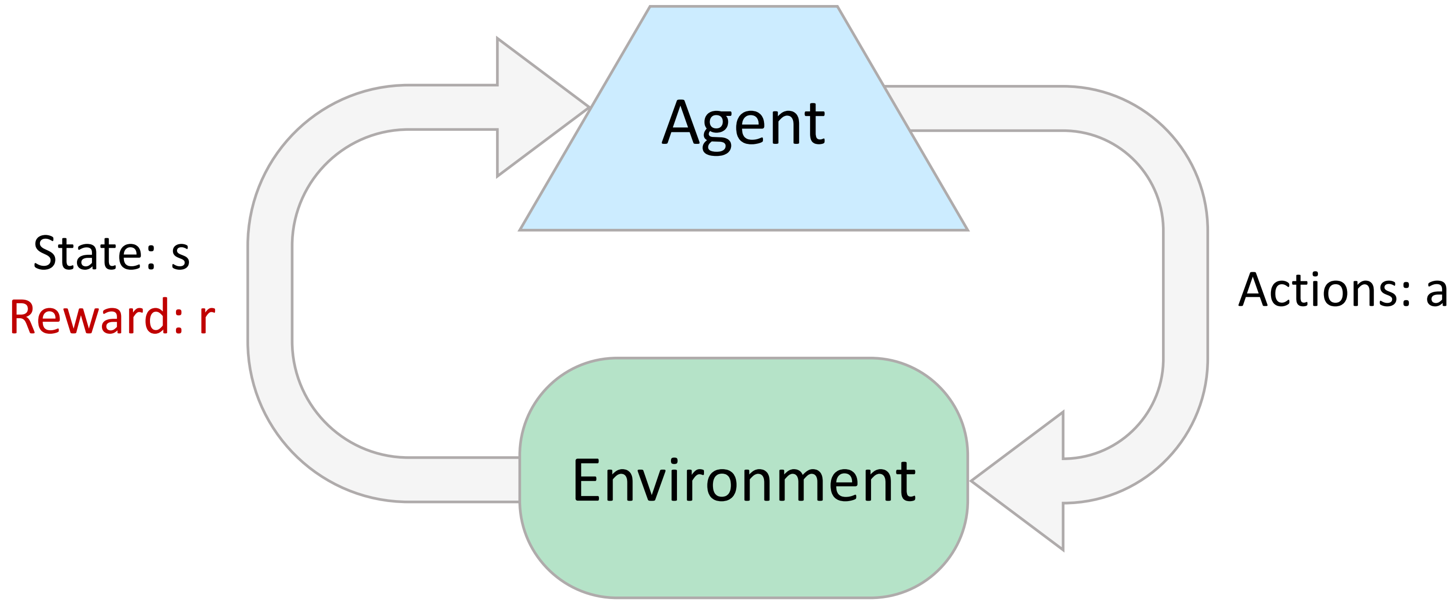
جمع‌بندی یادگیری تقویتی

درس هوش مصنوعی دکتر جوانمردی

زمستان ۱۴۰۰

RL: Agent + Environment



- MDP without T & R

Model-Based Learning

- Estimate T and Experience R

$$\hat{P}(a) = \frac{\mathrm{num}(a)}{N}$$

$$E[A] \approx \sum_a \hat{P}(a) \cdot a$$

Model-Free Learning

Passive

- fixed policy → learn the state values
- Direct Evaluation: Average together observed sample values (takes a long time to learn)
- Policy Evaluation? We don't have T & R!

$$V_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma V_k^{\pi}(s')]$$

Model-Free Learning

Passive

- Sample-Based Policy Evaluation (Indirect Evaluation)

$$sample_1 = R(s, \pi(s), s'_1) + \gamma V_k^{\pi}(s'_1)$$
$$sample_2 = R(s, \pi(s), s'_2) + \gamma V_k^{\pi}(s'_2)$$
$$\ldots$$
$$sample_n = R(s, \pi(s), s'_n) + \gamma V_k^{\pi}(s'_n)$$
$$V_{k+1}^{\pi}(s) \leftarrow \frac{1}{n} \sum_i sample_i$$

- Temporal Difference Learning: learn from every experience (exponential moving average)

$$sample = R(s, \pi(s), s') + \gamma V^{\pi}(s')$$
$$V^{\pi}(s) \leftarrow V^{\pi}(s) + \alpha(sample - V^{\pi}(s))$$

Model-Free Learning

- TD value learning can't calculate policy → TD Q-value learning

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

$$Q(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V(s') \right]$$

Active

- Active Reinforcement Learning: choose actions & learn optimal policy / values
- exploration vs. exploitation
- Q-learning: converges to optimal policy (off-policy learning)

$$sample = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

$$Q(s,a) \leftarrow (1-\alpha) Q(s,a) + (\alpha) \left[sample \right]$$

Exploration

ε-greedy

- random actions
- keeps thrashing around once learning is done
- lower ε over time

Exploation Functions

- explore areas whose badness is not yet established
- eventually stop learning

$$f(u, n) = u + k/n$$

- propagates the “bonus” back to states that lead to unknown states as well

Regret

- A measure of your total mistake cost
- The difference between your rewards and optimal rewards
- Minimizing regret requires **optimally learning** to be optimal
- Random exploration has higher regret than exploration functions

Generalizing Across States

- Basic Q-Learning keeps a table of all q-values
- Linear Value Functions (using Feature-Based Representations)

$$V(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$
$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

- experience is summed up in a few numbers
- similar states may have very different values
- Approximate Q-Learning

$$\text{difference} = \left[r + \gamma \max_{a'} Q(s', a')\right] - Q(s,a)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha\,[\text{difference}]$$ Exact Q's

$$w_i \leftarrow w_i + \alpha\,[\text{difference}]\, f_i(s,a)$$ Approximate Q's

Optimization

- Least Squares

$$\text{error}(w) = \frac{1}{2}\left(y - \sum_k w_k f_k(x)\right)^2$$

$$\frac{\partial\ \text{error}(w)}{\partial w_m} = -\left(y - \sum_k w_k f_k(x)\right) f_m(x)$$

$$w_m \leftarrow w_m + \alpha\left(y - \sum_k w_k f_k(x)\right) f_m(x)$$

- Overfitting

Policy Search

- Learn policies that maximize rewards, not the values that predict them
- Start with an initial linear value function or Q-function
- Nudge each feature weight up and down and see if your policy is better than before
- lookahead structure, sample wisely, change multiple parameters…