Introduction to Reinforcement Learning

Syllabus

Foundations: Basics of machine learning and reinforcement learning (RL) terminology.
Probability Concepts: Axioms of probability, random variables, distributions, and correlation.
Markov Decision Process: Introduction to MDPs, Markov property, and Bellman equations.
State and Action Value Functions: Concepts of MDP, state, and action value functions.
Tabular Methods and Q-networks: Dynamic programming, Monte Carlo, TD learning, and deep Q-networks.
Policy Optimization: Policy-based methods, REINFORCE algorithm, and actor-critic methods.
Recent Advances and Applications: Meta-learning, multi-agent RL, ethics in RL, and real-world applications.

Suggested Readings

- Reinforcement Learning: An Introduction by Richard S. Sutton and Andrew G. Barto, The MIT Press (1 January 1998).
- Deep Reinforcement Learning Hands-On by Maxim Lapan, Packt Publishing Limited (21 June 2018).
- Algorithms for Reinforcement Learning by Csaba Szepesvari, Morgan and Claypool Publishers (2010)
- Deep Reinforcement Learning: Fundamentals, Research and Applications by Hao Dong, Springer Verlag (2020)

Learn to Control

- Familiar models of Machine Learning
  - Supervised: Classification, Regression etc.
  - Unsupervised: Clustering, Frequent Patterns etc.

- How did you learn the cycle?

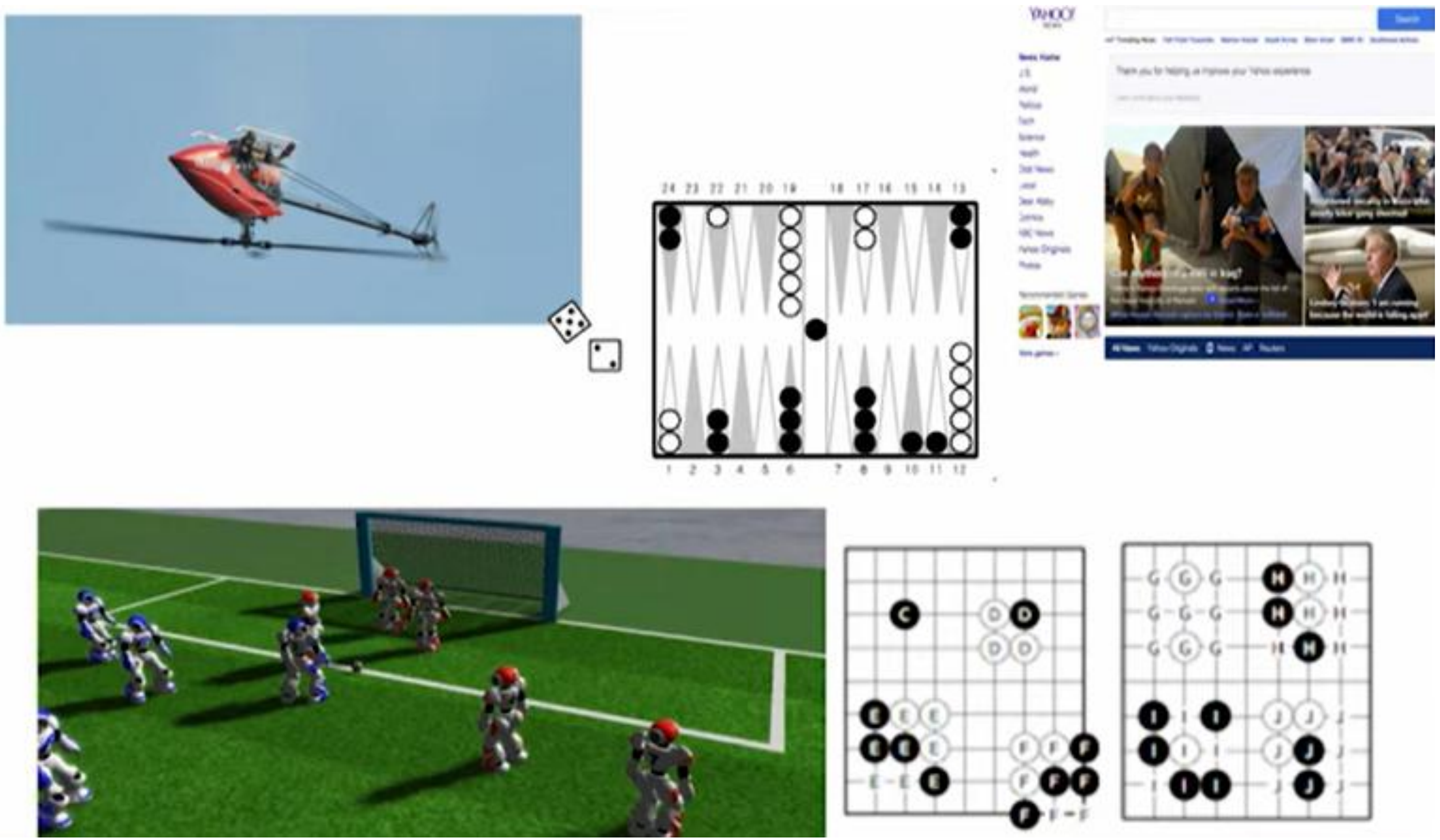
Reinforcement Learning

- A Trial-and-Error learning paradigm
- Learn about a system through interaction
- Inspired by behavioral psychology!
    - Pavlov’s dog

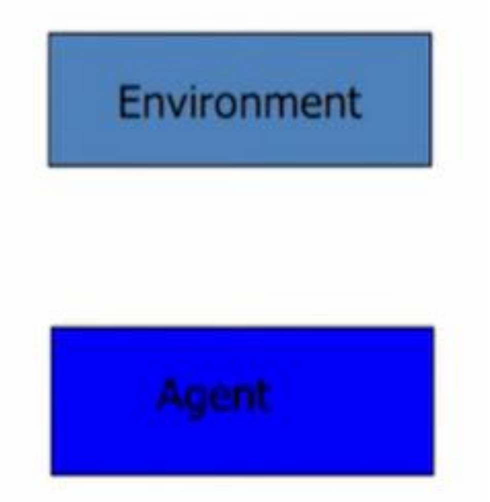
Reinforcement Learning

- Learning about stimuli and actions based on rewards and punishments alone.
- No detailed supervision available
- Trial-and-Error learning
- Delayed rewards
- Sequence of actions required to obtain reward
- Associative learning required
  - Need to associate actions to states
- Learn about policies not just actions
- Typically in a stochastic world

RL Applications

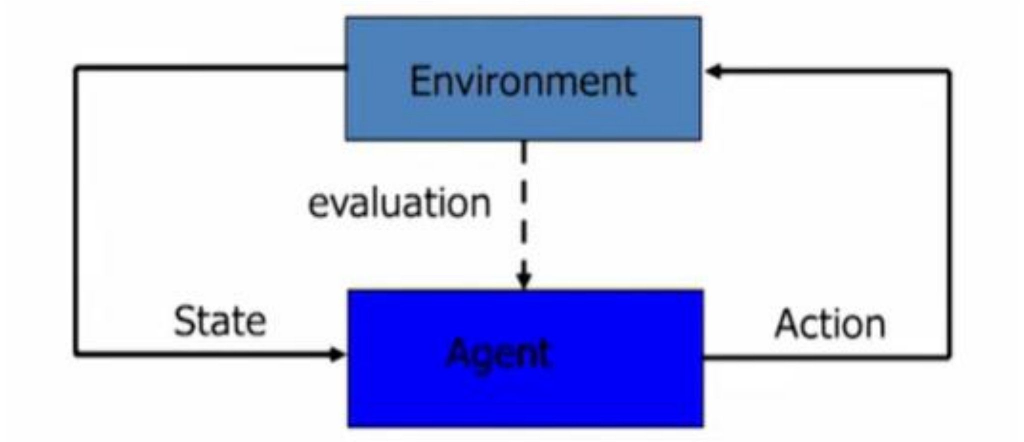


RL Framework



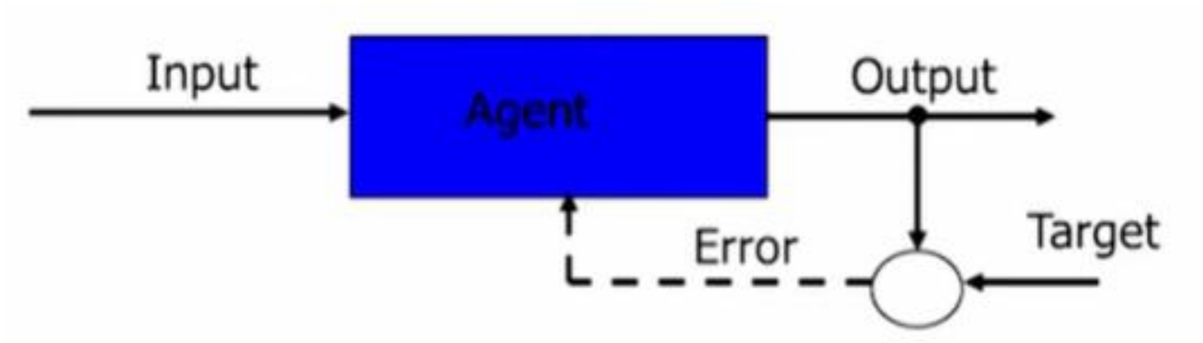
- Learn from close interaction
- Stochastic environment
- Noisy delayed scalar evaluation
- Maximize a measure of long term performance

RL Framework



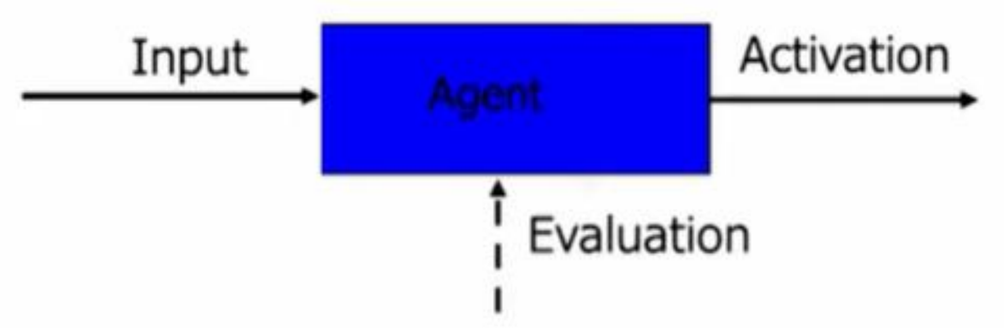
- Learn from close interaction
- Stochastic environment
- Noisy delayed scalar evaluation
- Maximize a measure of long term performance

Not Supervised Learning!



- Very sparse “Supervision”
- No target output provided
- No error gradient information available
- Action chooses next state
- Explore to estimate gradient – Trial and Error Learning

Not Unsupervised Learning!



- Sparse “Supervision” available
- Pattern detection not primary goal

RL Elements

- Apart from Agent and Environment, It has other crucial components:
  - Policy
  - Reward signals
  - Value functions
  - Model of environment

Policy

- Defines agent’s way of behaving at a given time
- In psychology, it is called a set of Stimulus-Response rules or associations
- May be a simple function or requires extensive computation

Reward Signals

- Defines the goal in a RL Problem
- Considered as a primary basis for altering the policy
- In general, reward signals may be stochastic functions of the state of the environment and the actions taken

Value function

- A value function defines what is good in the long run
- Value of state = total amount of reward an agent can except to accumulate over the future

Model of Environment

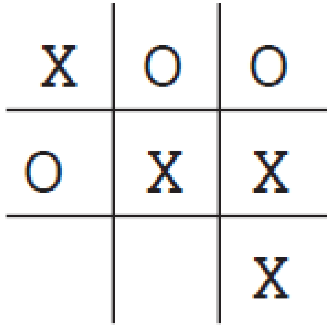
- Mimics the behavior of environment
- Model are used for *Planning*
- Methods for solving RL problems that use models and planning are called Model-based methods

Limitations and Scope

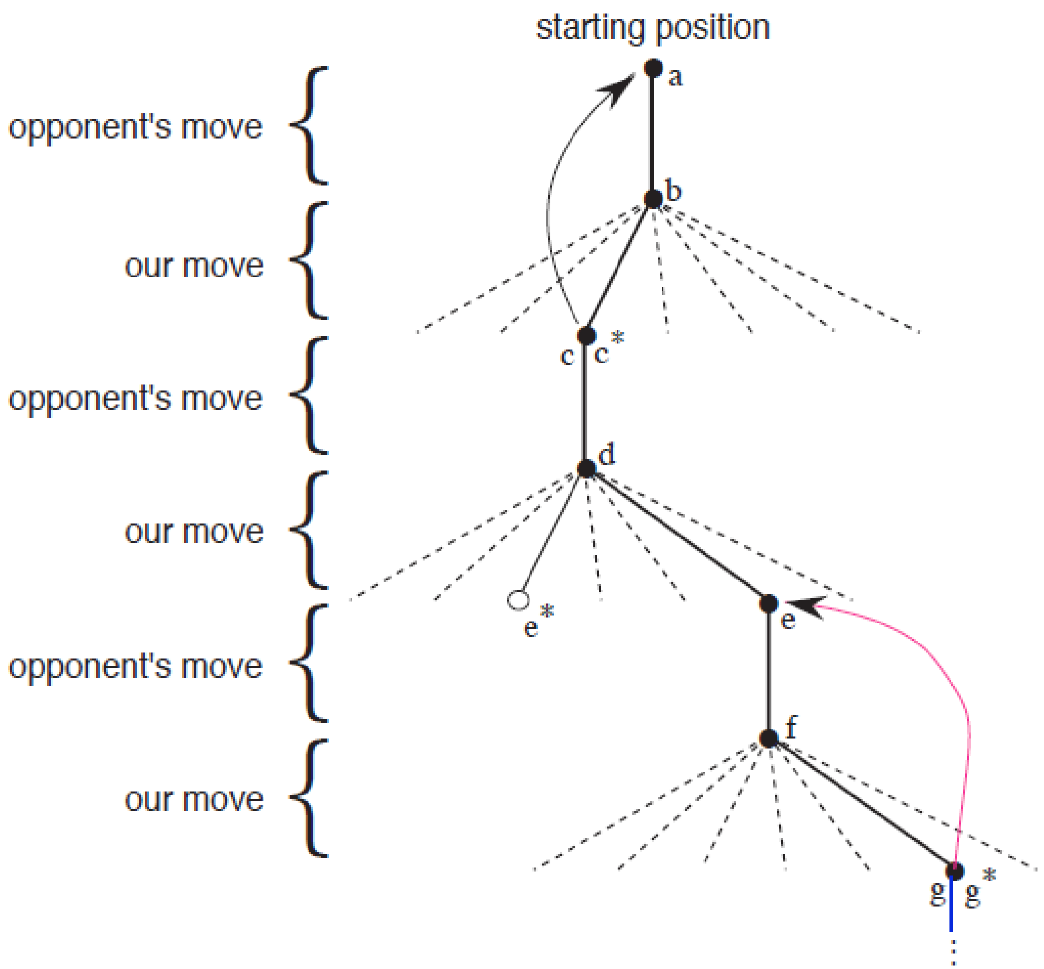
- Most of the RL methods are structured around estimating the Value Functions
- Evolutionary methods –
  - will be effective for sufficiently small space of policies, good policies are common and easy to find, or if a lot of time is available for searching.
  - Do not use the fact that policy they are searching for is a function from states to actions.
- Policy gradient methods

An Extended Example: Tic-Tac-Toe

- Solutions:
  - Classical Techniques
    - Minimax
    - Dynamic programming
  - Evolutionary Methods
  - RL Method



An Extended Example: Tic-Tac-Toe



An Extended Example: Tic-Tac-Toe

If we let s denote the state before the greedy move, and s' the state after the move, then the update to the estimated value of s, denoted $V(s)$, can be written as

$$V(s) \leftarrow V(s) + \alpha\Big[V(s') - V(s)\Big],$$

where α is a small positive fraction called the *step-size parameter*, which influences the rate of learning. This update rule is an example of a *temporal-difference* learning method, so called because its changes are based on a difference, $V(s') - V(s)$, between estimates at two different times.

Temporal Difference

- Simple rule to explain complex behaviours
- Intuition: Prediction of outcome at time *t+1* is better than the prediction at time *t*. Hence use the later prediction to adjust the earlier prediction.
- Has had profound impact in behavioral psychology and neuroscience!

Explore-Exploit Dilemma

- One key Question- the dilemma between exploration and exploitation
- Explore to find profitable actions
- Exploit to act according to the best observations already made
- Bandit problems encapsulate ‘Explore vs Exploit’