An Introduction to Reinforcement Learning

Ludwig Winkler

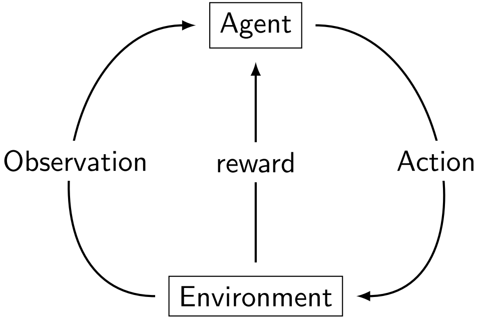
Machine Learning Group
TU Berlin

August 27, 2017

Outline

Intuition

- In between supervised and unsupervised learning
- Take actions in an environment that maximize reward
  - Actions $\mapsto$ Policy
  - Environment $\mapsto$ States
  - Reward $\mapsto$ Feedback from environment



Notation

Action	$a \in \mathcal{A}$
State	$s \in \mathcal{S}$
Policy	$\pi(a\|s)$
Transition model	$T(s_j\|s_i, a_k)$
Reward function	$R(s_i, a_k)$
Value function	$V(s_i)$
Action-Value function	$Q(s_i, a_k)$
Discount factor	$\gamma \in [0, 1)$
Learning rate	$\eta \in \mathbb{R}^+$

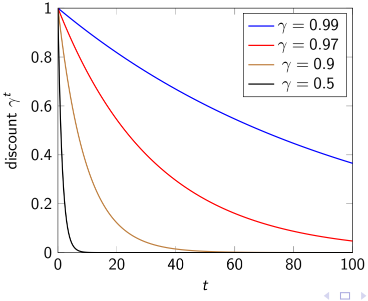
Markov Decision Processes

- Markov Property: *Future is independent of the past given the present*
- Policy: $\pi(a_k|s_i)$
  - Probability of taking action a_k in state s_i
- Transition model: $T(s_j|s_i, a_k)$
  - Probability of going to next state s_j given action a_k in state s_i
- Reward model: $R(s_i, a_k)$
  - Reward for taking action a_k in state s_i
- Discount factor γ
  - Difference of importance between future and present rewards
- MDP: 5-Tuple $(S, A, T(s_j|s_i, a_k), R(s_i, a_k), \gamma)$

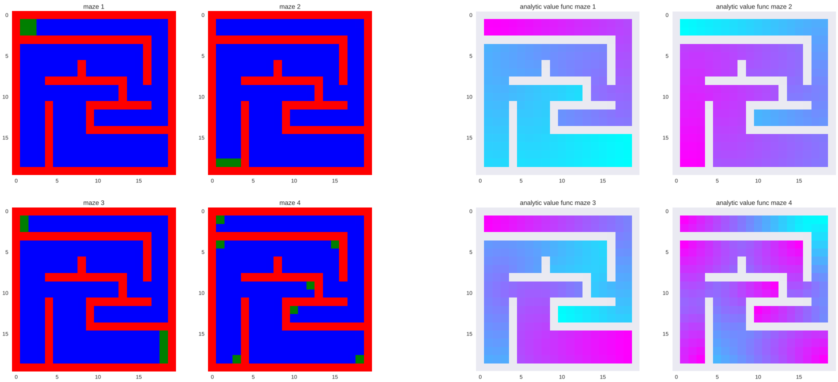
Value function

- Estimation of future rewards following policy π from state $s^{(0)}$
- Geometric weighting of future rewards
- Expected infinite sum of discounted future rewards:

$$V^\pi(s^{(0)}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right], \gamma \in [0, 1)$$



Value Function - Grid World Example



Bellman Equation

- Knowledge of environment to construct $T(s_j|s_i, a_k)$ and $R(s_i, a_k)$
- $V^\pi(s_i)$ is the future, discounted reward expected in state $s_i = s^{(0)}$:

$$V^\pi(s_i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right]$$

Bellman Equation

- Knowledge of environment to construct $T(s_j|s_i, a_k)$ and $R(s_i, a_k)$
- $V^\pi(s_i)$ is the future, discounted reward expected in state $s_i = s^{(0)}$:

$$
\begin{aligned}
V^\pi(s_i) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right] \\
&= \mathbb{E}\left[R(s^{(0)}, a^{(0)}) \middle| a^{(0)} \sim \pi(\cdot|s)\right] \\
&\quad + \gamma \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right]
\end{aligned}
$$

Bellman Equation

- Knowledge of environment to construct $T(s_j|s_i, a_k)$ and $R(s_i, a_k)$
- $V^\pi(s_i)$ is the future, discounted reward expected in state $s_i = s^{(0)}$:

$$\begin{aligned}
V^\pi(s_i) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right] \\
&= \mathbb{E}\left[R(s^{(0)}, a^{(0)}) \middle| a^{(0)} \sim \pi(\cdot|s)\right] \\
&\quad + \gamma \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right] \\
&= \mathbb{E}_\pi\left[R(s^{(0)}, a^{(0)})\right] + \gamma \mathbb{E}_\pi\left[V^\pi(s^{(1)}) \middle| s^{(1)} \sim T(\cdot|s, a)\right]
\end{aligned}$$

Bellman Equation

- Knowledge of environment to construct $T(s_j|s_i, a_k)$ and $R(s_i, a_k)$
- $V^\pi(s_i)$ is the future, discounted reward expected in state $s_i = s^{(0)}$:

$$
\begin{aligned}
V^\pi(s_i) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right] \\
&= \mathbb{E}\left[R(s^{(0)}, a^{(0)}) \middle| a^{(0)} \sim \pi(\cdot|s)\right] \\
&\quad + \gamma \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} a^{(t)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right] \\
&= \mathbb{E}_\pi\left[R(s^{(0)}, a^{(0)})\right] + \gamma \mathbb{E}_\pi\left[V^\pi(s^{(1)}) \middle| s^{(1)} \sim T(\cdot|s, a)\right] \\
&= \sum_{k=1}^{A} \pi(a_k|s_i) \left(R(s_i, a_k) + \gamma \sum_{j=1}^{S} T(s_j|s_i, a_k) V^\pi(s_j) \right)
\end{aligned}
$$

Bellman Equation

$$V^\pi(s_i) = \sum_{k=1}^{A} \pi(a_k|s_i)\left(R(s_i,a_k) + \gamma \sum_{j=1}^{S} T(s_j|s_i,a_k)V^\pi(s_j)\right)$$

$$= \underbrace{\sum_{k=1}^{A} \pi(a_k|s_i)R(s_i,a_k)}_{\text{policy controlled reward } R^\pi} + \gamma \underbrace{\sum_{j=1}^{S}\sum_{k=1}^{A} \pi(a_k|s_i)T(s_j|s_i,a_k)}_{\text{policy controlled transition } T^\pi} V^\pi$$

- Can be solved analytically:

$$V^\pi = R^\pi + \gamma\, T^\pi\, V^\pi \Leftrightarrow V^\pi = (I - \gamma T^\pi)^{-1} R^\pi$$

- Or via value iteration:

$$V^\pi \leftarrow R^\pi + \gamma\, T^\pi V^\pi$$

Model-based Reinforcement Learning

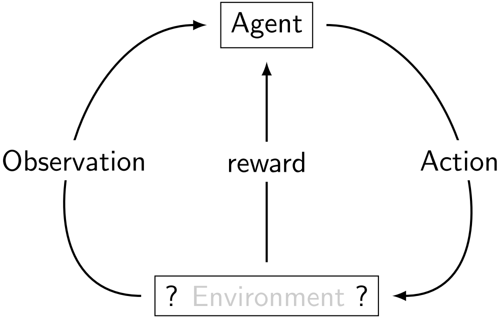
- Bellman operator $\widehat{B}^{\pi}[V^{\pi}]$ is a contraction mapping

$$\widehat{B}^{\pi}[V^{\pi}] = R^{\pi} + \gamma \, T^{\pi} V^{\pi}$$

- Optimal policy π^* can be derived from $V^{\pi^*}(s)$
  - Take action that such that next state has highest value
- Requires explicit transition and reward function for all states
  - Heat dissipation formulas for engine control
  - Avionic formulas for helicopter control

Model-Free Methods

- What if transition and reward functions are unknown?
- Agent has to learn by interacting with the environment
- Instead of being provided with model, agent builds its own model



Temporal Difference Learning - TD(0)

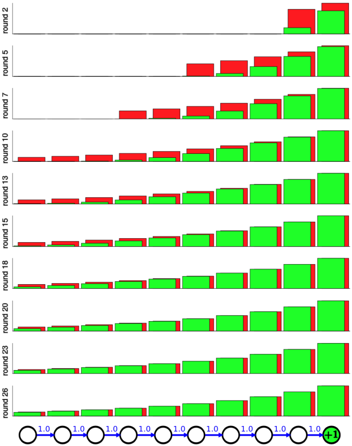
- Iterative difference between state values

$$\widehat{V}^{\pi}(s^{(t)}) \leftarrow \widehat{V}^{\pi}(s^{(t)}) + \eta\Big[\underbrace{\overbrace{R(s^{(t)}, a^{(t)}) + \gamma \widehat{V}^{\pi}(s^{(t+1)})}^{\text{Boot-strapped Backup}} - \widehat{V}^{\pi}(s^{(t)})}_{\text{TD-Error } \Delta\widehat{V}^{\pi}(s)}\Big]$$

- $\gamma \widehat{V}^{\pi}(s^{(t+1)})$ serves as an estimate for remaining future rewards $\gamma R(s^{(t+2)}, a^{(t+2)}) + \gamma^2 R(s^{(t+3)}, a^{(t+3)}) + \dots$
- Influence of learning rate η:
  - Large η: fast learning, large variance
  - Small η: slow learning, small variance
  - Decaying learning rate not practical due to non-stationarity

Temporal Difference Learning - Value Propagation

- Example:
  - 10 states, 1 action (forward)
  - Reward in last state
  - $\gamma = 0.9$
  - $\eta = 1$, $\eta = 0.5$
- Value Propagation requires:
  - 10 rounds for $\eta = 1$
  - 26 rounds for $\eta = 0.5$
- Source:[1]



n-step Temporal Difference Learning

- TD(0) uses only immediate next reward
- Temporal difference can be extended over *n*-steps

$$G_n^{(t)} = \sum_{\tau=0}^{n-1} \gamma^\tau R(s^{(t+\tau)}, a^{(t+\tau)}) + \gamma^n \widehat{V}^\pi(s^{(t+n)})$$

$$\text{e.g.} \quad G_1^{(t)} = R(s^{(t+1)}, a^{(t+1)}) + \gamma \widehat{V}^\pi(s^{(t+1)})$$

$$G_2^{(t)} = R(s^{(t+1)}, a^{(t+1)}) + \gamma R(s^{(t+2)}, a^{(t+2)}) + \gamma^2 \widehat{V}^\pi(s^{(t+2)})$$

$$\dots$$

- $\gamma^n \widehat{V}^\pi(s^{(t+n)})$ as estimate for future rewards at time steps $t > n$

n-step Temporal Difference Learning & TD(λ)

- Value function is updated with n-step return $G_n^{(t)}$

$$\widehat{V}^{\pi}(s^{(t)}) \leftarrow \widehat{V}^{\pi}(s^{(t)}) + \eta\Big[\underbrace{\overbrace{G_n^{(t)} + \gamma^n \widehat{V}^{\pi}(s^{(t+n)})}^{n\text{-step Backup}} - \widehat{V}^{\pi}(s^{(t)})}_{\text{TD-Error } \Delta\widehat{V}^{\pi}(s)}\Big]$$

- Use weighted mean with decaying weights

$$G_{\lambda}^{(t)} = (1-\lambda)\sum_{\tau=0}^{\infty} \lambda^{\tau} R(s^{(t+\tau)}, a^{(t+\tau)}) \qquad , \lambda \in [0,1)$$

- Smaller λ favor more immediate rewards
- Practically rewards are considered until $\lambda^{\tau} \approx 0$

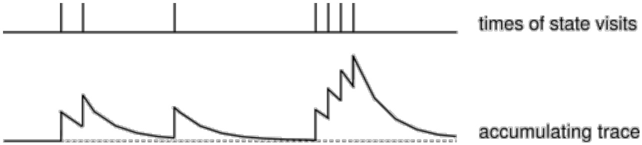
TD(λ) & Eligibility Traces

- Cumbersome *forward* calculation of $G_n^{(t)}$ for each step t
- *Backward* view TD(λ) with eligibility traces more efficient

$$\widehat{V}^{\pi}(s) = \widehat{V}^{\pi} + \eta\ e^{(t)}(s)\left(R(s^{(t)}, a^{(t)}) + \gamma \widehat{V}^{\pi}(s^{(t+1)}) - \widehat{V}^{\pi}(s^{(t)})\right)$$

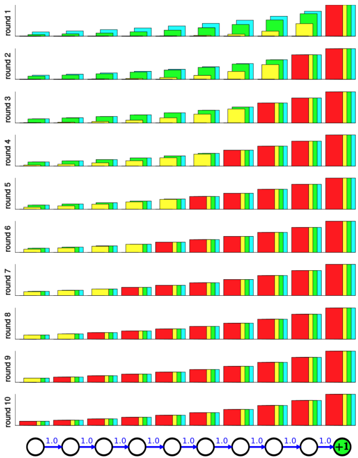
$$e^{(t)}(s) = \begin{cases} \gamma\lambda e^{(t-1)}(s) & \text{if } s \neq s^{(t)} \\ \gamma\lambda e^{(t-1)}(s) + 1 & \text{if } s = s^{(t)} \end{cases}$$

- 'Measures how eligible a state is for the accumulated reward'



TD(λ) - Value Propagation

- Example:
  - 10 states, 1 action (forward)
  - Reward in last state
  - $\gamma = 1, \eta = 1$
- Value Propagation requires:
  - 1 round for $\lambda = 1$
  - 4 rounds for $\lambda = 0.9$
  - 7 rounds for $\lambda = 0.5$
  - 10 rounds for $\lambda = 0$
- Source: [1]



Q-Value function

- $V^\pi(s_i)$ only provides state values
  - No information about value of possible actions a_k
- $Q^\pi(s_i, a_k)$ provides value for action a_k in state s_i

$$Q^\pi(s_i, a_k) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^{(t)}, a^{(t)}) \middle| \begin{array}{c} s^{(0)} = s_i; a^{(0)} = a_k \\ a^{(t+1)} \sim \pi(\cdot|s) \\ s^{(t+1)} \sim T(\cdot|s, a) \end{array}\right]$$

- More expressive but also more data needed

SARSA

- TD(0) learning for Q-values with TD-Errors

$$\widehat{Q}^{\pi}(s^{(t)}, a^{(t)}) \leftarrow \widehat{Q}^{\pi}(s^{(t)}, a^{(t)}) + \eta \Big(\underbrace{\overbrace{R(s^{(t)}, a^{(t)}) + \gamma \widehat{Q}^{\pi}(s^{(t+1)}, a^{(t+1)})}^{\text{Boot-strapped Back-up}} - \widehat{Q}^{\pi}(s^{(t)}, a^{(t)})}_{\text{TD-Error } \Delta \widehat{Q}^{\pi}(s^{(t)}, a^{(t)})} \Big)$$

- SARSA(λ) with multi-step methods
- On-Policy learning
  - Behavior and evaluation policy are the same
  - Behavior policy: Taking steps according to $\widehat{Q}^{\pi}$
  - Evaluation policy: Evaluate discounted temporal difference

Q-Learning

- Off-Policy learning
  - Behavior and evaluation policy are different
  - Behavior policy: Taking steps according to some policy π
  - Evaluation policy: Evaluate against maximum Q-Value
  - Greedy evaluation but 'curious' behaviour
- Q-Learning finds optimal policy π^* independent of behavioral policy

$$\widehat{Q}^\pi(s^{(t)}, a^{(t)}) \leftarrow \widehat{Q}^\pi(s^{(t)}, a^{(t)}) + \eta\Big(\underbrace{\overbrace{R(s^{(t)}, a^{(t)}) + \gamma \max_{a^{(t+1)}} \widehat{Q}^\pi(s^{(t+1)}, a^{(t+1)})}^{\text{Boot-strapped Back-up}} - \widehat{Q}^\pi(s^{(t)}, a^{(t)})}_{\text{TD-Error } \Delta\widehat{Q}^\pi(s^{(t)}, a^{(t)})}\Big)$$

- Q(λ) with multi-step methods

Exploration-Exploitation Dilemma

- Exploiting known policy vs exploring better policies
- Yet unknown policy might yield even higher rewards
- Sampling strategy should balance both
- Find expected rewards of policy with high values
- Find potential reward of policy of alternative actions

Exploration-Exploitation Dilemma - ϵ-greedy and softmax

- ϵ-greedy policy: Exploration controlled with parameter $\epsilon \in [0, 1]$

$$a = \begin{cases} \max_{a'} \widehat{Q}^{\pi}(s, a') & \text{with probability } 1 - \epsilon \\ \text{Random } a & \text{with probability } \epsilon \end{cases}$$

- The larger ϵ the more random actions are taken
- Softmax policy: Exploration controlled with parameter $\beta \in \mathbb{R}^{+}$

$$\pi(a|s) = \frac{\exp(\beta \ \widehat{Q}^{\pi}(s, a))}{\sum_{a'}^{A} \exp(\beta \ \widehat{Q}^{\pi}(s, a'))}$$

- The smaller β the more random actions are taken
- Decreasing exploration over time for agent

Exploration-Exploitation Dilemma - Optimistic Initialization

- Initialize all Q-Values with high values
- Unexplored actions are all very attractive
- TD-Error $\Delta\widehat{V}^{\pi}(s)$ can be negative

$$\Delta\widehat{Q}^{\pi}(s,a) = R(s^{(t)}, a^{(t)}) + \gamma\widehat{Q}^{\pi}(s^{(t+1)}, a^{(t+1)}) - \widehat{Q}^{\pi}(s^{(t)}, a^{(t)})$$

- Initially slower, but faster convergence to limit
- Works with SARSA(λ) and Q(λ)-Learning

Approximate Reinforcement Learning

- Previous methods worked all tabular
- Not recommendable for large or continuous state-action spaces
  - Board game Go on 19×19 board: $3^{19\times19} \approx 10^{172}$
  - Autonomous helicopter control is continuous
- Generalization is required for more complex problems
- Wide catalogue of function approximations available, but
  - Non-stationarity
  - Delayed targets
  - Boot-strapping

Value Function Approximation

- Parameterization of value function with $\boldsymbol{\theta}$: $\widehat{V}^{\pi}(s; \boldsymbol{\theta})$
- Minimize MSE between approximation $\widehat{V}^{\pi}(s; \boldsymbol{\theta})$ and true $V^{\pi}(s)$

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\pi}\left[\left(V^{\pi}(s) - \widehat{V}^{\pi}(s; \boldsymbol{\theta})\right)^2\right]$$

- Gradient descent finds the local minimum

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\pi}\left[\left(V^{\pi}(s) - \widehat{V}^{\pi}(s; \boldsymbol{\theta})\right)^2\right] \\ &= 2\ \mathbb{E}_{\pi}\left[\left(V^{\pi}(s) - \widehat{V}^{\pi}(s; \boldsymbol{\theta})\right) \nabla_{\boldsymbol{\theta}} \widehat{V}^{\pi}(s; \boldsymbol{\theta})\right] \end{aligned}$$

- Stochastic gradient descent with update rule

$$\Delta \boldsymbol{\theta} = \alpha \left(V^{\pi}(s) - \widehat{V}^{\pi}(s; \boldsymbol{\theta})\right) \nabla_{\boldsymbol{\theta}} \widehat{V}^{\pi}(s; \boldsymbol{\theta})$$

Value Function Approximation

- Target value function $V^\pi(s)$ unknown
- $V^\pi(s)$ is approximated with current reward
- TD(0): $V^\pi(s^{(t)}) \approx R(s^{(t)}, a^{(t)}) + \gamma \widehat{V}^\pi(s^{(t+1)}; \boldsymbol{\theta})$

$$\Delta\boldsymbol{\theta} = \alpha \Big(\underbrace{R(s^{(t)}, a^{(t)}) + \gamma \widehat{V}^\pi(s^{(t+1)}; \boldsymbol{\theta}) - \widehat{V}^\pi(s^{(t)}; \boldsymbol{\theta})}_{\text{TD-Error}\Delta\widehat{V}^\pi(s)} \Big) \nabla_{\boldsymbol{\theta}} \widehat{V}^\pi(s^{(t)}; \boldsymbol{\theta})$$

- n-step TD: $V^\pi(s^{(t)}) \approx G_n^{(t)}$

$$\Delta\boldsymbol{\theta} = \alpha \left(G_n^{(t)} - \widehat{V}^\pi(s^{(t+n)}; \boldsymbol{\theta}) \right) \nabla_{\boldsymbol{\theta}} \widehat{V}^\pi(s; \boldsymbol{\theta})$$

Action-Value Function Approximation

- Target value function $Q^\pi(s, a)$ unknown
- $Q^\pi(s, a)$ is approximated with current reward
- TD(0): $Q^\pi(s^{(t)}, a^{(t)}) \approx R(s^{(t)}, a^{(t)}) + \gamma \widehat{Q}^\pi(s^{(t+1)}, a^{(t+1)}; \boldsymbol{\theta})$

$$\Delta \boldsymbol{\theta} = \alpha \underbrace{\left(R(s^{(t)}, a^{(t)}) + \gamma \widehat{Q}^\pi(s^{(t+1)}, a^{(t+1)}; \boldsymbol{\theta}) - \widehat{Q}^\pi(s^{(t)}, a^{(t)}; \boldsymbol{\theta}) \right)}_{\text{TD-Error } \Delta \widehat{Q}^\pi(s^{(t)}, a^{(t)})} \nabla_{\boldsymbol{\theta}} \widehat{Q}^\pi(s^{(t)}, a^{(t)}; \boldsymbol{\theta})$$

- n-step TD: $Q^\pi(s^{(t)}, a^{(t)}) \approx G_n^{(t)}$

$$\Delta \boldsymbol{\theta} = \alpha \left(G_n^{(t)} - \widehat{Q}^\pi(s^{(t+n)}, a^{(t+n)}; \boldsymbol{\theta}) \right) \nabla_{\boldsymbol{\theta}} \widehat{Q}^\pi(s, a; \boldsymbol{\theta})$$

Policy Gradient Methods

- Previously determinstic value function approximation with implied policy
- Approximation of policy

$$\pi_{\boldsymbol{\theta}}(s, a) = \mathbb{P}\left[a|s; \boldsymbol{\theta}\right]$$

- Does not learn a value function, but directly a policy

- More effective in high-dimensional or continuous spaces
- Better convergence
- Can learn stochastic policies

Policy Gradient Methods

- Policy quality measured with objective function $J^{\pi_\theta}(s^{(0)})$
- Return measured by expectancy of reward over policy

$$J^{\pi_\theta}(s^{(0)}) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)})\right]$$
$$= \underbrace{\sum_{a^{(0:T)}} \pi_\theta(a^{(0:T)}|s^{(0:T)})}_{\text{controlled by policy}} \left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)})\right]$$

- $\sum_{a^{(0:T)}}$ are all possible actions $a^{(t)}$ at time step t with $t \in \{0, \ldots, T\}$
- Episodic sampling of trajectories

Policy Gradient Methods

- Policy gradient

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J^{\pi_{\boldsymbol{\theta}}}(s^{(0)}) &= \sum_{a^{(0:T)}} \underbrace{\nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a^{(0:T)} | s^{(0:T)})}_{(\log f(x))' = \frac{f(x)'}{f(x)}} \left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)}) \right] \\
&= \sum_{a^{(0:T)}} \pi_{\boldsymbol{\theta}}(a^{(0:T)} | s^{(0:T)}) \left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)}) \right] \nabla_{\boldsymbol{\theta}} \log \left[\pi_{\boldsymbol{\theta}}(a^{(0:T)} | s^{(0:T)}) \right] \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)}) \; \nabla_{\boldsymbol{\theta}} \log \left[\pi_{\boldsymbol{\theta}}(a^{(0:T)} | s^{(0:T)}) \right] \right]
\end{aligned}
$$

- Sampling of trajectories with corresponding rewards necessary
- Trajectory sampling inefficient and high-variance

Actor-Critic Methods

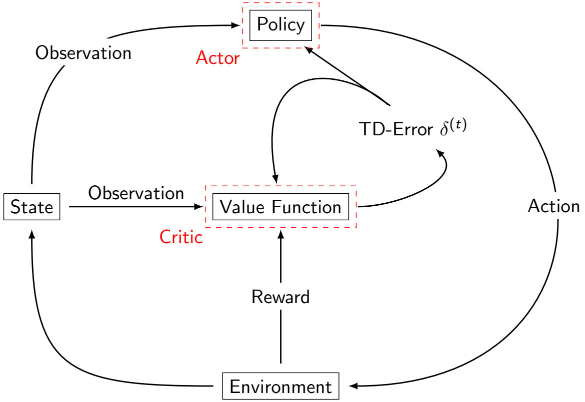
- Compatible Function Approximation Theorem:
  Replace trajectory reward with long-term value $Q^{\pi_\theta}(s^{(t)}, a^{(t)})$

$$\nabla_{\boldsymbol{\theta}} J^{\pi}(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} R(s^{(t)}, a^{(t)})\ \nabla_{\boldsymbol{\theta}} \log\left[\pi_{\boldsymbol{\theta}}(a^{(0:T)}|s^{(0:T)})\right]\right]$$
$$\approx \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[Q^{\pi_\theta}(s^{(t)}, a^{(t)}) \nabla_{\boldsymbol{\theta}} \log\left[\pi_{\boldsymbol{\theta}}(a^{(t)}|s^{(t)})\right]\right]$$

- Actor $\pi_{\boldsymbol{\theta}}$ performs and critic $Q^{\pi_\theta}(s, a)$ evaluates
- Critic determines value function and policy update
- Critic reduces variance

Actor-Critic Methods

$$\delta^{(t)} = R(s^{(t)}, a^{(t)}) + \gamma \widehat{Q}^{\pi}(s^{(t+1)}, a^{(t+1)}) - \widehat{Q}^{\pi}(s^{(t)}, a^{(t)})$$



End-to-End Reinforcement Learning

Human-level control through deep reinforcement learning

- Neural network architecture and training
  - Input($84 \times 84 \times 4$) $\rightarrow$ CL($8 \times 8 \times 32$) $\rightarrow$ CL($4 \times 4 \times 64$) $\rightarrow$ CL($3 \times 3 \times 64$) $\rightarrow$ FC(512) $\rightarrow$ Output(4-18)
  - ReLu activation function and RMSProp
- Experience Replay
  - Store transitions $(s^{(t)}, a^{(t)}, R(s^{(t)}, a^{(t)}), s^{(t+1)})$ in $\mathcal{D}$
  - Sample from $\mathcal{D}$ for mini-batches
- Fixed Q-Targets
  - Target Q-Values in TD-Error change every 10.000 iterations

$$\boldsymbol{\Delta\theta} = \mathbb{E}_{\mathcal{D}}\left[\left(R(s^{(t)}, a^{(t)}) + \gamma \underbrace{\max_{a^{(t+1)}} \widehat{Q}^{\pi}(s^{(t+1)}, a^{(t+1)}; \boldsymbol{\theta}^{-})}_{\text{Fixed}} - \widehat{Q}^{\pi}(s^{(t)}, a^{(t)}; \boldsymbol{\theta})\right) \nabla_{\boldsymbol{\theta}} \widehat{Q}^{\pi}(s^{(t)}, a^{(t)}; \boldsymbol{\theta})\right]$$

Sources

- [1] Machine Intelligence I (TU Berlin - WS16/17)
  Klaus Obermayer & Wendelin Böhmer
- [2] Reinforcement Learning (UCL - SS16)
  David Silver
- [3] Deep Learning at Oxford (Oxford - SS15)
  Nando de Freitas