Exploring the Application of Reinforcement Learning and Policy Methods in Multi-Armed Bandit Problems

Abstract

This project delves into the challenging domain of competitive Multi-Armed Bandit (MAB) problems, as exemplified by the Kaggle "Santa 2020 - The Candy Cane Contest". Our aim is to explore optimal decision-making strategies using reinforcement learning methods such as Q-learning, A2C, and policy-based strategies. Traditional approaches in reinforcement learning face difficulties with the inherent randomness of the 100 bandits in each round. To address this, our project combines Q-learning with the Beta distribution, showing improved performance. Although A2C, known for its application in complex state spaces, is adapted to the MAB context by incorporating state information, it faces challenges in convergence. This suggests that for a relatively simple problem like this, A2C might not be the best fit. However, policy-based strategies prove to be the most effective, leveraging historical information, win-loss records, and the continuous decrease in winning probability. The study proposes further exploration into machine learning predictors, time series methods, and hybrid approaches for enhanced performance in solving MAB problems, as demonstrated in the Santa 2020 challenge on Kaggle.

Key words: Multi-Armed Bandit (MAB), Reinforcement Learning, Q-learning, A2C (Advantage Actor-Critic), Policy-Based Strategies

1 Introduction

In the realm of probabilistic models and game theory, the multi-armed bandit problem presents a quintessential challenge for decision-making under uncertainty. This project embarks on an exploration of a complex variant of this problem instantiated in a scenario involving 100 automated vending machines. Each machine operates on a distinct probability distribution, dispensing candies as rewards to participating agents. As players engage with the machines over a series of rounds, their interactions are governed by the principle that selecting a vending machine consequently reduces its probability of reward by 3% in the following round. This dynamic introduces a strategic depth to the game, where each of the two intelligent agents must navigate through 2000 rounds of play, summing up to a total of 4000 interactions. Crucially, the agents are privy to their opponent's choices, albeit remaining oblivious to the actual reward acquisition, injecting an element of imperfect information into the strategic mix. This project attempts to use methods

such as reinforcement learning to construct an agent capable of making optimal decisions. Through multiple trials, we aim to gain a deeper understanding of reinforcement learning and the characteristics of this problem. We experimented with Q-learning, A2C, and other non-traditional reinforcement learning methods, only to find that reinforcement learning does not perform well on this problem. This may be attributed to the characteristics of the problem itself, as each iteration of the game generates a new set of 100 multi-armed bandits.

2 Method

There are numerous methods that could be applied to this problem. It is intuitive to consider some greedy algorithms, such as selecting the optimal solution based on the average value of rewards from each machine in the past, or employing the Epsilon-Delta Strategy to attempt to resolve a trade-off dilemma between "exploitation" and "exploration". We could also use well-known multi-armed bandit approaches like UCB (Upper Confidence Bound) or Thompson Sampling. However, the performance of these methods in this particular problem has not been satisfactory. Thus, we have attempted to address this issue from the perspective of reinforcement learning.

2.1 Q-learning

The status of each round of slot machines is unrelated to the previous game, and the effect obtained from simple Q-learning is very poor. In the context of the Multi-Armed Bandit problem, each arm can be considered a binary process (success or failure), making the Beta distribution highly suitable for modeling this scenario. Consider combining it with the beta distribution. Updating the parameters of the Beta distribution using Q-learning. The status of each round of slot machines is unrelated to the previous game, and the effect obtained from simple Q-learning is very poor. In the context of the Multi-Armed Bandit problem, each arm can be considered a binary process (success or failure), making the Beta distribution highly suitable for modeling this scenario. Consider combining it with the beta distribution. Updating the parameters of the Beta distribution using Q-learning.

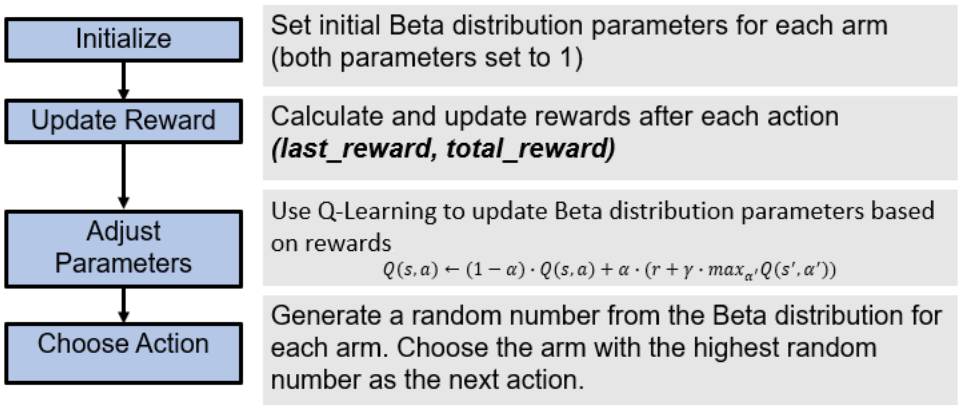

Figure 1: The Process of Q-learning Algorithm

2.2 A2C

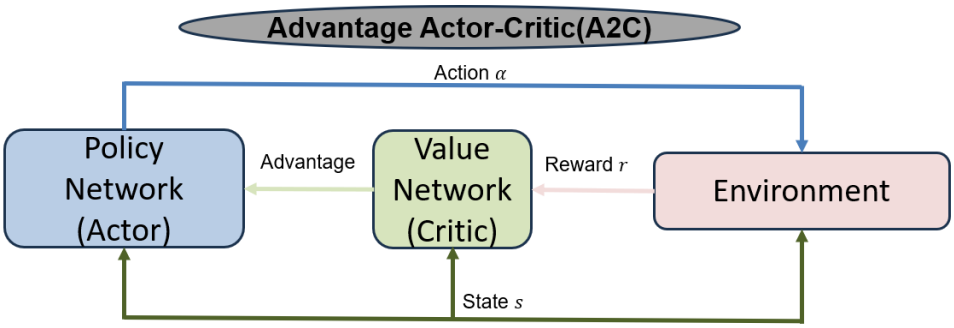

Figure 2: Actor Critic Process

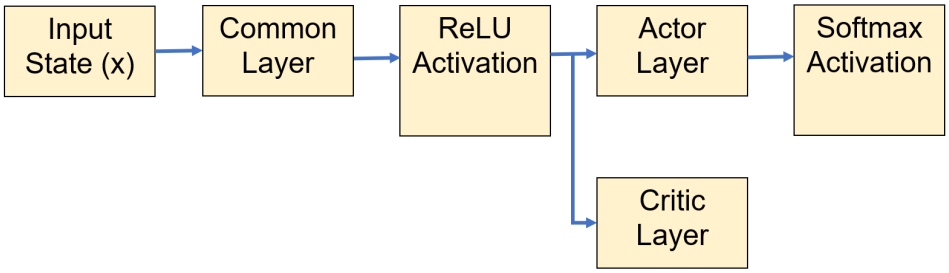

Figure 3: The Structure of A2C Algorithm

The Advantage Actor-Critic (A2C) algorithm is a reinforcement learning method typically used for problems with complex state spaces. In the context of Multi-Armed Bandit (MAB) problems, the application of A2C may not be as intuitive or common as in more complex environments, since the MAB problem is fundamentally a simplified decision-making task without state transitions or dynamic environment changes. However, if we slightly extend the MAB problem to include state information, the A2C algorithm can become applicable.

2.3 Policy Based

We can fully utilize the available historical information by tracking the win-loss record of each arm and the historical behavior of ourselves and our opponents, as well as utilizing the information that the probability of winning decreases by 3% with continuous selections. By doing so, we can fully optimize our strategy to make more informed decisions.
Maintain a bandit dictionary: Calculate the expected utility of each arm of the multi-armed bandit, taking into account the number of times the arm has won, the number of losses, the number of times the opponent has chosen that arm, and the frequency of past selections.
Choice of the next action: Balance exploration and exploitation, and take into account previous action choices.

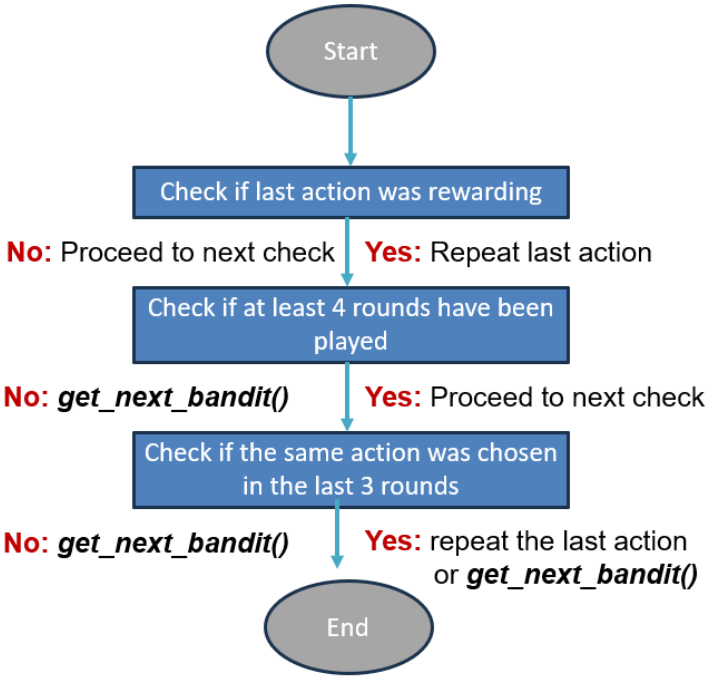

Figure 4: Strategy Logic

3 Results

3.1 Q-learning

We first experimented with Q-learning, and after running 5000 local iterations, we used the saved Q-values to initialize our agent's Q-values. However, when we tested this in battles, the performance was very poor. We attributed this to the fact that each of the 100 slot machines in every round is random. Consequently, we considered combining the Beta distribution with the Q-learning update strategy, which improved the performance beyond what was achieved using the Beta distribution alone.

Table 1: Parameters

Parameter	Value
`epsilon`	0.2
`alpha`	0.1
`gamma`	0.9
`step`	1.0
`decay_rate`	0.97

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\delta_t$$
$$\delta_t = r_t + \gamma \max_{a'} Q\left(s_{t+1}, a'\right) - Q(s_t, a_t)$$

3.2 A2C

We aim to train optimal parameters that allow the rewards to converge, and then input these final parameters into the agent for battle. The network is quite challenging to converge. Initially, we attempted training for 1000*2000 rounds but observed unsatisfactory results. We then conducted further training for 500*2000 rounds, only to find that the rewards still experienced significant fluctuations. Given the limitations in computing resources and time, we believe that this method might not be the best choice for such a simple problem context. We are confident that more training rounds could achieve convergence, but for a problem like this, it may not be worth the effort.

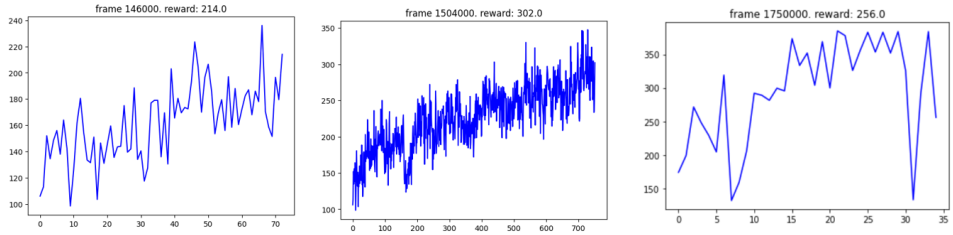

Figure 5: A2C Training Convergence Curve

3.3 Policy Based

This method is the best performing one we have tried so far. By assigning different weight parameters to various historical information and through parameter tuning, we have made full use of the historical data. We have developed a better method for calculating the expected value of each slot machine, which is used to determine the current choice.

3.4 Comparison of Models (win rate over 100 games)

Table 2: Comparison of Models

Model	vs. Random	vs. Pure Beta Distribution	vs. Policy Based
Pure Beta Distribution	100%	-	19%
Beta Distribution+Q-Learning	100%	56%	24%
Policy based	100%	100%	-

4 Discussion

4.1 Distinguishing Features of Competitive Multi-Armed Bandits from MDPs

Dynamic Reward Structure: Selecting a machine reduces its reward probability, introducing a dynamic reward system that contrasts with the static reward structures typically found in MDPs. Incomplete Information: Agents observe each other's choices but lack

information about obtained rewards, adding a layer of incomplete information not present in the fully observable environments assumed in MDPs. Strategic Complexity: Strategy formulation must account for long-term gains and the potential reactions of opponents, rather than just maximizing immediate returns, which is the focus in traditional MDP optimization.

4.2 Other potential methods to explore

Machine Learning Predicting Strategy: We can use a machine learning model(Linear Regressor, Random Forest Regressor, LightGBM) to predict the original probability. However, this method may have higher requirements for feature selection. Time Series Predicting Method: We can consider this problem as a time series issue, utilizing exponential smoothing and ARIMA, and applying their formulas for prediction. Hybrid Approaches: We could also combine some approaches: for instance, integrating the UCB strategy into an ML model, incorporating the UCB strategy into time series models, or employing transformers to optimize deep reinforcement learning networks.

5 Conclusion

For a problem with such a simple background, both deep reinforcement learning and traditional reinforcement learning methods suited for Markov Decision Processes have not shown strong performance. This could be due to suboptimal choices in state representation, training parameters, and network architecture, or it might require more training rounds. However, it's evident that for such problems, constructing a good update strategy based on the problem itself can also yield very effective results.

References

[1] Vivanti, R., "Can Q-learning solve Multi Armed Bantids?", 2021. doi: 10.48550/arXiv .2110. 10934.
[2] Manickam, A. S. Lan and R. G. Baraniuk, "Contextual multi-armed bandit algorithms for personalized learning action selection," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 2017, pp. 6344-6348, doi: 10.1109/ICASSP.2017.7953377.
[3] T. B. F. de Oliveira, A. L. C. Bazzan, B. C. da Silva and R. Grunitzki, "Comparing Multi-Armed Bandit Algorithms and Q-learning for Multiagent Action Selection: a Case Study in Route Choice," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 2018, pp. 1-8, doi: 10.1109/IJCNN.2018.8489655.
[4] https://hrl.boyuai.com
[5] https://towardsdatascience.com
[6] https://www.kaggle.com/code/ddrbcn/drl-mab