0015	34.38 ± 2.68	26.02 ± 2.04	0.541 ± 0.042
MF-DAT [26]	0.9857 ± 0.0019	44.17 ± 3.27	33.46 ± 2.47	0.694 ± 0.052
SVMD-LSTM [28]	0.9878 ± 0.0017	38.51 ± 2.93	29.14 ± 2.23	0.602 ± 0.046
Proposed VTG-TLN	**0.9915 ± 0.0012**	**29.90 ± 2.29**	**22.65 ± 1.74**	**0.470 ± 0.036**

Table 15. Comparative evaluation of existing methods with the proposed method on NASDAQ dataset

Methods	R^2	RMSE	MAE	MAPE (%)
ETICA-LSTM [21]	0.9758 ± 0.0029	223.47 ± 15.96	175.92 ± 12.61	1.151 ± 0.082
MA-MSTNet [24]	0.9794 ± 0.0023	195.28 ± 13.84	153.76 ± 10.89	1.006 ± 0.071
StockGAN++ [25]	0.9832 ± 0.0019	145.36 ± 10.67	114.52 ± 8.41	0.748 ± 0.055
MF-DAT [26]	0.9867 ± 0.0015	113.26 ± 8.77	89.24 ± 6.91	0.581 ± 0.045
Proposed VTG-TLN	**0.9890 ± 0.0012**	**98.89 ± 7.57**	**77.87 ± 5.96**	**0.508 ± 0.039**

3.3 Research Implication

The research implication of VTG-TLN encompass practical decision support for live stock market index closing price forecasting. The framework represent that separating short-term volatility and long-term trend information offer a structured basis for modeling dynamic market behavior. The integration of MEMD, sample-entropy grouping, TCN, LSTM, and candlestick features provides a specialized approach for multi-scale temporal forecasting. The consistent performance across folds and repeated runs denotes that model provide stable predictions. Thus, the research provides a data-driven approach that support reliable analysis of dynamic market behavior and assist stakeholders in obtaining short-horizon closing price forecast.

4. Discussion

The experimental results shows that proposed VTG-TLN method obtains lower MAE of 22.65 and 77.87 on S&P 500 and NASDAQ datasets by modeling short-term volatility and long-term trends through temporal branches. The MEMD method and sample entropy scale clustering separate appropriate market information whereas TCN captures local and rapidly changing price patterns in volatility branch and LSTM conserves long-term dependencies in the trend branch. Moreover, the inclusion of numerical candlestick features enhances the representation of short-term price action characteristics without creating image-based representations. The ablation study shows the contribution of each variants, as removing the temporal architecture, MEMD method, and numerical features leads to high prediction error and lower R^2. As a result, integrating decomposition, feature grouping, temporal modeling,

concatenation ensures VTG-TLN minimizes prediction error and offers reliable live stock market index closing price forecasting

5. Conclusion

This research proposed the VTG-TLN to forecast the live stock market index closing prices using S&P 500 and NASDAQ datasets. In the volatility branch, TCN is used with dilated convolution to capture rapidly changing price movements and short-term temporal dependencies. TCN expands the temporal receptive field across diverse time scales while preserving the chronological order of observations. Moreover, LSTM is used in the trend branch to update and retain relevant historical information through memory mechanisms for capturing long-term dependencies. Based on temporal complexity, sample entropy is employed to group the IMFs into volatility and trend representations whereas numerical candlestick features are included to enhance local price action behavior. Hence, the proposed VTG-TLN achieves lower MAE of 22.65 and 77.87 compared with existing ETICA-LSTM on the S&P 500 and NASDAQ datasets. However, this research does not consider multiple individual stocks; future work will extend the proposed method to learn and predict the behavior of multiple stocks simultaneously.

Data Availability Statement

The datasets generated during and/or analysed during the current study are available in the S&P 500 and NASDAQ datasets [https://finance.yahoo.com/quote/%5EGSPC/history] and [https://finance.yahoo.com/quote/%5EIXIC/history].

References

[1] Muhammad, D., Ahmed, I., Naveed, K. and Bendechache, M., 2024. An explainable deep learning approach for stock market trend prediction. Heliyon, 10(21).

[2] Latif, S., Javaid, N., Aslam, F., Aldegheishem, A., Alrajeh, N. and Bouk, S.H., 2024. Enhanced prediction of stock markets using a novel deep learning model PLSTM-TAL in urbanized smart cities. Heliyon, 10(6).

[3] Yañez, C., Kristjanpoller, W. and Minutolo, M.C., 2024. Stock market index prediction using transformer neural network models and frequency decomposition. Neural Computing and Applications, 36(25), pp.15777-15797.

[4] Zouaghia, Z., Kodia, Z. and Ben Said, L., 2025. Predicting the stock market prices using a machine learning-based framework during crisis periods. Multimedia Tools and Applications, 84(24), pp.28873-28907.

[5] Kumar, C.R. and Manikandan, S., 2025. SEMP-TA: A novel stock market prediction approach based on stacking ensemble machine learning for effective trend analysis. IEEE Access, 13, pp.117479-117490.

[6] Rao, K.V. and Reddy, B.V.R., 2024. OFML-SMF: Optimal feature selection with hybrid machine learning classifier for stock market price forecasting using social media and secondary data sources. Multimedia Tools and Applications, 83(2), pp.4703-4729.

[7] Caldeira, J. and Neves, R., 2026. Dynamic Ensemble of Specialized Models for Multi-Timeframe Stock Market Trend Prediction. Computational Economics, pp.1-34.

[8] Liu, X., Wu, Z. and Xin, J., 2025. Forecasting stock market time series through the integration of bee colony optimizer and multivariate empirical mode decomposition with extreme gradient boosting regression. Engineering Applications of Artificial Intelligence, 148, p.110353.

[9] 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, p.125380.

[10] Sangeetha, J.M. and Alfia, K.J., 2024. Financial stock market forecast using evaluated linear regression based machine learning technique. Measurement: Sensors, 31, p.100950.

[11] Huang, Y. and Yang, C., 2026. Enhancing stock price forecasting with a modular deep learning framework incorporating plug-and-play transformer variants. Expert Systems with Applications, p.131572.

[12] Kuo, R.J. and Chiu, T.H., 2024. Hybrid of jellyfish and particle swarm optimization algorithm-based support vector machine for stock market trend prediction. Applied Soft Computing, 154, p.111394.

[13] Chandar, S.K., 2024. Deep learning framework for stock price prediction using long short-term memory. Soft Computing, 28(17), pp.10557-10567.

[14] Sperlì, G. and Sichinolfi, M.A., 2025. Empowered stock market forecasting using large language model on social media content. Engineering Applications of Artificial Intelligence, 162, p.112727.

[15] Mohamed, Z.E., Refaie Ali, A. and Dabour, W., 2025. Optimizing adaptive neuro-fuzzy inference system model based Chaotic Harris Hawks algorithm for stock prediction. Scientific Reports, 15(1), p.32529.

[16] Shen, Y., Dai, J., Wang, M. and Arce, G.R., 2025. Stock price trend forecasting based on multi-channel complementary network with CEEMDAN decomposition and transformer residual prediction. Expert Systems with Applications, p.130028.
[17] Singh, C., Warraich, J., Thakkar, R. and Vo, N., 2025. SABER: a multimodal sentiment aware regression framework using ROBERTa BiLSTM and deep batch active learning for stock market prediction. International Journal of Information Technology, 17(9), pp.5497-5502.
[18] Teng, X., Zhang, L., Gao, P., Yu, C. and Sun, S., 2025. BERT-Driven stock price trend prediction utilizing tokenized stock data and multi-step optimization approach. Applied Soft Computing, 170, p.112627.
[19] Gkonis, V. and Tsakalos, I., 2025. A hybrid optimized deep learning model via the Golden Jackal Optimizer for accurate stock price forecasting. Expert Systems with Applications, 278, p.127287.
[20] Sun, W., Mei, J., Liu, S., Yuan, C. and Zhao, J., 2025. Research on deep learning model for stock prediction by integrating frequency domain and time series features. Scientific Reports, 15(1), p.30386.
[21] Dioubi, F., Hundera, N.W., Xu, H. and Zhu, X., 2024. Enhancing stock market predictions via hybrid external trend and internal components analysis and long short term memory model. Journal of King Saud University-Computer and Information Sciences, 36(10), p.102252.
[22] Ji, Y., Luo, Y., Lu, A., Xia, D., Yang, L. and Wee-Chung Liew, A., 2024. Galformer: a transformer with generative decoding and a hybrid loss function for multi-step stock market index prediction. Scientific Reports, 14(1), p.23762.
[23] Zhu, P., Li, Y., Hu, Y., Xiang, S., Liu, Q., Cheng, D. and Liang, Y., 2025. MCI-GRU: Stock prediction model based on multi-head cross-attention and improved GRU. Neurocomputing, 638, p.130168.
[24] Wang, X., Zhang, X., Liu, J. and Zuo, Z., 2025. MA-MSTNet: mixed attention-based multi-scale temporal network for stock trend prediction: X. Wang et al. The Journal of Supercomputing, 81(12), p.1207.
[25] Balaji, A., Babu, S.S., Deevi, H.K., Babu, V.D., Latha, V.A. and Srinivasarao, P., 2026. Stock market price prediction with efficient generative artificial intelligence with predictor network. Knowledge and Information Systems, 68(1), p.158.
[26] Huang, K., Li, X., Xiong, N. and Yang, Y., 2024. MF-DAT: a stock trend prediction of the double-graph attention network based on multisource information fusion: K. Huang et al. Multimedia Systems, 30(3), p.136.
[27] Lee, H., Kim, J.H. and Jung, H.S., 2024. Deep-learning-based stock market prediction incorporating ESG sentiment and technical indicators. Scientific Reports, 14(1), p.10262.
[28] Agarwal, S., Sharma, S., Faisal, K.N. and Sharma, R.R., 2025. Time-Series Forecasting Using SVMD-LSTM: A Hybrid Approach for Stock Market Prediction. Journal of Probability and Statistics, 2025(1), p.9464938.
[29] S&P 500 dataset link: https://finance.yahoo.com/quote/%5EGSPC/history (Accessed on August 14 2026)
[30] NASDAQ dataset link: https://finance.yahoo.com/quote/%5EIXIC/history (Accessed on August 14 2026)
[31] Kim, Y.S., Kim, M.K., Fu, N., Liu, J., Wang, J. and Srebric, J., 2025. Investigating the impact of data normalization methods on predicting electricity consumption in a building using different artificial neural network models. Sustainable Cities and Society, 118, p.105570.