Infinite Horizon Task Experiments

Tejas Pagare

June 25, 2022

1 Average Reward Objective

Objective: Maximize the per step reward i.e. finding the policy $\pi^\star = \{\mu_0, \mu_1, \ldots\}$ such that

$$\pi^\star = \underset{\pi}{\operatorname{argmax}}\ J^\pi_{\text{avg}} = \liminf_{N\to\infty} \frac{1}{N}\mathbb{E}\Big[\sum_{n=0}^{N-1} r(x_n, \mu_n(x_n))|x_0 = i\Big] \tag{1}$$

where $r(\cdot,\cdot)$ is the reward function from $\mathcal{S}\times\mathcal{A}\to\mathbb{R}$.

1.1 Full Gradient DQN

For stochastic environment we maintain a experience replay of transitions

$$\begin{aligned}\theta_{n+1} = \theta_n - a(n)\Big(\overline{r(X_n,U_n) + \max_v Q(X_{n+1},v;\theta_n) - f(Q;\theta_n) - Q(X_n,U_n;\theta_n)}\Big)\times\\ \Big(\nabla_\theta Q(X_{n+1},v_n;\theta_n) - \nabla_\theta f(Q;\theta_n) - \nabla_\theta Q(X_n,U_n;\theta_n)\Big)\end{aligned} \tag{2}$$

where $v_n = \underset{v}{\operatorname{argmax}}\ Q(X_{n+1},v;\theta_n)$ and the overline stands for averaging at time n over transitions $(X_k,U_k,X_{k+1}),\ k\le n$ for which $X_k = X_n, U_k = U_n$.
For deterministic environment, FGDQN does not require the use of experience replay and hence the equation becomes

$$\begin{aligned}\theta_{n+1} = \theta_n - a(n)\Big(r(X_n,U_n) + \max_v Q(X_{n+1},v;\theta_n) - f(Q;\theta_n) - Q(X_n,U_n;\theta_n)\Big)\times\\ \Big(\nabla_\theta Q(X_{n+1},v_n;\theta_n) - \nabla_\theta f(Q;\theta_n) - \nabla_\theta Q(X_n,U_n;\theta_n)\Big)\end{aligned} \tag{3}$$

1.2 DQN

Here, we maintain a target network with parameters θ^{targ} to calculate the "target" values and the current Q-values are calculated using a different network with parameters θ.

$$\begin{aligned}\theta_{n+1} = \theta_n + \frac{a(n)}{M}\times\\ \sum_{m=1}^{M}\Big((Z_{n(m)} - Q(X_{n(m)},U_{n(m)}))\nabla_\theta Q((X_{n(m)},U_{n(m)};\theta_{n(m)})\Big),\quad n\ge 0,\end{aligned} \tag{4}$$

where $(X_{n(m)},U_{n(m)}), 1\le m\le N$, are samples from past where the "target" values $Z_{n(m)}$ are calculated as follows:

$$Z_{n(m)} = r(X_{n(m)},U_{n(m)}) + \max_v Q(X_{n(m)+1},v;\theta^{\text{targ}}_{n(m)}) - f(Q;\theta^{\text{targ}}_{n(m)})$$

and the parameters θ^{targ} are updates slowly as follows:

$$\theta^{\text{targ}} \leftarrow \theta$$

2 Discounted Reward Objective

Objective: Maximize the discounted return with discount factor γ i.e. finding the policy $\pi^\star = \{\mu_0, \mu_1, \ldots\}$ such that

$$\pi^\star = \underset{\pi}{\text{argmax}}\ J^\pi_{\text{dis}} = \mathbb{E}\Big[\sum_{n=0}^{\infty} \gamma^n r(x_n, \mu_n(x_n))|x_0 = i\Big] \tag{5}$$

where $r(\cdot,\cdot)$ is the reward function from $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

2.1 Full Gradient DQN

For stochastic environment we maintain a experience replay of transitions

$$\begin{aligned}\theta_{n+1} = \theta_n - a(n)\Big(\overline{r(X_n, U_n) + \gamma \max_v Q(X_{n+1}, v; \theta_n) - Q(X_n, U_n; \theta_n)}\Big) \times \\ \Big(\gamma \nabla_\theta Q(X_{n+1}, v_n; \theta_n) - \nabla_\theta f(Q; \theta_n) - \nabla_\theta Q(X_n, U_n; \theta_n)\Big)\end{aligned} \tag{6}$$

where $v_n = \underset{v}{\text{argmax}}\ Q(X_{n+1}, v; \theta_n)$ and the overline stands for averaging at time n over transitions $(X_k, U_k, X_{k+1}),\ k \le n$ for which $X_k = X_n, U_k = U_n$.
For deterministic environment, FGDQN does not require the use of experience replay and hence the equation becomes

$$\begin{aligned}\theta_{n+1} = \theta_n - a(n)\Big(r(X_n, U_n) + \gamma \max_v Q(X_{n+1}, v; \theta_n) - Q(X_n, U_n; \theta_n)\Big) \times \\ \Big(\gamma \nabla_\theta Q(X_{n+1}, v_n; \theta_n) - \nabla_\theta f(Q; \theta_n) - \nabla_\theta Q(X_n, U_n; \theta_n)\Big)\end{aligned} \tag{7}$$

2.2 DQN

Here, we maintain a target network with parameters θ^{targ} to calculate the "target" values and the current Q-values are calculated using a different network with parameters θ.

$$\begin{aligned}\theta_{n+1} = \theta_n + \frac{a(n)}{M} \times \\ \sum_{m=1}^{M} \Big((Z_{n(m)} - Q(X_{n(m)}, U_{n(m)}))\nabla_\theta Q((X_{n(m)}, U_{n(m)}; \theta_{n(m)})\Big), \quad n \ge 0,\end{aligned} \tag{8}$$

where $(X_{n(m)}, U_{n(m)}), 1 \le m \le N$, are samples from past where the "target" values $Z_{n(m)}$ are calculated as follows:

$$Z_{n(m)} = r(X_{n(m)}, U_{n(m)}) + \gamma \max_v Q(X_{n(m)+1}, v; \theta^{\text{targ}}_{n(m)})$$

and the parameters θ^{targ} are updates slowly as follows:

$$\theta^{\text{targ}} \leftarrow \theta$$

3 Experiments

3.1 Catcher

Problem statement: In the Catcher game [1] the goal of the agent is to catch the following fruit with the paddle. The original game is a deterministic and episodic.

Episodic: The agent has 3 lives hence the game ends after 3 unsuccessful catch

Reward: The agent receives an award of +1 for a successful catch whereas, it loses a -1 point if the fruit is not caught.

Actions: The agent can take two actions, left and right.

Observation: Consists of x position and velocity of the agent as well as the x and y position of the falling fruit.

The velocity of the agent builds up in a particular direction if the agent continues to take action in that direction.
Due to the above characteristic, the task is deterministic since the observation is fully determined by the action and past observations.

We consider the infinite horizon, deterministic and stochastic setting of the problem.

Infinite Horizon: The problem is made infinite horizon by changing the lives of the agent to infinity.

Stochasticity: The environment is made stochastic by adding a friction to the paddle i.e. by taking a left action the agent ultimately takes left action with prob. p and remains idle with prob. $1-p$ and so on.

Action	Left $\leftarrow$	Idle	Right $\rightarrow$
$\leftarrow$	p	$1-p$	0
$\rightarrow$	0	$1-p$	p
Idle	$(1-p)/2$	p	$(1-p)/2$

$p = 1$ for deterministic setting.

NOTE: The fruit takes around 40 steps to reach the paddle from top to bottom and hence the optimal average reward is $\frac{1}{/40} = 0.025$.

3.2 Access Control Queuing Task

This problem was introduced in [2] as an example of a continuing (infinite-horizon) task. The problem description is as follows:
The agent has to decide whether or not to assign a server to the customers based on their priority and the status of the servers. The original problem contain 10 servers and the customers arrive with different priorities sampled randomly from $\{1, 2, 4, 8\}$. The queue never empties and each busy server becomes free with probability $p = 0.06$ on each time step.

Actions: 0 or 1 for deciding whether to accept or reject the customer for the access of the server.

Reward: It is equal to the priority of the customer if served and 0 if not. The rewards are scaled by the highest priority to maintain the range within [0,1] for experimentation.

Observation: The observation encodes the information of number of free servers and the priority of the arrived customer for that time step.

The total number of state-action pairs thus are 88 in total since there are four types of customers, 11 possible number of free servers (0 to 10), and two actions.

3.3 Forest Management

The objective of this problem is to maintain a forest for wildlife and make money by selling the wood. The setting we consider here is infinite-horizon and stochastic. The problem description is as follows:

State: The age of the forest $\in \{0, 1, 2, \ldots, M\}$; 0 being the youngest and M being the oldest.

Action: 0 and 1 for "wait" and "cut".

Reward: 0 for "wait" action and equal to the age of the forest for the "cut" action.

Dynamics: If the "cut" action is taken, the age of the forest resets to 0 whereas if "wait" action is taken, with probability p the fire breaks into the forest and the age resets to 0 and with probability $1 - p$ the age of the forest increases by one.

For experimentation, age is scaled by the maximum age i.e. M so that both state and reward remain within $[0, 1]$

4 Experiments

All the plots and detail of the hyperparameters used in the experiments can be found here RnD Experiments Wandb. Wandb [3] is an experiment tracking tool which allowed me to visualize the experiments on the go.

References

[1] N. Tasfi, "Pygame learning environment." https://github.com/ntasfi/PyGame-Learning-Environment, 2016.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[3] L. Biewald, "Experiment tracking with weights and biases," 2020. Software available from wandb.com.