DEEP REINFORCEMENT LEARNING ALGORITHMS APPLIED TO SNAKE GAME

Filippo Boni

1 INTRODUCTION

The goal of the classic Snake videogame is to increase the length of the snake in order to cover the totality of the game board and finally win. The snake grows by eating fruits placed randomly on the board and avoiding decreasing its length by eating himself or by hitting the wall. The main challenge of the game is to manage the space of the board efficiently in order to reach the fruit without decreasing the length of the snake. This obstacle becomes more and more difficult to tackle with the growth of the snake. In this work we apply Deep Reinforcement Learning agents to the Snake videogame and study in which measure they are able to tackle the challenge just explained. In particular we study the performances of three among the most used Deep RL algorithms: PPO, DQN and A2C. These three algorithms are also compared to a not learning baseline which has been developed considering the environment representing the videogame.

2 ENVIRONMENT

The board of the game in which the snake moves has dimension 5×5 and its surrounded by a layer of wall, making the final size of the game 7×7. When the fruit it is eaten, a new fruit is generated randomly in the one of the available cells. A visual representation of the described game environment is given in Fig 1. The possible movements(actions) are: UP, $DOWN$, $RIGHT$ and $LEFT$. To each possible outcome of an action is connected a reward. The rewards are summarized in Table 1.

3 BASELINE

As explained in the previous sections, the snake grows by eating the fruit. In addition, as it is possible to see from Table 1, eating the fruit is the goal that leads to the highest reward(apart from winning). Thus moving closer and closer to the fruit in the board could be a good strategy. The baseline developed is based on this hypothesis. By looking at the board, it finds the relative position between the head of the snake and the fruit. In particular the fruit can be upper-right, upper-left, lower-right or lower-left with respect to the head of the snake. The baseline algorithm then selects an action based on the identified zone. If the fruit is in an ordinal direction (e.g., upper-right), two actions are possible (e.g., UP or RIGHT), and the baseline randomly chooses between them using a uniform distribution. When the head and the fruit share the same latitude or longitude, only one action is possible. Table 2 shows a summary of the baseline.

Outcome of Action	Reward
WIN	1
EATING FRUIT	0.5
EATING BODY	-0.2
HITTING WALL	-0.1
NORMAL STEP	0.

Table 1: Rewards summary

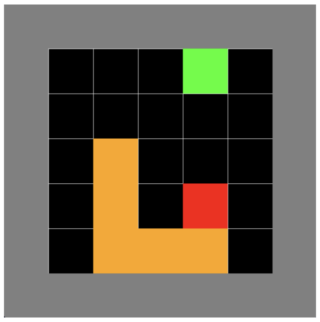

Figure 1: Example of a game iteration

Position Fruit	Action
(UP,RIGHT)	random(UP,RIGHT)
(DOWN,RIGHT)	random(DOWN,RIGHT)
(UP,LEFT)	random(UP,LEFT)
(DOWN,LEFT)	random(DOWN,LEFT)
UP	UP
DOWN	DOWN
RIGHT	RIGHT
LEFT	LEFT

Table 2: Baseline Schema

4 DEEP REINFORCEMENT LEARNING ALGORITHMS

4.1 Advantage Actor Critic(A2C)

Advantage Actor Critic(A2C) is an Actor Critic method which uses the Advantage Function to reduce the variance of the learning process. The Actor is in charge for updating the policy weights while the Critic has the goal to learn the Advantage Function. Among the methods that learn the policy, A2C is one of the simplest still very robust in performances and in the theoretical foundations, thus it is a good choice for a comparison with other algorithms.

4.2 DQN

Deep Q-Network (DQN) is a powerful algorithm in the field of reinforcement learning. It combines the principles of deep neural networks with Q-learning. It is a value based method which means that it learns weights in order to learn the value function of the problem. Then the action are chosen with the $\varepsilon - greedy$ method. DQN is one of the most used Deep RL algorithms, but being a value-based method it is very different from A2C, thus suitable for a comparison.

4.3 PPO

Proximal Policy Optimization, or PPO, is a policy gradient method. Like A2C it uses an Actor to learn the policy and a Critic to learn the Advantage function, but it uses a different cost function for the Actor. The modified cost function improves the learning stability by preventing too large policy updates. This is done by clipping the value of the function, which is finally formulated as following:

$$J(\theta) = \mathbb{E}_t \left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t) \right] \tag{1}$$

where $r_t(\theta)$ is the importance sampling ratio, A_t is the Advantage function and ε is a hyperparameter.

5 THE EXPERIMENTAL SET-UP

The algorithms were trained over 10000 iterations, each presenting a batch of 1000 boards. The neural network used for the actors, the critics of the A2C and PPO and for the online network and target network of the DQN share the same architecture (excluding the last layer, which is adapted according to the output needed). The architecture is the following:

- **1 Convolutional Layer:** 2D, 3 × 3 kernel (5 × 5 for PPO), 64 filters, ReLU
- **2 Dense Layers:** 64 units, ReLU
- **Batch Normalization:** Between the layers
- **1 Dense Layer:** For the output

A convolutional layer is introduced in order to learn the spatial features of the game, such as the position of the snake and its body on the board or the distance between the snake and the fruit. In the case of the PPO the training loop for the actor is repeated 5 times for each iteration.

6 RESULTS

In order to evaluate the algorithms we focus on two values: the reward, because it measures how good an RL agent is doing, and the length of the snake, in order to see if the agent pursues the final goal of the game that is growing as much as possible. This two measures are averaged along the number of boards for every iteration. In addition we analyze the number of boards in which: a fruit has been eaten, a body piece has been eaten, a wall has been hit. The value of the baseline for each metric is represented by the mean value over all the iterations.

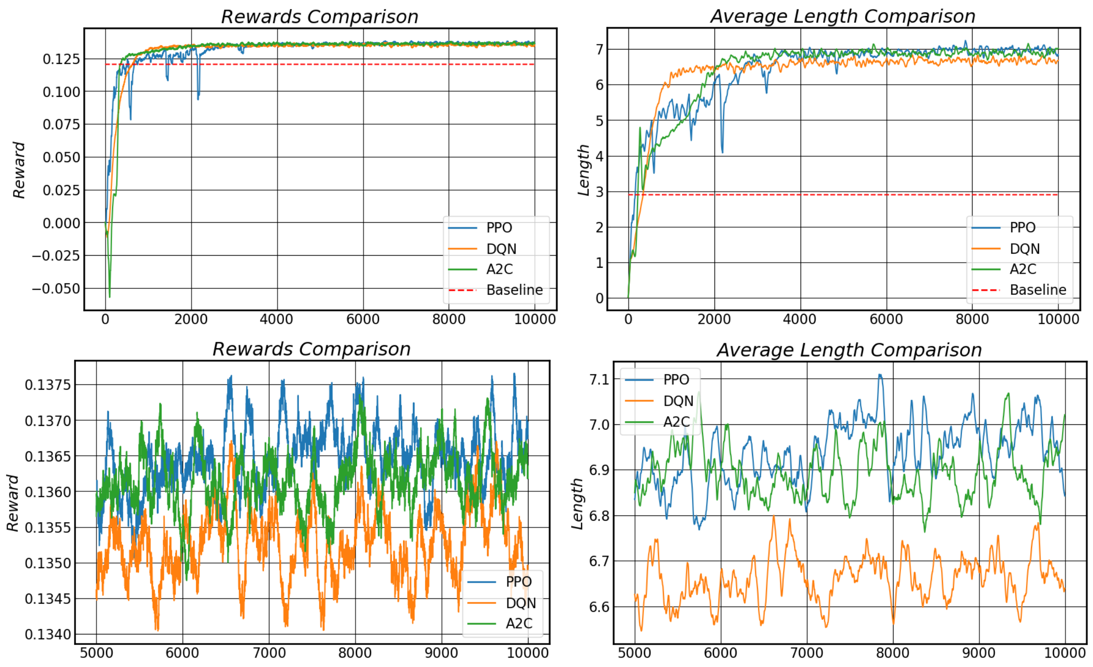

Figure 2: (top-left) reward comparison RL agents and baseline, (top-right) average length comparison RL agents and baseline, (bottom-left)RL agents reward comparison, (bottom-right)RL agents average length comparison

First of all the Deep RL agents can beat the baseline. Not only their reward is higher than the baseline one, but they reach an average length close to 7, which is more than twice the one of the baseline.
It is quite interesting to see that the number of eaten fruits is very similar between the Deep RL agents and the baseline(Fig 3 top left). In addition the plot of the wall hit (Fig 3 top right) highlights that the baseline does not hit the wall. Lets remember that the baseline moves the snake as close as possible to the fruit, which results in a high number of eaten fruits and in the impossibility of hitting the wall. This explains why the reward of the baseline is not very far from the RL agents ones(even if still very different) since eating the fruit is the action that leads to the highest reward (apart from winning) and hitting the wall results in a negative reward. On the other hand the difference in terms of length is very big. Thus eating as much fruits as possible and not hitting the wall are necessary but not the sufficient conditions for growing as much as possible, which is the primary goal of the game. The only metric that shows an extreme difference between the baseline and the agents is the number of eaten bodies(Fig 3 top center). Thus learning to avoid eating himself is what makes the crucial difference.
It is also evident that A2C and PPO achieve better results than the DQN, so policy learning based algorithms seem to be a better option than value based. By analyzing the plots starting from the 5000^{th} iteration(Fig 2 bottom) it is possible to see that DQN tends to generate a lower reward and length. Indeed it tends to eat less fruits, more bodies and to hit more walls than the other two agents (Fig 3 bottom). On the other hand it is quite difficult to choose between the A2C and the PPO. Even if the metrics are very similar, PPO tends to generated a bit longer snakes and to have a slightly higher reward. Probably because the number of eaten bodies is lower than the A2C one for the majority of the iterations(Fig 3 bottom center). Even if we have highlighted the differences between the Deep RL algorithms, they all achieve good results. The number of eaten fruits is higher than the baseline, even if close. Which is good since the baseline is built to eat as much fruits as possible in less moves possible. The number of wall hit is very low: close to 0(Fig 3 bottom right). As already said, the real advantage of the RL agents is learning to avoid the body.

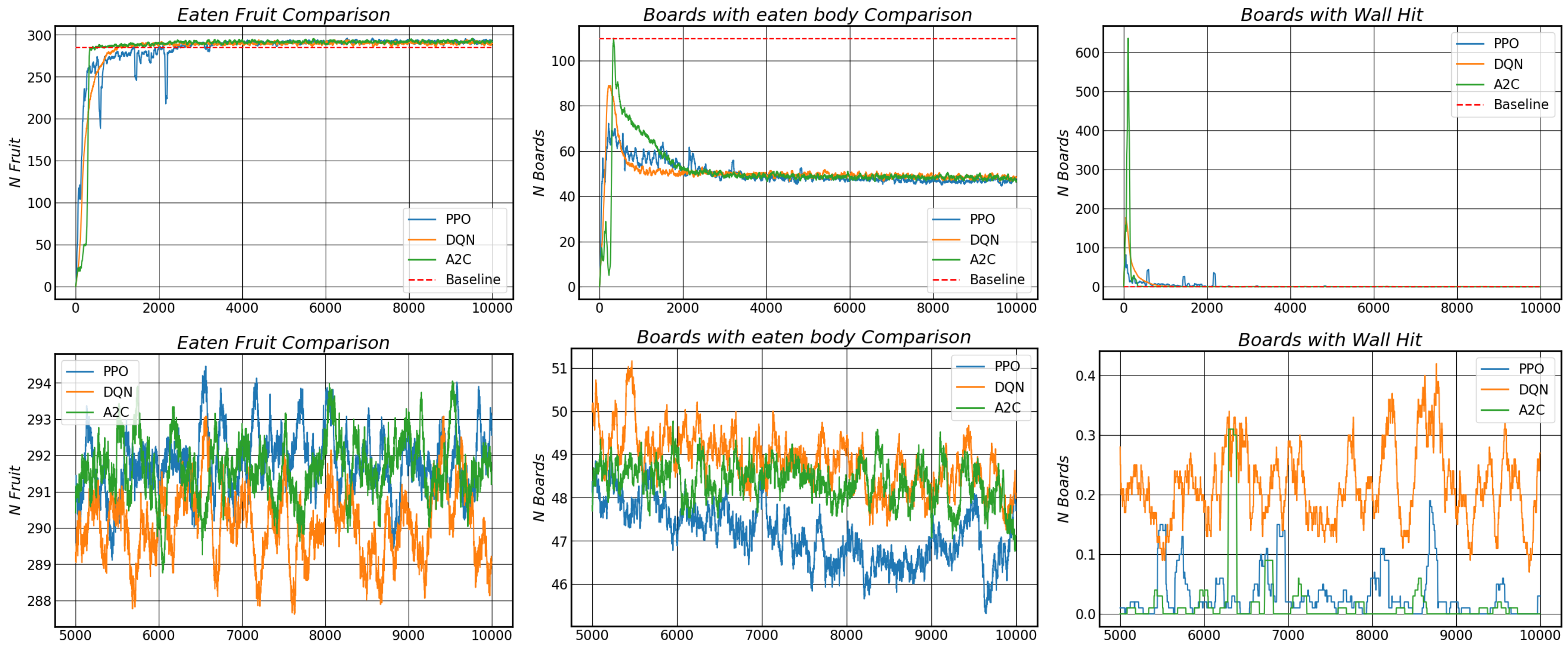

Figure 3: (top-left) eaten fruit comparison RL agents and baseline, (top-center) eaten body comparison RL agents and baseline, (top-right) wall hit comparison RL agents and baseline, (bottom-left) eaten fruit comparison RL agents, (bottom-center) eaten body comparison RL agents, (bottom-right) wall hit comparison RL agents

7 FINAL EVALUATION, CONCLUSION AND POSSIBLE IMPROVEMENTS

Once chosen the best RL agent(PPO), we proceed to the comparison of the trained agent with the baseline. The comparison is made considering 100 boards over 1000 iterations. The result can be seen in Fig 4, which shows that the RL agent achieves a higher reward and an average length that is more than twice the one of the baseline. The reasons why were explained in the previous section.

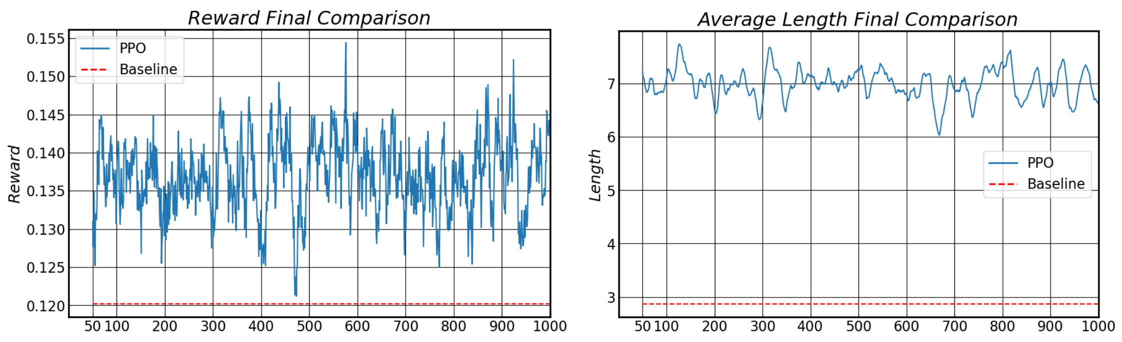

Figure 4: (left) reward comparison trained PPO and baseline, (right) average length comparison trained PPO and baseline

In summary, we developed multiple Deep RL agents that outperformed the baseline. The key finding was the importance of learning to avoid the snake's body. The comparison of agents showed the superiority of policy learning algorithms over value learning algorithms in this case. Suggestions for improvement are refining the reward function to consider body loss or the snake's current length upon eating a fruit. Also implementing masks to prevent negative outcomes from certain actions could enhance convergence.