Deep Deterministic Policy Gradient (DDPG)

Insoon Yang

Department of Electrical and Computer Engineering
Seoul National University

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \mathrm{Prob}(a_t = a | s_t = s)$$

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \mathrm{Prob}(a_t = a | s_t = s)$$

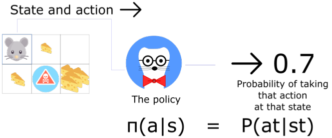
Probability that action a is selected given the current state is s

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \text{Prob}(a_t = a | s_t = s)$$

Probability that action a is selected given the current state is s



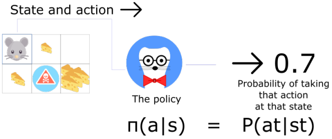
Stochastic policy

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \text{Prob}(a_t = a|s_t = s)$$

Probability that action a is selected given the current state is s



Stochastic policy

- Deterministic policy:

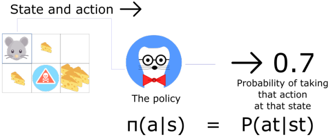
$$a = \mu(s)$$

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \mathrm{Prob}(a_t = a|s_t = s)$$

Probability that action a is selected given the current state is s



Stochastic policy

- Deterministic policy:

$$a = \mu(s)$$

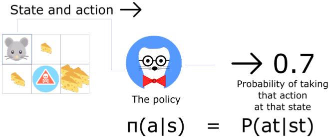
Action that is selected given the current state is s

Stochastic vs. Deterministic Policy

- Stochastic (randomized) policy:

$$\pi(a|s) = \mathrm{Prob}(a_t = a | s_t = s)$$

Probability that action a is selected given the current state is s

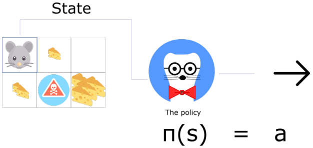


Stochastic policy

- Deterministic policy:

$$a = \mu(s)$$

Action that is selected given the current state is s



Deterministic policy

So far we considered stochastic policy gradient

Stochastic policy gradient:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_i \left(\sum_{t=0}^{T} \nabla_\theta \log \underbrace{\pi_\theta(a_t^i | s_t^i)}_{\text{stochastic policy}} \right) \left(\sum_{t=0}^{T} r(s_t^i, a_t^i) \right)$$

So far we considered stochastic policy gradient

Stochastic policy gradient:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_i \left(\sum_{t=0}^{T} \nabla_\theta \log \underbrace{\pi_\theta(a_t^i | s_t^i)}_{\text{stochastic policy}} \right) \left(\sum_{t=0}^{T} r(s_t^i, a_t^i) \right)$$

(Online) Actor-critic algorithm:

1. Take action $a \sim \pi_\theta(a|s)$, and observe (s, a, s', r);
2. Fit $v_\phi^\pi(s)$ using target $r + \gamma v_\phi^\pi(s')$;
3. Evaluate Advantage $A^\pi(s, a) = r + \gamma v_\phi^\pi(s') - v_\phi^\pi(s)$;
4. Estimate SG $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(a|s) A^\pi(s, a)$;
5. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

But deterministic policy looks simpler

But deterministic policy looks simpler

Q) Can we use deterministic policy?

But deterministic policy looks simpler

Q) Can we use deterministic policy?

- Yes, Deterministic policy gradient (DPG)

Deterministic Policy Gradient Algorithms

David Silver DAVID@DEEPMIND.COM
DeepMind Technologies, London, UK
Guy Lever GUY.LEVER@UCL.AC.UK
University College London, UK
Nicolas Heess, Thomas Degris, Daan Wierstra, Martin Riedmiller *@DEEPMIND.COM
DeepMind Technologies, London, UK

Towards deterministic policy gradient

- Parameterize deterministic policy μ by parameter vector θ: μ_θ

Towards deterministic policy gradient

- Parameterize deterministic policy μ by parameter vector θ: μ_θ
- Idea: Move the policy in the direction of $\nabla_\theta Q^{\mu^k}$ (Critic's Q-function)

Towards deterministic policy gradient

pi의 deterministic

- Parameterize deterministic policy μ by parameter vector θ: μ_θ
- Idea: Move the policy in the direction of $\nabla_\theta Q^{\mu^k}$ (Critic's Q-function)

$$\theta \leftarrow \theta + \alpha \mathbb{E}_{s \sim \rho^{\mu^k}} \left[\nabla_\theta Q^{\mu^k}(s, \mu_\theta(s)) \right],$$

where

$$\rho^\mu(s') := \int \sum_{t=1}^{\infty} \gamma^t p_0(s) p(s \to s', t, \mu) ds$$

denotes the discounted state visitation distribution

Applying chain rule

$$\theta \leftarrow \theta + \alpha \mathbb{E}_{s \sim \rho^{\mu^k}} \left[\nabla_\theta Q^{\mu^k}(s, \underbrace{\mu_\theta(s)}_{\text{deterministic policy}}) \right]$$

$$= \theta + \alpha \mathbb{E}_{s \sim \rho^{\mu^k}} \left[\underbrace{\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s, a)|_{a=\mu_\theta(s)}}_{\text{chain rule}} \right]$$

Applying chain rule

$$\theta \leftarrow \theta + \alpha \mathbb{E}_{s \sim \rho^{\mu^k}} \left[\nabla_\theta Q^{\mu^k}(s, \underbrace{\mu_\theta(s)}_{\text{deterministic policy}}) \right]$$

$$= \theta + \alpha \mathbb{E}_{s \sim \rho^{\mu^k}} \left[\underbrace{\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s, a)|_{a=\mu_\theta(s)}}_{\text{chain rule}} \right]$$

Q) Does it work? Is it valid?

Deterministic Policy Gradient Theorem

Yes, it's valid!

Deterministic Policy Gradient Theorem

Yes, it's valid!

- Performance objective:

$$J(\theta) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \underbrace{\mu_\theta(s_t)}_{\text{deterministic policy}})\right] = \mathbb{E}_{s\sim\rho^{\mu_\theta}}[r(s, \mu_\theta(s))]$$

Deterministic Policy Gradient Theorem

Yes, it's valid!

- Performance objective:

$$J(\theta) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \underbrace{\mu_\theta(s_t)}_{\text{deterministic policy}})\right] = \mathbb{E}_{s \sim \rho^{\mu_\theta}}\left[r(s, \mu_\theta(s))\right]$$

- Deterministic policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim \rho^\mu}\left[\nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a)\big|_{a=\mu_\theta(s)}\right]$$

Deterministic Actor-Critic algorithm

Initialize $\underbrace{\theta}_{\text{actor net}}, \underbrace{\phi}_{\text{critic net}};$

1. Take action $a = \mu_\theta(s)$, and observe $\{(s, a, s', r)\}$;
2. Fit $Q_\phi^\mu(s, a)$ using target $r + \gamma Q_\phi^\mu(s', \mu_\theta(s'))$;
3. Estimate $\nabla_\theta J(\theta) \approx \nabla_\theta \mu_\theta(s) \nabla_a Q_\phi^\mu(s, a)|_{a=\mu_\theta(s)}$;
4. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

Deterministic Actor-Critic algorithm

Initialize $\underbrace{\theta}_{\text{actor net}}, \underbrace{\phi}_{\text{critic net}};$

1. Take action $a = \mu_\theta(s)$, and observe $\{(s, a, s', r)\}$;
2. Fit $Q^\mu_\phi(s, a)$ using target $r + \gamma Q^\mu_\phi(s', \mu_\theta(s'))$;
3. Estimate $\nabla_\theta J(\theta) \approx \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu_\phi(s, a)|_{a=\mu_\theta(s)}$;
4. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

Q) Any issue?

Deterministic Actor-Critic algorithm

Initialize $\underbrace{\theta}_{\text{actor net}}, \underbrace{\phi}_{\text{critic net}}$;

1. Take action $a = \mu_\theta(s)$, and observe $\{(s, a, s', r)\}$;
2. Fit $Q^\mu_\phi(s, a)$ using target $r + \gamma Q^\mu_\phi(s', \mu_\theta(s'))$;
3. Estimate $\nabla_\theta J(\theta) \approx \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu_\phi(s, a)|_{a=\mu_\theta(s)}$;
4. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

Q) Any issue?

- On-policy: sample inefficient

And, Exploration 안되서 노이즈 따로 추가하기도 함

Off-Policy Deterministic Actor-Critic

Off-Policy Deterministic Actor-Critic

- Use any (behavior) policy: $\beta(s)$

Off-Policy Deterministic Actor-Critic

- Use any (behavior) policy: $\beta(s)$
- Modified objective (value function of μ_θ averaged over ρ^β):

$$J_\beta(\theta) = \int \rho^\beta(s) v^{\mu_\theta}(s) ds = \int \rho^\beta(s) Q^{\mu_\theta}(s, \mu_\theta(s)) ds$$

Off-Policy Deterministic Actor-Critic

- Use any (behavior) policy: $\beta(s)$
- Modified objective (value function of μ_θ averaged over ρ^β):

$$J_\beta(\theta) = \int \rho^\beta(s) v^{\mu_\theta}(s) ds = \int \rho^\beta(s) Q^{\mu_\theta}(s, \mu_\theta(s)) ds$$

- Modified policy gradient:

$$\begin{aligned}\nabla_\theta J_\beta(\theta) = \int \rho^\beta(s) \big[& \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} \\ & + \nabla_\theta Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} \big] ds\end{aligned}$$

Off-Policy Deterministic Actor-Critic

- Use any (behavior) policy: $\beta(s)$
- Modified objective (value function of μ_θ averaged over ρ^β):

$$J_\beta(\theta) = \int \rho^\beta(s) v^{\mu_\theta}(s) ds = \int \rho^\beta(s) Q^{\mu_\theta}(s, \mu_\theta(s)) ds$$

- Modified policy gradient:

$$\begin{aligned}\nabla_\theta J_\beta(\theta) = \int \rho^\beta(s) \big[& \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} \\ & + \nabla_\theta Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} \big] ds\end{aligned}$$

- Approximation (drop the second term):

$$\begin{aligned}\nabla_\theta J_\beta(\theta) &\approx \int \rho^\beta(s) \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} ds \\ &= \mathbb{E}_{s\sim\rho^\beta} \left[\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu_\theta}(s,a)|_{a=\mu_\theta(s)} \right]\end{aligned}$$

Off-Policy Deterministic Actor-Critic algorithm

Initialize θ, ϕ;

1. Take action $a = \beta(s)$, and observe $\{(s, a, s', r)\}$;
2. Fit $Q^{\mu}_{\phi}(s, a)$ using target $r + \gamma Q^{\mu}_{\phi}(s', \mu_\theta(s'))$; (No problem?)
3. Estimate $\nabla_\theta J(\theta) \approx \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu}_{\phi}(s, a)|_{a=\mu_\theta(s)}$;
4. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

Off-Policy Deterministic Actor-Critic algorithm

Initialize θ, ϕ;

1. Take action $a = \beta(s)$, and observe $\{(s, a, s', r)\}$;
2. Fit $Q^{\mu}_{\phi}(s, a)$ using target $r + \gamma Q^{\mu}_{\phi}(s', \mu_\theta(s'))$; (No problem?)
3. Estimate $\nabla_\theta J(\theta) \approx \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu}_{\phi}(s, a)|_{a=\mu_\theta(s)}$;
4. Update $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$;

- Advantage: Sample efficiency
- Disadvantage: Bias

Can we combine it with DQN?

Can we combine it with DQN?

Why not? Deep deterministic policy gradient (DDPG)

Published as a conference paper at ICLR 2016

CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING

Timothy P. Lillicrap[*], Jonathan J. Hunt[*], Alexander Pritzel, Nicolas Heess,
Tom Erez, Yuval Tassa, David Silver & Daan Wierstra
Google Deepmind
London, UK
{countzero, jjhunt, apritzel, heess,
etom, tassa, davidsilver, wierstra} @ google.com

ABSTRACT

We adapt the ideas underlying the success of Deep Q-Learning to the continuous action domain. We present an actor-critic, model-free algorithm based on the deterministic policy gradient that can operate over continuous action spaces. Using the same learning algorithm, network architecture and hyper-parameters, our algorithm robustly solves more than 20 simulated physics tasks, including classic problems such as cartpole swing-up, dexterous manipulation, legged locomotion and car driving. Our algorithm is able to find policies whose performance is competitive with those found by a planning algorithm with full access to the dynamics of the domain and its derivatives. We further demonstrate that for many of the tasks the algorithm can learn policies "end-to-end": directly from raw pixel inputs.

Best of Both Worlds

- Actor: Deterministic policy gradient
  1. Simple
  2. Continuous control

Best of Both Worlds

- Actor: Deterministic policy gradient
  1. Simple
  2. Continuous control
- Critic: DQN
  1. Off-policy (sample efficient)
  2. Experience replay (minimize correlations between samples)
  3. Target network (consistency)

Why not stochastic policy gradient + DQN?

- DQN is computationally inefficient to use with stochastic policies

Why not stochastic policy gradient + DQN?

- DQN is computationally inefficient to use with stochastic policies
- Q) Why?

Why not stochastic policy gradient + DQN?

- DQN is computationally inefficient to use with stochastic policies
- Q) Why?

  When computing target, it requires integral over action:

$$y_j^{-} := r_j + \gamma \underline{\int \pi_\theta(a|s_j') Q_{\phi^-}^{\pi}(s_j', a) da}$$

Deterministic:
gamma * Q(s*,a*) 끝

DDPG algorithm

- Initialize critic network Q_ϕ and actor network μ_θ with ϕ and θ;
- Initialize target networks Q_{ϕ^-}, μ_{θ^-} with $\phi^- \leftarrow \phi$ and $\theta^- \leftarrow \theta$;
- for episode $= 1 : M$
    - Initialize a random process $\mathcal{N}$ for exploration;
    - Receive initial state s_0;
    - for $t = 1 : T$
        1. Execute action $a_t = \mu_\theta(s_t) + \mathcal{N}_t$ and store (s_t, a_t, s_{t+1}, r_t) in **Buffer**;
        2. Sample a minibatch $\{(s_i, a_i, s_{i+1}, r_i)\}$ from **Buffer**;
        3. Set target $y_i^- := r_i + \gamma Q_{\phi^-}(s_{i+1}, \mu_{\theta^-}(s_{i+1}))$;
        4. Update the critic network by minimizing $L(\phi) := \frac{1}{N} \sum_i (Q_\phi(s_i, a_i) - y_i)^2$;
        5. Update the actor network by using deterministic policy gradient: $\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_i \nabla_\theta \mu_\theta(s_i) \nabla_a Q_\phi(s_i, a)|_{a=\mu_\theta(s_i)}$
        6. Update the target networks: $\phi^- \leftarrow \tau\phi + (1-\tau)\phi^-$, and $\theta^- \leftarrow \tau\theta + (1-\tau)\theta^-$ with small τ;

Advantages and Disadvantages

Advantages:

- Can handle continuous spaces (policy gradient)
- Sample efficiency (off-policy)
- Minimize correlations between samples (experience replay)
- Consistency (slowly changing target networks)

Advantages and Disadvantages

Advantages:

- Can handle continuous spaces (policy gradient)
- Sample efficiency (off-policy)
- Minimize correlations between samples (experience replay)
- Consistency (slowly changing target networks)

Disadvantages:

- Exploration

DDPG results