Gazi AKADEMİK YAYINCILIK

AIS ARTIFICIAL INTELLIGENCE STUDIES

ARTIFICIAL INTELLIGENCE STUDIES

A Deep Ensemble Reinforcement Learning Based Approach For Stock Trading

Seyfullah Arslan [*a], **Durmuş Özdemir** [b]

ABSTRACT

Algorithmic trading, also known as quantitative trading, plays a significant role in the finance and technology industries. Replacing some analytical methods used by stock market investors for buying and selling transactions, algorithmic trading benefits from advancements in machine learning and deep learning, leveraging the ability of deep learning methods to extract meaning from complex data. In this study, an ensemble learning framework was combined with reinforcement learning to enhance decision-making accuracy and optimize trading strategies in dynamic financial environments. The trained reinforcement learning agent was tested on the 2011 data of the Standard & Poor's 500 (GSPC) index, achieving a net profit of $2,258.27. In the proposed structure, a Long Short-Term Memory (LSTM) agent processed time-series data, while a Convolutional Neural Network (CNN) agent used an image created from this data as input. The predictions obtained from these inputs were combined, and the final result was derived from a Deep Q Network (DQN) model, forming the ensemble learning structure. The results show that predictions made using the ensemble learning method generated higher profits than individual predictions by the agents and methods in similar studies in the literature.

[a,*] Kütahta Dumlıpınar University,
Engineering Faculty,
Dept. of Computer Engineering
43100 - Kütahya, Türkiye
ORCID: 0000-0002-2573-273X

[b] Kütahta Dumlıpınar University,
Engineering Faculty,
Dept. of Computer Engineering
43100 - Kütahya, Türkiye
ORCID: 0000-0002-9543-4076

* Corresponding author.
e-mail: seyfullah.arslan@dpu.edu.tr

Keywords: Reinforcement learning, ensemble learning, algorithmic trading, stock trading

Hisse Senedi Ticareti İçin Derin Takviyeli Topluluk Öğrenme Tabanlı Bir Yaklaşım

ÖZ

Finans ve teknoloji endüstrisi için algoritmik ticaret, bir diğer adıyla kantitatif ticaret önemli bir role sahiptir. Borsa yatırımcılarının alım ve satım işlemlerinde kullandığı bazı analiz yöntemlerinin yerini dolduran algoritmik ticaret, makine öğrenmesi ve derin öğrenme alanında yaşanan gelişmeler ile derin öğrenme yöntemlerinin karmaşık verilerden anlam çıkarma yeteneğinden faydalanmaktadır. Bu çalışmada, topluluk öğrenme çerçevesi, dinamik finansal ortamlarda karar verme doğruluğunu artırmak ve ticaret stratejilerini optimize etmek için takviyeli öğrenme ile birleştirilmiştir. Eğitimi yapılan pekiştirmeli öğrenme aracısı , Standard & Poor's 500 (GSPC) endeksinin 2011 yılına ait verisi üzerinde test edilerek 2258,27 dolar gibi bir kar oranı elde etmiştir. Önerilen yapıda bir Uzun Kısa Süreli Bellek (LSTM) aracısı zaman serisi verilerini, Konvolüsyonel Sinir Ağı (CNN) aracısı ise bu verilerle oluşturulan bir görüntüyü girdi olarak almıştır ve bu girdilerle elde edilen tahminler birleştirilerek Derin Q Ağı (DQN) modelinden nihai sonuç alınmıştır ve bu şekilde topluluk öğrenme yapısı elde edilmiştir. Elde edilen sonuçlar, topluluk öğrenme yöntemi ile yapılan tahminlerin, aracıların bireysel tahminlerinden ve literatürdeki benzer çalışmalardaki yöntemlerden daha yüksek kar getirdiğini göstermektedir.

Anahtar Kelimeler: Takviyeli öğrenme, topluluk öğrenmesi, algoritmik ticaret, hisse senedi ticareti

Submitted: 29.11.2024
Revised: 19.12.2024
Accepted: 20.12.2024

doi: 10.30855/AIS.2024.07.02.05

1. Introduction

The finance industry aims to generate profit by optimally allocating available resources. Stock markets serve as significant trading hubs for individuals seeking opportunities to maximize profit levels. Investors engage in stock trading based on their perceptions of the market; however, this trading approach is often inefficient. This inefficiency arises because extracting meaning from market data typically involves time-series data that is noisy, influenced by external factors, and fraught with uncertainty. It has been noted that stock prices are affected by factors such as political events, corporate policies, exchange rates, and even the psychology of investors [1].

In the past few decades, various methods have been proposed for developing trading strategies in markets. Fundamental analysis [2], technical analysis [3], and algorithmic trading [4] are among these methods. In technical analysis, future predictions are made using historical data and past relationships between the data, while predictions in fundamental analysis rely on factors such as financial statements, economic conditions, and international relations. However, fundamental analysis cannot provide real-time or short-term predictions [5]. In a significant study in the literature, Eugene Fama stated that analyses such as technical or fundamental analysis would not yield above-average profits for investors [6]. Recently, the highest interest has been directed toward algorithmic trading. This is evident from the fact that it accounts for approximately 75% of the trading volume in U.S. stock markets [4].

Algorithmic trading, also known as quantitative trading, plays a crucial role within the financial technology industry, commonly referred to as FinTech. FinTech, a rapidly growing and evolving industry, has a very simple purpose: to bring innovation to financial activities and enhance them by leveraging technology extensively. In the coming years, the FinTech industry is expected to offer revolutionary solutions to various decision-making problems in the financial sector, including trading, investment, portfolio management, fraud detection, financial consulting, and risk management [7]. Such decision-making problems are extremely challenging to solve, as they often have a sequential structure and are highly stochastic.

With significant advancements in the field of machine learning in recent years, complex tasks such as stock trading have become topics of interest within the domain of machine learning. This task fundamentally involves using machine learning models to replicate the decision-making process carried out by traders. The goal is to maximize profits from stock trading transactions based on the decisions made. In general, machine learning is proficient at identifying non-linear patterns in data. However, recent studies in the literature indicate that deep learning methods, due to their multi-layered network structures, demonstrate better performance in extracting critical information from non-linear time-series data and are thus preferred for time-series forecasting. Research has shown that deep learning models can outperform market averages or achieve significantly high returns [8].

However, in addition to basic machine learning models, deep learning models such as feedforward neural networks (FFNN) and recurrent neural networks are inadequate when dealing with unstable time-series and long-term autoregression [9]. In the literature, reinforcement learning methods are commonly employed to provide efficient solutions to such problems. Reinforcement learning is a distinct subfield of machine learning, separate from supervised and unsupervised learning methods. It relies on a dynamic decision-making approach, where an agent interacts with unknown environments and learns from these interactions to make informed decisions [10].

In recent years, reinforcement learning (RL), whose areas of application have expanded, has become one of the algorithmic trading methods used for the challenging task of automated stock trading [11-16]. Despite still being in the very early stages of development, Deep Reinforcement Learning (DRL) is noted to have the potential to rival professional traders in stock trading [17]. Beyond stock markets, DRL applications are also being employed in Continuous Intraday Markets (CID), which share similarities with stock exchanges [18].

The primary aim of this study is to propose a deep learning algorithm capable of competing with existing algorithmic trading strategies. The study utilizes the Deep Q-Learning method, a combination of deep learning and reinforcement learning techniques, specifically Q-Learning. In addition to the DQN model constructed using a classical artificial neural network to provide the final decision, the proposed framework includes two auxiliary DQN models built with CNN and LSTM architectures to generate supporting decisions. These three distinct DQN models are integrated using an ensemble learning approach to combine their results.

The results obtained from the two different DQN models built using CNN and LSTM architectures are combined to form a state space. This state space is then used to train a unifying DQN model incorporating a classical artificial neural network. This approach aims to leverage the distinct strengths

of CNN and LSTM networks in extracting features from time-series data through the created ensemble learning structure.
The main contributions of this research article are as follows:

- A novel ensemble reinforcement learning model capable of achieving an optimal trading strategy has been proposed, based on the DQN algorithm.
- With the proposed model, the reward amount and coverage ratio obtained from a single stock have been improved to a higher level compared to the individual results of DQN models utilizing LSTM and CNN networks.
- A high-return Ensemble Learning approach for stock prices has been introduced, achieving high coverage ratios and more optimal Q values.

2. Related Works

Various machine learning techniques have been employed by researchers to address challenging problems in diverse domains, including, data classification[19], [20], [21], object recognition[22], [23], generative networks [24], and optimization [25], [26]. In the field of medical diagnosis, machine learning has demonstrated notable efficacy [27-32]. Apart from examples of such studies using machine learning, there are also studies in the literature that explain how machine learning works [33]. These studies highlight the versatility of machine learning.
Building on these applications, reinforcement learning has gained significant attention for its ability to tackle sequential decision-making problems by learning optimal strategies through interaction and feedback. This capability has found increasing relevance in the financial sector, where decision-making under uncertainty and the optimization of long-term outcomes are critical. Applications such as portfolio management, algorithmic trading, credit risk assessment and stock trading have demonstrated how reinforcement learning can adapt to dynamic financial environments, refine investment strategies, and improve risk-adjusted returns, showcasing its transformative potential in the field.
The studies focusing on stock price prediction based on reinforcement learning are generally categorized into methods involving Deep Q-Learning and Policy Gradient. Additionally, various deep learning and machine learning methods have been frequently utilized alongside these approaches to enhance the accuracy of the obtained results.
In general, three types of RL techniques are used in models proposed for stock prediction. The first is value-based methods such as Q-Learning. In these methods, the agent estimates the value of every action that can be taken in each state and selects the action with the highest value, that is, the highest return [34], [35]. These methods are also called critic-only methods. Second, methods such as Policy Gradient, where the agent directly learns the policy function, are also called actor-only methods [36], [37]. Third and finally, there are methods called Actor-Critic, where the actor performs an action at each step, and the critic evaluates the quality of the action taken [38], [39], [40].
DQN has been used alongside various tools to predict stock markets. One of these tools is CNN networks, which are frequently used in image classification tasks[41]. A DQN model that takes images created from stock data as input and produces a vector containing the probabilities of actions that the agent can take as output was trained using data from the U.S. stock market and later tested with data from the markets of 31 different countries [42]. The results of this study indicate that the trained market data contain patterns suitable for predicting global stock price movements. Studies have been conducted on visualizing time series to convert financial data into images for algorithmic trading using CNN networks and to develop a trading strategy using these images [43]. In addition, TDQN, which is proposed as a reinforcement learning solution to the algorithmic trading problem aimed at determining the optimal buy-sell positions in stock markets, is also a DQN-based algorithm [7]. This algorithm provides benefits such as versatility and robustness compared to classical methods through its use of Double DQN, Huber loss , Xavier initialization, and other data preprocessing methods [44]. Unlike methods using a single agent, there are also studies that use multiple DQN agents and make decisions through an ensemble learning approach. In one study, final decisions were made using results obtained from an ensemble of multiple agents trained with varying numbers of training iterations, aiming to minimize issues such as overfitting, which can occur with machine learning classifiers [45]. In another study using DQN, transfer learning approaches were proposed to prevent overfitting caused by insufficient financial data [46]. There are also studies using the DQN method for trading in foreign exchange markets [47].
In a study where trading operations were performed using DDPG, an Actor-Critic-based algorithm, CNN architecture was employed in both the Actor and Critic networks [48]. In another study, three Actor-

Critic-based algorithms such as Proximal Policy Optimization (PPO), Advantage Actor-Critic (A2C), and Deep Deterministic Policy Gradient (DDPG) were combined using the ensemble learning method. The goal of the ensemble learning method was to leverage the strongest aspects of each algorithm and achieve a more robust model against varying environmental conditions. The algorithms were tested on 30 Dow Jones stocks, and it was observed that the proposed deep ensemble learning strategy achieved better results than the three base algorithms as well as the two benchmark methods, DJIA and minimum variance [49].

The A2C architecture was combined with an autoencoder called Stacked Sparse Denoising Autoencoder (SSDAE) to reduce the impact of external factors on stock data and the effect of noisy data on outcomes in the literature. The results showed that this model performed better than DJIA and state-of-the-art methods [49], [50]. Due to the time-series nature of stock data, recurrent neural network structures have also been used to extract informative financial features from the data. In one study, two different trading strategies were proposed using reinforcement learning methods alongside the Gated Recurrent Unit (GRU) module [51]. One of these was the Gated Deep Q-Learning (GDQN) trading strategy based on the DQN method, while the other was the Gated Deterministic Policy Gradient (GDPG) trading strategy based on the Policy Gradient method.

In another study where financial time series were processed and future predictions were made using deep learning, LSTM was employed alongside wavelet transformations and autoencoders [52]. Additionally, recent studies have indicated that relatively older machine learning methods, such as Multi-Layer Perceptron (MLP), can outperform deep learning methods like GRU and CNN in extracting features from stock market data in certain cases [53]. Comparisons of the proposed model in this study with methods such as DDPG and GDQN in the literature revealed that the model used for feature extraction significantly impacts the performance of the deep reinforcement learning model.

There are also studies utilizing Support Vector Machines (SVM), a machine learning method, for predicting stock returns [54]. In another study based on the Policy Gradient method, portfolio management for cryptocurrencies was presented [55]. Finally, some libraries have been developed for use in financial reinforcement learning studies, and several of these libraries have been made available as open-source tools [56].

Another study proposes a DRL-based trading system employing a cascaded LSTM-PPO model to better capture hidden information in daily stock data, achieving 5% to 52% performance improvements over baseline models in metrics like cumulative returns and profitability across major indices including DJI, SSE50, SENSEX, and FTSE100 [57].

In the literature, a modified actor-critic RL model integrating technical analysis metrics to address multidimensional noise and transaction costs has been shown to outperform pure RL and traditional benchmarks on the S&P500 dataset [58].

In another study, a Multi-Agent Double Deep Q-Network (MADDQN) framework employing distinct time-series feature extraction networks (TimesNet and a Multi-Scale CNN) has shown the ability to balance risk and revenue, achieve an average cumulative return of 23.08%, and generalize robustly across various U.S. stock indices [59].

When the literature is reviewed, it is observed that methods such as RNN-based LSTM and GRU, which are effective on time-series data, are frequently preferred for stock price prediction, while CNN networks are commonly used for feature extraction. Although there are studies that use CNN for feature extraction followed by LSTM for stock price prediction by employing these two models sequentially [60], no study has been found that takes separate predictions from CNN and LSTM networks and combines them using an ensemble learning method. While CNN makes predictions based on the visual patterns in the data, LSTM makes predictions based on historical data. Leveraging the strengths of these two different methods will improve prediction accuracy. In this study, two different DQN methods based on CNN and LSTM are combined using the ensemble learning method, aiming to achieve a high level of accuracy.

3. Material and Methods

The methods, tools, dataset, and evaluation metrics used for training are summarized in Figure 3.1.

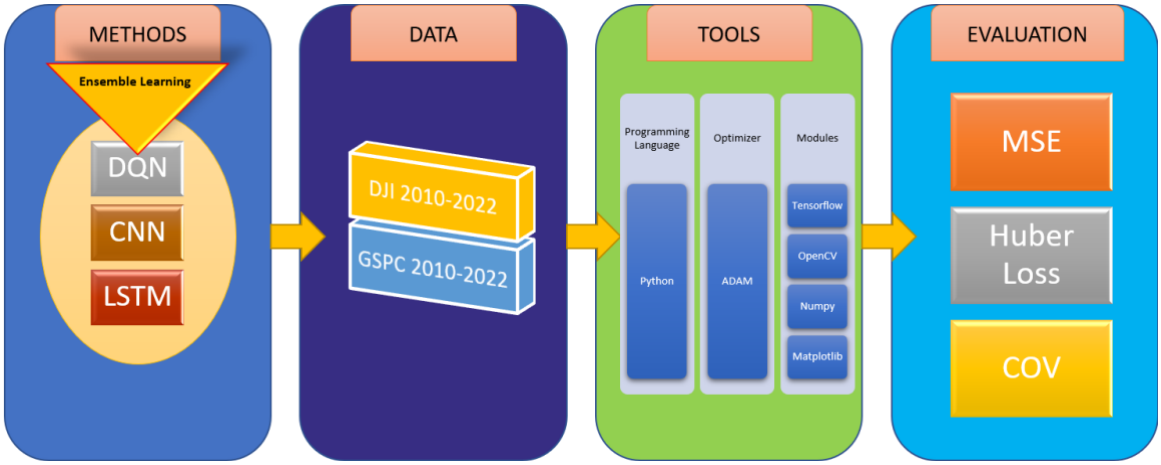

Figure 3.1. Summary of the methods, data, tools, and evaluation metrics used in the study.

3.1. Dataset

This study utilizes data from the Standard & Poor's 500 (GSPC) index [61]. A 10-year portion, equivalent to 2,515 days, of the 11 years of data from this index was used for training, while one year of data was utilized for testing. The datasets were obtained from Yahoo Finance, a well-known financial data platform. These datasets include columns such as date, opening price, daily high and low prices, and closing prices. Among these, the closing prices for each day were used to determine the profit generated when a stock was sold compared to the closing price on the day it was purchased.
To avoid misleading results in terms of accuracy due to training and testing the model on a single dataset, training and testing were conducted using two different datasets. The results and graphs related to these evaluations are discussed in detail in Chapter 4.

3.2. Q-Learning and Deep Q-Learning

Q-learning is a model-free RL approach based on learning the Q-value, which represents the quality of taking a specific action in a given state [62]. It enables learning to estimate the optimal value of an action to be taken in each encountered state for solving sequential decision-making problems. This value is defined as the expected total reward that can be obtained in the future by taking the action and then following the optimal policy. Under a specific policy π, the true value of action a in state s is defined as follows:

$$Q_\pi(s,a) \equiv \mathbb{E}[R_1 + \gamma R_2 + \cdots | S_0 = s, A_0 = a, \pi] \quad (1)$$

Here $\gamma \in [0,1]$ is a discount factor used to adjust the importance of future rewards. In this case, the Q^* which is the optimal Q value is given by $Q^*(s,a) = max_\pi Q_\pi(s,a)$. The optimal values can be achieved by selecting the action with the highest value for each state, thereby obtaining the optimal policy.
In the traditional Q-Learning method, the parameters are updated as follows using the reward R_{t+1} obtained from taking action A_t in state S_t and the new state S_{t+1} :

$$\theta_{t+1} = \theta_t + \alpha(Y_t^Q - Q(S_t, A_t; \theta_t))\nabla_{\theta_t} Q(S_t, A_t; \theta_t) \quad (2)$$

In the equation above, α represents the number of steps. Y_t^Q is defined as follows:

$$Y_t^Q \equiv R_{t+1} + \gamma max_a Q(S_{t+1}, a; \theta_t) \quad (3)$$

A Deep Q Network (DQN) refers to a multi-layer artificial neural network. By taking the state s as input and using the parameters θ, this neural network outputs a vector a containing the action values. This network acts as a function from $\mathbb{R}^n$ to $\mathbb{R}^m$, where n represents the dimension of the state space, and m represents the number of actions in the action space [63]. Two key structures introduced with the DQN architecture are the target network and experience replay [35].
The parameters θ^- representing the target network are copied from the trained Online Network every τ steps. The target used by the DQN architecture is as follows:

$$\theta^{-} = \theta \text{ if } t \bmod \tau = 0 \tag{4}$$

$$Y_t^{DQN} \equiv R_{t+1} + \gamma max_a Q(S_{t+1}, a; \theta_t^{-}) \tag{5}$$

For experience replay, a memory is created to store information observed during training, such as the state, action, reward, and subsequent state. When the specified memory capacity is reached, the oldest entry in the memory is removed to give place for new information.

3.3. Proposed Architecture

The proposed architecture for predicting stock market prices combines the DQN structure with the ensemble learning method. Instead of using a single agent to predict stock prices in a given state, two different DQN networks are used, one employing CNN networks for feature extraction and the other using LSTM networks. The results obtained from these networks are then combined using ensemble learning methods to produce the final action values. The structure of the proposed architecture is shown in Figure 3.2.

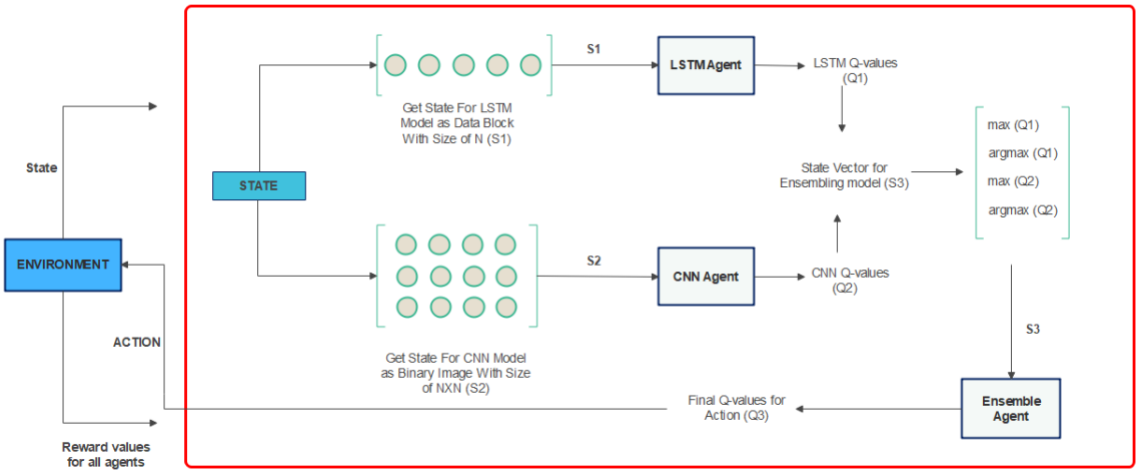

Figure 3.2. Structure of proposed architecture.

In this structure, time-series data is provided to the LSTM network in blocks, and the network makes predictions. The sliding window technique is used for the LSTM network, where *N* previous data points are taken at each time step to make a prediction.

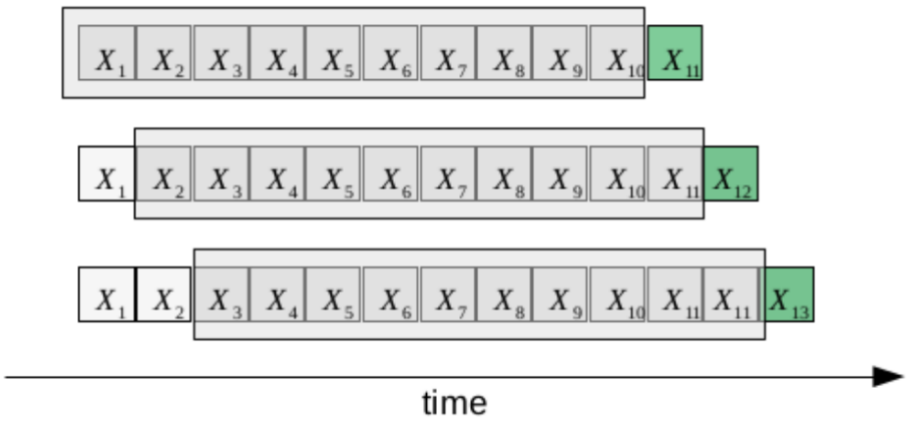

Figure 3.3. Sliding window architecture in LSTM [64].

The CNN network consists of 2D convolutional blocks and takes an image as input. This image matrix is generated using a method we developed to convert time series into binary images. At each training step, fixed-size binary images are created using time-series blocks of *N* data points. Regardless of the value range of the data in the block, the input size for the CNN network remains constant, and a low-

resolution graphical representation is created from the data in the block.
In this method, an $N \times N$ matrix is first created and initialized with white pixel values. Subsequently, the closing values in the block are sorted from smallest to largest, forming a new block. Finally, to generate the graphical image without altering the column order of the time-series data, the row for each column that should take a black pixel value is determined. This is done by checking the index of each time-series value in the sorted block. In this way, for each time-series data block, a graphical image as shown in Figure 3.4 is obtained at each training step. These images are then used to train the DQN network with the CNN structure.

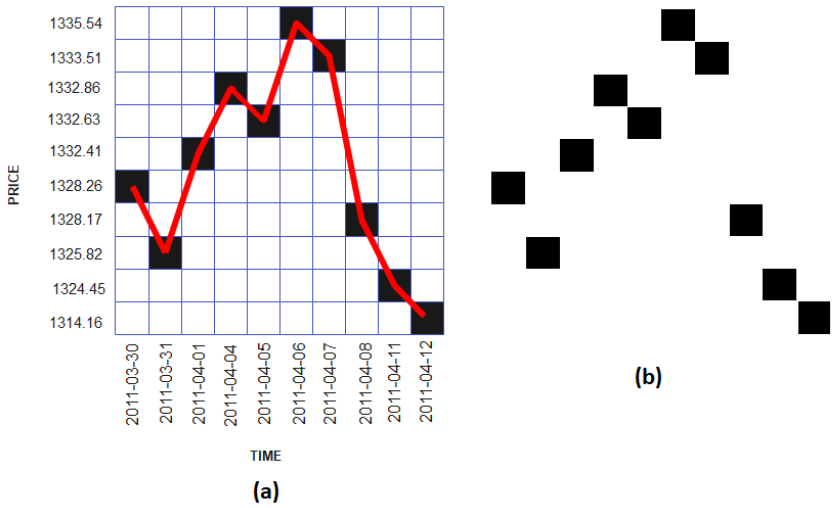

Figure 3.4. The visualization of the prices corresponding to a randomly selected 10-day data block from the dataset is shown in (a), and the binary image provided as input to the CNN network for these prices is shown in (b).

For the current state, the action vectors obtained from two different DQN networks are combined using an ensemble learning method to make the final decision on which action to take. In the ensemble learning method, the highest-value actions and their associated probability values from both networks are taken to create a new state space. The agent responsible for the final decision determines its action not based on the uncertain time-series data but on the results from the LSTM- and CNN-based DQN networks.
This approach aims to enable the final decision-making agent to learn how much weight to give to the results from each network in various situations by considering the action values and probabilities derived from the CNN and LSTM networks. The goal is to leverage the strengths of both CNN and LSTM networks to make more accurate predictions for both rising and falling stock prices. The diagram of the DQN structures incorporating LSTM and CNN networks is shown in Figure 3.5.

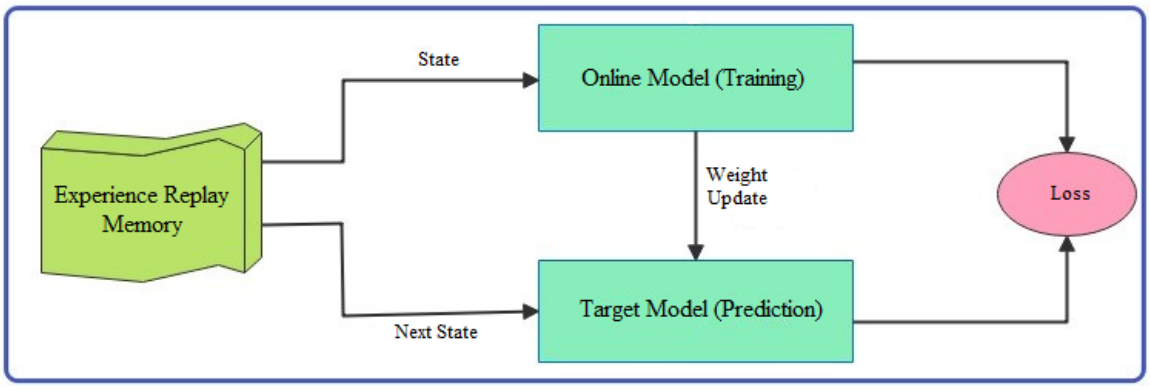

Figure 3.5. The diagram of the DQN structures incorporating LSTM and CNN networks.

3.4. Evaluation Metrics and Training Parameters

MSE (Mean Squared Error), CNN ve LSTM networks used as the loss function to determine how close the predicted values are to the actual values [65].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (6)$$

MSE (Mean Squared Error) is highly sensitive to outliers, and when the difference between the predicted value and the actual value is large, the loss becomes significantly high. While this is desirable in some cases, in DQN networks, a single training update can cause the network to change drastically, leading to significant changes in the target network as well. This can result in even greater errors. In contrast, MAE (Mean Absolute Error) penalizes large errors less, making it more suitable for training the final network that combines the two models. To train the agent more slowly and stably, Huber loss (H), which balances MSE and MAE losses, was used instead of MSE in the final network [7].

$$H(x) = \begin{cases} \frac{1}{2}x^2 & if\ |x| \le 1, \\ |x| - \frac{1}{2} & otherwise. \end{cases} \quad (7)$$

To measure the effectiveness of the entire proposed architecture on financial data, a metric called Coverage was used. [45]. The Coverage ratio indicates the proportion of days on which the trained agent performed trading (X') to the total number of days in the market ($|X|$). The Coverage ratio and the reward functions used during training are specified below.

$$COV = \frac{X'}{|X|} \quad (8)$$

$$B(I) = \begin{cases} 5, & I > D_t \\ -10, & I \le D_t \end{cases} \quad (9)$$

$$R_{ens}(a_1) = \begin{cases} -1, & a_1 = 0 \\ B, & a_1 = 1 \\ \max(D_t - P\ ,\ (D_t - P)/10), & a_1 = 2 \end{cases} \quad (10)$$

$$R_{base}(a_2) = \begin{cases} R_{ens}, & a_2 = a_1 \\ softmax(l)_{a_1}, & otherwise \end{cases} \quad (11)$$

$$softmax(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad (12)$$

Here, the R_{ens} value indicates the amount of reward obtained by the ensemble agent from the environment. The a_1 value represents the action of the ensemble agent. If this value is 0, the agent remains in a waiting state, and no buy or sell action is performed. However, to prevent prolonged waiting, a small penalty score, such as -1, is applied. If a_1 is 1, the agent is in a buying state. In this case, the I parameter sent to the B function represents the agent's current balance, and D_t indicates the price of the stock to be purchased. If a stock is attempted to be bought despite insufficient funds, a penalty is applied; otherwise, the agent receives a certain reward.

If a_1 is 2, the agent is in a selling state. Here, P represents the value of the stock at the time it was purchased. In the selling state, a reward or penalty is assigned based on the difference between the stock's value at the time of sale and its value at the time of purchase.

The R_{base} function represents the reward function for the CNN and LSTM agents that send their action values to the ensemble agent. The reward function is the same for both agents. Here, a_2 represents the action value of the base agents. If the base agent and the ensemble agent select the same action, the reward or penalty is also applied to the base agent. If the base agent selects a different action from the ensemble agent, the normalized Q value of the action chosen from the action list l is applied as a reward or penalty. All the parameters used during the training of the proposed model are presented in Table 1.

Table 1. Training parameters.

Parameter Type	Parametre Name	Parametre Value
Training	Window Size (N)	10
Training	Episode Number	20
Training	Initial Balance	50.000
Training	Batch Size	32
LSTM Agent	State Size	N
CNN Agent	State Size	NxN
Ensemble Agent	State Size	4

Agents	Action Size	3
Agents	Replay Memory Size	1000
Agents	Gamma	0.95
Agents	Epsilon Decay	0.9995
Agents	Tau	0.001
Models	Optimizer	Adam
Models	Optimizer Learning Rate	0.001
Models	Output Activation	Linear
LSTM Model	Loss Function	MSE
CNN Model	Loss Function	MSE
Ensemble Model	Loss Function	Huber Loss

4. Results and Discussion

This section is presented and discussed the results of the training. After training the proposed model on 10 years of stock market data, testing and validation were performed on 1 year of data. In the graph in Figure 4.1, the x-axis represents the date, and the y-axis represents the stock price according to the dates. In the tests conducted on the GSPC 2011 data, the proposed model achieved a net profit of $2,258.27. The green points on the graph represent the points where the model made purchases, and the red points represent the points where the model made sales.

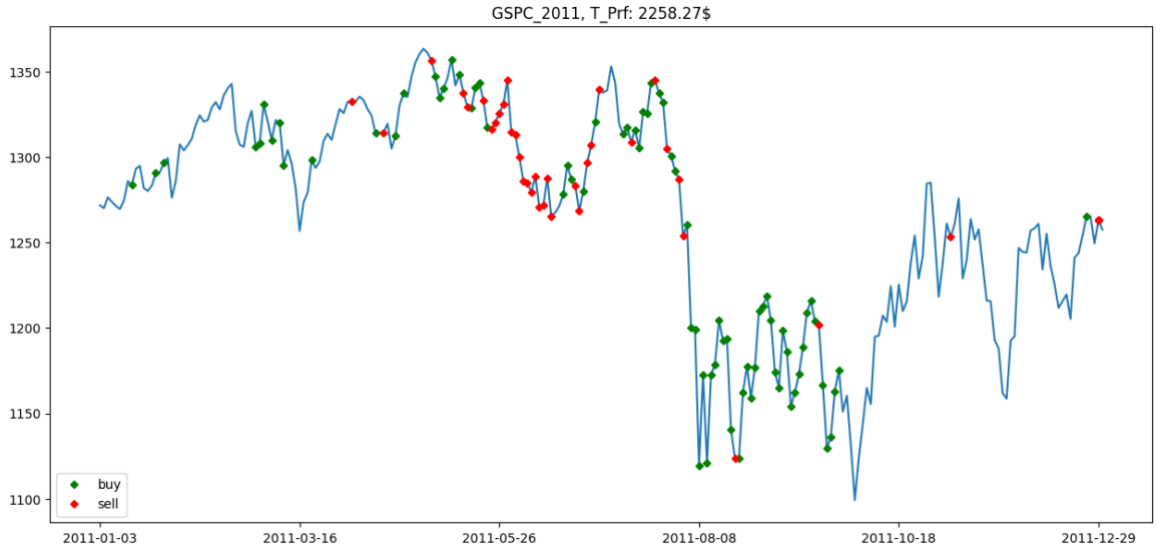

Figure 4.1. The test graph of the proposed model on the GSPC 2011 data and the profit achieved.

Figure 4.1 illustrates that the proposed model typically executes sell orders when stock prices are high and prioritizes buying when prices decline. While sales are generally made when the price is high, during sudden drops, the model quickly switches to the buying state and generates small profits. Additionally, in regions where prices are low and buying occurs, the market was tested before sudden drops and sales were made; however, when the decline continued, the model switched back to the buying state. In the final part of the graph, due to high uncertainty, sales were made near optimal points to generate profits. Outside of these selling points, since there was no stable pattern, the model preferred to remain in the waiting state. Figure 4.2 shows how much the proposed model contributes to stock price prediction compared to classical LSTM and CNN models.

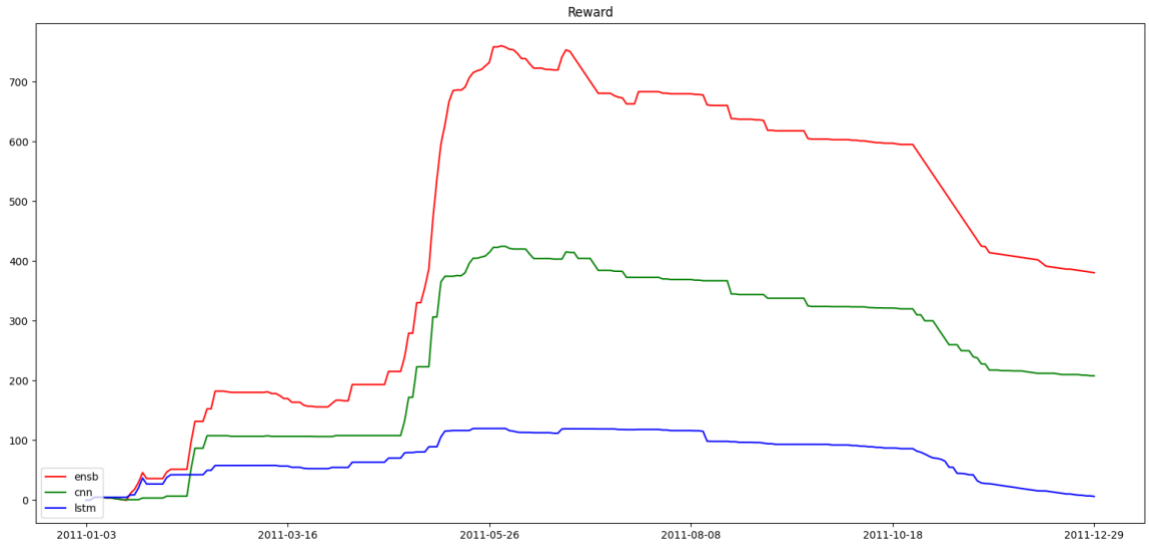

Figure 4.2. The reward amount achieved by the ensemble agent compared to the CNN and LSTM agents.

In the above figure, the reward amounts obtained by the agents using LSTM and CNN networks are compared with the reward amount achieved by the ensemble agent, which makes decisions using the Q values of these two methods. The obtained reward amounts were calculated based on the reward functions previously mentioned. The ensemble agent achieved a higher reward rate than the base agents by correctly evaluating the information received from the base agents and making buy-sell decisions at the right time. The higher reward amount of the ensemble agent compared to the base agents indicates that the proposed model helps make more accurate decisions by using the Q values of the base agents. Additionally, the Coverage ratios of the two base agents and the ensemble learning agent are shown in Figure 4.3.

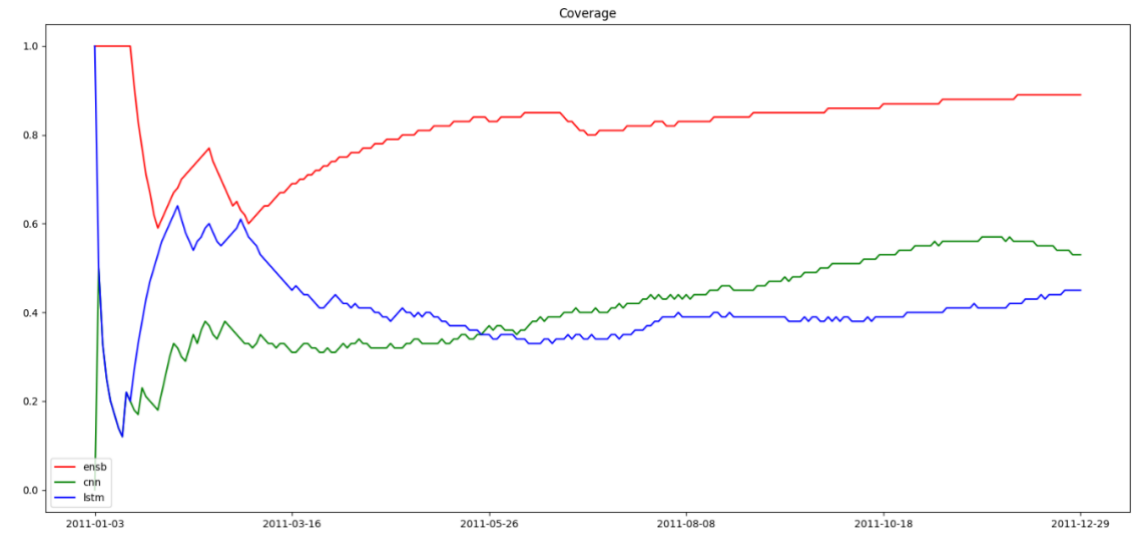

Figure 4.3. The Coverage ratios achieved by the ensemble agent compared to the CNN and LSTM agents (Higher is better).

The Coverage ratio represents the proportion of days on which the trained stock agent performed trading compared to the total number of days. Therefore, the ensemble agent in the proposed model tends to perform more buy and sell actions compared to the base agents. The fact that trading is done instead of simply waiting in the market, and that it is performed at the right time to increase the net profit, indicates that the proposed model operates more actively and with higher accuracy compared to classical models. The proposed method achieved a Coverage ratio of 0.86. In Table 2, a comparison of the proposed model with other reinforcement learning-based stock trading studies in the literature is presented.

The training was performed on a MacBook Air with an M1 processor and 8 GB of RAM, taking 204 hours. Although training in Deep Reinforcement Learning applications is long and costly, obtaining results from the trained models is extremely cheap and fast.

Table 2. Comparison of the proposed model with other RL approaches

Study	Methodology	Dataset	Coverage Ratio	Profit ($)
Carta et al. [16]	Ensemble RL	S&P 500, JPM, MSFT	**1.00**	1265.50
Chen & Gao [15]	Deep Recurrent Q-Network	SPY Test	N/A	338.8
Proposed Model	Ensemble RL with LSTM and CNN	S&P 500	0.86	**2,258.27**

Table 2 demonstrates that the proposed ensemble learning-based model achieves a significantly higher profit level compared to other reinforcement learning approaches in the literature. While some of the listed studies maintain a high coverage ratio by trading on all available days, the proposed model attains even greater net profit with a more selective trading strategy. This indicates that the model's success does not merely stem from frequent trading, but rather from making timely and accurate buy-sell decisions. Thus, by balancing the frequency of trades and profitability, the proposed model shows greater potential for more effective outcomes than traditional methods presented in previous research.

5. Conclusion

In the finance sector, trading based on stock market perceptions is much less efficient compared to algorithms that can extract meaning from data. Therefore, in recent decades, various analytical methods and decision-making mechanisms have been developed to assist investors in stock trading. Algorithmic trading has become an important solution for problems in the financial technology industry. Recent advancements in machine learning and deep learning methods have also begun to rapidly find their place in the field of algorithmic trading.

In this study, a trading algorithm model has been developed using reinforcement learning, one of the latest trends in the field of deep learning. The results obtained with a multi-agent reinforcement learning method were combined using an ensemble learning approach. The ensemble learning method, also known as meta-learning, can be described as obtaining a single result by using the outcomes from several different models. In this study, agents with CNN and LSTM models were used as base learners. Information about the Q values obtained by these agents was sent to another agent, which made the final prediction. This decision-making agent has a DQN structure and is trained not with raw data but with the results derived from the data, enabling the implementation of the ensemble learning method. With the proposed structure, the reward amount and Coverage ratio obtained from a single stock have been improved to a higher level compared to the individual results of the agents using LSTM and CNN networks. The proposed model achieved a net profit of $2,258.27 after one year in the stock market. As demonstrated in Table 2, the ensemble learning-based model presented here not only surpasses previously reported methods in terms of net profit but also achieves this with a more judicious trading approach, thereby underscoring its potential as a robust and efficient tool for algorithmic trading. This shows that the proposed model can be an important tool to assist investors' decisions in the stock market. The proposed model is not only useful for stock price prediction but is also presented as a model that can be applied in any field that requires the prediction of uncertain time-series data.

Acknowledgement

This study did not receive any dedicated funding from public, commercial, or non-profit organizations.

Conflict of Interest Statement

No conflict of interest was declared by the authors.

References

[1] Y. Zhang and L. Wu, "Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network," *Expert Syst Appl*, vol. 36, no. 5, pp. 8849–8854, 2009.
[2] B. Graham, D. L. F. Dodd, and S. Cottle, *Security analysis*, vol. 452. McGraw-Hill New York, 1934.
[3] J. J. Murphy, "John J Murphy-Technical Analysis Of The Financial Markets. pdf," *Pa Dent J (Harrisb)*, 1999.
[4] E. Chan, *Algorithmic trading: winning strategies and their rationale*, vol. 625. John Wiley & Sons, 2013.
[5] R. Ramezanian, A. Peymanfar, and S. B. Ebrahimi, "An integrated framework of genetic network programming and

multi-layer perceptron neural network for prediction of daily stock return: An application in Tehran stock exchange market," *Appl Soft Comput*, vol. 82, p. 105551, 2019.
[6] E. F. Fama, "Efficient capital markets: A review of theory and empirical work," *J Finance*, vol. 25, no. 2, pp. 383–417, 1970.
[7] T. Théate and D. Ernst, "An application of deep reinforcement learning to algorithmic trading," *Expert Syst Appl*, vol. 173, p. 114632, 2021.
[8] J. Zhang and Y. Lei, "Deep reinforcement learning for stock prediction," *Sci Program*, vol. 2022, 2022.
[9] J. W. Lee, E. Hong, and J. Park, "A Q-learning based approach to design of intelligent stock trading agents," in *2004 IEEE International Engineering Management Conference (IEEE Cat. No. 04CH37574)*, IEEE, 2004, pp. 1289–1292.
[10] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
[11] J. Li, R. Rao, and J. Shi, "Learning to trade with deep actor critic methods," in *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, IEEE, 2018, pp. 66–71.
[12] Z. Zhang, S. Zohren, and S. Roberts, "Deep reinforcement learning for trading," *The Journal of Financial Data Science*, vol. 2, no. 2, pp. 25–40, 2020.
[13] Q.-V. Dang, "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*, Springer, 2020, pp. 311–322.
[14] Z. Zhang, S. Zohren, and S. Roberts, "Deep learning for portfolio optimization," *The Journal of Financial Data Science*, vol. 2, no. 4, pp. 8–20, 2020.
[15] L. Chen and Q. Gao, "Application of deep reinforcement learning on automated stock trading," in *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, 2019, pp. 29–33.
[16] S. Carta, A. Corriga, A. Ferreira, A. S. Podda, and D. R. Recupero, "A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning," *Applied Intelligence*, vol. 51, pp. 889–905, 2021.
[17] T.-V. Pricope, "Deep reinforcement learning in quantitative algorithmic trading: A review," *arXiv preprint arXiv:2106.00123*, 2021.
[18] I. Boukas *et al.*, "A deep reinforcement learning framework for continuous intraday market bidding," *Mach Learn*, vol. 110, pp. 2335–2387, 2021.
[19] N. Yağmur, H. Temurtaş, and İ. Dağ, "Anemi Hastalığının Yapay Sinir Ağları Yöntemleri Kullanılarak Sınıflandırılması," *Journal of Scientific Reports-B*, no. 008, pp. 20–34, 2023.
[20] N. N. Arslan, E. Şahin, and M. Akçay, "Deep learning-based isolated sign language recognition: a novel approach to tackling communication barriers for individuals with hearing impairments," *Journal of Scientific Reports-A*, no. 055, pp. 50–59, Dec. 2023, doi: 10.59313/jsr-a.1367212.
[21] S. Serttaş and E. Deniz, "Disease detection in bean leaves using deep learning," *Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering*, vol. 65, no. 2, pp. 115–129, Dec. 2023, doi: 10.33769/aupse.1247233.
[22] F. Aydemir and S. Arslan, "A System Design With Deep Learning and IoT to Ensure Education Continuity for Post-COVID," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 2, pp. 217–225, May 2023, doi: 10.1109/TCE.2023.3245129.
[23] G. Arslan, F. Aydemir, and S. Arslan, "Enhanced license plate recognition using deep learning and block-based approach," *Journal of Scientific Reports-A*, no. 058, pp. 57–82, Sep. 2024, doi: 10.59313/jsr-a.1505302.
[24] E. Şahin and M. F. Talu, "WY-NET: A NEW APPROACH to IMAGE SYNTHESIS with GENERATIVE ADVERSARIAL NETWORKS," *Journal of Scientific Reports-A*, no. 050, pp. 270–290, 2022.
[25] D. Özdemir, S. Dörterler, and D. Aydın, "A new modified artificial bee colony algorithm for energy demand forecasting problem," *Neural Comput Appl*, vol. 34, no. 20, pp. 17455–17471, Oct. 2022, doi: 10.1007/s00521-022-07675-7.
[26] N. Yagmur, İ. Dag, and H. Temurtas, "Classification of anemia using Harris hawks optimization method and multivariate adaptive regression spline," *Neural Comput Appl*, vol. 36, no. 11, pp. 5653–5672, Apr. 2024, doi: 10.1007/s00521-023-09379-y.
[27] E. Şahin, D. Özdemir, and H. Temurtaş, "Multi-objective optimization of ViT architecture for efficient brain tumor classification," *Biomed Signal Process Control*, vol. 91, p. 105938, May 2024, doi: 10.1016/j.bspc.2023.105938.
[28] S. Dörterler, H. Dumlu, D. Özdemir, and H. Temurtaş, "Hybridization of Meta-heuristic Algorithms with K-Means for Clustering Analysis: Case of Medical Datasets," *Gazi Journal of Engineering Sciences*, vol. 10, no. 1, pp. 1–11, Apr. 2024, doi: 10.30855/gmbd.0705N01.
[29] P. O. Kavas, M. R. Bozkurt, I. Kocayigit, and C. Bilgin, "A New Medical Decision Support System for Diagnosing HFrEF and HFpEF Using ECG and Machine Learning Techniques," *IEEE Access*, vol. 10, pp. 107283–107292, 2022, doi: 10.1109/ACCESS.2022.3213065.
[30] P. Özen Kavas, M. Recep Bozkurt, İ. Kocayiğit, and C. Bilgin, "Machine learning-based medical decision support system for diagnosing HFpEF and HFrEF using PPG," *Biomed Signal Process Control*, vol. 79, p. 104164, Jan. 2023, doi: 10.1016/j.bspc.2022.104164.
[31] Ç. Erçelik and K. Hanbay, "Gauss Filtreleme ve ResNET50 Modeli Kullanılarak Beyin Tümörlerinin Sınıflandırılması," *Computer Science*, Oct. 2023, doi: 10.53070/bbd.1345848.
[32] Ç. Erçelik and K. Hanbay, "Beyin Tümörü Sınıflandırmada Histogram Eşitleme Yönteminin Bazı Derin Öğrenme Modellerine Etkileri," *Computer Science*, Dec. 2023, doi: 10.53070/bbd.1373990.
[33] E. Şahin, N. N. Arslan, and D. Özdemir, "Unlocking the black box: an in-depth review on interpretability, explainability, and reliability in deep learning," *Neural Comput Appl*, Nov. 2024, doi: 10.1007/s00521-024-10437-2.
[34] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.
[35] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
[37] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*, Pmlr, 2014, pp. 387–395.
[38] A. R. Costa and C. G. Ralha, "AC2CD: An actor–critic architecture for community detection in dynamic social networks," *Knowl Based Syst*, vol. 261, p. 110202, 2023.
[39] I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and

natural policy gradients," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.
[40] P. Christodoulou, "Soft actor-critic for discrete action settings," *arXiv preprint arXiv:1910.07207*, 2019.
[41] E. Deniz and S. Serttaş, "Deep learning-based distributed denial of service detection system in the cloud network," *Journal of Scientific Reports-A*, no. 055, pp. 16–33, Dec. 2023, doi: 10.59313/jsr-a.1333839.
[42] J. Lee, R. Kim, Y. Koh, and J. Kang, "Global stock market prediction based on stock chart images using deep Q-network," *IEEE Access*, vol. 7, pp. 167260–167277, 2019.
[43] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Appl Soft Comput*, vol. 70, pp. 525–538, 2018.
[44] G. P. Meyer, "An alternative probabilistic interpretation of the huber loss," in *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2021, pp. 5261–5269.
[45] S. Carta, A. Ferreira, A. S. Podda, D. R. Recupero, and A. Sanna, "Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting," *Expert Syst Appl*, vol. 164, p. 113820, 2021.
[46] G. Jeong and H. Y. Kim, "Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning," *Expert Syst Appl*, vol. 117, pp. 125–138, 2019.
[47] J. Carapuço, R. Neves, and N. Horta, "Reinforcement learning applied to Forex trading," *Appl Soft Comput*, vol. 73, pp. 783–794, 2018.
[48] S. Luo, X. Lin, and Z. Zheng, "A novel CNN-DDPG based AI-trader: Performance and roles in business operations," *Transp Res E Logist Transp Rev*, vol. 131, pp. 68–79, 2019.
[49] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid, "Deep reinforcement learning for automated stock trading: An ensemble strategy," in *Proceedings of the first ACM international conference on AI in finance*, 2020, pp. 1–8.
[50] H. Yue, J. Liu, D. Tian, and Q. Zhang, "A Novel Anti-Risk Method for Portfolio Trading Using Deep Reinforcement Learning," *Electronics (Basel)*, vol. 11, no. 9, p. 1506, 2022.
[51] X. Wu, H. Chen, J. Wang, L. Troiano, V. Loia, and H. Fujita, "Adaptive stock trading strategies with deep reinforcement learning methods," *Inf Sci (N Y)*, vol. 538, pp. 142–158, 2020.
[52] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS One*, vol. 12, no. 7, p. e0180944, 2017.
[53] M. Taghian, A. Asadi, and R. Safabakhsh, "Learning financial asset-specific trading rules via deep reinforcement learning," *Expert Syst Appl*, vol. 195, p. 116523, 2022.
[54] F. D. Paiva, R. T. N. Cardoso, G. P. Hanaoka, and W. M. Duarte, "Decision-making for financial trading: A fusion approach of machine learning and portfolio selection," *Expert Syst Appl*, vol. 115, pp. 635–655, 2019.
[55] Z. Jiang and J. Liang, "Cryptocurrency portfolio management with deep reinforcement learning," in *2017 Intelligent Systems Conference (IntelliSys)*, IEEE, 2017, pp. 905–913.
[56] X.-Y. Liu *et al.*, "FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance," *arXiv preprint arXiv:2011.09607*, 2020.
[57] J. Zou, J. Lou, B. Wang, and S. Liu, "A novel Deep Reinforcement Learning based automated stock trading system using cascaded LSTM networks," *Expert Syst Appl*, vol. 242, p. 122801, May 2024, doi: 10.1016/j.eswa.2023.122801.
[58] Z. Pourahmadi, D. Fareed, and H. R. Mirzaei, "A Novel Stock Trading Model based on Reinforcement Learning and Technical Analysis," *Annals of Data Science*, vol. 11, no. 5, pp. 1653–1674, Oct. 2024, doi: 10.1007/s40745-023-00469-1.
[59] Y. Huang, C. Zhou, K. Cui, and X. Lu, "A multi-agent reinforcement learning framework for optimizing financial trading strategies based on TimesNet," *Expert Syst Appl*, vol. 237, p. 121502, Mar. 2024, doi: 10.1016/j.eswa.2023.121502.
[60] H. Widiputra, A. Mailangkay, and E. Gautama, "Multivariate cnn-lstm model for multiple parallel financial time-series prediction," *Complexity*, vol. 2021, pp. 1–14, 2021.
[61] Yahoo Finance, "S&P 500 (^GSPC)," Yahoo Finance Historical Data. [Online]. Available: https://finance.yahoo.com/quote/%5EGSPC/history/
[62] T. Théate, A. Wehenkel, A. Bolland, G. Louppe, and D. Ernst, "Distributional reinforcement learning with unconstrained monotonic neural networks," *Neurocomputing*, 2023.
[63] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, 2016.
[64] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," *arXiv preprint arXiv:1705.05690*, 2017.
[65] T. Zhang, S. Song, S. Li, L. Ma, S. Pan, and L. Han, "Research on gas concentration prediction models based on LSTM multidimensional time series," *Energies (Basel)*, vol. 12, no. 1, p. 161, 2019.