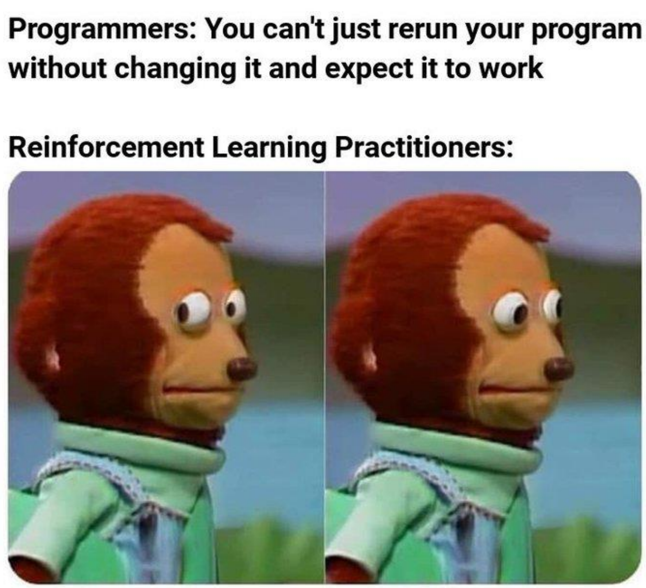
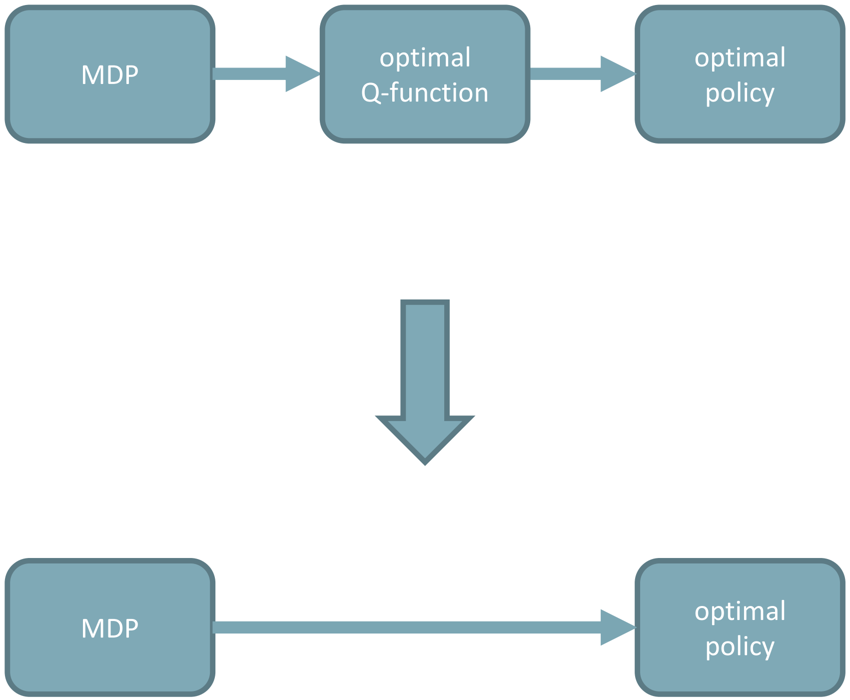
Methods for policy-based RL



https://twitter.com/halawa_marah/status/1382646849471926272

Methods for policy-based RL

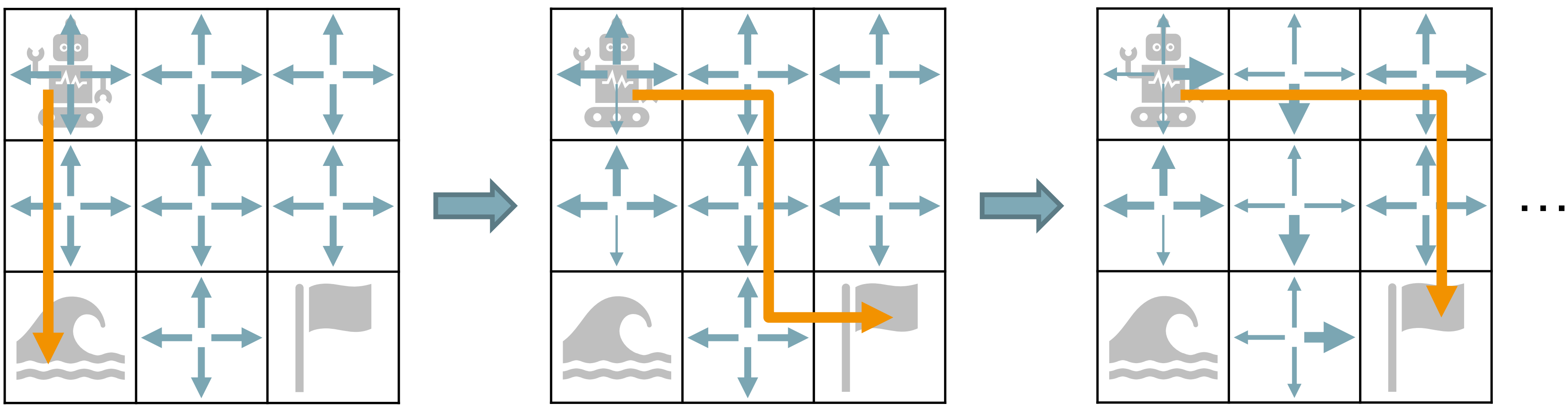
Idea: Obtain an optimal policy without calculating the (V-)/Q-function first



Methods for policy-based RL

General approach (simplified, discrete actions)

- Run episode and evaluate return
- If the return is high, increase percentage of taking the corresponding action,
  if the return is low, decrease percentage of taking the corresponding action
- Repeat



Methods for policy-based RL

Example: Multi-armed bandits

- Three actions for a given state: Pull lever of slot machine 1/2/3 and get reward
- Special case: Probabilistic (non-deterministic) rewards R
- At time t, agent selects action a_t and gets reward R_t
- Goal: Maximize (average) reward R_t

https://rl-lab.com/multi-armed-bandits/

Methods for policy-based RL

General approach (mathematical)

Policy gradient: Gradient ascent over policy $\pi_{\boldsymbol{\theta}} = \pi(a|s, \boldsymbol{\theta})$ with parameterizable weights $\boldsymbol{\theta}$

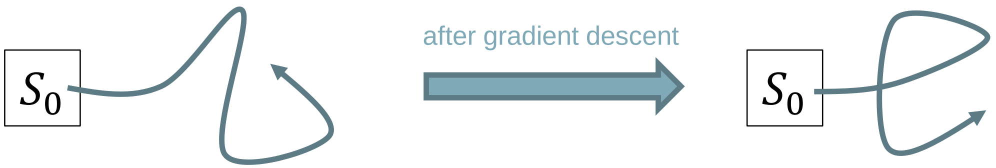
1. Set up cost function based on policy, e.g. V-function of start state S_0 for episodic tasks
$$J = V_{\pi_{\boldsymbol{\theta}}}(S_0)$$
2. Calculate gradient w.r.t. weights $\boldsymbol{\theta}$
$$\nabla_{\boldsymbol{\theta}} J = \nabla_{\boldsymbol{\theta}} V_{\pi_{\boldsymbol{\theta}}}(S_0)$$
3. Optimize weights $\boldsymbol{\theta}$ through gradient ascent to improve the policy and go to 1.
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \nabla_{\boldsymbol{\theta}} J$$

Methods for policy-based RL

Problem with $\nabla_{\boldsymbol{\theta}} J = \nabla_{\boldsymbol{\theta}} V_{\pi_{\boldsymbol{\theta}}}(S_0)$:

- The policy $\pi_{\boldsymbol{\theta}} = \pi(a|s, \boldsymbol{\theta})$ is a function of the state(s) s
- Hence when calculating the gradient $\nabla_{\boldsymbol{\theta}} V_{\pi_{\boldsymbol{\theta}}}(S_0)$ and using the chain rule, one has to calculate $\frac{\partial s}{\partial \boldsymbol{\theta}}$ at some point
- But this is impossible to calculate. It means that we have to calculate how the distribution of states s changes depending on $\boldsymbol{\theta}$
- This is something that depends on the (usually unknown) environment

Problem in simple terms: Changing the policy will change the visited states, but we usually don't know how



Methods for policy-based RL

Policy gradient theorem

- Allows to calculate the gradient $\nabla_{\boldsymbol{\theta}} L$ without knowing the state distribution change $\frac{\partial s}{\partial \boldsymbol{\theta}}$ as

$$\nabla_{\boldsymbol{\theta}} J \propto \underline{\sum_{s} \mu(s)} \sum_{a} Q_{\pi_{\boldsymbol{\theta}}}(s, a)\, \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s, \boldsymbol{\theta})$$

directly proportional

state distribution (percentage of being in each state)

summation over actions in each state

- Derivation shown on next slide (not important)

Methods for policy-based RL

Derivation of the policy gradient theorem (not important)

Proof of the Policy Gradient Theorem (episodic case)

With just elementary calculus and re-arranging of terms, we can prove the policy gradient theorem from first principles. To keep the notation simple, we leave it implicit in all cases that π is a function of $\boldsymbol{\theta}$, and all gradients are also implicitly with respect to $\boldsymbol{\theta}$. First note that the gradient of the state-value function can be written in terms of the action-value function as

$$
\begin{aligned}
\nabla v_\pi(s) &= \nabla\left[\sum_a \pi(a|s) q_\pi(s,a)\right], \quad \text{for all } s \in \mathcal{S} && \text{(Exercise 3.18)}\\
&= \sum_a \Big[\nabla\pi(a|s) q_\pi(s,a) + \pi(a|s)\nabla q_\pi(s,a)\Big] && \text{(product rule of calculus)}\\
&= \sum_a \Big[\nabla\pi(a|s) q_\pi(s,a) + \pi(a|s)\nabla \sum_{s',r} p(s',r|s,a)\big(r + v_\pi(s')\big)\Big]\\
& && \text{(Exercise 3.19 and Equation 3.2)}\\
&= \sum_a \Big[\nabla\pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p(s'|s,a)\nabla v_\pi(s')\Big] && \text{(Eq. 3.4)}\\
&= \sum_a \Big[\nabla\pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p(s'|s,a) && \text{(unrolling)}\\
&\qquad \sum_{a'} [\nabla\pi(a'|s') q_\pi(s',a') + \pi(a'|s') \sum_{s''} p(s''|s',a')\nabla v_\pi(s'')]\Big]\\
&= \sum_{x\in\mathcal{S}} \sum_{k=0}^{\infty} \Pr(s \to x, k, \pi) \sum_a \nabla\pi(a|x) q_\pi(x,a),
\end{aligned}
$$

after repeated unrolling, where $\Pr(s \to x, k, \pi)$ is the probability of transitioning from state s to state x in k steps under policy π. It is then immediate that

$$
\begin{aligned}
\nabla J(\boldsymbol{\theta}) &= \nabla v_\pi(s_0)\\
&= \sum_s \left(\sum_{k=0}^{\infty} \Pr(s_0 \to s, k, \pi)\right) \sum_a \nabla\pi(a|s) q_\pi(s,a)\\
&= \sum_s \eta(s) \sum_a \nabla\pi(a|s) q_\pi(s,a) && \text{(box page 199)}\\
&= \sum_{s'} \eta(s') \sum_s \frac{\eta(s)}{\sum_{s'}\eta(s')} \sum_a \nabla\pi(a|s) q_\pi(s,a)\\
&= \sum_{s'} \eta(s') \sum_s \mu(s) \sum_a \nabla\pi(a|s) q_\pi(s,a) && \text{(Eq. 9.3)}\\
&\propto \sum_s \mu(s) \sum_a \nabla\pi(a|s) q_\pi(s,a) && \text{(Q.E.D.)}
\end{aligned}
$$

Sutton, Barto: Reinforcement Learning

Methods for policy-based RL

Ostbayerische Technische Hochschule Amberg-Weiden

All-actions policy update

- Reformulation of the policy gradient theorem yields

$$\nabla_{\boldsymbol{\theta}} J \propto \sum_s \mu(s) \sum_a Q_{\pi_{\boldsymbol{\theta}}}(s,a)\, \nabla_{\boldsymbol{\theta}} \pi(a|s,\boldsymbol{\theta})$$

sampling

$$= E_{\pi_{\boldsymbol{\theta}}}\left[\sum_a Q_{\pi_{\boldsymbol{\theta}}}(S_t,a) \nabla_{\boldsymbol{\theta}} \pi(a|S_t,\boldsymbol{\theta})\right]$$

- Policy update through stochastic gradient descent
  - Run an episode
  - At each step of the episode, update weights using stochastic gradient ascent based on the approximated Q-function $\hat{Q}$ and the policy gradient $\nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}$ of each visited state

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \sum_a \hat{Q}_{\pi_{\boldsymbol{\theta}}}(S_t,a,\boldsymbol{w}) \nabla_{\boldsymbol{\theta}} \pi(a|S_t,\boldsymbol{\theta})$$

Problem: All-actions policy update still depends on approximated Q-function $\hat{Q}$

Methods for policy-based RL

REINFORCE

- Like all action-action policy update, but does not depend on an approximated Q-function
- Reformulation of the policy gradient theorem yields

$$\nabla_{\boldsymbol{\theta}} J \propto \sum_s \mu(s) \sum_a Q_{\pi_{\boldsymbol{\theta}}}(s, a) \, \nabla_{\boldsymbol{\theta}} \pi(a | S_t, \boldsymbol{\theta})$$

$$= E_{\pi_{\boldsymbol{\theta}}}[G_t \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})]$$

- Derivation shown on next slide (not important)
- Policy update using REINFORCE
  - Run an episode
  - At each step of the episode, update weights using stochastic gradient ascent based on the sampled return G_t and the policy gradient $\nabla_{\boldsymbol{\theta}} \ln \pi_{\boldsymbol{\theta}}$ of each visited state

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot G_t \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

weighting factor

gradient of the log prob.

Methods for policy-based RL

Derivation of REINFORCE (not important)

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J &\propto \sum_s \mu(s) \sum_a Q_{\pi_{\boldsymbol{\theta}}}(s,a)\, \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}} \\
&= E_{\pi_{\boldsymbol{\theta}}}\left[\sum_a Q_{\pi_{\boldsymbol{\theta}}}(S_t,a) \nabla_{\boldsymbol{\theta}} \pi(a|S_t,\boldsymbol{\theta})\right] \\
&= E_{\pi_{\boldsymbol{\theta}}}\left[\sum_a \pi(a|S_t,\boldsymbol{\theta})\, Q_{\pi_{\boldsymbol{\theta}}}(S_t,a) \frac{\nabla_{\boldsymbol{\theta}} \pi(a|S_t,\boldsymbol{\theta})}{\pi(a|S_t,\boldsymbol{\theta})}\right] && \text{expand with } \frac{\pi(a|S_t,\boldsymbol{\theta})}{\pi(a|S_t,\boldsymbol{\theta})} \\
&= E_{\pi_{\boldsymbol{\theta}}}\left[Q_{\pi_{\boldsymbol{\theta}}}(S_t,A_t) \frac{\nabla_{\boldsymbol{\theta}} \pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta})}\right] && \text{replace } a \text{ by the sample } A_t \sim \pi \\
&= E_{\pi_{\boldsymbol{\theta}}}\left[G_t \frac{\nabla_{\boldsymbol{\theta}} \pi(A_t|S_t,\boldsymbol{\theta})}{\pi(A_t|S_t,\boldsymbol{\theta})}\right] && E_\pi[G_t|S_t,A_t] = Q_\pi(S_t,A_t) \\
&= E_{\pi_{\boldsymbol{\theta}}}[G_t \nabla_{\boldsymbol{\theta}} \ln \pi(A_t|S_t,\boldsymbol{\theta})\,] && \nabla \ln x = \frac{1}{x} \nabla x
\end{aligned}
$$

Methods for policy-based RL

REINFORCE in literature

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s,\boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot,\boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t,\boldsymbol{\theta})$

Sutton, Barto: Reinforcement Learning

Methods for policy-based RL

Baselines

- Recap: Policy gradient theorem

$$\nabla_{\boldsymbol{\theta}} J \propto \sum_s \mu(s) \sum_a Q_{\pi_{\boldsymbol{\theta}}}(s,a)\, \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}$$

- Problem with above formulation: Huge variation of the gradient magnitude $|\nabla_{\boldsymbol{\theta}} L|$ across different states depending on $Q_{\pi_{\boldsymbol{\theta}}}(s,a)$ → numerical problems
- Solution: Introduce a baseline (a function $b(s)$ that only depends on the state) to make the gradient magnitude more similar across all states

$$\nabla_{\boldsymbol{\theta}} J \propto \sum_s \mu(s) \sum_a \big(Q_{\pi_{\boldsymbol{\theta}}}(s,a) - b(s)\big)\, \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}$$

- Does not change expected value of $\nabla_{\boldsymbol{\theta}} L$, because

$$\sum_a b(s)\, \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}} = b(s) \nabla_{\boldsymbol{\theta}} \sum_a \pi_{\boldsymbol{\theta}} = b(s) \nabla_{\boldsymbol{\theta}} 1 = 0$$

Methods for policy-based RL

- Common type of baseline: V-function

$$b(s) = V_\pi(s)$$

- Resulting term $Q_\pi(s,a) - V_\pi(s)$ is called **advantage function** $A_\pi(s,a)$

- Resulting update rule for REINFORCE with baseline (with approximate value function $\hat{V}_{\pi_{\boldsymbol{\theta}}}(s, \boldsymbol{w})$)

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \left(G_t - \hat{V}_{\pi_{\boldsymbol{\theta}}}(S_t, \boldsymbol{w})\right) \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

For comparison: REINFORCE update rule without baseline

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot G_t \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

- Challenge: Two neural networks to learn:
  Policy $\pi(a|s, \boldsymbol{\theta})$ and V-function approximation $\hat{V}_{\pi_{\boldsymbol{\theta}}}(s, \boldsymbol{w})$

Methods for policy-based RL

REINFORCE with baseline in literature

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)

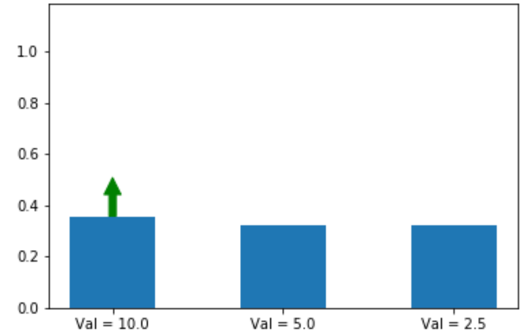
Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \dots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
        $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t,\mathbf{w})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Sutton, Barto: Reinforcement Learning

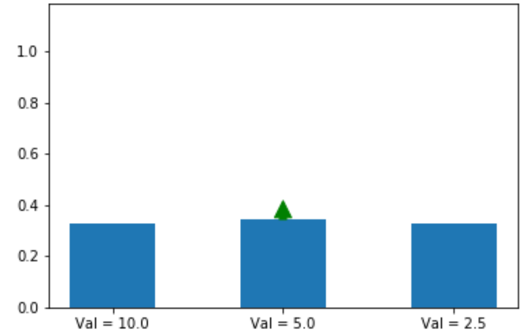
Methods for policy-based RL

Intuitive derivation for REINFORCE and baselines

1. Start with informed gradient ascent:
   Increase probability of taking the optimal action
   $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \nabla_{\boldsymbol{\theta}} \pi(a^* | s, \boldsymbol{\theta})$$
   Problem: Optimal action is not known



2. Weight gradient with Q-function:
   If the action corresponds to a high estimated Q-value the gradient step will be large, otherwise small
   $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \hat{Q}_{\pi_{\boldsymbol{\theta}}}(s, a, \boldsymbol{w}) \nabla_{\boldsymbol{\theta}} \pi(a | s, \boldsymbol{\theta})$$



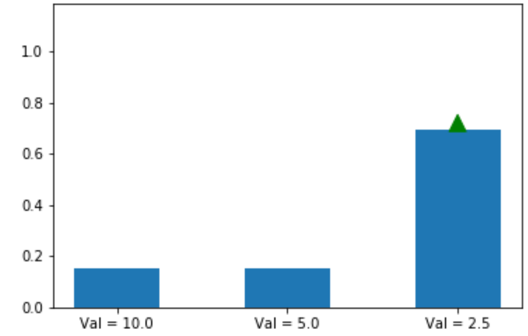
https://towardsdatascience.com/an-intuitive-explanation-of-policy-gradient-part-1-reinforce-aa4392cbfd3c

Methods for policy-based RL

3. Problem with

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \hat{Q}_{\pi_{\boldsymbol{\theta}}}(s, a, \boldsymbol{w}) \nabla_{\boldsymbol{\theta}} \pi(a|s, \boldsymbol{\theta})$$

It does not account for the amount of updates:
If a suboptimal action has an initial high probability of being taken, it might get taken/updated more frequently, increasing its probability even more

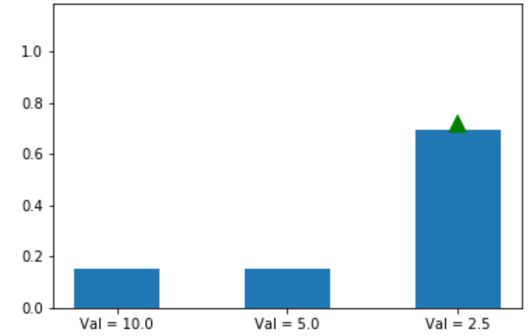


4. Weight gradient with inverse of update rate:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \hat{Q}_{\pi_{\boldsymbol{\theta}}}(s, a, \boldsymbol{w}) \frac{\nabla_{\boldsymbol{\theta}} \pi(a|s, \boldsymbol{\theta})}{\pi(a|s, \boldsymbol{\theta})}$$

This alleviates the problem described in 3.

With $\frac{\nabla x}{x} = \ln x$ and G_t instead of $Q_{\pi_{\boldsymbol{\theta}}}(s, a)$
this is the REINFORCE update rule



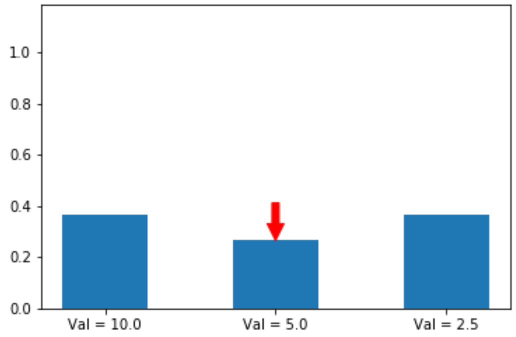
https://towardsdatascience.com/an-intuitive-explanation-of-policy-gradient-part-1-reinforce-aa4392cbfd3c

Methods for policy-based RL

5. Effect of advantage function

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \left(\hat{Q}_{\pi_{\boldsymbol{\theta}}}(s, a, \boldsymbol{w}) - \hat{V}(s, \boldsymbol{w}')_{\pi_{\boldsymbol{\theta}}}\right) \frac{\nabla_{\boldsymbol{\theta}} \pi(a|s, \boldsymbol{\theta})}{\pi(a|s, \boldsymbol{\theta})}$$

Actions corresponding to bad Q-values are now actively decreased in probability through gradient descent

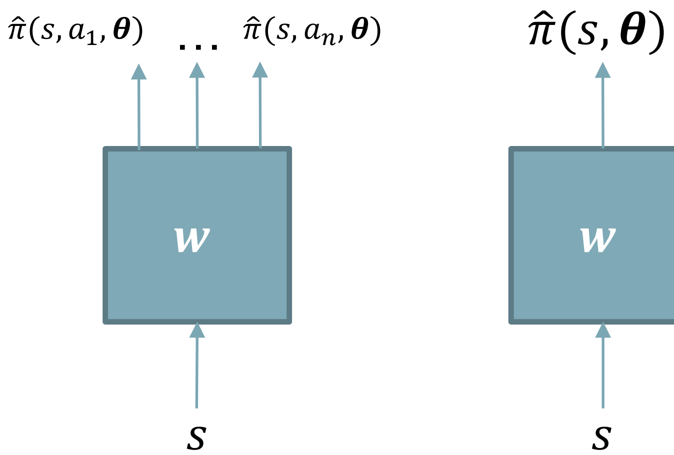


https://towardsdatascience.com/an-intuitive-explanation-of-policy-gradient-part-1-reinforce-aa4392cbfd3c

Methods for policy-based RL

Different types of trained neural networks for $\hat{\pi}(s, \boldsymbol{\theta})$

- Discrete action space (e.g. REINFORCE):
  Neural network with one output per action, representing its probability
- Continuous action space (e.g. Actor-Critic):
  Neural network with one output corresponding to the currently best action



Methods for policy-based RL

Actor-critic methods

- Actor-critic = REINFORCE + baseline + Bellman expectation equation
- Recap: REINFORCE + baseline update rule

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \big(G_t - \hat{V}_{\pi_{\boldsymbol{\theta}}}(S_t, \boldsymbol{w})\big) \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

- Actor-critic update rule

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \big(R_{t+1} + \hat{V}_{\pi_{\boldsymbol{\theta}}}(S_{t+1}, \boldsymbol{w}) - \hat{V}_{\pi_{\boldsymbol{\theta}}}(S_t, \boldsymbol{w})\big) \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \cdot \ldots$$

- Basis for many state-of-the-art reinforcement algorithms (e.g. A2C, A3C, DDPG, PPO, TRPO, TD3, SAC, …)

Methods for policy-based RL

Actor-critic in literature

One-step Actor–Critic (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$ (e.g., to $\mathbf{0}$)
Loop forever (for each episode):
    Initialize S (first state of episode)
    $I \leftarrow 1$
    Loop while S is not terminal (for each time step):
        $A \sim \pi(\cdot|S, \boldsymbol{\theta})$
        Take action A, observe S', R
        $\delta \leftarrow R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S,\mathbf{w})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} I \delta \nabla \ln \pi(A|S, \boldsymbol{\theta})$
        $I \leftarrow \gamma I$
        $S \leftarrow S'$

Sutton, Barto: Reinforcement Learning

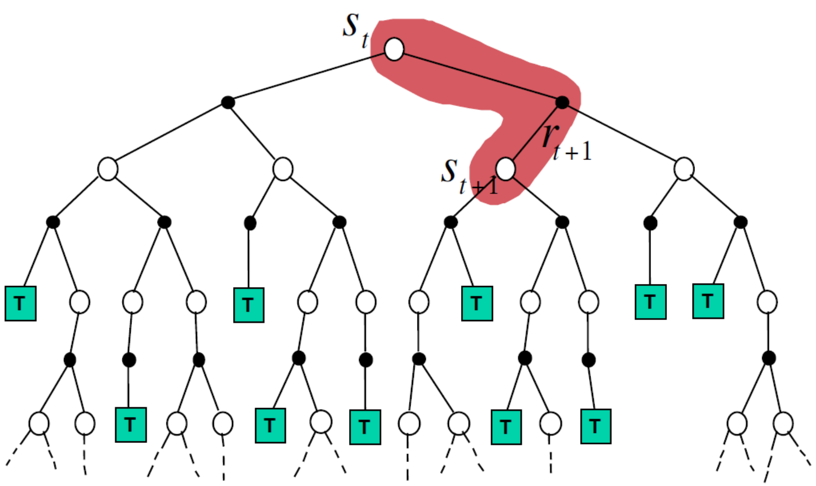
Methods for policy-based RL

Comparison of actor-critic and REINFORCE

- Actor-critic methods are the analog of temporal difference methods for policy gradients
- REINFORCE is the analog of Monte-Carlo methods for policy gradients

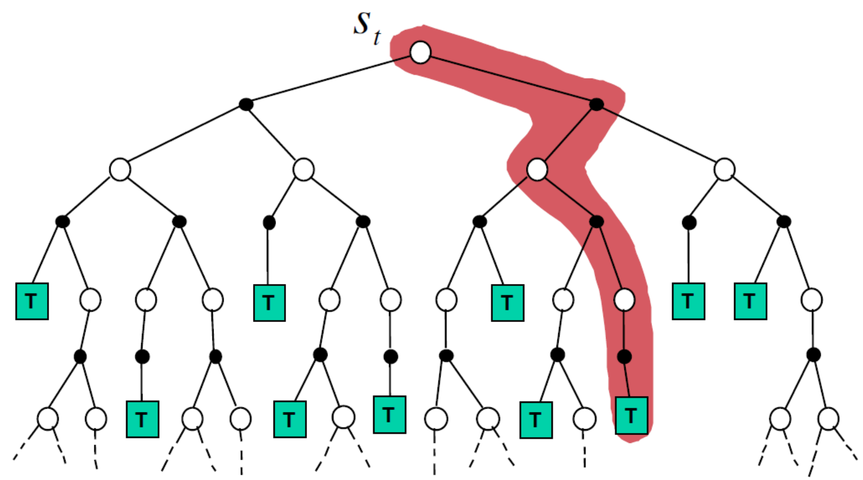
Actor-critic

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot \left(R_{t+1} + \gamma \hat{V}(S_{t+1}, \boldsymbol{w}) - \hat{V}(S_t, \boldsymbol{w}')\right) \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$



REINFORCE

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \cdot G_t \nabla_{\boldsymbol{\theta}} \ln \pi(A_t | S_t, \boldsymbol{\theta})$$

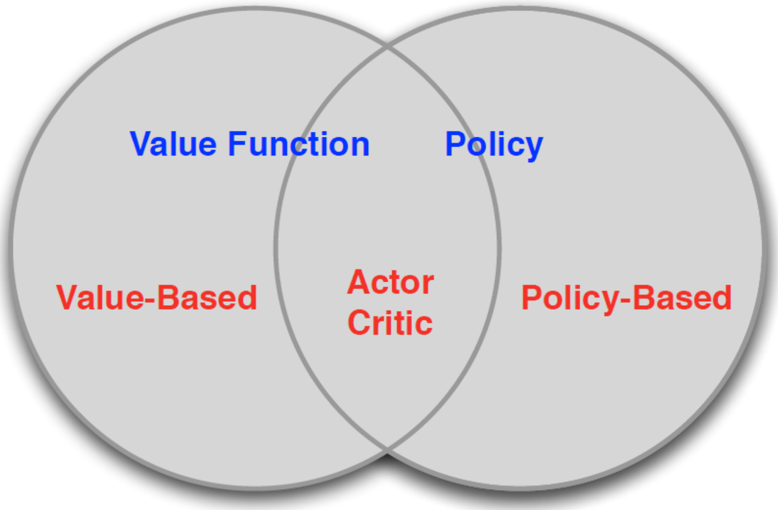


https://www.deepmind.com/learning-resources/introduction-to-reinforcement-learning-with-david-silver

Methods for policy-based RL

Comparison of value-based, policy-based and actor-critic methods

- value-based
  - learned value function
  - implicit policy (e.g. ϵ-greedy)
- policy-based
  - no value function
  - learned policy
- actor-critic
  - learned value function
  - learned policy



https://www.deepmind.com/learning-resources/introduction-to-reinforcement-learning-with-david-silver

Which one is best?

- Usually actor-critic, but depends on use-case (discrete/continuous states/actions)

Methods for policy-based RL

Task: What are the possible (dis-)advantages of value-based / policy-based / actor critic methods?

Methods for policy-based RL

Brief summary

- Policy-based methods directly update the policy without (in principle) having to calculate the V-/Q-function first (policy gradient)
- The policy gradient theorem is the foundation of all policy gradient methods as it allows to calculate the policy gradient based on sampled trajectories
- REINFORCE = policy gradient theorem + return instead of Q-function
- REINFORCE + baseline = REINFORCE + baseline
- Actor-critic = REINFORCE + baseline + Bellman expectation equation

Kahoot!