Policy gradient methods

Петрович Сергей, БПМИ181

Факультет компьютерных наук
Высшая школа экономики

16.02.2021

План доклада

1. Напоминание про обучение с подкреплением
2. Идея и мотивация градиента по стратегиям
3. Log-derivative trick
4. Policy gradient theorem
5. Policy gradient algorithms
    - REINFORCE
    - Baseline method
    - Actor-critic method
    - Advantage actor-critic method

Напоминание

- Множество состояний: S
- Множество действий: A
- Функция вознаграждения: $R : S \times A \to \mathbb{R}$
- Стратегия: $\pi(a|s) : A \times S \to \Pi(A)$
- Ценность состояния: $V^\pi(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma V^\pi(s'))$
- Ценность действия: $Q^\pi(s,a) = \sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma \sum_{a'} \pi(a'|s',\theta) Q(s',a'))$
- Ожидаемое вознаграждение при переходе из состояния s в s' после дейтсвия a: $R^a_{ss'}$
- Функция перехода между состояниями: $P^a_{ss'} : S \times A \to \Pi(S)$

Идея и мотивация метода

Мотивация:
хотим оптимизировать статегию π напрямую,
а не через функцию ценности действия Q.
Идея:
зададим стратегию параметрически как $\pi(a|s,\theta)$,
тогда подберем параметры θ, оптимизирую
некоторую целевую функцию, например:

$$J(\theta) = E\Big[\sum_{t=0}^{\infty} \gamma^t r_t\Big]$$

$$J(\theta) = E\big[V^{\pi}(s)\big] = \sum_{s \in S} p_0(s) V^{\pi}(s)$$

Сделать это можно градиентным подъемом - надо
только уметь считать градиенты.

Как считать градиент

Можно попробовать приблизить покомпонентными частными производными:

$$\frac{\partial J(\theta)}{\partial \theta_k} \approx \frac{J(\theta + \epsilon u_k) - J(\theta)}{\epsilon}$$

Основные недостатки:

- Сложно вычислительно
- Очень простой, высокая дисперсия

Но:

- Очень простой
- Работает даже для недифференцируемых функций

Оказывается, можно лучше!

Log-derivative trick

Вспомним свойства вероятностной плотности. Пусть задана многомерная вероятностная плотность $p(x|\theta)$ с вектором неизвестных параметров θ. Тогда можно заметить, что:

$$\nabla_\theta \log p(x|\theta) = \frac{\nabla_\theta p(x|\theta)}{p(x|\theta)} \Longrightarrow \nabla_\theta p(x|\theta) = p(x|\theta) \cdot \nabla_\theta \log p(x|\theta)$$

Зачем это нужно? Чтобы считать градиент математического ожидания некоторой $f(x)$:

$$\nabla_\theta E_{p(x|\theta)}[f(X)] = \nabla_\theta \int f(x) \cdot p(x|\theta) dx = \int f(x) \cdot \nabla_\theta p(x|\theta) dx =$$

$$= \int f(x) \cdot p(x|\theta) \cdot \nabla_\theta \log p(x|\theta) dx = E_{p(x|\theta)}[f(X) \cdot \nabla_\theta \log p(X|\theta)]$$

В частности, это означает, что мы можем считать градиент математического ожидания приближенно с помощью метода Монте-Карло.

Как считать градиент

Рассмотрим на примере одного шага марковского процесса:

$$J(\theta) = E\left[r\right] \Longrightarrow J(\theta) = \sum_{s \in S} p_0(s) \sum_{a \in A} \pi(a|s,\theta) R_s^a$$

$$\nabla_\theta J(\theta) = \sum_{s \in S} p_0(s) \sum_{a \in A} R_s^a \nabla_\theta \pi(a|s,\theta)$$

Заметим, что:

$$\nabla_\theta \pi(a|s,\theta) = \pi(a|s,\theta) \frac{\nabla_\theta \pi(a|s,\theta)}{\pi(a|s,\theta)} = \pi(a|s,\theta) \nabla_\theta \log \pi(a|s,\theta)$$

Таким образом, можно переписать выражение для градиента:

$$\nabla_\theta J(\theta) = \sum_{s \in S} p_0(s) \sum_{a \in A} R_s^a \pi(a|s,\theta) \nabla_\theta \log \pi(a|s,\theta) = E_{\pi(\theta)}[r \nabla_\theta \log \pi(a|s,\theta)]$$

Policy gradient theorem

Policy gradient theorem:

$$\nabla_\theta J(\theta) = E_{\pi(\theta)}[Q^{\pi(\theta)}(s,a) \cdot \nabla_\theta \log \pi(a|s,\theta)]$$

Целевая функция и ценность действия:

$$J(\theta) = \sum_s p_0(s) V^\pi(s) = \sum_s p_0(s) \sum_a \pi(a|s,\theta) Q^\pi(s,a)$$

$$Q^\pi(s,a) = \sum_{s'} P^a_{ss'} (R^a_{ss'} + \gamma \sum_{a'} \pi(a'|s',\theta) Q(s',a'))$$

REINFORCE algorithm

Самый просто метод, использующий идею policy gradient:

Алгоритм REINFORCE:

1. Инициализируем стратегию π_θ случайными значениями параметров θ
2. Проводим последовательность взаимодействий со средой с учетом текущей стратегии π_θ (состояние, действие, награда): $(s_1, a_1, r_2), (s_2, a_2, r_3), \ldots, (s_{T_i}, a_{T_i}, r_T)$
3. Для всех t от 1 до T обновляем параметры:

$$v_t = \sum_{i=t}^{T} r_i$$

$$\theta \longleftarrow \theta + \alpha \nabla_\theta \log \pi(a_t | s_t, \theta) \cdot v_t$$

4. Повторяем пункты $2., 3., 4.$

Baseline method

Усовершенстование **алгоритма REINFORCE**. Для этого посмотрим на выражение в **Policy gradient theorem** и заметим, что:

$$\nabla_\theta J(\theta) = E_{\pi(\theta)}[Q^{\pi(\theta)}(s,a) \cdot \nabla_\theta \log \pi(a|s,\theta)] = E_{\pi(\theta)}[(Q^{\pi(\theta)}(s,a) - b(s)) \cdot \nabla_\theta \log \pi(a|s,\theta)]\}$$

$$E_{p(x|\theta)}[\nabla_\theta \log p(X|\theta)] = \int \nabla_\theta \log p(x|\theta) \cdot p(x|\theta) dx = \int \frac{\nabla_\theta p(x|\theta)}{p(x|\theta)} \cdot p(x|\theta) dx = 0$$

Таким образом, при добавлении константного члена b относительно параметров θ несмещенность оценки градиента не пропадает, но при этом с помощью него можно регулировать дисперсию оценки.

Actor-critic method

Хотим и дальше уменьшать дисперсию оценки градиента. Для этого введем еще одну модель, которая будет подсказывать, в каком направлении изменять стратегию.

- **Critic model**: обновляет веса w в модели функции действия $Q_w(s, a)$
- **Actor model**: обновляет веса θ стратегии $\pi(a|s, \theta)$ в направлении, подсказанном критиком

Таким образом:

$$\nabla_\theta J(\theta) \approx E_{\pi(\theta)}\left[\nabla_\theta \log \pi(a|s, \theta) \cdot Q_w(s, a)\right]$$

Q Actor-critic algorithm

Алгоритм:

1. Инициализируем веса θ, w каким-то образом
2. Среда генерирует начально состояние s
3. Генерируем первое действие a согласно стратегии π_θ
4. На протяжении n шагов:
    - Окружение генерирует награду r и следующее состояние s'
    - Генерируем следующее действие a' согласно стратегии π_θ
    - $\delta = r + \gamma \cdot Q_w(s', a') - Q_w(s, a)$
    - $\theta \longleftarrow \theta + \alpha \nabla_\theta \log \pi(a|s, \theta) Q_w(s, a)$
    - $w \longleftarrow w + \beta \delta \nabla_w Q_w(s, a)$
    - $a \longleftarrow a'$, $s \longleftarrow s'$
5. Повторяем шаги $2., 3., 4., 5.$

Advantage actor-critic method

Теперь можем еще уменьшить дисперсию градиента - для этого разовьем идею **baseline**-метода и **actor-critic**-метода. Помимо обучаемой $Q_w(s, a)$ добавим еще опорное слагаемое, в роли которого будет оценка для $V^{\pi}(s)$!

$$Q^{\pi(\theta)}(s, a) \approx Q_w(s, a)$$

$$V^{\pi(\theta)}(s) \approx V_v(s)$$

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s) \quad \text{- advantage function}$$

$$A(s, a) = Q_w(s, a) - V_v(s)$$

Выражение для градиента можно переписать в виде:

$$\nabla_{\theta} J(\theta) = E_{\pi(\theta)}\left[A^{\pi}(s, a) \cdot \nabla_{\theta} \log \pi(a|s, \theta)\right]$$

Модели критика обучаем с помощью TD-обучения.

Итого

Преимущества:

- Лучшая сходимость среди остальных методов RL
- Эффективность в непрерывных пространствах действий
- Позволяет выучивать стохастические стратегии
- Позволяет учитывать предварительные знания о стратегии

Недостатки:

- Может сходиться к локальному оптимуму
- Может иметь большую дисперсию

Источники

- Картинка на слайде 4: https://www.bostondynamics.com/atlas
- Николенко С., Кадурин А., Архангельская Е., «Глубокое обучение. Погружение в мир нейронных сетей», Спб: Питер, 2020
- «RL Course by David Silver - Lecture 7: Policy Gradient Methods», https://www.youtube.com/watch?v=KHZVXao4qXs
- «Методы policy gradient и алгоритм асинхронного актора-критика», http://neerc.ifmo.ru/wiki/index.php?title=Методы _policy_gradient_и_алгоритм_асинхронного_актора-критика
- «Policy Gradient Algorithms», https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.htmlpolicy-gradient
- «Understanding Actor Critic Methods and A2C», https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f