Policy Gradient Methods for Reinforcement Learning with Function Approximation

Reminders

- REINFORCE algorithm performs Monte Carlo sampling of the stochastic policy, and updates the policy using policy gradient calculated from the partial returns (G_t) of episodes
  – Unbiased estimate of the gradient, but slow due to slow
  – Improves greatly aided with value function approximation

Overview

- Explicitly represent the policy with function approximator (parametrised)
- Update the parameters according to the gradient of expected reward
- Previous work:
  – Approximation of value-function + greedy *deterministic* policy
  – Optimal policy often stochastic and policy highly noise (from v-f) dependent
- Function approximators: e.g. NN with state as input and action probabilities as output
  – Update parameters using the policy gradient: $\triangle\theta = \alpha\frac{\partial J}{\partial\theta}$
- Here, small changes in θ cause only small changes in (stochastic) policy and thus state-visitation distribution, while small changes in v-f with deterministic policy have a larger influence
- Then, two different objectives used:
  – Global average reward:

$$J(\pi) = \lim_{n \to \infty} \mathrm{E}_\pi \left[\sum^{N} r_i \right] = \sum_s d^\pi(s) \sum_a \pi(s, a | \theta) \mathrm{E}[r | a, s]$$

  – Long-term reward from a start state (**prevalent**):

$$J(\pi) = \mathrm{E}_\pi \left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \middle| s_o \right]$$

Key ingredients

- Convergence of the policy iteration with arbitrary differentiable function approximation (local optimum)
- Policy gradient theorem:

$$\frac{\partial J}{\partial\theta} = \sum_s d^\pi(s) \sum_a \frac{\partial\pi(s, a | \theta)}{\partial\theta} Q^\pi(s, a)$$

- The effect of policy changes on the state distribution $\frac{\partial d^\pi(s)}{\partial\theta}$ does not appear
  – Convenient for sampling: simply draw samples following π to obtain unbiased estimate $\frac{\partial \log \pi(s, a | \theta)}{\partial\theta} Q^\pi(s, a)$ of the gradient

- Good, but still need to approximate Q!
  - Using actual returns leads to REINFORCE
  - Function approximation for Q^π speeds up learning and gives better performance
  - "Convenient" function approximation $Q^\omega = \frac{\partial \log \pi(s,a|\theta)^T}{\partial\theta}\omega$ (linear in ω) which we estimate from an unbiased Q^π estimate, e.g. the returns

Comments

- Well written and easy to follow
- Gives a lot of insights and useful comments on the matter
- Especially found useful comparisons with REINFORCE algorithm and insights about sampling
- Can see strong collaboration influences
- Sutton's book acts as a great supplement for understanding the paper