A Deep Multimodal Reinforcement Learning System Combined with CNN and LSTM for Stock Trading

Hong-Gi Shin
School of Robotics
Kwangwoon University
Seoul, Korea
Email: ghdrl95@gmail.com

Ilkyeun Ra
Dept. of Computer Science & Eng.
University of Colorado Denver
Denver, CO United States
Email: ilkyeun.ra@ucdenver.edu

Yong-Hoon Choi
School of Robotics
Kwangwoon University
Seoul, Korea
Email: yhchoi@kw.ac.kr

***Abstract*—This paper proposes a deep multimodal reinforcement learning policy combining convolutional neural network (CNN) and long short-term memory (LSTM) neural network for stock price prediction. The proposed machine learning system generates various charts from stock trading data and uses them as inputs to the CNN layer. The features extracted through CNN layer are divided into column vectors and input to the LSTM layer. In order to utilize various types of charts, a multimodal structure is used in which the input layer and the hidden layer are separated, and the individual operation results are combined. For reinforcement learning, we define agents' policy neural network structure, reward, and action, and provide buying, selling, and holding probabilities as final output. In order to verify the performance of the proposed policy, we conducted training and testing with the daily data of the 256 stocks listed on the Korea KOSPI. Data from March 2009 to March 2019 were used as training data. When April 2019 data was used as test data, the KOSPI rose 0.5%, but the proposed machine learning model achieved a profit of at least 0.89% up to 3.06%. When the data for May 2019 was used as a test, the KOSPI fell by 7.8%, but when invested using the proposed machine learning model, the loss rate ranged from a minimum of 0.82% to a maximum of 6.81%. Therefore, we can see that performance is excellent both in bear and bull market.**

Keywords—Reinforcement Learning, CNN, LSTM, DQN, stock trading, charts, stock price forecast

I. INTRODUCTION

With the rapid development of machine learning (ML) technologies, research is being actively conducted to use these technologies for stock investment. Unlike rule-based algorithm trading, the ML-based stock trading system learns the characteristics of stock data, generates signals, and optimizes portfolios by itself. In recent years, various studies have been conducted on ML models related to stock trading. The variables used in technical analysis are influenced by each other, but some variables such as price and volume have multimodality characteristics with different ranges of values and different patterns of movement. For example, as of August 2019, Samsung Electronics Co., Ltd. has a daily price of about 45,000 KRW and a daily trading volume of 13 million. It is desirable to apply techniques suitable for multimodal data learning such as ensemble learning or bootstrapping, rather than learning in one single learning model, because stock price and trading volume are multimodal data. In order to reflect this effectively, we use multimodal ML method. A comprehensive survey of multimodal ML can be found in work [1].

Traditionally, time series analysis such as autoregressive integrated moving average (ARIMA) model has been widely used for stock chart analysis. Recently, however, supervised machine learning model for stock price prediction has been actively studied [2]-[5]. Work [2] proposed a CNN-FG structure that predicts the next-day trend of the Korea composite stock price index (KOSPI). It divides the KOSPI by 5 days to generate a chart image and predict the trend by inputting the generated image. Work [3] is a study using a Japanese bar chart and a ceiling price to compensate for the lack of a moving average line. The author describes the input data generation method and vector in detail, but he did not disclose the neural network structure used in the experiments. Work [4] predicts stock price fluctuations using time series stock data as inputs to the bidirectional long short-term memory (LSTM) neural network. Since the input data size is small, the number of LSTM cells is also small. Works [5] and [6] used news texts related to the events as well as chart to predict the stock price of each stock. Work [5] used convolutional neural networks (CNN) to use chart images, and work [6] used LSTM neural network to apply past events to current judgments.

Reinforcement Learning (RL) is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences. In order to build an optimal policy in stock trading environment, the agent faces the dilemma of exploring new states while maximizing its reward at the same time. This is called exploration versus exploitation trade-off. Researches on applying RL to the stock environment have been studied, and representative studies on the structure of learning techniques and policies are [7]-[9]. Work [7] applied the agent using the temporal difference method to the stock environment. Work [8] proposed an agent that distributes portfolios by applying an actor/critic model. Work [9] proposed a multiagent scheme that divides the actions that should be performed in the stock environment into multiple agents and proceeds with the sequential trading.

This paper proposes a multimodal policy and a reinforcement learning approach combining CNN and LSTM. First, from stock market historical data, we convert it to various charts and use them as inputs to the proposed learning model. The CNN layer extracts the features from the generated charts, separates the features into column vectors, and provides them as inputs to the LSTM layer to learn the variability of data over

This research was supported by Basic Science Research Program through the National Research Foundation (NRF) of Korea funded by the Ministry of Education (NRF-2016R1D1A1B03934507).

time. The RL policy uses the deep Q-network (DQN) scheme with a structure that computes the Q value to compute the appropriate action for the state. In the course of the episode, we do not use the portfolio change rate as immediate reward but deliver the change rate for the last action when the episode ends and reward value for the previous actions through the discount factor. In this paper, we examine portfolio value changes and Sharpe ratios for stocks listed in the KOSPI. Data from March 2009 to March 2019 (i.e., 2500 trading days) are used as training data, and data for April and May 2019 are used as test data so that training and test data do not overlap. The length of the episode for learning and testing used in our experiments is 20 trading days.

The organization of this paper is as follows. In Section 2, we describe how to generate various images from stock transaction data, how to extract features by inputting images, LSTM layer to learn time series data, and reinforcement learning policy and environment. In Section 3, we show that the proposed learning model can increase the return on investment. Finally, conclusions are presented in Section 4.

II. The Proposed Method

A. Input Data Generation

If the range of values of each element of the input vector of the artificial neural network is different, there is a problem that the learning based on the gradient descent is not performed well. Because there are many factors that determine the stock price, it is very difficult to apply the normalization technique to change the value range of the elements of the vector similarly. In this paper, for data normalization, various stock trading indices are changed into formatted images. Since each image represents time series data, it can be appropriately separated in columns, and used as an input to LSTM neural network to learn the characteristics according to time. In addition, the use of image as an input to an artificial neural network can prevent the size of the input from increasing and the inequality in the range of values of each index (e.g., price, volume, and the range of fluctuation in price).

There are three types of images produced in this paper. The first image is a single image that combines the candlestick chart, four moving average curves (5, 20, 60, and 120 days), and bar graph of trading volume, as shown in Fig, 1(a). The second image is the DMI shown in Fig. 1(b), which is a trend indicator for assessing price direction and strength. The DMI is a combination of three indicators, the average directional movement index (ADX), the positive directional indicator

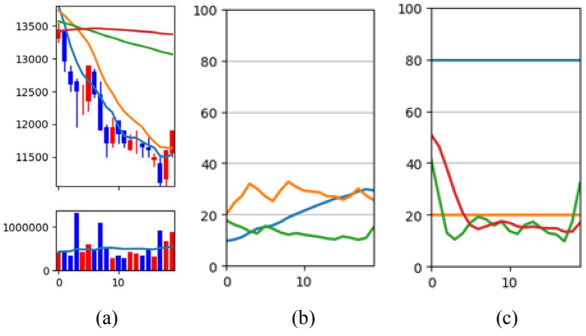

Fig. 1. The chart images created by this paper: (a) candlestick chart, four moving average curves, and bar graph of volume as one image, (b) DMI (ADX: blue line, +DI: green line, -DI: orange line), and (c) SSO

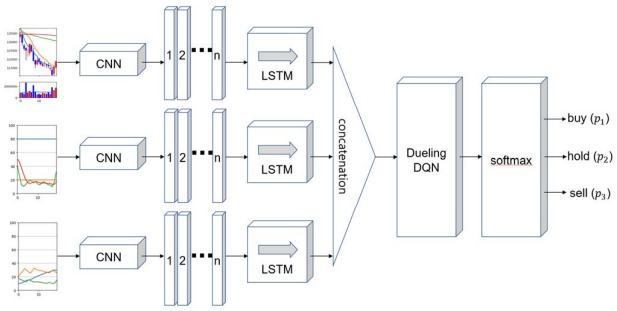

Fig. 2. The Proposed Deep Reinforcement Learning Structure

(abbreviated +DI), and negative directional indicator (-DI). The third image is the slow stochastic oscillator (SSO) indicator shown in Fig. 1(c), which is a momentum indicator that shows the location of the high-low range over a set number of periods. Each image is a chart for 20 trading days, equal to the length of the episode. Although there are more than 25 indicators used for technical analysis, only a few indicators such MA, DMI and SSO were used in this study. Learning models with additional auxiliary indicators are currently under study.

B. Deep Neural Network Structure

This paper combines CNN and LSTM to reflect the time properties of the image in the deep neural network (DNN). Generally, the result of passing through CNN is three-dimensional tensors composed of rows, columns and filters, but in this paper, we split the tensor based on the column and combine the row and filter to create a one-dimensional vector. The generated vectors contain the oldest index on the far left and the most recent index on the far right. Therefore, using these vectors from the left as inputs to the recurrent neural network (RNN), time series data analysis is possible.

In this paper, separate data (i.e., vector) is computed so that historical data could be used to influence the current, using the input from the left column to the right column sequentially as input to the LSTM layer. The data separated by the column sequentially passes through the LSTM layer, and the last operation value is transmitted to the next layer to calculate the action-specific probability value. The RL policy uses a DQN scheme with a structure that computes the Q value to determine the appropriate action for the state. We apply Dueling DQN [10] which separates the value for the state and the action-specific gain value for the state so that the Q value does not diverge, and Double DQN [11] which separates the learning network during the episode. The overall structure of the proposed artificial neural network learning model is shown in Fig 2.

In RL, the environment delivers state and reward to the agent, and the agent determines the action based on the state. We applied Bayesian RL, which utilizes a dropout before the output layer, so that the agent can perform various exploration during the learning process. In order to mitigate the loss when the action selected by the probability is not correct, the agent is designed to calculate the amount of stock to trade through (1) if the agent chooses to buy or sell.

$$\mathrm{TU}_{(s,a)} = U_{min} + \lfloor p(s,a) \times (U_{max} - U_{min}) \rfloor, \quad (1)$$

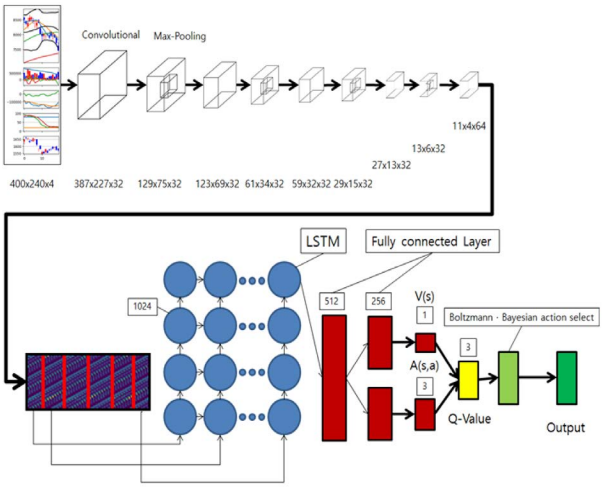

Fig. 3. Neural Network Structure used in Experiments

where state s is the indicator data used by the reinforcement learning model, action a is an action that can be selected by the agent, $\mathrm{TU}_{(s,a)}$ is the number of stocks to trade for the selected action in the current state, $p(s,a)$ is the probability of the action selected in the current state, U_{min} is the minimum possible number of trading stocks, and U_{max} is the maximum possible number of trading stocks. Therefore, as the probability value of the action selected by the agent is larger, the transaction is performed close to the maximum number of transaction stocks.

As stock investors analyze various indicators of stocks and determine the direction of investment, agents operating in stock trading environment also use various indices of stocks as input to determine the direction of investment. In this paper, we use the multimodal method to select the action based on the results of individual computation of the input images. Fig. 2 shows a schematic representation of a multimodal policy for selecting action by inputting candlestick chart, DMI, and stochastic images. The feature of the chart image is extracted through the CNN layer and the vector value of the current state is calculated by the LSTM layer. Then, the vector values of the surface are combined into one vector and used for Q value calculation.

We design an environment that is linked to the real stock environment, delivering the agent reward to the current state of the stock environment and action, and applying the agent's selected action to the real stock environment. In the learning process, in order to shorten the learning time, historical data of the real stock environment is collected and images of the indicators are stored in the hard disk in advance. An agent can learn wrong action if the agent gives immediate reward because the portfolio value changes as the stock trades. Thus, the reward is calculated through (2) and (3) when the episode ends.

$$r = 100 \times (P_T - P_0)/P_0, \tag{2}$$

where r is the rate of return during the trading period T, P_T is the portfolio value at the end of the episode, and P_0 is the portfolio value at the start of the episode. The discount reward G_t for each action selected at time t is

$$G_t = \gamma^{T-t} r, \tag{3}$$

where γ is discount factor. As the period T becomes longer, the γ must be close to 1 to calculate the meaningful discount reward for the early action of the episode.

III. Experiments

A. Experiment Environment

The data used in the experiment are 256 stocks with ticks of 50 KRW (i.e., US 50 cents) among the KOSPI listed stocks. The training data set is 2500 days of trading data from March 2009 for all stocks except for stocks that have not been traded for 2500 days. The data collected for learning are daily prices (e.g., opening price, closing price, daily high and daily low prices), trading volume, moving averages, closing strength, consumption rate, turnover rate, foreign net buying, institutional net buying, and personal net buying for each day. These data are used to generate the three types of images shown in Fig. 1.

Tests are also conducted for all 256 items. The first test is conducted from April 1, 2019 to April 26, 2019 for a total of 20 days. During the period, the KOSPI rose by 0.5% from 2168.28 to 2179.31. The second test is conducted from May 2, 2019 to May 30, 2019 for a total of 20 days. During the period, the KOSPI fell by 7.8% from 2212.75 to 2038.80. The experimental environment is summarized in Table I.

TABLE I. Summary of Experiment Environment

Item	Training data	Test data
Data collection period	March 2009 – March 2019 (2500 days)	April 2019 (20 days) May 2019 (20 days)
Price range	10,000 – 50,000 KRW	
Data type	Daily data	
Total number of items	256	
Amount of data collected	460,000	10,000
Number of columns	11	

From stock market historical data, we converted it to candlestick, DMI, and SSO charts as shown in Fig. 1. Indicators used in chart generation are daily prices (e.g., opening price, closing price, daily high and daily low prices), volume, moving average, DMI, and SSO. The width and height of the chart image were unified for use as CNN input. After creating and saving an image based on Matplotlib, if the generated image does not fit the specified size (i.e., $400 \times 215 \times 4$), the pixels on the right or bottom of the image are filled in with white. The minimum and maximum values of the DMI and the SSO chart are fixed at 0 and 100, and candlestick and trading volume images are generated based on the maximum and minimum values of the input data. In order to shorten the learning time, the charts of the items not created are stored in the hard disk and are called up when necessary.

As shown in Fig. 3, the 3D tensors resulting from the CNN layer are used as the input vectors of the LSTM sequentially from the left column. Only the last operation value among the operation results of the LSTM layer is extracted and used as the value of the current data. To generate the Dueling DQN structure, we separate the values from the fully connected layer and compute the current state and action reward values. Finally, we combine the state value and the reward value to compute the Q value for each action and obtain the probability of taking each action based on the Q value. When calculating the Q value in the learning process, we applied the Boltzmann

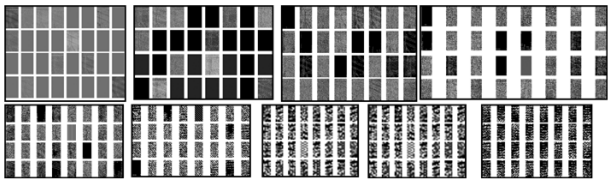

Fig. 4. CNN's layer-by-layer results. From the top left, the convolution layer and the max-pooling layer are alternately arranged.

Bayesian action selection method to explore various actions and then gradually exploit the policy. The specifications of the computer used in the experiment are as follows. CPU: AMD Ryzen 7 1700, GPU: NVIDIA GeForce GTX 1070Ti, and Memory: DDR4 32GB. Table II shows the hyperparameters set for the reinforcement learning policy.

TABLE II. THE HYPERPARAMETERS USED IN THE EXPERIMENT

Hyperparameters	Values
Optimizer	Adam
Learning Rate	0.005
Epsilon	1 – 0.1
Temp (Boltzmann action select parameter)	1 – 0.1
Dropout (Bayesian action select parameter)	0.1 – 1
Discount Rate	0.8
Replay Memory	20
Episode Length	20
Training Epoch	20,000
Episode Base Money	3,000,000 KRW
Trading Unit	0 – 100

We conducted three kinds of experiments, changing the number of policies.

- Experiment A: It applies RL using a single policy. The action is determined by inputting only images showing the candlestick, the moving average line, and the trading volume. That is, only the image of Fig. 1 (a) is used as input.
- Experiment B: It adds a policy to the experiment A to use the SSOs image as input. That is, the images of Fig. 1 (a) and Fig. 1 (c) are used as inputs.
- Experiment C: It adds a policy to the experiment B to use the DMI image as input. That is, all the images in Fig. 1 are used as inputs.

Experiments B and C use the values obtained by accumulating the probability values from each policy and taking an average when agents select the action. This paper examines the Sharpe ratio and portfolio value for three experiments.

B. Experimental Results

Fig. 4 shows the visualization of the feature extraction process of the CNN layer. It shows the extracted features of the CNN layer when a candlestick image is input. In the experiment, the convolution layer and the max-pooling layer are alternately used, without using the max-pooling layer only at the end. As shown in Fig. 4, the features are extracted through the CNN layer according to the direction of the graph included in the image.

Table III shows the results of the experiment when 256 items were invested from April 1, 2019 to April 26, 2019 based on the learned policy. Experiments have yielded a higher average return than the KOSPI growth rate although there are some variations. Depending on the type of experiment, the results calculated from (2) show somewhat large deviations, but there is no significant difference in the minimum rate of change. Because the transaction is based on the daily closing price data, even if the agent predicts the upward trend and buy it, it cannot sell at the high price point and the loss can occur.

TABLE III. INVESTMENT RESULTS FOR APRIL 2019

Type of Experiment	Average Return (%) [Std. Dev.]	Maximum Return (%)	Minimum Return (%)	Sharpe Ratio
Exp. A	1.77 [9.48]	98.38	-12.13	0.13
Exp. B	3.06 [11.31]	112.27	-12.05	0.27
Exp. C	0.89 [7.67]	109.745	-13.22	0.05

The Sharpe ratio characterizes how well the return of an asset compensates the investor for the risk taken. When comparing two assets versus a common benchmark (i.e., KOSPI growth rate), the one with a higher Sharpe ratio provides better return for the same risk. A negative Sharpe ratio means the portfolio has under-performed its benchmark. As can be seen in Table III, the Sharpe ratio is positive in all three experiments, so that investment returns can be expected. Experiment B showed the highest return on investment.

Table IV shows the results of the experiment when 256 items were invested from May 2, 2019 to May 30, 2019 based on the learned policy. During the test period, KOSPI fell 7.8%, and stock prices of most stocks fell. Therefore, the average rate of return for all three experiments was negative. However, when using the proposed learning model, we can see that the loss is less than the drop rate of KOSPI. Especially, in Experiment C, the Sharpe ratio is 2.77 because the loss is only 0.8%. Therefore, we can see that performance is excellent both in bear and bull market.

TABLE IV. INVESTMENT RESULTS FOR MAY 2019

Type of Experiment	Average Return (%) [Std. Dev.]	Maximum Return (%)	Minimum Return (%)	Sharpe Ratio
Exp. A	-4.68 [6.10]	21.15	-29.60	0.51
Exp. B	-6.81 [7.77]	25.40	-34.45	0.13
Exp. C	-0.82 [10.8]	10.80	-23.83	2.77

IV. CONCLUSIONS

ML make it easier to identify patterns that might otherwise not be detected by the human eye. In this paper, we proposed a multimodal reinforcement learning policy combining CNN and LSTM for stock trading. The proposed machine learning system generated various charts from stock trading data and used them as inputs to the CNN layer. The features extracted through CNN are divided into column vectors and input to the LSTM layer. In order to utilize various types of charts such as candle stick, DMI, and SSO chart, a multimodal structure is used in which the input layer and the hidden layer are separated, and the individual operation results are combined. For reinforcement learning, we defined agents' policy neural network structure, reward, and action, and provided buying, selling, and holding probabilities as final output. In order to verify the performance of the proposed policy, we conducted training and testing with the daily data of the 256 stocks listed on the Korea KOSPI. Data from March 2009 to March 2019 were used as training data. When April and May 2019 data were

used as a test, the Sharpe Ratios, which indicates investment performance, were positive for all six experiments.

Although there are more than 25 indicators used for technical analysis, only a few indicators such as MA, DMI and SSO were used in this study. In the future, we plan to include various indicators in the proposed learning model to improve prediction performance.

REFERENCES

[1] Tadas Baltrusaitis, Chaitanya Ahuja, and Louis-Philippe Morency, "Multimodal Machine Learning: A Survey and Taxonomy," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, issue 2, pp. 423–443, February 1 2019.

[2] Mo-Se Lee and Hyunchul Ahn, "A Time Series Graph based Convolutional Neural Network Model for Effective Input Variable Pattern Learning: Application to the Prediction of Stock Market," The Journal of Intelligence and Information Systems, vol. 24 No. 1, pp. 167–181, 2018

[3] Jae Won Lee, "A Stock Trading System based on Supervised Learning of Highly Volatile Stock Price Patterns," KIISE Transactions on Computing Practices, vol. 19, no. 1, pp. 23–29, January 2013.

[4] Il-Taeck Joo and Seung-Ho Choi, "Stock Prediction Model based on Bidirectional LSTM Recurrent Neural Network," Journal of Korea institute of information, electronics, and communication technology, vol. 11, no. 2, pp. 204–208, 2018.

[5] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan, "Deep Learning for Event-Driven Stock Prediction," Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015), pp.2327–2333, 2015.

[6] Ryo Akita, Akira Yoshihara, Takashi Matsubara, and Kuniaki Uehara, "Deep learning for stock prediction using numerical and textual information," 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), June 2016.

[7] Jae Won Lee, "Stock price prediction using reinforcement learning," Proceedings of 2001 IEEE International Symposium on Industrial Electronics, June 2001.

[8] Mauricio García-Galicia, Alin Carsteanu, and Julio Clempner, "Continuous-time reinforcement learning approach for portfolio management with time penalization," Expert Systems with Applications, vol.129, pp. 27-36, September 2019.

[9] Jae Won Lee, Jonghun Park, Jangmin O, Jongwoo Lee, Euyseok Hong, "A Multiagent Approach to Q-Learning for Daily Stock Trading," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 37, issue 6, pp. 864-877. November 2007.

[10] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas, "Dueling network architectures for deep reinforcement learning", Proceedings of the 33rd International Conference on International Conference on Machine Learning, vol. 48, pp. 1995-2003, June 2016.

[11] Van Hasselt, Hado, Arthur Guez, and David Silver, "Deep reinforcement learning with double q-learning", Thirtieth AAAI Conference on Artificial Intelligence, pp. 2094-2100, 2016.