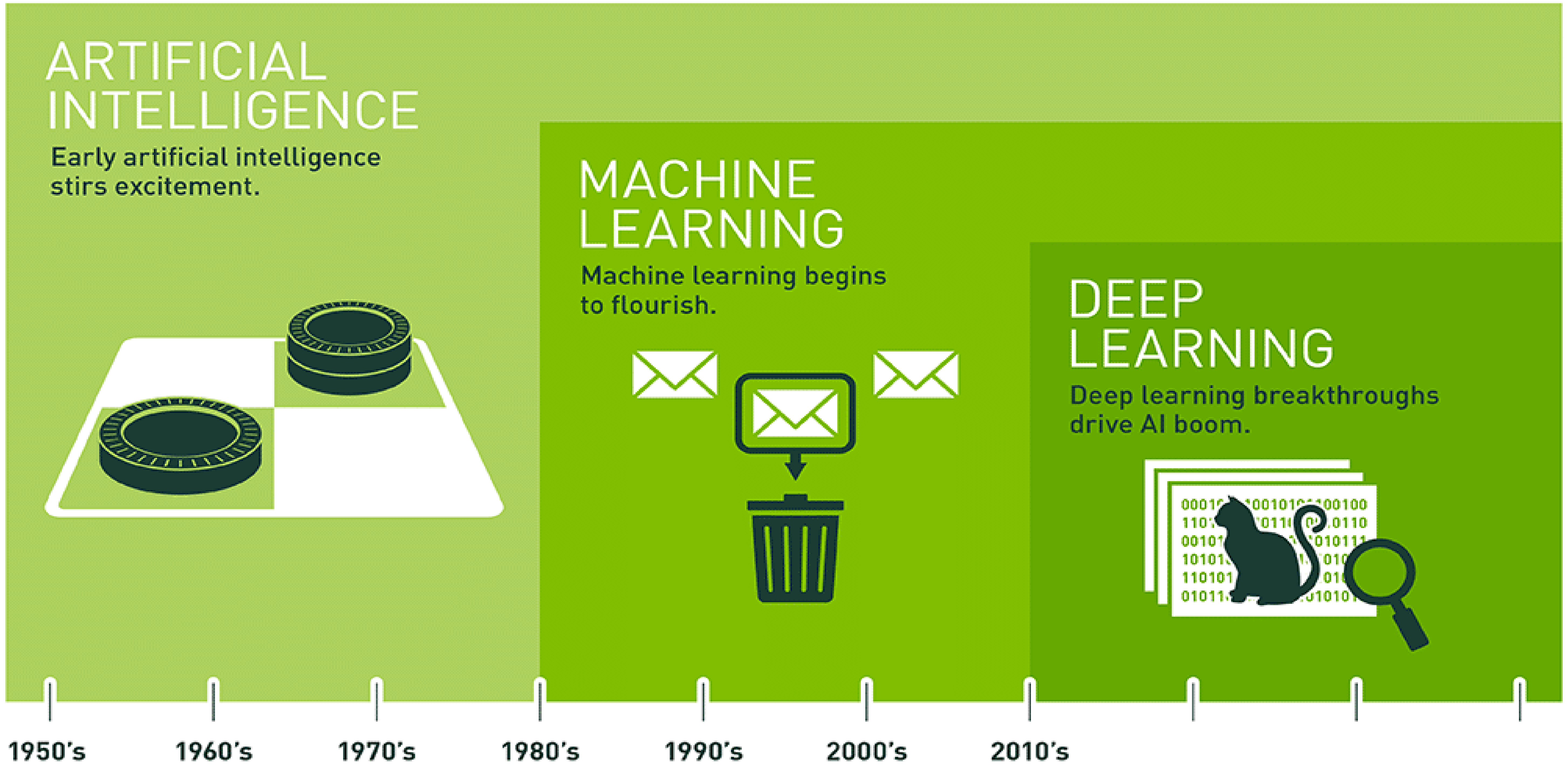
Martin Mundt, Dr. Iuliia Pliushch, Prof. Dr. Visvanathan Ramesh

Pattern Analysis & Machine Intelligence Praktikum: MLPR-SS21

Week 10: Introduction into Reinforcement Learning

AI, machine learning and NNs



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

https://blogs.nvidia.com/blog/2018/08/02/supervised-unsupervised-learning/

Different types of learning

TYPES OF MACHINE LEARNING

Supervised Learning

Train an algorithm to perform classifcation and regression with a labelled data set.

Unsupervised Learning

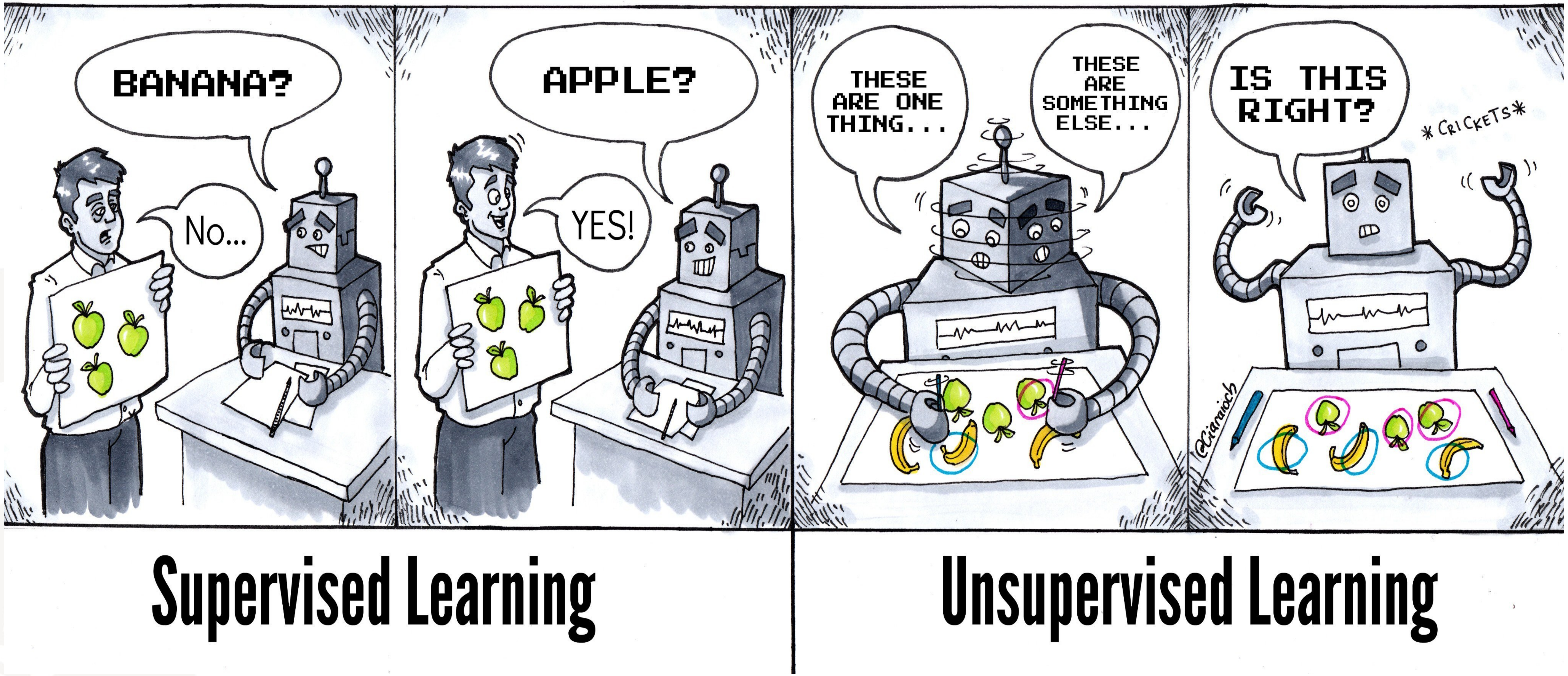
Train an algorithm to find clusters and associations in an unlabelled data set.

Reinforcement Learning

Train an agent to take certain actions in an environment without a data set.

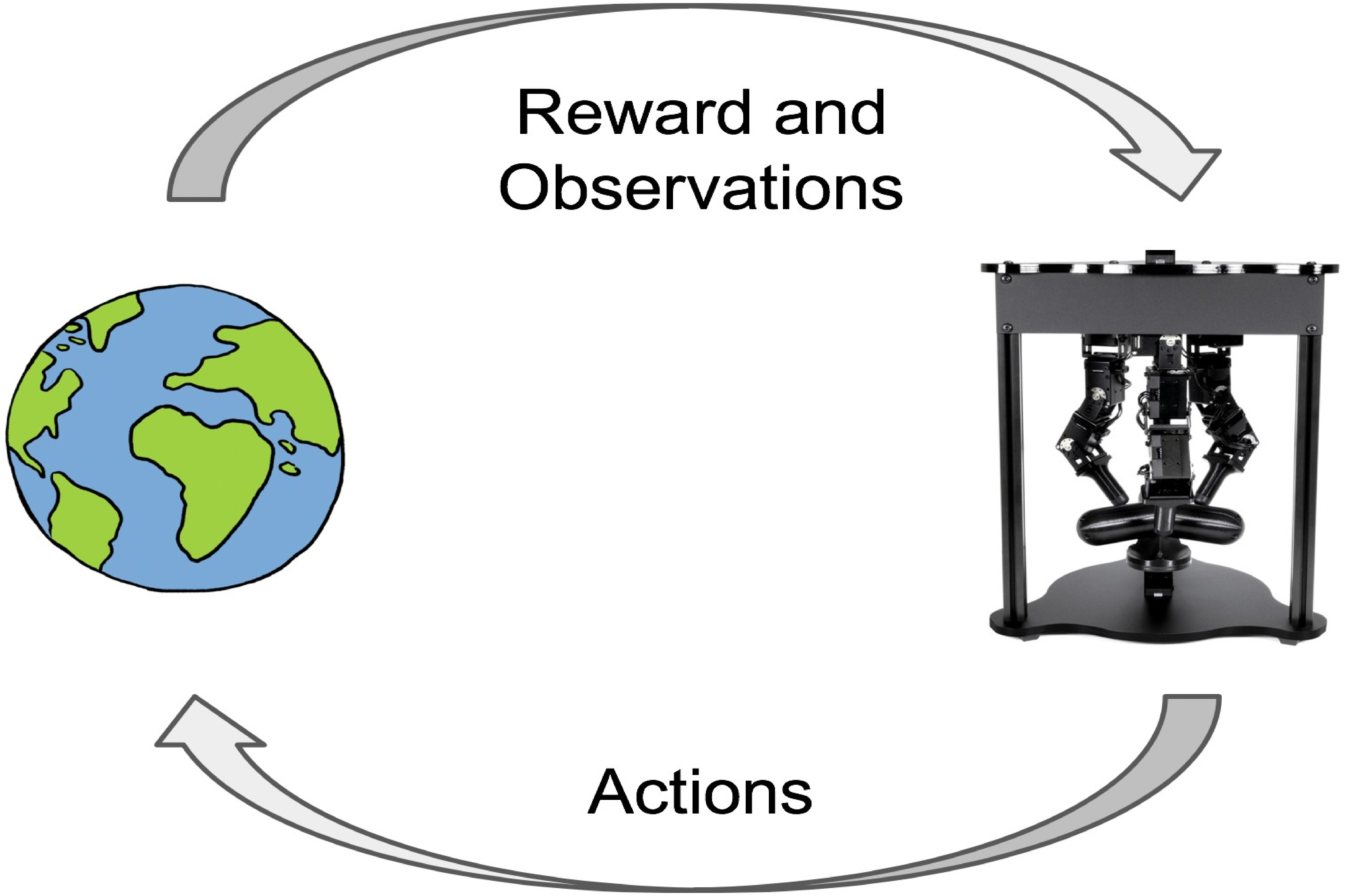
https://www.breakfreegraphics.com/design-blog/an-intro-to-machine-learning-for-Designers/

Supervised vs. unsupervised learning



https://twitter.com/athena_schools/status1063013435779223553photo/1

Reinforcement learning



https://aihub.org/2020/06/30/the-ingredients-of-real-world-robotic-reinforcement-learning/

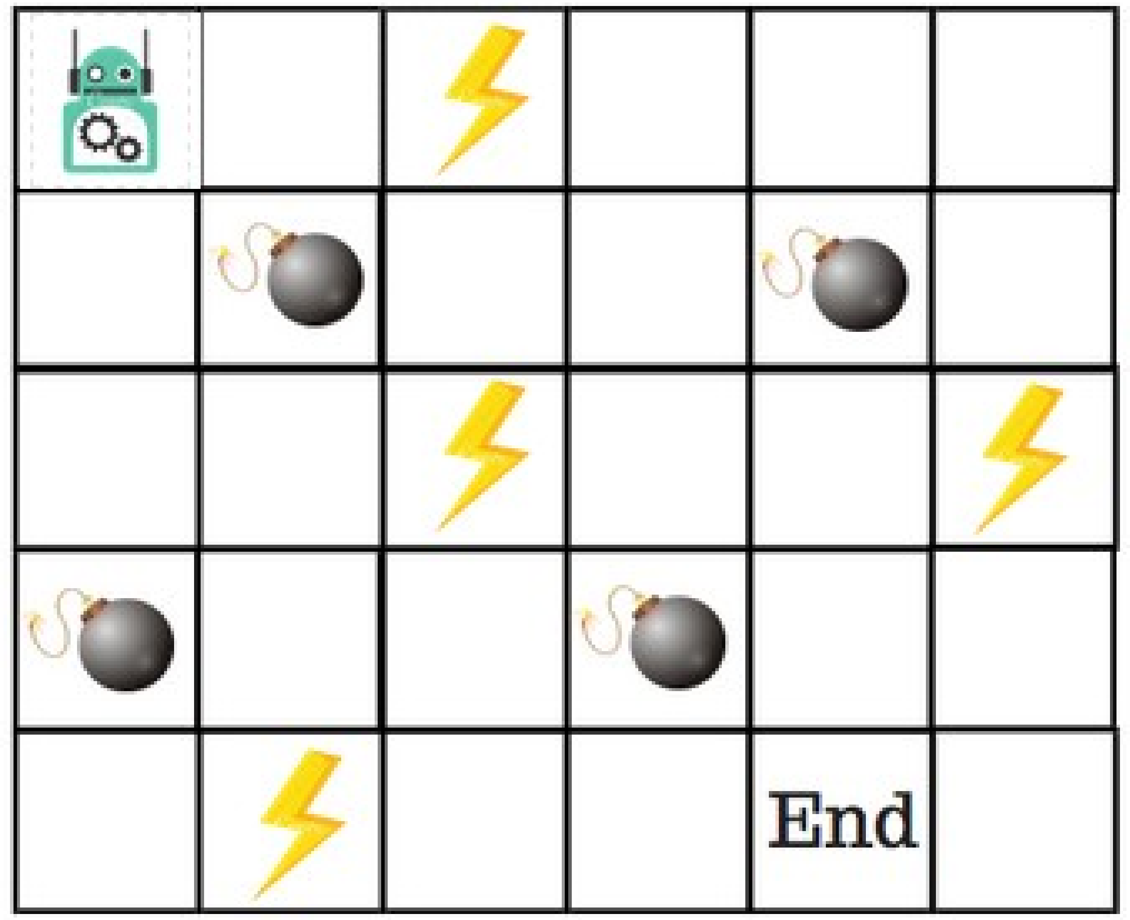
Reinforcement Learning: Applications

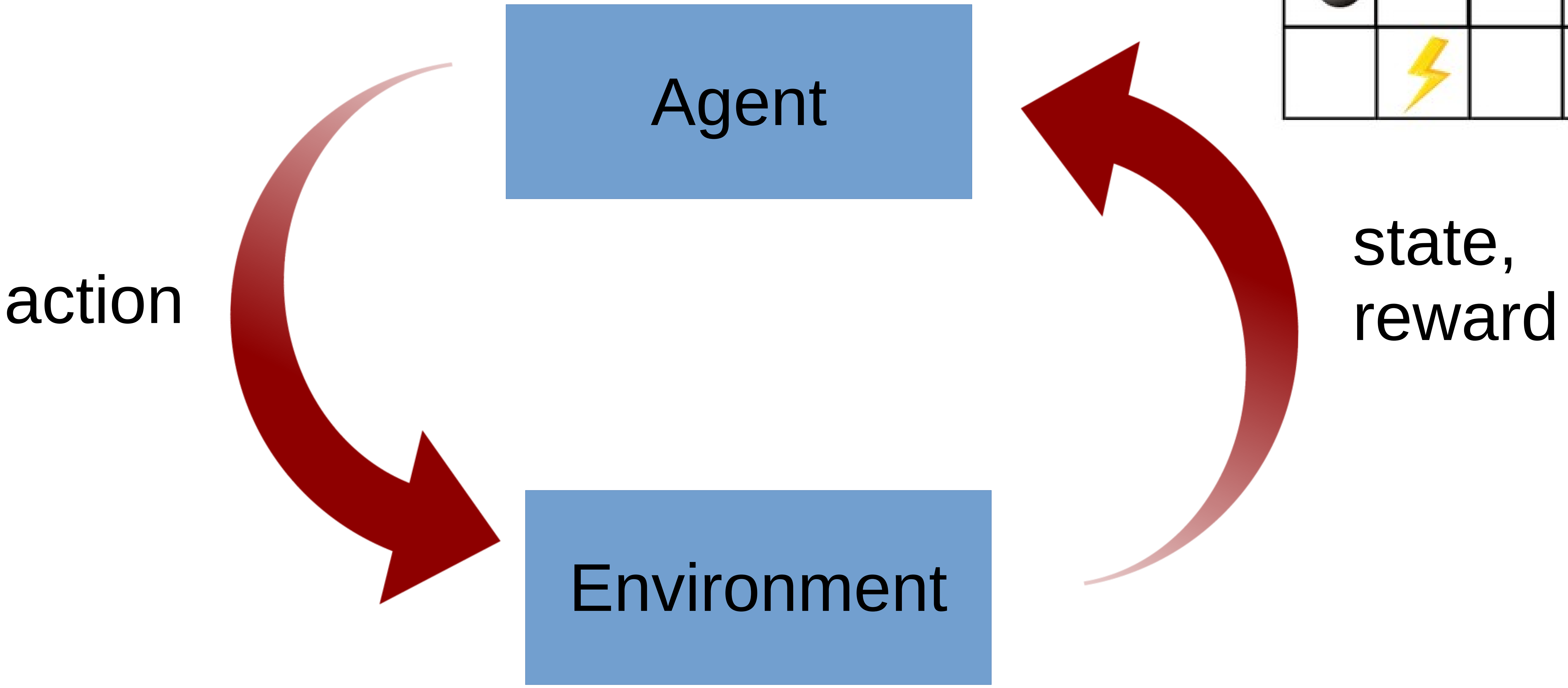
- Learning how to play games
- Robotics
- Finance
- Healthcare
- Meta-Learning

https://ai.googleblog.com/2018/06/scalable-deep-reinforcement-learning.html

Reinforcement Learning: Details

- Learning to maximize **rewards** by performing **actions** in an **environment**.

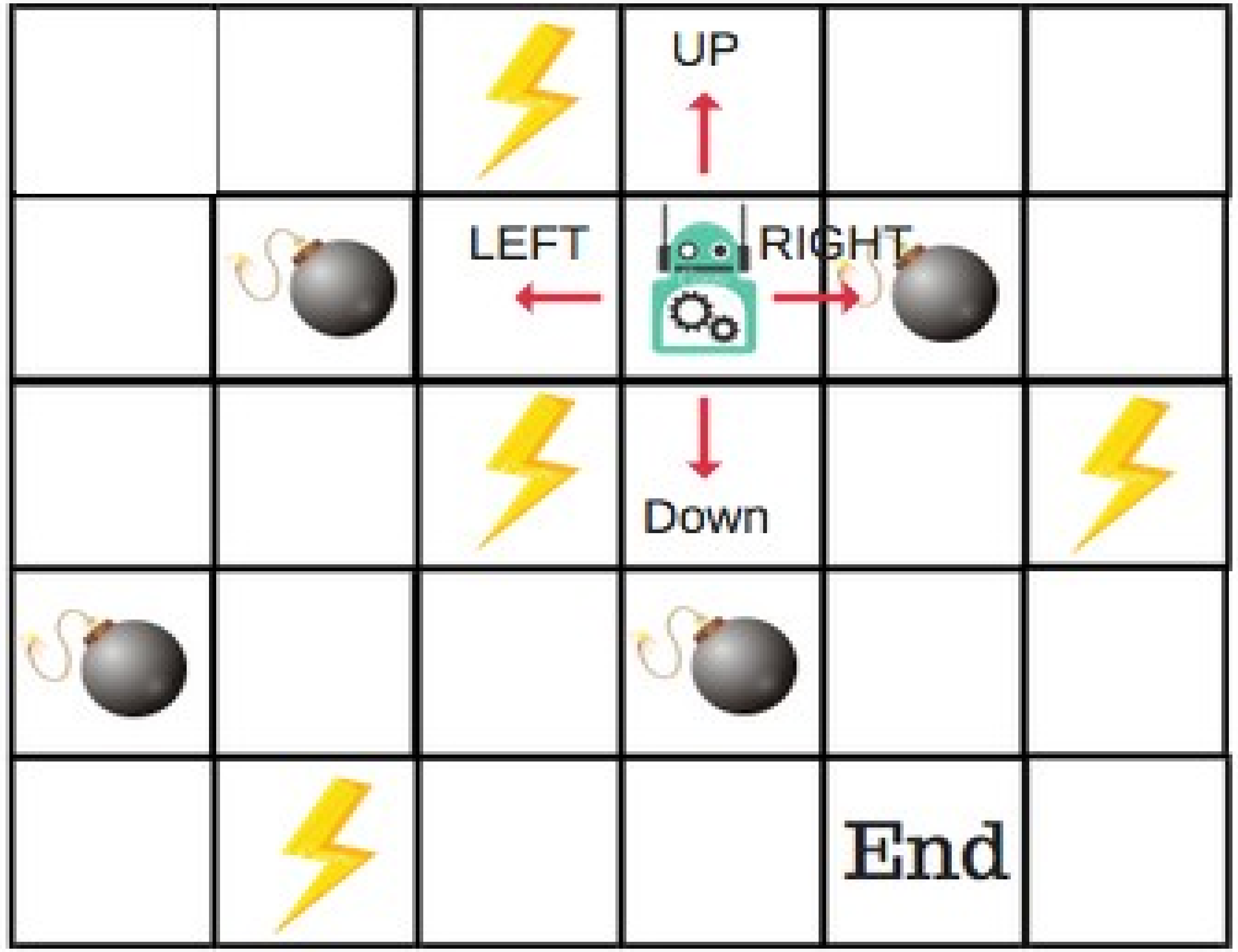




https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

Reinforcement Learning: Details

- **Environment**: maze
- **Actions**: left/right/up/down
- **Rewards**:
  - -1 on each step,
  - -100 to step on mine
  - 1 for lightning charge
  - 100 for end
- **Policy**: mapping from states to actions, e.g.
  - always go left until wall, then right
  - after stepping on mine, always go right+down



https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

Reinforcement Learning: Details

- **Q-Table**:

  a table storing the expected rewards for every (state, action)-pair

Actions :	↑	→	↓	←
Start				
Nothing / Blank				
Power				
Mines				
END				

https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

Reinforcement Learning: Details

Scenario:

- a learning **agent**
- **S**: a set of possible states
- **A**: a set of possible actions
- a ***state transition*** function

$$\delta : S \times A \rightarrow S$$

- a ***reward*** function

$$r : S \times A \rightarrow \mathbb{R}$$

Feedback loop:

- the agent repeatedly chooses an action according to some ***policy***

$$\pi : S \rightarrow A$$

- the environment changes to a new state according to δ
- some states provide the agent with feedback (***reinforcement***)

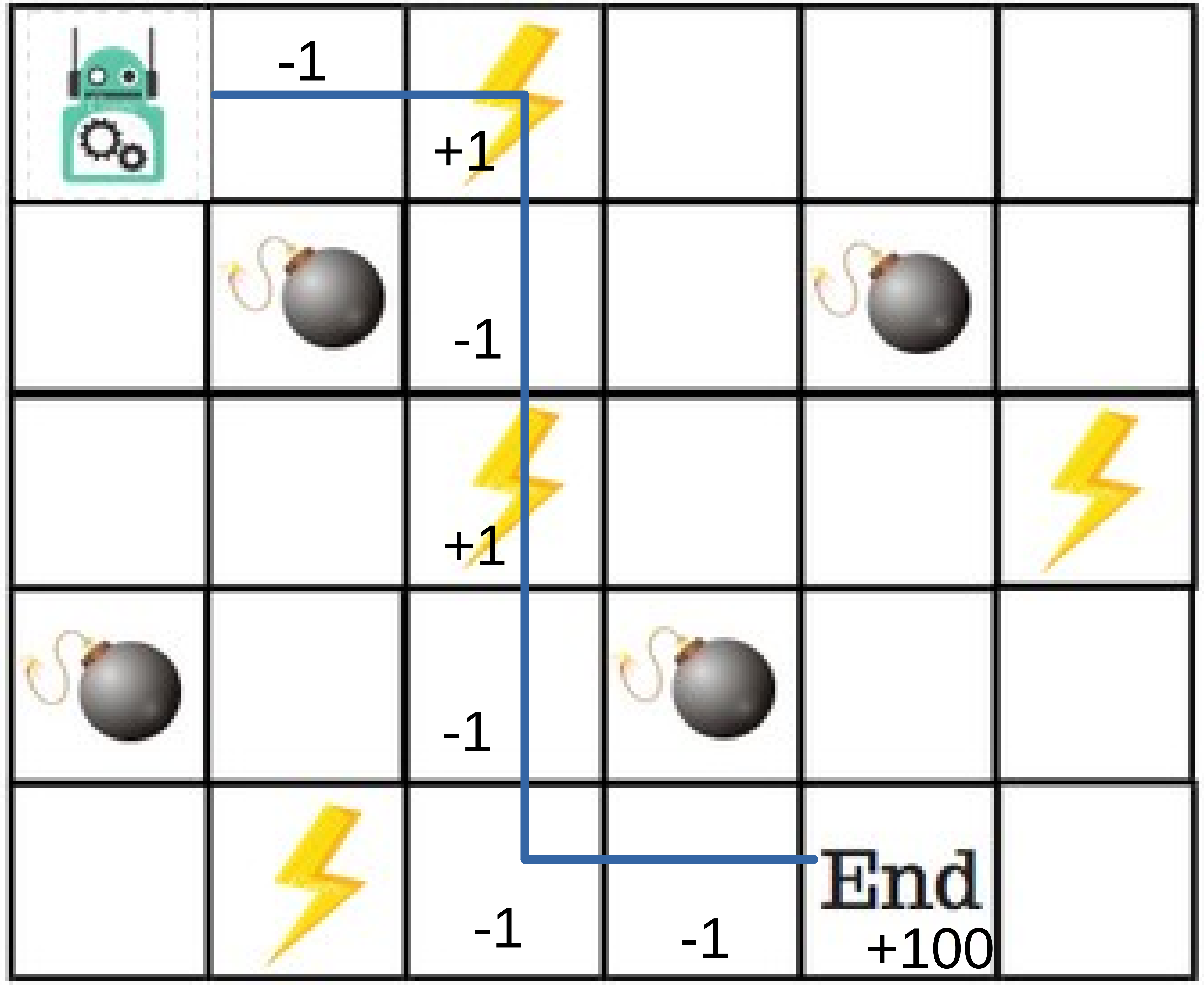
https://www.ke.tu-darmstadt.de/lehre/archiv/ss09/ki/reinforcement-learning.pdf

Reinforcement Learning: Reward

- Cumulative expected reward:

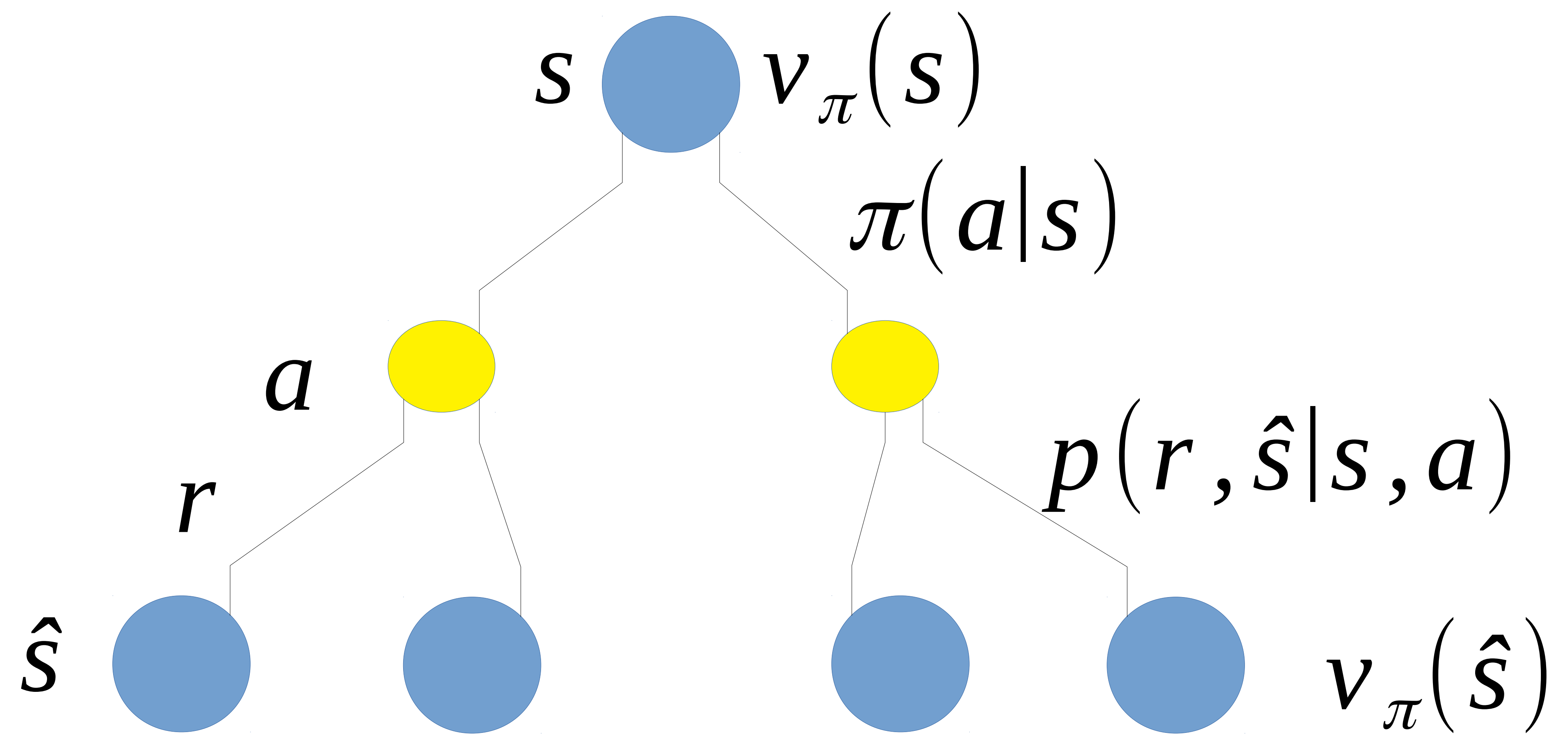
$$G_t = \sum_{i=0}^{\infty} \gamma^i * r_{t+i}$$

(γ makes the G_t finite)



https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

Reinforcement Learning: Optimal Policy



- Cumulative expected reward:

$$G_t = \sum_{i=0}^{\infty} \gamma^i * r_{t+i}$$

(γ makes the G_t finite)

- Bellman **expectation** for the state-value function:

$$v_\pi(s) = E[G_t | S_t = s] = E_\pi[R_t + \gamma * G_{t+1} | S_t = s]$$
$$= \underbrace{\sum_a \pi(a|s)}_{policy\ stochasticity} \underbrace{\sum_{r,\hat{s}} p(r, \hat{s}|s,a)}_{environment\ stochasticity} * [r + \gamma * \underbrace{E_\pi[G_{t+1}|S_{t+1} = \hat{s}]}_{v_\pi(\hat{s})}]$$

https://www.coursera.org/learn/practical-rl/home/welcome

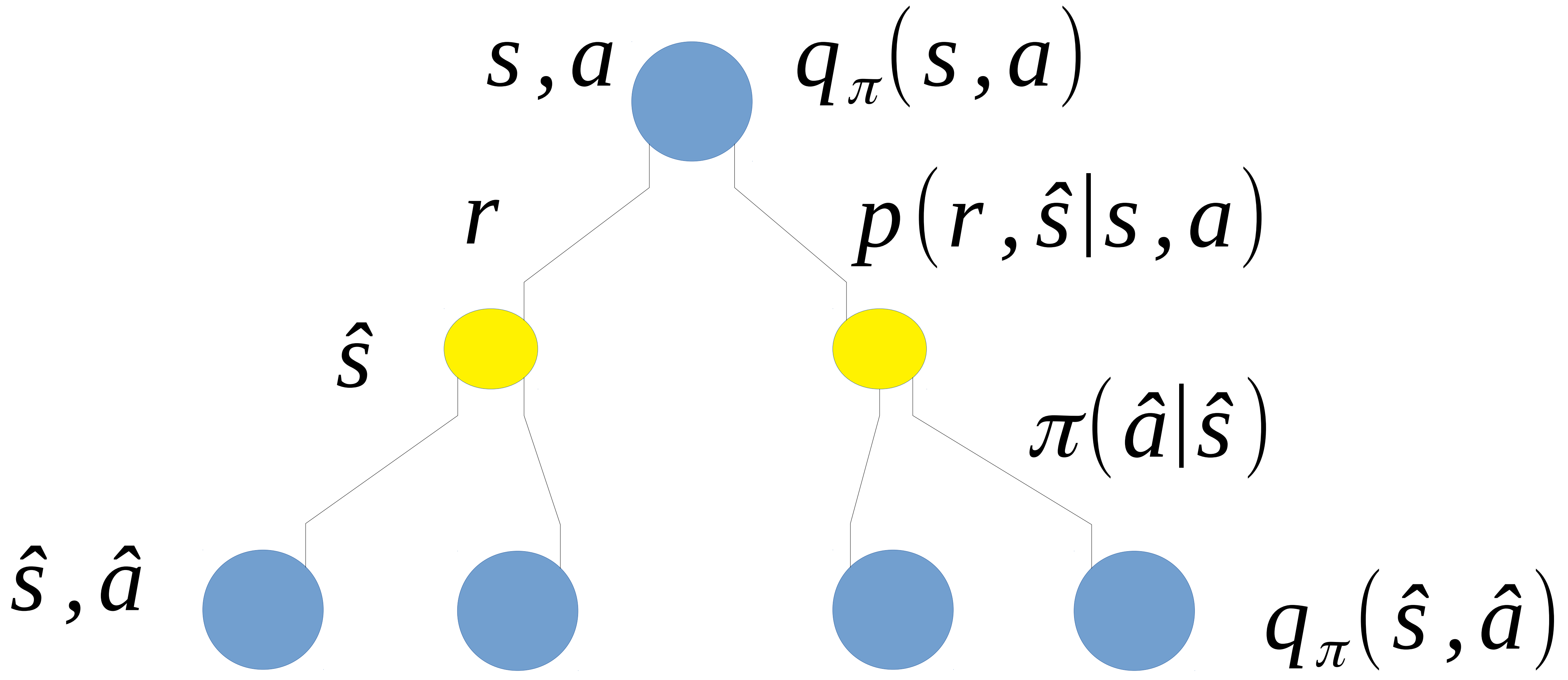
https://www.ke.tu-darmstadt.de/lehre/archiv/ss09/ki/reinforcement-learning.pdf

Reinforcement Learning: Optimal Policy

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

- State-value to action-value function

$$v_\pi(s) = \sum_a \pi(a|s)\, q_\pi(s,a)$$



- Bellman **expectation** for the action-value function:

$$q_\pi(s,a) = E[G_t|S_t=s, A_t=a] = E_\pi[R_t + \gamma * G_{t+1}|S_t=s, A_t=a]$$
$$= \underbrace{\sum_{r,\hat{s}} p(r,\hat{s}|s,a)}_{environment\ stochasticity} * [r + \gamma * \underbrace{E_\pi[G_{t+1}|S_{t+1}=\hat{s}]}_{v_\pi(\hat{s})}]$$

https://www.coursera.org/learn/practical-rl/home/welcome

Reinforcement Learning: Optimal Policy

$$v_{opt}(s) = max_{\pi} v_{\pi}(s)$$
$$\pi_{opt} = arg\,max_{\pi} v_{\pi}(s)$$

$$q_{opt}(s,a) = max_{\pi} q_{\pi}(s,a)$$
$$\pi_{opt}(s) = arg\,max_{a} q_{\pi}(s,a)$$

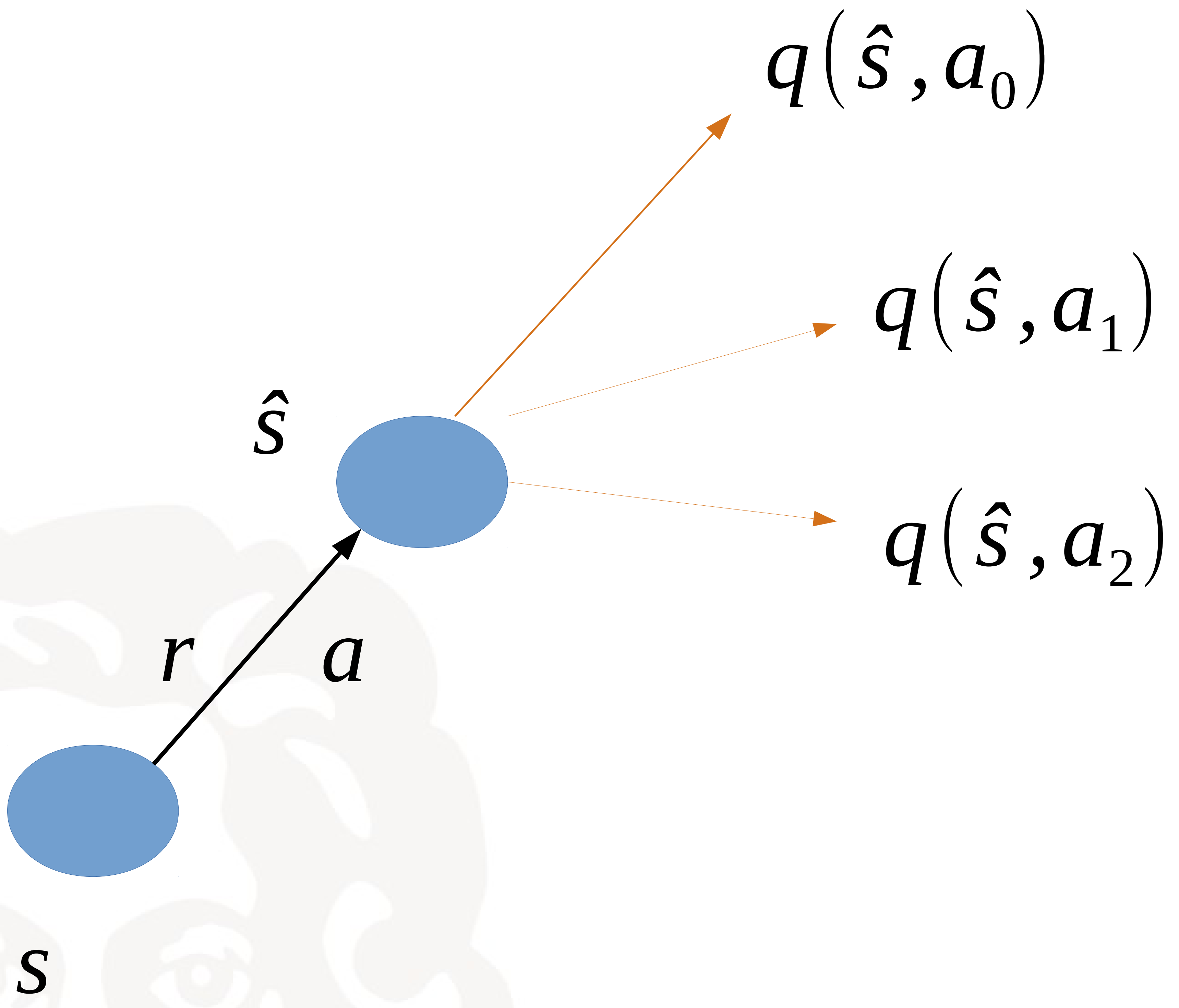
Bellman **optimality** equations:

$$v_{opt}(s) = max_a \underbrace{\sum_{r,\hat{s}} p(r,\hat{s}|s,a)}_{environment\ stochasticity} * [r + \gamma * v_{opt}(\hat{s})]$$

$$q_{opt}(s,a) = \underbrace{\sum_{r,\hat{s}} p(r,\hat{s}|s,a)}_{environment\ stochasticity} * [r + \gamma * max_{\hat{a}} q_{opt}(\hat{s},\hat{a})]$$

https://www.coursera.org/learn/practical-rl/home/welcome

Reinforcement Learning: Q-Learning



Model-free (train on trajectories),
Off-policy (not train on own policy)

$$\forall s \in S, \forall a \in A, q(s,a) = 0$$

$Loop:$

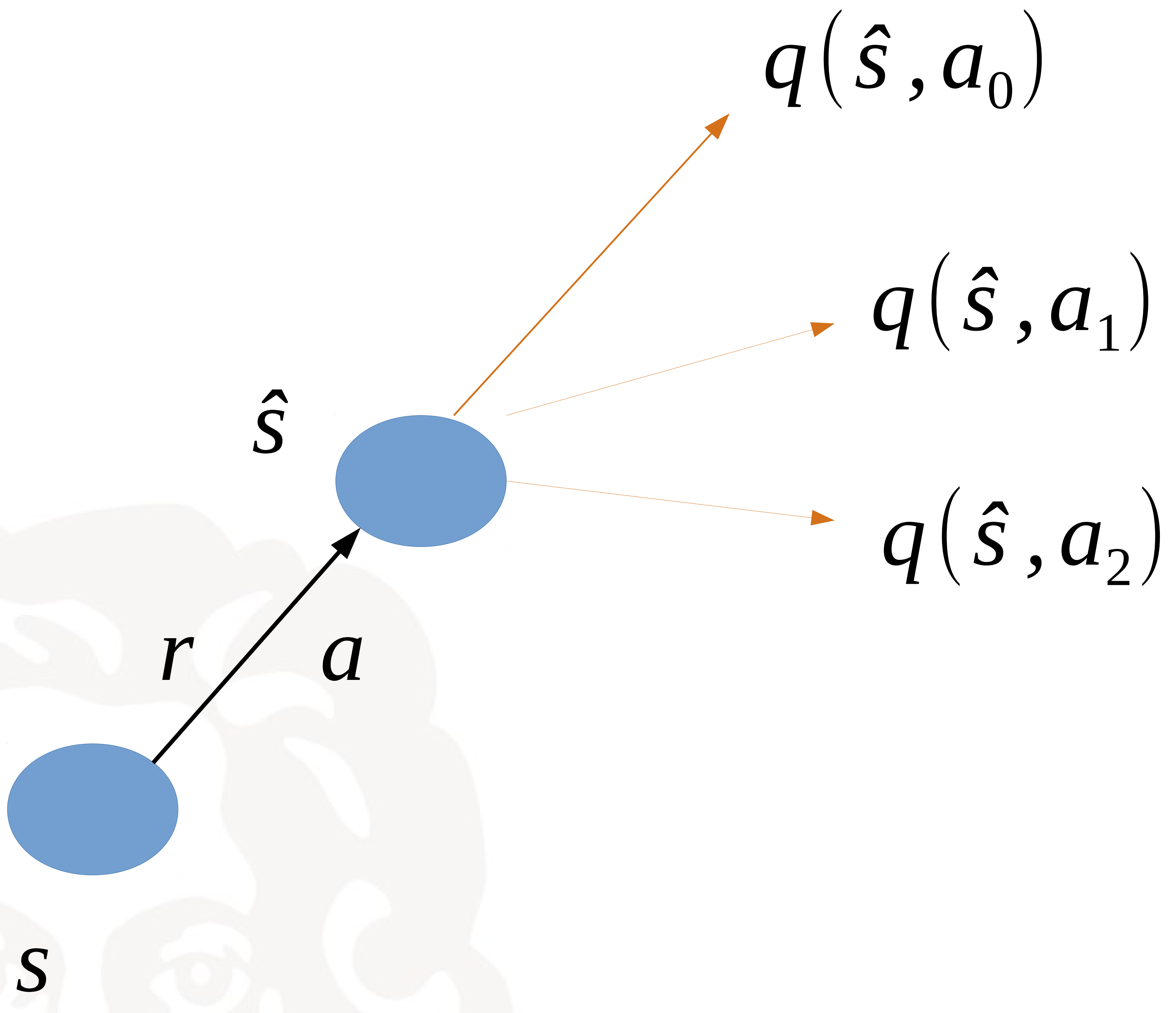
$Sample < s, a, r, \hat{s} >$

$Compute\, \tilde{q}(s,a) = r(s,a) + \gamma * max_{a_i} q(\hat{s}, a_i)$

$Update\, q(s,a) = \alpha * \tilde{q}(s,a) + (1 - \alpha) * q(s,a)$

http://icaps18.icaps-conference.org/fileadmin/alg/conferences/icaps18/summerschool/lectures/Lecture5-rl-intro.pdf
https://www.coursera.org/learn/practical-rl/home/welcome

Reinforcement Learning: Q-Learning



How to sample ŝ?

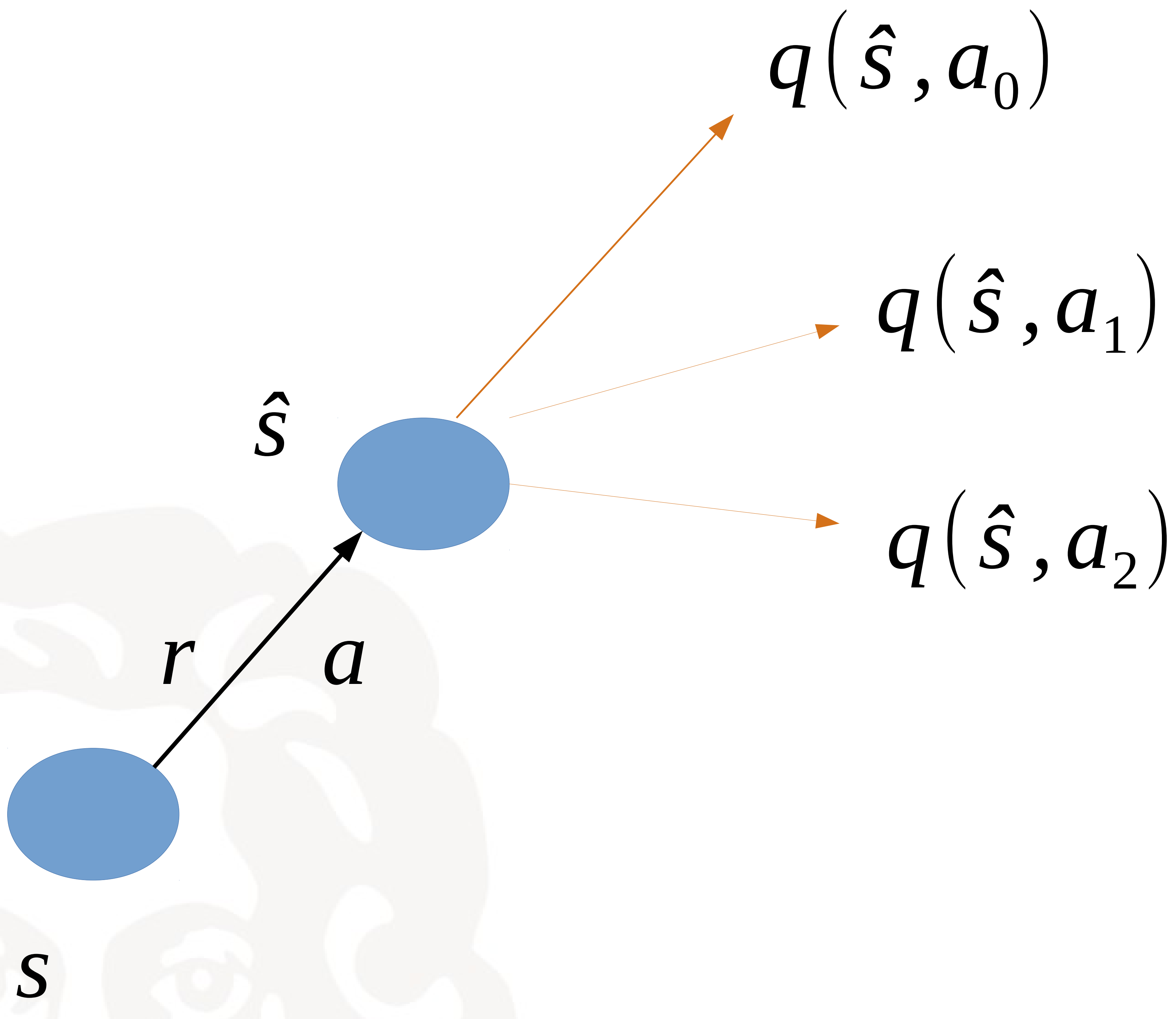
$\epsilon - greedy\ policy$

***Exploration-exploitation trade-off*:**

With probability ϵ choose a **random** action, else the **best** one.

https://www.coursera.org/learn/practical-rl/home/welcome

Reinforcement Learning: Openai Gym



https://gym.openai.com/envs/Taxi-v2/

https://www.coursera.org/learn/practical-rl/home/welcome