Introduction to Machine Learning

Session 4: Reinforcement Learning & Ethics

Benjamin Paaßen

The University of Sydney

IK 2020, Günne

Reinforcement Learning Basics

Motivation: Playing games

- How do I model an agent who should play a game?

Motivation: Playing games

- How do I model an agent who should play a game?
- Very **sparse** supervision: Game score or winning/losing

Motivation: Playing games

- How do I model an agent who should play a game?
- Very **sparse** supervision: Game score or winning/losing
- Future depends on agents' actions $\Rightarrow$ data are not i.i.d.

Motivation: Playing games

- How do I model an agent who should play a game?
- Very **sparse** supervision: Game score or winning/losing
- Future depends on agents' actions $\Rightarrow$ data are not i.i.d.
- General idea: **Reinforce** moves that end up in winning games, **punish** moves that end up in losing games

Motivation: Playing games

- How do I model an agent who should play a game?
- Very **sparse** supervision: Game score or winning/losing
- Future depends on agents' actions $\Rightarrow$ data are not i.i.d.
- General idea: **Reinforce** moves that end up in winning games, **punish** moves that end up in losing games $\Rightarrow$ reinforcement learning

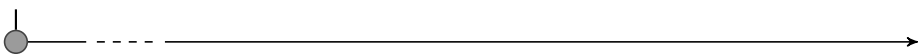
Motivation: Playing games

- How do I model an agent who should play a game?
- Very **sparse** supervision: Game score or winning/losing
- Future depends on agents' actions $\Rightarrow$ data are not i.i.d.
- General idea: **Reinforce** moves that end up in winning games, **punish** moves that end up in losing games $\Rightarrow$ reinforcement learning

Important note: These slides are inspired by Emma Brunskill's brilliant course on reinforcement learning ($\rightarrow$ Link)

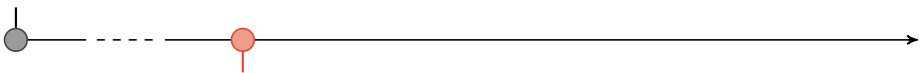
Timeline: Successes in Reinforcement Learning for Games

Backgammon (Tesauro 1995)



Timeline: Successes in Reinforcement Learning for Games

Backgammon (Tesauro 1995)



Atari games (Mnih et al. 2015)

Atari cartridges by the Digital Games Museum; usage according to CC-BY-2.0

Timeline: Successes in Reinforcement Learning for Games

Backgammon (Tesauro 1995)

Go (Silver et al. 2016)

Atari games (Mnih et al. 2015)

Atari cartridges by the Digital Games Museum; usage according to CC-BY-2.0

Timeline: Successes in Reinforcement Learning for Games

Backgammon (Tesauro 1995)

Go (Silver et al. 2016)

Atari games (Mnih et al. 2015)

DotA 2 (Pachocki et al. 2019)

Timeline: Successes in Reinforcement Learning for Games

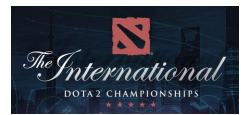
Backgammon (Tesauro 1995)

Go (Silver et al. 2016)

Starcraft 2 (Vinyals et al. 2019)

Atari games (Mnih et al. 2015)

DotA 2 (Pachocki et al. 2019)



Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent
- Let $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be a **reward** or **punishment** signal

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent
- Let $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be a **reward** or **punishment** signal
- Let $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ be the world's **state transition function**

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent
- Let $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be a **reward** or **punishment** signal
- Let $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ be the world's **state transition function**
- Then, the corresponding (deterministic) **reinforcement learning problem** is

$$\max_{\pi : \mathcal{X} \to \mathcal{A}} \quad \sum_t r\big(s_t, a_t\big)$$

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent
- Let $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be a **reward** or **punishment** signal
- Let $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ be the world's **state transition function**
- Then, the corresponding (deterministic) **reinforcement learning problem** is

$$\max_{\pi:\mathcal{X}\to\mathcal{A}} \quad \sum_t r\big(s_t, a_t\big)$$

where $a_t = \pi(s_t)$ and $s_{t+1} = f\big(s_t, a_t\big)$

Definition: Reinforcement Learning

- Let $\mathcal{S}$ be the set of possible world states
- Let $\mathcal{A}$ be the set of possible actions of our agent
- Let $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be a **reward** or **punishment** signal
- Let $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ be the world's **state transition function**
- Then, the corresponding (deterministic) **reinforcement learning problem** is

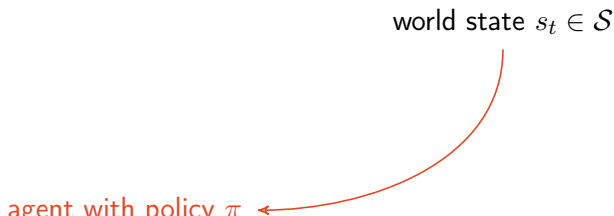
$$\max_{\pi:\mathcal{X}\to\mathcal{A}} \quad \sum_t r\big(s_t, a_t\big)$$

where $a_t = \pi(s_t)$ and $s_{t+1} = f\big(s_t, a_t\big)$; i.e. find the **policy** π of our agent that maximizes the overall reward

Reinforcement Learning Illustration

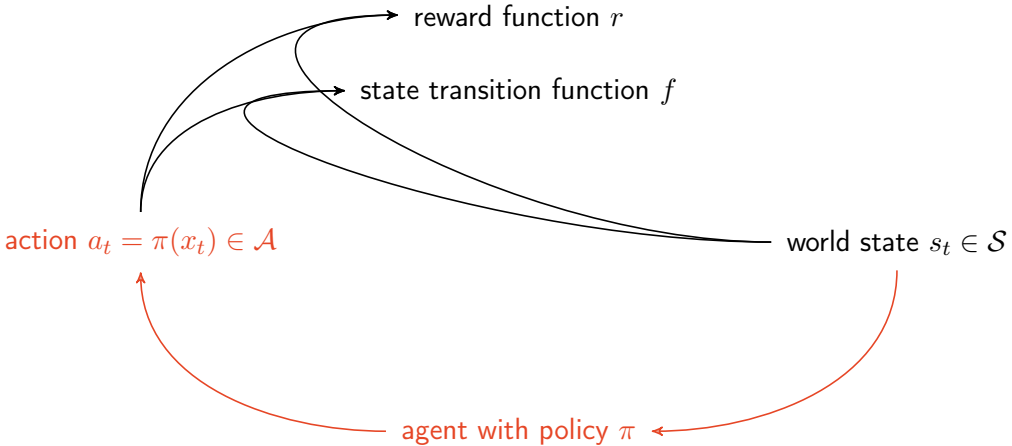
world state $s_t \in \mathcal{S}$

Reinforcement Learning Illustration

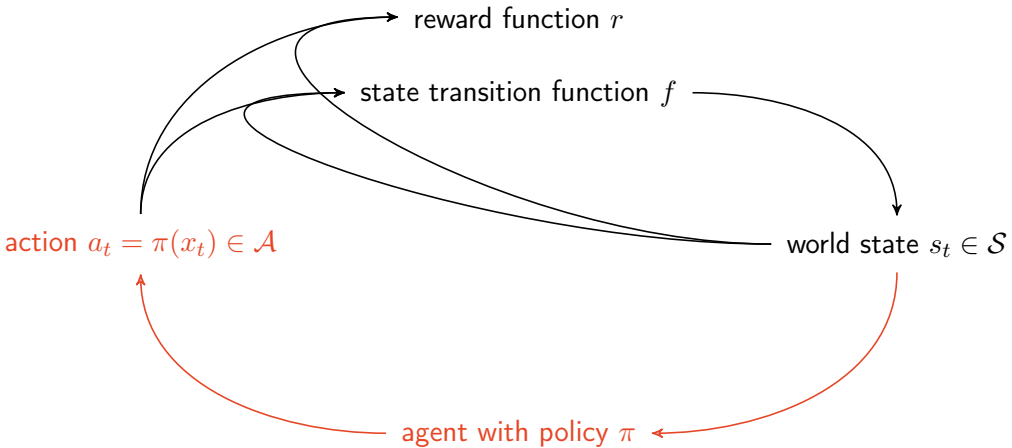


Reinforcement Learning Illustration

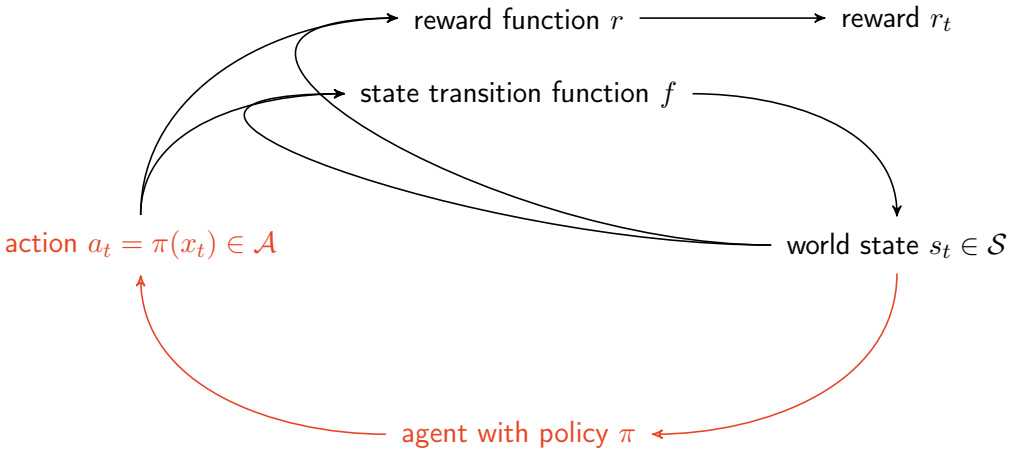
Reinforcement Learning Illustration



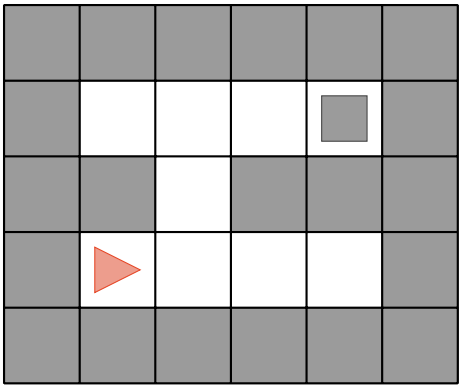
Reinforcement Learning Illustration



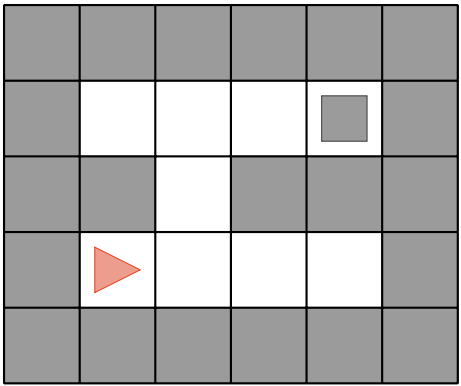
Reinforcement Learning Illustration

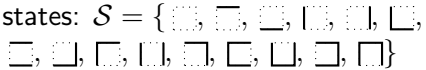


Gridworld Example

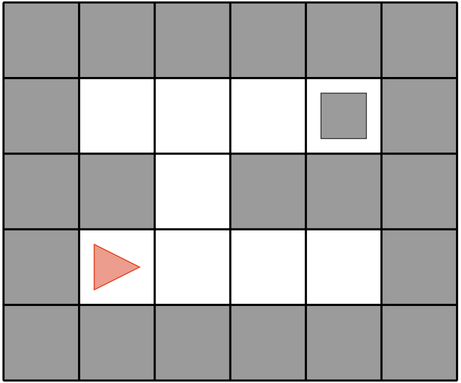


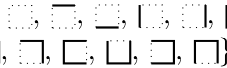
Gridworld Example



states: $\mathcal{S} = \{$  $\}$

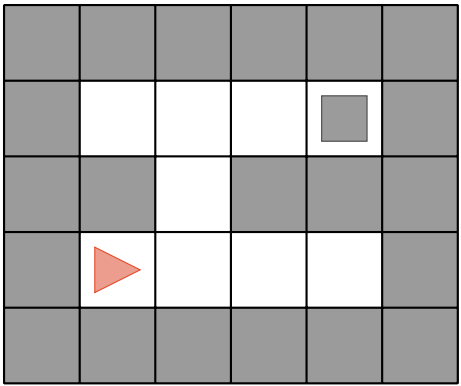
Gridworld Example



states: $\mathcal{S} = \{$  $\}$

actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

Gridworld Example

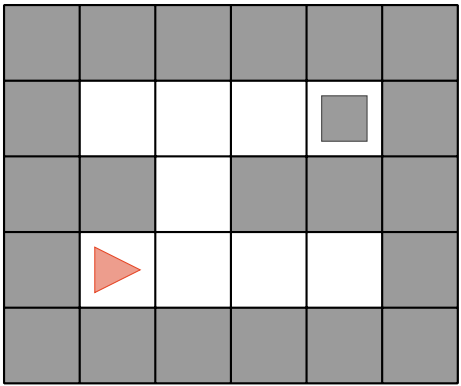


states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  ,  ,  $\}$

actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

Gridworld Example



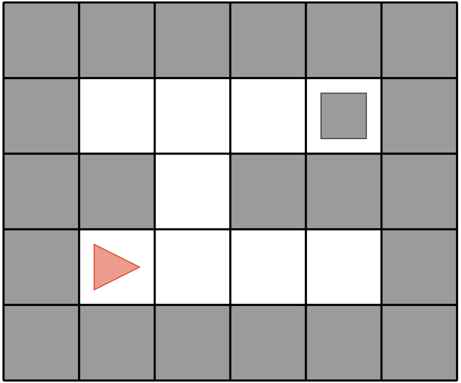
states: $\mathcal{S} = \{$, , , ,  , ,  ,  ,  ,  ,  ,  $\}$

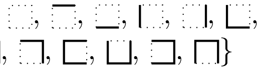
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

Gridworld Example



states: $\mathcal{S} = \{$  $\}$

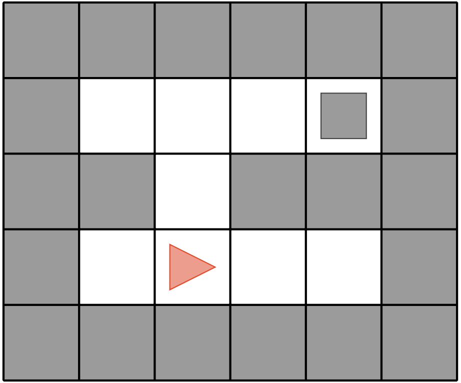
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

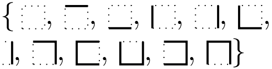
transition: see left; reward: +1 at goal

policy?

s_t	$a_t = \pi(s_t)$
	

Gridworld Example



states: $\mathcal{S} = \{$  $\}$

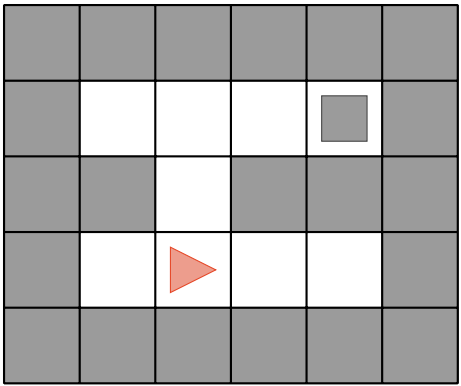
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: +1 at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$, , , ,  , ,  ,  ,  ,  ,  ,  , , , , , , $\}$

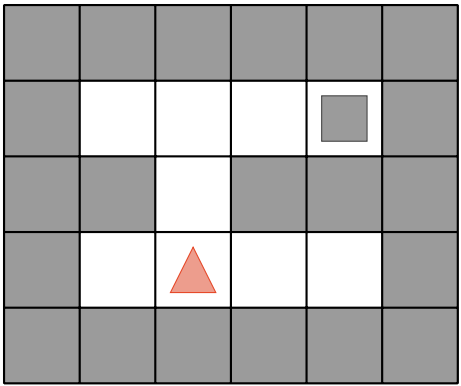
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$, , ,  ,  , ,  ,  ,  , ,  ,  , , , ,  , , $\}$

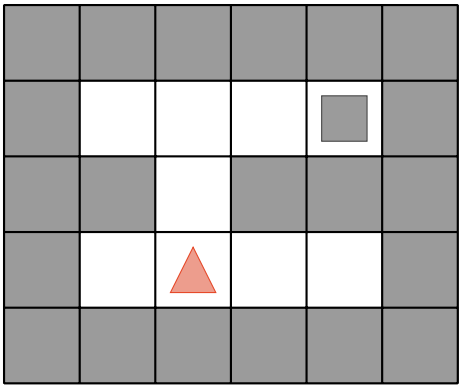
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$

Gridworld Example



states: $\mathcal{S} = \{$, , , ,  , ,  ,  ,  ,  ,  ,  $\}$

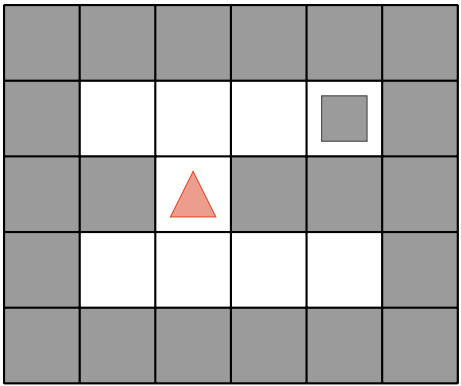
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

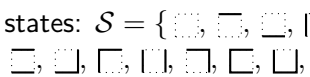
transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$

Gridworld Example



states: $\mathcal{S} = \{$  $\}$

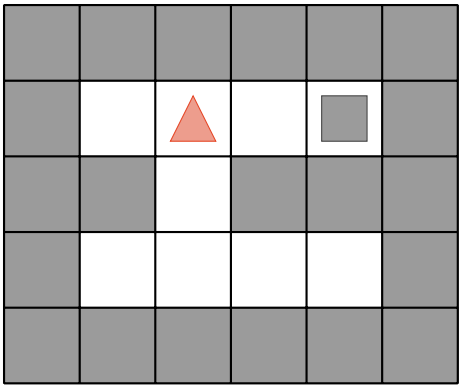
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  , ,  ,  ,  ,  ,  , ,  $\}$

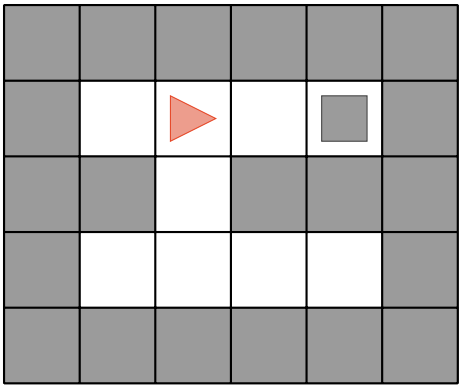
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	

Gridworld Example



states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  ,  ,  $\}$

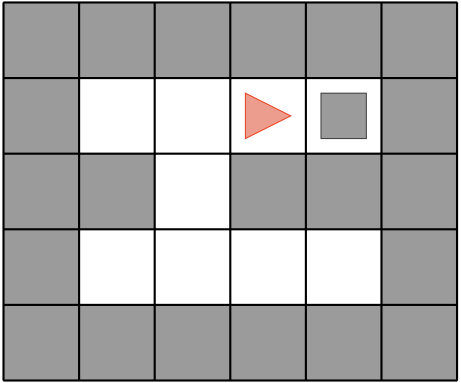
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

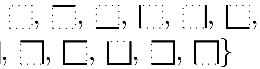
transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowleft$

Gridworld Example

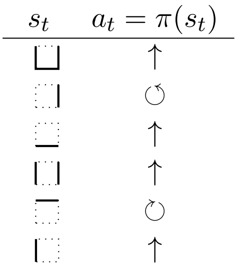


states: $\mathcal{S} = \{$  $\}$

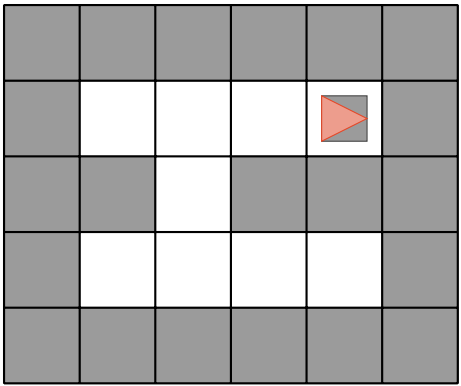
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: +1 at goal

policy?



Gridworld Example



states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  ,  ,  $\}$

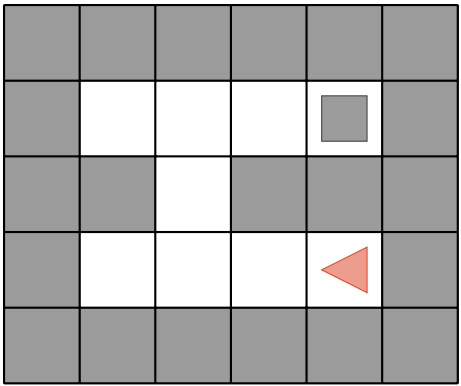
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowright$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  ,  ,  $\}$

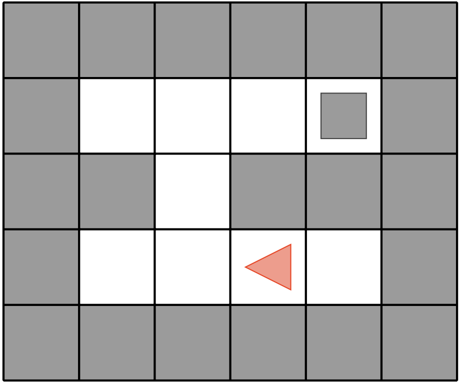
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

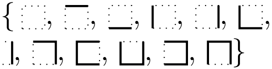
transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowright$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$  $\}$

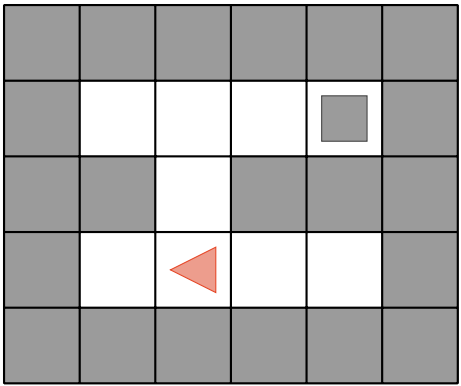
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: +1 at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowright$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$, , , , , ,  ,  ,  ,  ,  ,  $\}$

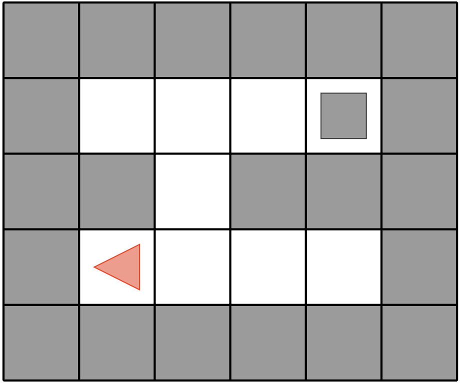
actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

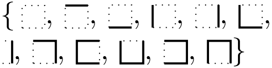
transition: see left; reward: $+1$ at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowright$
	$\uparrow$

Gridworld Example



states: $\mathcal{S} = \{$  $\}$

actions: $\mathcal{A} = \{\uparrow, \circlearrowleft, \circlearrowright\}$

transition: see left; reward: +1 at goal

policy?

s_t	$a_t = \pi(s_t)$
	$\uparrow$
	$\circlearrowleft$
	$\uparrow$
	$\uparrow$
	$\circlearrowright$
	$\uparrow$

Markov Decision Processes (MDPs)

- Idea: Consider an **indeterministic** world, i.e. state transition distribution $p(s_{t+1}|s_t, a_t)$ (policy and reward as before)

Markov Decision Processes (MDPs)

- Idea: Consider an **indeterministic** world, i.e. state transition distribution $p(s_{t+1}|s_t, a_t)$ (policy and reward as before)
- implies notion of **expected reward** of a policy π starting from state s, i.e. **state value** $V^\pi(s)$:

Markov Decision Processes (MDPs)

- Idea: Consider an **indeterministic** world, i.e. state transition distribution $p(s_{t+1}|s_t, a_t)$ (policy and reward as before)
- implies notion of **expected reward** of a policy π starting from state s, i.e. **state value** $V^\pi(s)$:

$$V^\pi(s) = r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s')$$

Markov Decision Processes (MDPs)

- Idea: Consider an **indeterministic** world, i.e. state transition distribution $p(s_{t+1}|s_t, a_t)$ (policy and reward as before)
- implies notion of **expected reward** of a policy π starting from state s, i.e. **state value** $V^\pi(s)$:

$$V^\pi(s) = r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s')$$

where $\gamma \in [0, 1]$ is a **discount** factor for future rewards

Markov Decision Processes (MDPs)

- Idea: Consider an **indeterministic** world, i.e. state transition distribution $p(s_{t+1}|s_t, a_t)$ (policy and reward as before)
- implies notion of **expected reward** of a policy π starting from state s, i.e. **state value** $V^\pi(s)$:

$$V^\pi(s) = r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s')$$

  where $\gamma \in [0, 1]$ is a **discount** factor for future rewards
- Maximization problem becomes:

$$\max_{\pi: \mathcal{S} \to \mathcal{A}} \quad V^\pi(s_0)$$

Policy Iteration Algorithm

1: **function** policy_iter(State set $\mathcal{S}$, action set $\mathcal{A}$, transition distribution p, discount factor γ, initial policy $\pi : \mathcal{S} \to \mathcal{A}$)
2: **while** π changes **do**

11: **end while**

13: **end function**

Policy Iteration Algorithm

1: **function** policy_iter(State set $\mathcal{S}$, action set $\mathcal{A}$, transition distribution p, discount factor γ, initial policy $\pi : \mathcal{S} \to \mathcal{A}$)
2: **while** π changes **do**
3: **while** V^π changes **do**
4: **for** $s \in \mathcal{S}$ **do**
5: $V^\pi(s) \leftarrow r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s')$.
6: **end for**
7: **end while**

11: **end while**

13: **end function**

Policy Iteration Algorithm

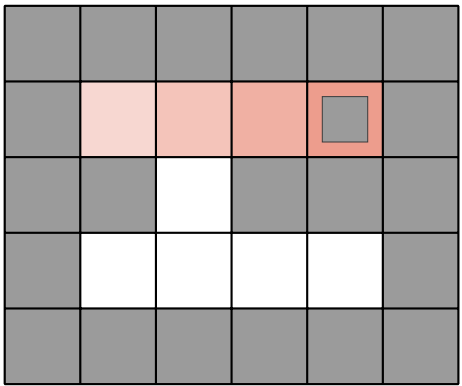
function policy_iter(State set $\mathcal{S}$, action set $\mathcal{A}$, transition distribution p, discount factor γ, initial policy $\pi : \mathcal{S} \to \mathcal{A}$)
    while π changes **do**
        while V^π changes **do**
            for $s \in \mathcal{S}$ **do**
                $V^\pi(s) \leftarrow r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s').$
            end for
        end while
        for $s \in \mathcal{S}$ **do**
            $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} r(s, a) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, a) \cdot V^\pi(s').$
        end for
    end while

end function

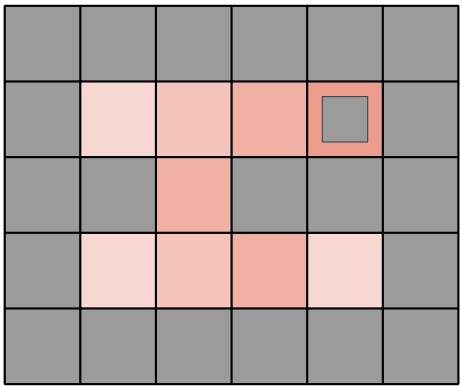
Policy Iteration Algorithm

function policy_iter(State set $\mathcal{S}$, action set $\mathcal{A}$, transition distribution p, discount factor γ, initial policy $\pi : \mathcal{S} \to \mathcal{A}$)
    while π changes **do**
        while V^π changes **do**
            for $s \in \mathcal{S}$ **do**
                $V^\pi(s) \leftarrow r\big(s, \pi(s)\big) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \cdot V^\pi(s')$.
            end for
        end while
        for $s \in \mathcal{S}$ **do**
            $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} r(s, a) + \gamma \cdot \sum_{s' \in \mathcal{S}} p(s'|s, a) \cdot V^\pi(s')$.
        end for
    end while
    return π.
end function

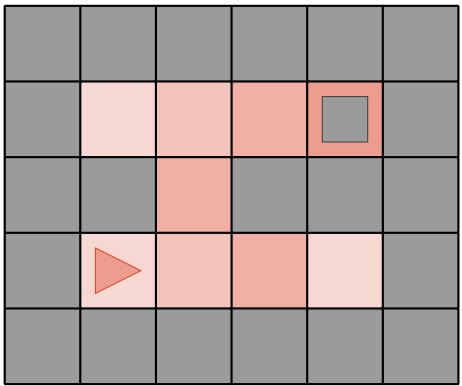
Gridworld Example



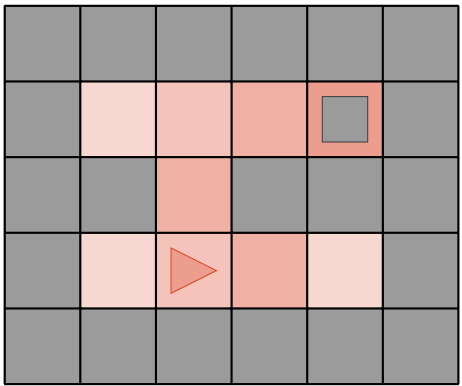
Gridworld Example



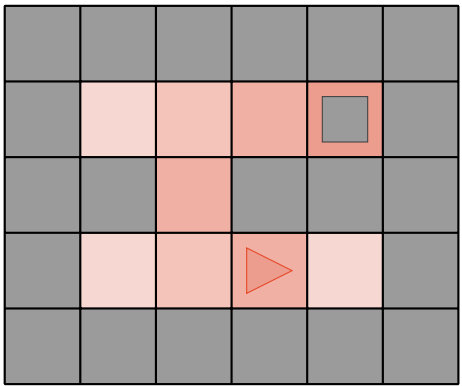
Gridworld Example



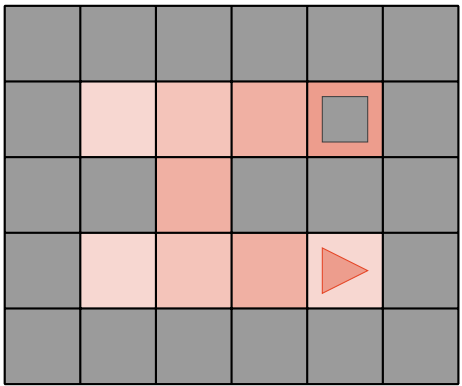
Gridworld Example



Gridworld Example



Gridworld Example



Monte Carlo Algorithm

1: **function** mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
2: **while** π changes **do**
3: $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.

15: **end while**

Monte Carlo Algorithm

1: **function** mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
2: **while** π changes **do**
3: $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.
4: **for** $i \in \{1,\ldots,N\}$ **do**
5: Simulate trajectory $(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)$ with current policy π.

11: **end for**

15: **end while**

Monte Carlo Algorithm

function mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
  while π changes **do**
    $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.
    for $i \in \{1,\ldots,N\}$ **do**
      Simulate trajectory $(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)$ with current policy π.
      for $t \in \{T,\ldots,1\}$ **do**
        $G_t \leftarrow r_t + \gamma \cdot G_{t+1} = \sum_{\tau=0}^{T-t} \gamma^\tau \cdot r_{t+\tau}$.
      end for
    end for
  end while

Monte Carlo Algorithm

function mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
  while π changes **do**
    $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.
    for $i \in \{1,\ldots,N\}$ **do**
      Simulate trajectory $(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)$ with current policy π.
      for $t \in \{T,\ldots,1\}$ **do**
        $G_t \leftarrow r_t + \gamma \cdot G_{t+1} = \sum_{\tau=0}^{T-t} \gamma^\tau \cdot r_{t+\tau}$.
        $N(s_t,a_t) \leftarrow N(s_t,a_t) + 1$.
        $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + G_t$.
      end for
    end for

  end while

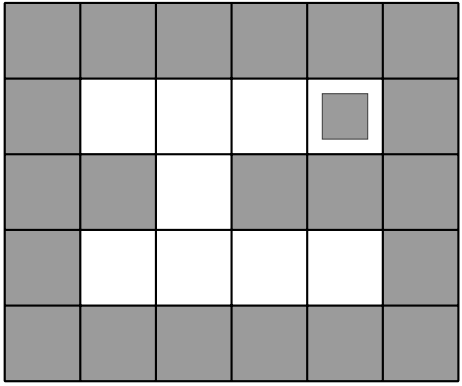
Monte Carlo Algorithm

function mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
  while π changes **do**
    $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.
    for $i \in \{1,\ldots,N\}$ **do**
      Simulate trajectory $(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)$ with current policy π.
      for $t \in \{T,\ldots,1\}$ **do**
        $G_t \leftarrow r_t + \gamma \cdot G_{t+1} = \sum_{\tau=0}^{T-t} \gamma^\tau \cdot r_{t+\tau}$.
        $N(s_t,a_t) \leftarrow N(s_t,a_t) + 1$.
        $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + G_t$.
      end for
    end for
    for $s \in \mathcal{S}$ **do**
      $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} Q(s,a)/N(s,a)$.
    end for
  end while

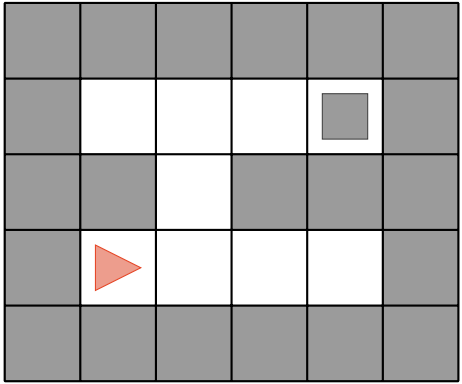
Monte Carlo Algorithm

function mc(State set $\mathcal{S}$, action set $\mathcal{A}$, simulator Sim, number of simulations N)
  while π changes **do**
    $N(s,a) \leftarrow 0$, $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$.
    for $i \in \{1,\ldots,N\}$ **do**
      Simulate trajectory $(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)$ with current policy π.
      for $t \in \{T,\ldots,1\}$ **do**
        $G_t \leftarrow r_t + \gamma \cdot G_{t+1} = \sum_{\tau=0}^{T-t} \gamma^\tau \cdot r_{t+\tau}$.
        $N(s_t,a_t) \leftarrow N(s_t,a_t) + 1$.
        $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + G_t$.
      end for
    end for
    for $s \in \mathcal{S}$ **do**
      $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} Q(s,a)/N(s,a)$.
    end for
  end while
  return π.

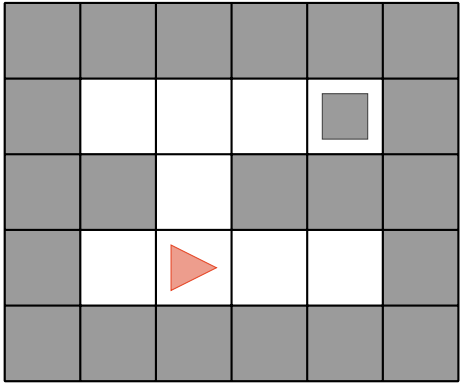
Gridworld Example



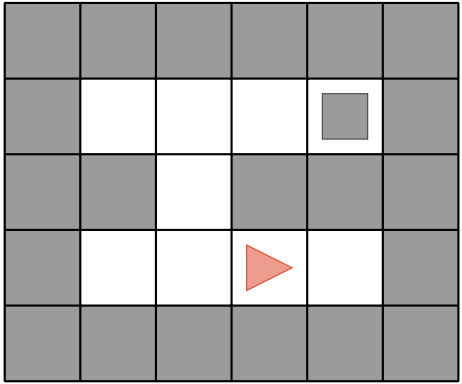
Gridworld Example



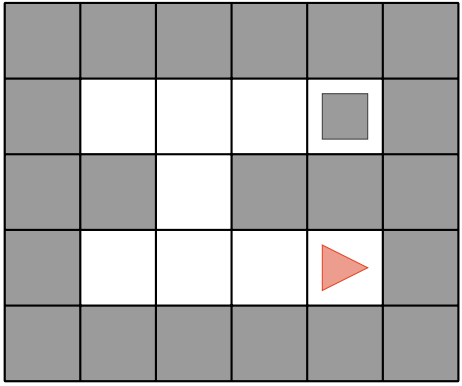
Gridworld Example



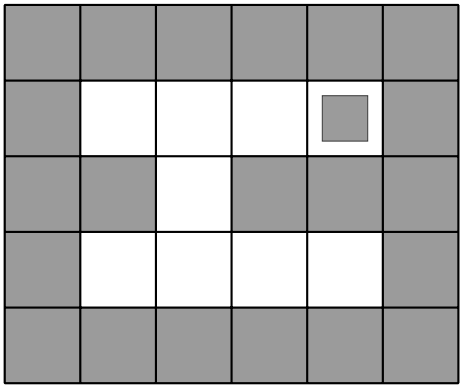
Gridworld Example



Gridworld Example

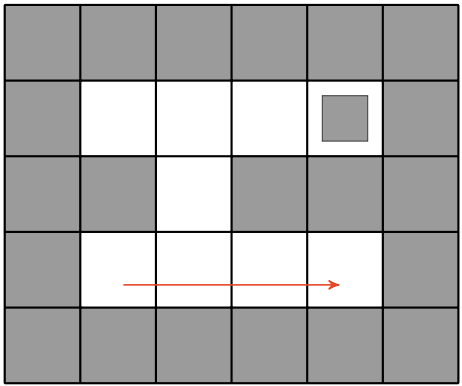


Gridworld Example



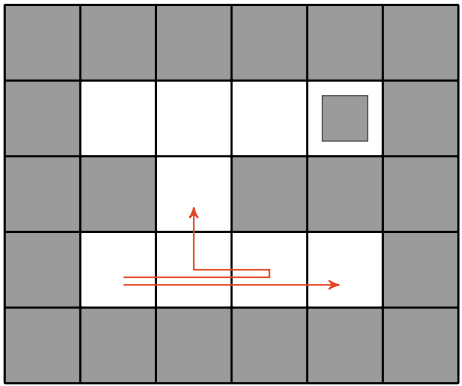
- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise

Gridworld Example



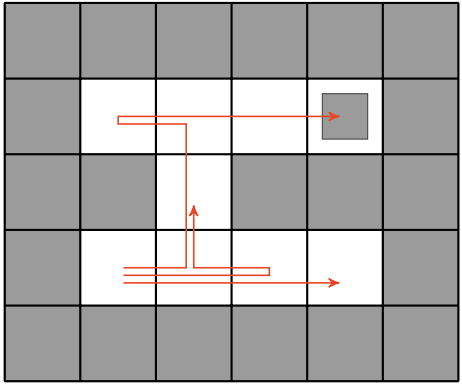
- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise

Gridworld Example



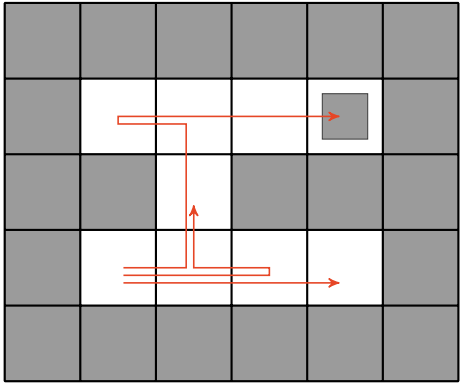
- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise

Gridworld Example

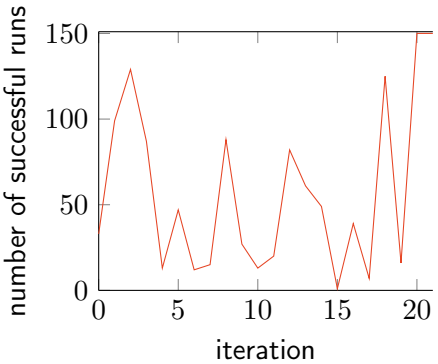


- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise

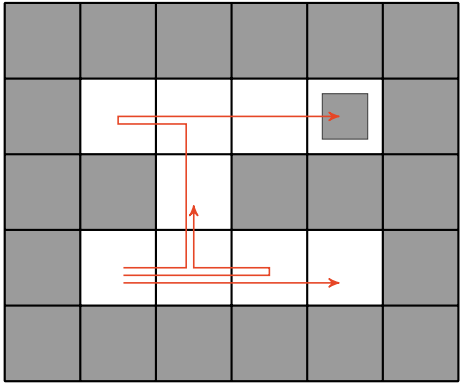
Gridworld Example



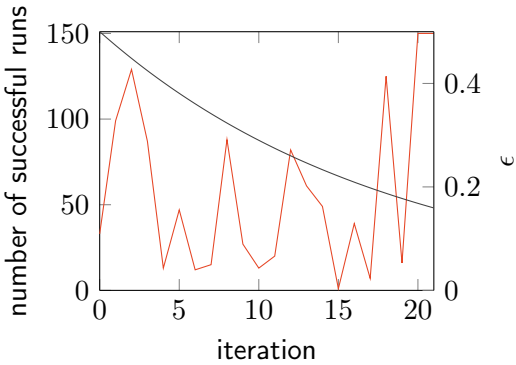
- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise



Gridworld Example



- ϵ-greedy policy: Do 'true' action $1 - \epsilon$ of the time and random action otherwise



Deep Q-Learning

- Motivation: What to do if $|\mathcal{S}|$ is very large? (million or higher)

Deep Q-Learning

- Motivation: What to do if $|\mathcal{S}|$ is very large? (million or higher)
- Idea: Don't tabulate $Q(s, a)$ but **learn** a regressor with parameters $\vec{\theta}$ (e.g. a deep neural net)

Deep Q-Learning

- Motivation: What to do if $|\mathcal{S}|$ is very large? (million or higher)
- Idea: Don't tabulate $Q(s,a)$ but **learn** a regressor with parameters $\vec{\theta}$ (e.g. a deep neural net)
- Algorithm: For every simulation time step (s_t, a_t, r_t) with next step s_{t+1} do a gradient step

$$\begin{aligned}\vec{\theta}_{t+1} &= \vec{\theta}_t - \eta \cdot \nabla_{\vec{\theta}} \|\hat{Q}(s_t,a_t) - Q(s_t,a_t|\vec{\theta})\|^2 \\ &= \vec{\theta}_t + 2\eta \cdot \left(\hat{Q}(s_t,a_t) - Q(s_t,a_t|\vec{\theta})\right) \cdot \nabla_{\vec{\theta}} Q(s_t,a_t|\vec{\theta})\end{aligned}$$

where $\hat{Q}(s_t,a_t) = r_t + \gamma \cdot \max_{a\in\mathcal{A}} Q(s_{t+1},a|\vec{\theta})$

Deep Q-Learning

- Motivation: What to do if $|\mathcal{S}|$ is very large? (million or higher)
- Idea: Don't tabulate $Q(s,a)$ but **learn** a regressor with parameters $\vec{\theta}$ (e.g. a deep neural net)
- Algorithm: For every simulation time step (s_t, a_t, r_t) with next step s_{t+1} do a gradient step

$$\begin{aligned}\vec{\theta}_{t+1} &= \vec{\theta}_t - \eta \cdot \nabla_{\vec{\theta}} \|\hat{Q}(s_t,a_t) - Q(s_t,a_t|\vec{\theta})\|^2 \\ &= \vec{\theta}_t + 2\eta \cdot \left(\hat{Q}(s_t,a_t) - Q(s_t,a_t|\vec{\theta})\right) \cdot \nabla_{\vec{\theta}} Q(s_t,a_t|\vec{\theta})\end{aligned}$$

  where $\hat{Q}(s_t,a_t) = r_t + \gamma \cdot \max_{a\in\mathcal{A}} Q(s_{t+1},a|\vec{\theta})$
- Tricks: 'replay' a buffer of the past multiple times; keep parameters for estimation of $\hat{Q}(s_t,a_t)$ fixed for a number of time steps

Atari Video Games

Imitation Learning

- Motivation: What if state space is large and simulations are expensive?

Imitation Learning

- Motivation: What if state space is large and simulations are expensive?
- Idea: Let an expert demonstrate a 'good enough' policy via a few sample trajectories and imitate it

Imitation Learning

- Motivation: What if state space is large and simulations are expensive?
- Idea: Let an expert demonstrate a 'good enough' policy via a few sample trajectories and imitate it
- Reduces policy learning to supervised regression

Imitation Learning

- Motivation: What if state space is large and simulations are expensive?
- Idea: Let an expert demonstrate a 'good enough' policy via a few sample trajectories and imitate it
- Reduces policy learning to supervised regression
- Note: Multiple subtleties involved, ongoing research (refer e.g. to ICML 2018 tutorial)

Policy Gradient algorithms

- Motivation: What if the action space is very large?

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s,\vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t\Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t,a_t) \cdot \pi(a_t|s_t,\vec{\theta})\Big)$$

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t \Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta}) \Big)$$

- Gradient:

$$\nabla_{\vec{\theta}} V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t \Big) \cdot \nabla_{\vec{\theta}} \Big(\prod_{t=1}^{T} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta}) \Big)$$

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t\Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta})\Big)$$

- Gradient:

$$\nabla_{\vec{\theta}} V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\prod \ldots\Big)\Big(\sum_{t=1}^{T} r_t\Big) \cdot \nabla_{\vec{\theta}} \log\Big[\prod_{t=1}^{T} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta})\Big]$$

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t\Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta})\Big)$$

- Gradient:

$$\nabla_{\vec{\theta}} V(\vec{\theta}) \approx \frac{1}{N} \sum_{i=1}^{N} \Big(\sum_{t=1}^{T_i} r_t^i\Big) \cdot \nabla_{\vec{\theta}} \log \Big[\prod_{t=1}^{T_i} p(s_{t+1}^i|s_t^i, a_t^i) \cdot \pi(a_t^i|s_t^i, \vec{\theta})\Big]$$

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s,\vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t\Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t,a_t) \cdot \pi(a_t|s_t,\vec{\theta})\Big)$$

- Gradient:

$$\nabla_{\vec{\theta}} V(\vec{\theta}) \approx \frac{1}{N} \sum_{i=1}^{N} \Big(\sum_{t=1}^{T_i} r_t^i\Big) \cdot \Big(\sum_{t=1}^{T_i} \nabla_{\vec{\theta}} \log\left[\pi(a_t^i|s_t^i,\vec{\theta})\right]\Big)$$

Policy Gradient algorithms

- Motivation: What if the action space is very large?
- Idea: Parametrize policy as $\pi(a|s, \vec{\theta})$ and learn it **directly** via gradient ascent
- Objective function: Expected reward over all possible trajectories

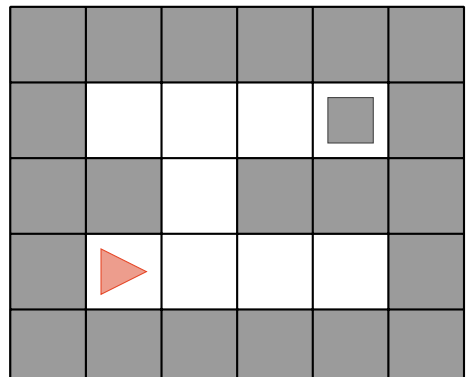
$$V(\vec{\theta}) = \sum_{(s_0,a_0,r_0),\ldots,(s_T,a_T,r_T)} \Big(\sum_{t=1}^{T} r_t\Big) \cdot \Big(\prod_{t=1}^{T-1} p(s_{t+1}|s_t, a_t) \cdot \pi(a_t|s_t, \vec{\theta})\Big)$$

- Gradient:

$$\nabla_{\vec{\theta}} V(\vec{\theta}) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T_i} \nabla_{\vec{\theta}} \log \left[\pi(a_t^i|s_t^i, \vec{\theta})\right] \cdot G_t^i$$

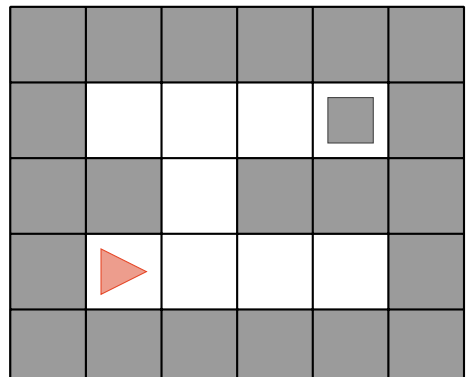
Gridworld Example

state	↑	↻	↺
	66.67%	16.67%	16.67%
	66.67%	16.67%	16.67%
	66.67%	16.67%	16.67%
	66.67%	16.67%	16.67%
	66.67%	16.67%	16.67%
	66.67%	16.67%	16.67%



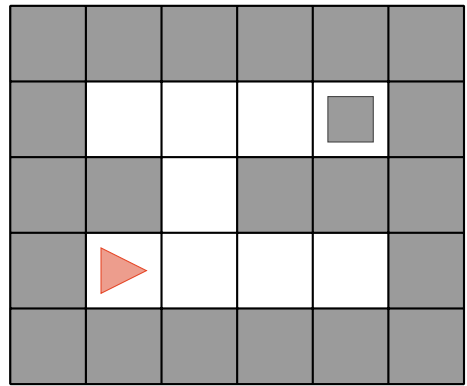
Gridworld Example

state	↑	↻	↺
	71.75%	14.42%	13.83%
	49.25%	14.62%	36.12%
	69.66%	14.20%	16.14%
	86.69%	7.02%	6.29%
	41.77%	44.98%	13.25%
	76.10%	12.89%	11.01%



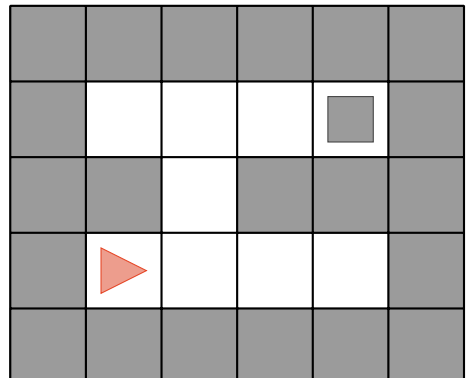
Gridworld Example

state	↑	↻	↺
	77.90%	12.35%	9.74%
	30.70%	8.97%	60.33%
	78.66%	10.89%	10.45%
	92.45%	4.56%	2.98%
	57.98%	39.00%	3.03%
	84.49%	8.99%	6.52%



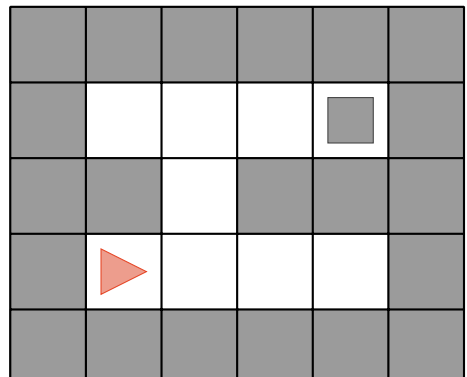
Gridworld Example

state	↑	↻	↺
	82.29%	10.54%	7.17%
	18.96%	5.39%	75.65%
	85.21%	7.55%	7.25%
	95.21%	2.88%	1.91%
	70.34%	29.41%	0.24%
	89.52%	5.38%	5.10%



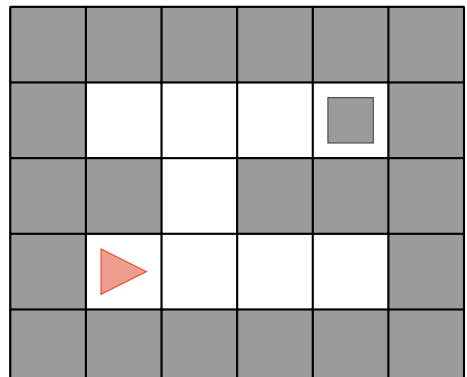
Gridworld Example

state	↑	↻	↺
	87.96%	7.16%	4.88%
	11.45%	3.54%	85.01%
	88.25%	6.63%	5.12%
	96.31%	2.40%	1.29%
	34.09%	65.88%	0.03%
	93.45%	3.36%	3.19%



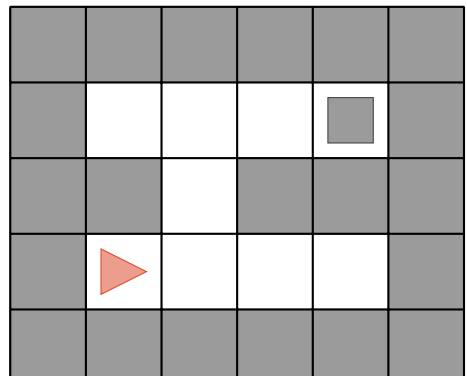
Gridworld Example

state	↑	↻	↺
	90.76%	5.45%	3.79%
	8.11%	2.35%	89.54%
	88.94%	7.41%	3.65%
	97.28%	1.77%	0.95%
	47.54%	52.46%	0.00%
	94.06%	2.48%	3.45%



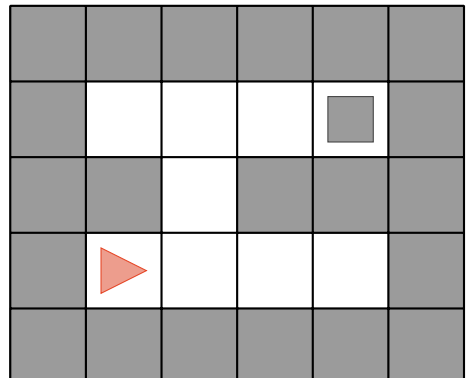
Gridworld Example

state	↑	↻	↺
	91.31%	5.62%	3.07%
	6.33%	1.83%	91.84%
	91.73%	5.54%	2.73%
	97.14%	2.10%	0.76%
	46.22%	53.78%	0.00%
	95.37%	1.94%	2.69%



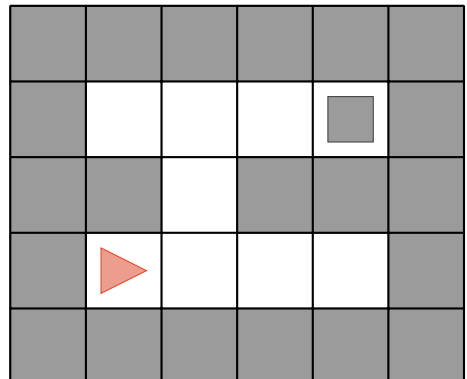
Gridworld Example

state	↑	↻	↺
	92.91%	4.59%	2.50%
	5.28%	1.51%	93.21%
	93.43%	4.40%	2.17%
	97.74%	1.66%	0.60%
	42.06%	57.94%	0.00%
	96.03%	1.57%	2.39%



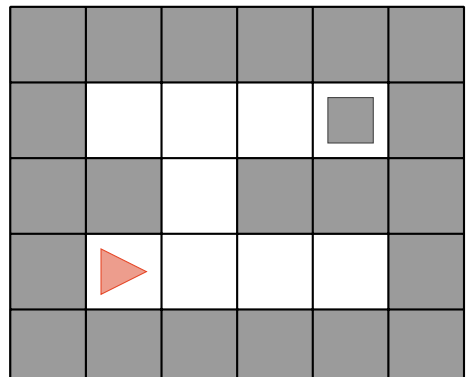
Gridworld Example

state	↑	↻	↺
	93.72%	4.15%	2.12%
	4.36%	1.25%	94.39%
	93.58%	4.65%	1.78%
	97.47%	2.04%	0.50%
	17.41%	82.59%	0.00%
	95.82%	1.32%	2.86%



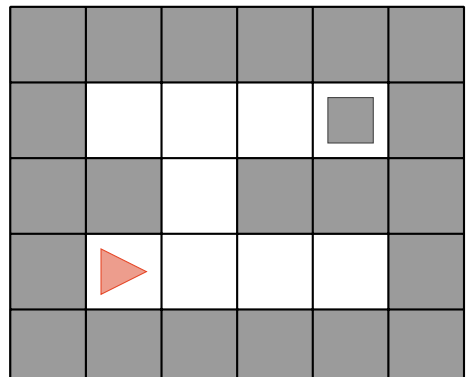
Gridworld Example

state	↑	↻	↺
	94.33%	3.84%	1.83%
	3.78%	1.08%	95.14%
	94.64%	3.87%	1.48%
	96.26%	3.34%	0.40%
	20.57%	79.43%	0.00%
	96.48%	1.11%	2.41%



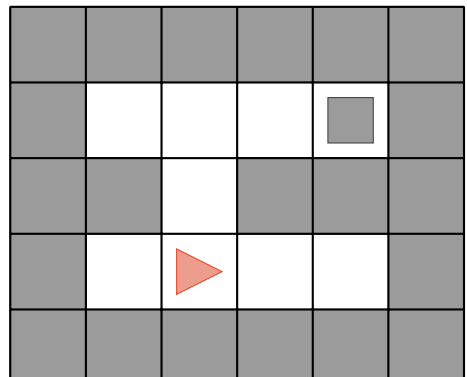
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



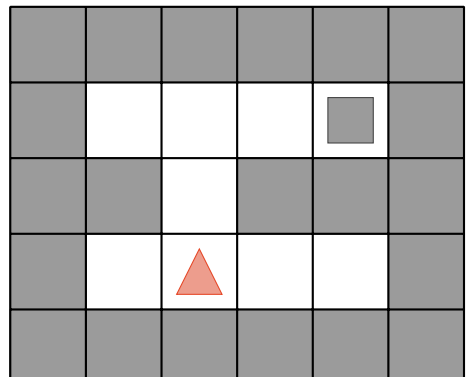
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



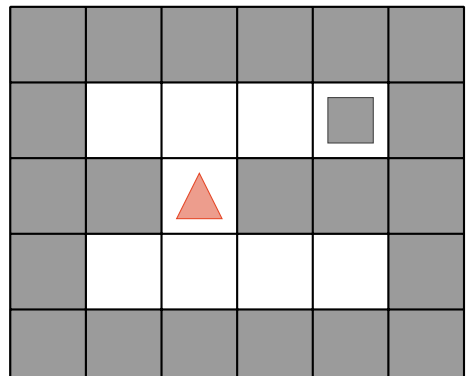
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



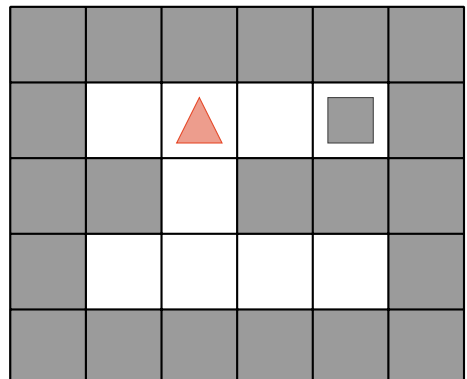
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



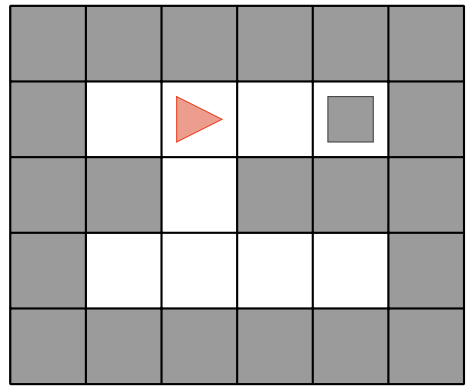
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



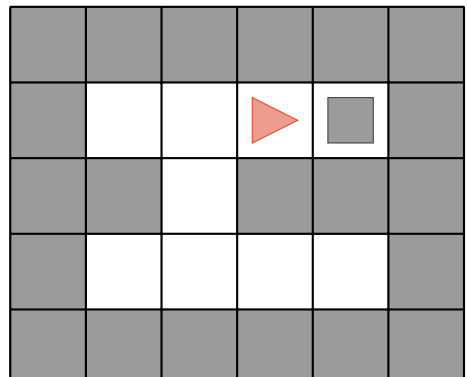
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



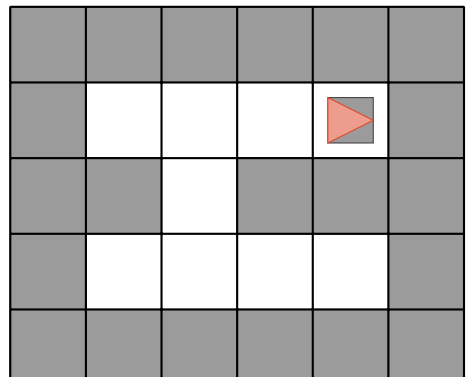
Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%



Gridworld Example

state	↑	↻	↺
	95.08%	3.33%	1.59%
	3.25%	0.93%	95.82%
	94.52%	4.16%	1.32%
	96.60%	3.12%	0.29%
	13.86%	86.14%	0.00%
	96.90%	1.06%	2.05%

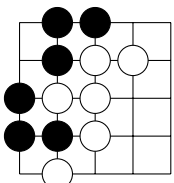


Practical Examples

Practical Example: AlphaGo

(Silver et al. 2016)

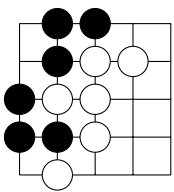
board state s

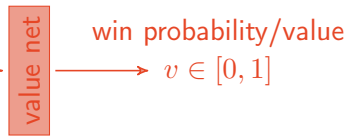


Practical Example: AlphaGo

(Silver et al. 2016)

board state s

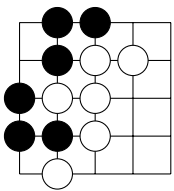




Practical Example: AlphaGo

(Silver et al. 2016)

board state s

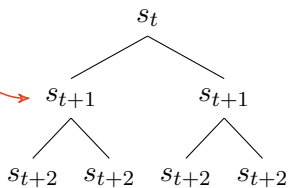


value net

win probability/value

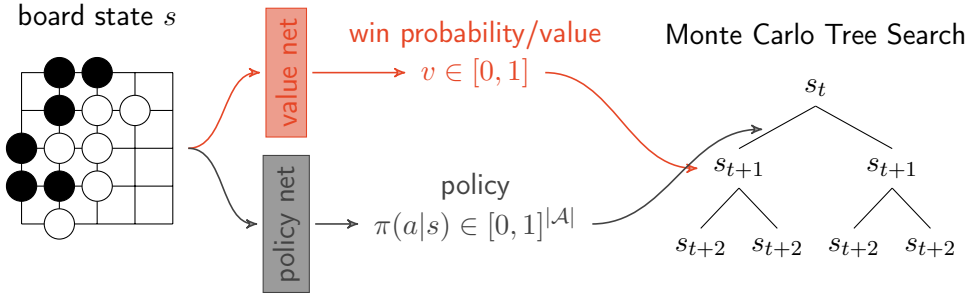
$v \in [0, 1]$

Monte Carlo Tree Search



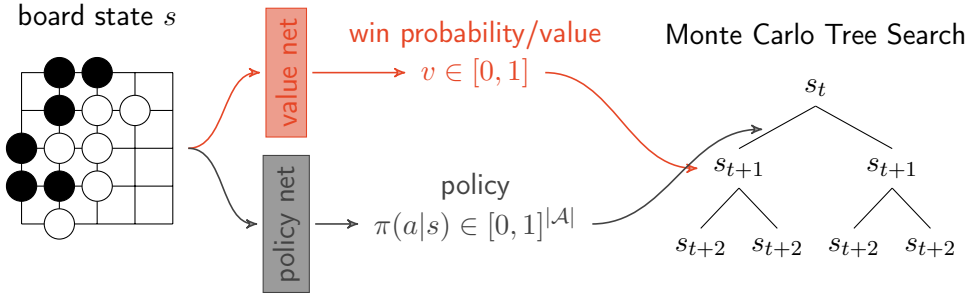
Practical Example: AlphaGo

(Silver et al. 2016)



Practical Example: AlphaGo

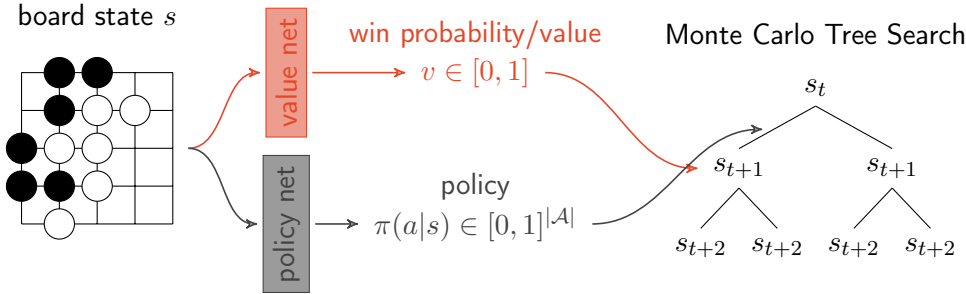
(Silver et al. 2016)



- Train value net to predict outcome of past games

Practical Example: AlphaGo

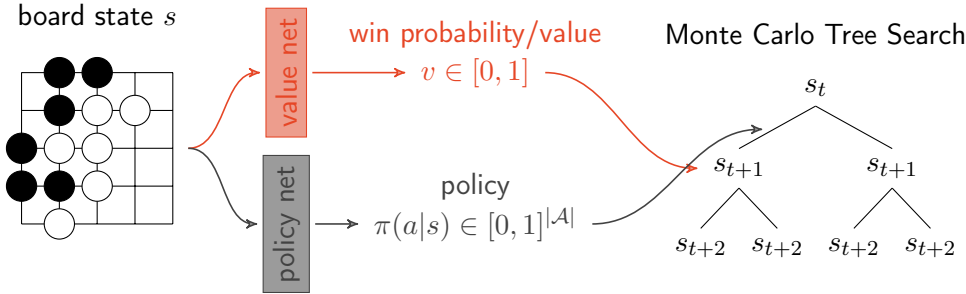
(Silver et al. 2016)



- Train value net to predict outcome of past games
- Train policy net to predict moves based on past games

Practical Example: AlphaGo

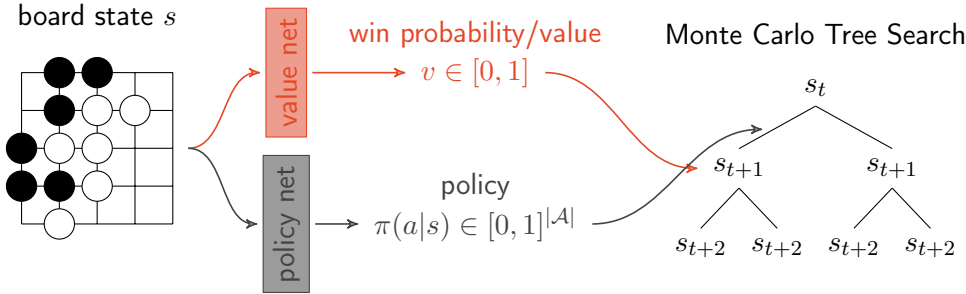
(Silver et al. 2016)



- Train value net to predict outcome of past games
- Train policy net to predict moves based on past games
- Use both nets to guide Monte Carlo Tree Search (MCTS)

Practical Example: AlphaGo

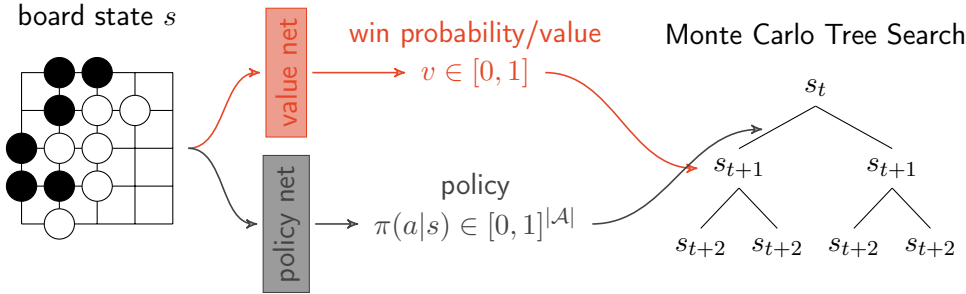
(Silver et al. 2016)



- Train value net to predict outcome of past games
- Train policy net to predict moves based on past games
- Use both nets to guide Monte Carlo Tree Search (MCTS); i.e. try moves with highest possible future value that are still probable

Practical Example: AlphaGo

(Silver et al. 2016)



- Train value net to predict outcome of past games
- Train policy net to predict moves based on past games
- Use both nets to guide Monte Carlo Tree Search (MCTS); i.e. try moves with highest possible future value that are still probable
- Note: Either value net or policy net would suffice to play the game, but MCTS yields stronger moves

Practical Example: AlphaStar (Vinyals et al. 2019)

- Grandmaster-level AI for 1vs1 StarCraft play

Practical Example: AlphaStar (Vinyals et al. 2019)

- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements

Practical Example: AlphaStar (Vinyals et al. 2019)

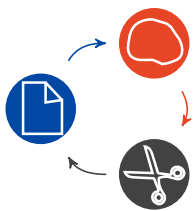
- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning

Practical Example: AlphaStar (Vinyals et al. 2019)

- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle

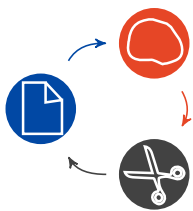
Practical Example: AlphaStar (Vinyals et al. 2019)

- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle



Practical Example: AlphaStar (Vinyals et al. 2019)

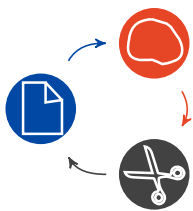
- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle



π			
agent 1	0.6	0.2	0.2
agent 2	0.2	0.6	0.2

Practical Example: AlphaStar (Vinyals et al. 2019)

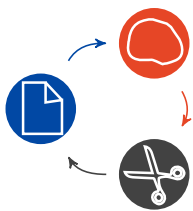
- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle



π			
agent 1	0.2	0.2	0.6
agent 2	0.2	0.6	0.2

Practical Example: AlphaStar (Vinyals et al. 2019)

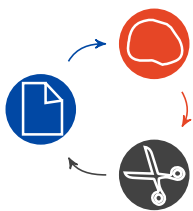
- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle



π			
agent 1	0.2	0.2	0.6
agent 2	0.6	0.2	0.2

Practical Example: AlphaStar (Vinyals et al. 2019)

- Grandmaster-level AI for 1vs1 StarCraft play
- System architecture too complex to cover here; key points: much more memory required than in AlphaGo, very different action space, real-time requirements
- Training starts off with imitation learning and then improves via self-play and reinforcement learning
- Key issue in training: Rock-paper-scissors principle



π			
agent	0.33	0.33	0.33
exploiter 1	1.0	0.0	0.0
exploiter 2	0.0	1.0	0.0
exploiter 3	0.0	0.0	1.0

Ethics in ML

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)
- limit human subconscious bias

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)
- limit human subconscious bias
- make decision processes transparent and explicit

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)
- limit human subconscious bias
- make decision processes transparent and explicit
- reveal subtle patterns of ethical violations (e.g. organized crime, discrimination)

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)
- limit human subconscious bias
- make decision processes transparent and explicit
- reveal subtle patterns of ethical violations (e.g. organized crime, discrimination)
- inform our understanding of impacts of our actions

The challenge of ethical mathematical models

Models can have many **ethically positive** impacts on society:

- predict and prevent dangerous situations (e.g. climate change)
- limit human subconscious bias
- make decision processes transparent and explicit
- reveal subtle patterns of ethical violations (e.g. organized crime, discrimination)
- inform our understanding of impacts of our actions

. . . but they can also be dangerous.

Case Study: 2007-2009 financial crisis (Thakor 2015)

- Financial institutions used risk scores to guide investment decisions

Case Study: 2007-2009 financial crisis (Thakor 2015)

- Financial institutions used risk scores to guide investment decisions
- Models to generate such scores were systematically flawed and underestimated risk (e.g. by independence assumptions on defaults)

Case Study: 2007-2009 financial crisis (Thakor 2015)

- Financial institutions used risk scores to guide investment decisions
- Models to generate such scores were systematically flawed and underestimated risk (e.g. by independence assumptions on defaults)
- Flaws were not apparent, though, because system appeared stable as long as general trust was high

Case Study: 2007-2009 financial crisis (Thakor 2015)

- Financial institutions used risk scores to guide investment decisions
- Models to generate such scores were systematically flawed and underestimated risk (e.g. by independence assumptions on defaults)
- Flaws were not apparent, though, because system appeared stable as long as general trust was high
- As soon as trust eroded, system immediately broke down

Case Study: 2007-2009 financial crisis (Thakor 2015)

- Financial institutions used risk scores to guide investment decisions
- Models to generate such scores were systematically flawed and underestimated risk (e.g. by independence assumptions on defaults)
- Flaws were not apparent, though, because system appeared stable as long as general trust was high
- As soon as trust eroded, system immediately broke down

$\Rightarrow$ (ML) models contributed to 2007-2009 financial crisis

Case Study: COMPAS (Angwin et al. 2016)

- System to predict **risk of recidivism** to decide on bail or not

Case Study: COMPAS (Angwin et al. 2016)

- System to predict **risk of recidivism** to decide on bail or not
- ProPublica research revealed that Black people received much higher risk scores, even if they did not reoffend (false positives)

Case Study: COMPAS (Angwin et al. 2016)

- System to predict **risk of recidivism** to decide on bail or not
- ProPublica research revealed that Black people received much higher risk scores, even if they did not reoffend (false positives)
- Conversely, white people received lower risk scores, even if they did reoffend (false negatives)

Case Study: COMPAS (Angwin et al. 2016)

- System to predict **risk of recidivism** to decide on bail or not
- ProPublica research revealed that Black people received much higher risk scores, even if they did not reoffend (false positives)
- Conversely, white people received lower risk scores, even if they did reoffend (false negatives)

$\Rightarrow$ Grounds to accuse system of racial bias

Case Study: Predictive Policing (Ensign et al. 2018)

- Many US cities employ models to predict the location of future crimes based on past crime records

Case Study: Predictive Policing (Ensign et al. 2018)

- Many US cities employ models to predict the location of future crimes based on past crime records
- This heightens police attention on predicted locations

Case Study: Predictive Policing (Ensign et al. 2018)

- Many US cities employ models to predict the location of future crimes based on past crime records
- This heightens police attention on predicted locations
- Could thus lead to higher number of reports and even more predictions

Case Study: Predictive Policing (Ensign et al. 2018)

- Many US cities employ models to predict the location of future crimes based on past crime records
- This heightens police attention on predicted locations
- Could thus lead to higher number of reports and even more predictions

$\Rightarrow$ Police attention concentrated on few areas

Case Study: Amazon recruiting tool (Dastin 2018)

- Amazon trained a model to predict job success probability from resumes for software engineers

Case Study: Amazon recruiting tool (Dastin 2018)

- Amazon trained a model to predict job success probability from resumes for software engineers
- Due to patterns in past data, the model predicted higher job success for men compared to women

Case Study: Amazon recruiting tool (Dastin 2018)

- Amazon trained a model to predict job success probability from resumes for software engineers
- Due to patterns in past data, the model predicted higher job success for men compared to women
- Amazon scrapped the model before applying it

Case Study: Austrias Unemployment Agency (Dornis 2018)

- Unemployment agency predicts hiring probability from features including gender and age

Case Study: Austrias Unemployment Agency (Dornis 2018)

- Unemployment agency predicts hiring probability from features including gender and age
- 'lowest class' of predicted probability could receive less support

Case Study: Austrias Unemployment Agency (Dornis 2018)

- Unemployment agency predicts hiring probability from features including gender and age
- 'lowest class' of predicted probability could receive less support
- Older women receive lower scores

Case Study: Austrias Unemployment Agency (Dornis 2018)

- Unemployment agency predicts hiring probability from features including gender and age
- 'lowest class' of predicted probability could receive less support
- Older women receive lower scores

$\Rightarrow$ Decreases hiring probability further

Case Study: Face Recognition (Lohr 2018)

- Face recognition software has been systematically shown to misidentify women more than men and Black people more than white people

Case Study: Face Recognition (Lohr 2018)

- Face recognition software has been systematically shown to misidentify women more than men and Black people more than white people
- Particularly controversial: 'Gorilla' label by a Google software in 2015 (Hern 2018)

Case Study: Face Recognition (Lohr 2018)

- Face recognition software has been systematically shown to misidentify women more than men and Black people more than white people
- Particularly controversial: 'Gorilla' label by a Google software in 2015 (Hern 2018)
- Concerning potential applications: Predict IQ from images, criminality, identifying terrorists for drone strikes

Let's try to structure

effects/harms

Let's try to structure

effects/harms group unfairness

Let's try to structure

effects/harms	group unfairness	individual unfairness

Let's try to structure

effects/harms group unfairness individual unfairness stereotypes

Let's try to structure

causes/mechanisms

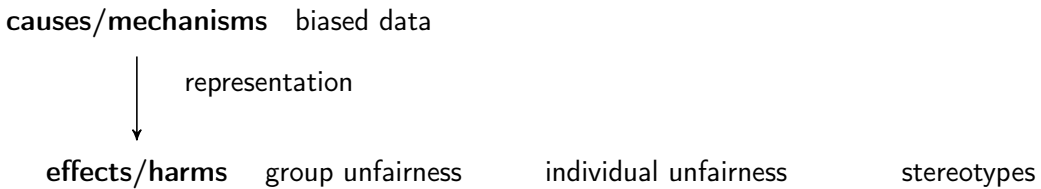
effects/harms group unfairness individual unfairness stereotypes

Let's try to structure

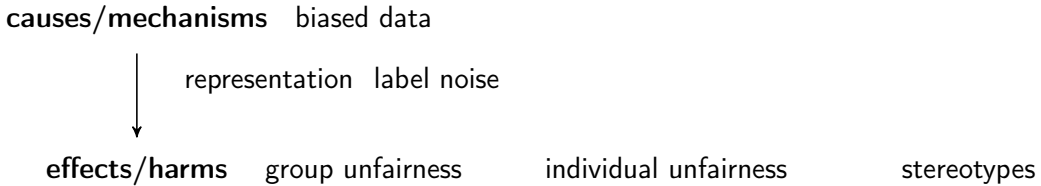
causes/mechanisms biased data

effects/harms group unfairness individual unfairness stereotypes

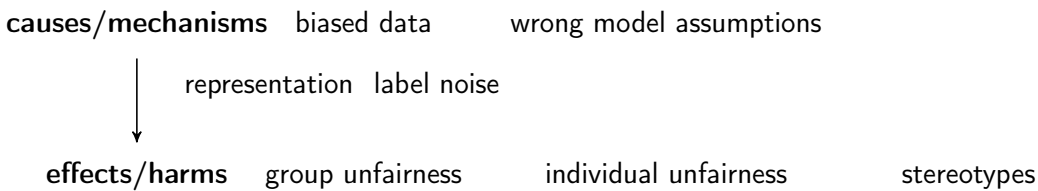
Let's try to structure



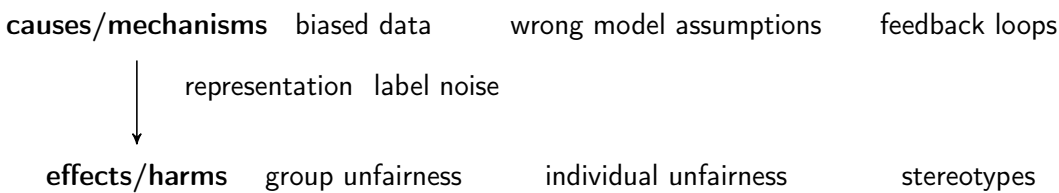
Let's try to structure



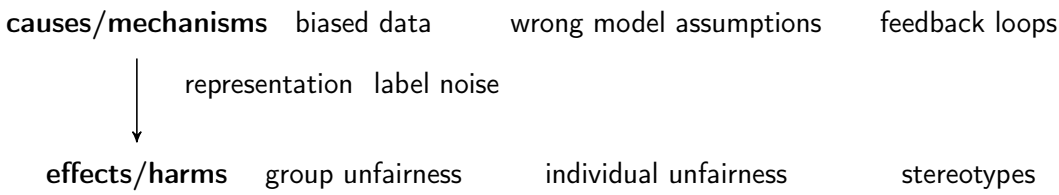
Let's try to structure



Let's try to structure

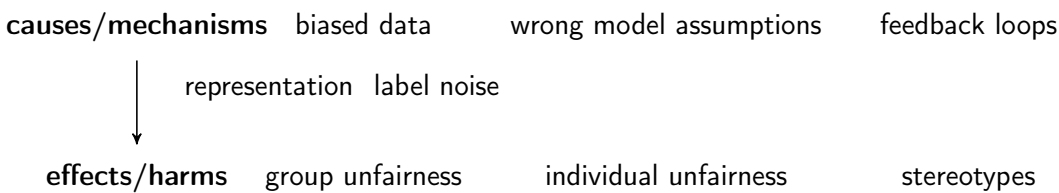


Let's try to structure



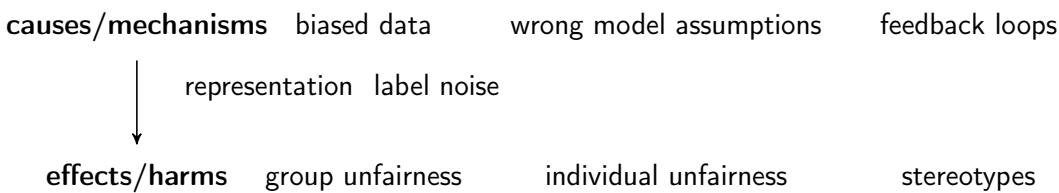
- This picture is incomplete!

Let's try to structure



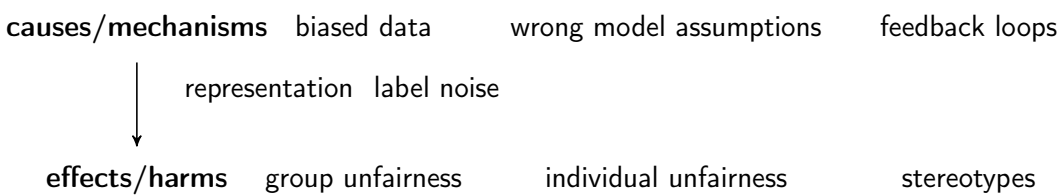
- This picture is incomplete! (e.g. harms: privacy, information asymmetry; causes: bugs, blind trust)

Let's try to structure



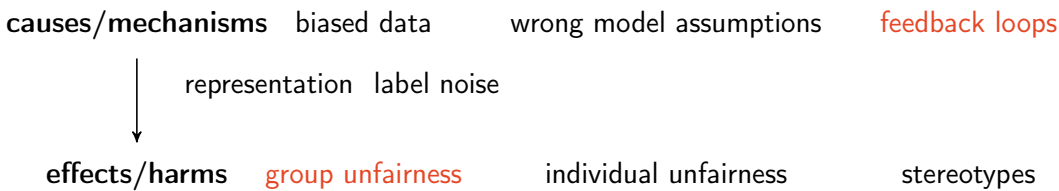
- This picture is incomplete! (e.g. harms: privacy, information asymmetry; causes: bugs, blind trust)
- Often, multiple causes interact

Let's try to structure



- This picture is incomplete! (e.g. harms: privacy, information asymmetry; causes: bugs, blind trust)
- Often, multiple causes interact
- Harms depend on **socio-technical embedding** (i.e. societal/institutional influence of system predictions)

Let's try to structure



- This picture is incomplete! (e.g. harms: privacy, information asymmetry; causes: bugs, blind trust)
- Often, multiple causes interact
- Harms depend on **socio-technical embedding** (i.e. societal/institutional influence of system predictions)

Definition: Our scenario

- Task: Design a binary classifier f

Definition: Our scenario

- Task: Design a binary classifier f that distributes a **desirable (limited) resource** (jobs, loans, good risk scores, getting free on bail, ...)

Definition: Our scenario

- Task: Design a binary classifier f that distributes a **desirable (limited) resource** (jobs, loans, good risk scores, getting free on bail, ...)
- Input $x \in \mathcal{X}$: Applicants for the resource (described by features)

Definition: Our scenario

- Task: Design a binary classifier f that distributes a **desirable (limited) resource** (jobs, loans, good risk scores, getting free on bail, ...)
- Input $x \in \mathcal{X}$: Applicants for the resource (described by features)
- Applicants also have **group labels** $\vec{c} \in \{0,1\}^C$ (i.e. being male, being female, being Black, being older than 60, ...)

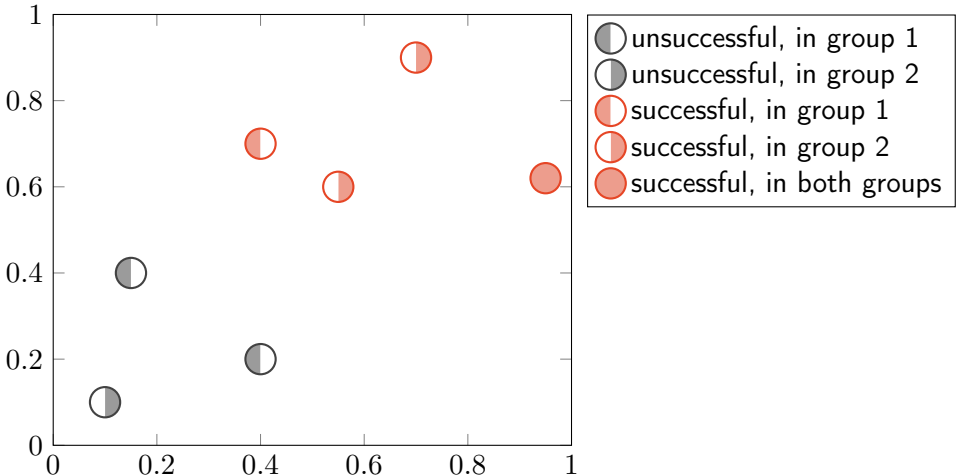
Definition: Our scenario

- Task: Design a binary classifier f that distributes a **desirable (limited) resource** (jobs, loans, good risk scores, getting free on bail, ...)
- Input $x \in \mathcal{X}$: Applicants for the resource (described by features)
- Applicants also have **group labels** $\vec{c} \in \{0,1\}^C$ (i.e. being male, being female, being Black, being older than 60, ...)
- Output $f(x) \in [0,1]$: 'success probability', where higher values mean higher probability of 'deserving' the resource

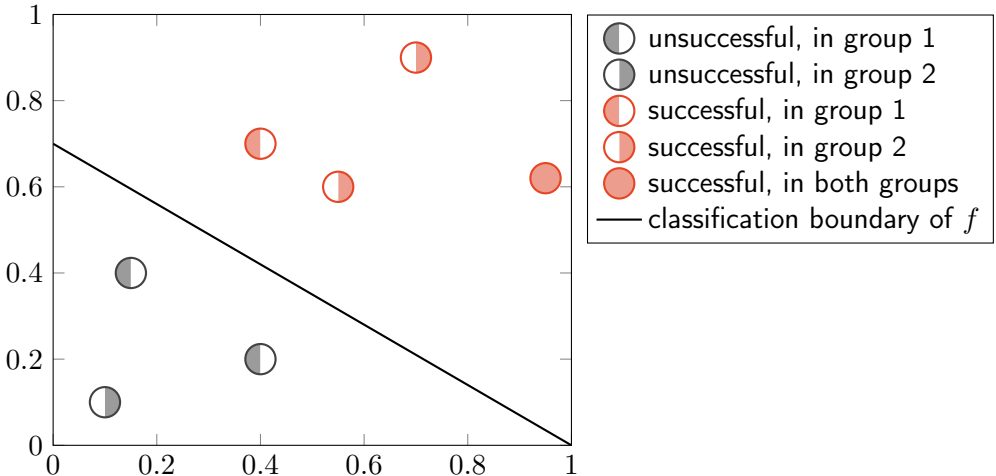
Definition: Our scenario

- Task: Design a binary classifier f that distributes a **desirable (limited) resource** (jobs, loans, good risk scores, getting free on bail, ...)
- Input $x \in \mathcal{X}$: Applicants for the resource (described by features)
- Applicants also have **group labels** $\vec{c} \in \{0,1\}^C$ (i.e. being male, being female, being Black, being older than 60, ...)
- Output $f(x) \in [0,1]$: 'success probability', where higher values mean higher probability of 'deserving' the resource
- Training data: $\mathcal{D} = \{(x_1, \vec{c}_1, y_1), \ldots, (x_N, \vec{c}_N, y_N)\}$ of past applicants and success labels $y_i \in \{0,1\}$

Scenario illustration



Scenario illustration



Fairness Definitions

- Threshold fairness: All individuals should have the same classifier (with low error) (Corbett-Davies and Goel 2018)

Fairness Definitions

- Threshold fairness: All individuals should have the same classifier (with low error) (Corbett-Davies and Goel 2018)
- Individual fairness: Similar individuals should receive similar predictions (Dwork et al. 2012)

Fairness Definitions

- Threshold fairness: All individuals should have the same classifier (with low error) (Corbett-Davies and Goel 2018)
- Individual fairness: Similar individuals should receive similar predictions (Dwork et al. 2012)
- Process fairness: Group membership should not (directly, causally) influence prediction (Grgić-Hlača et al. 2016; Kilbertus et al. 2017)

Fairness Definitions

- Threshold fairness: All individuals should have the same classifier (with low error) (Corbett-Davies and Goel 2018)
- Individual fairness: Similar individuals should receive similar predictions (Dwork et al. 2012)
- Process fairness: Group membership should not (directly, causally) influence prediction (Grgić-Hlača et al. 2016; Kilbertus et al. 2017)
- Demographic parity: Predictions should be statistically independent of group membership (i.e. all groups get the same mean prediction; Dwork et al. 2012)

Fairness Definitions

- Threshold fairness: All individuals should have the same classifier (with low error) (Corbett-Davies and Goel 2018)
- Individual fairness: Similar individuals should receive similar predictions (Dwork et al. 2012)
- Process fairness: Group membership should not (directly, causally) influence prediction (Grgić-Hlača et al. 2016; Kilbertus et al. 2017)
- Demographic parity: Predictions should be statistically independent of group membership (i.e. all groups get the same mean prediction; Dwork et al. 2012)
- Equalized odds: Predictions should be statistically independent of group membership **conditioned on label** (i.e. all groups have same classification error; Hardt, Price, and Nati Srebro 2016)

Illustration: Demographic parity vs Equalized Odds

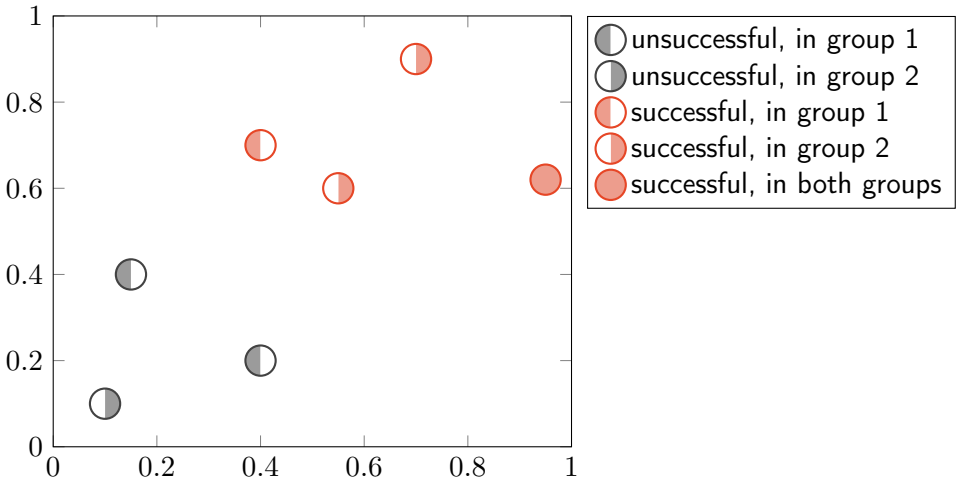


Illustration: Demographic parity vs Equalized Odds

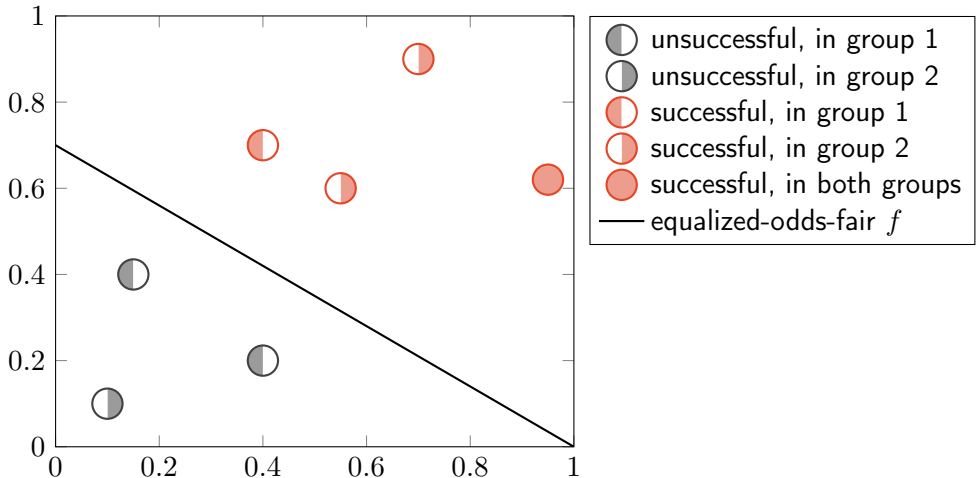
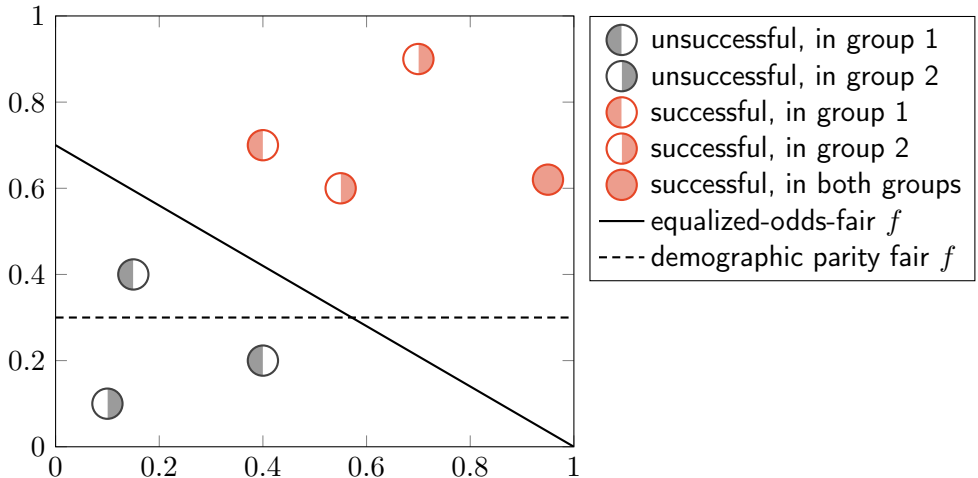


Illustration: Demographic parity vs Equalized Odds



Dynamic Fairness

(Paaßen et al. 2019)

- Definition: A model f is **dynamically fair** if it gets as close as possible to demographic parity **in the long run**

Dynamic Fairness

(Paaßen et al. 2019)

- Definition: A model f is **dynamically fair** if it gets as close as possible to demographic parity **in the long run**
- Key insight: Even a 100% accurate classifier

Dynamic Fairness

(Paaßen et al. 2019)

- Definition: A model f is **dynamically fair** if it gets as close as possible to demographic parity **in the long run**
- Key insight: Even a 100% accurate classifier, which is fair to all standards except demographic parity, is dynamically unfair

Dynamic Fairness

(Paaßen et al. 2019)

- Definition: A model f is **dynamically fair** if it gets as close as possible to demographic parity **in the long run**
- Key insight: Even a 100% accurate classifier, which is fair to all standards except demographic parity, is dynamically unfair, if classifier decisions influence future labels in a **positive feedback loop**

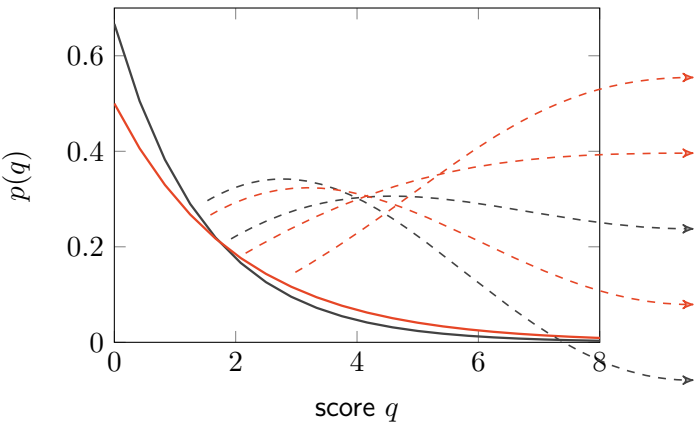
Dynamic Fairness

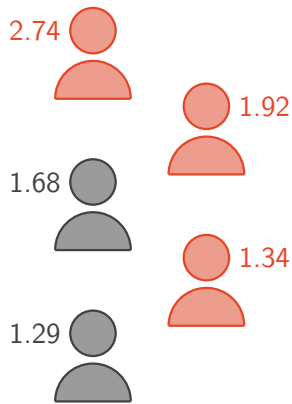
(Paaßen et al. 2019)

- Definition: A model f is **dynamically fair** if it gets as close as possible to demographic parity **in the long run**
- Key insight: Even a 100% accurate classifier, which is fair to all standards except demographic parity, is dynamically unfair, if classifier decisions influence future labels in a **positive feedback loop**
- Side note: Competing models with varying results exist (e.g. Hu and Chen 2018; Mouzannar, Ohannessian, and Nathan Srebro 2019; Creager et al. 2019)

A dynamic model for automatic decision making

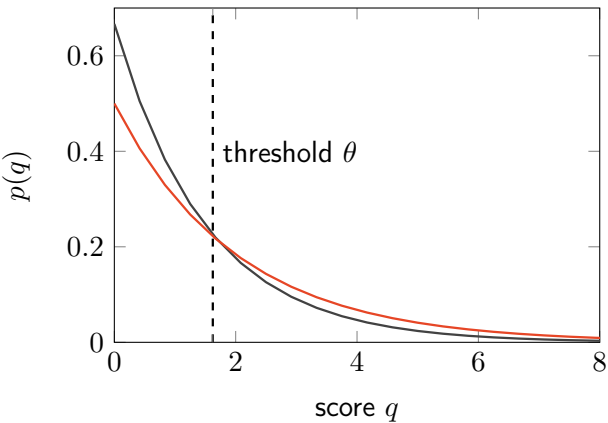
1. sample true scores from group-specific distributions

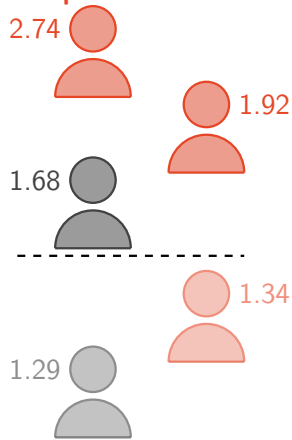




A dynamic model for automatic decision making

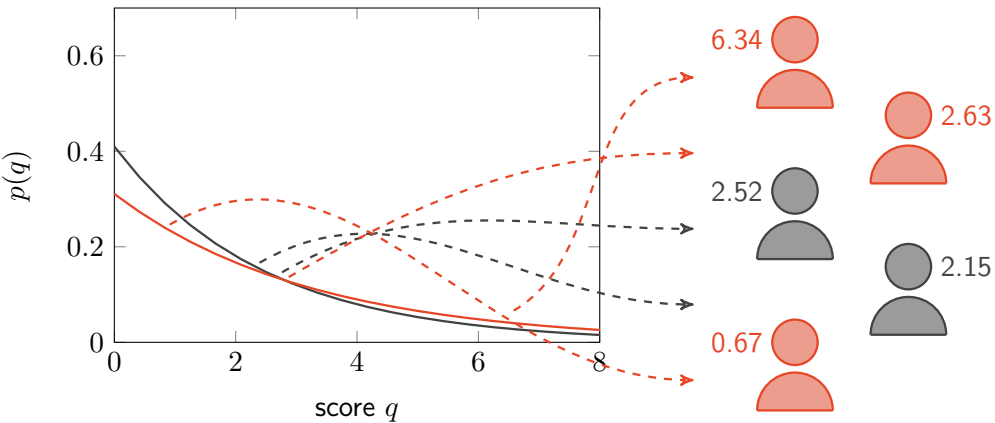
2. system has perfect knowledge and accepts top n scores





A dynamic model for automatic decision making

3. acceptance rates influence future means



Update Formula

- Assume group-specific means μ_c^t / $\mu_{\neg c}^t$ at time t.

Update Formula

- Assume group-specific means μ_c^t / $\mu_{\neg c}^t$ at time t.
- Set acceptance rates P_c^t / $P_{\neg c}^t$ such that top n people get accepted

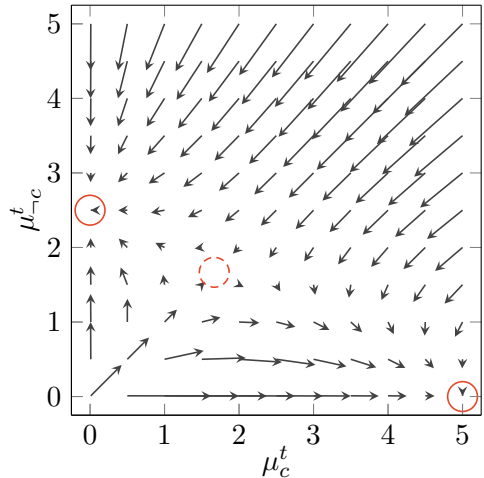
Update Formula

- Assume group-specific means μ_c^t / $\mu_{\neg c}^t$ at time t.
- Set acceptance rates P_c^t / $P_{\neg c}^t$ such that top n people get accepted
- Update means:

$$\begin{pmatrix} \mu_c^{t+1} \\ \mu_{\neg c}^{t+1} \end{pmatrix} = (1-\alpha) \cdot \begin{pmatrix} \mu_c^t \\ \mu_{\neg c}^t \end{pmatrix} + \beta \cdot \begin{pmatrix} P_c^t \\ P_{\neg c}^t \end{pmatrix}$$

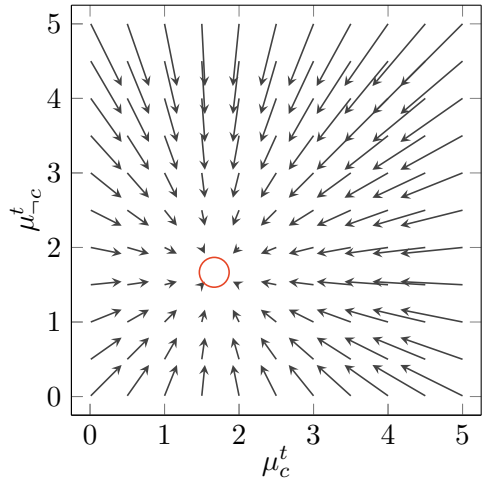
  where $\alpha \in [0, 1]$, $\beta > 0$ are hyper-parameters.

Dynamical system



Exponential distribution, $\alpha = 0.5$, $\beta = 5$, $m_{\neg c} = 2 \cdot m_c$, $n/m = 0.167$.

Dynamical system under demographic parity



Exponential distribution, $\alpha = 0.5$, $\beta = 5$, $m_{\neg c} = 2 \cdot m_c$, $n/m = 0.167$.

Qualitative findings

- Equilibria where all resources go to one group tend to be stable (across many distributions)

Qualitative findings

- Equilibria where all resources go to one group tend to be stable (across many distributions)
- Equality is possible but instable (across many distributions)

Qualitative findings

- Equilibria where all resources go to one group tend to be stable (across many distributions)
- Equality is possible but instable (across many distributions)
- Demographic parity solves both issues: Only single stable equilibrium at equality

Qualitative findings

- Equilibria where all resources go to one group tend to be stable (across many distributions)
- Equality is possible but instable (across many distributions)
- Demographic parity solves both issues: Only single stable equilibrium at equality $\Rightarrow$ yay for affirmative action policies

Summary

- Fairness in ML is a young but exploding and broad topic

Summary

- Fairness in ML is a young but exploding and broad topic
- Dissenting opinions on definitions, analytic tools, and policies

Summary

- Fairness in ML is a young but exploding and broad topic
- Dissenting opinions on definitions, analytic tools, and policies
- Close to consensus: Accuracy is not enough, breadth of views is required, fairness must be incorporated through the entire lifecycle (Myers West, Whittaker, and Crawford 2019; Smuha 2019)

Summary

- Fairness in ML is a young but exploding and broad topic
- Dissenting opinions on definitions, analytic tools, and policies
- Close to consensus: Accuracy is not enough, breadth of views is required, fairness must be incorporated through the entire lifecycle (Myers West, Whittaker, and Crawford 2019; Smuha 2019)
- My two cents: Long-term dynamics remain understudied and reveal a need for stricter policies (e.g. affirmative action)

Literature

Literature I

Angwin, Julia et al. (2016). "Machine Bias". In: **Pro Publica**. url: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

Corbett-Davies, Sam and Sharad Goel (2018). "The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning". In: **Tutorial at ICML 2018**. url: ttps://5harad.com/papers/fair-ml.pdf.

Creager, Elliot et al. (2019). **Causal Modeling for Fairness in Dynamical Systems**. arXiv: 1909.09141 [cs.LG].

Dastin, Jeffrey (2018). "Amazon scraps secret AI recruiting tool that showed bias against women". In: **Reuters**. url: https://uk.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUKKCN1MK08G.

Literature II

Dornis, Valentin (2018). "Arbeit aus dem Automaten". In: **Süddeutsche Zeitung**. url: https://www.sueddeutsche.de/digital/digitalisierung-arbeitslosigkeit-jobcenter-1.4178635.

Dwork, Cynthia et al. (2012). "Fairness Through Awareness". In: **ITCS 2012**. Cambridge, Massachusetts, pp. 214–226. doi: 10.1145/2090236.2090255.

Ensign, Danielle et al. (2018). "Runaway Feedback Loops in Predictive Policing". In: **Proceedings of the 1st Conference on Fairness, Accountability and Transparency (FAT 2018)**. Ed. by Sorelle A. Friedler and Christo Wilson. Vol. 81. Proceedings of Machine Learning Research, pp. 160–171. url: http://proceedings.mlr.press/v81/ensign18a.html.

Grgić-Hlača, Nina et al. (2016). "The Case for Process Fairness in Learning: Feature Selection for Fair Decision Making". In: **NIPS 2016 Workshop "ML and the Law"**. url: http://www.mlandthelaw.org/papers/grgic.pdf.

Literature III

Hardt, Moritz, Eric Price, and Nati Srebro (2016). "Equality of Opportunity in Supervised Learning". In: **NIPS 2016**, pp. 3315–3323. url: http://papers.nips.cc/paper/6374-equality-of-opportunity-in-supervised-learning.pdf.

Hern, Alex (2018). "Google's solution to accidental algorithmic racism: ban gorillas". In: **The Guardian**. url: https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people.

Hu, Lily and Yiling Chen (2018). "A Short-term Intervention for Long-term Fairness in the Labor Market". In: **WWW 2018**, pp. 1389–1398. doi: 10.1145/3178876.3186044.

Kilbertus, Niki et al. (2017). "Avoiding Discrimination through Causal Reasoning". In: **NIPS 2017**, pp. 656–666. url: http://papers.nips.cc/paper/6668-avoiding-discrimination-through-causal-reasoning.

Literature IV

Lohr, Steve (2018). "Facial Recognition Is Accurate, if You're a White Guy". In: **The New York Times**. url: https://www.nytimes.com/2018/02/09/technology/facial-recognition-race-artificial-intelligence.html.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: **nature** 518.7540, p. 529. doi: 10.1038/nature14236.

Mouzannar, Hussein, Mesrob I. Ohannessian, and Nathan Srebro (2019). "From Fair Decision Making To Social Equality". In: **Proceedings of the Conference on Fairness, Accountability, and Transparency**. FAT* '19. Atlanta, GA, USA, pp. 359–368. doi: 10.1145/3287560.3287599. url: https://arxiv.org/abs/1812.02952.

Myers West, Sarah, meredith Whittaker, and Kate Crawford (2019). **Discriminating Systems - Gender, Race, and Power in AI**. AI Now Institute. url: https://ainowinstitute.org/discriminatingsystems.html.

Literature V

Paaßen, Benjamin et al. (Apr. 2019). "Dynamic fairness – Breaking vicious cycles in automatic decision making". In: **Proceedings of the 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2019)** (Bruges, Belgium). Ed. by Michel Verleysen, pp. 477–482. url: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2019-134.pdf.

Pachocki, Jakub et al. (2019). **OpenAI Five**. url: https://openai.com/five.

Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: **nature** 529.7587, p. 484. doi: 10.1038/nature16961.

Smuha, Nathalie (2019). **Ethics guidelines for trustworthy AI**. EU High-Level Expert Group on Artificial Intelligence. url: https://ec.europa.eu/futurium/en/ai-alliance-consultation.

Tesauro, Gerald (1995). "Temporal Difference Learning and TD-Gammon". In: **Communications of the ACM** 38.3, pp. 58–68. doi: 10.1145/203330.203343.

Literature VI

Thakor, Anjan V. (2015). "The Financial Crisis of 2007–2009: Why Did It Happen and What Did We Learn?" In: **The Review of Corporate Finance Studies** 4.2, pp. 155–205. doi: 10.1093/rcfs/cfv001.

Vinyals, Oriol et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: **nature**, pp. 1–5. doi: 10.1038/s41586-019-1724-z.