Reinforcement Learning with Human Feedback

The secret sauce in Chat-GPT.

Soumadeep Saha

Contents

PART 1

- What is reinforcement learning?
- RL Algorithms in brief - Dynamic Programming and Monte Carlo

PART 2

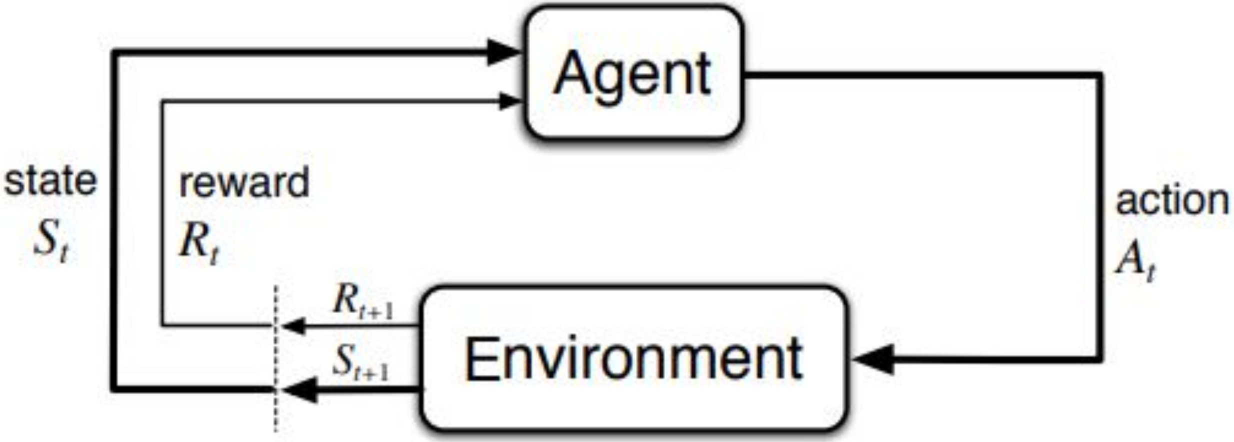
- Deep reinforcement learning algorithms
- Reward model
- RLHF

PART 1

A primer in RL

Markov Decision Process - Nomenclature

$$S \in \mathcal{S}, A(S) \in \mathcal{A}, r \in \mathcal{R} \subset \mathbb{R}.$$



$$p(s', r | s, a) = Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$$

Markov Decision Process - Transition function

Very useful, maybe unknown!

$$\text{We have, } r(s,a) = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a]$$
$$= \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r | s, a)$$

$$r(s,a,s') = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a, S_t = s']$$
$$= \sum_{r \in \mathcal{R}} r \frac{p(s', r | s, a)}{p(s' | s, a)}$$

The goal is to maximize reward - not immediate reward, **expected cumulative reward**

Markov Decision Process - Cumulative Reward

$$
\begin{aligned}
G_t &= R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots \\
&= R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \ldots) \\
&= R_t + G_{t+1}
\end{aligned}
$$

γ - Current value of future rewards.

Markov Decision Process - Policy, Value function

$$\pi : \mathcal{S} \to \mathcal{A},$$

$$\text{or, more generally } \pi : s \mapsto P(a|s)$$

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

$$= \mathbb{E}_\pi\Big[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \Big| S_t = s\Big]$$

Start from s, and follow policy π hence

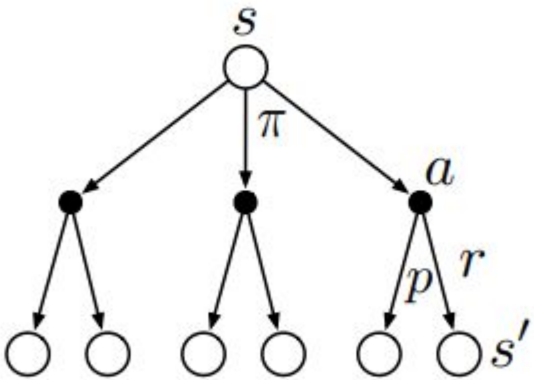
$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

$$= \mathbb{E}_\pi\Big[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \Big| S_t = s, A_t = a\Big]$$

Planning vs Control

- Planning refers to approximating/calculating v given π
- Control refers to finding optimal π^*
- v^* gives π^*. Can you see how?

Bellman Equation - consistency check



$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)\Big[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\Big] \\
&= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a)\Big[r + \gamma v_\pi(s')\Big], \quad \text{for all } s \in \mathcal{S},
\end{aligned}
$$

Bellman Equation - contd.

Good policy means lots of returns, better policy means more returns.

$$\pi \geq \pi' \iff v_{\pi}(s) \geq v_{\pi'}(s) \forall s \in \mathcal{S}$$

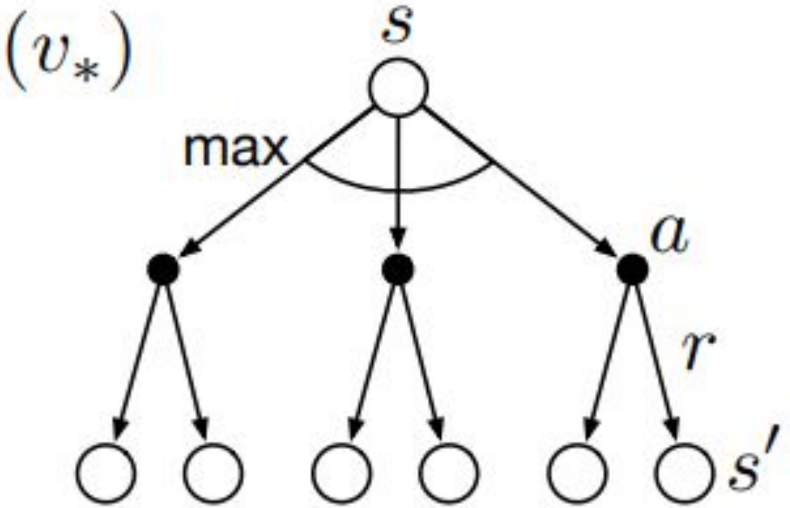
$$\pi^* \text{ is optimal } \iff \pi^* \geq \pi \, \forall \pi,$$

$$\text{and its state value function is } v_*$$

Bellman Optimality Equation

$$
\begin{aligned}
q_*(s,a) &= \mathbb{E}\Big[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \;\Big|\; S_t = s, A_t = a\Big] \\
&= \sum_{s',r} p(s', r | s, a)\Big[r + \gamma \max_{a'} q_*(s', a')\Big].
\end{aligned}
$$

$$
\begin{aligned}
v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r | s, a)\big[r + \gamma v_*(s')\big].
\end{aligned}
$$



Planning (Prediction) - Evaluating a policy

Example : policy iteration algorithm

$$v_0 \rightarrow v_1 \rightarrow \ldots v_k \ldots \rightarrow v_\pi$$

$$\text{random}$$

$$\begin{aligned} v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a)\Big[r + \gamma v_k(s')\Big] \end{aligned}$$

Planning (Prediction) - Evaluating a policy

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
    $\Delta \leftarrow 0$
    Loop for each $s \in \mathcal{S}$:
        $v \leftarrow V(s)$
        $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Example : policy iteration algorithm

Planning (Prediction) - Evaluating a policy

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
$\quad \Delta \leftarrow 0$
$\quad$ Loop for each $s \in \mathcal{S}$:
$\qquad v \leftarrow V(s)$
$\qquad V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
$\qquad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Can we improve this?
What are the potential problems?

Planning (Prediction) - Evaluating a policy

$$\begin{aligned} v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_k(s')\Big] \end{aligned}$$

$$\begin{aligned} v_{k+1}(s) &\doteq \max_a \mathbb{E}[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \max_a \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_k(s')\Big], \end{aligned}$$

Policy Improvement

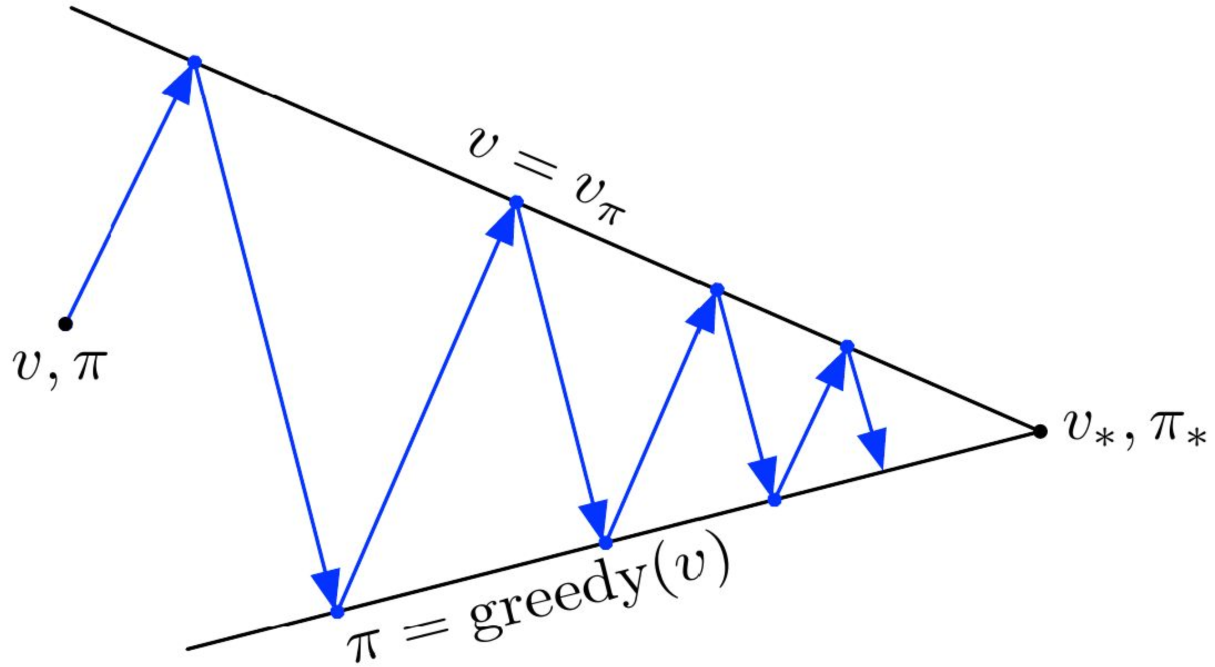
For a pair of deterministic policies π, π'

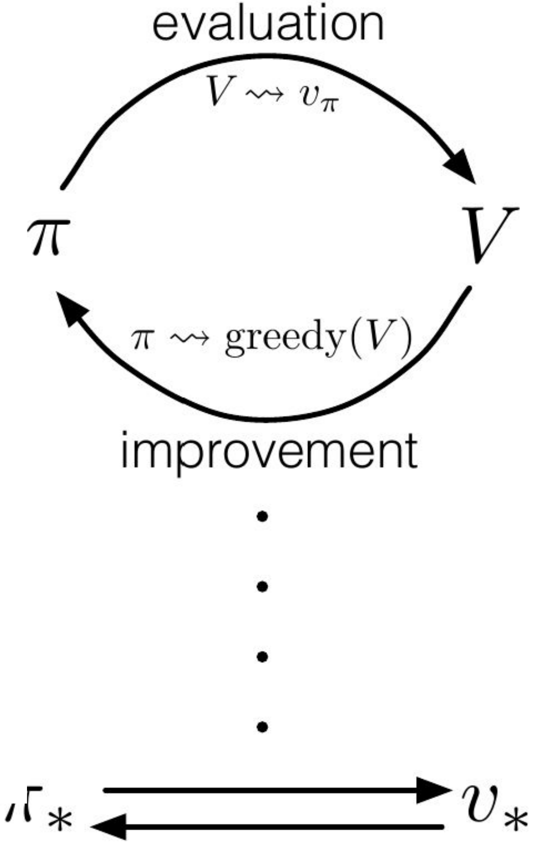
$$q_\pi(s, \pi'(s)) \geq v_\pi(s) \quad \forall s \in$$

$$\Rightarrow v_{\pi'}(s) \geq v_\pi(s)$$

$$\begin{aligned}
\pi'(s) &\doteq \arg\max_a q_\pi(s,a) \\
&= \arg\max_a \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \arg\max_a \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_\pi(s')\Big],
\end{aligned}$$

Putting it together...





Monte Carlo - Exploration vs Exploitation

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

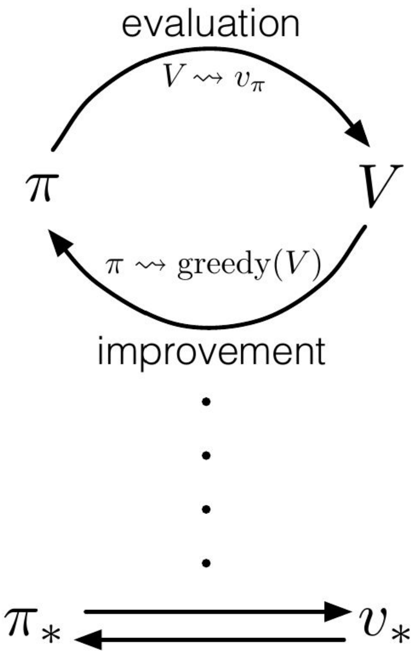
Key takeaways

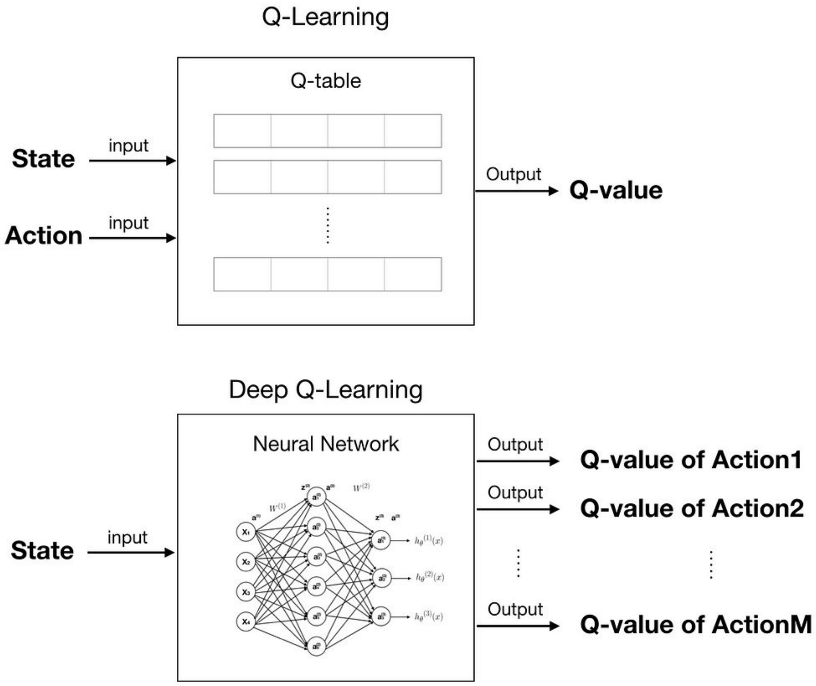
- MDP is a general framework underlying many real life problems of interest
- An agent seeks to optimize expected cumulative rewards in an environment
- v_π, q_π plays a pivotal role
- State/action spaces may be huge for tabular methods

PART 2

Human Feedback

Approximation methods - Deep learning



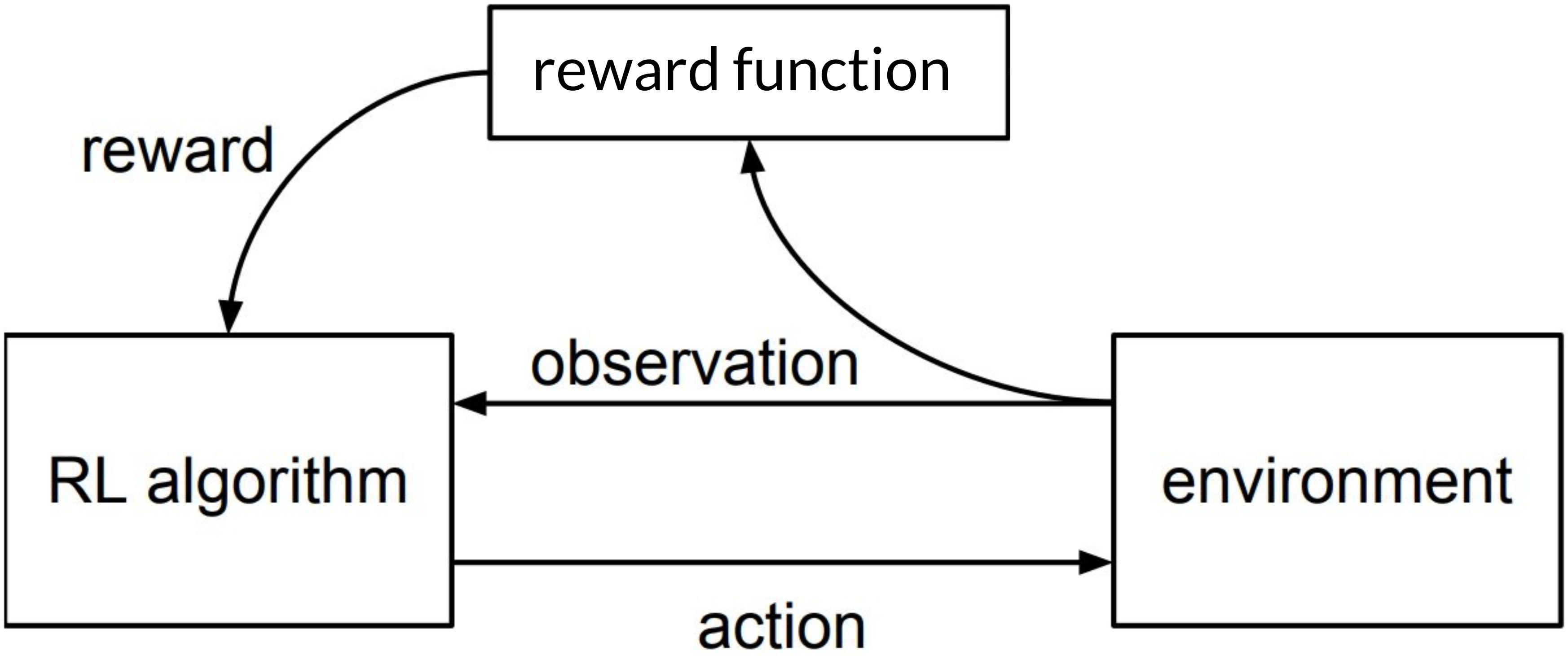


$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\Big]$$

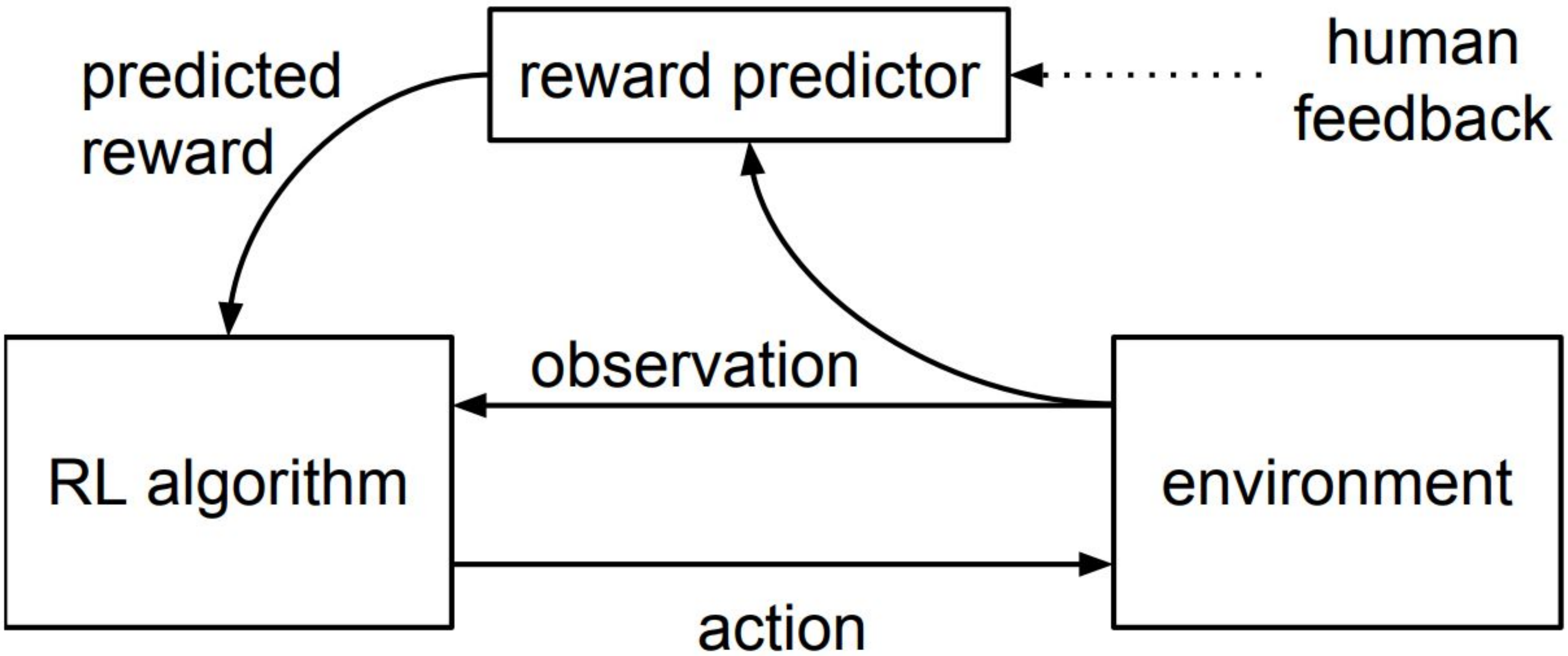
Deep reinforcement Learning

- There are several algorithms - DQN, DDQN, PPO, MCTS, etc.
- Approximate q(s,a) or v(s) with deep learning techniques.
- Initial policy -> Gather experience -> Memory -> Train model -> Improve policy -> Gather experience -> ...
- Generally harder than regular deep learning.

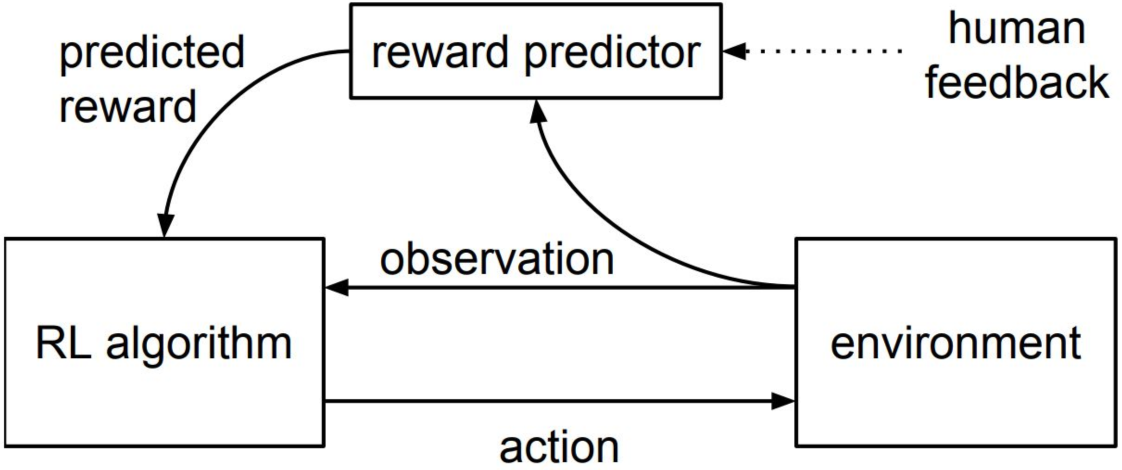
Reward MODEL



Reward MODEL



RLHF



- Predict how a human would reward a certain behaviour.
- Time efficient!
- Maximal disagreement examples to make efficient use of time - ensemble predictor.

Summary

- What is an MDP, agent, action, state, policy, etc
- How to evaluate a policy.
- How to improve a policy.
- Iterative improvement.
- Why we need approximations.
- How to introduce human feedback.