Hindsight Goal Prioritization for Sparse Reward Environments

Final Project

Reinforcement Learning

Flavio Maiorana (2051396)

08/09/2023

SAPIENZA
UNIVERSITÀ DI ROMA

Table of Contents

Problem statement

Robotics environments

- Complex and different goals
- Sparse rewards
- Continuous action space

⇓

Problems with exploration and reward shaping

- Goal may be too complex and observation space is big: we may never get reward 1
- Classical off-policy algorithms don't valorize much the failed episodes

⇓

Solution: Enhancing the Replay Buffer

Fetch

1 Introduction

- Based on the 7-DoF Fetch Manipulator arm, with a two-fingered parallel gripper
- Tasks: Reach, Push, Slide and Pick-and-Place
- Action: Box(-1.0, 1.0, (4,), float32) $\rightsquigarrow$ Displacement in meters of the EE
- Observation: dictionary with info about the robot's end effector state and goal
  - Observation: ndarray of shape (25,) $\rightsquigarrow$ kinematic info of the block object and EE
  - Desired goal: ndarray of shape (3,) $\rightsquigarrow$ desired position of the EE or the block
  - Achieved goal: ndarray of shape (3,) $\rightsquigarrow$ current position of the EE or the block
- Reward: if we use sparse rewards -1 for every timestep and 0 for reaching the goal
- Termination: episodes have no termination since they have infinite horizon. Thus, they are truncated after T steps (by default 50)

Table of Contents

2 DDPG

Architecture - 1

2 DDPG

- Two neural networks in performing actor-critic policy gradient
  - **Actor** $\mu_\theta(s_i)$: observed state $\rightarrow$ action maximising the action-value function
  - **Critic** $Q_\phi(s_i, \mu_\theta(s_i))$: state and action $\rightarrow$ value of the action-value function
- Experience is collected according to the behavior policy $\pi_b = \mu_\theta(s_i) + \mathcal{N}$
- Transitions are stored in a **replay buffer**
- Every learning step, a batch $B = \{(s_i, a_i, d_i, r_i, s_i')\}$ is sampled

Architecture - 2

2 DDPG

- Critic optimization through Bellman equation by minimizing

$$L(\phi) \approx \frac{1}{|B|} \sum_{i=1}^{|B|} \underbrace{\left[r_i + \gamma Q_{\phi^-}(s'_i, \mu_{\theta^-}(s'_i)) - Q_\phi(s_i, a_i)\right]^2}_{\text{Temporal Difference Error}} \tag{1}$$

- Actor optimization through policy gradient theorem by maximising

$$J(\theta) \approx \frac{1}{|B|} \sum_{i=1}^{|B|} Q_\phi(s_i, \mu_\theta(s_i)) \tag{2}$$

- **Target networks** μ_{θ^-} and Q_{ϕ^-} for stability, updated with polyak averaging
  - $Q_{\phi^-} \leftarrow \rho Q_{\phi^-} + (1-\rho) Q_\phi$
  - $\mu_{\theta^-} \leftarrow \rho \mu_{\theta^-} + (1-\rho) \mu_\theta$

Table of Contents

Structure

3 Replay buffer

- Matrix with M columns and T rows, with T the max number of steps in one episode
- The element (i, j) contains the j_{th} transition of the i_{th} episode
- A transition is defined as $(s_i, a_i, achieved_i, desired_i, r_i, s'_i)$
  - $achieved_i$ is the current state as if the goal was reached
  - $desired_i$ is the actual final goal of the episode
  - The reward is computed with $desired_i$

Training steps in Fetch

- Play an episode and store it in a row of the buffer
- Sample |B| transitions, each from a sampled episode
- Sample a future achieved goal to substitute into the transition
- Do a gradient step

HER

3 Replay buffer

- The intention is to valorize also failed episodes (majority in robotics environments)
- Done by storing episodes multiple times, substituting the desired goal with a **Hindsight Goal**, treating the episode as if it was successful
- For each episode ($t = 0 \ldots 50$) the following steps are done (within DDPG)
  - Store every transition $(s_t||g, a_t, r_t, s_{t+1}||g)$ of the episode
  - Sample a set of additional achieved goals G from the current episode
  - Store $(s_t||g', a_t, r_t, s_{t+1}||g')$ for every $g' \in G$
  - Perform gradient descent, feeding also the goals into the neural networks
- Different strategies can be adopted to sample goals
  - Final: final state
  - Future: any future state of the episode
  - Random: any state of the buffer

HER

Pseudocode of how transitions are sampled with future strategy

1: $p_{future} \leftarrow 1 - \frac{1}{1+k}$ ▷ k the number of additionally stored episodes: $p_{future} \propto k$
2: $i_{ep} \leftarrow$ episodes indices uniformly sampled among the stored ones
3: $i_t \leftarrow$ transitions indices uniformly sampled from 0 to T
4: $i_{HER} \leftarrow$ indices of "resampled" future transitions, according to p_{future}
5: $i_{future} \leftarrow$ future transitions indices uniformly sampled from t to T
6: Substitute $goals[i_{HER}]$with a future goal indexed by i_{future}
7: Recompute the reward of changed transitions

HGR

Prioritizing future goals

- For the j_{th} transition, a transition to be the hindsight goal is sampled according to

$$P'(j,i) = \frac{|\delta_{ji}|^{\alpha'}}{\sum_{j=0}^{T-2}\sum_{i=j+1}^{T-1}|\delta_{ji}|^{\alpha'}}, \text{ with } j+1 \leq i \leq T-1 \tag{3}$$

- Priorities: matrix of shape $\left(M, \frac{(T-1)\cdot T}{2}\right)$, with M the buffer size in episodes
- New transitions stored with maximal priority, to let them be replayed at least once

HGR

Prioritizing episodes

- Episodes are sampled according to the probability

$$P(n) = \frac{|\delta^{(n)}|^{\alpha}}{\sum_n |\delta^{(n)}|^{\alpha}} \tag{4}$$

  where $|\delta^{(n)}|$ is the average TD Error of an episode
- Sumtree for efficiently sampling from this probability
  - Priorities in the leaves
  - Every node contains the sum of its children

HGR

Annealing the bias

- Importance sampling weights helps to scale down the sample's gradient when it is updated frequently and remains the same when it is rarely updated

$$w_{ji}^{(n)} = \left(\frac{1}{N_e}\frac{1}{P(n)}\right)^{\beta} \cdot \left(\frac{2}{(T-1)\cdot T}\cdot\frac{1}{P'(j,i)}\right)^{\beta'} \tag{5}$$

- In order to alway scale the gradients down we set $w_{ji}^{(n)} = \frac{w_{ji}^{(n)}}{max(w_{ji}^{(n)})}$

Table of Contents

Reach

4 Results

Push

4 Results

Pick-and-Place

4 Results

Hindsight Goal Prioritization for Sparse Reward Environments

Thank you for listening!
Any questions?