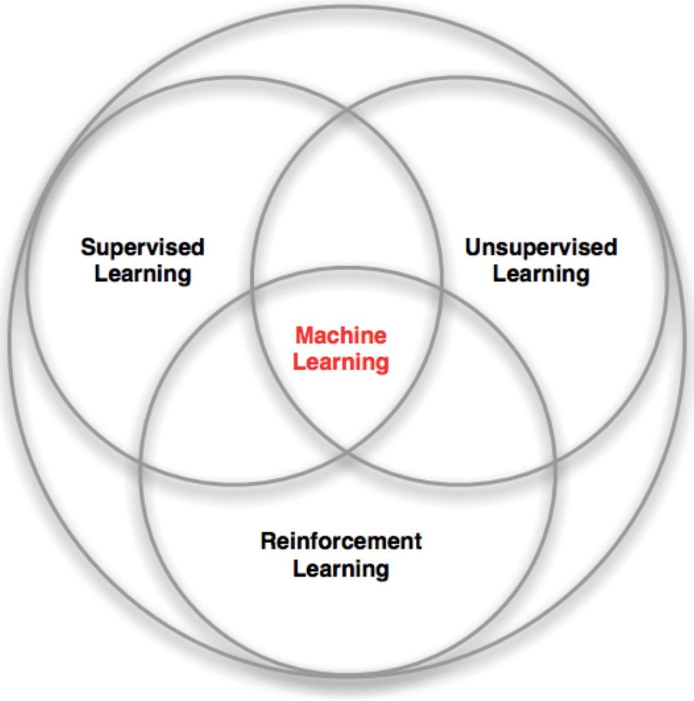
Reinforcement Learning: An Introduction

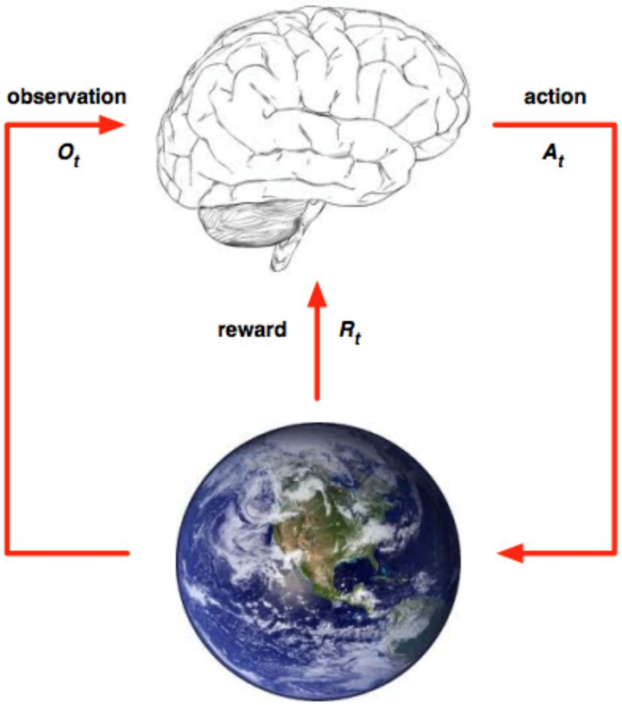
Milos Jordanski, PhD student at Faculty of Mathematics

Machine Learning



Agent and Environment

- At each time step **t** the agent:
  - Executes action A_t
  - Receives observation O_t
  - Receives scalar reward R_t
- At each time step **t** the environment:
  - Receives action A_t
  - Emits observation O_t
  - Emits scalar reward R_t



Comparison with other ML paradigms

- There is no supervisor, only a reward signal
- Feedback is delayed, not instantaneous
- Sequential data, not independent and identically distributed
- Agent’s actions affect the subsequent data it receives

Learning from interaction

- The agent is not told what to do so it must discover the best behavior
- The actions that it takes affect future outcomes
- It has to learn to map its current position to actions

Examples of Reinforcement Learning

- Fly stunt manoeuvres in a helicopter
- Defeat the world champion at Backgammon / Go / Chess
- Manage an investment portfolio
- Make a humanoid robot walk
- Play Atari games better than humans

Rewards

- A reward R_t is a scalar feedback signal
- Reward indicates how well the agent is doing
- The agent’s goal is to maximize cumulative reward
- All goals in RL can be described by maximizing cumulative reward

Examples of rewards

- Defeat the world champion at Backgammon / Go
  - + reward for winning
  - – reward for losing
- Manage an investment portfolio
  - + reward for each $ in bank
- Make a humanoid robot walk
  - + reward for forward motion
  - – reward for falling over
- Play Atari games
  - + reward for increasing the score
  - – reward for decreasing the score

Sequential Decision Making

- Goal: section sequence of actions to maximize total cumulative reward
- Reward may be delayed
- Actions may have long term consequences
- It may be better to sacrifice immediate reward to gain more long-term rewards

Fully Observable Environments

- Agent observes environment state
- A state S_t is Markov if and only if:

$$P[S_{t+1} | S_t] = P[S_{t+1} | S_1, S_2, \ldots, S_t]$$

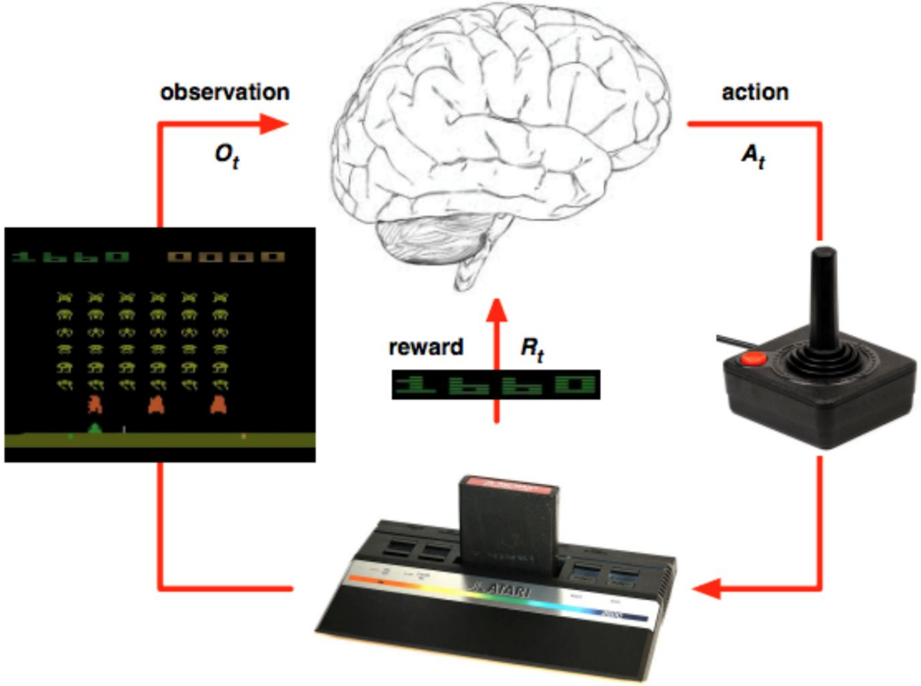
- The future is independent of the past given the present
- The state is sufficient statistics of the future

Learning and Planning

- Learning:
  - The environment is initially unknown
  - The agent interacts with the environment
  - The agent improves its policy
- Planning:
  - The model of environment is known
  - The agent performs computations with its model (reasoning, thought, search)
  - The agent improves its policy

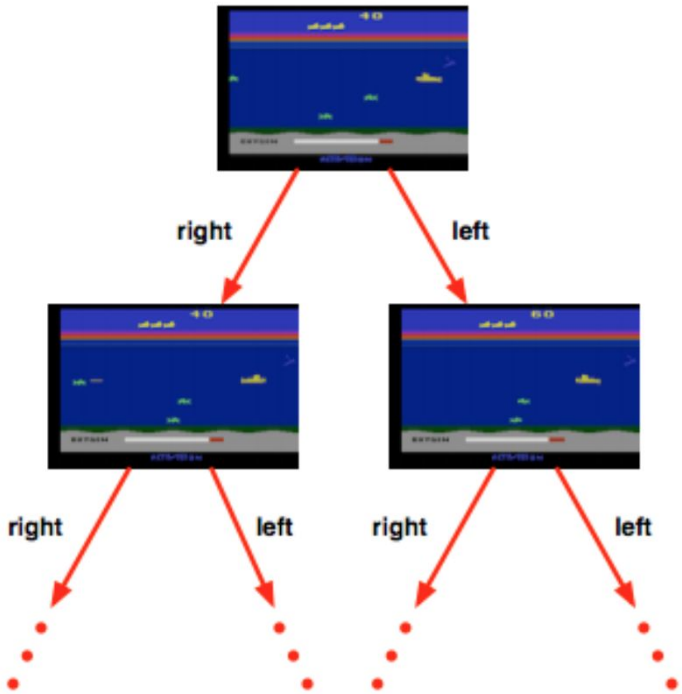
Atari example: Learning

- Rules of the game are unknown
- Learn directly from interaction with environment
- Pick actions on joystick, see observations (pixels) and scores



Atari example: Planning

- Rules of the game are known
- If the agent takes actions ***a*** from state ***s***:
  - What would be the next state?
  - What would the score be?
- Plan ahead to find the optimal policy



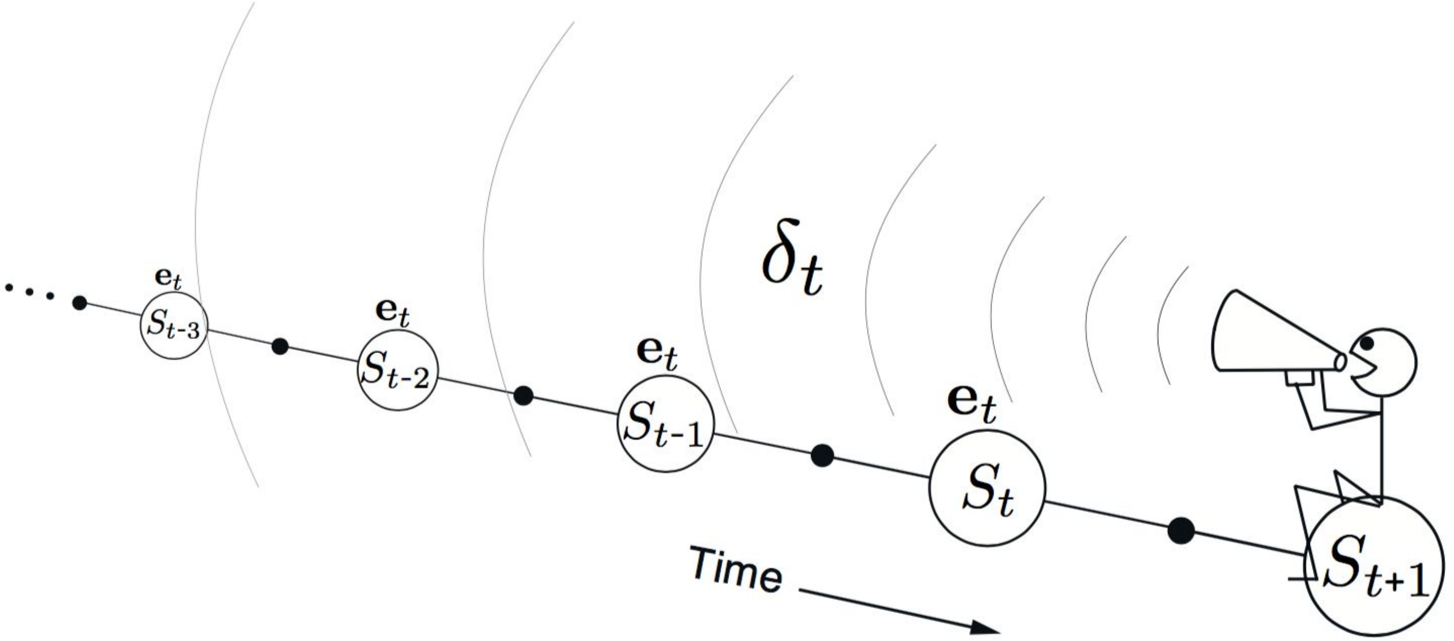
Exploration and Exploitation

- Exploration finds more information about the environment
- Exploitation exploits known information to maximize immediate reward
- It is import to explore as well as to exploit

Exploration and Exploitation: Examples

- Restaurant Selection
  - Exploitation: Go to your favorite restaurant
  - Exploration: Try a new restaurant
- Online Banner Advertisements
  - Exploitation: Show the most successful advert
  - Exploration: Show a different advert
- Game Playing
  - Exploitation: Play the move you believe is the best
  - Exploration: Play an experimental move

Credit Assignment



Markov Decision Process (MDP)

- Markov Decision Process is a tuple <S, A, P, R, γ>
  - S is a finite set of states
  - A is a set of actions (continue or discrete)
  - P is a state transition probability matrix (Markov property)

$$P_{ss'}^{a} = P[S_{t+1} = s | S_t = s, A_t = a]$$

  - R is a reward function

$$R_s^a = E[R_{t+1} | S_t = s, A_t = a]$$

  - $\gamma \in [0, 1]$ is a discount factor

Return

- The return G_t is the total reward from time step t:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- The discount factor $\gamma \in [0, 1]$ is the present value of future rewards
  - γ close to 0 leads to “myoptic” evaluation
  - γ close to 1 leads to “far-sighted” evaluation
- Uncertainty about the future may not be fully represented
- It is mathematically convenient to discount rewards
- Avoids infinite returns in cyclic Markov processes

Policy

- A policy π is a distribution over actions given states
  - Deterministic policy: a = π(s)
  - Stochastic policy: $\pi(a|s) = P[A_t = a \mid S_t = s]$
- A policy fully defines the behaviour of an agent
- MDP policies depend on the current state
- Policies are stationary (time - independent)

$$A_t \sim \pi(\cdot|S_t), \forall t > 0$$

Value Function

- The state-value function v_π(s) of an MDP is the expected return starting from state s, and then following policy π:

$$v_\pi(s) = E_\pi[G_t | S_t = s]$$

- The action-value function q_π(s, a) is the expected reward starting from state s, taking action a, and then following policy π:

$$q_\pi(s, a) = E_\pi[G_t | S_t = s, A_t = a]$$

Categorizing RL agent

- Value based:
  - No policy (implicit)
  - Value function
- **Policy based:**
  - Policy
  - No value function
- Actor-Critic:
  - Policy
  - Value function

Categorizing RL agent

- **Model Free:**
  - Policy and / or Value function
  - No model of environment
- Model Based:
  - Policy and / or Value function
  - Model the environment

Policy Gradient

- Model-free reinforcement learning
- Direct optimization of the policy:

$$\pi_\theta(s, a) = P[a|s, \theta]$$

- Advantages:
  - Better convergences properties
  - Effective in high-dimensional and continuous action spaces
  - Learning stochastic policies
- Disadvantages:
  - Converges to local optimum
  - High variance in evaluating a policy

Policy Objective Functions

- How to measure the quality of a policy: $\pi_\theta(s, a)$
- Start value:

$$J_1(\theta) = V^{\pi_\theta}(s_1) = E_{\pi_\theta}[v_1]$$

- Average value:

$$J_{av\_V}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

- Average reward per time-step:

$$J_{av\_R}(\theta) = \sum_s d^{\pi_\theta} \sum_a \pi_\theta(s, a) R_s^a$$

Policy Optimization

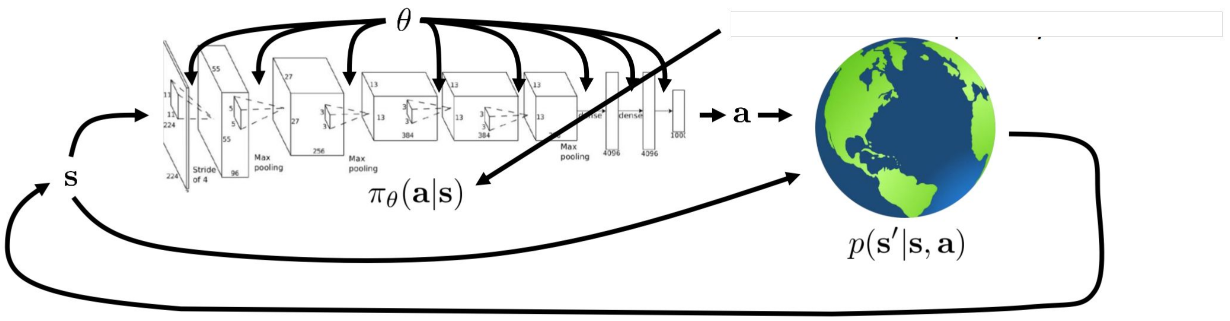
- Policy based Reinforcement Learning is an optimization problem
- Find θ that maximizes J(θ)
- Any optimization algorithm could be applied
- Gradient based optimization algorithms

Policy Optimization

$$J(\theta) = E_{\tau \sim \pi_\theta}[r(\tau)]$$

$$\theta^* = argmax_\theta E_{\tau \sim \pi_\theta}[\sum_t r(s_t, a_t)]$$

$$J(\theta) = E_{\tau \sim \pi_\theta}[\sum_\tau r(s_t, a_t)] \approx \frac{1}{N} \sum_i \sum_t r(s_{it} a_{it})$$



Score function

$$J(\theta) = E_{\tau \sim \pi_\theta}[r(\tau)] = \int \pi_\theta(\tau) r(\tau) d\tau$$

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta \nabla_\theta \, log \pi_\theta(\tau) r(\tau) d\tau$$

$$= E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta log \pi_\theta(\tau) r(\tau)]$$

Likelihood ratio trick:

$$\nabla_\theta \pi_\theta(\tau) = \pi_\theta(\tau) \frac{\nabla_\theta \pi_\theta(\tau)}{\pi_\theta(\tau)}$$

$$= \pi_\theta(\tau) \nabla_\theta \, log \pi_\theta(\tau)$$

Softmax policy

- Probability of action is proportional to exponentiated weight

$$\pi_\theta(s,a) \propto e^{\phi(s,a)^T\theta}$$

- The score function is

$$\nabla_\theta log\pi_\theta(s,a) = \phi(s,a) - E_{\pi_\theta}[\phi(s,\cdot)]$$

Gaussian Policy

- In continuous action spaces
- Mean is a linear combination of state features: $\mu(s) = \phi(s)^T\theta$
- Variance can be fixed or can also be parametrized
- Policy is Gaussian:

$$a \sim N(\mu(s), \sigma^2)$$

- Score function:
-

$$\nabla_\theta log\pi_\theta(s,a) = \frac{(a-\mu(s))\phi(s)}{\sigma^2}$$

REINFORCE Algorithm

- Replace instantaneous reward r with long-term value
- Use return as unbiased estimate of action-value function
- Initialize $\boldsymbol{\theta}$
- For each episode $\{s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_T, a_T, r_T\}$
  - For each t = 1 to T-1

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log\pi_\theta(s_t, a_t) \sum_t r(s_t, a_t)$$