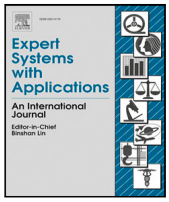
Expert Systems With Applications 240 (2024) 122581

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



A novel deep reinforcement learning framework with BiLSTM-Attention networks for algorithmic trading

Yuling Huang [a], Xiaoxiao Wan [b], Lin Zhang [c], Xiaoping Lu [a,*]

[a] School of Computer Science and Engineering, Macau University of Science and Technology, Taipa, Macao Special Administrative Region of China
[b] Faculty of Finance, City University of Macau, Taipa, Macao Special Administrative Region of China
[c] School of Accounting and Finance, Beijing Institute of Technology, Zhuhai, China

ARTICLE INFO

Keywords:
Deep reinforcement learning
Bidirectional long short-term memory
Attention mechanism
Algorithmic trading

ABSTRACT

The financial market, as a complex nonlinear dynamic system frequently influenced by various factors, such as international investment capital, is very challenging to build trading strategies from the obtained market information. Deep Reinforcement Learning (DRL) combines the perceptual capability of deep learning and the control decision making capability of reinforcement learning to learn the mapping between financial market states and trading decisions by interacting with the environment. In this paper, an enhanced stock trading strategy, denominated efficient deep State-Action-Reward-State-Action (SARSA), is presented to tackle the algorithmic trading problem of determining optimal trading positions in the daily trading activities of the stock market. This algorithm is recognized for its properties of stable learning and convergence, attributes of critical significance within the financial domain, where stability assumes paramount importance, and excessive risk-taking should be averted. Furthermore, a novel deep network architecture called Bidirectional Long Short-Term Memory (BiLSTM)-Attention is introduced to address the challenge of accurately presenting the complex and volatile stock market. The BiLSTM-Attention greatly enhances the network's capacity to recognize key features and patterns in stock market data, allowing agents to focus on the most relevant aspects of the data. Evaluations on DJI, SP500, GE, IXIC datasets from January 1, 2008 to December 31, 2022 show that our efficient deep SARSA algorithm outperforms a wide range of traditional strategies (B&H, S&H, MR, TF) and DRL-based strategies (TDQN, DQN-Vanilla). For example, on the IXIC dataset, the efficient deep SARSA strategy achieves an attractive Cumulative Return (CR) of 582.17% and a Sharpe Ratio (ShR) of 1.86, outperforming all other methods. These experimental results prove the performance of our method in enhancing stock trading strategies.

1. Introduction

Algorithmic trading is a research area that has received a lot of attention from researchers. The agent's goal is to take opportunity of market fluctuations by repeating buy and sell orders to optimize the return on the capital investment. However, the intricate and volatile nature of the stock market presents serious challenges, and these complexities are well-known to the financial industry. With the rapid development of machine learning, the automatic identification of trading opportunities in financial asset markets has become an imperative pursuit. How to automatically select trading strategies in complex and dynamic financial markets is an important research direction in modern finance. In the field of artificial intelligence, the two main research directions of trading strategies are prediction-based methods and reinforcement learning-based methods.

In the field of trading strategy research, deep learning has gained prominence as a means of predicting price movements and subsequently developing trading strategies based on these predictions. However, the generation of trading strategies based on price predictions often relies on human intuition or theoretical constructs that emphasize prediction accuracy rather than capital returns. Thus, prediction results are just an indicator that higher prediction accuracy does not necessarily lead to higher returns.

However, after AlphaGo defeated the top human Go player, the deep reinforcement learning algorithm has received more and more attention and is widely used in other fields (Silver et al., 2016). The same situation is gradually applied to the study of trading strategies. In the context of reinforcement learning-based trading strategies, two

* Corresponding author.
E-mail addresses: 2109853gia30003@student.must.edu.mo (Y. Huang), f22092100330@cityu.mo (X. Wan), 08110@bitzh.edu.cn (L. Zhang), xplu@must.edu.mo (X. Lu).

https://doi.org/10.1016/j.eswa.2023.122581
Received 18 July 2022; Received in revised form 12 November 2023; Accepted 12 November 2023
Available online 14 November 2023

primary methodologies have emerged. The first centers on estimating the state–action value function, wherein the agent learns the action values and selects actions based on these estimate. The second revolves around policy gradients, wherein the agent learns policy parameters by optimizing a performance metric $J(\theta)$ via gradient ascent. Furthermore, the approximation of the policy and value function learned by the agent is often referred to as an "Actor-Critic" approach, where "Actor" refers to the learned policy and "Critic" refers to the learned value function, i.e., usually the state value function. By utilizing historical asset prices as input data, the researchers designed customized neural networks to enhance trading behavior with the ultimate goal of maximizing cumulative returns or improving SHARPE RATIO (SHR). The efficacy of Deep Reinforcement Learning (DRL) has been demonstrated in various aspects of finance, including optimal execution, portfolio optimization, and market making, with notable success (Ge et al., 2022; Khare et al., 2023; Liu et al., 2023; Luo & Duan, 2023; Tran et al., 2023; Xiao, 2023; Ye & Schuller, 2023).

Inspired by the previous literature review, this paper proposes an innovative DRL model, called efficient deep State-Action-Reward-State-Action (SARSA), for developing single-asset trading strategies. SARSA is an on-policy reinforcement learning algorithm, meaning it updates its policy while following the current policy. SARSA's remarkable feature includes its tendency to exhibit stable learning and convergence properties, a crucial attribute in the realm of financial markets where stability is paramount. This cautious approach mitigates the adverse consequences associated with erratic or overly aggressive trading strategies, thereby preventing excessive risk-taking or unwarranted adherence to sub-optimal trading methods. Moreover, due to the intricate and ever-evolving nature of stock markets, accurately representing the market's condition within the framework of reinforcement learning poses a significant challenge. To address this challenge, an innovative deep network architecture is introduced, where Bidirectional Long Short-Term Memory (BiLSTM) and an Attention Mechanism (AM) are integrated. This fusion significantly enhances the network's capacity to identified key features and patterns, allowing the agent to direct its attention towards the most salient aspects within the intricate landscape of stock market data. In addition, we pragmatically select a set of 33 technical indicators as states to address this challenge and mitigate computational complexity. This choice both simplifies the learning process and avoids the complexity related to modeling the state of the stock market, ultimately improving efficiency.

In summary, our study makes the following major contributions:

- Proposing an innovative DRL model called Efficient Deep SARSA for the development of single-asset trading strategies. SARSA, an on-policy reinforcement learning algorithm, forms the foundation of this model, known for its stable learning and convergence properties, which are critical in the financial domain to ensure stability and avoid excessive risk-taking.
- Introducing a novel deep network architecture named BiLSTM-Attention to tackle the challenge of accurately representing the complex and ever-changing nature of stock markets. The network greatly enhances the ability of the network to recognize key features and patterns in stock market data, allowing agents to focus on the most relevant aspects of the data.
- Evaluations on four datasets show that the proposed efficient deep SARSA algorithm performs well when compared to multiple classical strategies (B&H, S&H, MR, and TF) and DRL-based strategies (TDQN, DQN-Vanilla).

The remainder of this paper is arranged as follows. In Section 2, a comprehensive review of existing research in feature extraction networks for financial time series and value-based deep reinforcement learning in algorithmic trading. Section 3 depicts the detailed methods of the proposed method, namely efficient deep SARSA. Section 4 shows the experimental details including the comparison of efficient deep SARSA method and baseline models. Finally, in Section 6, conclusions are presented, the limitations of this study are discussed, and directions for future research are outlined.

2. Related work

In order to discuss more clearly the similarities and differences in the literature covered by the proposed approach, we review two branches of the literature: feature extraction networks for financial time series and value-based deep reinforcement learning in algorithmic trading. By dividing the related work into different branches, a focused and organized presentation of the different areas of research related to the topic can be presented.

2.1. *Feature extraction networks for financial time series*

Financial markets have intricate patterns and dynamic behaviors. With the rapid development of deep learning techniques, researchers are increasingly interested in building diverse feature extraction networks to extract relevant features or patterns in historical time series data. These identified features are key inputs to predictive models, including Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Multi-Layer Perception (MLP). This work is critical to the decision-making process involved in investment strategy, risk management and the development of algorithmic trading methods.

Several notable contributions in the literature have explored the application of deep learning techniques for forecasting stock prices. Roondiwala et al. (2017), Shah et al. (2021), Sunny et al. (2020) and Zhang et al. (2021) delved into the use of LSTM and BiLSTM neural networks, achieving optimal predictive performance by capturing temporal information evolution. Lu et al. (2020) introduced the CNN-BiLSTM-AM method, comprising CNN, BiLSTM, and AM, to predict the closing price of stocks for the following day. Wu, Li et al. (2021) presented the Stock Array Convolutional LSTM for Stock Sequences (SACLSTM) framework, a novel fusion of CNN and LSTM, designed to enhance the accuracy of stock price predictions. Huang et al. (2021) extended their prediction model by incorporating Variational Mode Decomposition (VMD) and LSTM, introducing an input generation step for forecasting future data points. Moreover, Sridhar and Sanagavarapu (2021) proposed a time series prediction model integrating news headline polarity and BiLSTM to forecast stock prices. Juairiah et al. (2022) utilized LSTM, BiLSTM, Autoregressive Integrating Moving Average (ARIMA), Hidden Markov Model (HMM), and multiple AM to predict Microsoft stock volatility over a span of several decades. Kumar and Anitha (2022) introduced a hybrid deep model consisting of CNN, BiLSTM, and AM to automatically identify essential features for constructing an optimal stock price prediction model. Yu (2022) developed a forecasting pattern based on BiLSTM-Gate Recurrent Unit (GRU) to predict stock price indices, such as the SPI. Mo and Jing (2022) conducted a comparative study involving BiLSTM, LSTM, Support Vector Regression (SVR), and differential ARIMA models for financial time series forecasting. Qiao et al. (2022) presented a stock market prediction model based on LSTM, with a focus on predicting stock returns using data from the Shanghai and Shenzhen stock markets spanning from 2019 to 2021. Md et al. (2022) proposed a stock price prediction method centered on the multilayer sequential long short-term memory (MLSLSTM) model. Sharaf et al. (2022) combined LSTM with sentiment analysis techniques to develop a hybrid system specifically tailored for predicting stock movements and trends during the COVID-19 pandemic. Xu et al. (2022) introduced an LSTM-CNN hybrid model designed for predicting stock price trends, successfully applying it to forecast stock price volatility in the Chinese market. Bansal et al. (2022) presented a predictive model built upon the BiLSTM architecture, utilizing the Seq2Seq framework to forecast stock closing prices. The model's performance was evaluated through a comparative analysis against established algorithms such as K-Nearest Neighbor, Decision Tree, and Linear Regression. The results of this comparison showcased a significant reduction in squared error, underscoring the heightened predictive prowess of the proposed BiLSTM-based approach. Pholsri and Kantavat (2023) explored stock time series feature extraction using BiLSTM and

CNN architectures, while Shi et al. (2023) introduced a novel two-stage approach integrating Graph Convolutional Networks (GCN) and LSTM, cooperatively employed to predict stock price movements.

These academic contributions represent only a small fraction of the current research efforts aimed at refining feature extraction networks for financial time series analysis. They collectively demonstrate the growing promise of deep learning for applications in the field of financial forecasting and trading strategy development.

2.2. Value-based deep reinforcement learning in algorithmic trading

The utilization of machine learning algorithms in the domain of financial trading has garnered substantial attention in recent years, representing a prominent avenue for the automation and enhancement of trading processes. Machine learning algorithms offer traders a more adaptive, data-centric, and impartial approach to financial trading, ultimately facilitating improvements in returns and risk management (Kim et al., 2017). These algorithms possess the capability to unveil intricate underlying patterns that often elude human traders' discernment. The successful application of DRL in gaming has prompted a surge of interest in deploying DRL techniques to formulate trading strategies for financial assets, with the aim of supplanting human expertise, intuition, and experience.

Within this burgeoning field, the strategies implemented by various authors can be categorized primarily as value-based Reinforcement Learning (RL) (Bajpai, 2021; Brim & Flann, 2022; Carta, 2021; Chakole & Kurhekar, 2020; Cheng et al., 2021; Corazza et al., 2019; Cornalba et al., 2022; Dang, 2020; de Oliveira et al., 2020; Huang et al., 2023; Khare et al., 2023; Li et al., 2022, 2020; Liu et al., 2023; Luo & Duan, 2023; Ma et al., 2021; Nan et al., 2022; Pavel et al., 2021; Shi et al., 2021; Taghian et al., 2022; Théate & Ernst, 2021; Tran et al., 2023; Ye & Schuller, 2023; Zhou & Tang, 2021), policy-based RL (Lele et al., 2020; Liu, Li et al., 2021; Mahayana et al., 2022; Wang et al., 2021; Xiao, 2023), or actor-critic RL (Ge et al., 2022; Lima Paiva et al., 2021; Liu, Liu et al., 2020; Liu, Yang et al., 2020; Ponomarev et al., 2019; Vishal et al., 2021). It is worth noting that the DRL algorithm primarily prefers a value-based approach when developing trading strategies for a single stock. Table 1 offers a comprehensive overview of the diverse modeling choices encompassing datasets, state and action spaces, reward functions, methodologies, and performance evaluation metrics, as observed across the spectrum of articles pertaining to value-based RL. It should be acknowledged that this tabulation, while comprehensive, does not encompass the entirety of the available literature. For example, Corazza et al. (2019) conducted a comparative analysis involving SARSA, Q-Learning, and Greedy-GQ algorithms in the context of daily trading within the Italian stock market.

Recent advancements have seen the amalgamation of multiple reinforcement learning algorithms to tackle more generalized algorithmic trading challenges. For instance, Zhang et al. (2020) harnessed three reinforcement learning algorithms, namely Deep Q-Network (DQN), Policy Gradients, and Advantage Actor-Critic, to devise trading strategies tailored for continuous futures contracts. These strategies included adjustments to trading positions contingent upon market volatility. Yang et al. (2020) introduced a novel ensemble strategy for autonomously generating transaction behaviors. This approach leveraged Proximal Policy Optimization, Advantage Actor Critic (A2C), and Deep Deterministic Policy Gradient algorithms, amalgamating their strengths to adapt effectively to diverse market scenarios, thereby optimizing returns. Li et al. (2022) combined CNN and LSTM networks to analyze stock data and candlestick charts. Their work integrated Double Deep Q-Network (DDQN) and Dueling DQN algorithms, enabling the acquisition of optimal dynamic trading strategies.

Inspired by the previous literature review, this paper proposes an innovative DRL model, called efficient deep SARSA, for developing single-asset trading strategies. SARSA is an on-policy reinforcement learning algorithm, meaning it updates its policy while following the current policy. SARSA's remarkable feature includes its tendency to exhibit stable learning and convergence properties, a crucial attribute in the realm of financial markets where stability is paramount. This cautious approach mitigates the adverse consequences associated with erratic or overly aggressive trading strategies, thereby preventing excessive risk-taking or unwarranted adherence to suboptimal trading methods. Moreover, to enhance the model's performance further, An innovative deep network architecture is introduced which combines BiLSTM with an attention mechanism. This fusion empowers the agent to focus its attention on key features and patterns within stock market data. Given the complex nature of stock market dynamics, accurately representing the market state within the RL framework presents a formidable challenge.

3. Methodology

Reinforcement learning is much more focused on goal-directed learning from interaction than are other approaches to machine learning. Reinforcement learning is concerned with the sequential interaction of an agent with its environment. First of all, the agent observes the environment of state, then the agent executes the action resulting from its policy and receives a reward as a result of its action. Thus, the reinforcement learning approach is concerned with designing policies that maximize a criterion that depends on the immediate rewards observed over a certain time horizon. Driven by the progress of valued-based RL algorithm, the efficient deep SARSA method was proposed to generate stock trading strategies. The valued-based RL algorithm has been widely utilized in optimizing trading strategies using stock states as inputs.

3.1. Problem definition

In this section, we define the algorithmic trading sequential decision-making problem as Markov Decision Processes (MDPs) of RL. MDPs is a mathematical framework for modeling decision-making problems in which agents interact with their environment to achieve specific goals. It is widely applied in several fields, such as finance and economics, to analyze and optimize decision-making processes. Specifically, MDPs can be used to understand and improve decision-making strategies in financial markets.

3.1.1. State space

The state space represents the set of all possible states that the trading environment can be in at any given time. In the context of single-asset trading, the state space can include various features or indicators that describe the current market conditions. In general, different types of information such as historical price movements, trading volumes, financial statements, can be used as the current state. In this paper, the closing price of the previous n days and TA are used as the state at time t. Therefore, the size of the environmental state at the moment of t is $m * n$, where $m = 33$ represents close price and 32 different TA (Details are shown in Table 3) calculated from the price and volume, and $n = 10$ represents the window length. It can be represented as Eq. (1).

$$s_t = [I_{t-n+1}, I_{t-n+2}, \dots, I_t], I_i = [I_i^1, I_i^2, \dots, I_i^m], \tag{1}$$

where I_i^j is the jth normalized feature in ith day.

3.1.2. Action space

The action space defines the set of all possible actions that the trading agent can take within the trading environment. At each time step t, the RL agent first acquires the environment state s_t. Based on the RL policy $\pi(a_t|s_t)$, the agent chooses an action a_t. In this paper, we consider a strategy that assumes buying and selling a single security in a discrete action space in financial markets, i.e. the strategy involves phases of opening, holding and closing positions. Therefore, the final

Table 1
Value-based DRL algorithms for algorithmic trading.

Article	Dataset	State space	Action space	Reward	Method	Performance
Chakole and Kurhekar (2020)	2 U.S indices, 2 Indian indices	Close price	Long, short, neutral	return	DQN	Return, MDD, ShR, Skewness, Annualized Return (AR), kurtosis etc.
Li et al. (2020)	10 U.S. stocks	Close price	Buy, sell, hold	none	Dueling DQN, DDQN	Profit
Dang (2020)	Google	Price	Buy, sell, hold	Profit	Dueling DQN, DQN, DDQN + CNN	Profit
de Oliveira et al. (2020)	Simulate stock	Price	Buy, sell, hold	Profit	SARSA	Profit
Zhou and Tang (2021)	Tapai Group	Close price, TA, sentiment	Buy, sell, hold	Maximum profit	DRQN + LSTM	Cumulative Return (CR)
Pavel et al. (2021)	NASDAQ stocks	Price	Buy, sell, hold	Customized	DRQN + ARIMA	Profit, reward, cost
Ma et al. (2021)	Chinese stocks	OHLCV, TA	Buy, sell, hold	Return	DDQN + DNN, LSTM	Returns, ShR
Shi et al. (2021)	Chinese stocks, U.S. stocks	OHLCV, action	Long/buy, short/sell	Return	DDQN + CNN	Accuracy, profit, return, MDD
Théate and Ernst (2021)	30 stocks	OHLCV, position	Buy, sell	Return	DDQN + CNN	Profit, ShR, MDD, MDDD, AR, SoR etc.
Carta (2021)	S&P 500, JPM, MSFT	OHLC	Long, short, neutral	Return	DDQN + CNN	Precision, MDD, RoMaD, COV
Bajpai (2021)	Indian market	Prices, volume	Buy, sell, hold	none	DQN, DDQN, Dueling DDQN + CNN	Profit
Cheng et al. (2021)	Taiwan stocks	OHLCV	Various position size	Return	DQN + LSTM, RNN, DNN	CR, MDD, return over MDD
Li et al. (2022)	S&P 500, Chinese stocks	TA, candlestick charts	Long, short, neural	ShR + PR	Dueling DQN, DDQN + CNN, BiLSTM	AR, ShR, MDD
Taghian et al. (2022)	AAPL, GOOGL, KSS, BTC/USD	OHLC	Buy, sell, hold	Customized	SARSA, DQN + MLP, CNN, GRU	AR, TR, VaR, ShR, MDD
Brim and Flann (2022)	S&P 500 index	OHLC	Long, short, neutral	Returns $a \times r \times N$	DDQN + CNN	Profit, ShR, MDD
Nan et al. (2022)	MSFT, AMZN, TSLA, others	Cash, open price, sentiment etc.	Buy, sell, hold	customized	DQN + RNN	ShR
Cornalba et al. (2022)	3 U.S stocks, 3 virtual currency	Position, returns	Buy, sell, hold	LR, ALR, ShR, POWC	DQN-HER + MLP	AmR, profit, ShR
Liu et al. (2023)	Chinese stock, S&P 500 stocks	OHLCV, TA, candlestick charts	Buy, sell, hold	profit	Dueling DQN, DDQN + CNN, LSTM, BiLSTM	Profit, AR, ShR, MDD, MDDD
Tran et al. (2023)	BTC-USDT	OHLC	Buy, sell, hold	SR	Dueling DQN, DDQN + CNN	Average return
Huang et al. (2023)	Dow Jones, NASDAQ, AAPLE, General Electric	OHLCV	Buy, sell, hold	SR	SARSA + CNN	Profit, AR, ShR
Ye and Schuller (2023)	10 stocks (AAPL etc.)	Close price	Buy, sell, hold	Profit	Multi-step DQN + LSTM	Return, CR
Luo and Duan (2023)	Google, Apple, Tesla, Meta, IBM, Microsoft	Balance, price, shares	Buy, sell, hold	Profit	Q-learning, SARSA, policy gradient	AR, profit
Khare et al. (2023)	SP 500	Daily return	Buy, sell	Profit	Value iteration, SARSA, Q-learning	Profit
Our	DJI, SP500, IXIC, GE	Close price, TA	Buy, sell, hold	SR	SARSA + BiLSTM + AM	CR, AR, ShR

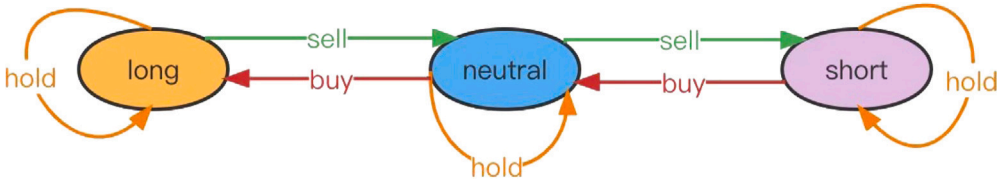


Fig. 1. The transfer of positions.

Table 2
The actual trading operation based on the signal and the account position.

Position (POS_t)	Signal (a_t^*)	Actual operation (a_t)	Description
0	0	0	Hold the cash.
0	1	1	Open a long position.
0	−1	−1	Open a short position.
1	0	0	Hold the long position.
1	1	0	Hold the long position.
1	−1	−1	Close the long position.
−1	0	0	Hold the short position.
−1	1	1	Close the short position.
−1	−1	0	Hold the short position.

action and updated position information are computed as Eqs. (2) to (3). The state–action value is referred to as the Q^*-value and represents the expected cumulative reward, expressed as Eq. (14).

$$a_t^* = \begin{cases} -1, & \text{if } Q^*(s,a) = Q_t^{\text{Sell}}; \\ 0, & \text{if } Q^*(s,a) = Q_t^{\text{Hold}}; \\ 1, & \text{if } Q^*(s,a) = Q_t^{\text{Buy}}. \end{cases} \tag{2}$$

where Q_t^{Sell}, Q_t^{Hold}, and Q_t^{Buy} is decided by the Q-network at each time step t.

$$a_t = a_t^* \cdot \left(a_t^* * \text{POS}_{t-1}\right), \tag{3}$$

where $a_t = \{-1, 0, 1\} = \{\text{sell, hold, buy}\}$, $\text{POS}_t \in \{\text{short, neutral, long}\} = \{-1, 0, 1\}$.

Therefore, considering the position information POS_t, the signal a_t^*, and the actual action a_t at time t according to the state s_t, is shown as Table 2.

According to the above definition, different actions are allowed to be performed at different stages of the trading. In the case of no position, it is possible to execute the action to buy (sell) a long (short) or to keep no position; in the case of long (short) position, it is possible to execute the action to sell (buy) to close the position or to keep long (short) position, i.e. the validity of the trading action depends on the position status. Therefore, the trading system obtains valid trading behavior based on signals and account positions, as shown in Fig. 1.

3.1.3. Reward function

The reward function quantifies the immediate feedback or reward that the trading agent receives after taking an action in a particular state. In algorithmic trading, the reward function is often defined to reflect the profitability or performance of the trading strategy. It can take various forms, such as profit and loss (P&L), risk-adjusted return. We chose risk-adjusted return, which is a composite measure that can consider both return and risk, such as the ShR (Sharpe & William, 1994) first proposed by Nobel laureate William Sharpe. In this paper, we design the short-term ShR as the reward function, which reflects information about the next K days, while also focusing on information about the agent's position. It can be expressed mathematically as Eq. (4).

$$SSR_t = POS_t * SR_t, \tag{4}$$

where

$$SR_t = \frac{\text{mean}(R_t^k)}{\text{std}(R_t^k)}, \tag{5}$$

$$R_t^k = [\frac{p_{t+1} - p_t}{p_t}, \frac{p_{t+2} - p_t}{p_t}, \dots, \frac{p_{t+k} - p_t}{p_t}] \tag{6}$$

where p_t is the close price at time step t.

3.2. Methodology

3.2.1. System overview

In the intricate financial market, it is a difficult challenge for traders to pursue higher profits and reduce risks at the same time. To address this challenge, this study has inspired from the value-based RL algorithm and finally introduced an innovative method called efficient deep SARSA. The efficient deep SARSA is an extension of the SARSA algorithm, which address the complexity inherent in trading. Traditionally, value-based reinforcement learning agents have required large amounts of training data to obtain optimal trading strategies and maximize cumulative rewards. However, the field of algorithmic trading have unique problems, including a potentially unstable training process, such as noise, stochastic on stock data. Given these challenges, our main goal is to achieve stable training while efficiently exploring optimal strategies within the constraints of limited data resources. Fig. 2 illustrates the structural configuration of the efficient deep SARSA, summarizing the core of our proposed method.

For episode $t = \{1, 2, \dots, T\}$, the efficient deep SARSA will generate a sequence of transactions with a reward of $\{r_1, r_2, \dots, r_t, \dots, r_T\}$. Meanwhile, the discounted return G_t, representing the expected return of efficient deep SARSA, is defined as Eq. (7).

$$G_t = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \cdots + \gamma^{T-t} \cdot r_T, \tag{7}$$

where the parameter γ is the discount factor ($\gamma \in [0, 1]$).

Therefore, the expected value of $G_t(s_t)$ is estimated by observing trading, defined as Eq. (8).

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}[G_t | S_t = s, A_t = a] \\ &= \mathbb{E}[r_t + \gamma \cdot (r_{t+1} + \gamma r_{t+2} + \cdots) | S_t = s, A_t = a] \\ &= \mathbb{E}[r_t + \gamma \cdot G_{t+1} | S_t = s, A_t = a] \\ &= \mathbb{E}[r_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a], \end{aligned} \tag{8}$$

The expected return for taking action a in the current state s is denoted by $Q(s,a)$. To update the current state–action value function, the next state–action value was adopted for estimation. Here, an on-policy value-based method, SARSA, was employed to derive the optimal state–action value function (Sutton & Barto, 2018). Therefore, Eq. (9) defines the updated equation for the state–action value.

$$\begin{aligned} Q_\pi(s_t, a_t) \leftarrow & Q_\pi(s_t, a_t) - \alpha[r_t + \\ & \gamma Q_\pi(s_{t+1}, a_{t+1}) - Q_\pi(s_t, a_t)], \end{aligned} \tag{9}$$

It can be seen that the computation of the temporal-difference value involves the utilization of the current state–action value and the next state–action value. To perform the update step, it is therefore necessary to know the next action controlled by our policy. Consequently, a quintuple $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$ is sequentially generated during each update process, forming a training sample that is retained in the historical data D. However, the existence of multiple potential states of the stock market makes it necessary to transfer the task from updating the Q values in the table to function approximations.

Table 3
Description of exploited TA.

Feature name	Description	Feature name	Description
Cl_t	Closing Price	ADOSC	Chaikin A/D Oscillator
DEMA	Double Exponential Moving Average	OBV	On Balance Volume
SMA_3	Simple Moving Average(3 days)	NATR	Normalized Average True Range
SMA_5	Simple Moving Average(5 days)	ATR	Average True Range
SMA_10	Simple Moving Average(10 days)	TRANGE	True Range
BBANDS_UP	Bollinger BandsDEMA(UP)	BETA	Beta
BBANDS_MIDDLE	Bollinger BandsDEMA(MIDDLE)	TSF	Time Series Forecast
BBANDS_LOW	Bollinger BandsDEMA(LOW)	ADXR	Average Directional Movement Index Rating
ADX	Average Directional Movement Index	APO	Absolute Price Oscillator
AROONOSC	Aroon Oscillator	AROON_UP	Aroon
RSI	Relative Strength Index	CCI	Commodity Channel Index
MFI	Money Flow Index	PLUS_DI	Plus Directional Indicator
MOM	Momentum	CMO	Chande Momentum Oscillator
WILLR	Williams %R	ROCP	Rate of change Percentage: (price-prevPrice)/prevPrice
TRIX	1-day Rate-Of-Change (ROC) of a Triple Smooth EMA	RR	Daily Simple Rate Of Return
PPO	Percentage Price Oscillator	LOG_RR	Daily Log Returns
AD	Chaikin A/D Line		

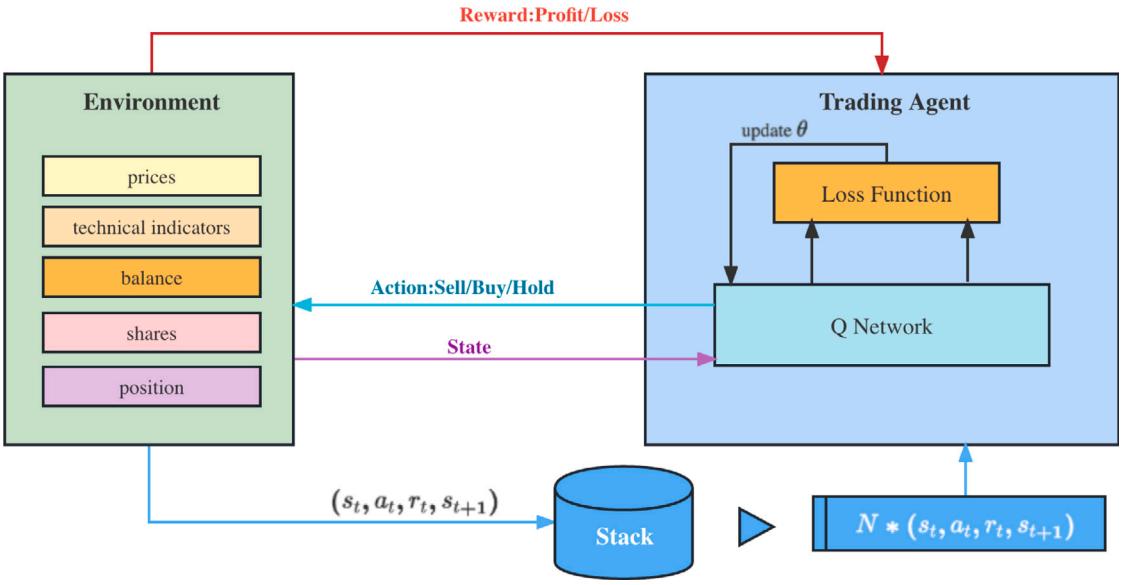


Fig. 2. The structure of efficient deep SARSA.

3.2.2. The structure of Q-network based on BiLSTM-attention

It is well known that it is advantageous to represent the action-value function Q in a functional approximation, especially when the state space is large or continuous. Therefore, given the current state s, select the action a by the ϵ-greedy method, observe the next state s' and reward r_t, then the current state–action value is $Q(s, a)$. Therefore, the current optimal state action value can be estimated as Eq. (10).

$$Q^*(s, a) = \mathbb{E}[r_t + \gamma Q(S_{t+1}, A_{t+1})|S_t = s, A_t = a], \tag{10}$$

where a_{t+1} is the next action selected by ϵ-greedy.

The effectiveness of DRL algorithms largely depend on the architecture of the value/policy network used by agents. Consequently, a network structure has been devised to approximate the action-value function of BiLSTM-Attention. This network configuration encompasses four primary layers: input layer, encoder layer, attention layer, and action layer, as depicted in Fig. 3. In this study, the hierarchical attention mechanism proposed by Yang, Yang, Dyer, He, Smola et al. (2016) is adopted to construct the aforementioned network.

Input Layer: As mentioned in Section 3.1.1, the input layer comprises M components, encompassing various technical indicators and the closing price from the previous N days, representing the state at time step t. In the training process, a mini-batch is chosen to train the network.

Encoder Layer: Initially, the input is subjected to processing through a BiLSTM network to acquire a hidden representation, denoted as h_i, effectively capturing temporal information (Graves & Schmidhuber, 2005). The fundamental concept behind BiLSTM is the acquisition of dynamic rules governing time series data. Firstly, M features, encompassing the stock closing data of the previous N days and several TA, are supplied to the network as the state at time step t. Subsequently, a hidden representation h_t is derived, encapsulating summarized dynamic time sequence data rules via the BiLSTM network. BiLSTM comprises two LSTM models operating in opposing directions. Within the LSTM network (Hochreiter & Schmidhuber, 1997), a memory cell incorporates four principal components: an input gate i_t, a self-recurrent connection neuron c_t, a forget gate f_t, and an output gate o_t. The detailed structure of LSTM is illustrated in Fig. 4.

The mathematical formalization of an LSTM is defined as Eq. (11). The self-recurrent connection ensures that the state of the memory cell remains constant from one time step to another. Input gates allow the input signals to change the state of a memory cell or prevent it. The output gate allows the state of a memory cell to affect other neurons or prevent it. The forget gate regulates the self-recurrent connections of memory cells, allowing cells to remember or forget their previous states as needed.

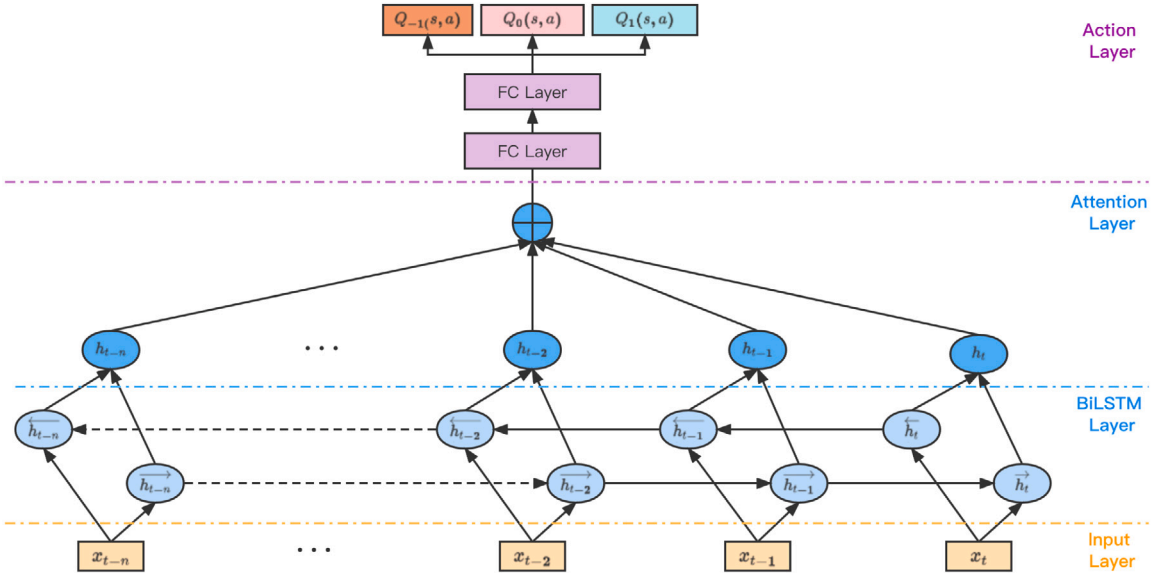


Fig. 3. The structure of Q network.

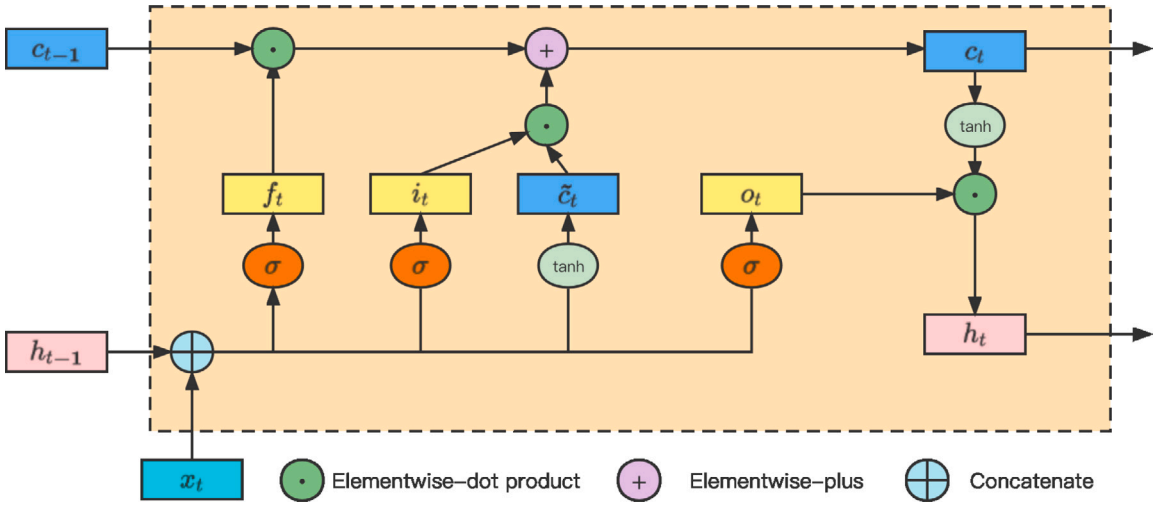


Fig. 4. The structure of LSTM.

$$
\begin{aligned}
i_t &= \sigma(x_t w_{xi} + h_{t-1} w_{hi} + b_i) \\
f_t &= \sigma(x_t w_{xf} + h_{t-1} w_{hf} + b_f) \\
o_t &= \sigma(x_t w_{xo} + h_{t-1} w_{ho} + b_o) \\
\tilde{c}_t &= \sigma(x_t w_{xc} + h_{t-1} w_{hc} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh(c_t),
\end{aligned} \tag{11}
$$

where the weights matrix $w_{xi}, w_{hi}, w_{xf}, w_{hf}, w_{xo}, w_{ho}, w_{xc}, w_{hc}$, and biases b_i, b_f, b_o, b_c are all the model parameters. $\odot$ denotes the element-wise product (Hadamard product).

However, BiLSTM is an improvement and enhancement of LSTM, including a forward LSTM and a backward LSTM in the hidden layer. The forward LSTM captures the feature information of positive temporal heading, and the backward LSTM captures the feature information of negative temporal heading. BiLSTM utilizes directional temporal information through a reverse-updated hidden layer. The mathematical formalization of BiLSTM is defined as Eq. (12).

$$
\begin{aligned}
\overrightarrow{h} &= \phi(x_t w_{xh}^{(f)} + \overrightarrow{h}_{t-1} w_{hh}^{(f)} + b_h^{(f)}) \\
\overleftarrow{h} &= \phi(x_t w_{xh}^{(b)} + \overleftarrow{h}_{t-1} w_{hh}^{(b)} + b_h^{(b)}) \\
h_t &= [\overrightarrow{h}, \overleftarrow{h}] \\
o_t &= h_t w_{hq} + b_q,
\end{aligned} \tag{12}
$$

where the weights matrix $w_{xh}^{(f)} \in \mathbb{R}^{d\times h}, w_{hh}^{(f)} \in \mathbb{R}^{h\times h}, w_{xh}^{(b)} \in \mathbb{R}^{d\times h}, w_{hh}^{(b)} \in \mathbb{R}^{h\times h}, w_{xq} \in \mathbb{R}^{2h\times q}$, and biases $b_h^{(f)} \in \mathbb{R}^{1\times h}, b_h^{(b)} \in \mathbb{R}^{1\times h}, b_q \in \mathbb{R}^{1\times q}$ are all the model parameters. $\odot$ denotes the element-wise product (Hadamard product).

Attention Layer: In addition, Mnih et al. (2014) was the first to apply an AM to an RNN model for image classification, achieving extremely high accuracy. Since different hidden states of different days have different weights for the RL agent, we adopted a hierarchical attention mechanism to flexibly utilize the most relevant part of the weights for each day and each hidden state by weighting the combination of all the encoded input vectors, where the most relevant vectors are given the highest weights (Yang, Yang, Dyer, He and Hovy, 2016).

Algorithm 1: Efficient deep SARSA algorithm

Input: stock close price, TA;
Initialize data stack D with size of N;
Initialize the Q network with random weights θ;
for *episode = 1 to N* **do**
  Initialize sequence $s_1 = \{x_1\}$;
  Select a_1 with ϵ-greedy method;
  for *t=1 to T* **do**
    Execute the action a_t in the environment and get reward r_t and the next state s_{t+1};
    Store the transition $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$ into stack D;
    Sample data from stack D;
    select a_{t+1} with ϵ-greedy method;
    Set:

$$y_i = \begin{cases} r_i, & \text{if episode terminates at step } i+1, \\ r_i + \gamma Q(s_{i+1}, a_{t+1}; \theta)), & \text{otherwise.} \end{cases}$$

    Train the network with loss function $L(\theta) = E[(y_i - Q(s_i, a_i; \theta))^2]$; $s_t \leftarrow s_{t+1}$; $a_t \leftarrow a_{t+1}$;
  end
end

It is designed as Eq. (13).

$$\begin{aligned} u_i &= \tanh(W h_i + b) \\ \alpha_i &= \frac{\exp(u_i^T u)}{\sum_i \exp(u_i^T u)} \\ v &= \sum_i \alpha_i h_i, \end{aligned} \tag{13}$$

where h_i represents hidden state, W and b are the model parameters.

Action Layer: Finally, an attention value vector v is fed into an action network composed of two fully connected layers. The purpose of this process is to derive the Q-values related to the three available actions.

3.2.3. Efficient deep SARSA algorithm

To summarize, the generation of optimal trading strategies by interacting with the environment was facilitated through the utilization of the efficient deep SARSA algorithm. In algorithmic trading, when the agent selects a profitable trading action, it garners a positive return. Conversely, if a loss ensues subsequent to the selection of a trading action, the agent receives a negative reward. These rewards can be used as incentives to guide the agent to make the right choice of action in the future. Consequently, the detailed training procedure for efficient deep SARSA is encapsulated in Algorithm 1.

4. Experiment and results

4.1. Dataset

The proposed method was tested in the three major stock indices of the US stock market and General Electric (GE). The three stock indices included the S&P 500 (S&P), the Dow Jones (DJI) and the NASDAQ (IXIC). At the same time, we chose the past 18 years as the trading period, which was divided into the training set from Jan-01-2008 to Dec-31-2017 and the test set from Jan-01-2018 to Dec-31-2022. The price history of each asset are shown in Fig. 5.

However, feature selection and extraction was a crucial process in many financial asset trading models as it greatly influenced the performance of the model. In this paper, a total of 33 features were used, such as closing price and some TA. Table 3 describes each technical indicator.

4.2. Evaluation metrics

To comprehensively and objectively assess the performance of a trading strategy, we selected three performance metrics shown below.

- **Cumulative return (CR):** CR measures the total return on an investment over a specified period and represents the cumulative effect of all returns earned on an investment since the beginning of the investment horizon. The formula for calculating the CR in percentage terms is shown in Eq. (14). W_T is the final value of the investment. W_0 is the initial amount of the investment.

$$CR = \left(\frac{W_T - W_0}{W_0}\right) * 100 \tag{14}$$

- **Sharpe ratio (ShR):** ShR shows the average return over the risk-free rate per unit of total risk and is calculated as Eq. (15), where R_f is the return on the risk-free asset and $E\{R_p\}$ is the expected value of the portfolio value. In this study, $R_f = 0$ is assumed.

$$\text{ShR} = \frac{E\{R_p\} - R_f}{\sigma_p} \tag{15}$$

- **Annualized return (AR):** AR is the annual average profit and loss (%) provided by an investment during trading activity. The formula for calculating the AR in percentage terms is shown in Eq. (16).

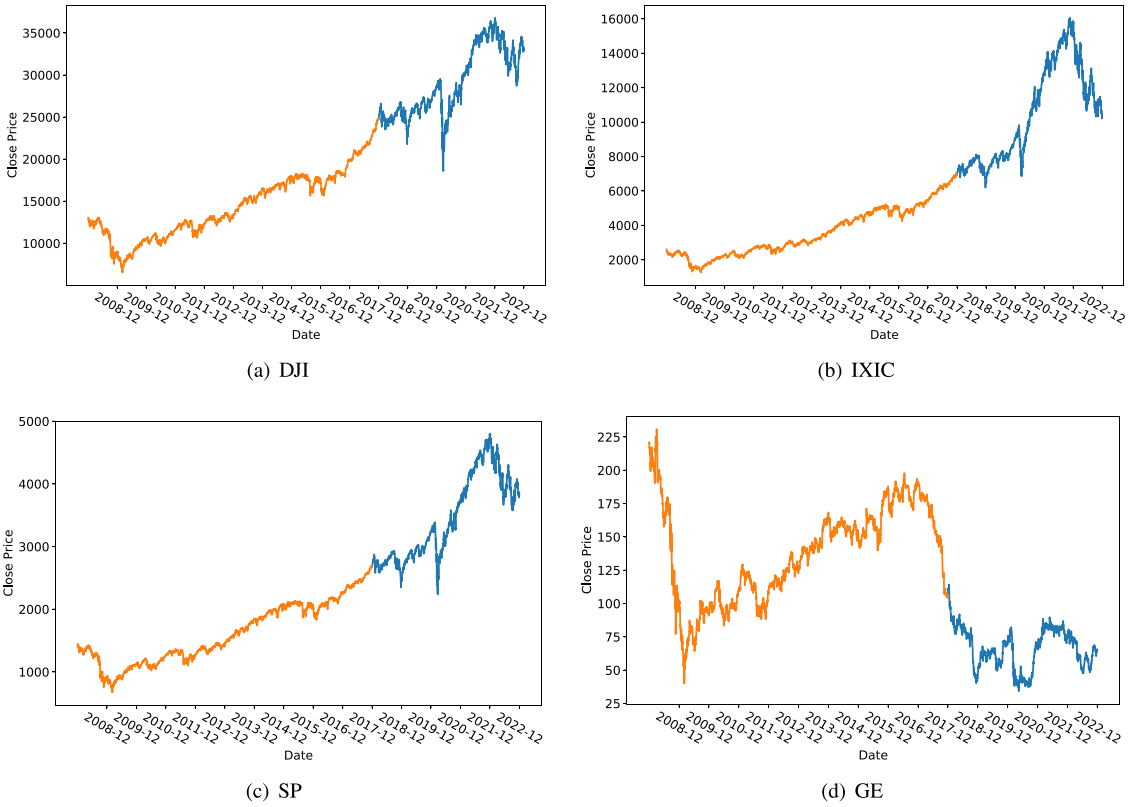
$$\text{AR} = ((1 + \text{CuR})^{\frac{365}{\text{Days Held}}} - 1) * 100 \tag{16}$$

where CuR is calculated by $\sum_{t=1}^{T} \frac{C_t - C_{t-1}}{C_{t-1}}$.

4.3. Baseline methods

To objectively evaluate the performance of the efficient deep SARSA algorithm, we compare the proposed efficient deep SARSA algorithm with value-based reinforcement learning methods (e.g., TDQN, DQN-vanilla) and some traditional trading strategies (e.g., buy-and-hold, sell-and-hold, moving average mean reversion, and moving average trend following).

- **Buy and Hold (B&H):** In this strategy, an investor selected an asset and takes a long position in the first step of investing. Assets purchased are held until the end of the period, regardless of price fluctuations.



(a) DJI (b) IXIC

(c) SP (d) GE

Fig. 5. The price movement of four assets[01/01/2008–12/31/2022][Train set in orange, Test set in blue].

- **Sell and Hold (S&H):** In this strategy, an investor selected an asset and takes a short position in the first step of investing. Assets purchased are held until the end of the period, regardless of price fluctuations.
- **Mean Reversion with Moving Averages (MR):** MR aims to profit from the assumption that asset prices tend to revert to their historical mean or average over time. In this paper, a 10-days Simple Moving Average (SMA) is used.
- **Trend Following with Moving Averages (TF):** TF is a popular trading strategy used by traders and investors to identify and profit from trends in financial markets. The strategy relies on the use of moving averages, which are statistical calculations of an asset's average price over a specific period of time. In this paper, 5 days SMA and 10 days SMA are used.
- **TDQN:** TDQN was proposed by Théate and Ernst (2021) in order to solve the challenge of determining the optimal trading position in stock market activities. This approach combines DQN with a Q-network consisting of five fully connected layers.
- **DQN-Vanilla:** DQN-Vanilla was proposed by Taghian et al. (2022) to create trading rules based on stock candlestick data. This approach combines DQN with a Q-network consisting of three fully connected layers.

4.4. Experimental setup

We compared the proposed methods with six baselines: B&H, S&H, MR, TF, TDQN, and the DQN-Vanilla. The initial capital was set at $500,000, and the transaction cost was assumed to be 0.01% For the LSTM/BiLSTM part, the window length M was set to 10, the number of layers was set to 1, and the number of neurons was 64.

To ensure that the neural network's inputs fall within a reasonable range, we normalize the data by calculating the mean and variance. The proposed method uses the Adam optimizer to train all models, setting the learning rate to 0.001. The models were implemented using the Python Pytorch library and the training process was terminated after reaching 100 iterations. The calculation of TA indicators was performed using the Python wrapper for the TA-Lib software library.[1] The hardware and software used in this paper are as follows: processor is Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70 GHz, GPU is NVIDIA Tesla V100 with the 32 GB memory. The software versions used are: Python is v3.7, Pytorch is v1.10, and cuda is v10.1. The system is Ubuntu 18.04.6 LTS.

Fig. 6 describes the progression of cumulative reward during the training of the efficient deep SARSA model using short ShR on four different datasets. As shown in Fig. 6, the cumulative rewards increased as the number of training episodes increases, eventually reaching a point of convergence. These cumulative rewards could be used to evaluate the agent's training progress. They represented the sum of rewards earned by the agent over the entire trajectory from beginning to end. When the accumulated rewards remained relatively stable over multiple consecutive episodes, with no abrupt drops or spikes in subsequent episodes, it indicated that the training has reached a state of convergence. These cumulative rewards were a vital performance metric used to evaluate the effectiveness of the efficient deep SARSA algorithm. They quantify the total rewards acquired by the model over training episodes, reflecting its ability to make profitable trading decisions. It is evident from the figure that there were no sudden drops or spikes in accumulated rewards for each agent after 10 or 30 episodes.

[1] http://ta-lib.org/.

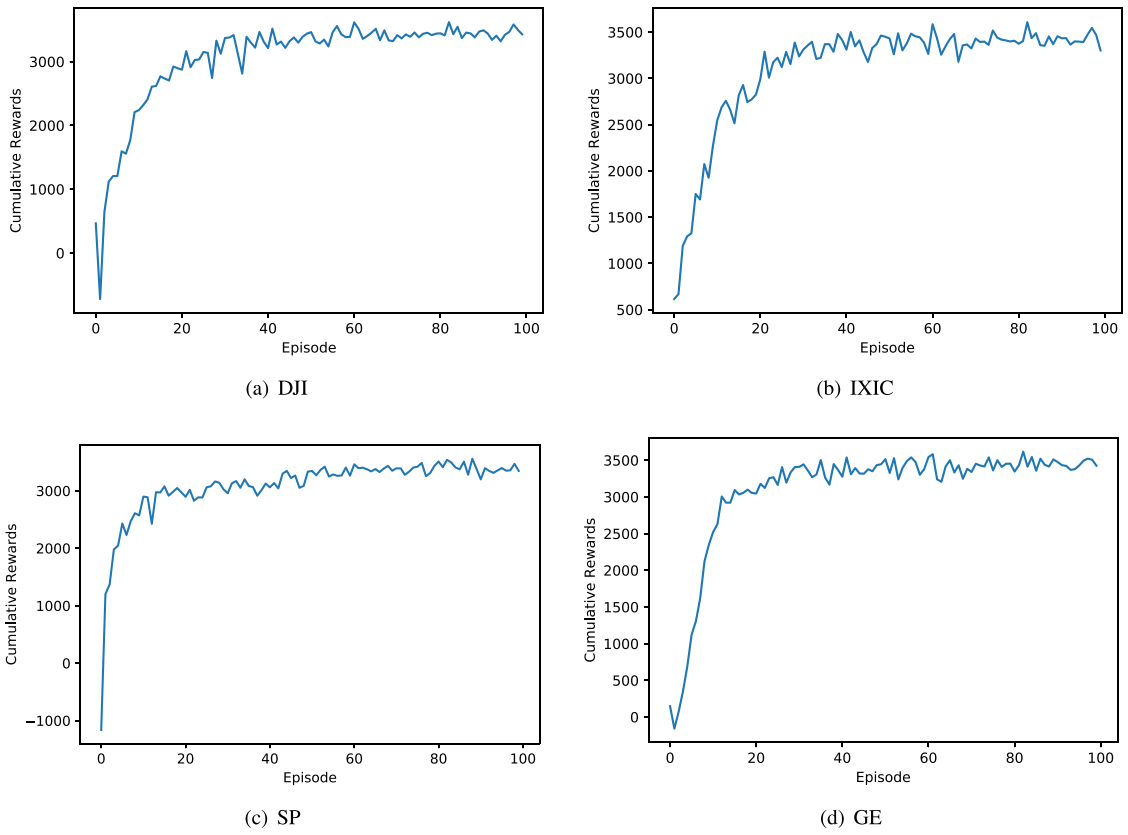


Fig. 6. Accumulated reward of the efficient deep SARSA with BiLSTM in four assets.

Table 4
Performance of the efficient deep SARSA on four assets with network of LSTM and BiLSTM.

Assets	Metrics	LSTM	Bi-LSTM
DJI	CR%	123.52	**150.50**
	AR%	13.94	**15.26**
	ShR	0.92	**1.05**
IXIC	CR%	231.41	**582.17**
	AR%	18.68	**25.34**
	ShR	1.19	**1.86**
SP500	CR%	87.05	**109.07**
	AR%	11.74	**13.24**
	ShR	0.73	**0.82**
GE	CR%	187.95	**242.58**
	AR%	19.82	**21.27**
	ShR	0.74	**0.85**

4.5. Experimental results

4.5.1. Comparison with different network

To evaluate the efficiency of the efficient deep SARSA algorithm when using different network architectures, we conducted experiments across four datasets. Table 4 displays the outcomes related to various performance metrics, such as CR, AR, and ShR, when employing LSTM and BiLSTM network configurations. Upon analyzing the table, it became evident that the BiLSTM network outperformed LSTM in terms of delivering superior results.

4.5.2. Comparison with baselines

In this section, we conducted a comprehensive comparison between the proposed efficient deep SARSA algorithm and six baseline methods to evaluate its performance across four different stock datasets, as shown in Table 5. The performance assessment focused on key trading metrics, including CR, ShR, and AR.

Table 5 presents the trading performance of various methods across the four assets. Notably, the efficient deep SARSA strategy exhibited the most favorable results among all the listed metrics, indicating its promising performance. Specifically, it achieved significantly higher cumulative returns compared to the other strategies. For instance, on the IXIC dataset, the efficient deep SARSA strategy achieved a remarkable CR of 582.17%, outperforming all other methods. Additionally, it demonstrated a ShR of 1.86, indicating favorable risk-adjusted returns. For the GE dataset with relatively large fluctuations, the efficient deep SARSA strategy achieved a CR of 242.58%, surpassing all other methods. It also exhibited a ShR of 0.85, further highlighting its ability to generate favorable risk-adjusted returns. These observations hold true for the remaining assets listed in the table. The efficient deep SARSA consistently outperformed the baseline methods across CR, ShR, and AR metrics, demonstrating its superior performance. These results affirm the effectiveness of the proposed efficient deep SARSA algorithm in developing profitable trading strategies with reduced risk.

Figs. 7 to 10 describe the cumulative return curves generated by employing different trading strategies, encompassing B&H, S&H, MR, TF, TDQN, DQN-Vanilla, and the proposed efficient deep SARSA strategy. The figures clearly demonstrated the outstanding performance of the efficient deep SARSA strategy relative to the other approaches. It demonstrated an impressive capability to minimize the risk associated with substantial losses while achieving superior returns. Furthermore, the four assets managed under the efficient deep SARSA strategy exhibited a more smoother upward movement compared to the benchmark strategy.

4.5.3. Analysis of strategy

Figs. 11 through 14 present the trading signals generated by the efficient deep SARSA strategy across different asset categories. These visual

Table 5
The performance of various trading approaches on four assets.

Assets	Metrics	B&H	S&H	MR	TF	TDQN	DQN-Vanilla	Efficient deep SARSA
DJI	CR%	28.81	−28.83	−37.66	0.58	59.67	57.44	**150.50**
	AR%	9.70	−5.97	−13.14	2.44	9.59	13.97	**15.26**
	ShR	0.35	−0.15	−0.40	0.10	0.55	0.53	**1.05**
IXIC	CR%	48.09	−48.11	−37.37	−0.26	56.77	46.63	**582.17**
	AR%	13.98	−100.00	−10.94	3.53	10.04	13.45	**25.34**
	ShR	0.44	−0.39	−0.27	0.11	0.48	0.43	**1.86**
SP500	CR%	39.66	−39.68	−24.98	−8.78	32.87	47.22	**109.07**
	AR%	11.70	−1.72	−5.75	−0.12	7.03	12.61	**13.24**
	ShR	0.42	−0.03	−0.18	−0.01	0.37	0.46	**0.82**
GE	CR%	−38.07	38.05	−89.95	52.96	−35.51	24.29	**242.58**
	AR%	−0.18	10.62	−100.00	18.50	−0.41	6.58	**21.27**
	ShR	−0.01	0.45	−0.80	0.42	−0.01	0.46	**0.85**

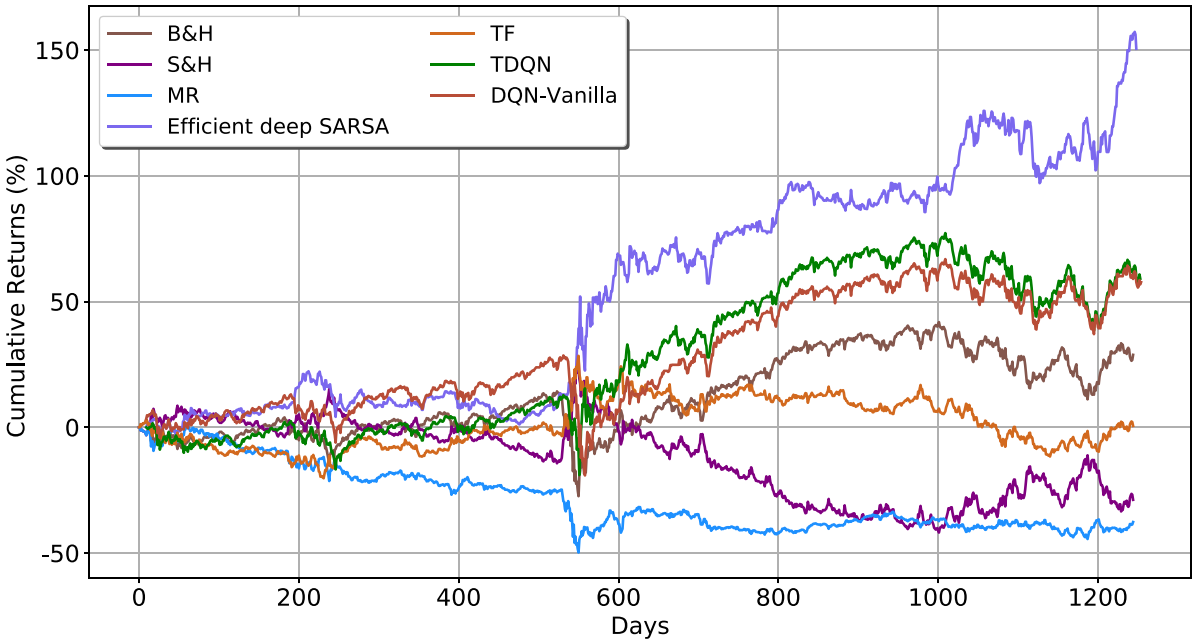


Fig. 7. Comparison of cumulative returns of different methods on the DJI test set.

representations underscore the strategy's effectiveness in accurately identifying market trends and its capacity to adapt positions to manage risks, especially during turbulent market periods. The trading signals produced by the efficient deep SARSA strategy underscore its ability to recognize favorable market conditions and respond accordingly.

As an example, the strategy exhibits a preference for opening short positions during bearish market conditions, as evidenced by its actions with the GE stock, where prices primarily exhibit a downward trend over time. Conversely, for assets such as DJI, IXIC, and SP500, the strategy identifies the profit potential in establishing long positions, aligning with the predominantly bullish market trends characterized by rising prices over time. This sensitivity to market dynamics can be primarily attributed to the BiLSTM's proficiency in extracting meaningful insights from extensive and noisy historical price data.

5. Discussion

In this experiment, we conducted a comprehensive analysis of the proposed method from two distinct angles, comparing the efficient deep SARSA approach with several baseline strategies. Firstly, in terms of cumulative return, the proposed method consistently outperformed the baselines across all datasets. Moreover, it exhibited a notably high ShR, indicating its capacity to effectively manage risks while maximizing returns. These results align with the theoretical underpinnings established in previous experiments.

Secondly, in terms of trading decisions, the agent showed a remarkable ability to anticipate future trends in stock prices and execute favorable trading decisions. The strategy's effectiveness in accurately identifying market trends and its capacity to adapt positions to manage risks, especially during turbulent market periods.

Thirdly, Zeng et al. (2023) introduced a set of simple one-layer linear models named LTSF-Linear outperforms existing sophisticated Transformer-based LTSF models, such as LogTrans (Li et al., 2019), Informer (Zhou et al., 2021), Autoformer (Wu, Xu et al., 2021), Pyraformer (Liu, Yu et al., 2021), and FEDformer (Zhou et al., 2022). Moreover, consideration will be also given to the combination of Transformers with reinforcement learning. In some environments, the performance of reinforcement learning-based Transformer methods is found to be comparable to that of stochastic strategies, thereby raising questions about their suitability for the handling of temporal information in reinforcement learning. Furthermore, Transformers are acknowledged for their slower training speed and greater resource consumption, particularly when dealing with large temporal context windows. Additionally, the advantages of the Transformer architecture may not be as evident in cases where the dimensionality of the ambient data is low, as its performance and efficiency benefits may not be fully realized.

Furthermore, the agents within the proposed framework demonstrated noteworthy convergence throughout the training process. This convergence signifies the framework's effectiveness in overcoming

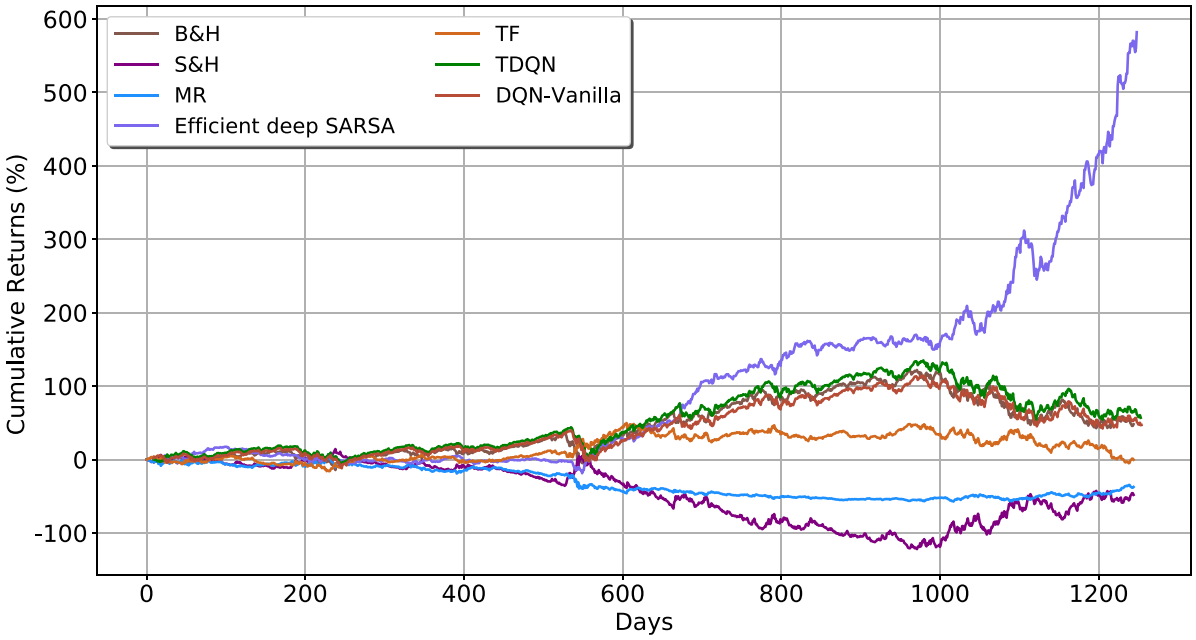


Fig. 8. Comparison of cumulative returns of different methods on the IXIC test set.

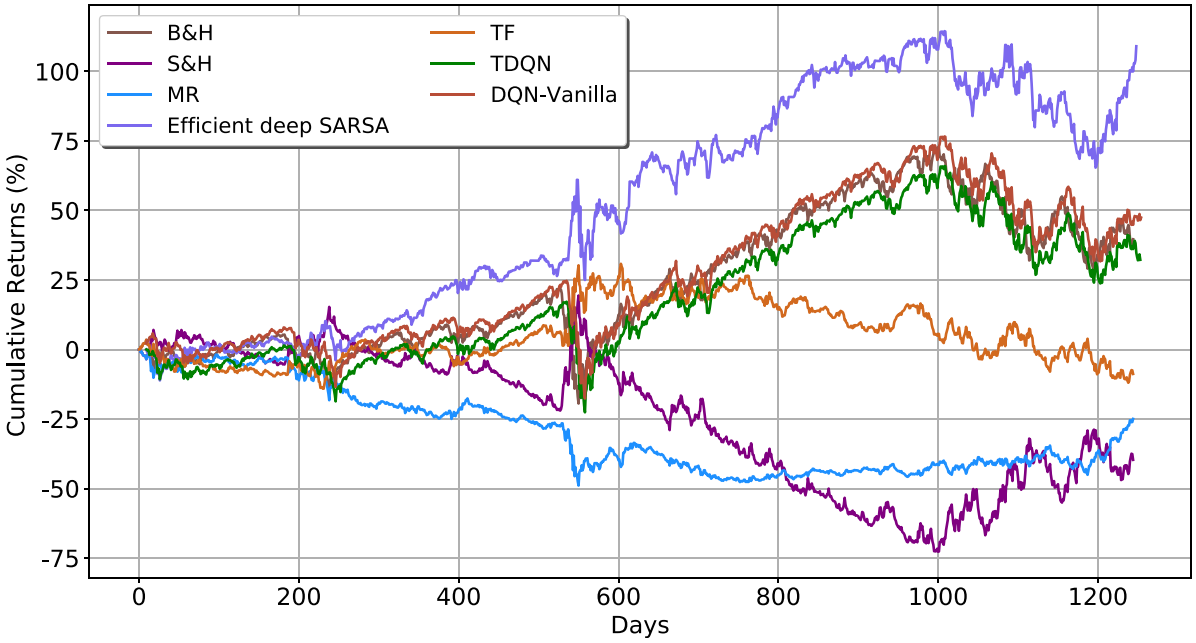


Fig. 9. Comparison of cumulative returns of different methods on the SP500 test set.

the inherent noise and uncertainties frequently present in financial data, highlighting its robustness in handling complex financial dynamics.

The advantages of the efficient deep SARSA approach are twofold. Firstly, its on-policy nature allows the strategy to swiftly adapt to evolving market conditions. Secondly, the utilization of BiLSTM enables the detection of market states from both raw and noisy data, enhancing its ability to make informed decisions.

Overall, the experimental findings provide substantial evidence of the effectiveness of the proposed framework. The agents not only exhibit strong convergence but also consistently outperform baseline methods, validating the efficacy of the approach.

6. Conclusion

This paper introduces an innovative financial trading framework, efficient deep SARSA, which harnesses the power of deep reinforcement learning to tackle the intricate challenges posed by financial markets, characterized by high volatility, non-linearity, and complex dynamics. By incorporating an agent to learn and interact with the market environment, this framework enhances the decision-making process and overall trading performance. The key contributions are summarized as follows:

1. A novel financial trading framework is presented, built upon SARSA, renowned for its on-policy nature, enabling rapid adaptation to

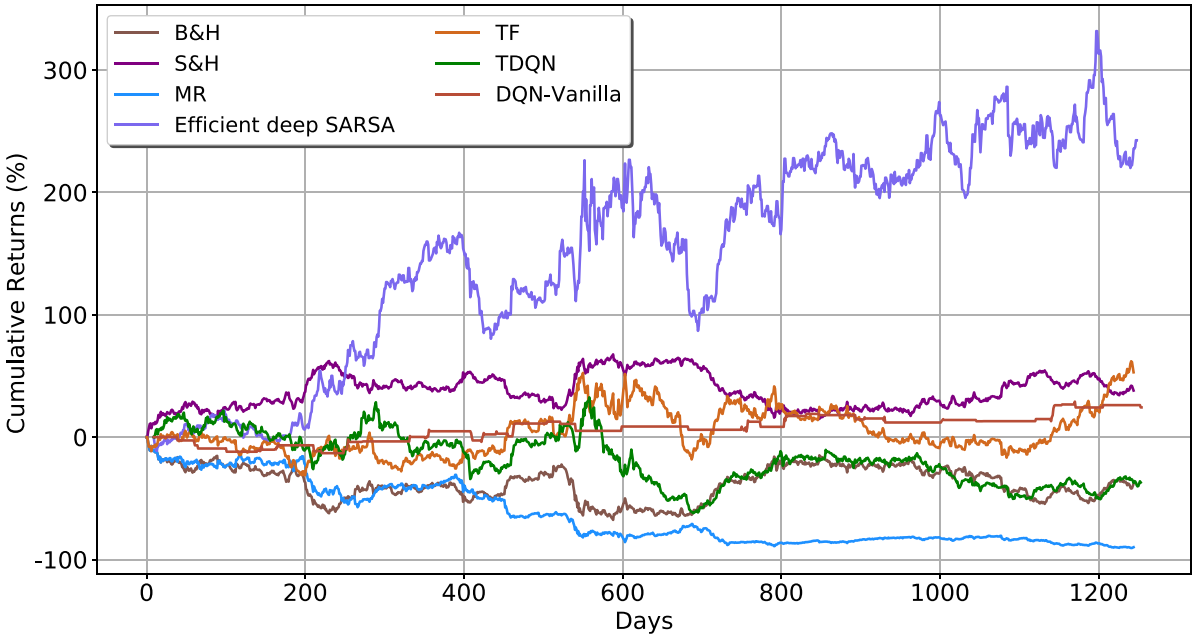


Fig. 10. Comparison of cumulative returns of different methods on the GE test set.

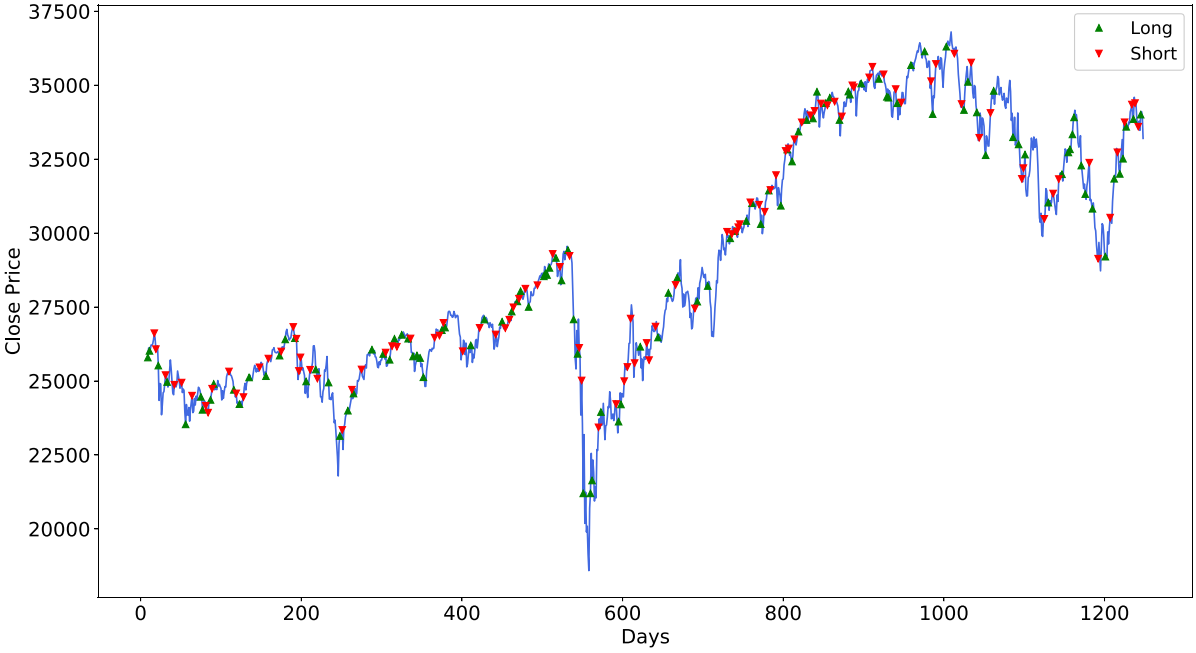


Fig. 11. The trading signals for DJI based on the efficient deep SARSA algorithm.

changing market conditions. Furthermore, SARSA exhibits notable convergence, signifying its effectiveness in handling the inherent noise and uncertainties within financial data, thereby demonstrating resilience in addressing intricate financial dynamics.

2. The BiLSTM-Attention framework integrates BiLSTM and an attention mechanism, enabling precise predictions of stock trends. A comparative analysis between BiLSTM and LSTM reveals that BiLSTM excels in capturing market states from raw and noisy data.

3. The evaluation covered the three major U.S. stock market indices and General Electric stock, comparing the performance of the efficient deep SARSA to various baseline methodologies. The methodology is capable of delivering excess returns in highly volatile market conditions and allows for timely position reorientation to accurately capture market trends and mitigate risk.

While promising results are demonstrated by the efficient deep SARSA algorithm, the need for further refinement is acknowledged.

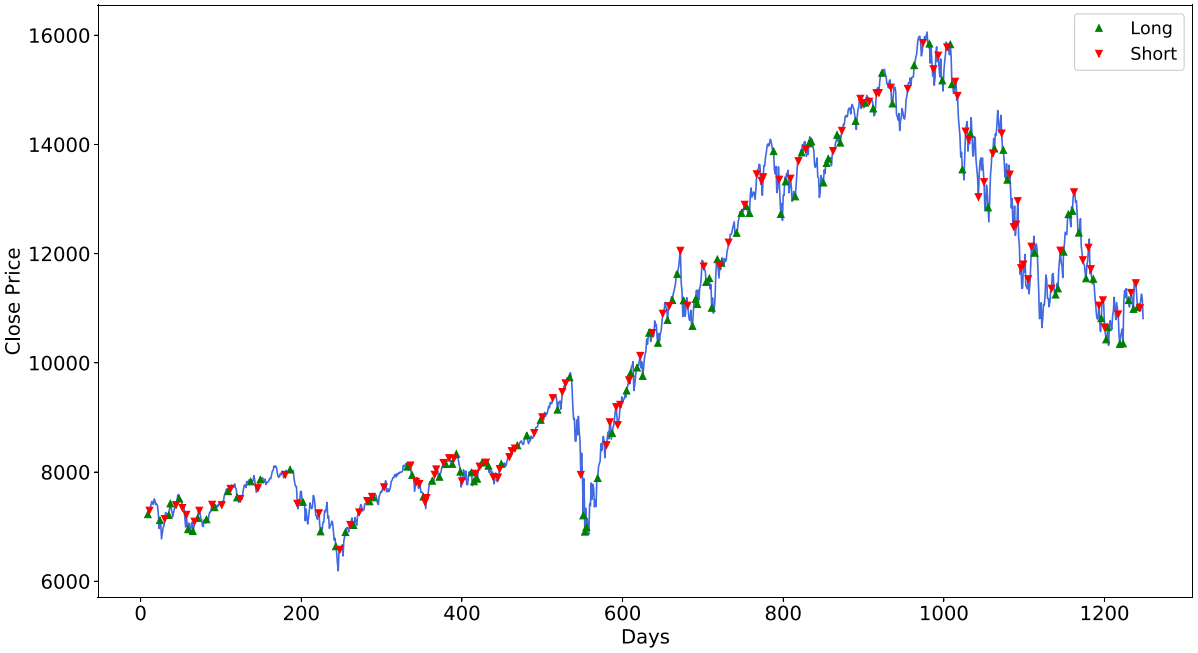


Fig. 12. The trading signals for IXIC based on the efficient deep SARSA algorithm.

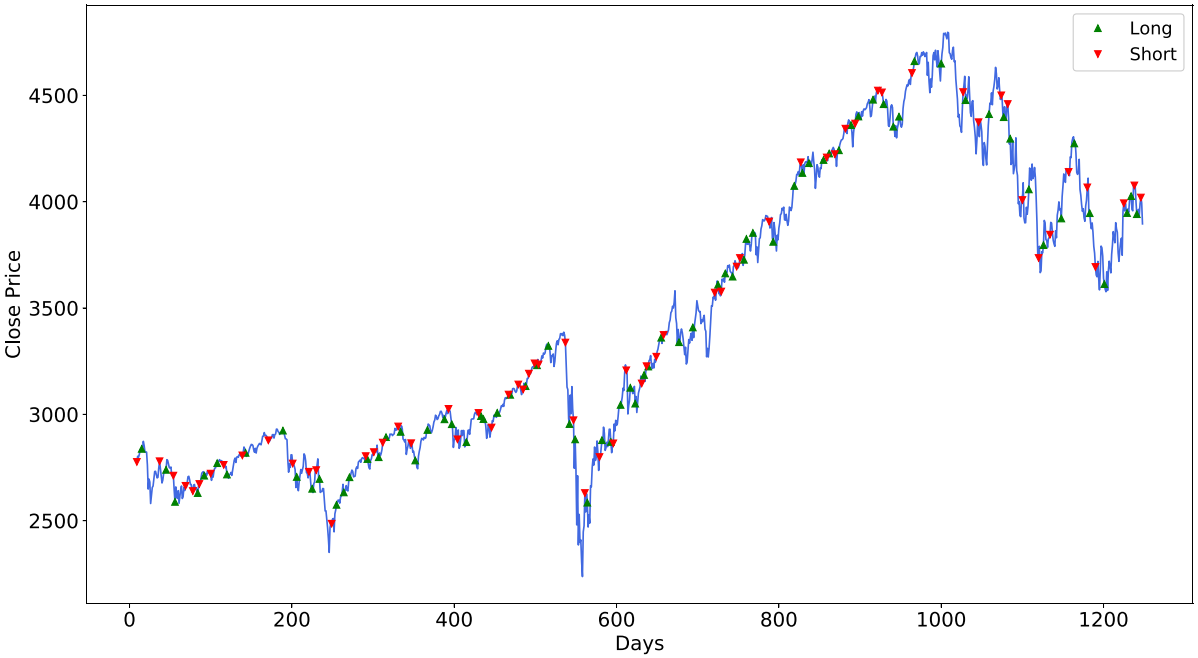


Fig. 13. The trading signals for SP500 based on the efficient deep SARSA algorithm.

Firstly, the on-policy mechanism of SARSA underutilizes empirical interaction data, prompting consideration for incorporating the empirical playback mechanism of DDQN to enhance data utilization. Secondly, the current algorithms are tailored to specific assets, limiting their application to a subset of underlying assets. In addition, consideration should be given to the impact of trading behavior on the market environment as well as addressing slippage, both of which are relevant aspects of real-world trading scenarios.

Looking ahead, future research efforts will delve into more complex scenarios. The development of ChatGPT has prompted exploration into enhancements through the training of generalized models on mixed data or combining extensive model training with pre-trained on data from different markets and subsequent fine-tuning on specific assets. Furthermore, by considering the impact of trading behavior on the environment and addressing slippage, better simulation of real-world trading conditions can be achieved. These advancements will contribute to the development of more reliable and efficient trading strategies, providing practical insights for financial professionals and policymakers.

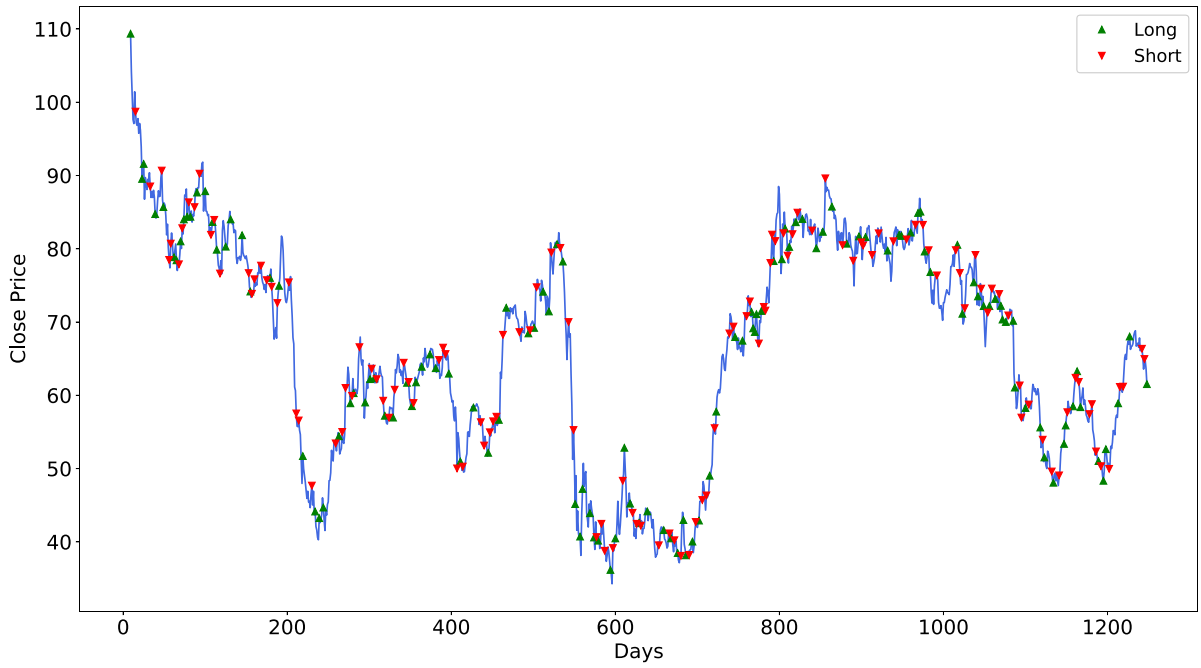


Fig. 14. The trading signals for GE based on the efficient deep SARSA algorithm.

CRediT authorship contribution statement

Yuling Huang: Methodology, Conceptualization, Investigation, Software, Writing – original draft. **Xiaoxiao Wan:** Methodology, Validation. **Lin Zhang:** Methodology, Visualization, Writing – original draft. **Xiaoping Lu:** Supervision, Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported in part by the Faculty Research Grants, Macau University of Science and Technology (Project no. FRG-22-001-INT). The authors would like to express the appreciation to Mr. Zhen Guo, a master's degree student at the School of Computer Science and Engineering, Macau University of Science and Technology. His insightful inputs and suggestions have significantly enriched the content of this paper.

References

Bajpai, S. (2021). Application of deep reinforcement learning for Indian stock trading automation. arXiv, abs/2106.16088.

Bansal, A., Patel, D. N., Rishabh, K., & Sneha, M. (2022). Stock prediction model using Seq2Seq and bi-directional LSTM. In *2022 4th International Conference on Circuits, Control, Communication and Computing (I4C)* (pp. 275–278).

Brim, A., & Flann, N. S. (2022). Deep reinforcement learning stock market trading, utilizing a CNN with candlestick images. *PLoS One*, *17*(2), Article e0263181.

Carta, S. (2021). A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *51*(2).

Chakole, J., & Kurhekar, M. (2020). Trend following deep Q-learning strategy for stock trading. *Expert Systems*, *37*(4), 12514.

Cheng, L.-C., Huang, Y.-H., Hsieh, M.-H., & Wu, M.-E. (2021). A novel trading strategy framework based on reinforcement deep learning for financial market predictions. *Mathematics*, *9*(23), 3094.

Corazza, M., Fasano, G., Gusso, R., & Pesenti, R. (2019). A comparison among reinforcement learning algorithms in financial trading systems. *ERN: Other Econometrics: Econometric & Statistical Methods - Special Topics (Topic)*.

Cornalba, F., Disselkamp, C., Scassola, D., & Helf, C. (2022). Multi-objective reward generalization: Improving performance of deep reinforcement learning for selected applications in stock and cryptocurrency trading. ArXiv, abs/2203.04579.

Dang, Q.-V. (2020). Reinforcement learning in stock trading. In *Advanced computational methods for knowledge engineering: proceedings of the 6th international conference on computer science, applied mathematics and applications, ICCSAMA 2019 6* (pp. 311–322).

de Oliveira, R. A., Filho, H. S. R., Dalip, D. H., & Pereira, A. M. (2020). A tabular sarsa-based stock market agent. In *Proceedings of the first ACM international conference on AI in finance*.

Ge, J., Qin, Y., Li, Y., Huang, y., & Hu, H. (2022). Single stock trading with deep reinforcement learning: A comparative study. In *2022 14th international conference on machine learning and computing (ICMLC)* (pp. 34–43).

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, *18*(5–6), 602–610.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Huang, Y., Cui, K., Song, Y., & Chen, Z. (2023). A multi-scaling reinforcement learning trading system based on multi-scaling convolutional neural networks. *Mathematics*, *11*(11), 2467.

Huang, Y., Gao, Y., Gan, Y., & Ye, M. (2021). A new financial data forecasting model using genetic algorithm and long short-term memory network. *Neurocomputing*, *425*, 207–218.

Juairiah, F., Mahatabe, M., Jamal, H. B., Shiddika, A., Shawon, T. R., & Mandal, N. C. (2022). Stock price prediction: A time series analysis. In *2022 25th international conference on computer and information technology (ICCIT)* (pp. 153–158).

Khare, I. S., Martheswaran, T., Dassanaike-Perera, A., & Ezekiel, J. B. (2023). Evaluation of reinforcement learning techniques for trading on a diverse portfolio. ArXiv, abs/2309.03202.

Kim, Y., Ahn, W., Oh, K. J., & Enke, D. (2017). An intelligent hybrid trading system for discovering trading rules for the futures market using rough sets and genetic algorithms. *Applied Soft Computing*.

Kumar, K. V., & Anitha, R. (2022). A novel ensemble model by combining LSTM, BiLSTM, and facebook prophet algorithm to forecast stock prices. In *2022 third international conference on intelligent computing instrumentation and control technologies (ICICICT)* (pp. 1044–1047).

Lele, S., Gangar, K., Daftary, H., & Dharkar, D. (2020). Stock market trading agent using on-policy reinforcement learning algorithms. Available at SSRN 3582014.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, *32*.

Li, Y., Liu, P., & Wang, Z. (2022). Stock trading strategies based on deep reinforcement learning. *Scientific Programming*, *2022*.

Li, Y., Ni, P., & Chang, V. (2020). Application of deep reinforcement learning in stock trading strategies and stock forecasting. *Computing*, *102*(5).

Lima Paiva, F. C., Felizardo, L. K., Bianchi, R. A. d. C., & Costa, A. H. R. (2021). Intelligent trading systems: a sentiment-aware reinforcement learning approach. In *Proceedings of the Second ACM International Conference on AI in Finance* (pp. 1–9).

Liu, F., Li, Y., Li, B., Li, J., & Xie, H. (2021). Bitcoin transaction strategy construction based on deep reinforcement learning. *Applied Soft Computing*, *113*, Article 107952.

Liu, Y., Liu, Q., Zhao, H., Pan, Z., & Liu, C. (2020). Adaptive quantitative trading: An imitative deep reinforcement learning approach. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 2128–2135).

Liu, X.-Y., Yang, H., Chen, Q., Zhang, R., Yang, L., Xiao, B., & Wang, C. D. (2020). FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance. arXiv, abs/2011.09607.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., & Dustdar, S. (2021). Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*.

Liu, P., Zhang, Y., Bao, F., Yao, X., & Zhang, C. (2023). Multi-type data fusion framework based on deep reinforcement learning for algorithmic trading. *Applied Intelligence*, *53*(2), 1683–1706.

Lu, W., Li, J., Wang, J., & Qin, L. (2020). A CNN-BiLSTM-AM method for stock price prediction. *Neural Computing and Applications*, *33*, 4741–4753.

Luo, Y., & Duan, Z.-S. (2023). Agent performing autonomous stock trading under good and bad situations. ArXiv, abs/2306.03985.

Ma, C., Zhang, J., Liu, J., Ji, L., & Gao, F. (2021). A parallel multi-module deep reinforcement learning algorithm for stock trading. *Neurocomputing*.

Mahayana, D., Shan, E., & Fadhl'Abbas, M. (2022). Deep reinforcement learning to automate cryptocurrency trading. In *2022 12th International Conference on System Engineering and Technology (ICSET)* (pp. 36–41).

Md, A. Q., Kapoor, S., A.V., C. J., Sivaraman, A. K., Tee, K. F., H., S., & N., J. (2022). Novel optimization approach for stock price forecasting using multi-layered sequential LSTM. *Applied Soft Computing*, *134*, Article 109830.

Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. *Advances in Neural Information Processing Systems*, *3*.

Mo, Y. A., & Jing, W. A. (2022). Adaptability of financial time series prediction based on BiLSTM. *Procedia Computer Science*, *199*, 18–25.

Nan, A., Perumal, A., & Zaiane, O. R. (2022). Sentiment and knowledge based algorithmic trading with deep reinforcement learning. In *Database and expert systems applications: 33rd international conference, DEXA 2022, Vienna, Austria, August 22–24, 2022, proceedings, part I* (pp. 167–180).

Pavel, M. I., Muhtasim, D. A., & Faruk, O. (2021). Decision making process of stock trading implementing DRQN and ARIMA. In *2021 IEEE Madras section conference (MASCON)* (pp. 1–6).

Pholsri, P., & Kantavat, P. (2023). Intraday stock trading strategy based on analysis using bidirectional long short-term memory networks. In *2023 6th international conference on artificial intelligence and big data (ICAIBD)* (pp. 572–578).

Ponomarev, E., Oseledets, I. V., & Cichocki, A. (2019). Using reinforcement learning in the algorithmic trading problem. *Journal of Communications Technology and Electronics*, *64*, 1450–1457.

Qiao, R., Chen, W., & Qiao, Y. (2022). Prediction of stock return by LSTM neural network. *Applied Artificial Intelligence*, *36*.

Roondiwala, M., Patel, H., & Varma, S. (2017). Predicting stock prices using LSTM. *International Journal of Science and Research (IJSR)*, *6*(4).

Shah, J., Jain, R., Jolly, V., & Godbole, A. (2021). Stock market prediction using bi-directional LSTM. In *2021 international conference on communication information and computing technology (ICCICT)* (pp. 1–5).

Sharaf, M., Hemdan, E. E.-D., El-Sayed, A., & El-Bahnasawy, N. A. (2022). An efficient hybrid stock trend prediction system during COVID-19 pandemic based on stacked-LSTM and news sentiment analysis. *Multimedia Tools and Applications*, 1–33.

Sharpe, & William, F. (1994). The sharpe ratio. *Journal of Portfolio Management*, *21*(1), 49–58.

Shi, Y., Li, W., Zhu, L., Guo, K., & Cambria, E. (2021). Stock trading rule discovery with double deep Q-network. *Applied Soft Computing*, *107*, Article 107320.

Shi, Y., Wang, Y., Qu, Y., & Chen, Z. (2023). Integrated GCN-LSTM stock prices movement prediction based on knowledge-incorporated graphs construction. *International Journal of Machine Learning and Cybernetics*, 1–16.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*.

Sridhar, S., & Sanagavarapu, S. (2021). Analysis of the effect of news sentiment on stock market prices through event embedding. In *2021 16th conference on computer science and intelligence systems (FedCSIS)* (pp. 147–150).

Sunny, M., Maswood, M., & Alharbi, A. G. (2020). Deep learning-based stock price prediction using LSTM and bi-directional LSTM model. In *2020 2nd novel intelligent and leading emerging sciences conference (NILES)*.

Sutton, R., & Barto, A. (2018). *Reinforcement learning:an introduction*. MIT Press.

Taghian, M., Asadi, A., & Safabakhsh, R. (2022). Learning financial asset-specific trading rules via deep reinforcement learning. *Expert Systems with Applications*, Article 116523.

Théate, T., & Ernst, D. (2021). An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications*, *173*, Article 114632.

Tran, M., Pham-Hi, D., & Bui, M. (2023). Optimizing automated trading systems with deep reinforcement learning. *Algorithms*, *16*, 23.

Vishal, M., Satija, Y., & Babu, B. S. (2021). Trading agent for the Indian stock market scenario using actor-critic based reinforcement learning. In *2021 IEEE international conference on computation system and information technology for sustainable solutions (CSITSS)* (pp. 1–5).

Wang, Z., Lu, W., Zhang, K., Li, T., & Zhao, Z. (2021). A parallel-network continuous quantitative trading model with GARCH and PPO. arXiv, abs/2105.03625.

Wu, J. M.-T., Li, Z., Herencsar, N., Vo, B., & Lin, J. C.-W. (2021). A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. *Multimedia Systems*, 1–20.

Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, *34*, 22419–22430.

Xiao, X. (2023). Quantitative investment decision model based on PPO algorithm. *Highlights in Science, Engineering and Technology*, *34*, 16–24.

Xu, X., Yang, M., Liu, H., & Zhang, D. (2022). A hybrid improved LSTM-CNN model for Chinese stock price trend prediction. In *2022 IEEE 4th international conference on civil aviation safety and information technology (ICCASIT)* (pp. 76–83).

Yang, H., Liu, X.-Y., Zhong, S., & Walid, A. (2020). Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1–8).

Yang, Z., Yang, D., Dyer, C., He, X., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies* (pp. 1480–1489).

Ye, Z. J., & Schuller, B. W. (2023). Human-aligned trading by imitative multi-loss reinforcement learning. *Expert Systems with Applications*, *234*, Article 120939.

Yu, Y. (2022). Research on the forecast of stock price index based on BiLSTM-GRU. In *2022 Euro-Asia conference on frontiers of computer science and information technology (FCSIT)* (pp. 81–85).

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023). Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence, Vol. 37* (pp. 11121–11128).

Zhang, Z., Dai, H. N., Zhou, J., Mondal, S. K., & Wang, H. (2021). Forecasting cryptocurrency price using convolutional neural networks with weighted and attentive memory channels. *Expert Systems with Applications*, *183*, Article 115378.

Zhang, Z., Zohren, S., & Roberts, S. (2020). Deep reinforcement learning for trading. *The Journal of Financial Data Science*, *2*(2), 25–40.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning* (pp. 27268–27286).

Zhou, P., & Tang, J. (2021). Improved method of stock trading under reinforcement learning based on DRQN and sentiment indicators ARBR. arXiv, abs/2111.15356.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 35* (pp. 11106–11115).