Thesis for Bachelor's Degree

Effective reinforcement learning for highly distributed data through K-mean clustering

Lee, Hojoon (이 호준 李 鎬俊)

School of Electrical Engineering and Computer Science

Gwangju Institute of Science and Technology

2023

Effective reinforcement learning for highly distributed data through K-mean clustering

K-mean clustering과 Deep Q-Network를 이용한 고분산 주식 데이터 강화 학습

Effective reinforcement learning for highly distributed data through K-mean clustering

Advisor	:	Professor	Jeon, Hae-Gon
Co-Advisor	:	Professor	Kim, Kyung-Joong

by

Lee, Hojoon

School of Electrical Engineering and Computer Science
Gwangju Institute of Science and Technology

A thesis submitted to the faculty of Gwangju Institute of Science and Technology in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical Engineering and Computer Science

Gwangju, Republic of Korea

2021. 09. 04.

Approved by

Professor Jeon, Hae-Gon

[Advisor]

Effective reinforcement learning for highly distributed data through K-mean clustering

Lee, Hojoon

Accepted in partial fulfillment of requirements
for the degree of Bachelor of Science

December. 09. 2022.

BS/EC
20185151

Lee, Hojoon (이 호준). Effective reinforcement learning for highly distributed data through K-mean clustering (K-mean clustering과 Deep Q-Network를 이용한 고분산 주식 데이터 강화 학습). School of Electrical Engineering and Computer Science. 2023. 12p. Advisors Prof. Jeon, Hae-Gon, Prof. Kim, Kyung-Joong.

Abstract

Many papers studied stock data using various methods. From the existing traditional method to artificial intelligence, stocks have been studied in various ways such as deep learning and reinforcement learning. Stock data is data with a lot of uncertainty, so we wondered whether stock data could learn more stable by clustering it together with data with similar characteristics. In this paper, stock data was preprocessed through k-mean clustering, and created DQNs corresponding to each cluster and learn separately. The number of clusters for k-mean clustering is determined by the elbow method and trained DQNs suggest the best action to take at any state. we used data extracted per minute from the KODEX 200 ETF as a learning target, and for an evaluation, we first compare the learned clustered DQNs with two policy that takes random action or buy throughout the whole step to check whether clustered DQNs is properly trained or not. And to check the effectiveness of clustering, we compare clustered DQNs with DQN without clustering. As a result, clustered DQNs perform better than buy-only policy and random policy, and while non-clustered DQN could not learn properly, in the case of clustered DQNs, it performed well.

Contents

Chapter 1. Introduction

For modern people, stocks are a close field that is really hard to separate. Huge stock markets influence people every day. It is easy to find evidence for it. In 2021, the U.S. stock market was the world's No. 1 stock market with 30.43 trillion dollars. Second place went to China with 6.3249 trillion dollars, and third place was Japan with 5.29 trillion dollars. And Korea ranked 10th with 1.4412 trillion dollars. [1] In the case of Korea, the transaction volume, which was 29,804,702 million won in January 2016, was 37,540,262 million won in January 2018 and 44,828,149 million won in January 2022. Like this, stocks are an area of interest over time. [2]

So there were many attempts to study it. With the invention of many learning techniques, the field of stocks is being studied in more various ways. Stock research is usually done in a way that provides users with choices that are ideal to make at a particular point in a situation. Eventually, the goal of all stock research is to maximize revenue over the same period. Stocks are generally very noisy and have a lot of uncertainty. Stocks are also difficult data to handle in learning as they are not simply influenced by stock trade, but also by social phenomena, current government policies, international situations, the psychology of traders, and other markets. [3]

Nevertheless, stocks are an area where various approaches have been attempted and are showing results despite various difficulties. There are papers that researched stocks using LSTM [4], CNN [5] to predict price fluctuations, and there are ways that use RL to select the action to be taken per state [6]. For this paper, the study is based on Deep Q-Network(DQN).

DQN is a type of reinforcement learning(RL). RL is a machine learning technology inspired by behavioral psychology, and there are a few basic components. RL has an environment and a definition of the state, the agent selects a specific action according to the state, and the state changes due to the action triggered by the agent and receives a reward from the environment. And the purpose of RL is to maximize the return, which is the sum of all the rewards. Therefore, in order to study stocks with DQN, it is necessary to convert and fit stocks to the necessary elements in RL.

In addition, we will use k-mean clustering to cluster the data and aim to improve performance with a cluster-specific DQN where the model doesn't have to represent the whole data.

The contributions to this paper are as follows.

1. It provides a DQN model that could train cluster-specific DQN. We selected the DQN model from Chen et al. (2019) [6] as the base DQN for the paper and modified the model to perform a clustering version.

2. It shows how efficient clustering is in studying stock data.

The remaining paper is organized as follows. Chapter 2 explains which related paper will be used for this paper and the description of each technology. Chapter 3 explains how the cluster-specific DQNs model is organized in detail and explains the training phase. Chapter 4 explains the result of this paper and Chapter 5 sums up the paper and suggests future work.

Chapter 2. Background and Related work

2.1 k-mean clustering

K-mean clustering is one of the most widely known clustering methods, and it gives an unsupervised, non-deterministic solution by an iterative method. K-mean clustering aims to find the best centroid for the number of clusters given by the hyper-parameter and to bind the data accordingly.

$$SSE = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left| x_j - \mu_i \right|^2 \tag{2.1}$$

The goal is to minimize SSE which is the sum of the square errors of the data and its related centroid where k is a number of clusters and S_i is a set of data in the cluster i and x_j is data in the cluster and μ_i is the centroid of the cluster.

The steps for obtaining the solution are as follows:

First, the k-mean clustering algorithm initializes the first k centroid randomly.

$$i = \mathrm{argmin}_i \left| x_p - \mu_i \right|^2, x_p \in S_i \tag{2.2}$$

$$\mu_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j \tag{2.3}$$

For each iteration, the cluster for each data is reassigned to minimize the distance from the data to the centroid, and the centroid of each cluster is computed by the data. This process will be repeated until there is no further change or change smaller than the threshold. [7]

For the k-mean clustering algorithm, there is no optimal answer for which value k should be as a hyper-parameter. So there are many algorithms to find a suitable k value, and the elbow method is one of the most popular methods. The elbow method calculates a sum of square error (SSE) for the different number of clusters and selects the one with the greatest SSE difference. [8]

2.2 deep Q-network

DQN stands for deep Q-network, a method introduced by Mnih et al. (2015). [9] This was introduced by applying deep learning to predict the Q-value in Q-learning [10] and applying techniques such as separating the target network from Q-network and using replay buffer fixing the problems that Q-learning had in the past.

DQN originated from Q-learning which is a method of calculating Q-value, the expected value of the return obtained when the action is taken in the state, and the agent taking the action with the largest Q-value at each state according to the correctly calculated Q-value.

$$Q(s,a) = E\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s, a_t = a, \pi\right] \tag{2.4}$$

$$Q^*(s,a) = \max_\pi E\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s, a_t = a, \pi\right] \tag{2.5}$$

Optimal Q-value Eq. 2.5 for policy π is computed by the Bellman optimality equation.

$$Q(s,a) = Q(s,a) + \alpha\left(Q^*(s,a) - Q(s,a)\right) \tag{2.6}$$

The Q-learning method has two major drawbacks: the observed results affect the Q-network immediately and Q-network is the same as the target network so the target continues to change. This makes the Q-learning method difficult to learn well. Also, once the observed results are used, they are discarded immediately after learning, making it difficult and inefficient to learn the Q-network.

DQN handled both of the problems. DQN solved the first problem by using the different weights for the target network which calculates the optimal value with the weights of the Q-network. DQN used different periods for the Q-network and the target network to update its weights so two weights can be different and the weights of the target network slowly follow the weights of the Q-network.

In addition, DQN solved the disadvantage of using the observed result only once by storing the observed results used to train the network for the first time in the replay buffer and randomly selecting them again when training the network.

Therefore, loss function for DQN is as follows.

$$L_i(\theta_i) = E_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q\left(s', a'; \theta_i^-\right) - Q(s,a;\theta_i)\right)^2\right] \tag{2.7}$$

where $U(D)$ is batch sampled from the replay buffer, γ is the discount rate, and θ_i^-, θ_i each stand for the weights of the target network and the weights of Q-network.

2.3 traditional stock papers with RL

Chen et al. (2019) [6] is a paper that applied DQN to stock data. This paper studied the 19 years of daily data of the S&P 500 ETF. The states were defined based on the daily closing price and the actions were daily choices whether to sell, buy or do nothing, and the rewards were profit from the action.

For Conegundes et al. (2020), [11] actions are the ratio of the holding N stocks in each time step, and the information from the agent's point of view is only past stock information, so the environment is partially observable MDP. The states are the relative prices, which is the ratio of the closing price and opening price of each stock in the past for a certain period of time. Profits from the action are used as rewards and trained the model with DDPG using five years of BOVA11 data.

For Liu et al. (2018), [12] they selected 30 Dow Jones stocks and studied them with 10 years of data. The states are the price, holding amount, and cash holding amount of the current stock, and the actions are the changes in the holding amount of each stock, and the rewards are the profit from each action and trained with DDPG.

2.4 sharpe ratio

Sharpe ratio [13] is a representative method of evaluating investment performance along with the information ratio, Treynor ratio, and Sortino ratio, The Sharpe ratio is as follows:

$$S_a = \frac{E\left[R_a - R_b\right]}{\sigma_a} \tag{2.8}$$

where R_a is the return on the investment, R_b is the return on the benchmark portfolio and σ_a is the standard deviation of the return on the investment.

Sharpe ratio is an indicator of how much the return on investment is for risk. This means that the investment method with a higher Sharpe ratio earns more profit for the same return on the benchmark portfolio, which means higher returns. Sharpe ratio equates volatility with risk, so an investment strategy gets a higher Sharpe ratio when it makes a more stable result

Chapter 3. Methodology

3.1 model

Although it is difficult to consider external factors that affect stocks such as politics, and other markets, there is still a lot of information about stocks that we can use. Various information such as opening price, closing price, high price, low price, trading volume, and MACD can be used to construct the elements necessary for RL.

3.1.1 state

States were constructed based on the closing price, as shown by Chen et al. (2019). We also set the length of states to 20. In this paper, the states will be classified through k-mean clustering, and if it is clustered without preprocessing, it will be classified according to the price range of the stock which does not show the characteristic of the states well. Since the purpose of k-mean clustering is to cluster data that show similar trends, the relative differences between 1 day ago 20 days ago closing price and the 21 days ago closing price were used as the states.

3.1.2 action

There are three actions; buying, selling, and holding. In this paper, in order to simplify the model, we set it up to only buy and sell the whole thing when buying or selling.

3.1.3 reward

The reward is the profit earned by the action.

3.2 hyper-parameter

There is a lot of hyperparameter in this model. Q-network and target network consist of 4 fully connected layers. The number of features is (20, 10), (10, 10), (10, 10), and (10, 3), respectively. It uses the Relu function for the non-linear activation function. The policy seems like a random policy at the beginning and slowly turns into a learned policy. ϵ represents the probability of the network policy behaving like a random policy. ϵ starts from 1 and slowly exponentially decreases until it reaches 0.01 as the network gets trained. Also, we used Adam optimizer and implemented a learning rate scheduler so the learning rate decayed from 0.001 to 0.0001. We used discount factor γ as 0.79 which is suggested by Chen et al. (2019). The length of the states is 20. We stored 50000 observations in the replay buffer and selected uniformly random 100 samples to train the Q-network. The weights of the target network are updated every 250 steps as the weights of the Q-network. We used the elbow method to determine the suitable number of clusters. In the case of KODEX 200, the suitable number of clusters was 7.

3.3 algorithm

Algorithm 1 deep Q-network with k-mean clustering

```
Use elbow method to get $k$
Apply k-mean clustering to $Data$ and get $Data_i$ where $1 \leq i \leq k$
for label=1, k do
    Initialize the weight of the Q-network $\theta$
    Initialize the weight of the target network $\theta^- \leftarrow \theta$
    Initialize replay memory $D$

    for update=1, update_limit do
        $E$ provides random $s \in Data_{label}$

        while episode not finished do
            if $\epsilon$ then
                $a_{curr} \leftarrow$ random action
            else
                $a_{curr} \leftarrow \text{argmax}_a Q(s, a; \theta)$
            end if
            Agent do $a_{curr}$, $E$ gives $r$, $s'$
            save $(s, a_{curr}, s', r)$ in $D$
            if $s' \in Data_{label}$ then
                episode not finished
            else
                episode finished
            end if
            $s \leftarrow s'$
        end while
            update $\leftarrow$ update + len(episode)
            Select random batch $B$ from $D$
            Gradient descent for $L(\theta) = E_{(s,a,r,s')\sim B}\left[\left(r + \gamma \text{max}_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\right)^2\right]$
            $\theta^- \leftarrow \theta$ once in a certain period of update
        end for
    end for
```

3.4 data and training

We used KODEX 200 data recorded per minute from 2021-03-19,11:47:00 to 2021-11-25,15:30:00. The data is shown in Fig 3.1.

We used 80% of data for training and 20% of data for testing.

Fig 3.2 shows some examples of the training phase of DQN for clustered data and the training phase of DQN for non-clustered data. Fig 3.2 shows that the network learns better with clustered data than with non-clustered data.

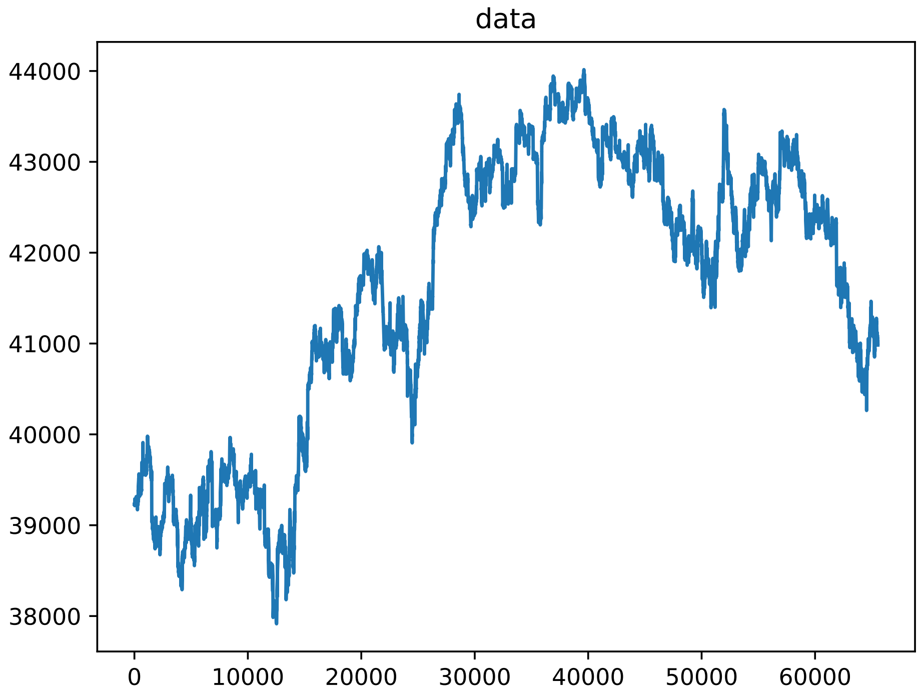
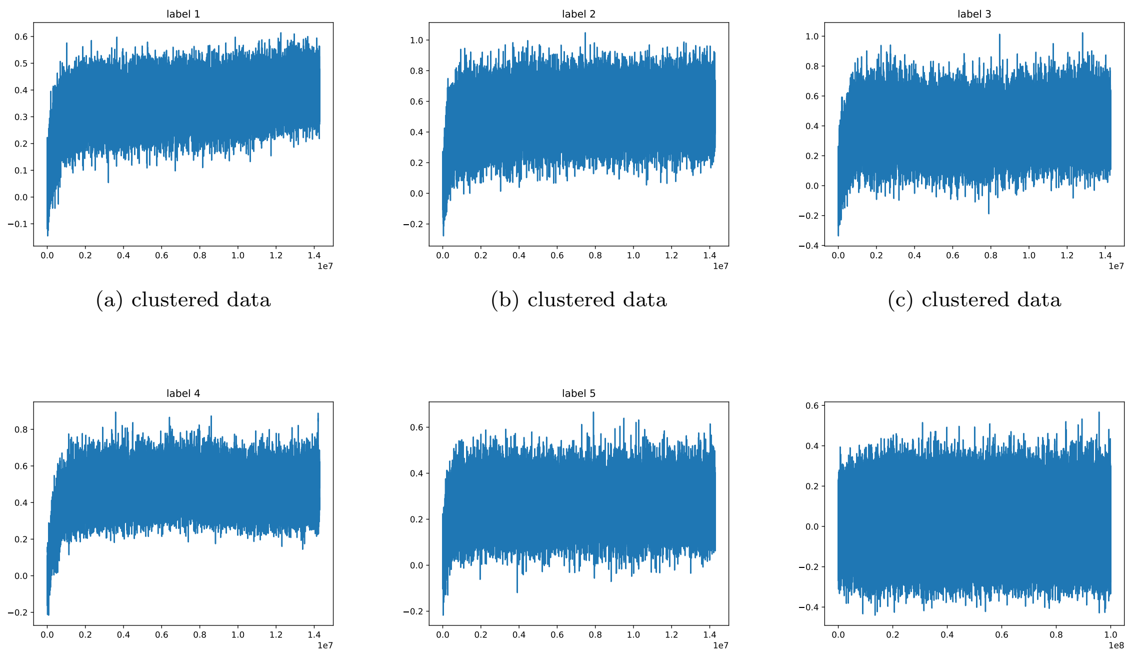


Figure 3.1: KODEX 200 data recorded per minute



(a) clustered data

(b) clustered data

(c) clustered data

(d) clustered data

(e) clustered data

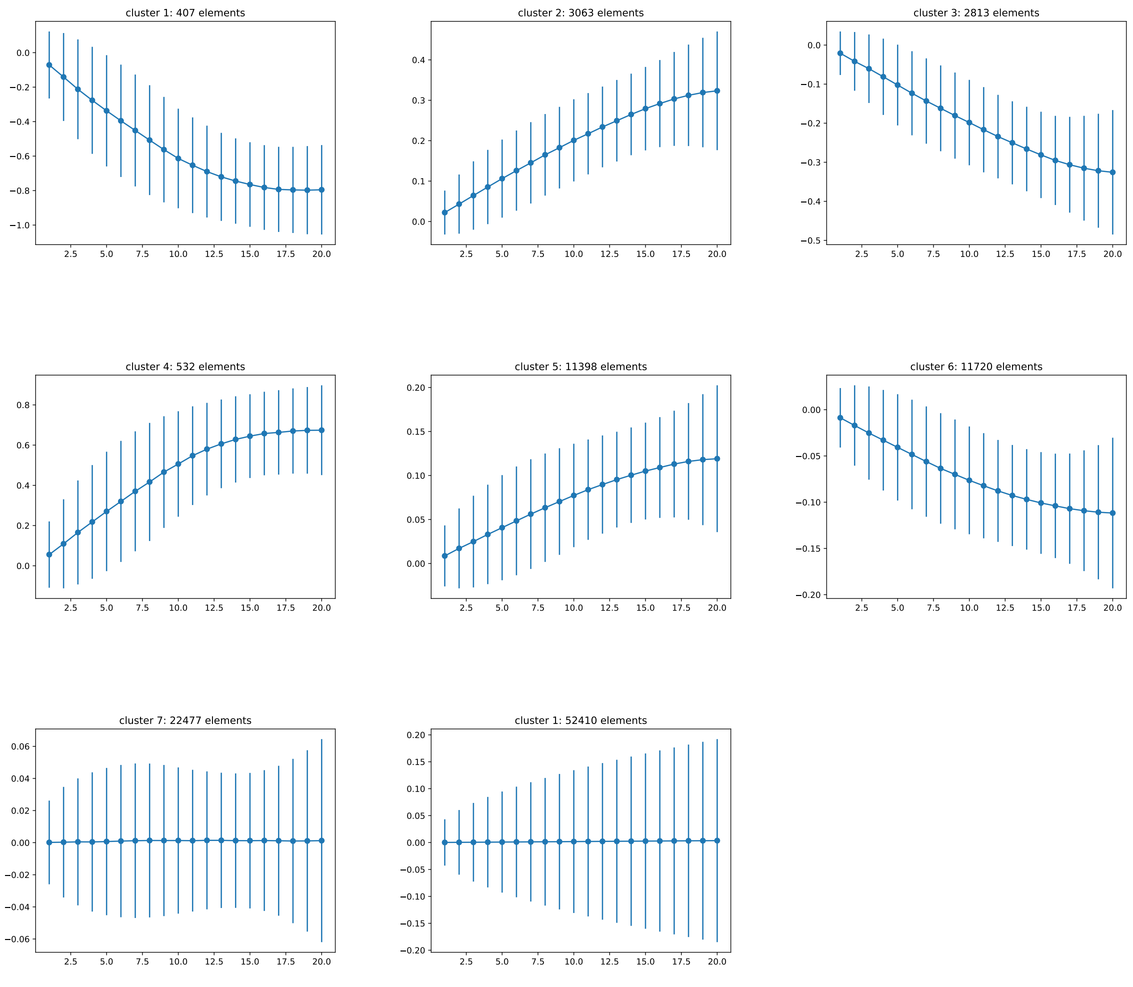
(f) non-clustered data

Figure 3.2: training process

Chapter 4. Experiment Result

4.1 data distribution

The first 7 figures in Fig 4.1 show how data is distributed for each cluster separated by k-mean clustering. There is one cluster that does not show a trend in data, and the rest is a cluster that tends to increase or decrease. Compared with the last figure in Fig 4.2 which shows data distribution for non-clustered data, it can be seen that k-mean clustering significantly distinguished the characteristics of the data.



non-clustered data

Figure 4.1: data distribution of data

4.2 algorithm

Algorithm 2 evaluation

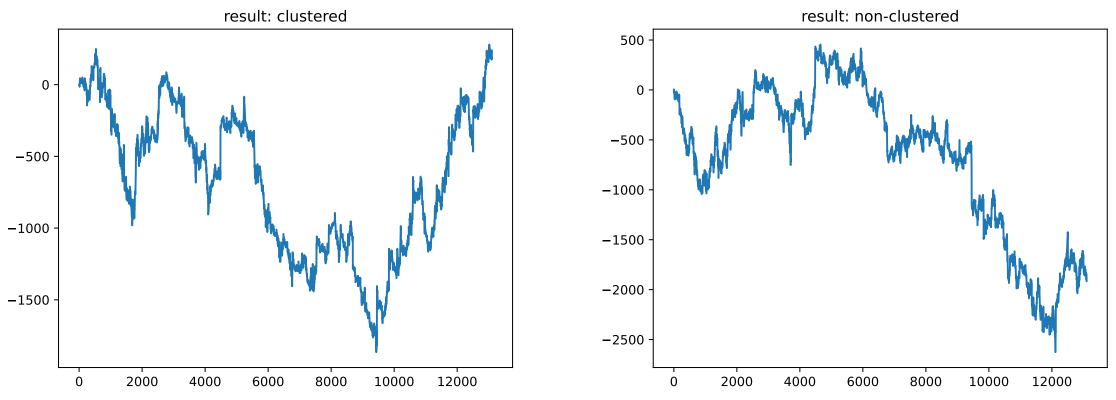
```
Set the length of test data as $n$
Set the number of clusters as $k$
return $\leftarrow 0$
**for** curr data=1, n **do**
    Classify current data to the certain cluster using MSE
    Load the weight of the model for the label
    $a_{curr} \leftarrow \text{argmax}_a Q(s, a)$ for the label
    Agent do $a_{curr}$, $E$ gives $r$, $s'$
    $s \leftarrow s'$
    return $\leftarrow$ return $+ r$
**end for**
```

4.3 analysis

To evaluate the result, we first compare each state in the test set with all the centroids of the clusters and determine which cluster the state should belong to. And then for each state, we select the appropriate Q-network and compute the action and reward according to it. Lastly, we calculate cumulative profit and compare it with Q-network that only learns from non-clustered data.

Fig 4.2 shows a cumulative profit for each time step to measure the performance of DQNs trained with clustered data and non-clustered data. For networks that used the clustering method, the yield is 238.0, on the other hand, for networks that did not use the clustering method yield is -1915.0. Also, the Sharpe ratio is calculated by Eq 2.8 and the network trained with clustered data got 122.840992 points for the Sharpe ratio while the network trained with non-clustered data got -1.192604.

Table 4.1 shows the test results according to various policies. K-mean clustering DQN also shows better performance compared to random policy and buy-only policy.



(a) clustered data (b) non-clustered data

Figure 4.2: data distribution of data

policy	k-mean clustering DQN	random(av. 100 iter)	buy-only
return	238.0	-193.76	-1895.0

Table 4.1: return due to policy

We conducted a test to see if the performance improvement was actually due to the clustering. We obtained test statics based on the results of five experiments, each with and without clustering, and the t-value was 6.012 which is similar to when the degree of freedom is 8 and α is 0.00015. So it rejects the null hypothesis. This means that clustering data had a positive effect on training the Q-network.

Chapter 5. Conclusion and Future work

In this work, we presented a model to solve the problem by combining the k-mean clustering algorithm with DQN. We show that clustering of similar data using clustering algorithms such as k-mean algorithms makes learning easier and produces better results than the network trained with non-clustered data even in a difficult environment. In particular, in this paper, we used stock data per minute as a metric to evaluate the effectiveness of the algorithm. It also had the advantage of being able to learn it in parallel because the model was different for each cluster.

We made a few limitations in the environment such as simply using the closing price and limiting the action. As a future study, it will get more meaningful results by substituting the clustering method into a more general situation. It would also be a good study to use fuzzy mean clustering instead of k-mean clustering or DDPG instead of DQN.

References

1. https://eiec.kdi.re.kr/publish/naraView.do?fcode=00002000040000100012&amp;cidx=13365.
2. 한국예탁결제원,「증권예탁통계」, 2022.09, 2022.11.27, 매매결제 현황 ; 유가증권시장 상장증권 결제. https://kosis.kr/statHtml/statHtml.do?orgId=348&tblId=TX_34801_A013&conn_path=I2
3. Y. Zhang and L. Wu, “Stock market prediction of S&amp;P 500 via combination of improved BCO approach and BP Neural Network,” Expert Systems with Applications, vol. 36, no. 5, pp. 8849–8854, 2009.
4. R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, “Deep learning for stock prediction using numerical and textual information,” 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), 2016.
5. M. U. Gudelek, S. A. Boluk, and A. M. Ozbayoglu, “A deep learning based stock trading model with 2-D CNN trend detection,” 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017.
6. L. Chen and Q. Gao, “Application of deep reinforcement learning on Automated Stock Trading,” 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), 2019.
7. S. Na, L. Xumin and G. Yong, ”Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm,” 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, 2010, pp. 63-67, doi: 10.1109/IITSI.2010.74.
8. P. Bholowalia and A. Kumar, ”EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN,” 2014 International Journal of Computer Applications, 2014, pp. 17-24
9. V. Mnih, K. Kavukcuoglu and D. silver, ”Human-level control through deep reinforcement learning,” Nature 518, 2015, pp. 529-533
10. C. J. Watkins and P. Dayan, “Q-learning,” Machine Learning, vol. 8, no. 3-4, pp. 279–292, 1992.
11. L. Conegundes and A. C. Pereira, “Beating the stock market with a deep reinforcement learning day trading system,” 2020 International Joint Conference on Neural Networks (IJCNN), 2020.
12. X. Liu, Z. Xiong, S. Zhong, H. Yang and A. Walid, ”Practical deep reinforcement learning approach for stock trading,” arXiv preprint arXiv:1811.07522, 2018.
13. The Sharpe ratio. [Online]. Available: https://web.stanford.edu/ wfsharpe/art/sr/SR.htm.

Summary

Effective reinforcement learning for highly distributed data through K-mean clustering

많은 논문들이 다양한 방법을 이용해서 주식이라는 데이터를 연구했습니다. 기존의 전통적인 방법부터 인공지능이 등장하고 나서는 deep learning, reinforcement learning 등 주식을 다양하게 연구했습니다. 이 때 주식 데이터는 매우 불규칙적인 데이터인데 이를 비슷한 특징의 데이터끼리 묶으면 더 안정적으로 학습을 할 수 있지 않을까라는 의문이 들었습니다. 이 논문에서는 주식 데이터를 k-mean clustering을 통해서 전처리 과정을 거치고 각각의 cluster에 해당하는 DQN를 만들어서 각각을 따로 학습을 시켰습니다. k-mean clusering은 elbow method에 의해서 clustering 하고 DQN은 매 순간 어떤 행동을 취해야 하는지 알려줍니다. KODEX 200 ETF의 1분 단위의 데이터를 학습 대상으로 사용했고 평가는 먼저 DQN의 학습 정도를 알아보기 위해서 학습된 DQN과 전체 기간 동안 매수만 했을 때, 임의의 action을 취할 때를 비교합니다. 그리고 clustering의 효과를 확인하기 위해서 clustering을 한 DQN과 clustering을 하지 않은 DQN을 비교합니다. 그 결과 clustering 된 DQN은 buy-only policy와 random policy보다 성능이 좋다는 것을 알 수 있었습니다. 또한 clustering을 하지 않으면 학습이 잘 이루어 지지 않지만 clustering된 DQN의 경우 유의미한 학습이 이루어 진 것을 확인할 수 있었습니다.

감 사 의 글

논문 작성을 처음 접해서 방황하고 헤매던 제게 조언을 아끼지 않으신 전해곤 지도교수님과 김경중 공동 지도교수님께 감사드립니다. 또한 지금까지 저를 사랑으로 이끌어주신 부모님께 감사드립니다. 이외에도 제게 도움을 주신 모든 분들께 감사드립니다.

약 력

이 름:	이호준
생 년 월 일:	1999년 3월 24일
출 생 지:	서울특별시
주 소:	서울특별시 강서구 공항대로39길 100

학 력

2015. 3. – 2018. 2.	마포고등학교
2018. 2. – 2023. 2.	광주과학기술원 전기전자컴퓨터공학부 (학사)