Reinforcement Learning & Advanced Deep

M2 DAC

TME 7. Continuous Actions

Ce TME a pour objectif d'expérimenter l'approche DDPG pour environnements à actions continues.

DDPG

Implémenter l'algorithme DDPG suivant:

Algorithm 1 Deep Deterministic Policy Gradient

Input: initial policy parameters θ, Q-function parameters ϕ, empty replay buffer $\mathcal{D}$
Set target parameters equal to main parameters $\theta_{\text{targ}} \leftarrow \theta$, $\phi_{\text{targ}} \leftarrow \phi$
repeat
  Observe state s and select action $a = \text{clip}(\mu_\theta(s) + \epsilon, a_{Low}, a_{High})$, where $\epsilon \sim \mathcal{N}$
  Execute a in the environment
  Observe next state s', reward r, and done signal d to indicate whether s' is terminal
  Store (s, a, r, s', d) in replay buffer $\mathcal{D}$
  If s' is terminal, reset environment state.
  if it's time to update **then**
    for however many updates **do**
      Randomly sample a batch of transitions, $B = \{(s, a, r, s', d)\}$ from $\mathcal{D}$
      Compute targets

$$y(r, s', d) = r + \gamma(1 - d) Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))$$

      Update Q-function by one step of gradient descent using

$$\nabla_\phi \frac{1}{|B|} \sum_{(s,a,r,s',d) \in B} \left(Q_\phi(s, a) - y(r, s', d)\right)^2$$

      Update policy by one step of gradient ascent using

$$\nabla_\theta \frac{1}{|B|} \sum_{s \in B} Q_\phi(s, \mu_\theta(s))$$

      Update target networks with

$$\phi_{\text{targ}} \leftarrow \rho \phi_{\text{targ}} + (1 - \rho)\phi$$
$$\theta_{\text{targ}} \leftarrow \rho \theta_{\text{targ}} + (1 - \rho)\theta$$

    end for
  end if
until convergence

Ajout d'un bruit gaussien pour exploration + clip pour rester dans des valeurs admissibles

Utilisation de réseaux cible (à la fois pour Q et pour μ) pour le calcul de la cible de Q

Gradient similaire à DPG, selon les transitions du batch

Mise à jour "soft" des paramètres des réseaux cible

Appliquer l'algorithme aux 3 problèmes suivants:

- MountainCarContinuous-v0
- LunarLanderContinuous-v2
- Pendulum-v0

Bonus: Q-Prop

Comparez les résultats avec l'algorithme Q-Prop (version conservative):

Algorithm 1 Adaptive Q-Prop

Initialize w for critic Q_w, θ for stochastic policy π_θ, and replay buffer $\mathscr{R} \leftarrow \emptyset$.
repeat
  for $e = 1, \ldots, E$ **do** ▷ Collect E episodes of on-policy experience using π_θ
    $\boldsymbol{s}_{0,e} \sim p(\boldsymbol{s}_0)$
    for $t = 0, \ldots, T-1$ **do**
      $\boldsymbol{a}_{t,e} \sim \pi_\theta(\cdot|\boldsymbol{s}_{t,e})$, $\boldsymbol{s}_{t+1,e} \sim p(\cdot|\boldsymbol{s}_{t,e}, \boldsymbol{a}_{t,e})$, $r_{t,e} = r(\boldsymbol{s}_{t,e}, \boldsymbol{a}_{t,e})$
  Add batch data $\mathscr{B} = \{\boldsymbol{s}_{0:T,1:E}, \boldsymbol{a}_{0:T-1,1:E}, r_{0:T-1,1:E}\}$ to replay buffer $\mathscr{R}$
  Take $E \cdot T$ gradient steps on Q_w using $\mathscr{R}$ and π_θ ← Off-Policy Critic
  Fit $V_\phi(\boldsymbol{s}_t)$ using $\mathscr{B}$
  Compute $\hat{A}_{t,e}$ using GAE(λ) and $\bar{A}_{t,e} = \nabla_{\boldsymbol{a}} Q_w(\boldsymbol{s}_t, \boldsymbol{a})|_{\boldsymbol{a}=\boldsymbol{\mu}_\theta(\boldsymbol{s}_t)}(\boldsymbol{a}_t - \boldsymbol{\mu}_\theta(\boldsymbol{s}_t))$.
  Set $\eta_{t,e}$
  Compute and center the learning signals $l_{t,e} = \hat{A}_{t,e} - \eta_{t,e}\bar{A}_{t,e}$ ← On-Policy Actor
  Compute $\nabla_\theta J(\theta) \approx \frac{1}{ET} \sum_e \sum_t \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t,e}|\boldsymbol{s}_{t,e}) l_{t,e} + \eta_{t,e} \nabla_{\boldsymbol{a}} Q_w(\boldsymbol{s}_{t,e}, \boldsymbol{a})|_{\boldsymbol{a}=\boldsymbol{\mu}_\theta(\boldsymbol{s}_{t,e})} \nabla_\theta \boldsymbol{\mu}_\theta(\boldsymbol{s}_{t,e})$
  Take a gradient step on π_θ using $\nabla_\theta J(\theta)$, optionally with a trust-region constraint using $\mathscr{B}$
until π_θ converges.