“Real World” Reinforcement Learning

CS4246/CS5446

AI Planning and Decision Making

Policy Search

Policy gradient, actor critic, correlated sampling

Policy Search

- A policy $\pi: S \rightarrow A$ is a mapping from state to action
- Assume the policy is parametrized by some parameters θ
  - Dimensions of θ should be smaller than the number of states
- Often use: Q-function parameterized by θ to represent π

$$\pi(s) = \arg\max_{a} \hat{Q}_{\theta}(s, a)$$

What problem can this representation pose when trying to optimize the policy?

- Policy search adjusts θ to improve the policy
- Idea:
  - Keep *twiddling* the policy as long as its performance improves, then stop

Intuition

- Policy search tries to find a good policy, e.g., represented as Q-function
  - Results in process that learns Q-functions
  - Q-learning with function approximation: find a value of θ such that $\hat{Q}_\theta$ is close to Q^*, the optimal Q-function
  - Policy search: find a value of θ that results in good policy
- Difference between good Q-function and optimal Q-function
  - Approximate Q-function defined by $\hat{Q}_\theta = \frac{Q^*}{100}$ gives optimal performance, even though it is not at all close to Q^*

Stochastic Policy

- For policy representation of the form:

$$\pi(s) = \arg\max_{a} \hat{Q}_\theta(s, a)$$

- Problem:
  - When actions are discrete, policy is a discontinuous function of θ
  - This makes gradient-based search difficult
- Stochastic policy:
  - $\pi_\theta(s, a)$ specifies the probability of selecting an action a in state s
  - E.g., Softmax function, with $\beta > 0$ modulating softness of the softmax:

Distribution of action: probability of selecting a in s

$$\pi_\theta(s, a) = \frac{e^{\beta \hat{Q}_\theta(s,a)}}{\sum_{a'} e^{\beta \hat{Q}_\theta(s,a')}}$$

Differentiable function of θ

Normalizer

How to Improve Policy?

- Definition:
  - Let $\rho(\theta)$ be the policy value – expected reward-to-go when π_θ is executed.
- For deterministic policy and deterministic environment:
  - If $\rho(\theta)$ is differentiable:
    Take a step in the direction of the policy gradient vector $\nabla_\theta \rho(\theta)$ – Look for the local optimum

Use gradient ascent or stochastic gradient ascent

- For stochastic environment and/or policy $\pi_\theta(s, a)$:
  - Obtain an unbiased estimate of the gradient at θ, $\nabla_\theta \rho(\theta)$ directly from results of trials executed at θ

Policy Gradient

- Consider: single action from single state s_0

$$\nabla_\theta \rho(\theta) = \nabla_\theta \sum_a R(s_0, a, s_0)\pi_\theta(s_0, a) = \sum_a R(s_0, a, s_0)\, \nabla_\theta \pi_\theta(s, a)$$

- Approximate the summation using samples generated from $\pi_\theta(s_0, a)$:

$$\nabla_\theta \rho(\theta) = \sum_a \pi_\theta(s_0, a) \cdot \frac{R(s_0,a,s_0)\nabla_\theta \pi_\theta(s, a)}{\pi_\theta(s_0, a)} \approx \frac{1}{N} \sum_{j=1}^{N} \frac{R(s_0,a_j,s_0)\nabla_\theta \pi_\theta(s_0, a_j)}{\pi_\theta(s_0, a_j)}$$

- For sequential case, this generalizes to:

Sample using policy

$$\nabla_\theta \rho(\theta) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{u_j(s)\nabla_\theta \pi_\theta(s, a_j)}{\pi_\theta(s, a_j)}$$

for each state s visited, where a_j is executed in s on the jth trial and $u_j(s)$ is the total reward received from state s onward in the jth trial.

Derivation: REINFORCE*[1]

- Alternately, for sequential case, policy gradient by sample approximation generalizes to

$$\nabla_\theta \rho(\theta) = \nabla_\theta \sum_\tau p_\theta(\tau) u(\tau)$$

We want to find the θ that maximizes the value of $\sum_\tau p_\theta(\tau) u(\tau)$

Where, τ is the trajectory generated by the policy and $u(\tau)$ is the sum of rewards from trajectory τ

- Using the policy gradient theorem[1] this can be written as:

$$\nabla_\theta \rho(\theta) \propto \sum_s p_{\pi_\theta}(s) \sum_a \nabla_\theta \pi_\theta(s,a) \hat{Q}_{\pi_\theta}(s,a)$$

Q-function at that state

States generated by policy

Gradient of policy

[1]Sutton & Barto Section 13.2

Derivation: REINFORCE*2

Sample using policy

- We can approximate the gradient using sampling

$$\nabla_\theta \rho(\theta) \propto \sum_s p_{\pi_\theta}(s) \sum_a \frac{\pi_\theta(s,a)\nabla_\theta \pi_\theta(s,a)\hat{Q}_{\pi_\theta}(s,a)}{\pi_\theta(s,a)}$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{\nabla_\theta \pi_\theta(s_{ij},a_{ij})u_i(s_{ij})}{\pi_\theta(s_{ij},a_{ij})}$$

i: trial, j: seq within trial

  - Where, a_{ij} is executed in s_{ij} on the j^{th} step of the i^{th} trial and $u_i(s_{ij})$ is the total reward (return) received from the j^{th} step onward in the i^{th} trial

REINFORCE

- Using an online update, we get the REINFORCE algorithm:

$$\theta_{j+1} = \theta_j + \alpha u_j \frac{\nabla_\theta \pi_\theta(s, a_j)}{\pi_\theta(s, a_j)}$$

- Using the identity:

$$\nabla_\theta \ln \pi_\theta(s, a_j) = \frac{\nabla_\theta \pi_\theta(s, a_j)}{\pi_\theta(s, a_j)}$$

- Rewriting:

$$\theta_{j+1} = \theta_j + \alpha\, u_j\, \nabla_\theta \ln \pi_\theta(s, a_j)$$

Ref: SB Section 13.3
Original ref: Williams, R. 1992

Variance reduction using a Baseline

- We estimate

$$\nabla_\theta \rho(\theta) = \sum_s p_{\pi_\theta}(s) \sum_a \nabla_\theta \pi_\theta(s, a_j) \hat{Q}_{\pi_\theta}(s, a)$$

- This is the same as

$$\sum_s p_{\pi_\theta}(s) \sum_a \nabla_\theta \pi_\theta(s, a_j) \overbrace{\left(\hat{Q}_{\pi_\theta}(s, a) - B(s) \right)}^{\text{Lower variance}}$$

For any function $B(s)$ because

$$\sum_a \nabla_\theta \pi_\theta(s, a_j) B(s) = B(s) \nabla_\theta \overbrace{\sum_a \pi_\theta(s, a_j)}^{\text{Sum to 1}} = B(s) \nabla_\theta 1 = 0$$

Variance reduction using a Baseline

- Using a baseline function $B(s)$ can reduce variance
- Natural choice: estimated $\hat{U}_{\pi_\theta}(s)$
- The function $A_{\pi_\theta}(s,a) = \hat{Q}_{\pi_\theta}(s,a) - \hat{U}_{\pi_\theta}(s)$ is called the advantage function

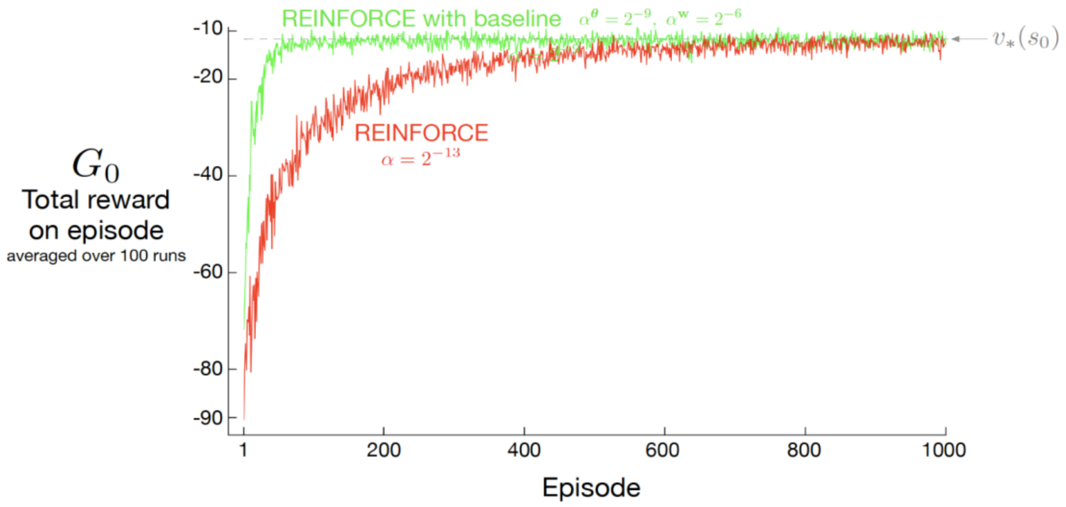

Figure 13.1 Sutton & Barto

Actor-Critic

- In policy search, a gradient step to update parameters w (to differentiate from the policy parameters) for the utility (value) function estimator is usually also done
  - To estimate both utility (value) and policy
- This is one form of actor-critic method
  - Learn a policy (actor) that takes action
  - Simultaneously, learn a utility (value) or Q-function that is used *only* for evaluation (critic)

Actor-Critic

- REINFORCE: Uses a Monte Carlo estimate of the advantage function, which has a higher variance
- For the TD method, the advantage function is:

$$\hat{Q}_{\pi_\theta}(s,a) - \hat{U}_{\pi_\theta}(s) = E\left[r + \gamma \hat{U}_{\pi_\theta}(s')\right] - \hat{U}_{\pi_\theta}(s)$$

- Using utility (value) function estimator $\hat{U}(s,w)$ with parameter w, TD-type update becomes:

$$\theta_{j+1} = \theta_j + \alpha \nabla_\theta \ln \pi_\theta(s_j, a_j) \underbrace{\boxed{\left(r_j + \gamma \hat{U}(s_{j+1}, w) - \hat{U}(s_j, w)\right)}}_{\text{advantage}}$$

- It is common to use multiple steps of rewards instead of one step:

$$r_j + \gamma r_{j+1} + \gamma^2 r_{j+2} + \cdots + \gamma^k \hat{U}(s_{j+k} + w)$$

Correlated Sampling

- Improve performance of policy search
  - Given environment simulator with repeatable random-number sequences
  - Generate a number of experiences in advance, and check the policies with the same set of experiences
  - Eliminate errors due to actual experiences encountered
- Main idea:
  - No. of random sequences required to ensure value of every policy is well estimated depends only on complexity of policy space, and not on complexity of underlying domain
- Example:
  - PEGASUS: for stable autonomous helicopter flighted (Ng and Jordan 2000)

Other Recent RL Approaches

- Policy Search
  - Trust Region Policy Optimization
  - Proximal Policy Optimization
  - GGPC
- Actor Critic
  - SAC
  - A2C
  - A3C
- Reward shaping
- Hierarchical reinforcement learning
- Apprenticeship learning
  - Imitation learning
  - Inverse reinforcement learning
- Etc.

Human Factors in Reinforcement Learning

- Complexity and uncertainty in real-world settings
  - COVID-19 pandemic response and recovery
  - MARS exploration
- Some promising trends
  - Hierarchical reinforcement learning
  - Apprenticeship reinforcement learning
    - Inverse Reinforcement learning
    - Imitation learning
  - Human experience and expertise as guides and constraints
    - Reward shaping
    - Priority sweeping
    - Heuristic functions
  - Mixed-initiative, responsible reinforcement learning (to be invented)

OpenAI Five Beat Top Human Players at Dota 2

- OpenAI vs human players
  - Policy gradient (Proximal Policy Optimization) with Recurrent neural networks (LSTM)
  - Beat human world champion Dota2 team (April 2019)

Video: https://www.youtube.com/watch?v=eHipy_j29Xw

Source: https://openai.com/five/

Homework

- Readings:
  - RN: 23.4.1, 23.4.2, 23.4.3, 23.5
- References:
  - SB: Chapter 13
    - [SB] Sutton, R. S. and A. G. Barto. Reinforcement Learning: An introduction. 2nd ed. MIT Press, 2018, 2020
      [Book website: http://incompleteideas.net/book/the-book.html]
      [e-Book for personal use: http://incompleteideas.net/book/RLbook2020.pdf]
- Online resources on reinforcement learning:
  - Silver, D. Lectures on Reinforcement Learning. 2015;
    Available from: https://www.davidsilver.uk/teaching/.

References

(Journal articles publicly available online or through NUS Library e-Resources)

- Deep reinforcement learning (DeepMind series):
  - DQN: Mnih, V., et al., Human-level control through deep reinforcement learning. Nature, 2015. 518(7540): p. 529-533.
  - AlphaGo: Silver, D., et al., Mastering the game of Go with deep neural networks and tree search. Nature, 2016. 529: p. 484+.
  - AlphaGo Zero: Silver, D., et al., Mastering the game of Go without human knowledge. Nature, 2017. 550: p. 354+.
  - MuZero: Schrittwieser, J., et al., *Mastering Atari, Go, chess and shogi by planning with a learned model.* Nature, 2020. **588**(7839): p. 604-609.
- Policy optimization (search)
  - Schulman, J., et al., Trust Region Policy Optimization, in Proceedings of the 32nd International Conference on Machine Learning, B. Francis and B. David, Editors. 2015, PMLR: Proceedings of Machine Learning Research. p. 1889--1897.
  - Schulman, J., et al., Proximal Policy Optimization Algorithms. CoRR, 2017. abs/1707.06347. Accessible from: http://arxiv.org/abs/1707.06347