Reinforcement Learning

Morgan Sinclaire

You see an agent, run.

– The 2D Tensor

Outline

Overview: Layers of an Onion

Goal: Leveraging longer inference from base LLMs

Training Setup: Verifiable rewards on CoTs

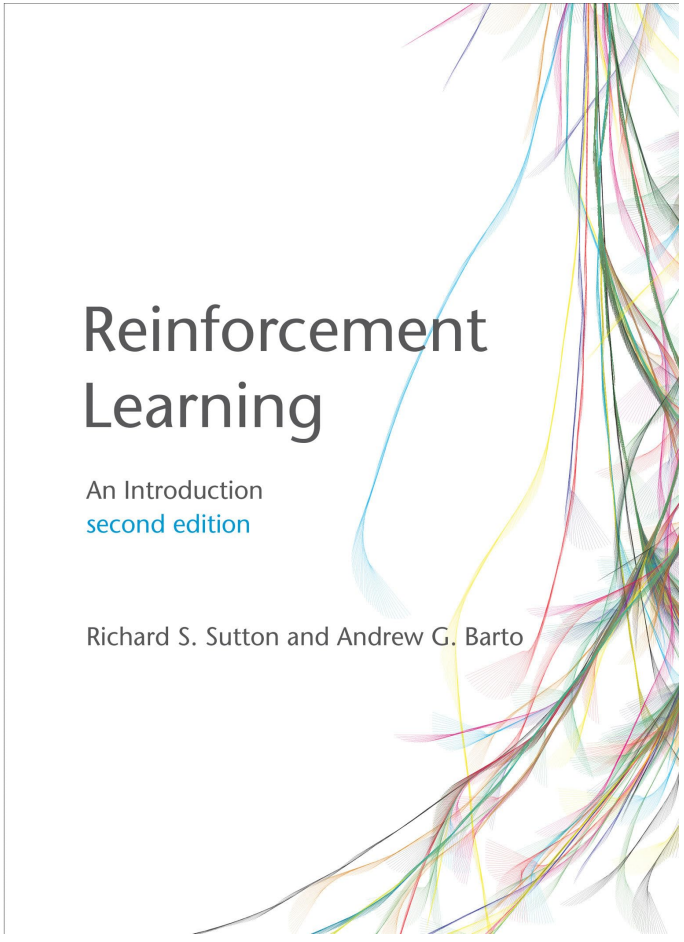
Methods Applied: RL

Inner Algorithm: PPO

Sutton x Barto

Andrew Barto and Richard Sutton were awarded the 2025 ACM Turing Award this month! They:

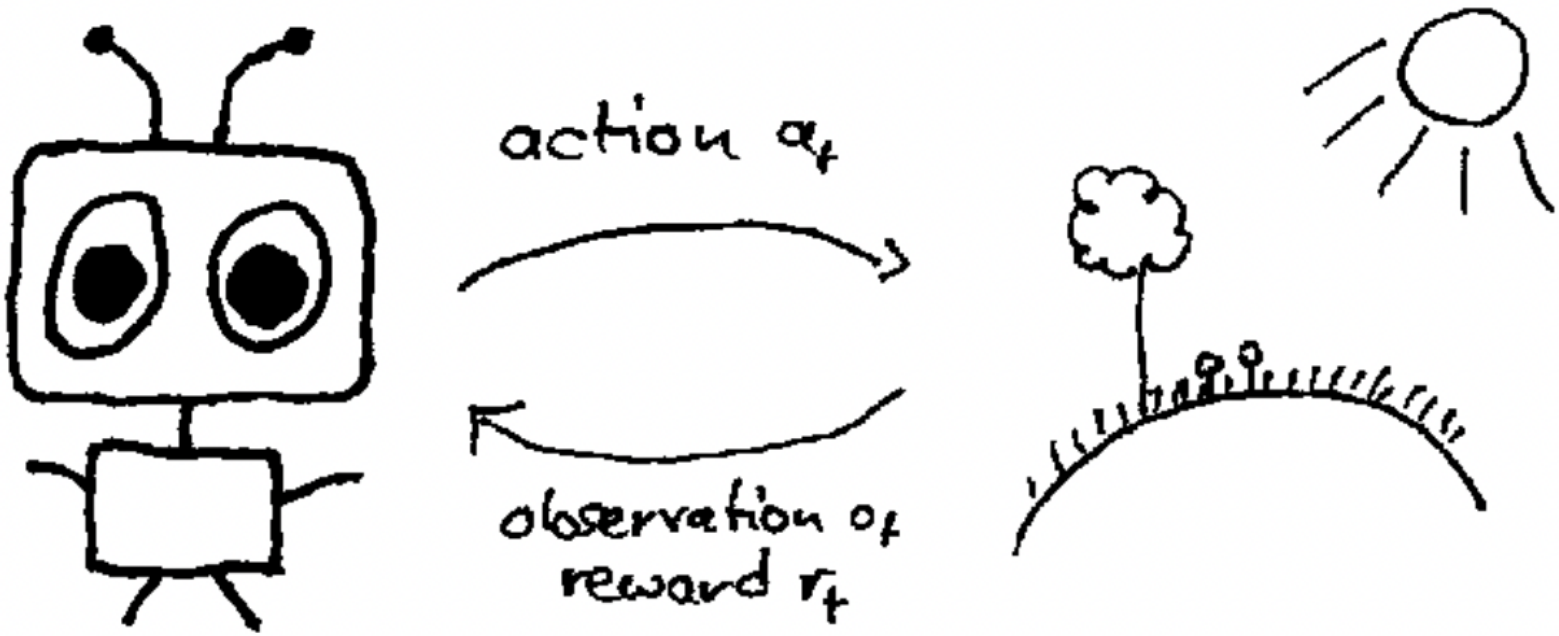
- Developed the mathematical theory of RL (e.g. Markov decision processes).
- Devised the basic algorithms.
- Wrote the still-popular book on it.
- Sutton also wrote a blog post that cool people recommended in January.



RL: Introductory References

- Sutton & Barto's book
- David Silver's lectures
- Sergey Levine's lectures from Berkely CS 285
- OpenAI's Intro to RL

RL = Real Life



RL = Real Life

In SL, we output a fixed vector (e.g. next-token probs) and get immediate feedback.

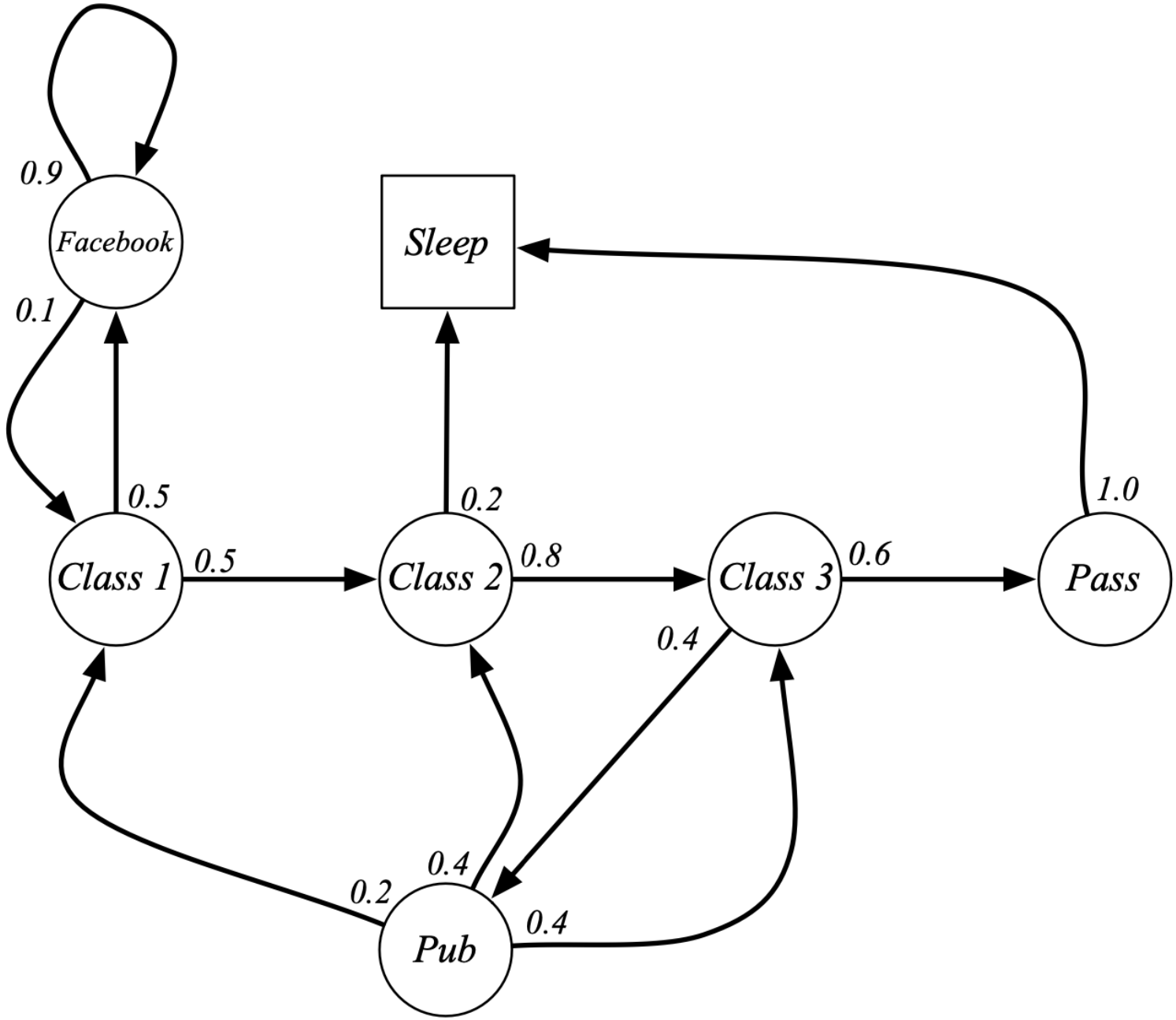
In RL, we take just take actions a_t in the environment indefinitely and get various reward signals r_t:

- The reward r_t at time t may have been the result of actions taken long before (*credit assignment problem*).
- Meaningful rewards can be sparse, may need to take actions to get more information (*explore/exploit tradeoff*[1]).

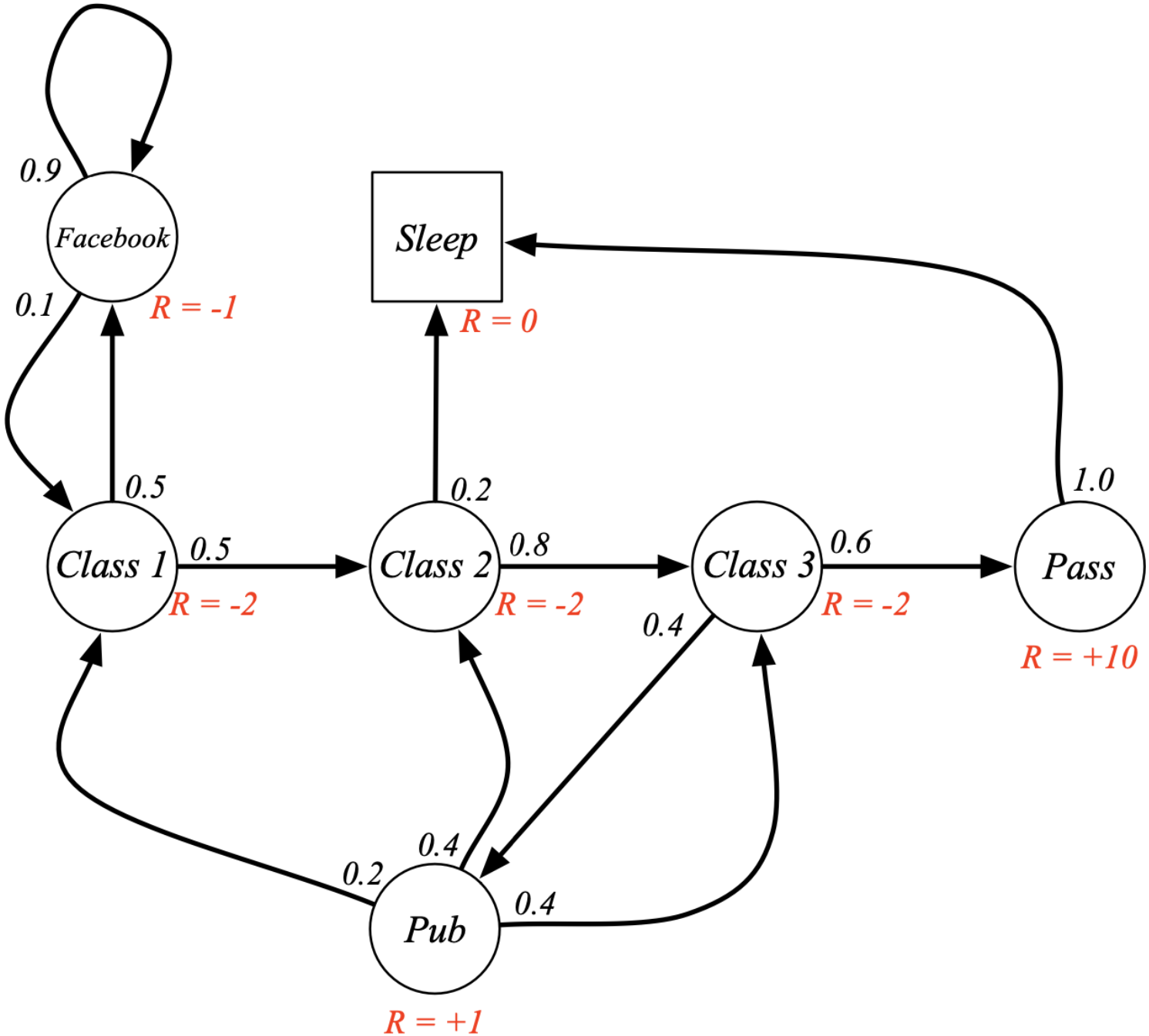
RL $\approx$ a strict superset of SL with these extra problems.

[1] "[T]he problem is a classic one; it was formulated during the war, and efforts to solve it so sapped the energies and minds of Allied analysts that the suggestion was made that the problem be dropped over Germany" - Whittle '79

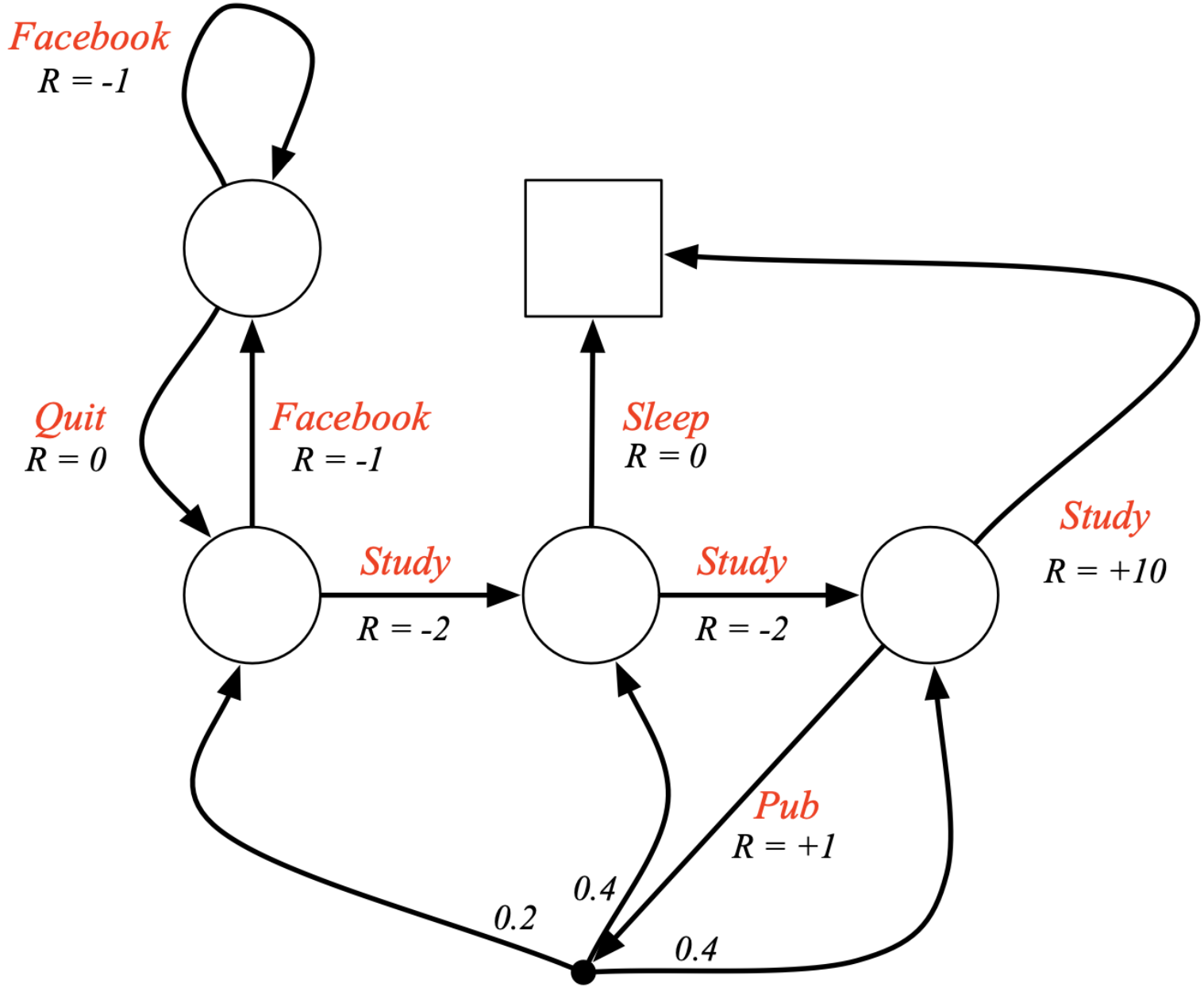
Markov Process



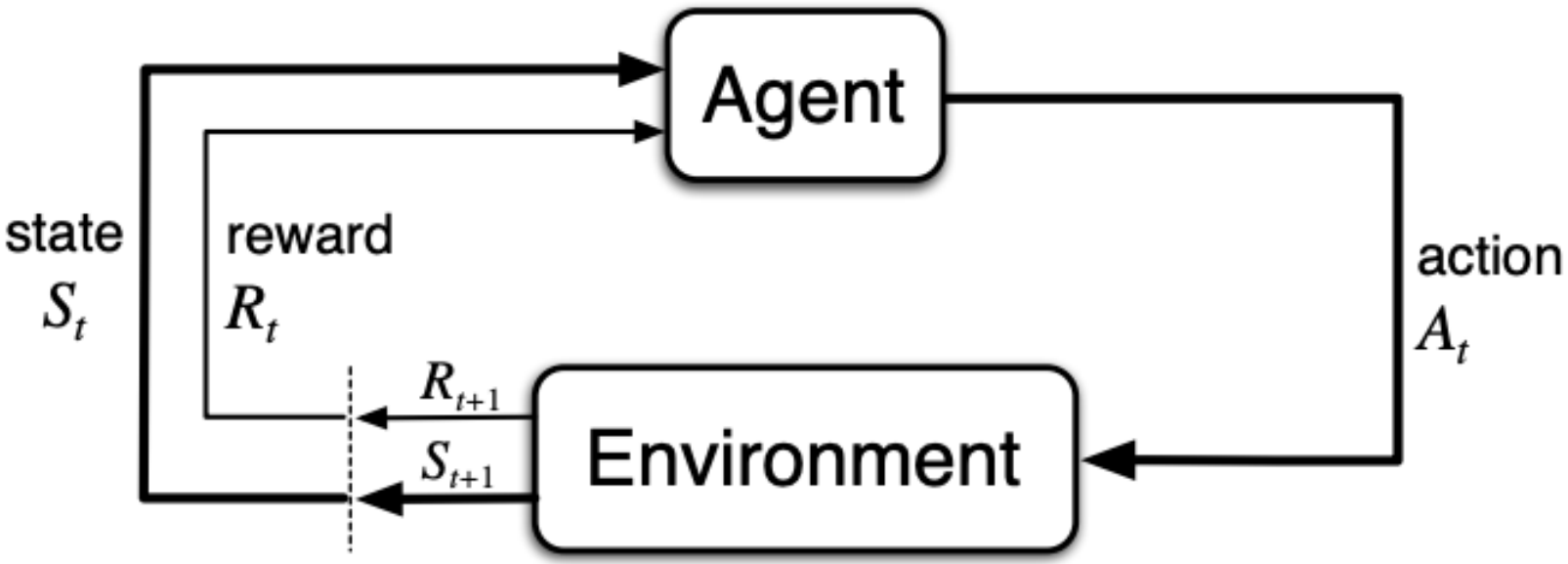
Markov Reward Process (MRP)



Markov Decision Process (MDP)



RL: Formalism



This gives a sequence (the *trajectory*)
$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \ldots$. We will assume:

- The environment is fully observable: $O_t = S_t$.
- Markov assumption:
  $\mathbb{P}(s_{t+1} = s | s_{\leq t}, a_{\leq t}) = \mathbb{P}(s_{t+1} = s | s_t, a_t)$.

RL: The Overarching Goal

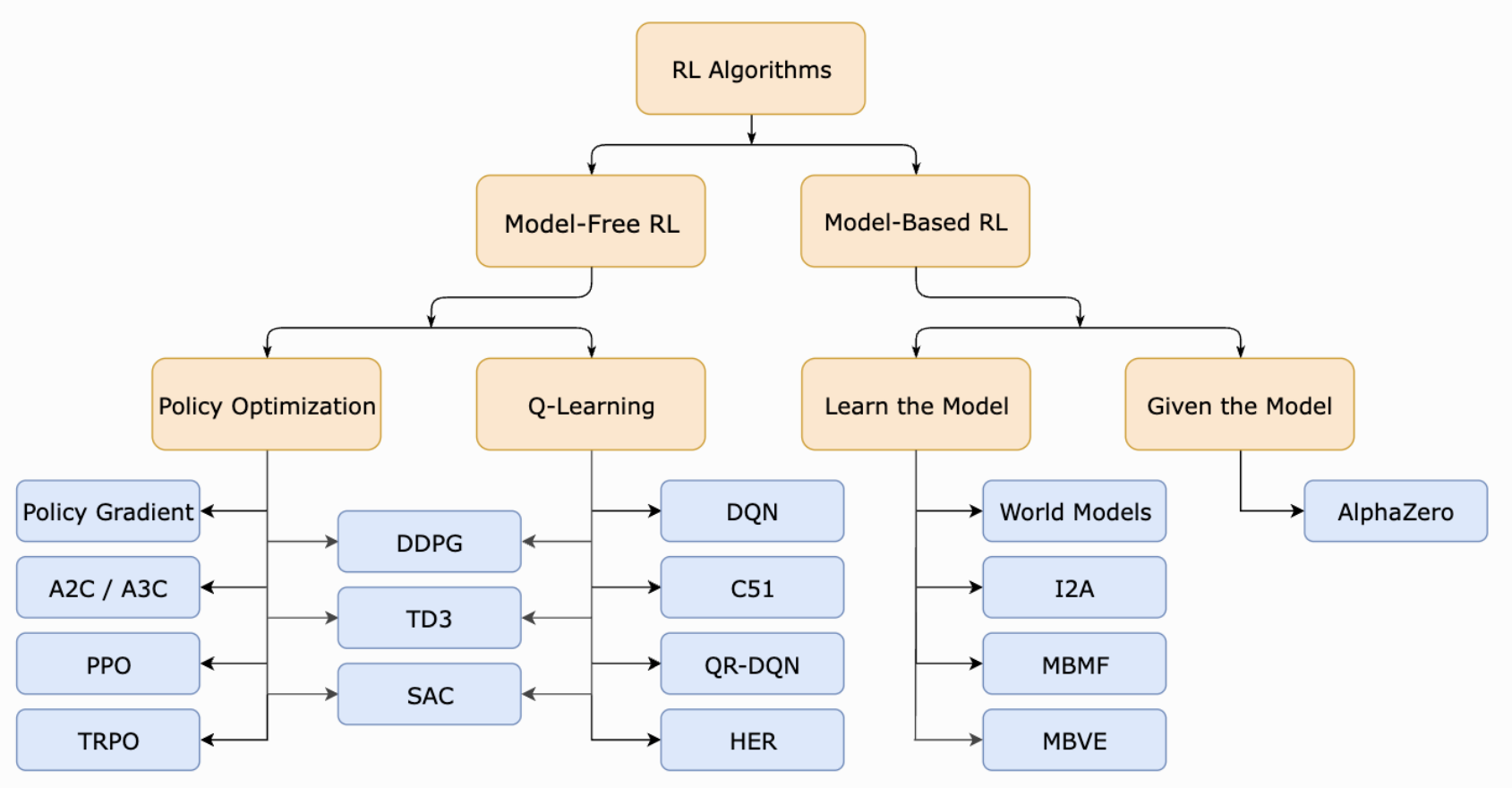
Fix some *discount rate* $\gamma \in [0, 1]$.

The goal of an RL agent is to maximize the time-discounted[2] future rewards:

$$R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots$$

[2]Why discount? Because the horizon may be infinite and we don't want a divergent series.

RL Algorithms: A Taxonomy



RL: Formalism

A **policy** π is the agent's behavior: a stochastic mapping of states to actions, $\pi(a|s) = \mathbb{P}(A_t = a|S_t = s)$.

Given a policy π and a state s, the **value function** V^π is the expected reward from following π from state s:

$$V^\pi(s) := \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... \mid S_t = s]$$

It is convenient to analogously define the **action-value function** $Q^\pi(s, a)$ as the expected reward from taking action a in state s, then following π:

$$Q^\pi(s, a) := \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... \mid S_t = s, A_t = a]$$

RL: The Overarching Goal

Ultimately, we want to learn the best policy, i.e. the policy which maximizes our future rewards:

$$\arg\max_{\pi} V^{\pi}(s) := \mathbb{E}_{\pi}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s]$$

We call this π^*, and the resulting value function V^*.[3]

[3] Define the partial order $\pi \preceq \pi'$ by $(\forall s)\ V^{\pi}(s) \leq V^{\pi'}(s)$. Under standard conditions it can be proved this has a maximal element.

Outline

Example: A Small Gridworld

actions

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

$R_t = -1$
on all transitions

- Terminal states in gray boxes.
- Finite time horizons: $\gamma = 1$ (no discounting).
- Actions leading out of the grid leave state unchanged.
- $R_t = -1$ while in non-terminal states.
- RL agent does *not* see the grid, just gets S_t, R_t.

Example: A Small Gridworld

actions

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

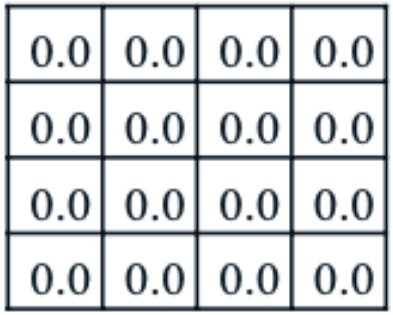
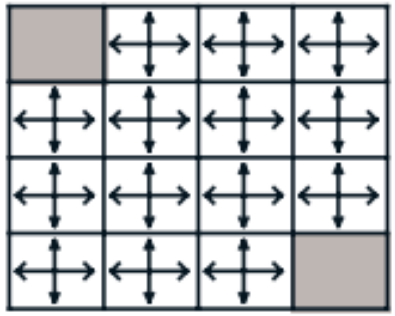
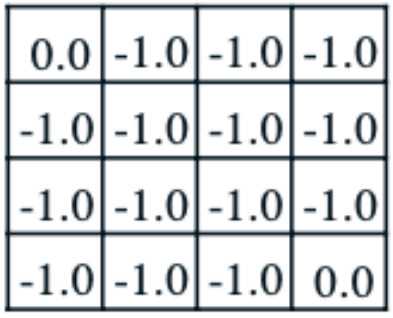
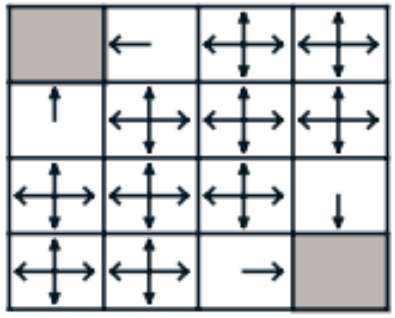
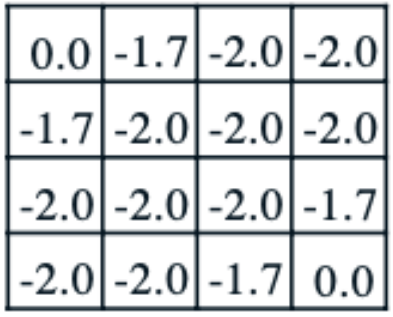
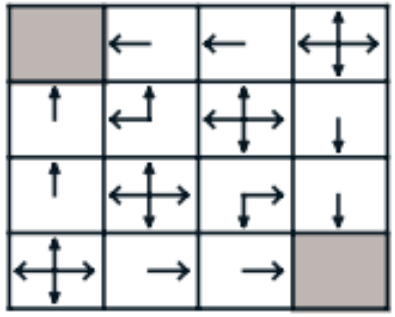
$$R_t = -1$$
on all transitions

- RL agent does *not* see the grid, just gets S_t, R_t.
- Let's suppose the agent adopts the uniformly random policy:

$$\pi(\uparrow \mid \cdot) = \pi(\downarrow \mid \cdot) = \pi(\leftarrow \mid \cdot) = \pi(\rightarrow \mid \cdot) = 0.25$$

- We'll iteratively evaluate this policy $v_1^\pi, v_2^\pi, v_3^\pi, \ldots$ such that $v_\infty^\pi = V^\pi$.

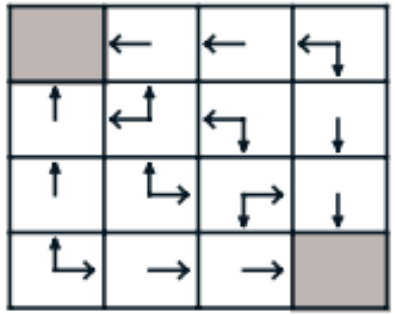
Iterative Policy Evaluation

	v_k for the random policy	greedy policy w.r.t. v_k
$k = 0$		
$k = 1$		
$k = 2$		

Iterative Policy Evaluation

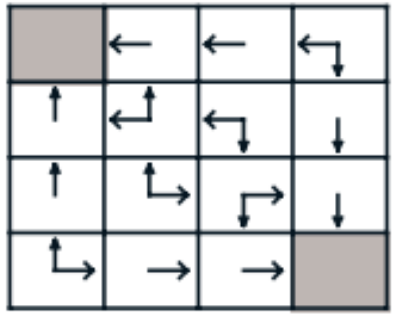
$k = 3$

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0



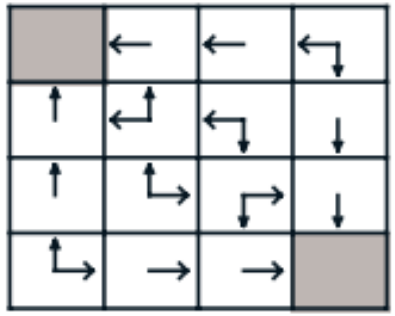
$k = 10$

0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0



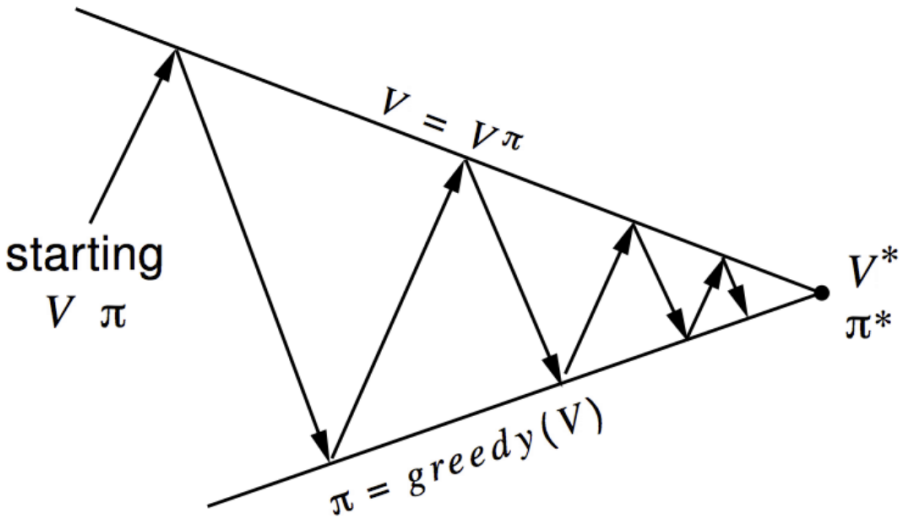
$k = \infty$

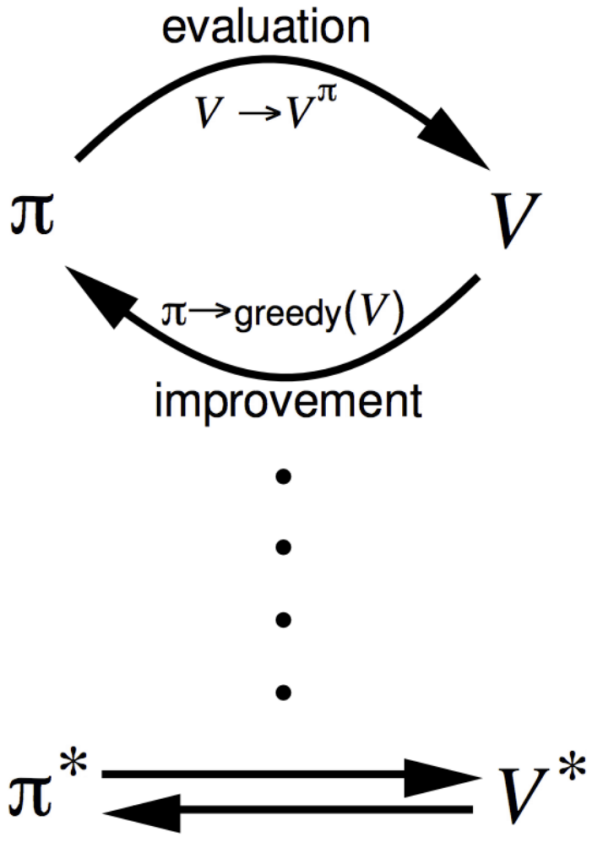
0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0



optimal policy

Policy Iteration





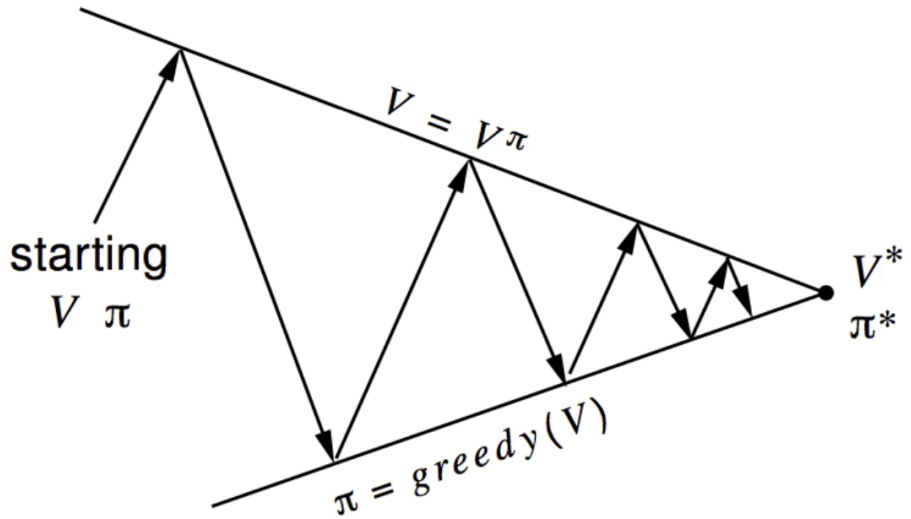
Policy evaluation Estimate v_π

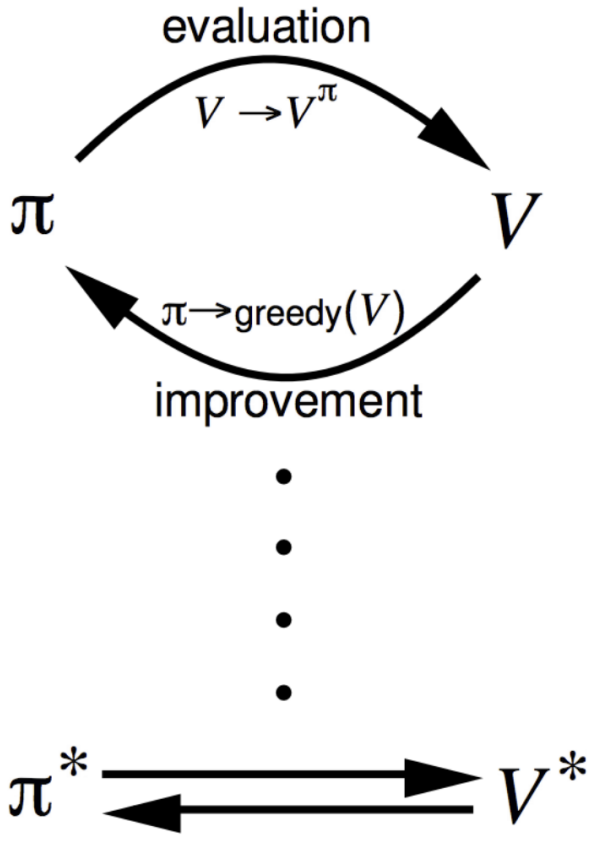
Iterative policy evaluation

Policy improvement Generate $\pi' \geq \pi$

Greedy policy improvement

Generalized Policy Iteration





Policy evaluation Estimate v_π

Any policy evaluation algorithm

Policy improvement Generate $\pi' \geq \pi$

Any policy improvement algorithm

Outline

Proximal Policy Optimization (PPO)

Out of time, maybe next semester!

Outline

Agents

An **agent** is a goal-directed entity which works to achieve/maximize some coherent objective(s). Agency exists on a spectrum, e.g. people can be more or less agentic:

- AJ is independent, takes initiative, knows what he wants and does what it takes to get it, even if that means going against the grain. AJ is an agent: he has goals (good or bad) and uses the full range of his capabilities to achieve them.
- Reese simply clocks in to his 9-5 job, follows established procedures, and does exactly what's required when asked. Reese is not an agent, he generally just reads in his "prompt", reacts to it, and responds to his environment rather than changing it.

A base LLM is *not* an agent (even though it can simulate agents).

Non-Agents

Non-agentic AI can in principle take many forms:

Tool AI: Optimized to carry out some specific task.

- A base LLM is not this: it only cares about next-token prediction, not your problems.

Oracle AI: Gives true answers to questions.

- An LLM simulates dishonest text just as much as true information.

Imitation AI: Optimized to imitate humans.

- An LLM is also trained to predict (e.g.) the results section of a scientific paper.

Non-Agents

Non-agentic AI can in principle take many forms:

Tool AI: Optimized to carry out some specific task.

- A base LLM is not this: it only cares about next-token prediction, not your problems.

Oracle AI: Gives true answers to questions.

- An LLM simulates dishonest text just as much as true information.

Imitation AI: Optimized to imitate humans.

- An LLM is also trained to predict (e.g.) the results section of a scientific paper.

A base LLM is a predictor/simulator: it can be inserted into the middle of any text (book/paper/conversation/argument), and it will simulate the continuation of that text.

- Much like a weather model inputs any initial conditions, and "evolves the system" according to the rules it has learned.

Why Agents?

There's a consilience of empirical/theoretical reasons why we expect AGI/ASI to be agentic:

0. Base rate: evolution created agents without "intending to".
1. Autonomous systems are more economically valuable than mere tools.
    - This is why most YC startups are working on exactly this.
2. The new RL/reasoning paradigm is fundamentally *about* making LLMs more like autonomous agents.
3. When we can empirically measure agency in LLMs, we find the newer ones are scoring higher (e.g. long-horizon planning, concordance with VNM utility).
4. Coherence theorems (e.g. VNM utility theory) imply "smarter" systems tend to converge to goal-directed behavior.
    - Empirically, agency and intelligence seem to be correlated in humans.