ELTE | FACULTY OF INFORMATICS

REINFORCMENT LEARNING

Introduction to Machine Learning – Lecture 6.

Balázs Nagy, PhD

ELTE | IK

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

BOSCH

References

- MIT course

  © Alexander Amini and Ava Amini

  MIT 6.S191: Introduction to Deep Learning

  http://introtodeeplearning.com/

- Georgia Tech – Machine Learning course

  https://www.youtube.com/watch?v=Jk2V9yA82YU

- What is general intelligence? How can it be defined?

A very general mental capability that, among other things, involves ***the ability to*** *reason, plan,* ***solve problems****, think abstractly, comprehend complex ideas, learn quickly* ***and learn from experience****. It is not merely book learning, a narrow academic skill, or test-taking smarts. Rather, it reflects a broader and deeper capability for comprehending our surroundings—"catching on," "****making sense****" of things, or "****figuring out****" what to do.*

/Wall Street Journal, 1994/

- Intelligence in the context of machines – Artificial Intelligence (AI)

  The theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

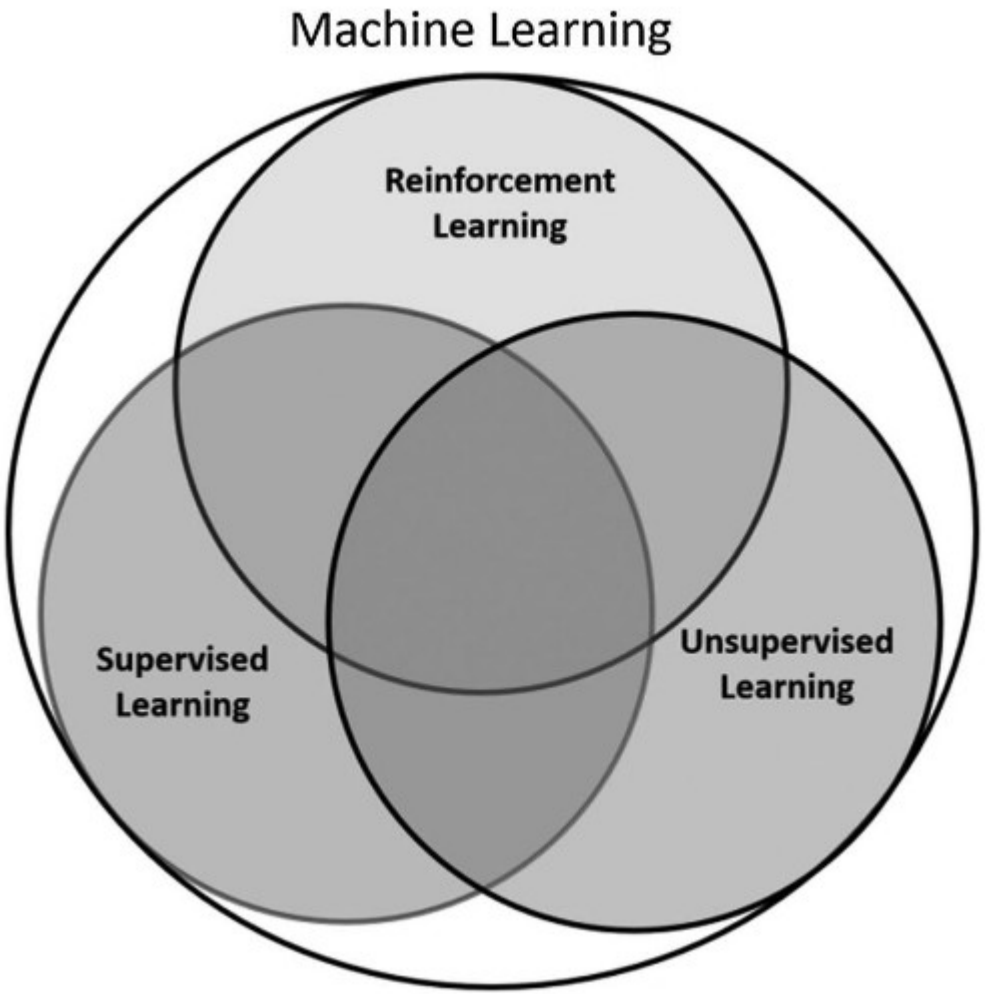
- Goal:

  Study the theory and design of algorithms that help machines (**agents**) gain the capability to perform tasks by **interacting with the environment** and **continually learning** from its successes, failures, and **rewards**

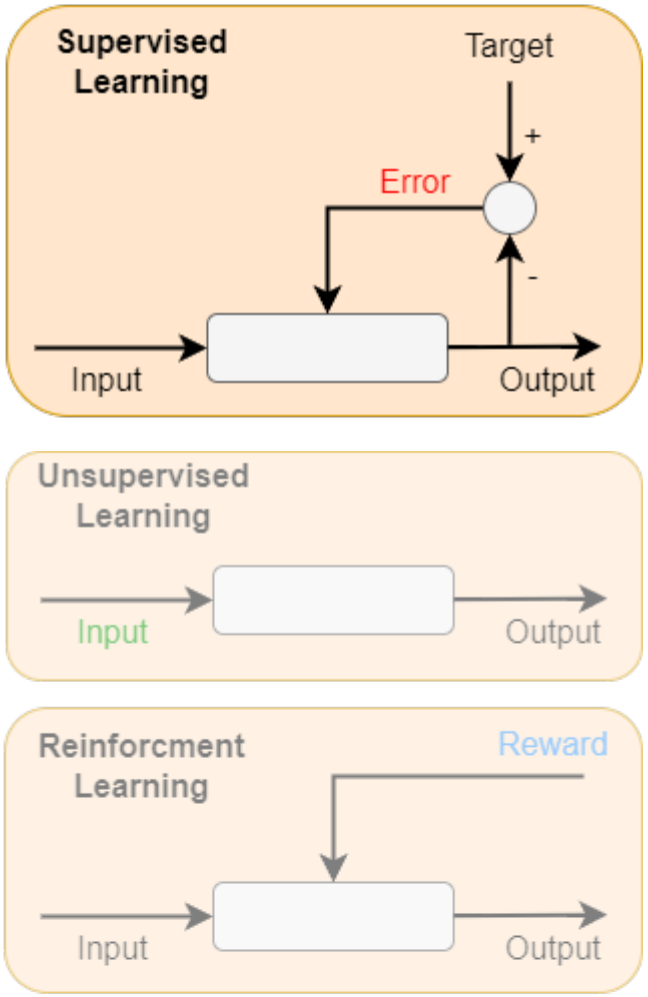
Artificial Intelligence
Any technique that enables computers to mimic human intelligence.

Machine Learning
A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience.

Deep Learning
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks by exposing multilayered neural networks to vast amount of data.

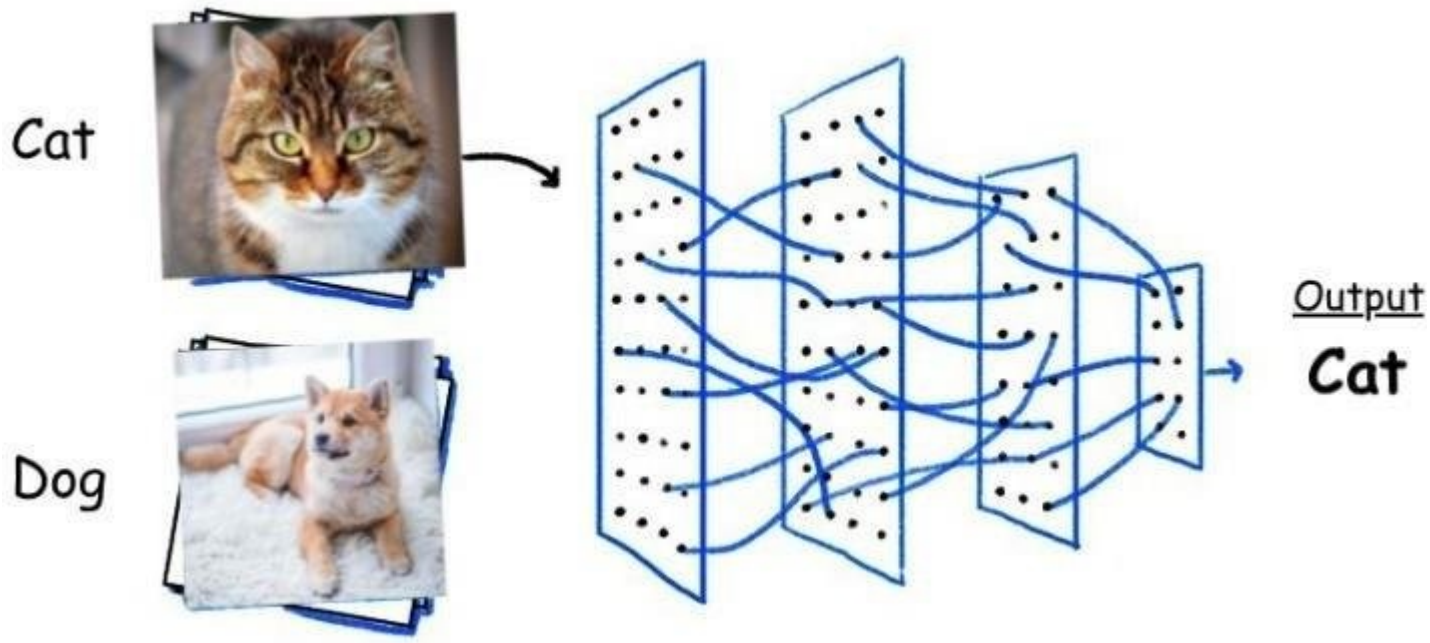


Learning methods

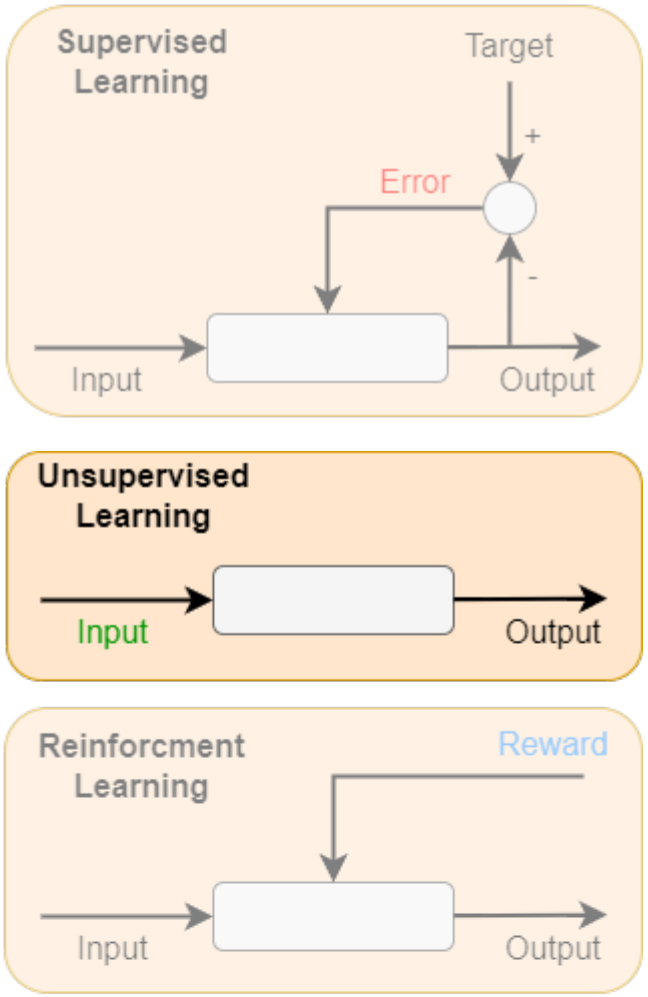


Supervised Learning

- system is presented with the labeled data
- the objective is to generalize the knowledge so that new unlabeled data can be labeled

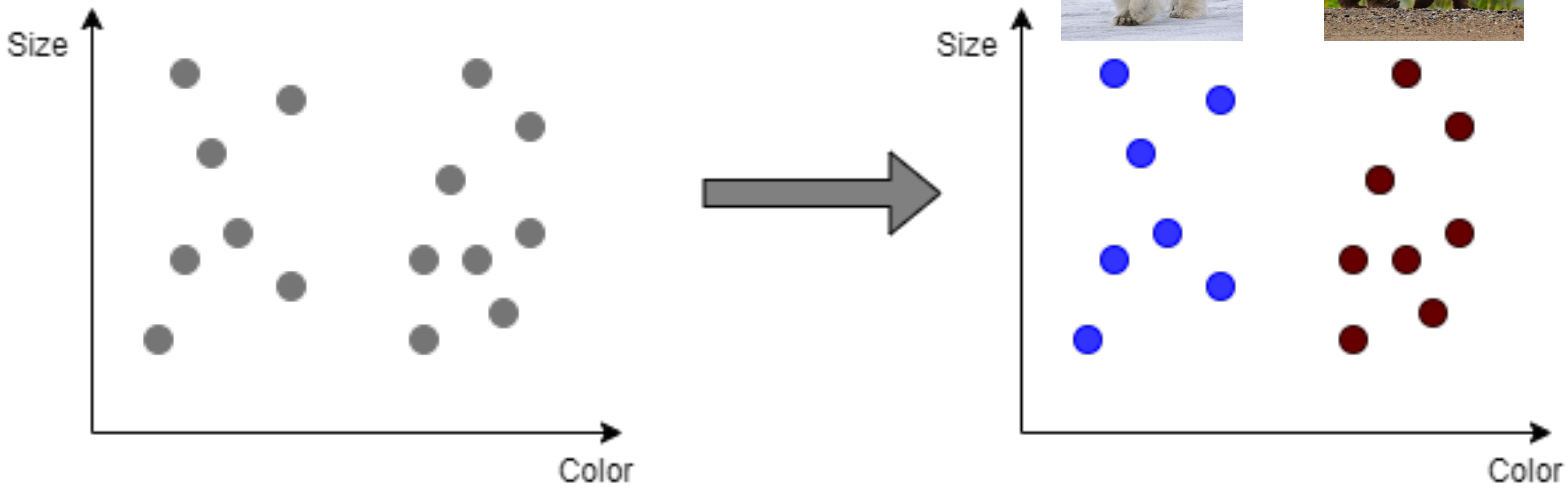


Learning methods

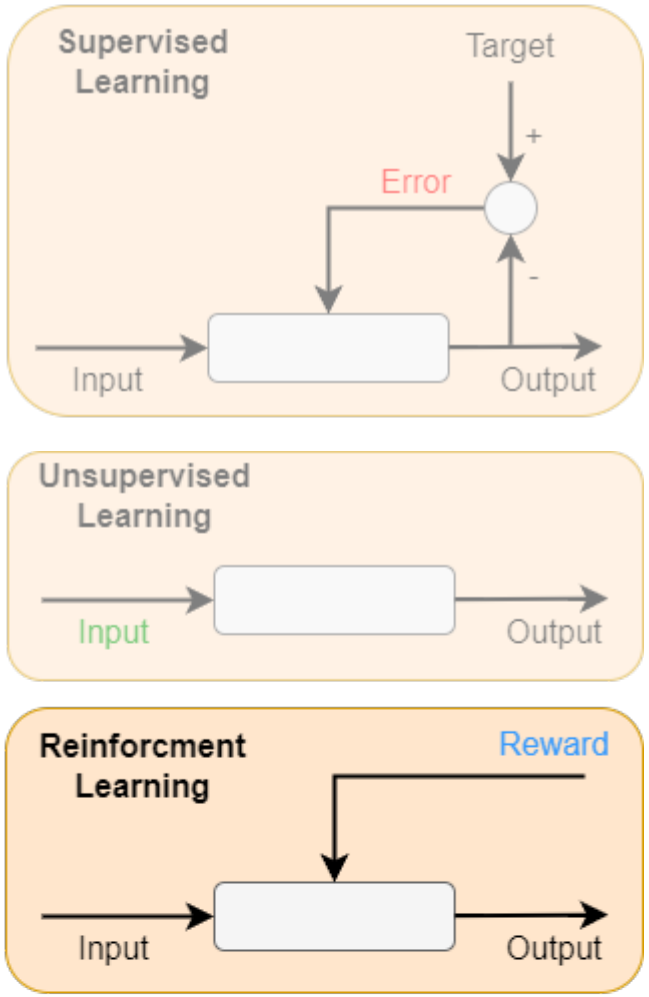


Unsupervised Learning

- No labels, only has the inputs
- The system uses this data to learn the hidden structure of the data so that it can cluster/categorize the data into some broad categories.
- Often used for feature extraction



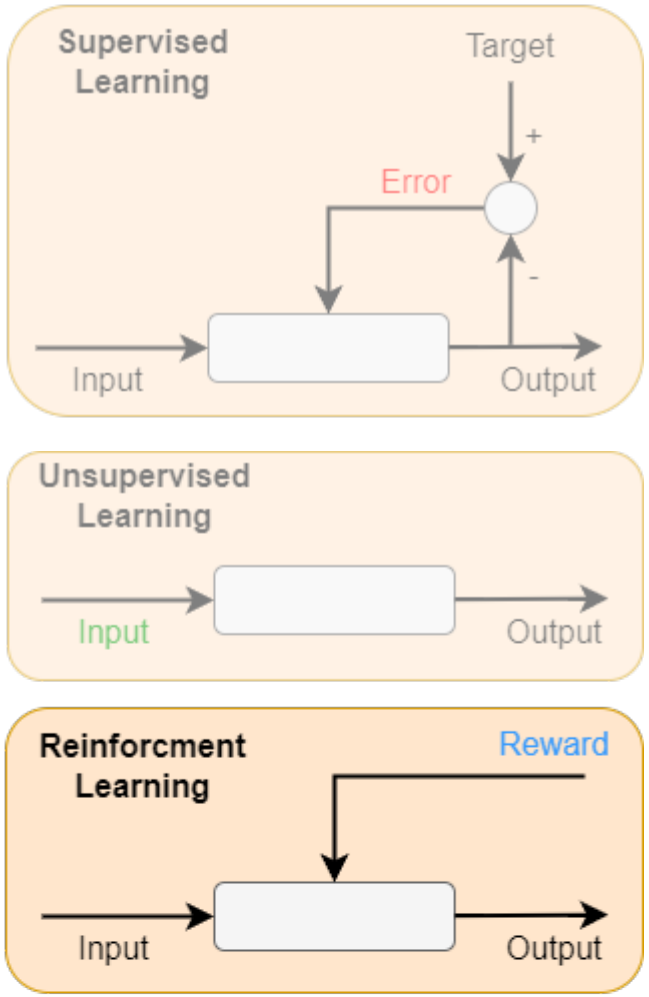
Learning methods



Reinforcment Learning

- The agent does not have prior knowledge of the system
- It gathers feedback and uses that feedback to plan/learn actions to maximize a specific objective.
- As it does not have enough information about the environment initially, it must explore to gather insights.
- Once it gathers “enough” knowledge, it needs to **exploit** that **knowledge** to start **adjusting** its **behavior** to **maximize the objective** it is chasing.

Learning methods



Reinforcement Learning (RL) – Key concept

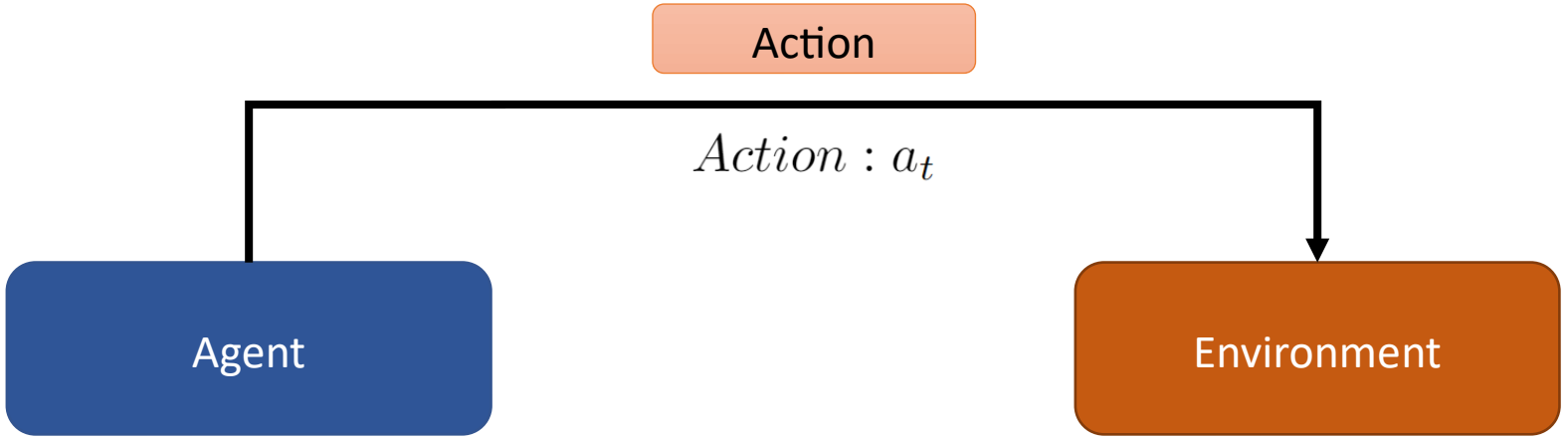
Reinforcement Learning (RL) – Key concept

Agent: takes actions

Reinforcement Learning (RL) – Key concept

Environment: the world in wich the agent exists and operates

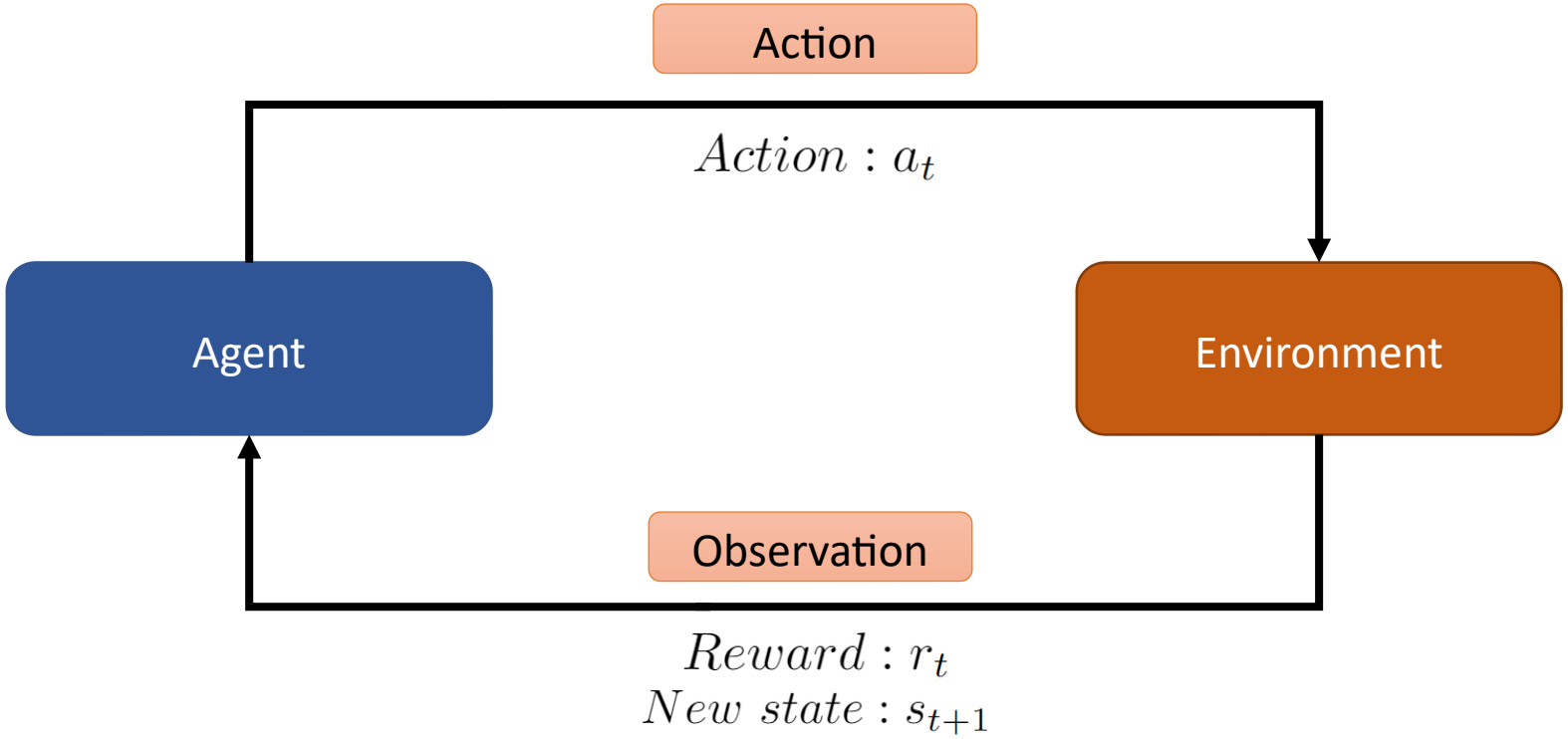
Reinforcement Learning (RL) – Key concept



Action: a movement the agent can make in the environment

Action space *A*: the set of possible actions an agent can make in the environment

Reinforcement Learning (RL) – Key concept



Reward: feedback that measures the success or failure of the agent's action

Grid world example

Environment

Agent

State

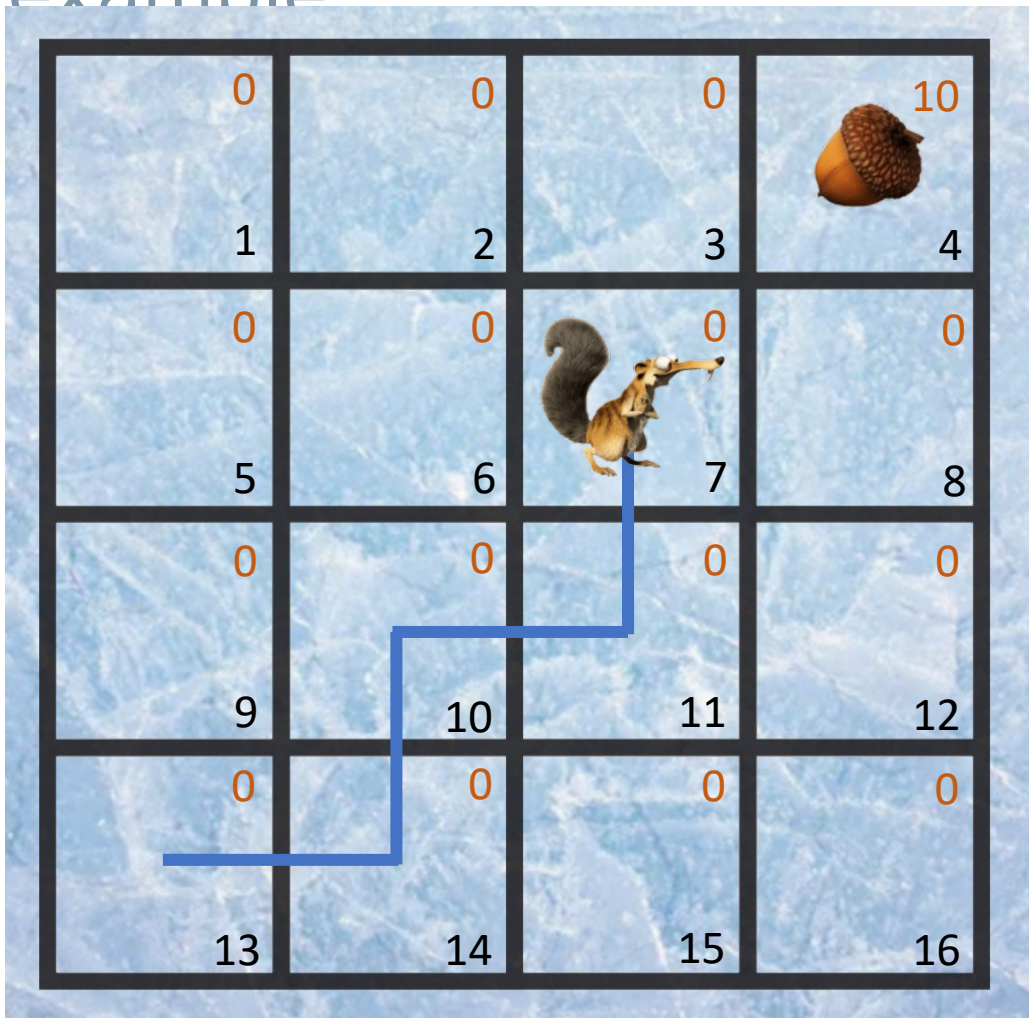
Action

Reward

Model: rules of the game, the physics of the world

Policy: A policy is a strategy that an agent uses in pursuit of goals.

Grid world example



Environment

Agent

State

Action

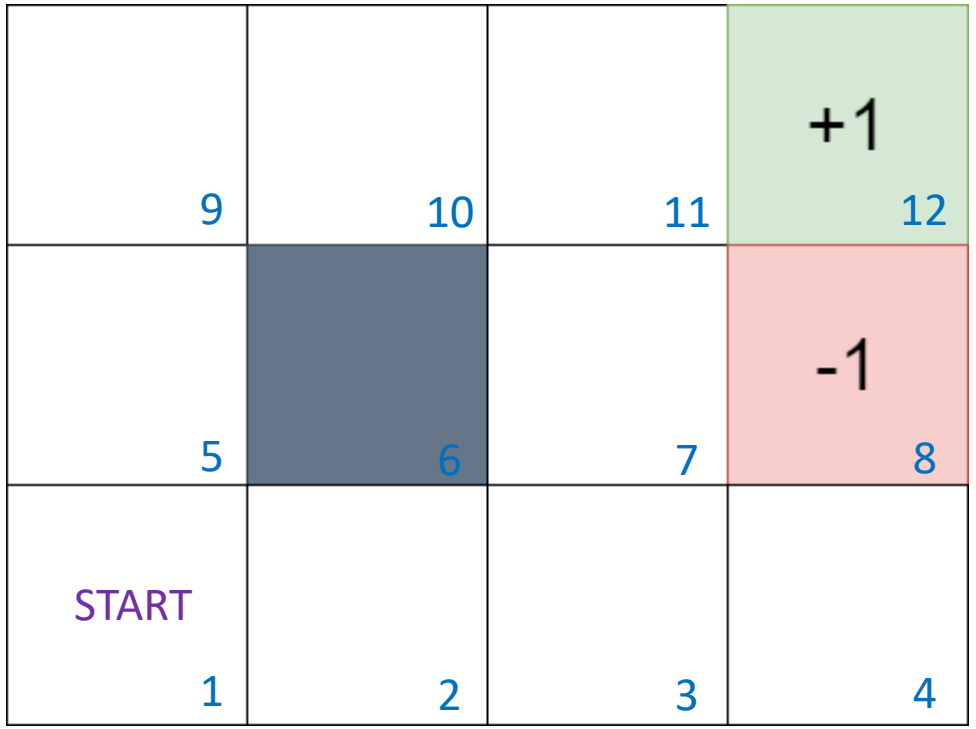
Reward

} Trajectory

$$S_{13} \rightarrow A_r \rightarrow R(+0) \rightarrow S_{14} \rightarrow A_u \rightarrow R(+0) \rightarrow$$
$$\rightarrow S_{10} \rightarrow A_r \rightarrow R(+0) \rightarrow \ldots$$

Markov Decision Process

Markov Decision Process



[up, down, left, right]

States: S =[1,2,3,4,5,6,7,8,9,10,11,12]

Model: $T(s, a, s') \sim Pr(s'|s, a)$

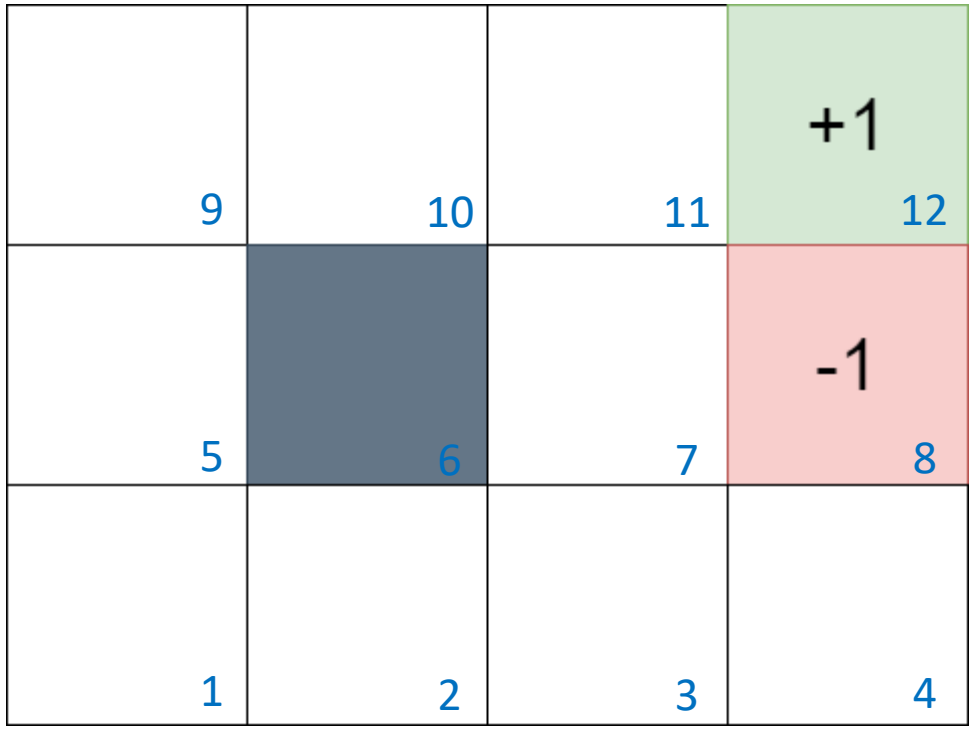
Actions: $A(s), A$ =[up, down, left, right]

Reward: $R(s), R(s, a), R(s, a, s')$

Policy: $\pi(s) \rightarrow a$

π^*

Markov Decision Process



[up, down, left, right]

Model: $T(s, a, s') \sim Pr(s'|s, a)$

Deterministic:

$$Pr(s_5|s_1, a_{up}) = 1$$
$$Pr(s_2|s_1, a_{up}) = 0$$

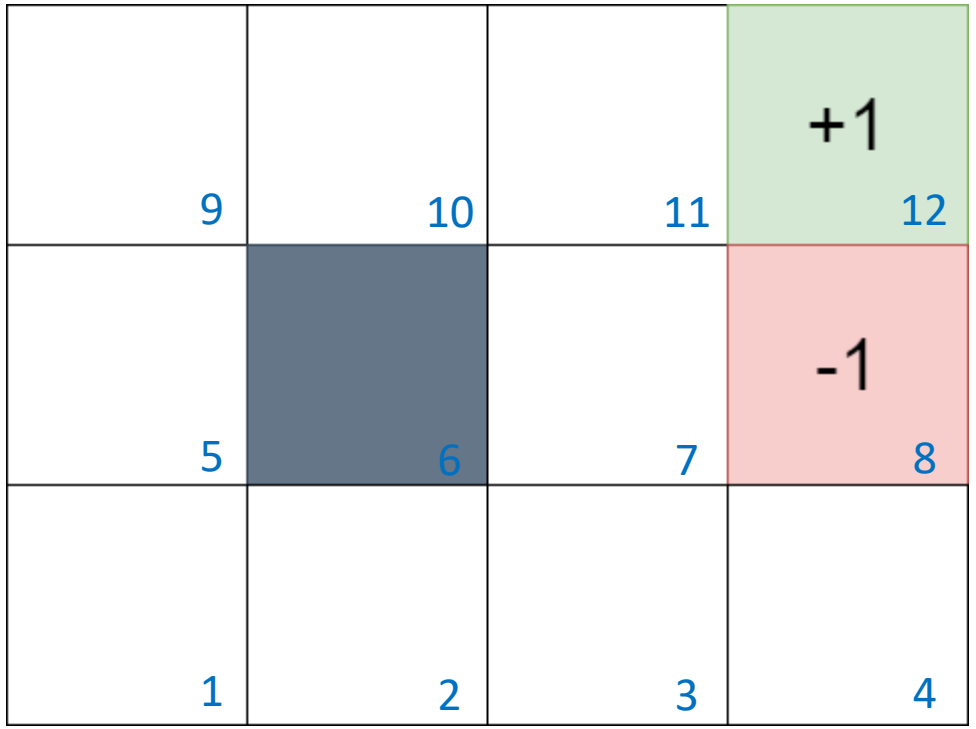
Sum = 1

Non-deterministic or Sto

$$Pr(s_7|s_3, a_{up}) = 0.8$$
$$Pr(s_4|s_3, a_{up}) = 0.1$$
$$Pr(s_2|s_3, a_{up}) = 0.1$$
$$Pr(s_1|s_3, a_{up}) = 0$$

Sum = 1

Markov Decision Process



[up, down, left, right]

Markovian property:

- Only present matters
- Stationary (rules do not change)

MDP

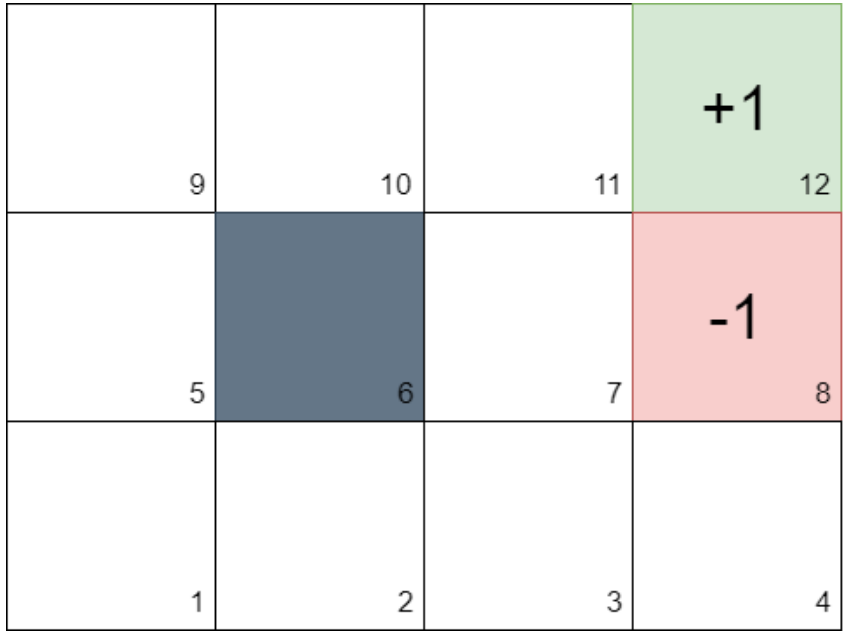
States:	S
Model:	$T(s, a, s') \sim Pr(s'\|s, a)$
Actions:	$A(s), A$
Reward:	$R(s), R(s, a), R(s, a, s')$

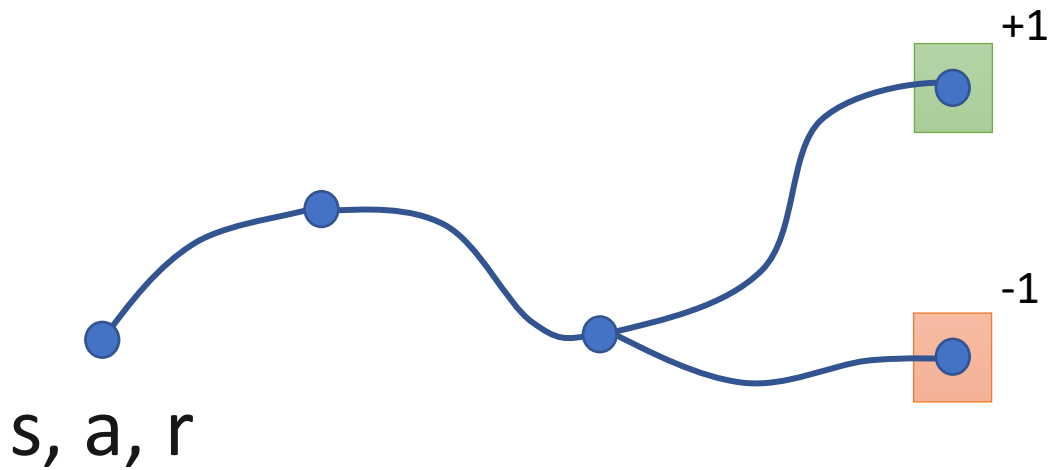
Problem definition

Policy:	$\pi(s) \rightarrow a$
π^*	**Optimal policy:** maximises the long term expected reward

Solution

Markov Decision Process

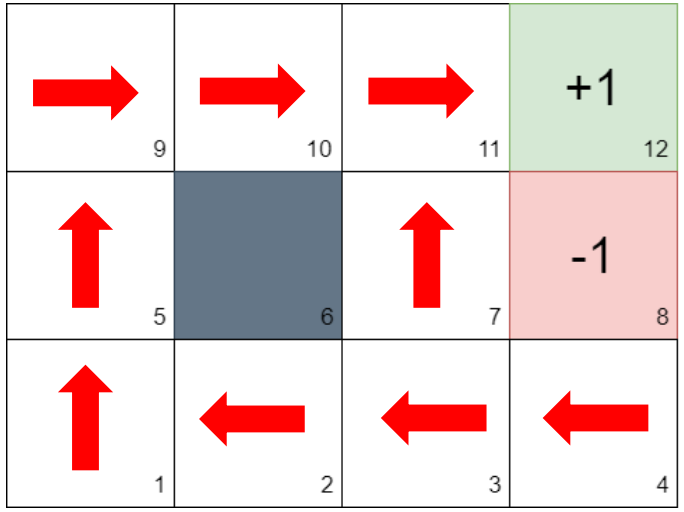




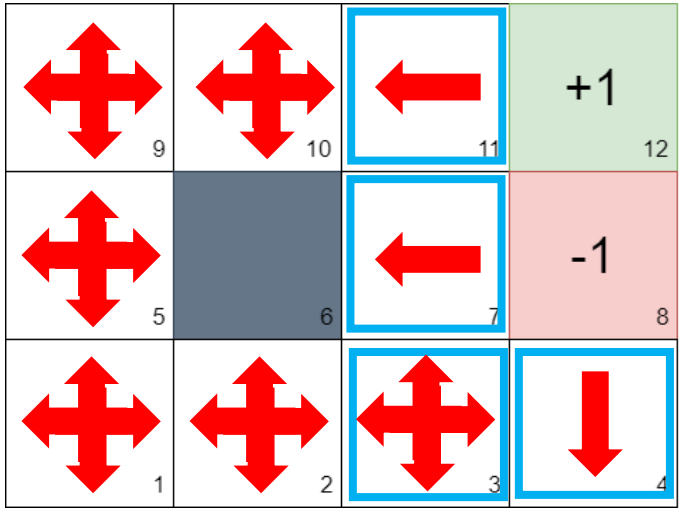
- Delayed reward
- Minor changes matters

Temporal Credit Assigment

Markov Decision Process



$$R(s) = +2$$



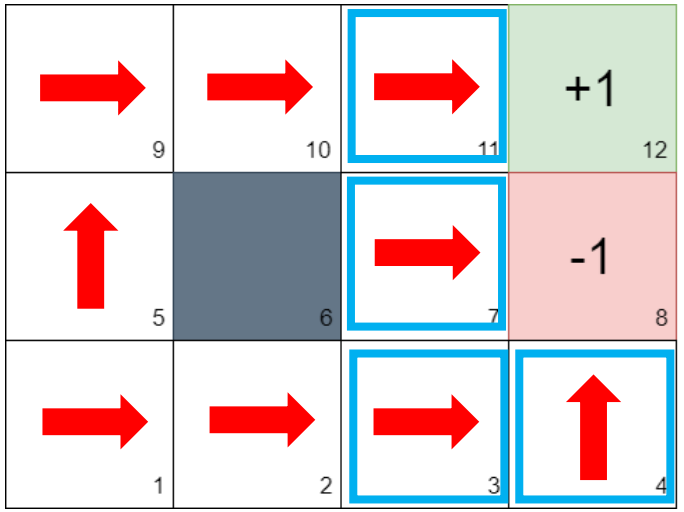
Action Space:

Rules:

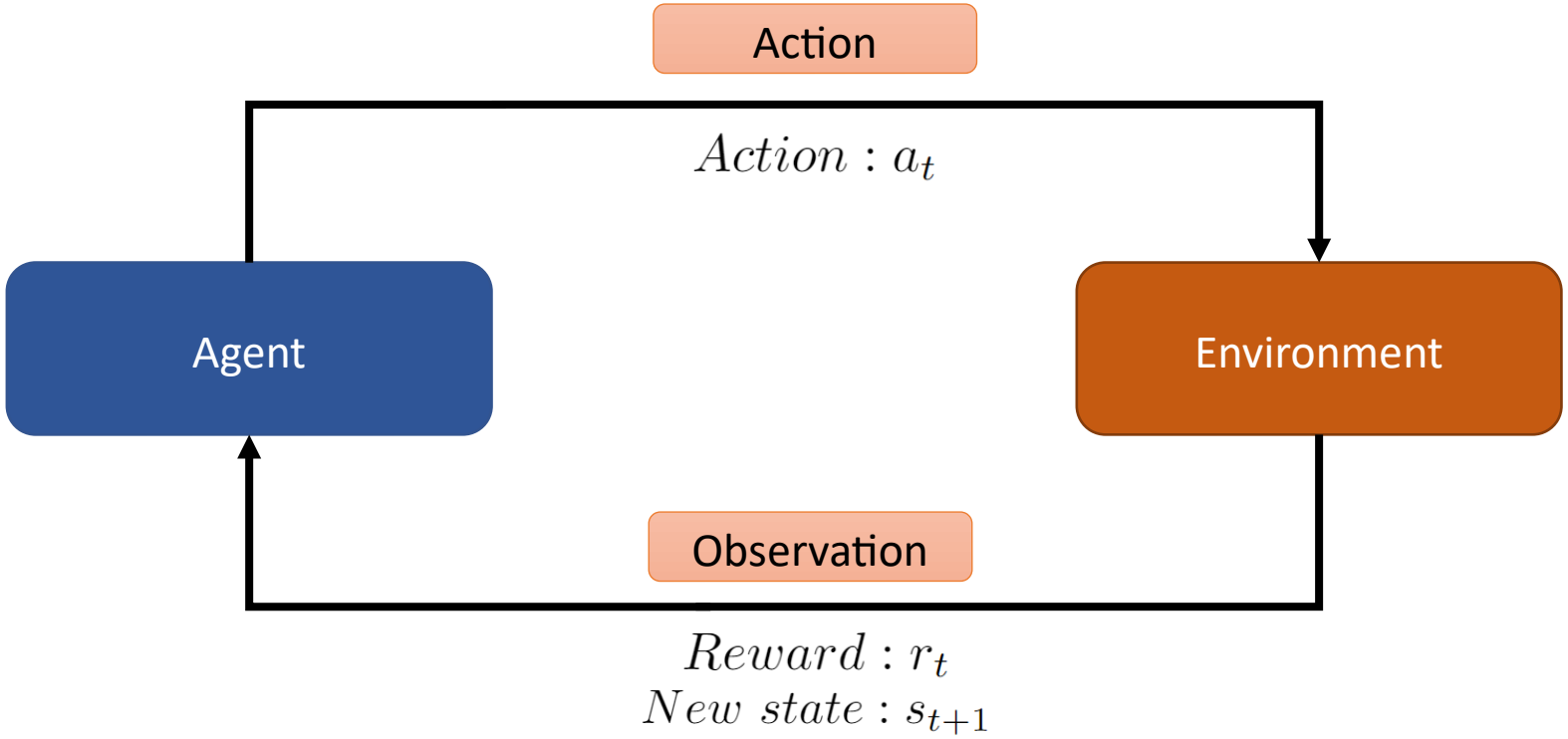
- **Stochastic**
- **Rewards given**

$$R(s) = -0.04$$

$$R(s) = -2$$



Reinforcement Learning (RL) – Key concept



Total Reward

$$R_t = \sum_{i=t}^{\infty} r_i$$

$$R_t = \sum_{i=t}^{\infty} \gamma^i r_i$$

Discounted Total Reward

Defining the Q-function

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots$$

Total reward, $\mathbf{R_t}$, is the discounted sum of all rewards obtained from time **t**

$$Q(s_t, a_t) = \mathbb{E}[R_t | s_t, a_t]$$

The Q-function captures the expected total future reward an agent in state, s, can receive by executing a certain action, a

How to take actions given a Q-function?

$$Q(s_t, a_t) = \mathbb{E}[R_t | s_t, a_t]$$

Ultimately, the agent needs a **policy $\pi(s)$**, to infer the best action to take at its state, ***s***

Strategy: the policy should choose an action that maximizes future reward

$$\pi^*(s) = \arg\max_a Q(s, a)$$

Deep Reinforcment Learning Algorithms

Value Learning

Find $Q(s,a)$

$a = \underset{a}{\text{argmax}}\ Q(s,a)$

Policy Learning

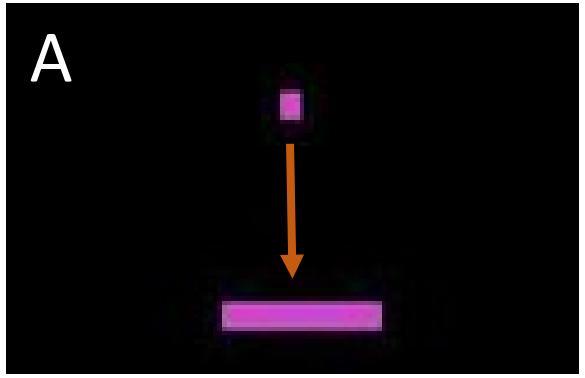
Find $\pi(s)$

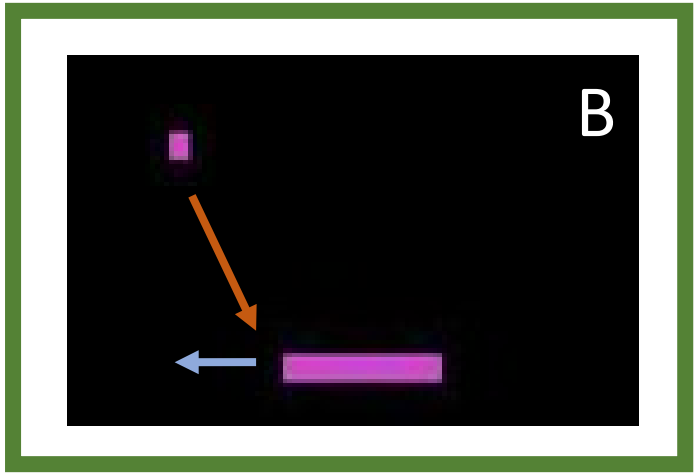
Sample $a \sim \pi(s)$

Q function intuition

Atari game - Breakout

It can be very difficult for humans to accurately estimate Q-values

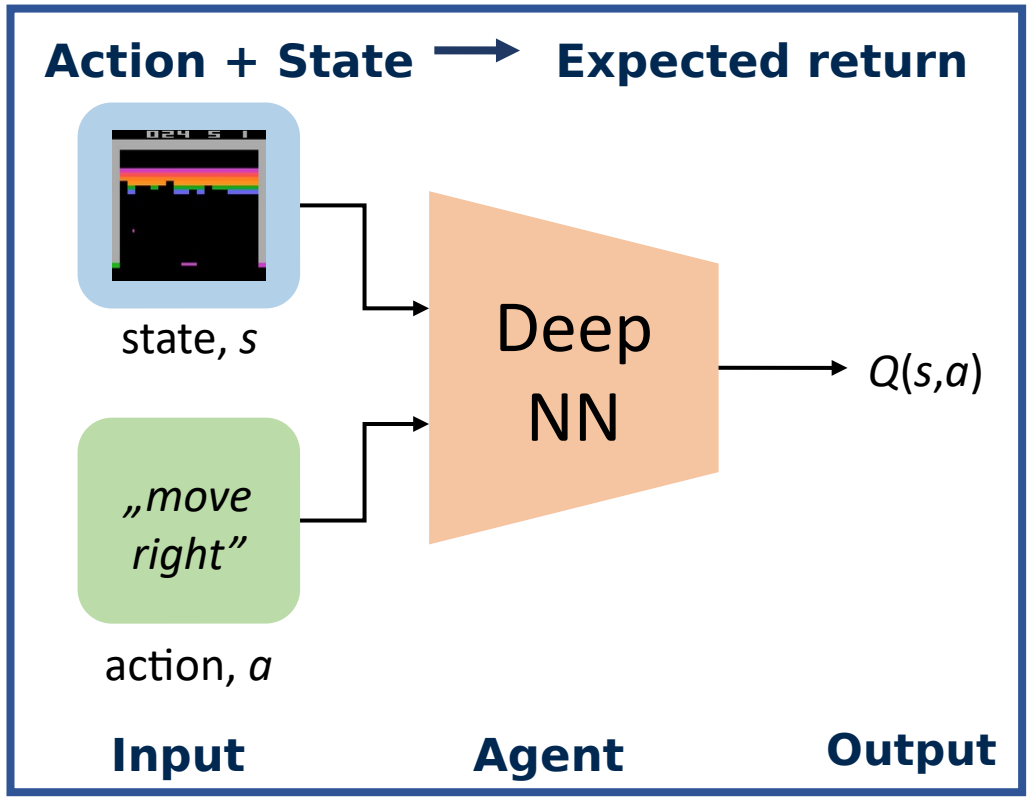


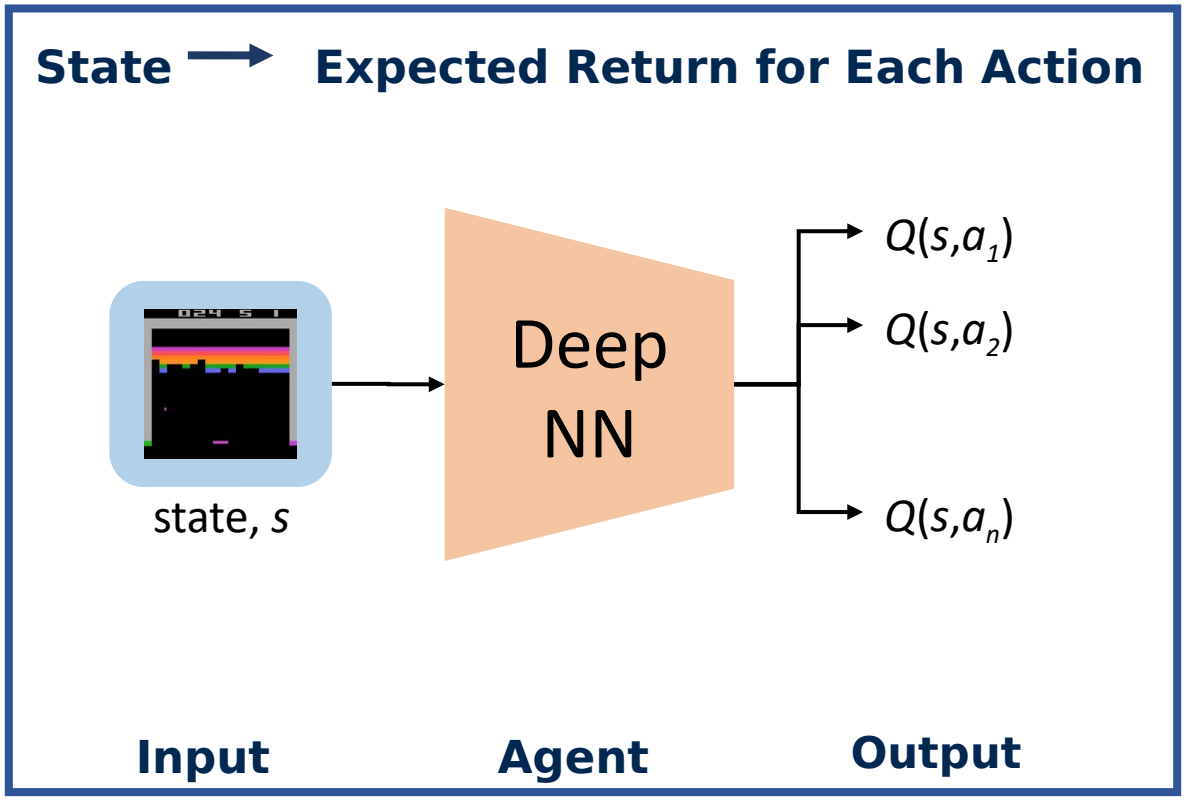


Which (s, a) pair has a higher Q-value? 🤔

Deep Q Networks (DQN): Training

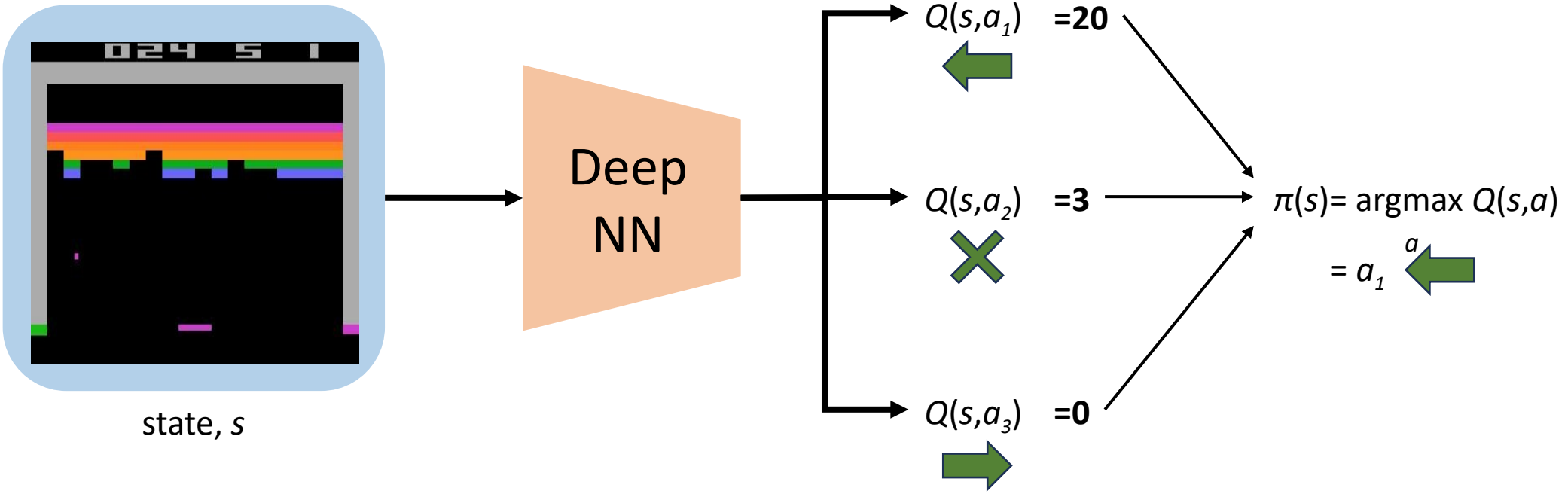
How can we use deep neural networks to model Q-functions?





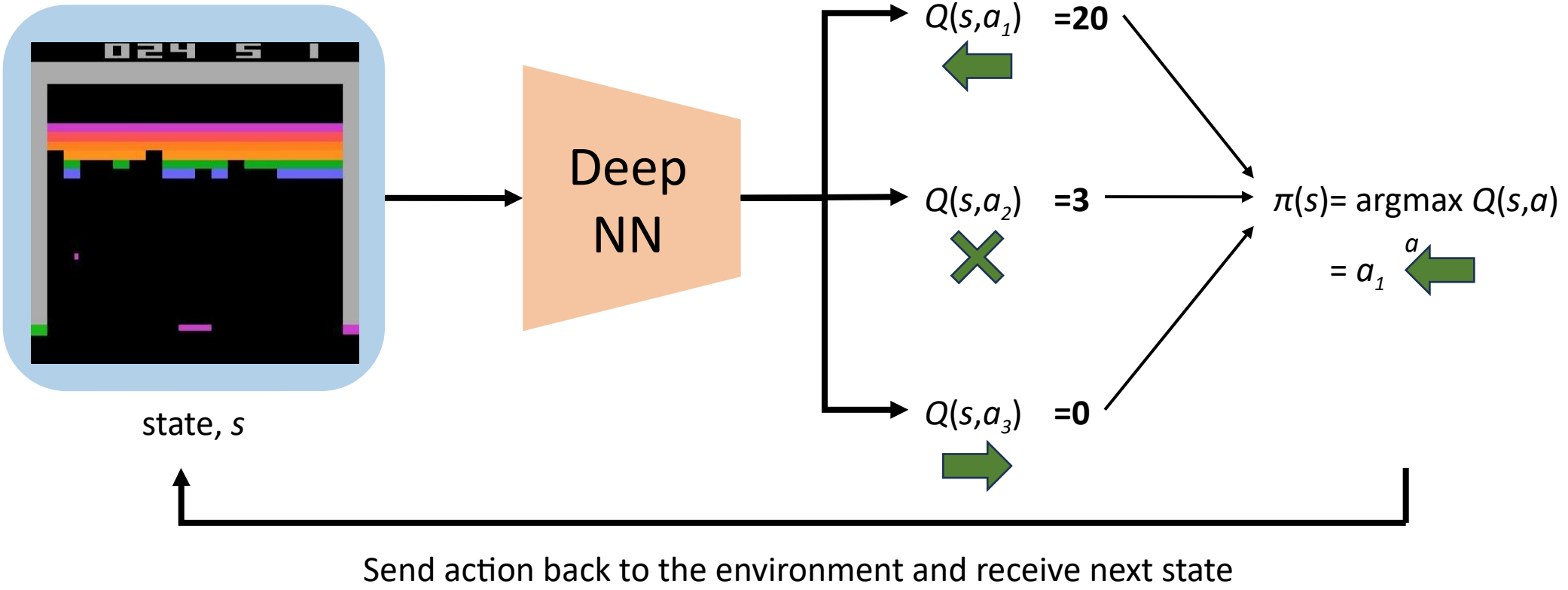
Deep Q Networks Summary

Use NN to learn Q-function and then use to infer the optimal policy, $\pi(s)$

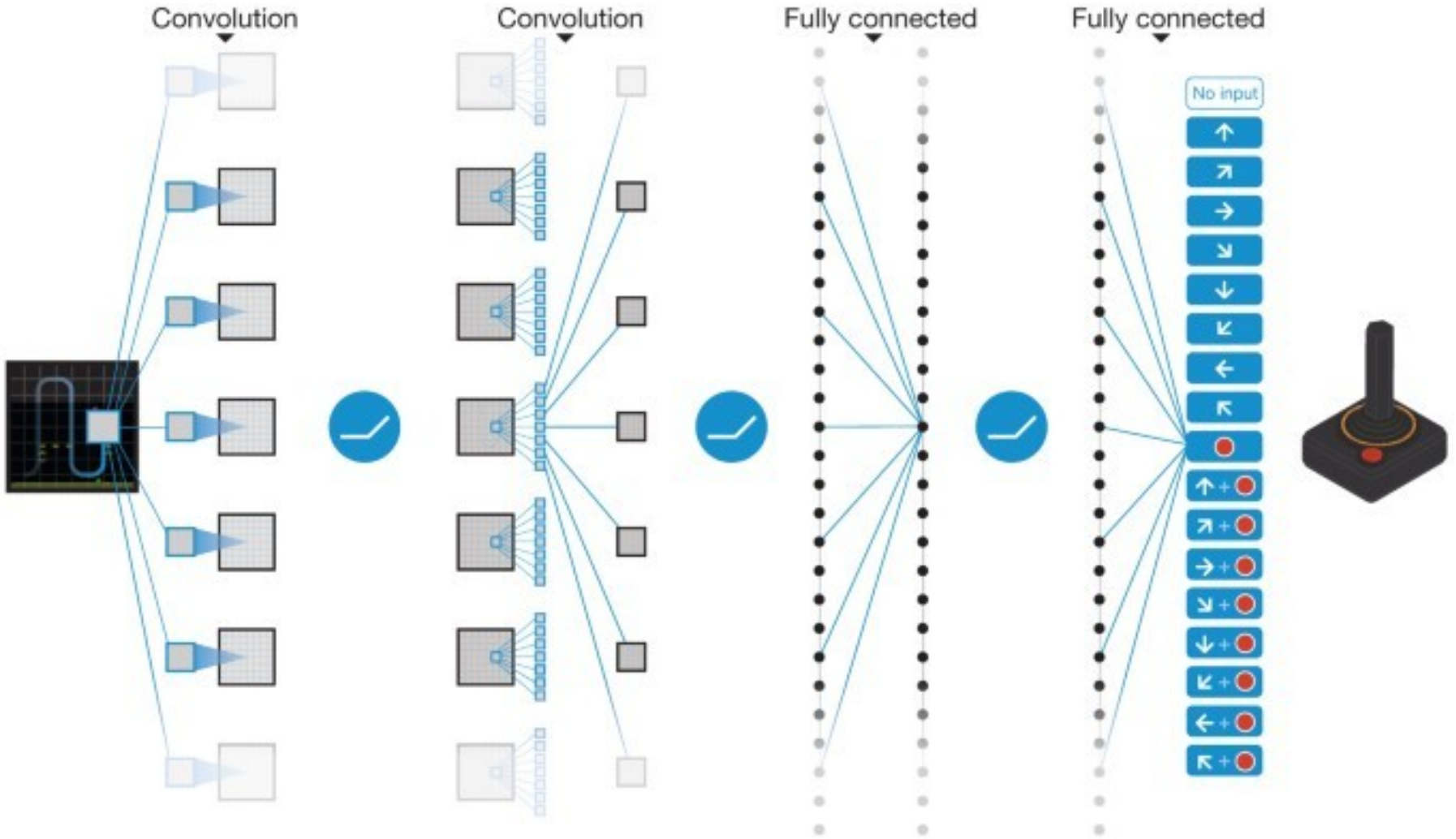


Deep Q Networks Summary

Use NN to learn Q-function and then use to infer the optimal policy, $\pi(s)$

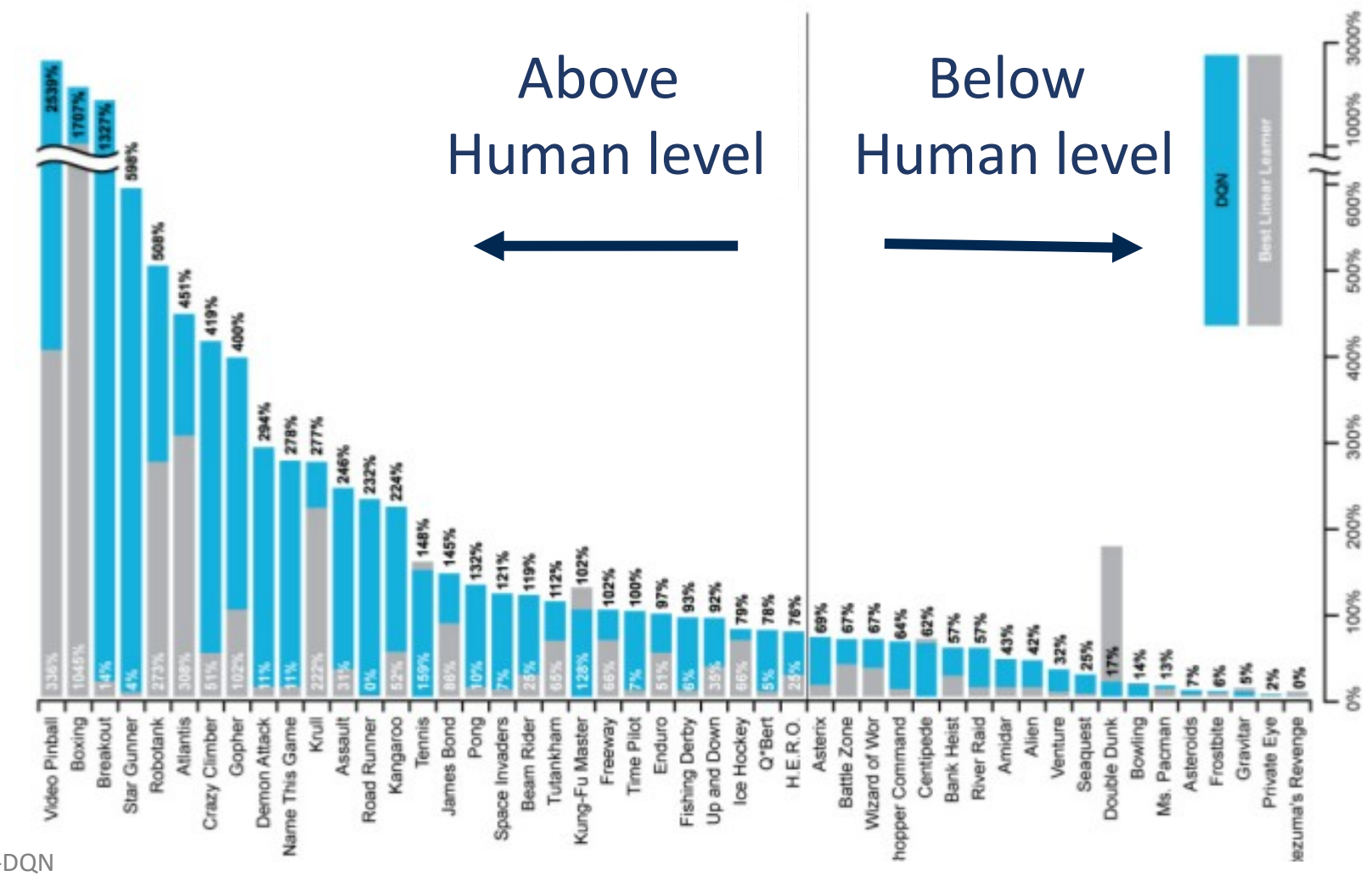


DQN Atari playing Network



kcuoglu, K., Silver, D. *et al.* Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015). https://doi.org/10.1038/nature14236

DQN Atari Results



https://github.com/Neo-47/Atari-DQN

Downsides of Q-learning

Complexity:

- Can model scenarios where the action space is discrete and small
- Cannot handle continuous action spaces

Flexibility:

- Policy is deterministically computed from the Q function by maximizing the reward → cannot learn stochastic policies

To address these, consider a new class of RL training algorithms:

Policy gradient methods

Deep Reinforcement Learning Algorithms

Value Learning

Find $Q(s,a)$

$a = \underset{a}{\text{argmax}}\ Q(s,a)$

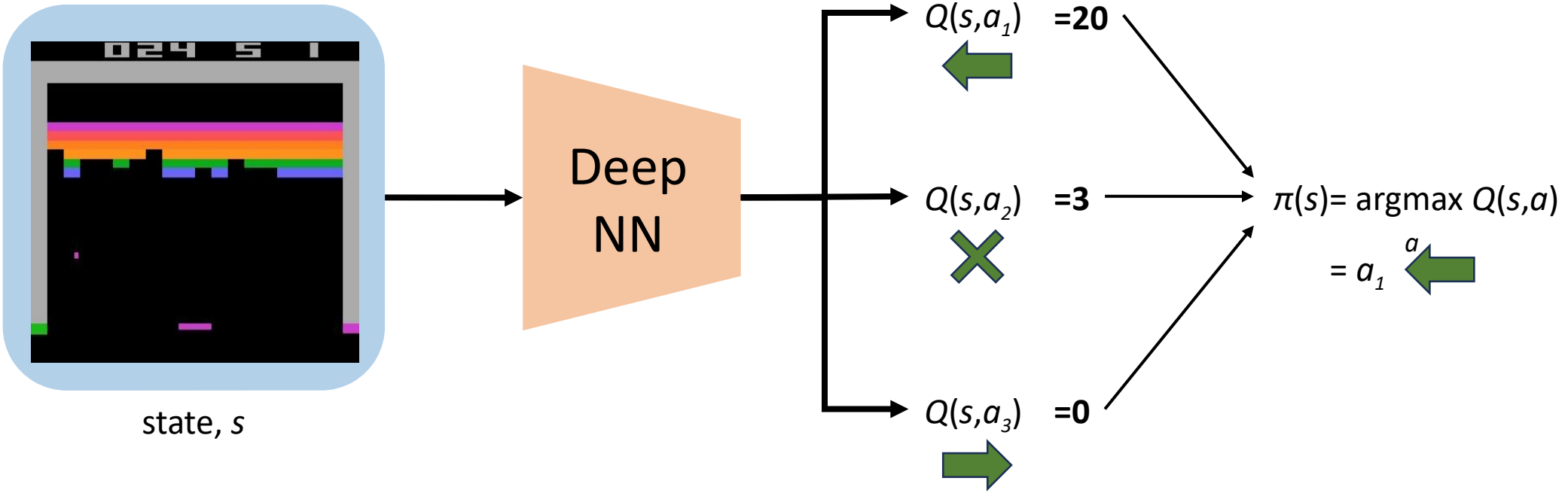
Policy Learning

Find $\pi(s)$

Sample $a \sim \pi(s)$

Deep Q Networks Summary

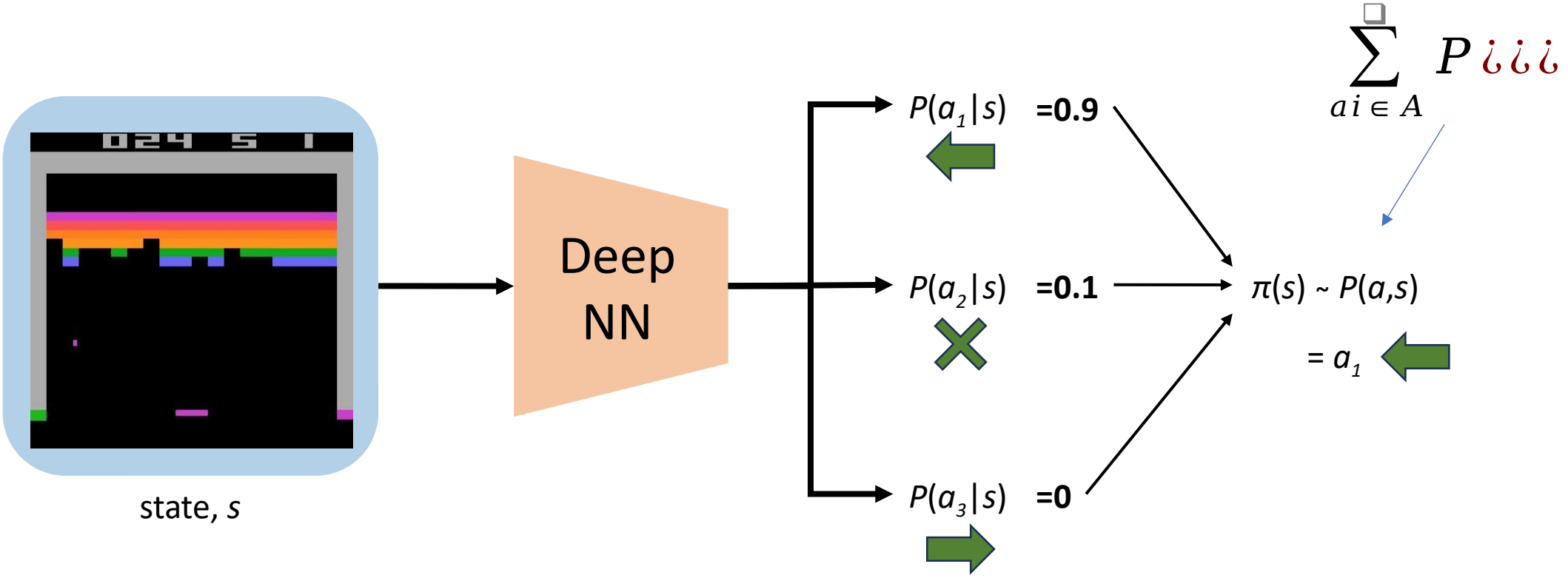
DQN: Approximate Q function and use to infer the optimal policy, $\pi(s)$



Policy Gradient (PG): Key Idea

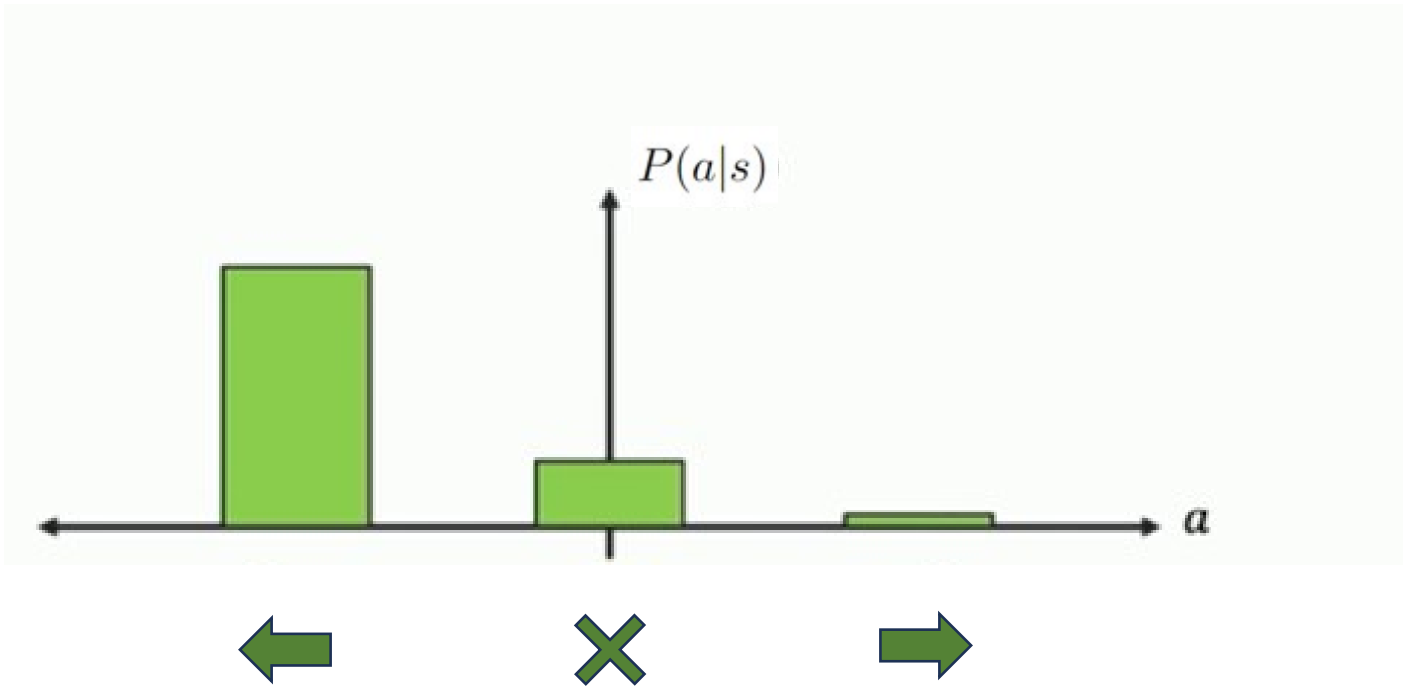
DQN: Approximate Q function and use to infer the optimal policy, $\pi(s)$

Policy Gradient: Directly optimize the policy $\pi(s)$



Discrete vs Continous Action Spaces

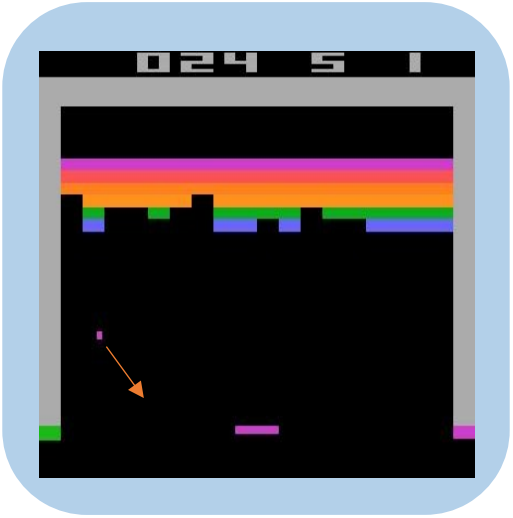
Discrete action space: which direction should I move?

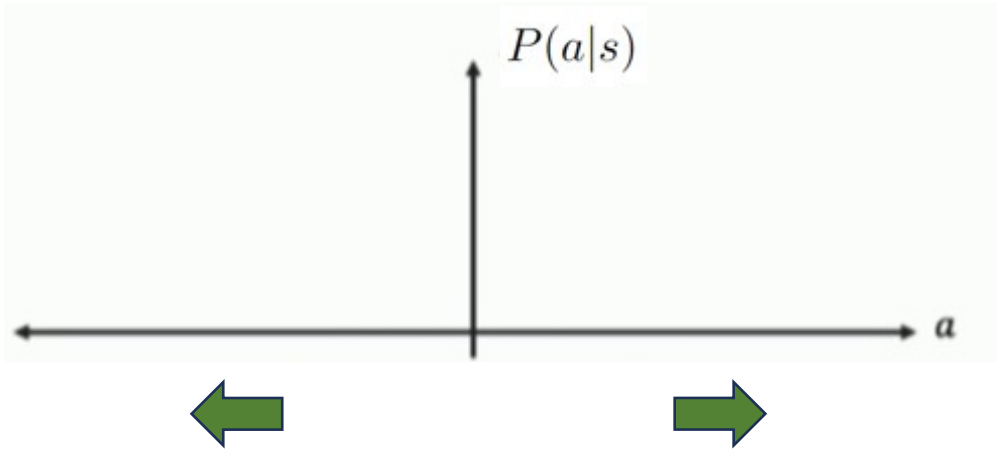


Discrete vs Continous Action Spaces

Discrete action space: which direction should I move?

Continous action space: how fast should I move?

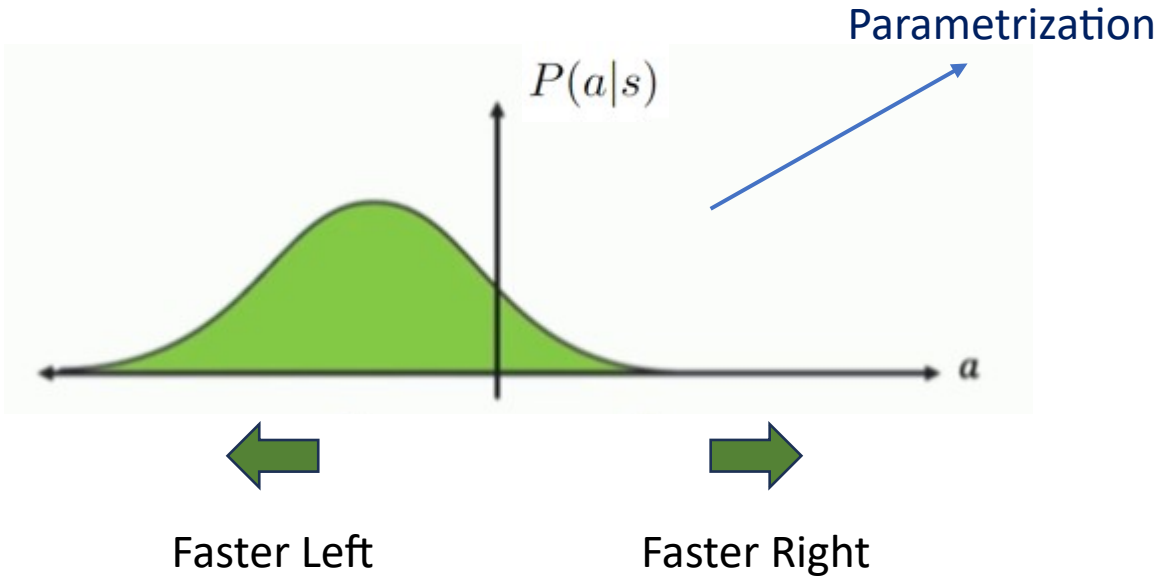




Discrete vs Continous Action Spaces

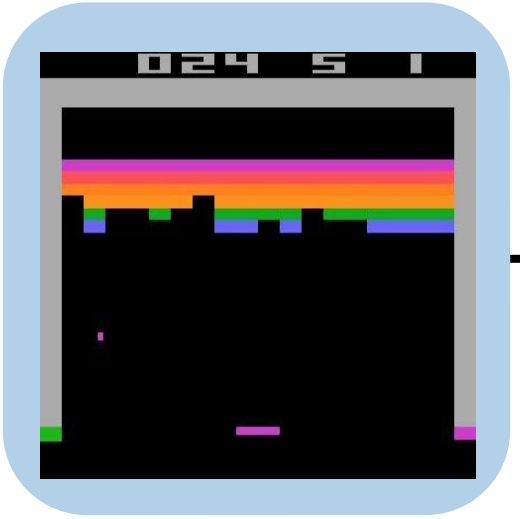
Discrete action space: which direction should I move?

Continous action space: how fast should I move?



Policy Gradient (PG): Key Idea

Policy Gradient: Enables modeling of continous action space



state, *s*

Deep NN

Mean, μ =-1

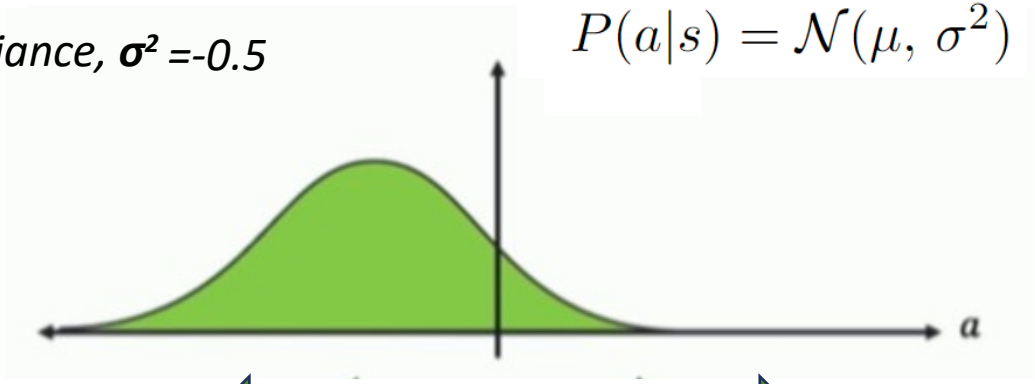
Variance, σ^2 =-0.5

$$P(a|s) = \mathcal{N}(\mu, \sigma^2)$$

$$\pi^*(s) \sim P(a|s)$$

$$= -0.8[m/s]$$

$$\int_{a=-\infty}^{\infty} P(a|s) = 1$$



Faster Left

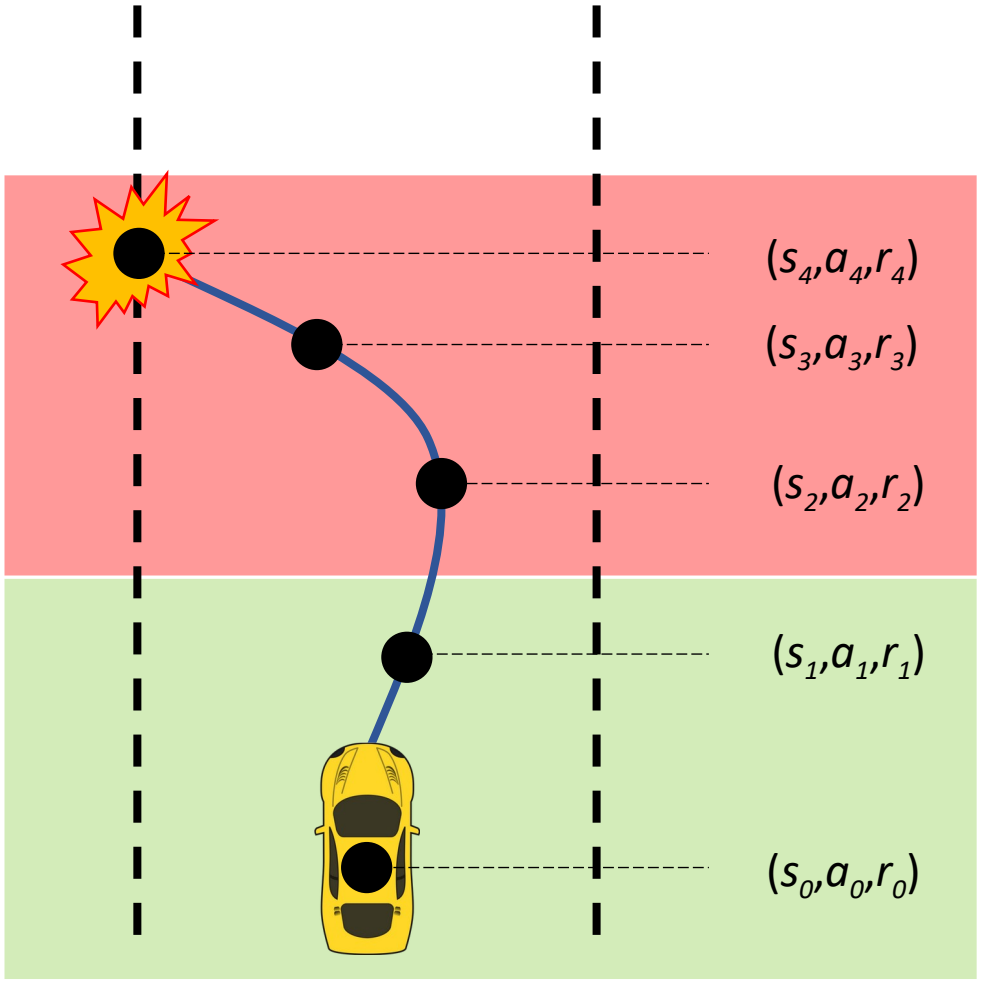
Faster Right

Trainig Policy Gradients

Training Algorithm

1. Initialize the agent
2. Run a policy until termination
3. Record all states, actions, rewards
4. Decrease probability of actions that resulted in low reward
5. Increase probability of actions that resulted in high reward

RUN AGAIN



Trainig Policy Gradients

Training Algorithm

1. Initialize the agent
2. Run a policy until termination
3. Record all states, actions, rewards
4. Decrease probability of actions that resulted in low reward
5. Increase probability of actions that resulted in high reward

Log-likelihood of action

$$\textbf{loss} = -\log P(a_t \mid s_t) R_t$$

reward

Gradient descent update

$$w' = w - \nabla \textbf{loss}$$

$$w' = w + \nabla \log P(a_t \mid s_t) R_t$$

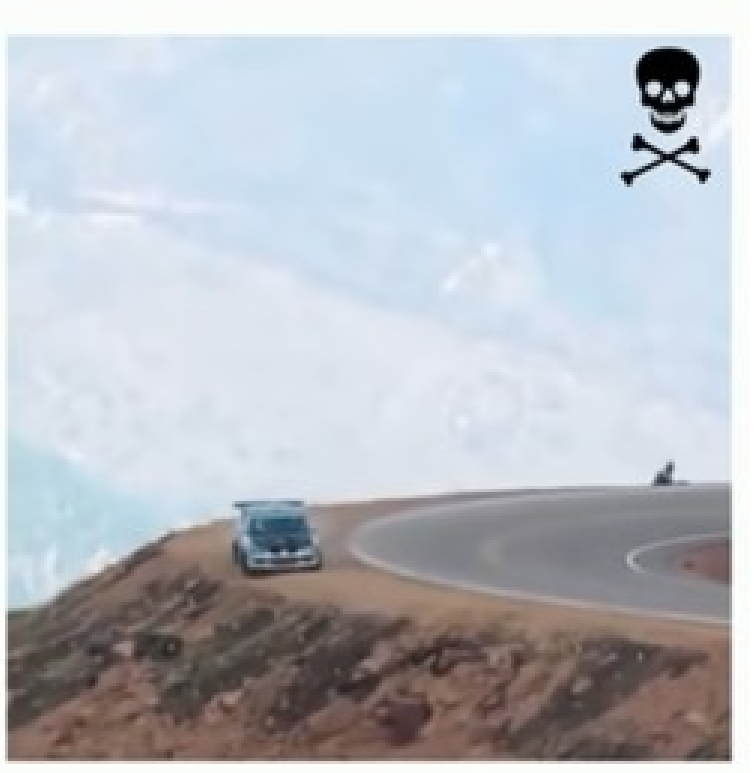
Policy gradinet

Reinforcment learning in real life

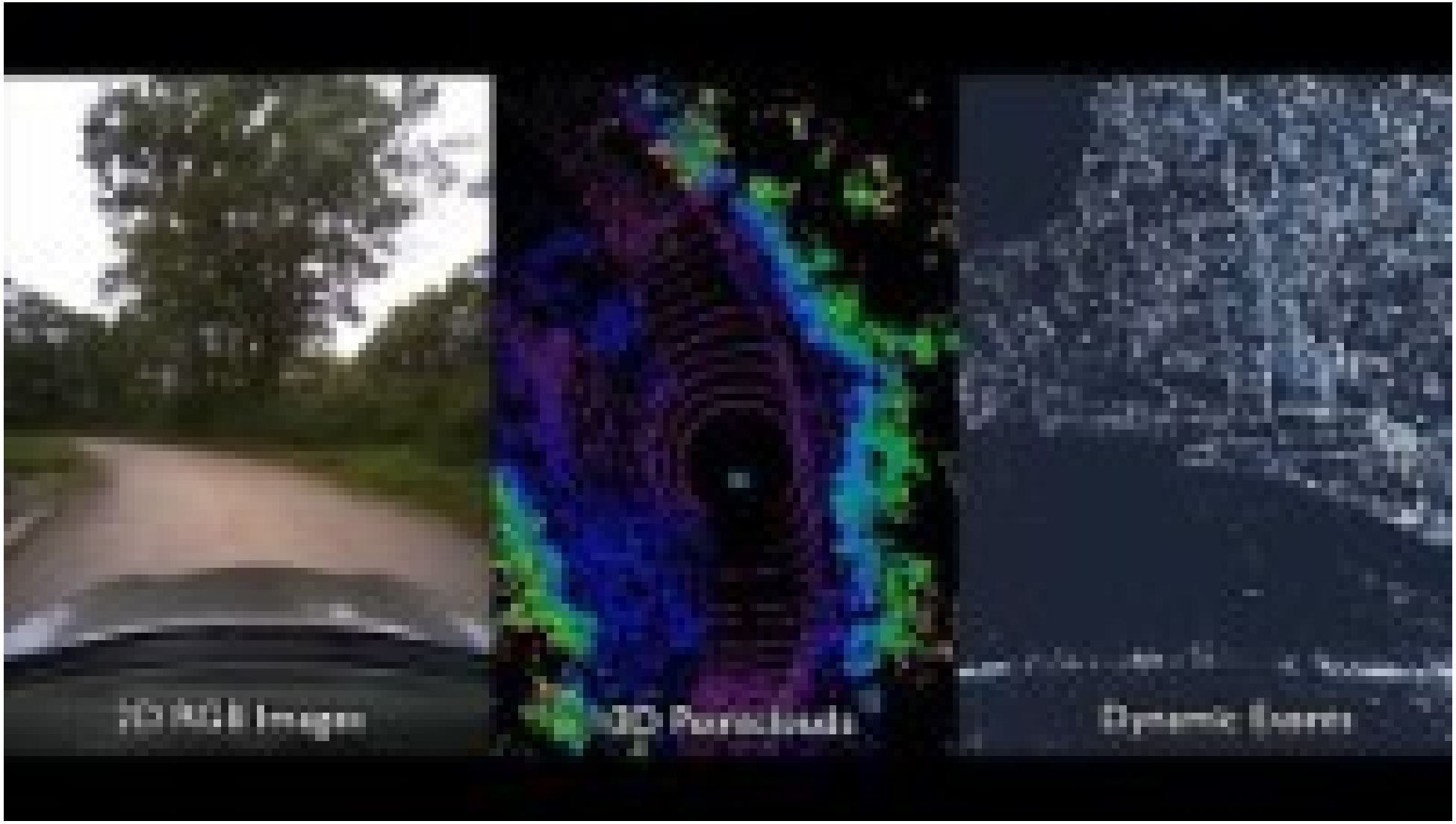
SIMULATION

Training Algorithm

1. Initialize the agent
2. Run a policy until termination
3. Record all states, actions, rewards
4. Decrease probability of actions that resulted in low reward
5. Increase probability of actions that resulted in high reward



VISTA – An open source simulator for self-driving cars



ELTE | FACULTY OF INFORMATICS

Thank you for your attention!