IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 9, Issue 3, March 2020

Reinforcing Portfolio Management through Ensemble Learning

Satyam Kumar[1], Jayesh Bapu Ahire[2], Atharva Abhay Karkhanis[3], Ishana Vikram Shinde[4]

Department of Computer Engineering, Sinhgad College of Engineering (SPPU), Pune, India[1,2, 3, 4]

Abstract: It has been observed that "Stereoscopic Portfolio Optimisation Frameworks" introduce the concept of bottom-up optimisation through the utilization of machine learning ensembles applied to some market micro-structure element. But in contrast to the normal belief, it doesn't always pan out as expected. One of the popular and widely used "Deep Q-Learning" algorithms is quite unstable due to the shake in the Q-values and also due to the fact that over-estimation action values under certain conditions. These issues tend to affect their performance adversely. Inspired by the breakthroughs in DQN and DRQN, we suggest a modification to the last layers, to handle pseudo-continuous action spaces, as required for the portfolio management task. The implementation used currently, called as the "Deep Soft Recurrent Q-Network (DSRQN)" is dependent on a fixed and implicit policy. In this paper, we have described and developed an Ensembled Deep Reinforcement Learning architecture based on implementation of temporal ensemble, in order to stabilize the training process, achieved by reducing the variance of target approximation error. As a result of ensembling the target values, overestimation is reduced and it also makes the performance better by estimating more accurate Q-value. Our aggregate architecture leads to more accurate and optimized statistical results for this classical portfolio management and optimization problem.

Keywords: Temporal Ensemble, Reinforcement Learning, Deep Learning, Finance Technology, Algorithmic Trading

I. Introduction

Reinforcement Learning (RL) is appropriate for learning to adapt and regulate associate agent by material possession. RL adapts according to the associate atmosphere. Quite recently, Deep Neural Networks (DNN) have been incorporated into Reinforcement Learning, in return they have achieved an excellent success on approximation of the value function. Mnih et al introduced the first Deep Q-network (DQN) algorithm which successfully integrated a powerful nonlinear function approximation technique known as DNN along with the Q-learning algorithm. In this paper, we are introducing an experience replay mechanism. Following the DQN work, a variety of solutions have been proposed to stabilize the recently introduced algorithms. The Deep-Q Networks have achieved unforeseen success in challenging areas like the Atari 2600 and few different games.

Although DQN algorithms resolve several issues due to their powerful perform approximation ability and strong generalization between similar state inputs, they are still not capable in resolution of some problems.

Following primary reasons explain the occurrence of the issues above:

(1) Randomness of the sampling leads to very varying environment and a serious shock.

(2) Errors of such systematic nature lead to instability, poor performance, as well as divergence in learning.

The averaged target DQN (ADQN) algorithm is implemented, in order to address these issues and construct target values by combining target Q-networks continuously with a single learning network. The Bootstrapped DQN algorithm is introduced to get more efficient exploration and better performance with the parallel use of several Q-networks learning. Even though these algorithms do scale back the overestimate, they are not capable in assessing the importance of the past learned networks. Besides, there is still a higher variance in target values combined with the max operator. There are some ensemble algorithms solving this issue in Reinforcement Learning, however these existing algorithms don't seem to be compatible with nonlinearly parameterized value functions.

In this paper, we propose the ensemble algorithm as a solution to this current incompetency. In order to increase the final performance and learning speed, we integrate multiple Reinforcement Learning algorithms into a single agent with several ensemble algorithms, thereby to evaluate the actions or action probabilities. In supervised learning, ensemble algorithms such as bagging, boosting, and mixtures of experts are often used for learning and combining multiple classifiers. But in Reinforcement Learning, ensemble algorithms are used for representing and learning the value function.

DOI 10.17148/IJARCCE.2020.9313

Based on an Associate in nursing agent integrated with multiple Reinforcement Learning algorithms, multiple price functions' area units are learned at the identical time. The ensembles combine the policies derived from the worth functions in an exceedingly final policy for the agent. The bulk pick (MV), the rank pick (RV), the Boltzmann multiplication (BM), and also the Boltzmann addition (BA) area unit want to mix RL algorithms. Whereas these ways are area unit pricey in deep reinforcement learning (DRL) algorithms, we tend to mix totally different DRL algorithms that learn separate price functions and policies. Therefore, in our ensemble approaches we tend to mix the various policies derived from the update targets learned by deep Q-networks, deep Sarsa networks, double deep Q-networks, and totally different DRL algorithms. As a consequence, this leads to reduced overestimations, plenty stable learning technique, and improved performance.

II. Reinforcement Learning Applied to Finance

There are many previously published papers which have used Reinforcement Learning in Stock Trading, managing and optimising the Portfolio. Moody et al. Was one of the early birds in applying the RL paradigm to the problem of stock trading and portfolio optimization. He has put forward the idea of Recurrent Reinforcement Learning (RRL) for Direct Reinforcement. RRL, is an adaptive policy search algorithm, that is capable of learning an investment strategy online. Direct Reinforcement was a term coined to highlight the algorithms that don't need to learn a value function for deriving a policy. To put it in simpler words, policy gradient algorithms belonging to a Markov Decision Process framework are generally referred to as Direct Reinforcement. Moody et al. has demonstrated that we can formulate a differential form of the Sharpe Ratio and Downside Deviation in order to enable Direct Reinforcement to be efficient in online learning.

David W. Lu introduced the idea of incorporating LSTM learning agent along with Direct Reinforcement, to learn how to trade in a Forex and commodity futures market. Du et al. used Q-Learning which is a value function-based algorithm, to achieve algorithmic trading. He evaluated the performance of the approach using different forms of value functions like Sharpe Ratio, Derivative Sharp Ratio and Interval Profit.

Tang et al. used an actor-critic based portfolio investment method taking into consideration the risks involved in asset investment. The paper introduced by him setups a Markov Decision model. This model is for the multi-time segment portfolio with transaction cost. This has been implemented through approximate dynamic programming.

Jiang et al. in his one of the first papers which provides a detailed Deep Reinforcement Learning framework which can be used in the task of Portfolio Management in a crypto-currency market exchange. He has introduced a network called Ensemble of Identical Independent Evaluators (EIIE), trained with the concept of a Portfolio Vector Memory. They take into consideration market risks and the transaction costs associated with buying and selling assets in a stock exchange.

III. Recurrent Reinforcement Learning

We have considered the decision-making of investment developed by J. Moody and M. Saffell as a stochastic problem in this approach. We have identified the strategies independently. Instead of using direct enforcement approaches, which try to estimate a value function for the control problem, they have introduced incorporation of Dynamic programming and enhancement algorithms like TD-learning and Q-learning. This prevents Bellman's dimensionality and offers convincing efficiency benefits by facilitating the representation of the problem through the RRL Direct Reinforcement Framework. How can direct reinforcement be used to optimize risk-adjusted returns on investment, taking account costs, has been demonstrated by them. They use real financial information intra-daily and find that their RRL-based approach produces better trade strategies than Q-learning systems.

Steve Y. Yang and Saud Almahdi are also taking another approach to solving optimal asset allocation problems and a number of trading decision schemes based on methods of enhanced learning. An optimum allocation of variable weights in line with a consistent downside risk measure (MDD) has been established by them. Their method is specified by the Calmar Ratio, using the RRL method for both buying and selling signals and asset allocation weights, along with a performance goal that has been consistently risk-adjusted. The expected maximum risk downward-focused objective function is shown through the most frequently traded exchange fund's portfolio as a higher return than previously proposed RRL functions (i.e. Sharpe or Sterling Ratio), and in various scenarios of transactions cost equal portfolios, the variable weight portfolios.

Deep learning (DL) combined with reinforcement learning in the work of Deng et al., introduced a recurrent deep neural network (NN) for real-time financial signal representation and trading. DL decides to accumulate ultimate income in an unknown environment by automatically detecting the dynamic market conditions for informative learning, after the RL module has interacted with deep representations. A complex NN with a highly recurring structure performs the system of learning. A time-based task-back-cutting to tackle the problem of deep training slowdown has thus been proposed by them. The stock and commodity markets under wide-ranging test conditions confirm the strength of the neural system.

DOI 10.17148/IJARCCE.2020.9313

Dynamic programs and strengthening algorithms, such as TD-learning and Q-learning, which try to approximate calculate a value function for the control problem, are very different from the RRL approach. With the RRL the dimensionality of Bellman is avoided. In RRL framework, simple, elegant problem representation is created. Efficiency of RRL offers compelling advantages when compared to Q-learning when exposed to rowdy data sets. RRL has a more stable performance when compared to others. Due to the recursive dynamic optimization property, Q-learning algorithm is more sensitive to selecting the value. On the other hand, RRL algorithms can save time by choosing the objective function.

IV. Ensemble Methods for Deep Reinforcement Learning

DQN classes have strong generalization ability between similar state inputs, due to their use of DNNs to approximate the value function. Divergence is caused in the case of repeated bootstrapped temporal difference updates due to the generalisation. This issue can be solved by integrating different versions of the target network.

Ensemble algorithms have proven to be more effective, in contrast to a single classifier. Ensemble algorithms lead to a higher accuracy. Multiple classifiers can be trained by methods like Bagging, Boosting, and Ada Boosting. In case of RL, ensemble algorithms are used for representing and learning the value function. RL and Ensemble Learning are combined by major voting, Rank Voting, Boltzmann Multiplication, mixture model, and other ensemble methods. Classification accuracy can be significantly improved, if the errors of the single classifiers are not strongly correlated.

V. The Ensemble Network Architecture

The temporal and target values ensemble algorithm (TEDQN) which we are proposing in this paper is an integrated architecture inspired by the value-based DRL algorithms. The ensemble network architecture has two parts to avoid divergence and improve performance as discussed earlier.

Our ensemble algorithm architecture is shown in the Figure 1 mentioned below. The two parts have been combined together by evaluated network.

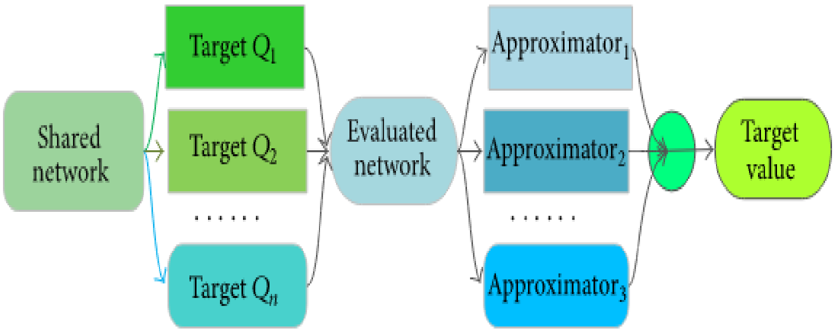
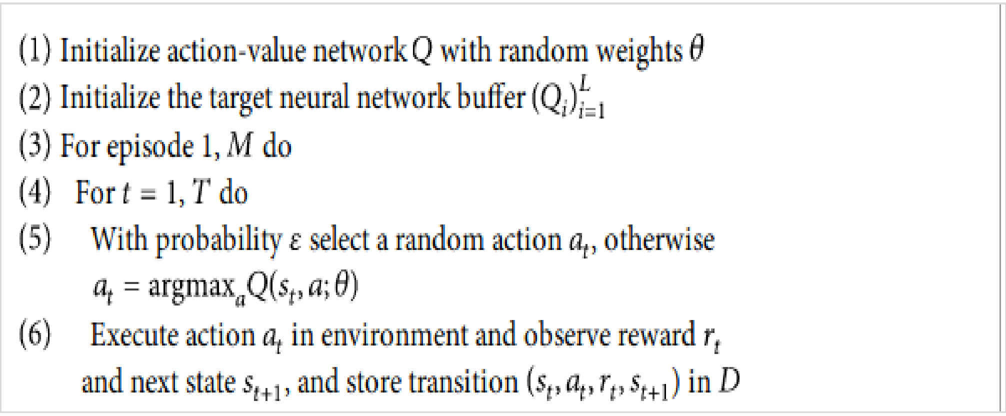
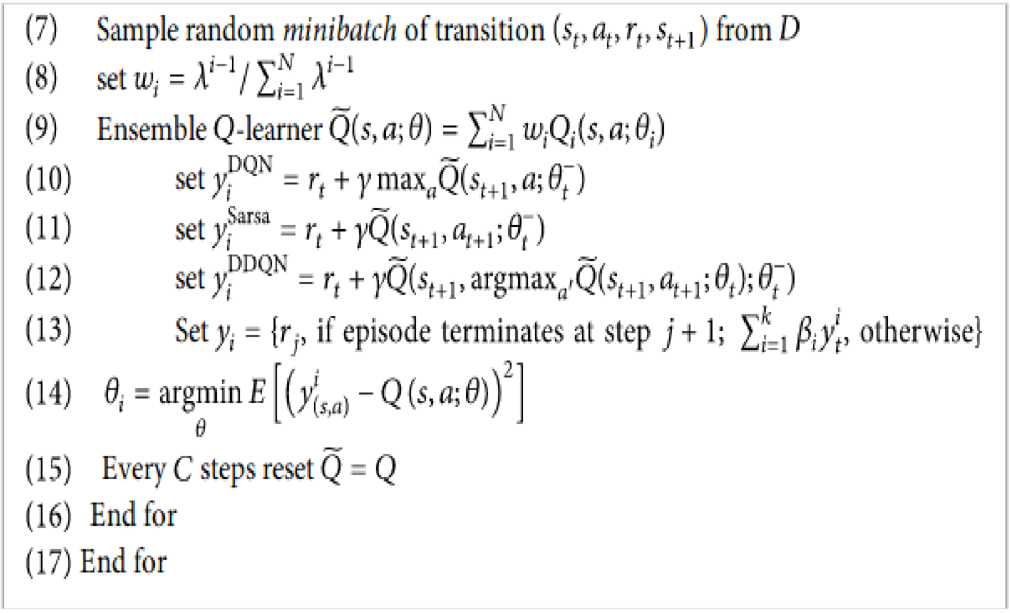


Fig. 1: The architecture of the ensemble algorithm

The training process is stabilized, by reducing the variance of target approximation error, through the temporal ensemble [10]. The overestimate is reduced and performance is made better by estimating more accurate Q-value, through the ensemble of target values. Algorithm 1 gives the temporal and target values ensemble algorithm.



```
(1) Initialize action-value network Q with random weights $\theta$
(2) Initialize the target neural network buffer $(Q_i)_{i=1}^{L}$
(3) For episode 1, $M$ do
(4)   For $t = 1, T$ do
(5)     With probability $\varepsilon$ select a random action $a_t$, otherwise
        $a_t = \text{argmax}_a Q(s_t, a; \theta)$
(6)     Execute action $a_t$ in environment and observe reward $r_t$
        and next state $s_{t+1}$, and store transition $(s_t, a_t, r_t, s_{t+1})$ in $D$
```



```
Sample random minibatch of transition $(s_t, a_t, r_t, s_{t+1})$ from $D$
set $w_i = \lambda^{i-1} / \sum_{i=1}^{N} \lambda^{i-1}$
Ensemble Q-learner $\widetilde{Q}(s, a; \theta) = \sum_{i=1}^{N} w_i Q_i(s, a; \theta_i)$
        set $y_i^{\mathrm{DQN}} = r_t + \gamma \max_a \widetilde{Q}(s_{t+1}, a; \theta_t^-)$
        set $y_i^{\mathrm{Sarsa}} = r_t + \gamma \widetilde{Q}(s_{t+1}, a_{t+1}; \theta_t^-)$
        set $y_i^{\mathrm{DDQN}} = r_t + \gamma \widetilde{Q}(s_{t+1}, \mathrm{argmax}_{a'} \widetilde{Q}(s_{t+1}, a_{t+1}; \theta_t); \theta_t^-)$
        Set $y_i = \{r_j$, if episode terminates at step $j+1$; $\sum_{i=1}^{k} \beta_i y_t^i$, otherwise$\}$
$\theta_i = \underset{\theta}{\mathrm{argmin}}\, E\left[\left(y_{(s,a)}^i - Q(s, a; \theta)\right)^2\right]$
Every $C$ steps reset $\widetilde{Q} = Q$
End for
End for
```

Fig. 2: The temporal and target values ensemble algorithm

As the ensemble network architecture shares the same input-output interface with standard Q-networks and target networks, we can recycle all learning algorithms with Q-networks to train the ensemble architecture.

VI. CONCLUSION

In this paper, we introduced a new learning architecture, making temporal extension as well as the ensemble of target values for deep learning algorithms, at the same time sharing a generic learning module. The newly introduced ensemble architecture, results in dramatic improvements over existing approaches for Deep Reinforcement Learning in the challenging classical control issues, along with some algorithmic improvements. In practice, due to the fact that it can be easily integrated to the RL methods based on the approximate value function, this ensemble architecture can be very convenient.

Even though ensemble algorithms are superior to a single Reinforcement Learning algorithm, it is to be noted that the computational complexity is higher. The experiments we conducted show that the ensemble of a variety of algorithms makes the estimation of the value more accurate and that temporal ensemble makes the training process more stable, and. Due to the fact that ensembles improve independent algorithms most if the algorithms predictions are less correlated, the combination of the two ways of Ensemble Learning and RL, enables the training to achieve a stable convergence. The output of the network based on the choice of action can thus achieve balance between exploration and exploitation.

The performance for ensemble algorithms is very much dependent on the independence of the ensemble algorithms and their elements. For future work, we want to analyse the role of each algorithm and each network in different stages, so as to further enhance the performance of the ensemble algorithm.

REFERENCES

[1]. S. Mozer and M. Hasselmo, “Reinforcement learning: an introduction,” IEEE Transactions on Neural Networks and Learning Systems, vol. 16, no. 1, pp. 285-286, 2005. View at Publisher · View at Google Scholar

[2]. L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: a survey,” Journal of Artificial Intelligence Research, vol. 4, pp. 237–285, 1996. View at Google Scholar · View at ScopusR. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.

[3]. V. Mnih, K. Kavukcuoglu, D. Silver et al., “Playing Atari with deep reinforcement learning [EB/OL],” https://arxiv.org/abs/1312.5602.

[4]. M. A. Wiering and H. van Hasselt, “Ensemble algorithms in reinforcement learning,” IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 38, no. 4, pp. 930–936, 2008. View at Publisher · View at Google Scholar · View at Scopus

[5]. S. Whiteson and P. Stone, “Evolutionary function approximation for reinforcement learning,” Journal of Machine Learning Research (JMLR), vol. 7, pp. 877–917, 2006. View at Google Scholar · View at MathSciNet

[6]. P. Preux, S. Girgin, and M. Loth, “Feature discovery in approximate dynamic programming,” in Proceedings of the 2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2009, pp. 109–116, April 2009. View at Publisher · View at Google Scholar · View at Scopus

[7]. T. Degris, P. M. Pilarski, and R. S. Sutton, “Model-Free reinforcement learning with continuous action in practice,” in Proceedings of the 2012 American Control Conference, ACC 2012, pp. 2177–2182, June 2012. View at Scopus

[8]. V. Mnih, K. Kavukcuoglu, D. Silver et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015. View at Publisher · View at Google Scholar · View at Scopus

DOI 10.17148/IJARCCE.2020.9313

[9]. H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-Learning," in Proceedings of the 30th AAAI Conference on Artificial Intelligence, AAAI 2016, pp. 2094–2100, February 2016. View at Scopus

[10]. O. Anschel, N. Baram, N. Shimkin et al., "Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning [EB/OL]," https://arxiv.org/abs/1611.01929.

[11]. I. Osband, C. Blundell, A. Pritzel et al., "Deep Exploration via Bootstrapped DQN [EB/OL]," https://arxiv.org/abs/1602.04621.

[12]. S. Faußer and F. Schwenker, "Ensemble Methods for Reinforcement Learning with Function Approximation," in Multiple Classifier Systems, pp. 56–65, Springer, Berlin, Germany, 2011. View at Google Scholar

[13]. A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 4–37, 2000. View at Publisher · View at Google Scholar · View at Scopus

[14]. T. Schaul, J. Quan, I. Antonoglou et al., "Prioritized Experience Replay [EB/OL]," https://arxiv.org/abs/1511.05952.

[15]. I. Zamora, N. G. Lopez, V. M. Vilches et al., "Extending the OpenAI Gym for robotics: a toolkit for reinforcement learning using ROS and Gazebo [EB/OL]," https://arxiv.org/abs/1608.05742.

[16]. D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," Journal of Machine Learning Research (JMLR), vol. 6, no. 2, pp. 503–556, 2005. View at Google Scholar · View at MathSciNet

DOI 10.17148/IJARCCE.2020.9313