Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

A convolutional deep reinforcement learning architecture for an emerging stock market analysis

Anita Hadizadeh[a], Mohammad Jafar Tarokh[a*] and Majid Mirzaee Ghazani[a]

[a]*Department of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran*

C H R O N I C L E

Article history:
Received: October 16, 2024
Received in revised format: November 30, 2024
Accepted: January 22 2024
Available online: January 22, 2025

Keywords:
Deep reinforcement learning
DDQN
Convolutional neural network
Stock Market Prediction
Q-learning
Overfitting Prevention

A B S T R A C T

In the complex and dynamic stock market landscape, investors seek to optimize returns while minimizing risks associated with price volatility. Various innovative approaches have been proposed to achieve high profits by considering historical trends and social factors. Despite advancements, accurately predicting market dynamics remains a persistent challenge. This study introduces a novel deep reinforcement learning (DRL) architecture to forecast stock market returns effectively. Unlike traditional approaches requiring manual feature engineering, the proposed model leverages convolutional neural networks (CNNs) to directly process daily stock prices and financial indicators. The model addresses overfitting and data scarcity issues during training by replacing conventional Q-tables with convolutional layers. The optimization process minimizes the sum of squared errors, enhancing prediction accuracy. Experimental evaluations demonstrate the model's robustness, achieving a 67% improvement in directional accuracy over the buy-and-hold strategy across short-term and long-term horizons. These findings underscore the model's adaptability and effectiveness in navigating complex market environments, offering a significant advancement in financial forecasting.

© 2025 by the authors; licensee Growing Science, Canada.

1. Introduction

Financial markets are characterized by volatility, presenting investors with opportunities for gains and risks of losses due to their non-linear dynamics, complexity, and chaotic nature (Sharma et al., 2017). Traditional methods, such as statistical models and time series analysis, have been widely employed for market prediction. However, these methods often struggle to capture financial data's complex, non-linear relationships. Learning algorithms have been used to predict volatility in financial markets and have garnered significant attention due to their user-friendly nature and impressive predictive capabilities (Hu et al., 2021). Over the years, various methods have been employed to forecast stock market trends, ranging from traditional statistical modeling to advanced machine learning (ML) and deep learning (DL) approaches. DL algorithms rely on feature extraction and robust perception, enhancing the predictive prowess (Zhang et al., 2020). While these techniques have demonstrated significant promise in capturing complex market behaviors, they often fail to address financial environments' dynamic and uncertain nature.

Reinforcement learning (RL) and its extensions, such as deep reinforcement learning (DRL), have emerged as powerful tools for tackling these challenges. RL algorithms offer a promising approach for addressing the dynamic nature of fluctuating environments and tackling time series challenges, including those in financial markets. The robust feature extraction capability of DL can be harnessed to generate informative states for RL, facilitating the development of a strong dynamic decision-making system.

* Corresponding author.
E-mail address mjtarokh@kntu.ac.ir (M. J. Tarokh)

ISSN 1929-5812 (Online) - ISSN 1929-5804 (Print)
© 2025 by the authors; licensee Growing Science, Canada.
doi: 10.5267/j.dsl.2025.1.006

Integrating DL and RL can establish a comprehensive framework to optimize decision-making processes in dynamic time series environments, including diverse stock markets (Dong et al., 2020).

Despite the growing interest in DRL, most existing research has been focused on well-established markets, with limited attention given to emerging markets. The Iran stock market (TSM), a developing market with unique characteristics, remains relatively underexplored in the context of DRL-based prediction models. This study aims to bridge this gap by proposing a novel convolutional deep reinforcement learning architecture for stock market analysis tailored to TSM. The proposed model offers a robust solution for predicting stock market trends in an emerging market setting by leveraging convolutional techniques to enhance feature extraction and mitigate overfitting.

This paper outlines the proposed architecture, focusing on integrating the Dual Deep Q Network (DDQN) with convolutional neural networks (CNN). Extensive experiments evaluate the effectiveness of this hybrid approach, demonstrating significant improvements in prediction accuracy compared to traditional stock market strategies. Ultimately, this study contributes to the growing body of research in financial forecasting and aims to provide valuable insights into the application of DRL in emerging stock markets.

Contributions of this paper include:

1. Proposing a DRL architecture that integrates the DDQN algorithm with a CNN. This integration enhances feature extraction capabilities while addressing overfitting challenges and effectively managing data scarcity.
2. Utilizing daily stock prices and independent indicators derived from OHLC (Open, High, Low, Close) prices and trading volume as inputs to a convolutional neural network. The model focuses on the most liquid stocks of the TSM and incorporates trading volume as a complementary predictor to improve accuracy.
3. Developing an optimization process that minimizes the sum of squared errors, ensuring the alignment of the model's predictions with actual outcomes.
4. Conducting a thorough evaluation of the proposed model across both short-term and long-term timeframes and offering a detailed comparison with the buy-and-hold approach.

Experimental results demonstrate that the proposed model significantly improves prediction accuracy and investment returns, particularly compared to conventional strategies. This research represents a crucial step toward advancing financial research and applying DRL algorithms in emerging markets.

The structure of this paper is organized as follows: Section 2 reviews relevant research in the field. Section 3 describes the data and features used in this study. Section 4 explains the proposed methodology. Section 5 outlines the implementation process. Section 6 presents the experimental results and analysis. Section 7 offers a comprehensive discussion of the findings. Finally, Section 8 concludes the study and provides recommendations for future research.

2. Literature review

Research in stock market prediction utilizes various methods, including statistical modeling, machine learning (ML), DL, DRL, and hybrid approaches. Each technique offers unique advantages and limitations in capturing financial markets' complex and dynamic nature.

Several studies have emphasized the significance of deep neural networks (DNN) in stock price prediction, with some demonstrating that deep Q-networks (DQN) offer superior trend prediction accuracy compared to alternative models. Deng et al. (2016) compare DL and reinforcement learning (RL) concepts, demonstrating that DRL models reduce uncertainty in time series and improve learning systems' effectiveness in summarizing market conditions and taking optimal actions. Meng and Khushi (2019) depict the advantages of the RL approach, including integrating forecasting and prototype construction, aligning ML performance with investor goals, and considering transaction costs, market liquidity, and investor risk aversion. Yang et al. (2020) show DRL models' superiority over traditional models but note slow convergence as challenging. Li et al. (2020) prove the reliability and availability of the DRL model with experimental data. Their results show that DRL models are most similar to human learning.

Existing models often focus on developed markets with abundant historical data, overlooking emerging markets' unique characteristics and challenges. Research on emerging markets remains limited, and no comprehensive studies utilizing DRL have explored the Tehran Stock Market. Moghadam et al. (2019) demonstrate the value of TSM on stock profitability. Abounoori and Tour (2019) examine the interactions among the TSM, the United States, Turkey, and the United Arab Emirates. They reveal that TSM is an independent market, and regarding interactions with other countries, it is relatively stable compared to others. However, shocks from the Turkish stock market have a slight unidirectional impact on Iran, while the volatility of the TSM slightly affects Turkey and the United Arab Emirates.

Various studies have assessed the profitability of the TSM using neural network-based models and other advanced techniques. Katani et al. (2021) analyze TSM data with the imperialist competitive algorithm (ICA), ARMA, and GARCH models, incorporating features like moving averages and statistical tests. BELAGHİ et al. (2018) use non-parametric methods, such as the fuzzy Markov chain, and parametric methods, including GARCH and ARIMA-GARCH, to predict stock values. Nabipour et al. (2020) forecast trends in different sectors using decision trees, random forests, various boosting algorithms, RNN, and LSTM networks, with LSTM showing the best results. LSTM shows the best results, while the decision tree is the riskiest method. The study highlights that error values increase with longer prediction horizons and metal group predictions yield better results. Therefore, TSM studies indicate a need for more comprehensive research using advanced artificial intelligence and deep learning techniques.

Table 1 summarizes some essential studies on emerging literature about applying DRL models in various markets. The reviewed studies demonstrate the effectiveness of DRL models in time series forecasting, emphasizing the potential for further advancements in this area. DRLs, inspired by human learning through rewards and penalties, have shown successful outcomes in market prediction. However, emerging markets, such as the TSM, present unique challenges that require tailored approaches to address these limitations while leveraging advanced techniques.

Table 1
The summary of studying DRL and CNN methods for stock exchange research

Ref.	Method	Data & Features	Results
(Shetty & Tamane, 2025)	DQN with Boltzmann	Bitcoin	Improving the security and resilience of Bitcoin networks against evolving ransomware threats
(Song et al., 2025)	Portfolio optimization with RL	Different cryptocurrencies	Enabling dynamic adjustments of asset investment weights and ensuring the maximization of cumulative returns
(Qiu et al., 2024)	DRL and Quantum Finance Theory	Forex and U.S. stock market	Superior profitability, stability, and flexibility compared to RL systems in back-testing.
(Zhou et al., 2024)	Reward-driven DDQN	HIS, IXIC, S&P500, GOOGL, MSFT, INTC	Daily and weekly data enhance the performance of the agent.
(Huang et al., 2024)	Multi-Agent RL	U.S. stock indices	Employing intelligent agents enhances trading performance.
(Wang, 2024)	RL	Initial stock price, Initial shareholdings, Initial money holdings	Demonstrating the flexibility and robustness of the RL framework,
(Chen et al., 2024)	Composite Multi-Role-DQN for multi-scale data analysis	Dow Jones Industrial Average, the CAC 40 Index from France, the Nasdaq Composite Index, Nikkei, Ping An Bank, and Ziguang Co., Ltd.	Underscoring the efficiency and effectiveness of DQN in handling multi-scale time series data and optimizing stock market decisions
(Vetrina & Kobergb, 2024)	comparing various combinations of DRLs to evaluate robustness and effectiveness	Binance exchange	Proposing RL methods for static algorithm performance
(Vergara & Kristjanpoller, 2024)	Combining classical statistical arbitrage theory with the DRL frameworks	Bitcoin, Litecoin, Solana	Positive returns and outperforming the market
(Zhong et al., 2025)	RL	Stock indices of the United States and China	Offering enhanced adaptability and robustness RLs in the face of market volatility
(Papageorgiou et al., 2024)	DDQN	NVIDIA's short-term stock movements	Reveals the impact of combining diverse data sources to enhance the effectiveness of algorithmic trading strategies
(Du & Shen, 2024)	Integrating DRL with Sentiment Analysis	Chinese stock market, with data from 90 stocks, ranked in the top 150 for 2022-2023	Achieving the highest total assets on test datasets
(Cui et al., 2023)	A CNN based on DDQN	Dow Jones (DJI), Apple (AAPL), and General Electric (GE)	Utilizing CNN architectures within the DQN algorithm can lead to highly accurate predictions.
(Lu et al., 2022)	Deep Q-network structural break-aware deep Q-network (SADQN)	Taiwan stock market datasets	Proposing a transaction cost-aware objective function and risk-aware tools for states and rewards for the DQN algorithm.
(Ma & Yan, 2022)	CNN	Chinese Stock Market 27 technical indicators, five original price series, and trading volume data of the CSI 300 index setting social media sentiment	Recommending technical indicators within the original price series and CNN.
(Sun et al., 2022)	RL and LSTM	A six-month data set, including ten stocks from the Shanghai Stock Exchange	The results of RL models are more accurate than those of traditional models.

Table 1
The summary of studying DRL and CNN methods for stock exchange research (Continued)

(Thakkar & Chaudhari, 2021b)	Nine DL and DRL models	Forecasting the trend Interdependent features	DRL is more accurate than other models. Using independent features is effective.
(Carta et al., 2021)	DQN	OHLC prices, Sortino ratio, Maximum drawdown, Coverage, The equity curve, S&P 500	Using price factors along with other indicators presents more accurate results.
(Shi et al., 2021)	RL	OHLC prices, Volume, Dow Jones, S&P 500, NASDAQ, Shanghai-Shenzhen index, CSI 500 quality and growth index	Using volume along with price factors is efficacious in improving the result.
(Théate & Ernst, 2021)	TDQN	Sharpe ratio	The DRL approach can detect significant trends but hesitates to change market behavior during increased volatility.
(Jiang, 2021)	Reviewing stock market studies	Various features	More studies are based on stock price records; in the second place, they focus on price records and technical analysis.
(Chakole et al., 2021)	DRL, especially DQN	Indian and US stock market	The trading approach is only sometimes profitable for some stocks.
(Katani et al., 2021)	DL using Imperialist Competitive Algorithm (ICA), Autoregressive Moving-Average model (ARMA), and GARCH	TSM, 2009-2014 average, maximum price, minimum price, skewness, Kurtosis, and normality of the JACQUE-Bera	DRLs are suitable for TSM.
(Wu et al., 2021)	Portfolio management using DRL	different price scales for each stock and sharp ratio	CNN alone does not lead to the desired results, and it is necessary to use it in fusion.
(Polamuri et al., 2020)	Multi-model Based on Linearity and nonlinearity	Stock Market data set	Using nonlinear systems for predicting leads to more accurate results
(Nabipour et al., 2020)	Comparing various MLs with LSTM	TSM-oil groups, non-metallic minerals, and fundamental metals.	LSTM is the best result, and the decision tree is the riskiest method.
(Li et al., 2020; Yang et al., 2020)	Comparing traditional models with DRL models, including DQN, DDQN, DDDQN	30 stocks from Dow Jones data	DQN presents a more suitable performance in decision-making. DRL models are most similar to human learning. The slow convergence speed is the challenge that this method faces.
(Lee & Kim, 2020)	A new regularization method for CNN	S & P 500, KOSPI200 and FTSE100	Neural networks perform better than the buy-and-hold approach.
(Xiong et al., 2018)	RL	Dow Jones Index and Minimum Variance Portfolio Allocation approach in 30 major market symbols based on daily price	The DRL approach is a more suitable performance to balance risk and return.
(Deng et al., 2016)	DRL	Automatically calculating features	Employing all technical indicators slows the analysis and makes calculations redundant.

3. Data and features

As the CRISP model, our proposed method indicates the direction in five steps: data receiving and preparation, classifying, modeling, forecasting, evaluating, and deploying the trend. Therefore, we explain our data and selected features in the first and second parts. Third, we will introduce our proposed methods to achieve the result.

3.1 Data description

In recent years, TSM has drawn significant attention from investors seeking higher profits despite its smaller size and the limited availability of reliable historical data (Moghadam et al., 2019).

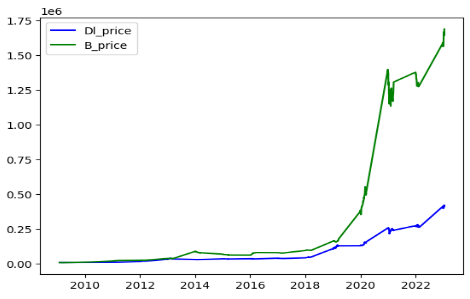


Fig. 1. The movement of the TSM index (green) and the Rial value of the dollar (blue) in the last 14 years (Bourseview, 2023)

Fig. 1 illustrates the performance of the TSM index (green graph) compared to the dollar value (blue graph) over the past 14 years. During this period, the dollar value of TSM increased 4.7 times (Bourseview, 2023), whereas the New York Stock Exchange experienced a 2.8-fold increase (Yahoo Finance, 2023). This graph demonstrates that investing in the TSM is both profitable and appealing. This study uses TSM as the target market and examines liquid trading symbols from 2011 to 2021. The proposed model analyzes TSM trading symbols from 2011 to 2020, with the testing period from January 28, 2021, to July 3, 2021.

3.2 Employed features

Emerging markets, such as the Tehran Stock Market (TSM), present unique challenges, including limited historical data, lower liquidity, and higher volatility. To address these challenges, it is crucial to utilize relevant features. In the TSM, the buying and selling queues play a vital role in forecasting accuracy, with liquidity emerging as a key factor. High-liquidity stocks are typically prioritized, and Amihud's liquidity measure is widely recognized as a standard metric for evaluating illiquidity in financial literature (Amihud, 2002). In this context, we selected the 22 most liquid symbols out of 700 in the TSM.

Additionally, this study employs a set of technical indicators as input features. These features are specifically crafted to capture various aspects of market behavior and volatility. As a result, 12 features, including seven listed in Table 2, are incorporated alongside OHLCV data.

By combining these features, the model provides a comprehensive view of the stock market, addressing both short-term price fluctuations and long-term trends, thereby enhancing its predictive accuracy.

Table 2
Employed features (Investopedia, 2021; Jansen, 2020)

Function	**Definitions**	
$ILLIQ = \frac{1}{N}\sum_{t=1}^{T}\frac{\lvert r_t\rvert}{\$V_t}$	$ILLIQ$: Amihod's illiquidity criterion T: The number of days V_t: The daily value of trades r_t: The return on investment on day t	(1)
$Return = \frac{closing\ price}{previous\ day\ closing\ price} - 1$	Return: Return on investment compared to the previous day	(2)
$RSI = 100 - \frac{100}{1+\frac{Average\ gain}{Average\ loss}}$	The average gain or loss used in this calculation is the average percentage gain or loss during a look-back period.	(3)
$MACD = EMA12 – EMA\ 9$	MACD is subtracting the long-term EMA and the short-term. An EMA is a moving average with greater weight and significance on the most recent data points.	(4)
$ATR = \frac{1}{n} * \sum_{i}^{n} TR_i$	TR_i: Particular true range n: Number of periods	(5)
$K\% = \frac{(Current\ Close - Lowest\ Low)}{(Highest\ High - Lowest\ Low)} * 100$	Stochastic: A momentum indicator comparing a particular closing price to a range of prices over a certain period.	(6)
$ULTOSC_t = 100 * \frac{4Avg_t(4)+2Avg_t(7)+Avg_t(28)}{4+2+1}$ $Avg_t(T) = \frac{\sum_{i=0}^{T-1} BP_{t-i}}{\sum_{i=0}^{T-1} TR_{t-i}}$ $BP_{t-i} = P_t^{Close} - min(P_{t-1}^{Close}, P_t^{Low})$ $TR_t = max(P_{t-1}^{Close}, P_t^{High}) - min(P_{t-1}^{Close}, P_t^{Low})$	ULTOSC: Measures the average difference between the current close and the previous lowest price over three timeframes. BP_t: The buying pressures TR_t: The true range	(7)

4. Methodology

This paper proposes a DRL model for predicting the profit of TSM, a profitable but relatively emerging market. The model utilizes the DDQN algorithm and employs two CNNs to learn the Q-values associated with different actions in the given environment. The CNNs process the dataset from the environment to generate Q-values for each action. This dual-network setup helps mitigate the overestimation bias of Q-values, ultimately enhancing the model's performance. The architecture of the proposed model is illustrated in Fig. 2 and Fig. 3.

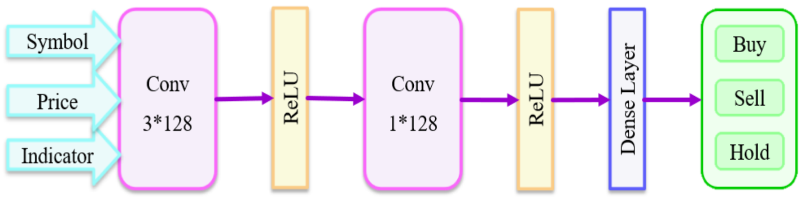


Fig. 2. Proposed convolution architecture

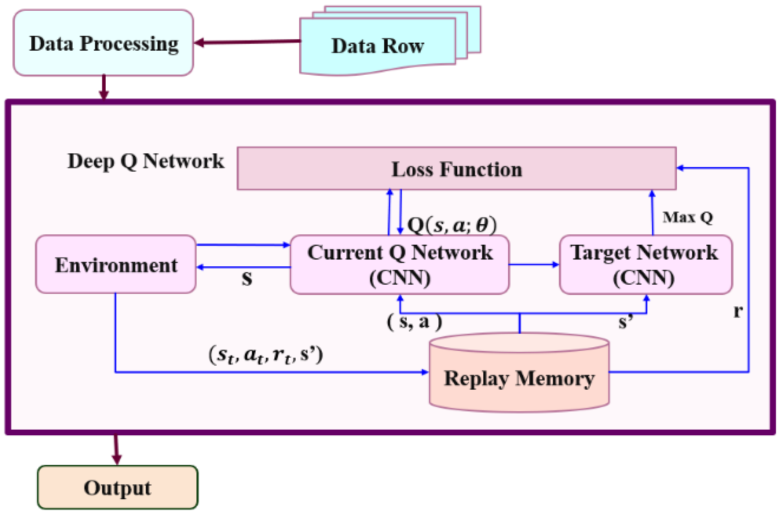


Fig. 3. Proposed model of DRL to predict the TSM

4.1 CNN for Feature Extraction

CNNs are widely employed in image processing and time series forecasting (Chen et al., 2022; Qin et al., 2018). By leveraging weight sharing, CNNs reduce the number of parameters and enhance learning outcomes (Lu et al., 2021).

A typical CNN architecture consists of three key layers: Convolutional layers extract features from input data, pooling layers reduce spatial dimensions to enhance computational efficiency and generalization, and fully connected layers consolidate features and generate the final output, as follows (Goodfellow et al., 2016):

$$S(i) = (I * K)(i) = \sum_{m} I(i-m)\, K(m) \tag{8}$$

where $S(i)$ denotes the result of the convolution, and shows the feature map at position i. $I(i-m)$ is the input data (stock symbols, prices, indicators) at the position $i-m$. $K(m)$, convolutional kernels weight at position m, detect patterns in the input data. This function extracts local features by applying kernels to input data, capturing time-sequential relationships in the stock market.

CNNs are especially effective for sequential data, preserving the input structure and capturing neighborhood relationships. The shared kernel weights across all input data reduce the number of parameters, minimizing overfitting. Pooling layers reduce the feature space, improving the model's generalization capabilities (Jansen, 2020). The CNN architecture is integrated into the DDQN framework to predict TSM profitability. The CNN processes input features and their relationships to generate Q-values for possible actions (buy, sell, hold) (Hao & Gao, 2020).

The cost function used in this architecture is the sum of squared errors. It measures the difference between actual and predicted outputs and is minimized during optimization.

A detailed explanation of each layer and its role in the architecture is as follows:

1. Input Features: The input includes stock symbols, historical prices, and financial indicators. These inputs provide essential information for the model to analyze market conditions.

2.First Convolutional Layer (Conv 3 × 128): This layer applies 128 convolutional filters with a kernel size of 3 × 1 to extract local features from the input data.

3.ReLU Activation: The Rectified Linear Unit (ReLU) activation function introduces nonlinearity, allowing the model to capture complex relationships (Agarap, 2018).

$$f(x) = ReLU(x) = Max(0, x) \tag{9}$$

where $f(x)$ is the activation function, $Max(0, x)$ represents the $ReLU$ activation function, that outputs x for positive values and 0 for negative values.

4.Second Convolutional Layer (Conv 1 × 128): This layer applies 128 convolutional filters of size 1 × 1 to abstract further the features extracted from the previous layer, followed by another ReLU activation for non-linearity.

5.Fully Connected Layer: The dense layer integrates the features learned by the convolutional layers and generates a final output vector representing the Q-values for the three possible actions (buy, sell, hold). This layer connects all neurons, consolidating the extracted features into actionable outputs (LeCun et al., 2015).

$$Q(s.a;\, \theta) = f(W.h + b) \tag{10}$$

where $Q(s.a;\, \theta)$ denotes Q-values for actions (buy, sell, hold), W is the weight matrix of the dense layer, h is the feature vector from the previous layer, and b shows the bias term.

6.The output layer maps the Q-values generated by the dense layer to their corresponding actions (buy, sell, hold). These Q-values represent the expected future rewards for each action, enabling the model to make informed trading decisions based on the current market state.

This dual-network approach, where one CNN selects the action with the highest Q-value and the other evaluates that action's Q-value, reduces overestimation bias in Q-values. This integration of CNNs ensures more accurate, stable predictions, leading to improved trading decisions in the TSM.

4.2 Integration with DDQN

DDQNs effectively address the overestimation bias in Q-values inherent in standard DQNs (Van Hasselt et al., 2016). By utilizing two separate networks, a primary Q-network for selecting actions and a target Q-network for evaluating them, DDQNs achieve more stability and accuracy in RL tasks. Integrating this framework with the CNN structure described in Section 4.1 allows the model to combine feature extraction and decision-making for financial forecasting effectively. The step-by-step integration is as follows:

1. Q-Value Calculation: The Q-value $Q\,(s.a;\, \theta)$ quantifies the expected reward for taking action a in state s as computed by the CNN layers and dense network (Mnih et al., 2015). The resulting Q-values guide actions such as buy, sell, or hold.
2. Bellman Equation for Target Q-Value: The Bellman equation estimates the target Q-value y_i balancing immediate and future rewards to enable long-term optimization (Van Hasselt et al., 2016):

$$y_i = \mathbb{E}\left[r + \gamma \max_{a'} Q\,(s'.a'; \hat{\theta}|s.a)\right] \tag{11}$$

where y_i represent the target value, r is the immediate reward, γ is the discount factor that prioritizes future rewards, $s'$is the next state, and θ' is the parameter of the target network.

3. Loss Function: The loss function minimizes the $Q\,(s.a;\, \theta)$ and the target value y_i, driving convergence over iterations (Mnih et al., 2016):

$$\mathrm{L_i(\theta_i) = (y_i - Q\,(s.a;\, \theta))^2} \tag{12}$$

4. Experience Replay: To enhance learning stability, the DDQN leverages experience replay by storing transitions (s_t, a_t, r_t, s_{t+1}) in a replay memory D, from which random batches are sampled for training (Lin, 1993). This approach reduces temporal correlations in training data and improves generalization.

$$D = \{(s_t, a_t, r_t, s_{t+1})|t \in Training\ Episods\} \tag{13}$$

5.Target Network Update: The parameters of the target network are periodically updated to stabilize training by ensuring the Q-values used for evaluation remain consistent over training iterations (Van Hasselt et al., 2016).

$$\acute{\theta} = \theta \tag{14}$$

where $\acute{\theta}$ and θ are parameters of the target and primary Q-network, respectively.

During training, CNN parameters are optimized to minimize the loss function. In the prediction phase, the trained CNN evaluates the current market state and generates Q-values for actions (buy, sell, hold). The action with the highest Q-value is selected for trading decisions. The dual-network approach integrates CNNs into the DDQN framework, reducing overestimation bias in Q-values and improving decision-making and trading performance in financial markets.

4.3 Algorithm design

The following pseudocode represents the workflow of the proposed DDQN model for predicting profitability in the TSM in six steps. The structure includes data pre-processing, environment setup, DDQN initialization, and the training loop with experience replay and target network updates.

Algorithm 1. Pseudocode for the proposed framework

```
    Inputs: trading days, trading cost, time cost
    Outputs: Predicted Next prices
Start:
    # Step 1:  Install Libraries:
        Gym; Talib; matplotlib; numpy; pandas
    #Step 2:  Initialize hyperparameters:
        $\gamma$; $\tau$; learning_rate; replay_capacity; batch_size; epsilon_start; epsilon_end; epsilon_exponential_decay
    # Step 3: Preparing Data
          1: Price_normal_data, Tec_normal_data= MinMax_Normalisation (Price_data, Tec_data)
          2: Price _Vectors = Price_multi_step_division (Price_normal_data)
          3: Tec_Vectors = Tec_multi_step_division (Tec_normal_data)
    # Step 4: Set up Gym Environment
        4: trading_days, trading_cost_bps, time_cost_bps; trading_env; max_episode_steps
    # Step 5: Frame Sequence Initialization and DDQN Update Process
        5: Initialize: Create an empty frame sequence $x \leftarrow ()$
        6: for each time step $t \in \{0,1,2,\dots\}$:
            Set the current state. $s \leftarrow x$
            Select an action a using the behavior policy $\pi_\beta$
            Execute action a in the environment, resulting in:
              A new frame $x^t$ sampled from the environment $\varepsilon$ given $(s, a)$
              A reward r
              An updated status indicating whether the episode is complete
                Append $x^t$ to x
            Maintain sequence length: If $|x| > N_f$, delete the oldest frame $x_{t_{min}}$ from x.
        7: Store Transition:
            Set the next state $s' \leftarrow x$.
            Add a transition tuple $(s, a, r, s')$ to the replay memory D, Replacing the oldest tuple if $|D| > N_r$
        8: Sample and Update:
            Define $a^{max}(s';\theta) = \arg\max_{a'} Q(s', a'; \theta)$
            Set target $y_j = \begin{cases} r & \text{if } s' \text{ is terminal} \\ r + \gamma Q(s', a^{max}(s';\theta); \theta^-), & \text{otherwise} \end{cases}$
        9: Gradient Descent:
            Perform a gradient descent step to minimize the loss $\| y_j - Q(s, a; \theta) \|^2$
        10: Target Network Update:
            Every $N^-$ steps, update the target parameters by setting $\theta^- \leftarrow \theta$
    # Step 6: Plot price data over time with buy/hold/sell actions marked and Return Predicted following prices
End
```

5. Implementation

The proposed model's implementation leverages Python and the TA-Lib library. TA-Lib provides over 150 technical indicators commonly utilized by commercial software developers for time series analysis. The model employs a three-layer CNN as the core architecture for Q-learning, as outlined in Table 3.

The CNN processes price data and selected indicators through the first convolutional layer. This one-dimensional convolution examines data across working days, prevents overfitting, and generates 128 outputs per layer. The model comprises 35,842 trainable parameters, where weak neurons are pruned, and essential neurons are retained for optimized learning. The fourth layer, a fully connected dense network, integrates the learned features to provide the final output. This output represents two primary actions, buy and hold, across 258 parameters.

Table 3
CNN structure of the model

Layer (type)	Number of Parameters	Output shape
Dense-1	2560	128
Dense-2	16512	128
Dense-3	16512	128
Output	258	2
Total Parameters	35842	
Trainable Parameters	35842	

As illustrated in Fig. 4, the model's learning process begins with small rewards. Over time, the predicted outcomes align closely with the performance of the buy-and-hold market index strategy and eventually surpass it. The training framework consists of three main components: data preparation, data collection, and simulation performance.

The trading environment is implemented using GYM ENV[1], defines three critical aspects:

1. The cost of transactions.
2. Three primary actions: buy, hold, and sell.
3. A multidimensional observation space encompassing minimum and maximum values.

In the proposed model, a trader agent is responsible for decision-making and is guided by the DDQN architecture. The agent continuously monitors the environment, processes convolutional observations, and makes informed buying, selling, or holding decisions. Rewards or penalties are applied proportionally based on the profitability of the chosen actions. The forecast results improve as training progresses, gradually surpassing the market index and buy-and-hold strategy. The action yielding the highest reward is saved for the subsequent day's decision-making process. The simulation runs for one year, randomly selecting the starting point to ensure robustness. The learning rate is initialized at $0.1 \times \gamma$, where γ serves as a loss factor coefficient, controlling the accuracy of rewards. The model refines its learning based on time steps and the delay associated with achieving optimal answers.

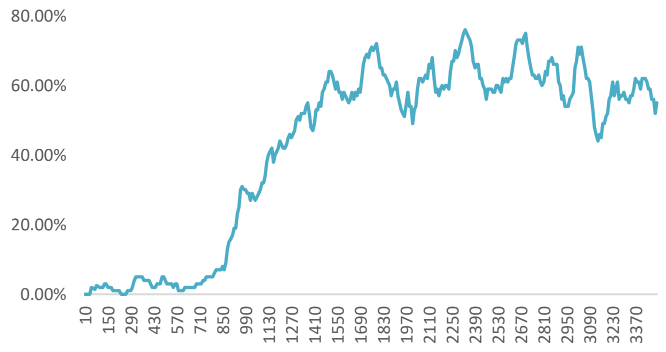


Fig. 4. The process of winning profitable investments in the learning step of the intelligent agent

1 OpenAI gym is an environment for developing and testing learning agents.

For optimization, the model employs the Sum of Squares of Errors (SSE) as the cost function, which is minimized to enhance accuracy. The SSE quantifies the discrepancy between observed and predicted values, as defined by (Montgomery et al., 2021).

$$SSE = \sum_i (x_i - \hat{x_i})^2 \tag{15}$$

where x_i represents the observed values, and, $\hat{x_i}$ denotes the predicted values. This approach ensures that the model effectively aligns its predictions with actual market outcomes.

6. Experiment Results

The proposed model analyzes 15 years of stock market data using 5-day time intervals. A convolution process progressively reduces the output volume over time. The model operates on input data spanning five days, each containing 19 features, and compresses this information to produce a single day's output. Due to the triple convolution process, two features are eliminated at each step. A total of 128 kernels are employed for feature extraction, playing a vital role in enhancing the accuracy of market forecasting.

As illustrated in Fig. 5, annual returns during the agent's learning period are compared to the buy-and-hold strategy. The agent's performance, shown in blue, begins below the market performance (orange) but eventually surpasses it. Fig. 6 highlights the differences in monthly returns, where the agent transitions from learning to optimizing investment outcomes, outperforming the market after the 1,000th iteration. Fig. 6 illustrates the difference between the neural network's output and the market performance.

Table 4 and Fig. 7 compare the agent's profit and loss against the buy-and-hold approach. For instance, in the case of "IKCO stocks", the agent achieves a profit of 41.26, while the buy-and-hold strategy results in a loss of -25.57. The testing period, from January 28, 2021, to July 3, 2021, shows a 2.2% return on the market index (Fig. 8.a). However, the proposed system yields a profit of 31.77%, as seen in Fig. 8. b. This figure also depicts prediction errors: green descending and red ascending segments reflect mispredictions while matching color trends represent correct forecasts.

The experimental results demonstrate the proposed model's superior performance over traditional market strategies. The intelligent agent significantly improves investment returns and delivers robust market forecasting solutions by leveraging advanced convolutional techniques and reinforcement learning.

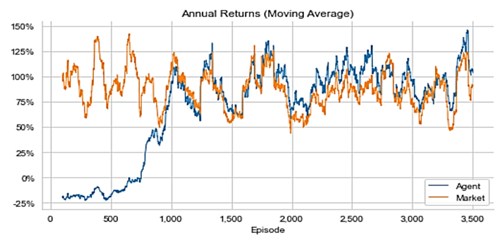


Fig. 5. Annual return on investment in the learning step of the agent

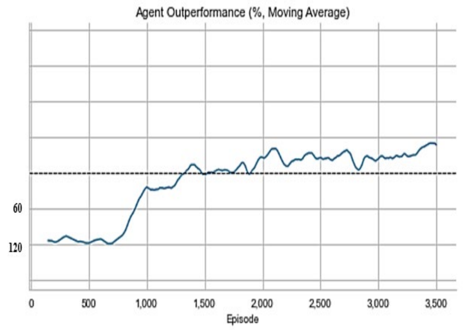


Fig. 6. The difference in monthly return on investment in the learning step of the agent and market

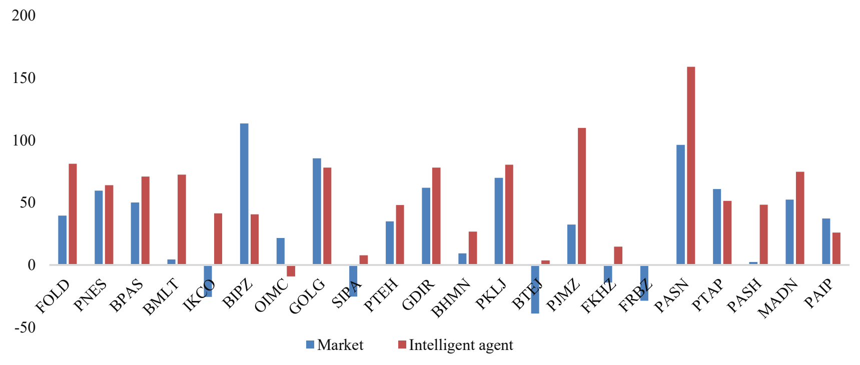


Fig. 7. Comparing the performance of the proposed DRL model and the market in the test step

Table 4
Comparing the performance of the proposed model and the market

	Company Name	Market Profits	Agent Profit	Difference between the two strategies
1	Mobarakeh Steel	39.70	81.16	41.47
2	Esfahan Oil Refiner company	59.57	63.88	4.30
3	Pasargad Bank	50.11	70.92	20.81
4	Mellat Bank	4.34	72.48	68.14
5	Iran Khodro	-25.57	41.26	66.84
6	Pasargad Insurance	113.32	40.53	-72.80
7	Omid Inv. Mng.	21.65	-9.20	-30.85
8	Gol-E-Gohar.	85.37	78.06	-7.32
9	Saipa	-25.34	7.83	33.17
10	Palayesh Tehran	34.92	47.98	13.06
11	Ghadir Inv.	61.83	77.91	16.08
12	Bahman Group	9.42	26.67	17.26
13	Khalij Fars	69.88	80.44	10.55
14	Tejarat Bank	-38.82	3.66	42.47
15	Jam Petr.	32.29	109.82	77.53
16	Khouz. Steel	-14.05	14.73	28.78
17	Iran Fara Bourse	-28.66	-0.52	28.14
18	Parsian Oil&Gas	96.31	158.91	62.60
19	Tamin Petro.	60.78	51.44	-9.34
20	Paksho	2.30	48.25	45.95
21	Metals & Min.	52.49	74.82	22.33
22	Pars Aryan Co.	37.17	26.00	-11.17

Note: The buy-and-hold approach, namely the market in the table, means buying the same share of the 22 symbols mentioned equally at the beginning of the test period and selling them at the end, from January 28, 2021, to July 3, 2021. If the agent gets more profit than the buy-and-hold approach, the profit gained is better than the stock market index and bank deposit.

6.1 Evaluation of Forecasting Performance

Two methodologies were applied to evaluate forecasting performance: comparison with the buy-and-hold approach and direction accuracy (DA) evaluation.

The buy-and-hold strategy is a baseline for assessing the DDQN model's profitability. The results reveal that the proposed system achieves a 53% gain compared to the buy-and-hold strategy's 31.77%, demonstrating a 67% improvement. This highlights the model's efficiency in analyzing equity markets and identifying profit trends.

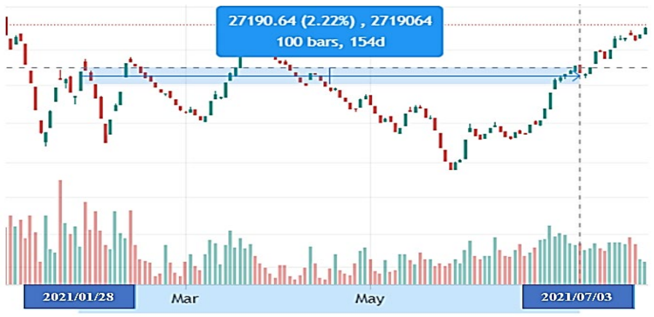
DA evaluates the model's ability to predict market movement directions (Muntean & Militaru, 2023). This metric measures the proportion of correctly predicted directional changes in stock prices. The proposed model achieves a DA of 67%, significantly higher than the average 45.5% accuracy of six comparative models (Table 5).

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} * 100 \tag{16}$$

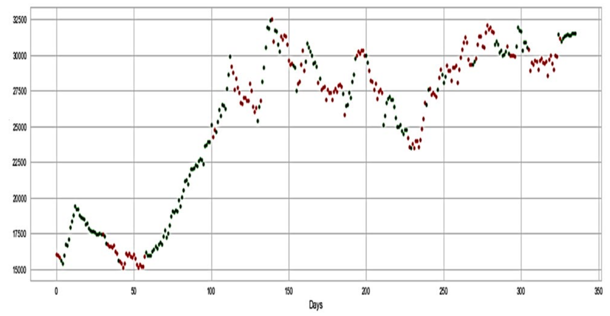
Table 5
The performance evaluation in comparison with six other models

	Models	References	DA%
1	CNN–LSTM	(Livieris et al., 2020)	46
2	LSTM-Attention	(Abbasimehr & Paki, 2022)	47
3	Linear Transformer	(Xu et al., 2023)	46
4	LSTM-Based Transformer	(Yue & Ma, 2023)	45
5	Deep CNN	(Semenoglou et al., 2023)	44
6	TSF-transformer	(Li et al., 2023)	45
7	Average Improve	-	45.5
8	DDQN	Proposed Model	67

Table 5 displays experimental evaluations of the proposed model against six state-of-the-art methods using TSM features and price datasets. The results highlight an average DA improvement of 22.5%, demonstrating the proposed framework's robustness. The findings validate that the DDQN model achieves superior forecasting performance compared to existing methods, achieving a remarkable accuracy of 67%.



(a) Market index during the test step (rahavard, 2021)



(b) The intelligent agent's prediction for return on investment during the testing phase

Fig. 8. Market index in comparison with the intelligent trader in the test period from January to July 2021

7. Discussion

The proposed model leverages the DDQN method to predict stock market trends and optimize trading strategies, focusing on liquid shares using data from 2011 to 2022. The dataset, sourced from BourseView, underwent meticulous preprocessing to ensure accuracy, with the analysis centered on a 22-symbol portfolio from January 28, 2021, to July 3, 2021. This study incorporated a diverse range of features, including short-term (5-day), medium-term (20-day), and long-term (60-day) returns, RSI, MACD, ATR, stochastic indicators, and OHLCV prices, emphasizing the use of non-overlapping feature sets to ensure robustness. The model's core is a CNN-based architecture, replacing traditional Q-tables to address overfitting risks and the challenges associated with limited training data. Pooling layers were employed to reduce feature dimensionality, enhancing the model's computational efficiency. Two evaluation metrics were utilized: comparing the traditional buy-and-hold strategy and assessing directional accuracy. The comparative analysis revealed that the proposed model consistently outperformed the buy-and-hold strategy, achieving greater profitability across varying market conditions. During the testing phase, the DDQN model demonstrated substantial gains, surpassing the performance of the stock market index and conventional bank deposits. These results validate the model's effectiveness, even in volatile market environments. The evaluation of directional accuracy further underscored the model's robustness, showcasing a marked improvement in predicting market trends compared to existing methodologies. The optimization process minimizes the sum of squared errors and highlights the model's reliability in aligning predictions with actual outcomes. This demonstrates the proposed approach's capability to serve as a dependable tool for investment decision-making, providing a clear advantage in navigating complex financial markets.

8. Conclusion

Forecasting stock prices are of immense significance, offering substantial rewards and enabling investors to make well-informed decisions. However, the task is inherently challenging due to the stationary, noisy, and chaotic nature of stock market data, which complicates confident investment strategies. These challenges are especially pronounced in emerging markets like the Tehran Stock Market (TSM), which presents unique opportunities and risks. Despite its short operational history and identified limitations, including liquidity risks, emotional-driven overvaluation, and currency volatility, the TSM remains an attractive investment avenue with significant potential for profitability. The rapid advancements in deep learning and its application in financial forecasting have paved the way for more accurate predictions and enhanced returns. Building upon these developments, this study introduces a deep reinforcement learning approach specifically tailored to address the unique characteristics of the TSM. The proposed model, which combines advanced DRL techniques with innovative feature extraction methods, is a reliable, data-driven strategy for navigating the complexities of dynamic financial markets. The results affirm its capability as a trustworthy, data-driven strategy for navigating the complexities of dynamic financial markets. The proposed model represents a substantial advancement in stock market forecasting. Integrating state-of-the-art DRL techniques and CNN-based architectures offers investors a robust tool for navigating volatile market conditions and achieving superior investment outcomes. The findings highlight the model's potential to enhance profitability and reduce risks, making it a compelling solution for addressing the challenges inherent in financial decision-making.

References

Abbasimehr, H., & Paki, R. (2022). Improving time series forecasting using LSTM and attention models. *Journal of Ambient Intelligence and Humanized Computing, 13*(1), 673-691.

Abounoori, E., & Tour, M. (2019). Stock market interactions among Iran, USA, Turkey, and UAE. *Physica A: Statistical mechanics and its applications, 524*, 297-305.

Agarap, A. F. (2018). Deep learning using rectified linear units (ReLU). arXiv preprint arXiv:1803.08375.

Amihud, Y. (2002). Illiquidity and stock returns: cross-section and time-series effects. *Journal of financial markets, 5*(1), 31-56.

Belaghi, R. A., Aminnejad, M., & Alma, Ö. G. (2018). Stock market prediction using nonparametric fuzzy and parametric garch methods. *Turkish Journal of Forecasting, 2*(1), 1-8.

Bourseview. (2023). Iran Stock market comparisons. https://www.bourseview.com/

Carta, S., Corriga, A., Ferreira, A., Podda, A. S., & Recupero, D. R. (2021). A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. *Applied Intelligence, 51,* 889-905.

Chakole, J. B., Kolhe, M. S., Mahapurush, G. D., Yadav, A., & Kurhekar, M. P. (2021). A Q-learning agent for automated trading in equity stock markets. *Expert Systems with Applications, 163*, 113761.

Chen, X., Wang, Q., Hu, C., & Wang, C. (2024). A stock market decision-making framework based on CMR-DQN. *Applied Sciences, 14*(16), 6881.

Chen, Y., Li, L., Li, W., Guo, Q., Du, Z., & Xu, Z. (2022). AI Computing Systems: An Application Driven Perspective. Elsevier.

Cui, K., Hao, R., Huang, Y., Li, J., & Song, Y. (2023). A novel convolutional neural networks for stock trading based on DDQN algorithm. *IEEE Access, 11*, 32308-32318.

Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2016). Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems, 28*(3), 653-664.

Dong, H., Dong, H., Ding, Z., Zhang, S., & Chang, T. (2020). *Deep Reinforcement Learning*. Singapore: Springer Singapore.

Du, S., & Shen, H. (2024). A Stock Prediction Method Based on Deep Reinforcement Learning and Sentiment Analysis. *Applied Sciences, 14*(19), 8747.

Goodfellow, I. (2016). *Deep learning (Adaptive Computation and Machine Learning series).* The MIT Press.

Hao, Y., & Gao, Q. (2020). Predicting the trend of stock market index using the hybrid neural network based on multiple time scale feature learning. *Applied Sciences, 10*(11), 3961.

Hu, Z., Zhao, Y., & Khushi, M. (2021). A survey of forex and stock price prediction using deep learning. *Applied System Innovation, 4*(1), 9.

Huang, Y., Zhou, C., Cui, K., & Lu, X. (2024). A multi-agent reinforcement learning framework for optimizing financial trading strategies based on timesnet. *Expert Systems with Applications, 237*, 121502.

Investopedia. (2021). www.investopedia.com

Jansen, S. (2020). Machine Learning for Algorithmic Trading: Predictive models to extract signals from market and alternative data for systematic trading strategies with Python. Packt Publishing Ltd.

Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications, 184*, 115537.

Katani, S., Samadi, F., & Hajiha, Z. (2017). Optimization of multi-objective portfolio using imperialist competitive algorithm in Tehran Stock Exchange. *International Journal of Business Management, 2*(4-2017), 85-97.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

Lee, S. W., & Kim, H. Y. (2020). Stock market forecasting with super-high dimensional time-series data using ConvLSTM, trend sampling, and specialized data augmentation. *Expert systems with applications, 161*, 113704.

Li, Y., Ni, P., & Chang, V. (2020). Application of deep reinforcement learning in stock trading strategies and stock forecasting. *Computing, 102*(6), 1305-1322.

Li, Z., Zhang, X., & Dong, Z. (2023). TSF-transformer: a time series forecasting model for exhaust gas emission using transformer. *Applied Intelligence, 53*(13), 17211-17225.

Lin, L. J. (1992). *Reinforcement learning for robots using neural networks*. Carnegie Mellon University.

Liu, X. Y., Xiong, Z., Zhong, S., Yang, H., & Walid, A. (2018). Practical deep reinforcement learning approach for stock trading. arXiv preprint arXiv:1811.07522.

Livieris, I. E., Pintelas, E., & Pintelas, P. (2020). A CNN–LSTM model for gold price time-series forecasting. *Neural computing and applications, 32,* 17351-17360.

Lu, J. Y., Lai, H. C., Shih, W. Y., Chen, Y. F., Huang, S. H., Chang, H. H., ... & Dai, T. S. (2022). Structural break-aware pairs trading strategy using deep reinforcement learning. *The Journal of Supercomputing, 78*(3), 3843-3882.

Lu, W., Li, J., Wang, J., & Qin, L. (2021). A CNN-BiLSTM-AM method for stock price prediction. *Neural Computing and Applications, 33*(10), 4741-4753.

Ma, C., & Yan, S. (2022). Deep learning in the Chinese stock market: the role of technical indicators. *Finance Research Letters, 49,* 103025.

Meng, T. L., & Khushi, M. (2019). Reinforcement learning in financial markets. *Data, 4*(3), 110.

Mnih, V. (2016). Asynchronous Methods for Deep Reinforcement Learning. arXiv preprint arXiv:1602.01783.

Moghadam, H. E., Mohammadi, T., Kashani, M. F., & Shakeri, A. (2019). Complex networks analysis in Iran stock market: The application of centrality. *Physica A: Statistical Mechanics and its Applications, 531*, 121800.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). *Introduction to linear regression analysis*. John Wiley & Sons.

Muntean, M., & Militaru, F. D. (2023, January). Metrics for evaluating classification algorithms. *In Education, Research and Business Technologies: Proceedings of 21st International Conference on Informatics in Economy (IE 2022)* (pp. 307-317). Singapore: Springer Nature Singapore.

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., & Salwana, E. (2020). Deep learning for stock market prediction. *Entropy, 22*(8), 840.

Nabipour, M., Nayyeri, P., Jabani, H., Shahab, S., & Mosavi, A. (2020). Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. *IEEE Access, 8*, 150199-150212.

Papageorgiou, G., Gkaimanis, D., & Tjortjis, C. (2024). Enhancing Stock Market Forecasts with Double Deep Q-Network in Volatile Stock Market Environments. *Electronics, 13*(9), 1629.

Polamuri, S. R., Srinivas, K., & Mohan, A. K. (2020). Multi model-based hybrid prediction algorithm (MM-HPA) for stock market prices prediction framework (SMPPF). *Arabian Journal for Science and Engineering, 45*(12), 10493-10509.

Qin, L., Yu, N., & Zhao, D. (2018). Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video. *Tehnički vjesnik, 25*(2), 528-535.

Qiu, Y., Liu, R., & Lee, R. S. (2024). The design and implementation of a deep reinforcement learning and quantum finance theory-inspired portfolio investment management system. *Expert Systems with Applications, 238*, 122243.

rahavard. (2021). https://rahavard365.com/

Semenoglou, A. A., Spiliotis, E., & Assimakopoulos, V. (2023). Image-based time series forecasting: A deep convolutional neural network approach. *Neural Networks, 157*, 39-53.

Sharma, A., Bhuriya, D., & Singh, U. (2017, April). Survey of stock market prediction using machine learning approach. *In 2017 International Conference of Electronics, communication and Aerospace Technology (ICECA) (Vol. 2, pp. 506-509). IEEE.*

Shetty, M., & Tamane, S. (2025). Unveiling bitcoin network attack using deep reinforcement learning with Boltzmann exploration. *Peer-to-Peer Networking and Applications, 18*(1), 1-19.

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

Song, G., Zhao, T., Ma, X., Lin, P., & Cui, C. (2025). Reinforcement learning-based portfolio optimization with deterministic state transition. *Information Sciences, 690*, 121538.

Sun, T., Huang, D., & Yu, J. (2022). Market making strategy optimization via deep reinforcement learning. *IEEE Access, 10*, 9085-9093.

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. *Expert Systems with Applications, 177,* 114800.

Thakkar, A., & Chaudhari, K. (2021). Fusion in stock market prediction: a decade survey on the necessity, recent developments, and potential future directions. *Information Fusion, 65*, 95-107.

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

Van Hasselt, H., Guez, A., & Silver, D. (2016, March). Deep reinforcement learning with double q-learning. *In Proceedings of the AAAI conference on artificial intelligence* (Vol. 30, No. 1).

Vergara, G., & Kristjanpoller, W. (2024). Deep reinforcement learning applied to statistical arbitrage investment strategy on cryptomarket. *Applied Soft Computing, 153*, 111255.

Vetrina, R. L., & Kobergb, K. (2024). Reinforcement learning in optimisation of financial market trading strategy parameters. *COMPUTER, 16*(7), 1793-1812.

Wang, L. (2024). Reinforcement Learning for Solving Financial Problems.

Wu, M. E., Syu, J. H., Lin, J. C. W., & Ho, J. M. (2021). Portfolio management system in equity market neutral using reinforcement learning. *Applied Intelligence, 51*(11), 8119-8131.

Xu, C., Li, J., Feng, B., & Lu, B. (2023). A financial time-series prediction model based on multiplex attention and linear transformer structure. *Applied Sciences, 13*(8), 5175.

Yahoofinance. (2023). NYSE COMPOSITE. https://finance.yahoo.com/quote/%5ENYA/chart

Yang, H., Liu, X. Y., Zhong, S., & Walid, A. (2020, October). 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).

Yue, M., & Ma, S. (2023). LSTM-based transformer for transfer passenger flow forecasting between transportation integrated hubs in urban agglomeration. *Applied Sciences, 13*(1), 637.

Zhang, Y., Zhao, P., Wu, Q., Li, B., Huang, J., & Tan, M. (2020). Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering, 34*(1), 236-248.

Zhong, X., Wei, J., Li, S., & Xu, Q. (2025). Deep reinforcement learning for dynamic strategy interchange in financial markets. *Applied Intelligence, 55*(1), 1-19.

Zhou, C., Huang, Y., Cui, K., & Lu, X. (2024). R-DDQN: Optimizing Algorithmic Trading Strategies Using a Reward Network in a Double DQN. *Mathematics, 12*(11), 1621.

© 2025 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).