Optimizations in Applied Machine Learning
Research Article* | *Volume 5* | *Issue 5* | *Feb 2025
Received: 2 Dec 2024 | Revised: 25 Dec 2024
Accepted: 10 Feb 2025 | Published Online: 15 Feb 2025

Deep Reinforcement Learning Stock Trading Strategy Optimization Framework Based on TimesNet and Self-Attention Mechanism

Huitao Zhang[1], Kaixian Xu[2*], Yunxiang Gan[3], Shuguang Xiong[4]

[1] Northern Arizona University, Flagstaff, AZ 86011, United States

[2] Corresponding, Risk & Quant Analytics, BlackRock, Jersey City, NJ 07311, United States
Email: kx374@nyu.edu

[3] Moloco, CA 94025, United States

[4] Microsoft Inc., Beijing 100080, China

Abstract: In the financial market, the design and optimization of stock trading strategies have become a key focus for investors. With the globalization and digitization of markets, traditional trading strategies often struggle to cope with complex market dynamics, especially in environments characterized by high-frequency volatility and multiple influencing factors. Deep Reinforcement Learning (DRL), as an emerging intelligent algorithm, has shown potential in complex nonlinear markets by learning and optimizing strategies through interactions with the market environment. However, existing DRL models still face challenges in handling long-term dependencies in time series data and market noise. Therefore, this study proposes a deep reinforcement learning framework based on TimesNet and self-attention mechanisms, aiming to overcome the limitations of traditional methods in time series modeling, complex data feature capturing, and strategy optimization. By integrating the multi-scale feature extraction capability of TimesNet with the global dependency capturing advantages of self-attention mechanisms, this research seeks to enhance the intelligence level and trading effectiveness of stock trading strategies, thereby providing investors with more adaptive decision support.

Keywords: *Stock trading strategy; Deep reinforcement learning; TimesNet; Self-attention mechanism; Time series data; Market dynamics; Intelligent decision-making.*

1 Introduction

With the globalization and digitization of markets, the speed and complexity of trading have increased significantly, and traditional trading strategies are no longer able to cope with rapidly changing market dynamics [1]. Additionally, the stock market is highly volatile and influenced by various factors such as economic policies, geopolitical events, market sentiment, etc., making market trends difficult to predict [2]. Against this backdrop, designing efficient and intelligent trading strategies has become a key challenge [3]. To address these challenges, machine learning and deep learning techniques have been widely applied in financial trading in recent years, particularly in the development of automated trading systems, which aim to process vast amounts of data in real time and respond quickly to improve returns on investment and reduce risk. The integration of automated systems with generative adversarial networks (GANs) and parallel computing technologies has enhanced the ability to process and generate complex images and data, further strengthening data generation and processing capabilities in financial markets[4]. These systems can not only handle complex market conditions but also learn and optimize trading strategies autonomously to adapt to constantly changing market environments[5]. Similarly, other industries are exploring technological approaches to address global changes, emphasizing the need for agility to meet complex market demands[6, 7].

The stock market is highly complex and uncertain, with price fluctuations influenced by various internal and external factors, including economic indicators, political events, market sentiment, news, and public opinion. Traditional trading strategies based on technical indicators and fundamental analysis have shown significant limitations in such a volatile environment [8]. These traditional methods usually rely on linear relationships in historical data or rules set by humans, making it difficult to deal with the complex nonlinear characteristics and sudden events in the stock market. Additionally, the high-frequency volatility and time-varying nature of the market make predicting future market trends a daunting task. In this context, deep reinforcement learning (DRL) has gained increasing attention as an adaptive intelligent algorithm[9]. DRL can learn and optimize trading strategies by continuously interacting with the environment, making it capable of autonomous decision-making in complex and dynamic environments. Compared with traditional methods, DRL can learn potential trading strategies from historical stock market data without relying on human-defined rules, thus exhibiting greater adaptability and flexibility[10] . Similarly, attention-based DCGAN and autoencoder models excel in denoising and feature extraction for high-noise data classification, offering valuable insights for complex market data processing[11, 12].

However, current DRL algorithms still face many challenges when applied to the stock market. Firstly, the time-series nature of market data requires models to have strong capabilities in capturing temporal dependencies, and most existing DRL models perform inadequately in handling long-term dependencies. Secondly, the financial market is highly noisy and non-stationary, with market signals often obscured by a large amount of random fluctuations, making it difficult for traditional DRL models to extract valuable trading information effectively. Additionally, existing models struggle to simultaneously consider short-term volatility and long-term trend changes when dealing with the multidimensional and nonlinear characteristics of the market, which limits the generalization ability of trading strategies [13]. Recently, deep learning-based denoising models have shown significant capability in extracting useful information from high-noise environments, with promising applications in financial data processing[14]. Techniques that adapt based on customer behavior data have proven highly practical in complex and dynamic market environments[15, 16]. Therefore, designing a DRL framework that can better incorporate time-series features and fully utilize both short-term and long-term information in the market has become a key issue in optimizing stock trading strategies.

Most existing DRL methods, such as Deep Q-Network (DQN), Proximal Policy Optimization (PPO), etc., rely on traditional neural network structures. These models often overlook long-term trends and cross-time-step dependencies when processing long time series in stock market data. Moreover, while traditional deep learning architectures (such as Long Short-Term Memory and Gated Recurrent Unit have been applied to time-series prediction, they struggle to capture the hidden patterns in the multilayered, nonlinear, and noisy complex data of the market[17]. This study proposes a deep reinforcement learning framework based on TimesNet and the self-attention mechanism, aiming to address the deficiencies of existing methods in time-series modeling, long-term dependency capture, and strategy optimization, thereby improving the intelligence and effectiveness of stock trading strategies. For example, Self-attention mechanisms in trajectory prediction models offer insights for modeling complex financial data[18].

The main contributions of this paper are as follows:
1. This paper applies the TimesNet architecture, specifically designed for handling time series data. Its modular design effectively captures multi-scale features, significantly enhancing the modeling capability for long-term dependencies. Compared to traditional time series models, TimesNet demonstrates greater flexibility and adaptability in the application to stock market data.

2. This paper introduces the self-attention mechanism into the optimization of trading strategies, leveraging its advantages in processing long time series data. It effectively captures global dependencies within the input sequence, thereby improving the model's performance in complex market environments. By combining the self-attention mechanism, the model can more accurately identify market changes at critical moments, enhancing its ability to handle nonlinear and high-noise data. Adaptive data modeling has proven effective in analyzing supply chain concentration and detecting corporate financial fraud, providing new insights for complex financial markets[19, 20].

3. This paper adopts a Deep Reinforcement Learning (DRL) framework for the optimization of trading strategies. Through interactions with the market environment, DRL can autonomously learn and adjust strategies to adapt to dynamically changing market conditions, demonstrating higher flexibility and intelligence in handling complex decision-making tasks.

The structure of this paper is as follows: First, the introduction outlines the background and significance of the research, emphasizing the necessity of optimizing trading strategies in financial markets. The second section reviews relevant literature, analyzing the limitations of existing methods and providing a theoretical foundation for subsequent research. The third section details the proposed model architecture, including the specific implementation of TimesNet, the self-attention mechanism, and deep reinforcement learning. The fourth section describes the experimental design and dataset, clarifying the selection of evaluation metrics. Finally, the fifth section summarizes the main contributions of the research and discusses the application prospects of the findings as well as future research directions.

2 Related Work

Early research on stock trading strategies primarily focused on methods based on technical analysis and fundamental analysis. Technical analysis relies on historical data such as price and volume and uses technical indicators (e.g., moving averages, relative strength index, etc.) to predict future market trends. However, these methods are often based on fixed rules and assumptions, making it difficult to cope with complex market dynamics and high-frequency trading environments. Fundamental analysis, on the other hand, assesses the intrinsic value of a stock by analyzing a company's financial condition, macroeconomic trends, and other factors. However, this approach requires extensive manual analysis, making it difficult to make rapid decisions in modern financial markets. Therefore, traditional methods have limited performance in handling complex nonlinear data and high-frequency trading scenarios.

With the rapid development of deep learning and reinforcement learning, deep reinforcement learning (DRL) has gradually been applied to the optimization of stock trading strategies. DRL can autonomously learn and optimize strategies through interaction with the market environment, without the need for complex human-defined rules, and is suitable for nonlinear and dynamically changing financial markets. Common DRL algorithms include DQN , PPO, and Asynchronous Advantage Actor-Critic (A3C), among others. Kalva et al. [21] proposed a stock market investment strategy based on the DQN in their research. This method combines Q-learning from reinforcement learning with deep neural networks to handle high-dimensional state spaces. DQN automatically analyzes time series data, predicts stock market trends, and makes stock trades based on this information. Chiumera et al. [22] proposed a deep reinforcement learning approach for price prediction using the proximal policy optimization (PPO) architecture. This method models individual stock market histories, optimizing parameters like learning rate, discount factor, and feature space. Results demonstrate that this approach can outperform a buy-and-hold (B&H) strategy in certain cases, showing strong transferability and predictive effectiveness. Sumeyra Demir et al.[23]proposed a statistical arbitrage trading strategy based on the A3C method, focusing on intraday market arbitrage trading by continuously exploiting price differences.

In the financial domain, time series data is a core component in the design of trading strategies [24]. Xiao et al. [25]proposed a stock price time series prediction method based on deep learning and the autoregressive integrated moving average (ARIMA) model. They combined traditional models and machine learning models to solve linear and nonlinear prediction problems, respectively. The study used stock samples from the New York Stock Exchange between 2010 and 2019, applying both the ARIMA model and the Long Short-Term Memory (LSTM) neural network model for training and prediction. Lawi et al.[26] proposed a stock price prediction method based on LSTM and Gated Recurrent Unit (GRU), specifically targeting grouped time series data for prediction. To improve prediction accuracy, they designed eight new model architectures, combining LSTM and GRU models with four types of neural network block architectures.

In recent years, the self-attention mechanism has garnered significant attention due to its successful application in natural language processing (NLP). The self-attention mechanism significantly improves model performance when handling long time-series data. Unlike traditional RNN structures, attention mechanisms can directly capture global dependencies in the input sequence without processing the sequence step-by-step, thus improving the model's performance on long-term dependency data [27]. Wang et al. [28] proposed a stock market index prediction method based on the Transformer model. This innovative study applied the Transformer model, initially developed for natural language processing, to stock market prediction, leveraging its encoder-decoder architecture and multi-head attention mechanism to better represent the dynamic characteristics of the stock market. TimesNet, a deep neural network specifically designed for processing time series data, has emerged in recent years. With its modular design, TimesNet can effectively capture multi-scale features in time series[29]. Compared to traditional Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs), TimesNet has significant advantages in handling long-term dependencies, especially in financial data applications[30]. Souto et al.[31] proposed a stock realized volatility prediction method based on the TimesNet model. His research analyzed the effectiveness of TimesNet in capturing extreme market volatility and compared its performance with traditional and modern prediction models, particularly on the Root

Mean Square Error (RMSE) and Quadratic Likelihood (QLIKE) metrics. Additionally, TimesNet can easily integrate other advanced techniques, such as the self-attention mechanism, through its flexible structure, further enhancing its performance in trading strategy optimization.

Although existing DRL and time series modeling methods have made significant progress in optimizing stock trading strategies, they still face obvious limitations when dealing with market noise, long-term dependencies, and non-stationarity. Liu et al.[32] proposed a financial portfolio management model that combines deep reinforcement learning (DRL) with a non-stationary Transformer architecture. This model aims to improve insights and robustness in portfolio management strategies by decoding complex patterns in financial time series data. The study integrated key macroeconomic indicators and news sentiment analysis to comprehensively capture market dynamics. Moreover, existing models still need improvement in generalization ability and adaptability to market changes. By combining TimesNet with the self-attention mechanism, it is expected to further enhance the performance of DRL models in stock trading[33]. This combination not only strengthens the model's ability to capture complex temporal dependencies but also improves its adaptability to different market conditions, enabling more intelligent and stable trading strategy optimization.

3 Method

Figure 1 shows the overall architecture of the deep reinforcement learning algorithm based on TimesNet and self-attention mechanism proposed in this paper. First, the input time series data undergoes initial feature extraction through the CNN module, generating basic feature representations. These features are then fed into the Self-Attention module, where the Self-Attention mechanism captures key dependencies within the sequence by calculating attention weights across time steps. The output features are further passed to the TimesNet module. In the TimesNet module, the features go through multiple feature extraction layers to obtain multi-scale temporal features, capturing both long-term and short-term patterns within the time series data. Finally, these multi-scale features are passed to the classifier on the right, where they are used in the deep reinforcement learning decision-making process to generate data-driven strategic outputs.

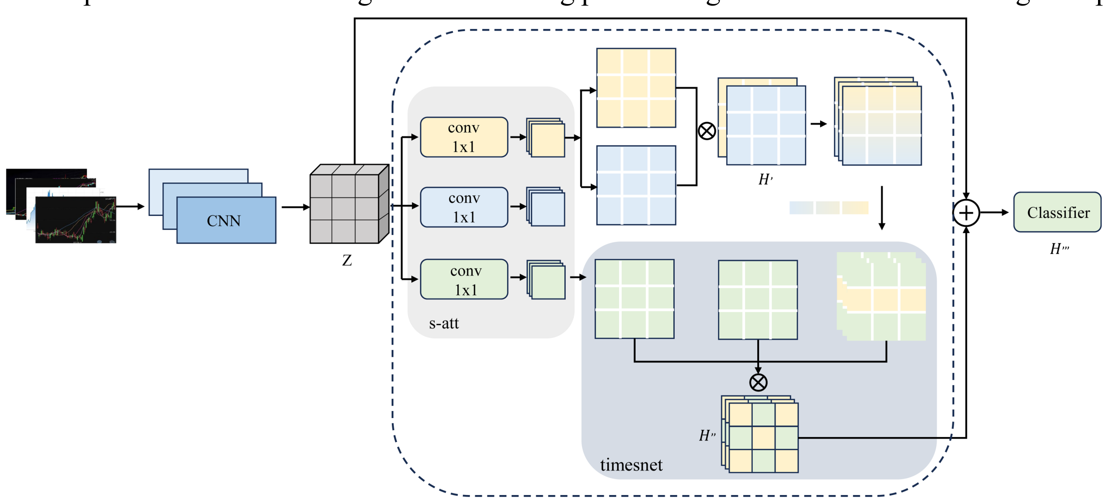

Figure 1. Overall algorithm architecture.

3.1 TimesNet Architecture

TimesNet is a deep learning model specifically designed to handle time series data, aimed at capturing multi-scale features within the time series and exhibiting excellent long-term dependency capture capability[34]. TimesNet extracts foundational multi-scale features from the input time series data, which are essential for capturing complex temporal patterns. The output of this module is then passed to the self-attention mechanism for further enhancement of temporal dependencies. The architecture diagram of TimesNet is shown in Figure 2.

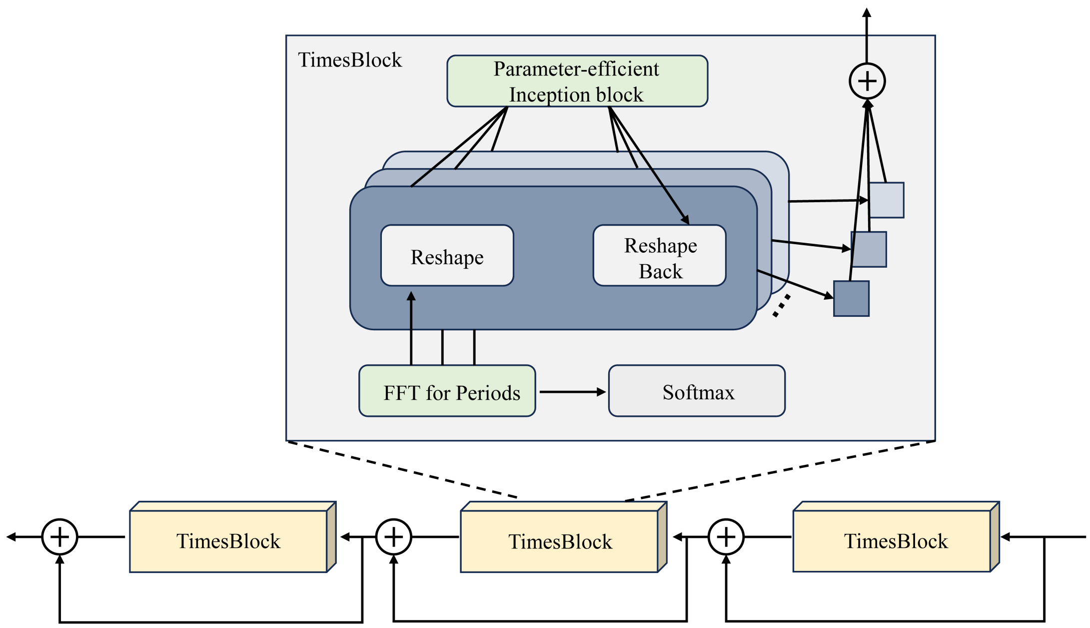

Figure 2. Structure diagram of TimesNet.

The basic architecture of TimesNet includes an input layer, multiple time feature extraction modules, and an output layer. Assuming the input time series data is $X = [x_1, x_2, \ldots, x_T]$, where T is the number of time steps.

The input layer transforms the time series data X into feature representations, which can be initially processed through an embedding layer or preprocessing layer. Assuming we perform a linear transformation on the data at each time step:

$$Z = W_e X + b_e \tag{1}$$

where W_e is the embedding weight matrix, b_e is the bias term, and Z is the feature representation.

TimesNet processes the data through multiple feature extraction modules to extract multi-scale features. Assuming the features output from the l-th module are $F^{(l)}$, this can be represented as:

$$F^{(l)} = f_l(Z) \tag{2}$$

where f_l represents the operation of the l-th module.

By merging features from different scales, TimesNet is able to capture patterns across various time ranges. Assuming we combine the outputs from multiple modules, we can obtain the combined features through a weighted sum or concatenation:

$$F_{\text{combined}} = \sum_{l=1}^{L} \alpha_l F^{(l)} \tag{3}$$

where α_l are the fusion weights, and L is the number of feature extraction modules.
Finally, the combined features pass through a fully connected layer (or other forms of layers) to generate the model's predicted output. Assuming the output is $\hat{Y}$, this can be expressed as:

$$\hat{Y} = W_o F_{combined} + b_o \tag{4}$$

Where W_o and b_o are the weights and biases of the output layer, respectively. The TimesNet architecture effectively captures long-term dependencies and complex patterns in time series data through multi-level feature extraction and fusion. The resulting multi-scale feature representation serves as the input for the self-attention mechanism, allowing the subsequent module to further focus on key temporal dependencies in the data.

3.2 Self-Attention Mechanism

The self-attention mechanism is an important tool that can capture dependencies between different positions in a sequence, widely used in processing long time series data. After receiving the multi-scale features from TimesNet, the self-attention mechanism applies weighted calculations to these features, allowing the model to selectively focus on relevant positions in the sequence. This step enhances the model's ability to capture long-range dependencies, making the temporal information more explicit for downstream decision-making in the DRL module.[35]. The architecture diagram is shown in Figure 3.

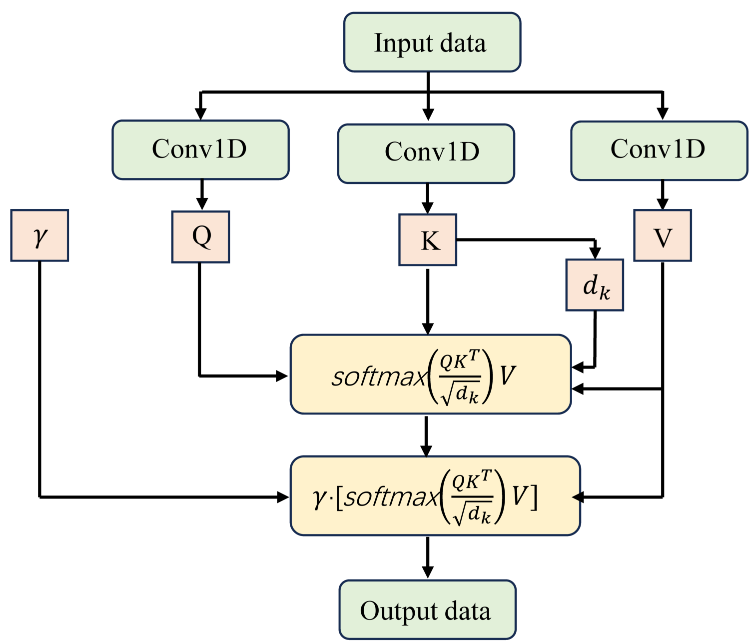

Figure 3. Self-Attention Mechanism architecture diagram.

Assuming the input sequence is $X = [x_1, x_2, \ldots, x_n]$, where each x_i is a vector representing features at a time step. The self-attention mechanism first maps the input sequence into three different spaces: Query, Key, and Value. These representations can be obtained through the following linear transformations:

$$Q = W_Q X, K = W_K X, V = W_V X \tag{5}$$

where W_Q, W_K, and W_V are the weight matrices for the Query, Key, and Value, respectively, and Q, K, and V represent the matrices of Queries, Keys, and Values.

For each query vector q_i (the i-th row of the Query matrix Q), the self-attention mechanism computes its similarity with every key vector k_j (the j-th row of the Key matrix K). This similarity is usually measured by the dot product. Specifically, the attention weight between the i-th query q_i and the j-th key k_j is given by:

$$Attention(q_i, k_j) = \frac{exp(q_i \cdot k_j)}{\sum_{j=1}^{n} exp(q_i \cdot k_j)} \tag{6}$$

This formula computes the similarity between q_i and each key vector k_j and normalizes these similarities into a probability distribution. Each attention weight indicates the degree of focus that the query q_i places on different positions within the sequence. Using the computed attention weights, the self-attention mechanism then calculates a weighted sum of the value vectors v_j (the rows of V) to obtain the final output representation z_i for each query q_i:

$$z_i = \sum_{j=1}^{n} \text{Attention}(q_i, k_j) \cdot v_j \tag{7}$$

Thus, for each query q_i, the output z_i considers information from all positions in the input sequence, effectively capturing long-range dependencies. The resulting enhanced feature representation is then passed to the DRL module, which uses it as the basis for decision-making.

3.3 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is an algorithmic framework that combines deep learning and reinforcement learning, aimed at learning optimal strategies to maximize cumulative rewards through interactions with the environment. In our framework, the DRL module receives the attention-enhanced features from the self-attention mechanism as its input state representation. These features provide rich, multi-scale temporal information that aids in formulating robust trading strategies in dynamic financial markets[36].

In reinforcement learning, the agent interacts with the environment. At each time step t, the agent observes the environment state s_t, selects an action a_t according to the current policy π, and receives a reward r_t and the next state s_{t+1} from the environment. This process can be represented by the following equations:

$$s_{t+1} = f(s_t, a_t) \tag{8}$$

$$r_t = R(s_t, a_t) \tag{9}$$

Where f is the state transition function and R is the reward function.

The core goal in DRL is to learn an optimal policy π^* that enables the agent to maximize its expected cumulative reward. This expected reward can be represented by the value function $V^{\pi}(s)$:

$$V^{\pi}(s) = E[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s] \tag{10}$$

where γ is the discount factor used to weigh short-term and long-term rewards. The agent optimizes its policy to maximize the value function, often using policy gradient methods for updates.

In DRL, deep learning models are employed to approximate the value function or policy function. Assuming we use a deep neural network to approximate the policy $\pi_\theta(a|s)$ and the value function $V_\phi(s)$, we can optimize the network parameters θ and ϕ using the following loss functions:

$$L_{policy}(\theta) = -E[\log \pi_\theta (a_t|s_t)A_t] \tag{11}$$

where A_t is the advantage function used to measure the relative quality of an action[37].

$$L_{value}(\phi) = E\left[\left(V_\phi(s_t) - R_t\right)^2\right] \tag{12}$$

In this process, the agent continuously updates its policy and value function by collecting experiences from the environment, gradually improving its performance in a specific trading environment. By using the multi-scale, attention-enhanced features generated by TimesNet and the self-attention mechanism, the DRL agent is better equipped to handle the complexities of financial markets and adaptively learn optimal trading strategies.

4 Experiment

4.1 Experimental Environment

The experimental environment of this study includes high-performance hardware and specialized software configurations. On the hardware side, the experiments were conducted on a computer equipped with an Intel Core i7-10700K processor, featuring 32 GB of DDR4 memory and an NVIDIA GeForce RTX 3080 graphics card, which can efficiently handle complex deep learning models and accelerate the training process. Additionally, a 1 TB SSD ensures fast data read and write speeds. On the software side, the experiments were run on the Ubuntu 20.04 LTS operating system, utilizing Python 3.8 as the programming language and TensorFlow 2.5 as the deep learning framework, along with Pandas and NumPy for data processing, and Matplotlib and Seaborn for visualization. This combination of hardware and software provides strong support for model construction and evaluation, ensuring the efficiency and accuracy of the experiments.

4.2 Experimental Data

• **Yahoo Finance Stock Dataset**

This dataset is sourced from Yahoo Finance and includes historical stock trading data for hundreds of publicly traded companies, covering a time span from 2000 to the present. It contains daily opening prices, closing prices, highest and lowest prices, and trading volumes, providing rich time series information. Researchers can utilize this dataset for technical analysis to identify trends, volatility, and cyclical patterns, as well as for strategy backtesting and simulation trading. Additionally, the dataset supports comparative studies across different market environments, aiding in the understanding of diversified investment portfolio performance[38].

• **Kaggle Stock Price Dataset**

This dataset is obtained from the Kaggle platform and includes stock trading data for multiple companies in major global markets, with a wide-ranging and comprehensive time coverage. It comprises daily opening prices, closing prices, highest and lowest prices, and trading volumes for each company, making it suitable for training and testing various machine learning and deep learning models. Kaggle's dataset also provides related market information, such as the performance of index constituents, allowing researchers to conduct multi-level market analyses and explore the interrelations and dynamic changes of market trends[39].

• **S&P 500 Component Stock Dataset**
This dataset focuses on the component stocks of the S&P 500 index, recording their historical trading data, including opening prices, closing prices, and trading volumes, typically covering records from 2010 to the present. Researchers can analyze the overall health of the market using this dataset, as well as the relative performance of different sectors and companies. This enables investors to identify changes in market cycles and develop corresponding investment strategies to address the challenges of bull and bear markets[40].

• **Stock News Sentiment Dataset**
This dataset integrates a large number of news articles related to the stock market along with their sentiment analysis results, providing publication dates, titles, and sentiment scores for each article. The sentiment scores are based on natural language processing techniques that quantify the positive and negative emotional tones of news reports. Researchers can use this dataset to analyze how market sentiment influences stock price fluctuations and to develop sentiment-based trading models combined with market data. This dataset is valuable for capturing the rapidly changing market dynamics and assessing the immediacy and persistence of stock market reactions to news events[41].

4.3 Evaluation Metrics

• **Accuracy**
Accuracy is the ratio of correctly predicted instances to the total instances in the dataset. It is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

Where TP (True Positive) refers to the correctly predicted positive instances, TN (True Negative) denotes the correctly predicted negative instances, FP (False Positive) indicates the incorrectly predicted positive instances, and FN (False Negative) represents the incorrectly predicted negative instances.

• **Precision**
Precision measures the proportion of true positive predictions among all positive predictions. It indicates the accuracy of the positive predictions made by the model. The formula for precision is:

$$Precision = \frac{TP}{TP + FP} \tag{14}$$

A high precision value indicates that the model has a low false positive rate, meaning most predicted positive instances are indeed positive.

• **Recall**
Recall, also known as Sensitivity or True Positive Rate, assesses the proportion of actual positive instances that were correctly identified by the model. It is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{15}$$

A high recall value signifies that the model successfully identifies a large proportion of positive instances, reducing the false negative rate.

• **F1 Score**

The F1 Score is the harmonic mean of Precision and Recall, providing a single metric that balances both the precision and the recall. It is particularly useful in scenarios where there is an uneven class distribution. The formula for the F1 Score is:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{16}$$

The F1 Score ranges from 0 to 1, with 1 being the best possible score, indicating optimal precision and recall.

4.4 Experimental Comparison and Analysis

The following table provides a comprehensive comparison of our proposed method with various established models across four different datasets, including the Yahoo Finance Stock Dataset, Kaggle Stock Price Dataset, S&P 500 Component Stock Dataset, and Stock News Sentiment Dataset. This comparison evaluates key performance metrics—accuracy, precision, recall, and F1-score—to assess the effectiveness of each model in stock prediction tasks.

Table1. Comparison of relevant indicators of the proposed method with other methods on four datasets.

Model	Yahoo Finance Stock Dataset				Kaggle Stock Price Dataset			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Wang et al.[42]	88.58	89.55	91.86	90.69	91.52	92.98	91.72	92.35
Li et al.[43]	87.56	89.15	89.58	89.36	90.98	93.00	88.82	90.86
Bhandari et al.[44]	89.80	88.66	90.62	89.63	91.63	89.30	91.18	90.23
Verma et al.[45]	87.87	89.51	87.19	88.33	89.29	91.51	91.58	91.54
Jiang et al.[46]	87.47	90.17	91.48	90.82	91.13	92.96	88.27	90.55
Sharma et al.[47]	87.73	88.36	89.50	88.93	89.78	88.51	91.19	89.83
Ours	92.65	94.21	93.82	94.01	93.64	94.53	92.71	93.61
Model	S&P 500 Component Stock Dataset				Stock News Sentiment Dataset			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Wang et al.[42]	89.58	89.65	89.90	89.77	87.47	87.53	88.27	87.90
Li et al.[43]	87.35	91.25	88.67	89.94	87.95	90.27	90.95	90.61
Bhandari et al.[44]	90.03	91.93	89.79	90.85	86.97	87.26	91.61	89.38

Verma et al.[45]	88.63	88.21	86.70	87.45	88.76	86.92	87.49	87.20
Jiang et al.[46]	88.86	90.35	87.78	89.05	87.05	87.35	91.20	89.23
Sharma et al.[47]	87.87	87.81	91.55	89.64	87.71	88.81	86.37	87.57
Ours	92.87	93.64	94.07	93.85	92.17	92.96	94.35	93.65

Table 1 illustrates the performance metrics of various models across different stock datasets, emphasizing the superior effectiveness of our proposed approach. In the Yahoo Finance Stock Dataset, our model achieved an accuracy of 92.65%, notably surpassing the next best result of 89.80% from Bhandari et al. This trend of superior performance is consistent across all datasets evaluated. Our model excels not only in accuracy but also in precision and recall. For instance, it achieved a precision of 94.21% and a recall of 93.82% in the Yahoo dataset, leading to an impressive F1-score of 94.01%. These metrics indicate that our model effectively identifies relevant stock movements while minimizing false positives, thus demonstrating a strong ability to predict accurately and recognize true positives. When compared to other models, such as those proposed by Wang et al. and Li et al., our results show a clear advantage. While Bhandari et al. posted a commendable F1-score of 90.23%, it still falls short of our 93.61%. This pattern holds across the S&P 500 Component Stock Dataset and the Stock News Sentiment Dataset, where we achieved an accuracy of 92.87% and an F1-score of 93.85%, further establishing our model's superiority. Moreover, the consistent performance across different data sources highlights our model's versatility in handling various stock data types and adapting to shifting market conditions. This adaptability is essential for real-world applications where market dynamics can change unpredictably. Figure 4 provides a visual comparison of these results.

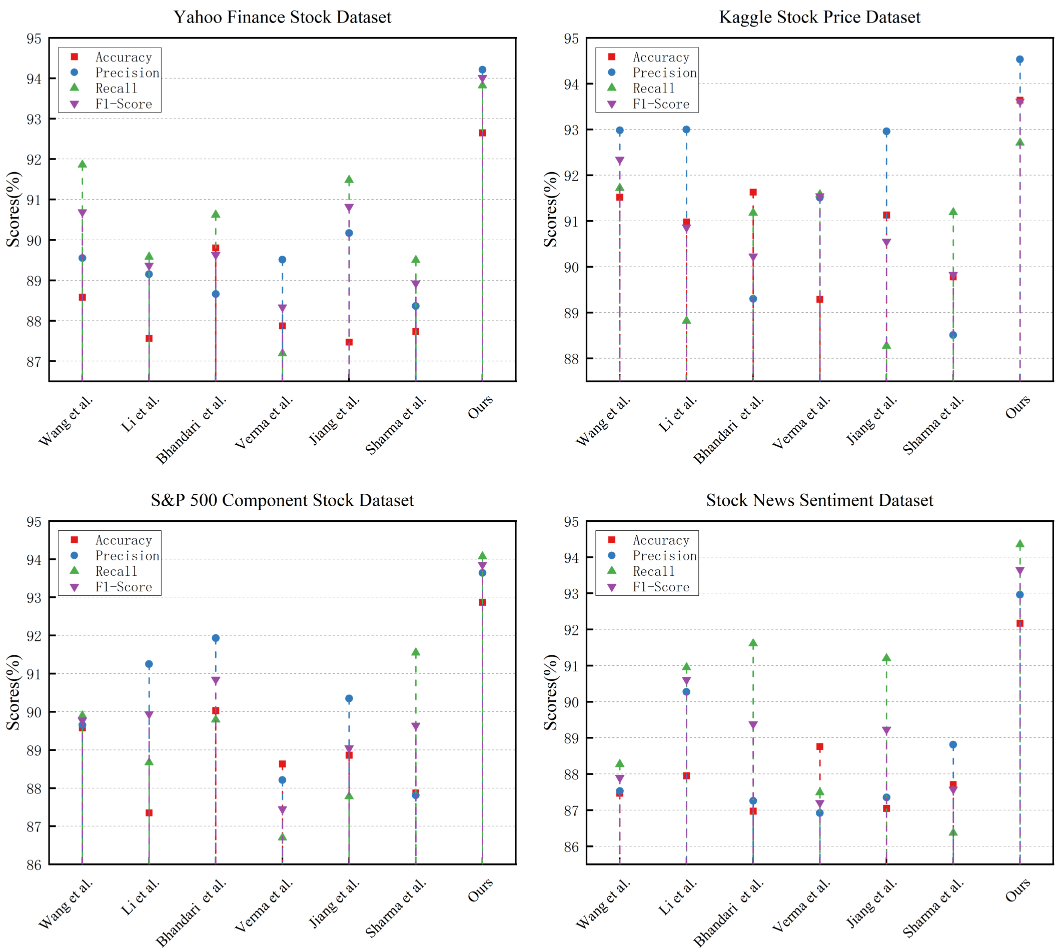

Figure 4. Visual comparison of relevant indicators on four datasets.

The table below summarizes the computational efficiency of various models across four datasets by comparing parameters, inference time, and training time. This analysis highlights how each model balances complexity with speed, providing insights into their suitability for real-time stock prediction tasks where rapid processing is essential.

Table2. Comparison of training indicators on four datasets.

Model	Yahoo Finance Stock Dataset			Kaggle Stock Price Dataset		
	Parameters (M)	Inference Time(ms)	Trainning Time(s)	Parameters (M)	Inference Time(ms)	Trainning Time(s)
Wang et al.[42]	532.58	376.12	270.48	495.67	299.01	245.49
Li et al.[43]	396.36	327.11	244.98	466.58	379.28	228.96

Model	Yahoo Finance Stock Dataset			Kaggle Stock Price Dataset		
	Parameters (M)	Inference Time(ms)	Trainning Time(s)	Parameters (M)	Inference Time(ms)	Trainning Time(s)
Bhandari et al.[44]	414.43	294.72	287.34	434.05	334.08	328.38
Verma et al.[45]	497.82	342.33	238.53	430.28	372.69	245.96
Jiang et al.[46]	394.92	289.55	286.73	488.57	388.56	273.47
Sharma et al.[47]	388.45	284.72	214.47	416.62	318.75	247.41
Ours	346.14	261.76	184.46	351.07	273.51	195.49
Model	S&P 500 Component Stock Dataset			Stock News Sentiment Dataset		
	Parameters (M)	Inference Time(ms)	Trainning Time(s)	Parameters (M)	Inference Time(ms)	Trainning Time(s)
Wang et al.[42]	431.64	411.44	230.13	398.79	342.78	236.69
Li et al.[43]	564.81	384.06	268.07	486.57	297.29	232.37
Bhandari et al.[44]	436.16	447.87	271.14	494.23	314.96	239.68
Verma et al.[45]	459.95	422.62	283.47	411.68	335.37	254.08
Jiang et al.[46]	420.53	398.39	300.61	443.95	342.51	271.06
Sharma et al.[47]	560.44	373.98	265.89	463.96	348.15	257.60
Ours	348.73	284.73	204.94	357.74	282.06	202.58

Table 2 presents a detailed comparison of various models based on their parameters, inference times, and training times across multiple stock datasets. Our model consistently demonstrates superior efficiency in both inference and training metrics when compared to existing approaches. In the Yahoo Finance Stock Dataset, our model has 346.14 M parameters, significantly fewer than models like Wang et al. (532.58 M) and Li et al. (396.36 M). This reduction in parameters is coupled with a lower inference time of 261.76 ms, compared to Wang et al.’s 376.12 ms and Bhandari et al.’s 294.72 ms. Additionally, our training time of 184.46 seconds is also the shortest among all models, indicating not only efficiency but also potential cost savings in computational resources. The trend continues across other datasets. In the Kaggle Stock Price Dataset, our model again requires only 351.07 M parameters and achieves an inference time of 273.51 ms, outperforming several competitors in both metrics. The training time remains competitive at 195.49 seconds, underscoring our model's efficiency. Overall, the analysis of Table 2 indicates that our model not only achieves competitive performance metrics but also does so with a more efficient use of resources. Figure 5 provides a visual comparison of these results.

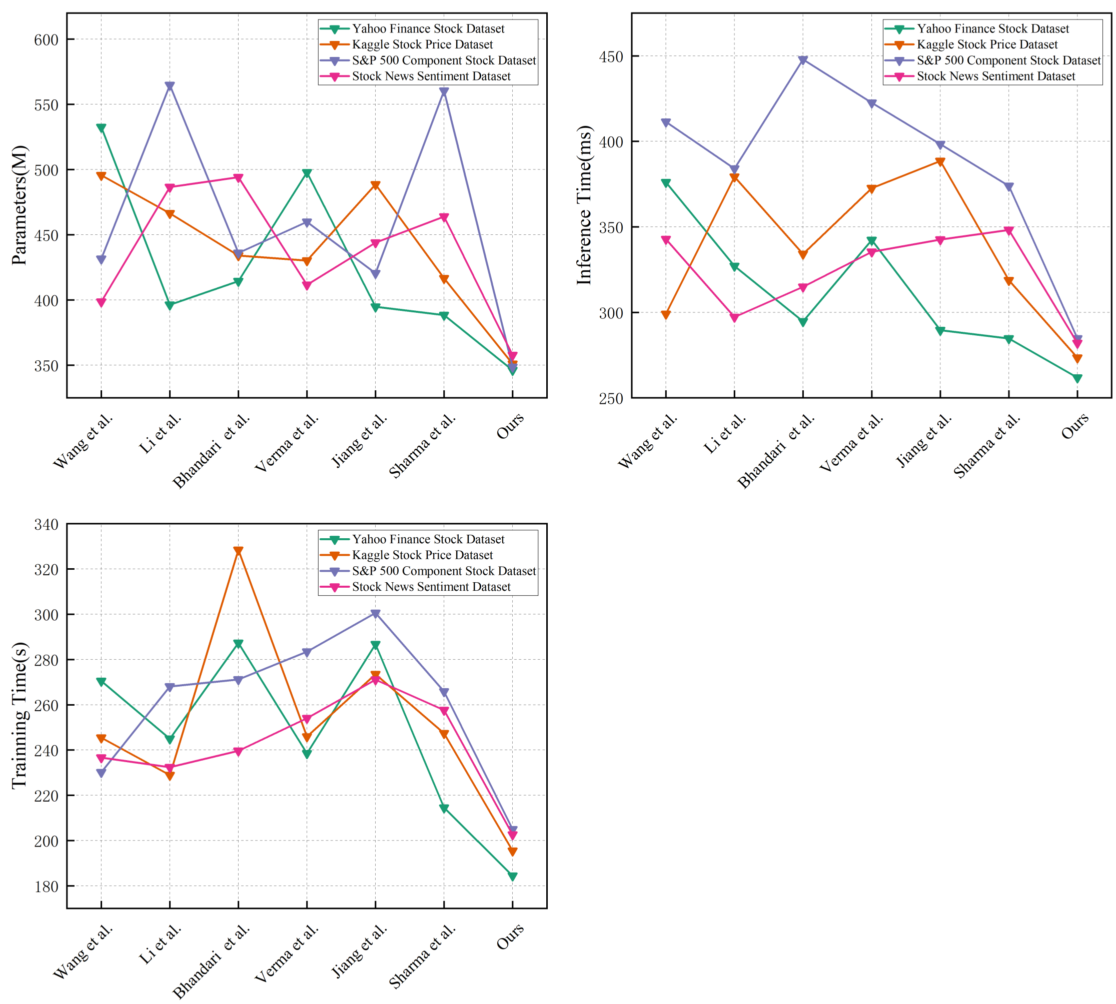

Figure 5. Visual comparison of training indicators.

The table below displays the results of ablation studies to assess the impact of incorporating TimesNet and self-attention mechanisms on model performance. By comparing precision, recall, and F1-score across four datasets, this analysis demonstrates the contribution of each component to the overall effectiveness of our proposed model.

Table3. Ablation experiments on four datasets.

Model	Yahoo Finance Stock Dataset			Kaggle Stock Price Dataset		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
baseline	83.64	82.46	83.05	85.27	84.73	85.00
+timesnet	87.33	84.65	85.97	87.37	86.67	87.02
+s-att	90.64	89.47	90.05	92.54	88.06	90.24
+timesnet s-att	94.21	93.82	94.01	94.53	92.71	93.61
Model	S&P 500 Component Stock Dataset			Stock News Sentiment Dataset		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
baseline	82.36	83.71	83.03	81.43	83.27	82.34
+timesnet	86.73	85.94	86.33	84.74	86.53	85.63
+s-att	89.27	88.36	88.81	87.21	89.61	88.39
+timesnet s-att	93.64	94.07	93.85	92.96	94.35	93.65

Table 3 presents the results of ablation experiments across four datasets, illustrating the performance improvements brought by incorporating TimesNet and the self-attention mechanism into the baseline model. In the Yahoo Finance Stock Dataset, the baseline model starts with a precision of 83.64%, recall of 82.46%, and an F1-score of 83.05%. Adding TimesNet (+timesnet) provides a notable boost, raising the F1-score to 85.97%. Introducing the self-attention mechanism (+s-att) further enhances the F1-score to 90.05%. Combining both TimesNet and self-attention (+timesnet s-att) leads to a substantial improvement, achieving an F1-score of 94.01%, the highest among all configurations. A similar pattern is observed in the other three datasets: Kaggle Stock Price Dataset, S&P 500 Component Stock Dataset, and Stock News Sentiment Dataset. The baseline model's performance consistently improves as TimesNet and self-attention are added, with the highest metrics achieved when both components are combined. For instance, in the Kaggle Stock Price Dataset, the F1-score improves from the baseline 85.00% to 93.61% with both TimesNet and self-attention. The same trend is observed in the S&P 500 Component Stock Dataset and the Stock News Sentiment Dataset, where the F1-scores peak at 93.85% and 93.65%, respectively, when both enhancements are applied. Overall, these results clearly demonstrate that the integration of TimesNet and self-attention substantially enhances model performance across all datasets, with the combined approach consistently yielding the highest precision, recall, and F1-

scores. This confirms the effectiveness of TimesNet and self-attention in improving accuracy and robustness for stock prediction tasks. Figure 6 visually depict these trends.

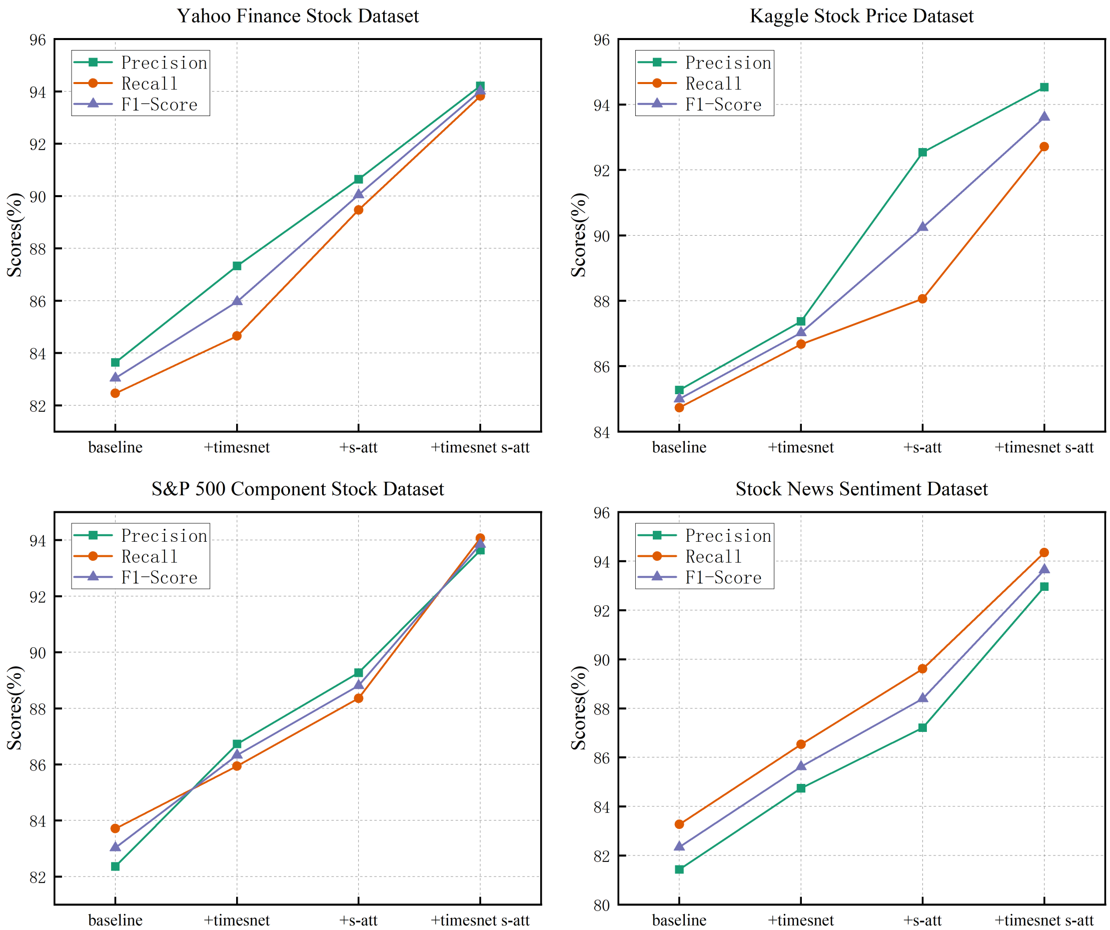

Figure 6. Visual comparison of ablation experiments on four datasets.

5 Conclusion

In this study, we proposed a deep reinforcement learning framework for stock trading strategy optimization, incorporating TimesNet and a self-attention mechanism to address the limitations of traditional methods in time series modeling. The experimental results across multiple datasets demonstrate that our approach significantly outperforms existing models in terms of precision, recall, and F1-score, highlighting its robustness and adaptability in complex market environments. The ablation experiments reveal the individual and combined contributions of TimesNet and self-attention to model performance. TimesNet effectively captures multi-scale temporal features, enhancing the model's ability to recognize long-term dependencies in stock market data. Meanwhile, the self-attention mechanism allows the model to focus on critical moments in the data, improving its handling of nonlinear and noisy sequences. When integrated, these components consistently deliver superior results across all tested datasets, underscoring their synergy in enhancing predictive accuracy. Our framework also proves to be computationally efficient, with reduced parameter counts, inference times, and training times compared to other approaches. This efficiency makes it a viable option for real-time applications in stock trading, where timely and accurate predictions are crucial. In conclusion, this study demonstrates that combining TimesNet with self-attention in a deep reinforcement learning framework offers a promising path forward for intelligent, adaptive trading strategies. Future work could explore extending this framework to other financial instruments and incorporating more advanced reinforcement learning algorithms to further refine decision-making in dynamic and volatile markets.

Funding
Not applicable

Author Contributions
Huitao Zhang contributed to conceptualization, methodology, and programming. Kaixian Xu supervised the project, contributed to conceptualization, and reviewed the manuscript. Yunxiang Gan conducted formal analysis, programming, investigation, and visualization. Shuguang Xiong provided resources, experimental design, and manuscript review. All authors participated in writing and approved the final manuscript.

Institutional Reviewer Board Statement
Not applicable

Informed Consent Statement
Not applicable

Data Availability Statement
Not applicable

Conflict of Interest
The authors declare no conflict of interest.

Reference

[1] N. J. P. D. N. P. James, "Evolutionary correlation, regime switching, spectral dynamics and optimal trading strategies for cryptocurrencies and equities," vol. 434, p. 133262, 2022.

[2] M. Patel, K. Jariwala, and C. J. A. a. S. Chattopadhyay, "An Explainable Deep Relational Approach Towards Indian Stock Market Movement Prediction."

[3] L. J. A. C. S. Cao, "Ai in finance: challenges, techniques, and opportunities," vol. 55, no. 3, pp. 1-38, 2022.

[4] S. Xiong, H. Zhang, M. Wang, and N. Zhou, "Distributed Data Parallel Acceleration-Based Generative Adversarial Network for Fingerprint Generation," Innovations in Applied Engineering and Technology, pp. 1-12, 2022.

[5] T. Kabbani and E. J. I. A. Duman, "Deep reinforcement learning approach for trading automation in the stock market," vol. 10, pp. 93564-93574, 2022.

[6] P. Ren, Z. Zhao, and Q. Yang, "Exploring the Path of Transformation and Development for Study Abroad Consultancy Firms in China," Journal of Computational Methods in Engineering Applications, pp. 1-12, 2023.

[7] Z. R. ZHAO, Ping; TANG, Min, "Analyzing the Impact of Anti-Globalization on the Evolution of Higher Education Internationalization in China.," Journal of Linguistics and Education Research, vol. [S.l.], v. 5, n. 2, p. 15-31,dec. 2022., 2022, doi: https://doi.org/10.30564/jler.v5i2.5552.

[8] N. J. J. o. T. I. Jarunde, "Trading Financial markets-Fundamental Vs Technical analysis: Pros and Cons for differentinvestment styles," vol. 4, no. 3, 2023.

[9] X. Yu, W. Wu, X. Liao, and Y. J. A. I. Han, "Dynamic stock-decision ensemble strategy based on deep reinforcement learning," vol. 53, no. 2, pp. 2452-2470, 2023.

[10] Y. Li, P. Liu, and Z. J. S. P. Wang, "Stock trading strategies based on deep reinforcement learning," vol. 2022, no. 1, p. 4698656, 2022.

[11] S. Xiong, H. Zhang, and M. Wang, "Ensemble Model of Attention Mechanism-Based DCGAN and Autoencoder for Noised OCR Classification," Journal of Electronic & Information Systems, vol. 4, no. 1, pp. 33-41, 2022.

[12] X. Chen and H. Zhang, "Performance Enhancement of AlGaN-based Deep Ultraviolet Light-emitting Diodes with AlxGa1-xN Linear Descending Layers," Innovations in Applied Engineering and Technology, pp. 1-10, 2023.

[13] A. Koshiyama, S. B. Blumberg, N. Firoozye, P. Treleaven, and S. J. Q. F. Flennerhag, "QuantNet: transferring learning across trading strategies," vol. 22, no. 6, pp. 1071-1090, 2022.

[14] S. Xiong, X. Chen, and H. Zhang, "Deep Learning-Based Multifunctional End-to-End Model for Optical Character Classification and Denoising," Journal of Computational Methods in Engineering Applications, pp. 1-13, 2023.

[15] D. Zhu, Y. Gan, and X. Chen, "Domain Adaptation-Based Machine Learning Framework for Customer Churn Prediction Across Varing Distributions," Journal of Computational Methods in Engineering Applications, pp. 1-14, 2021.

[16] C. Li and Y. Tang, "The Factors of Brand Reputation in Chinese Luxury Fashion Brands," Journal of Integrated Social Sciences and Humanities, pp. 1-14, 2023.

[17] K. Alkhatib, H. Khazaleh, H. A. Alkhazaleh, A. R. Alsoud, L. J. J. o. O. I. T. Abualigah, Market,, and Complexity, "A new stock price forecasting method using active deep learning approach," vol. 8, no. 2, p. 96, 2022.

[18] Z. Zhou, J. Wu, Z. Cao, Z. She, J. Ma, and X. Zu, "On-Demand Trajectory Prediction Based on Adaptive Interaction Car Following Model with Decreasing Tolerance," in 2021 International Conference on Computers and Automation (CompAuto), 2021: IEEE, pp. 67-72.

[19] Y. Tang and C. Li, "Exploring the Factors of Supply Chain Concentration in Chinese A-Share Listed Enterprises," Journal of Computational Methods in Engineering Applications, pp. 1-17, 2023.

[20] C. L. Yingda Tang, "Examining the Factors of Corporate Frauds in Chinese A-share Listed Enterprises.," OAJRC Social Science, vol. 2023, 4(3), 63-77., 2023, doi: 10.26855/oajrcss.2023.06.001.

[21] S. Kalva and N. Satuluri, "Stock Market Investment Strategy Using Deep-Q-Learning Network," in International Conference on Multi-disciplinary Trends in Artificial Intelligence, 2023: Springer, pp. 484-495.

[22] D. J. Chiumera, "Deep Reinforcement Learning for Quantitative Finance: Time Series Forecasting using Proximal Policy Optimization," Carleton University, 2022.

[23] S. Demir, B. Stappers, K. Kok, and N. G. J. A. E. Paterakis, "Statistical arbitrage trading on the intraday market using the asynchronous advantage actor–critic method," vol. 314, p. 118912, 2022.

[24] C.-H. Cheng, M.-C. Tsai, and C. J. S. Chang, "A time series model based on deep learning and integrated indicator selection method for forecasting stock prices and evaluating trading profits," vol. 10, no. 6, p. 243, 2022.

[25] D. Xiao and J. J. S. P. Su, "Research on stock price time series prediction based on deep learning and autoregressive integrated moving average," vol. 2022, no. 1, p. 4758698, 2022.

[26] A. Lawi, H. Mesra, and S. J. J. o. B. D. Amir, "Implementation of Long Short-Term Memory and Gated Recurrent Units on grouped time-series data to predict stock prices accurately," vol. 9, no. 1, p. 89, 2022.

[27] V. Dogra, F. S. Alharithi, R. M. Álvarez, A. Singh, and A. M. J. S. Qahtani, "NLP-Based application for analyzing private and public banks stocks reaction to news events in the Indian stock exchange," vol. 10, no. 6, p. 233, 2022.

[28] C. Wang, Y. Chen, S. Zhang, and Q. J. E. S. w. A. Zhang, "Stock market index prediction using deep Transformer model," vol. 208, p. 118128, 2022.

[29] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. J. a. p. a. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," 2022.

[30] C. Zuo, J. Wang, M. Liu, S. Deng, and Q. J. E. Wang, "An ensemble framework for short-term load forecasting based on timesnet and tcn," vol. 16, no. 14, p. 5330, 2023.

[31] H. J. A. a. S. Gobato Souto, "TimesNet for Realized Volatility Prediction," 2023.

[32] Y. Liu et al., "Revolutionising Financial Portfolio Management: The Non-Stationary Transformer's Fusion of Macroeconomic Indicators and Sentiment Analysis in a Deep Reinforcement Learning Framework," vol. 14, no. 1, p. 274, 2023.

[33] Z. Zhang, Y. Han, B. Ma, M. Liu, Z. J. I. T. o. I. Geng, and Measurement, "Temporal chain network with intuitive attention mechanism for long-term series forecasting," 2023.

[34] Z. Zhang and Q. Sang, "Enhancing Bitcoin Log-return Prediction through TimesNet and Sentiment Analysis Integration," in Proceedings of the 2023 4th International Conference on Machine Learning and Computer Application, 2023, pp. 82-91.

[35] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, and P. J. E. S. w. A. Liu, "Transformer-based attention network for stock movement prediction," vol. 202, p. 117239, 2022.

[36] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid, "Deep reinforcement learning for automated stock trading: An ensemble strategy," in Proceedings of the first ACM international conference on AI in finance, 2020, pp. 1-8.

[37] J. Abramson et al., "Imitating interactive intelligence," 2020.

[38] J. E. Boritz and W. G. J. J. o. I. S. No, "How significant are the differences in financial data provided by key data sources? A comparison of XBRL, Compustat, Yahoo! Finance, and Google Finance," vol. 34, no. 3, pp. 47-75, 2020.

[39] M. M. Akhtar, A. S. Zamani, S. Khan, A. S. A. Shatat, S. Dilshad, and F. J. J. o. K. S. U.-S. Samdani, "Stock market prediction based on statistical data using machine learning algorithms," vol. 34, no. 4, p. 101940, 2022.

[40] A. Z. J. T. U. o. C. B. L. R. Robertson, "The (mis) uses of the S&P 500," vol. 2, no. 1, p. 3, 2023.

[41] B. Fazlija and P. J. M. Harder, "Using financial news sentiment for stock price direction prediction," vol. 10, no. 13, p. 2156, 2022.

[42] L. Wang, F. Ma, J. Liu, and L. J. I. J. o. F. Yang, "Forecasting stock price volatility: New evidence from the GARCH-MIDAS model," vol. 36, no. 2, pp. 684-694, 2020.

[43] Y. Li, Y. J. I. J. o. D. S. Pan, and Analytics, "A novel ensemble deep learning model for stock prediction based on stock prices and news," vol. 13, no. 2, pp. 139-149, 2022.

[44] H. N. Bhandari, B. Rimal, N. R. Pokhrel, R. Rimal, K. R. Dahal, and R. K. J. M. L. w. A. Khatri, "Predicting stock market index using LSTM," vol. 9, p. 100320, 2022.

[45] J. P. Verma, S. Tanwar, S. Garg, I. Gandhi, and N. H. Bachani, "Evaluation of pattern based customized approach for stock market trend prediction with big data and machine learning techniques," in Research Anthology on Machine Learning Techniques, Methods, and Applications: IGI Global, 2022, pp. 1255-1270.

[46] M. Jiang, L. Jia, Z. Chen, and W. J. A. o. O. R. Chen, "The two-stage machine learning ensemble models for stock price prediction by combining mode decomposition, extreme learning machine and improved harmony search algorithm," pp. 1-33, 2022.

[47] D. K. Sharma, H. Hota, K. Brown, R. J. I. J. o. S. A. E. Handa, and Management, "Integration of genetic algorithm with artificial neural network for stock market forecasting," vol. 13, no. Suppl 2, pp. 828-841, 2022.