An Introduction to Reinforcement Learning

March 16, 2022

Overview

- What is Reinforcement Learning?
  - A Brief Definition
  - From the perspective of CS: ML v.s. RL
  - From the perspective of Econ/OR: DP v.s. RL
  - Practical Implementation

Overview

- What is Reinforcement Learning?
  - A Brief Definition
  - From the perspective of CS: ML v.s. RL
  - From the perspective of Econ/OR: DP v.s. RL
  - Practical Implementation
- Reinforcement Learning in Economics:
  - The Latest Literature
  - Sub-Field: Multi-Agent Reinforcement Learning

Overview

- What is Reinforcement Learning?
  - A Brief Definition
  - From the perspective of CS: ML v.s. RL
  - From the perspective of Econ/OR: DP v.s. RL
  - Practical Implementation
- Reinforcement Learning in Economics:
  - The Latest Literature
  - Sub-Field: Multi-Agent Reinforcement Learning
- Goal:
  - Realize that the "AI" is nothing mysterious (Is RL $\approx$ AI?)
  - arise interest to implement RL in your research

What is Reinforcement Learning?

- The first thought: Library Cafe v.s. Rootes

What is Reinforcement Learning?

- The first thought: Library Cafe v.s. Rootes
- The exploration-exploitation trade-off

What is Reinforcement Learning?

- The first thought: Library Cafe v.s. Rootes
- The exploration-exploitation trade-off
- Reinforcement Learning is about an Agent learns via interacting with an Environment
  - State of the Environment
  - Action taken by the Agent
  - Reward from the Action

What is Reinforcement Learning?

- The first thought: Library Cafe v.s. Rootes
- The exploration-exploitation trade-off
- Reinforcement Learning is about an Agent learns via interacting with an Environment
  - State of the Environment
  - Action taken by the Agent
  - Reward from the Action
- The "Literal Decomposition":
  - Learning: Optimal Policy
  - Reinforcement: Reward-Driven

What is RL: A Brief Definition

- Definition: A Markov decision process (MDP) is a 4-tuple (S, A, P_a, R_a), where:
  - S is a set of states called the state space
  - A is a set of actions called the action space
  - $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is the prob. that action a in state s at time t will lead to state s' at time $t + 1$
  - $R_a(s, s')$ is the immediate reward received after transitioning from state s to state s', due to action a

What is RL: A Brief Definition

- Definition: A Markov decision process (MDP) is a 4-tuple (S, A, P_a, R_a), where:
  - S is a set of states called the state space
  - A is a set of actions called the action space
  - $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is the prob. that action a in state s at time t will lead to state s' at time $t + 1$
  - $R_a(s, s')$ is the immediate reward received after transitioning from state s to state s', due to action a
- RL solves problems of MDPs:
  - agent observes state $s_t \in \mathcal{S}$, takes an action $a_t \in \mathcal{A}$ based on a policy $\pi \in \mathcal{S} \rightarrow \mathcal{A}$, the environment produces a reward r_t and moves to s_{t+1}
  - the goal is to find an optimal policy that obtaining accumulative rewards $\sum_{i=1}^{n} \gamma^t R_t$

What is Reinforcement Learning?

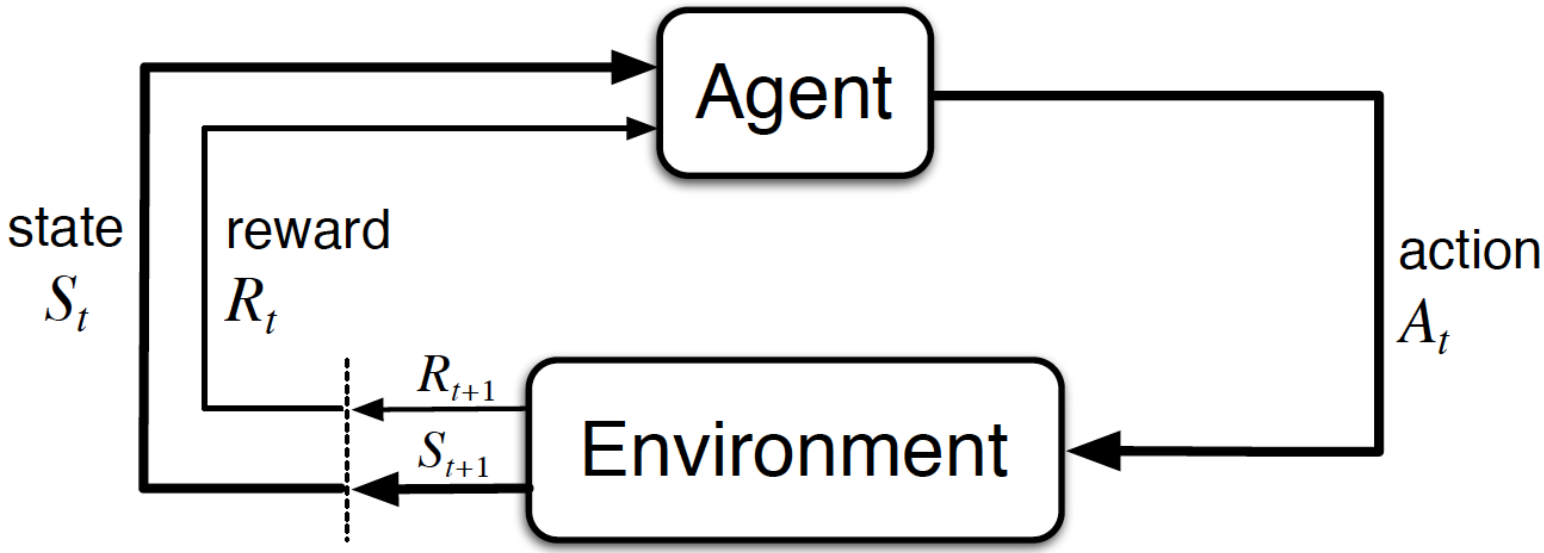


Figure: Agent-Envrionement Interaction by Sutton and Barto(2008)

What is RL: an Illustration

- State: current position
- Action: Up, Low, Left, Right
- Reward: ?

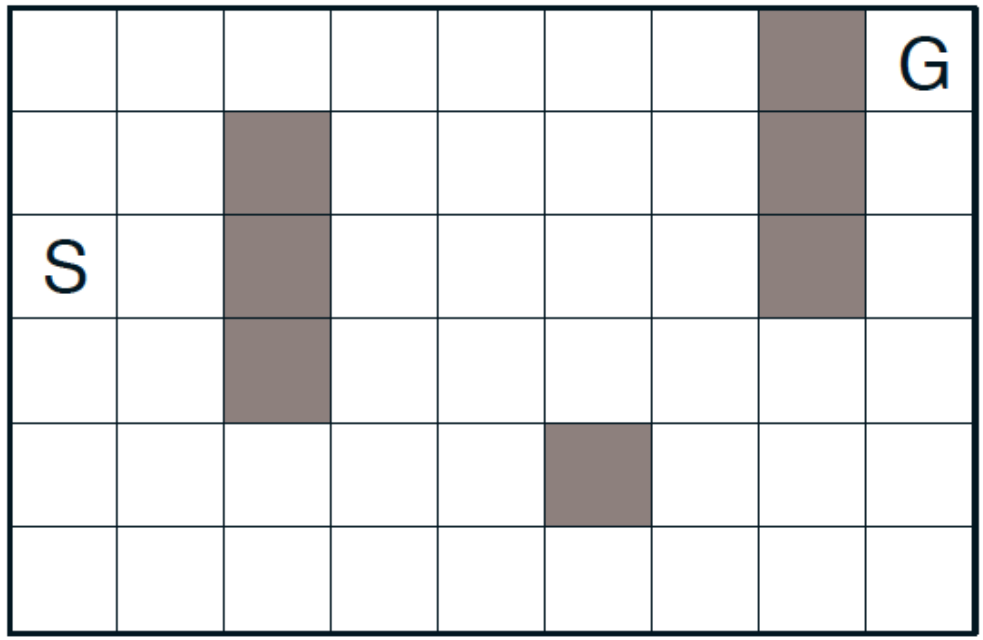


Figure: An Maze Problem

What is Reinforcement Learning?

Decision Making is a BIG inter-disciplinary topics!

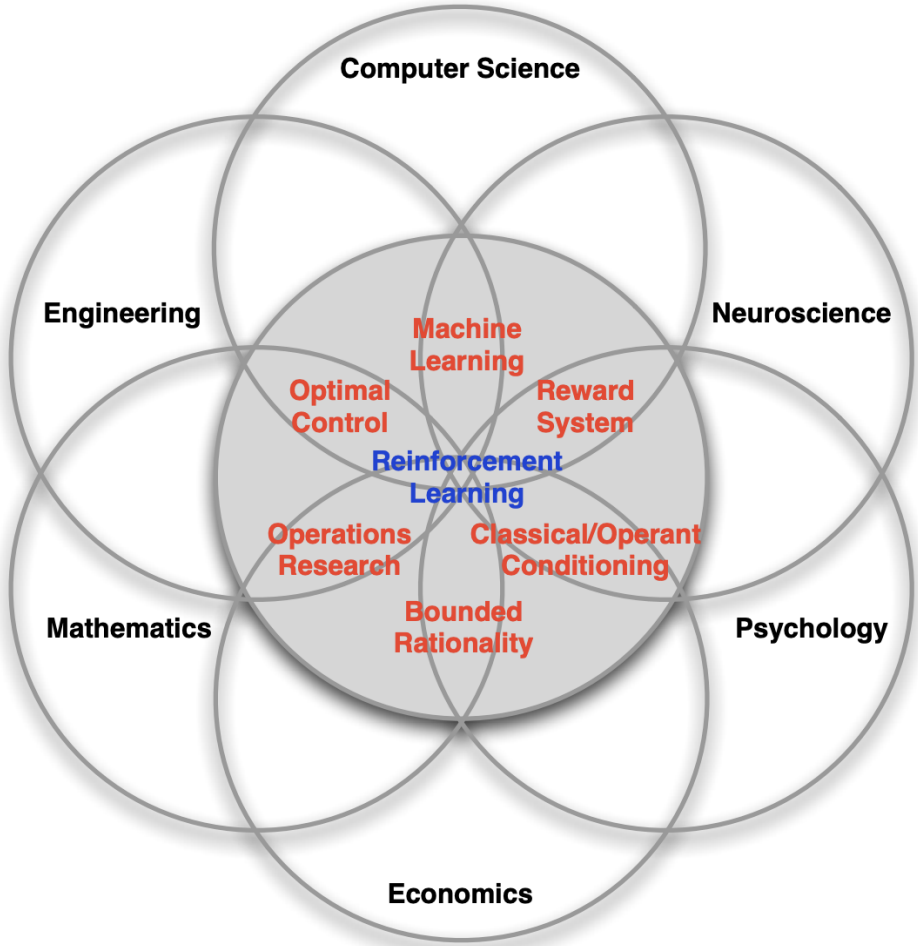


Figure: Related Disciplines by David Silver

Machine Learning: Supervised Learning

- Data: $\{x_i, y_i\}_{i=1...N}$
- Task: find $f : \mathbb{X} \to \mathbb{Y}$ such that $f(x) \approx y$
- the training is to minimize the loss w.r.t a criteria, e.g. the mean-square-error (MSE): $\sum_i (f(x_i) - y_i)^2$
- Two categories: regression and classification
- "Supervised": You know what is true

Machine Learning: Supervised Learning

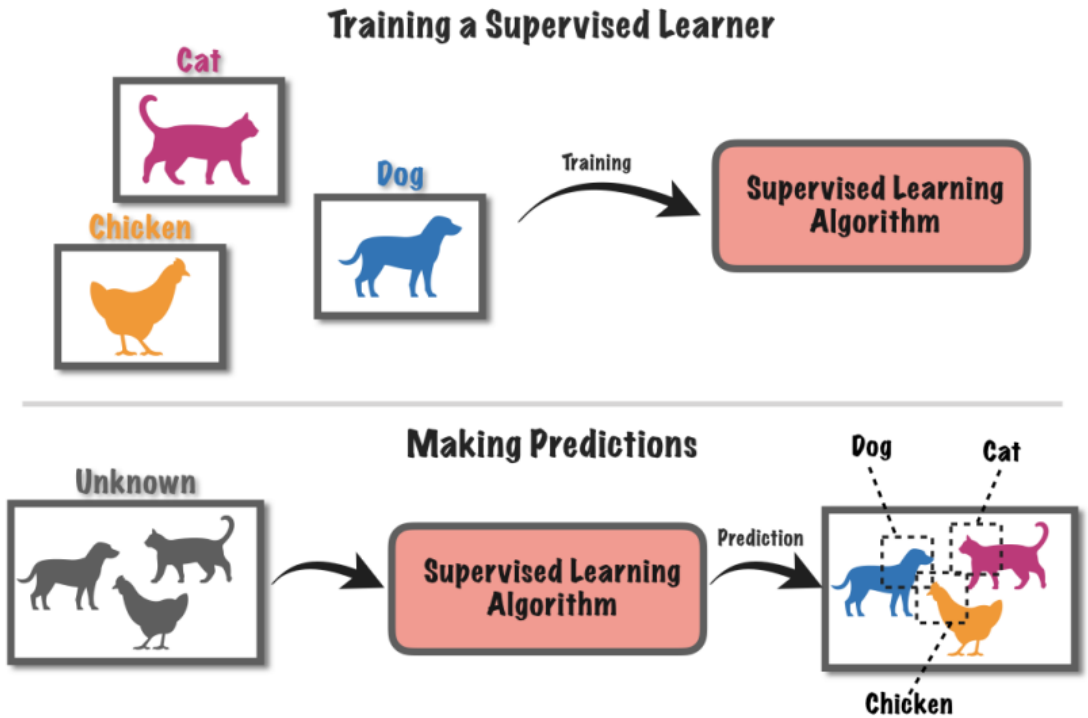


Figure: Supervised Learning

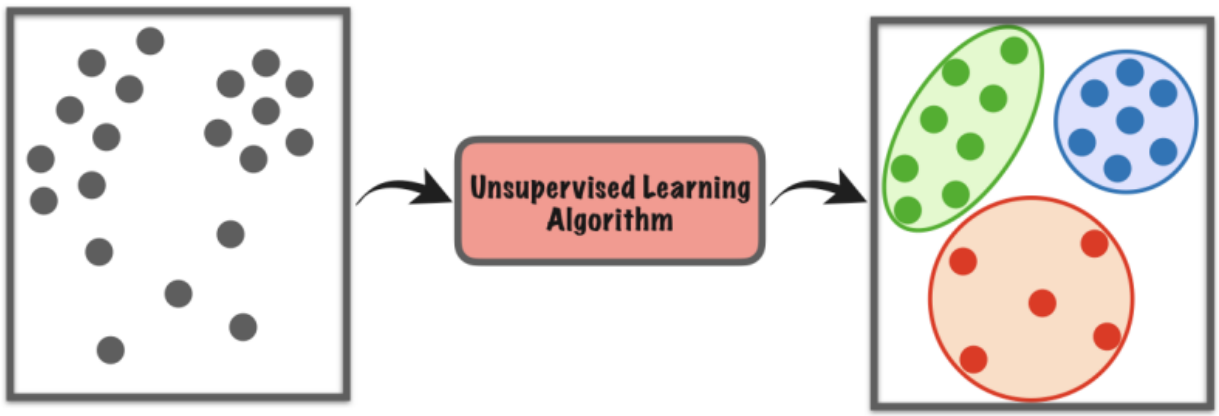
Supervised Learning: An illustration

The "Hello World" problem in supervised learning

Figure: MNIST data

Machine Learning: Unsupervised Learning

- Data: $\{x_i\}_{i=1...N}$
- Task: find some sort of underlying structure, correctly label/group the data based on the characteristics x_i
- "Unsupervised": You DON'T know what is true



Machine Learning: Reinforcement Learning

- Reinforcement Learning belongs to "Semi-Supervised Learning"
- We don't directly observe what is optimal, but rather some signals

Machine Learning: Reinforcement Learning

- Reinforcement Learning belongs to "Semi-Supervised Learning"
- We don't directly observe what is optimal, but rather some signals
- Also, what turns out to be "optimal" is sequentially-dependent

Machine Learning: Reinforcement Learning

- Reinforcement Learning belongs to "Semi-Supervised Learning"
- We don't directly observe what is optimal, but rather some signals
- Also, what turns out to be "optimal" is sequentially-dependent
- The case of GO

What is RL: The Value Function

- First assume $\mathcal{S}$ and $\mathcal{A}$ are discrete and finite for simplicity

What is RL: The Value Function

- First assume $\mathcal{S}$ and $\mathcal{A}$ are discrete and finite for simplicity
- define the accumulative reward $G_t = \sum_{t=1}^{n} \gamma^t R_t$

What is RL: The Value Function

- First assume $\mathcal{S}$ and $\mathcal{A}$ are discrete and finite for simplicity
- define the accumulative reward $G_t = \sum_{t=1}^{n} \gamma^t R_t$
- Formally, the Bellman Equation:

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left[R_t + \gamma G_{t+1} \mid S_t = s, A_t \sim \pi(s)\right] \\ &= \mathbb{E}\left[R_t + \gamma V_\pi\left(S_{t+1}\right) \mid S_t = s, A_t \sim \pi(s)\right] \end{aligned}$$

Here $a \sim \pi(s)$ means a is chosen by policy π in state s (note that π may be deterministic or stochastic)

What is RL: The Value Function

- First assume $\mathcal{S}$ and $\mathcal{A}$ are discrete and finite for simplicity
- define the accumulative reward $G_t = \sum_{t=1}^{n} \gamma^t R_t$
- Formally, the Bellman Equation:

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left[R_t + \gamma G_{t+1} \mid S_t = s, A_t \sim \pi(s)\right] \\ &= \mathbb{E}\left[R_t + \gamma V_\pi\left(S_{t+1}\right) \mid S_t = s, A_t \sim \pi(s)\right] \end{aligned}$$

  Here $a \sim \pi(s)$ means a is chosen by policy π in state s (note that π may be deterministic or stochastic)

- A similar equation holds for the optimal value:

$$V_*(s) = \max_a \mathbb{E}\left[R_t + \gamma V_*\left(S_{t+1}\right) \mid S_t = s, A_t = a\right]$$

What is RL: The Value Function (Cont.)

- Look familiar? In a typical Economic problem:

$$\max_{\{k_t\}} \sum_{t=1}^{n} \gamma^t \mathbb{E}[u(c_t)] \quad \text{s.t.} f(k_t, c_t) = 0$$

What is RL: The Value Function (Cont.)

- Look familiar? In a typical Economic problem:

$$\max_{\{k_t\}} \sum_{t=1}^{n} \gamma^t \mathbb{E}[u(c_t)] \quad \text{s.t.} f(k_t, c_t) = 0$$

- We can rewrite it in a Value Function format:

$$V_*(k) = \max_{k'} \mathbb{E}[u(c) + \gamma V_*(k')]$$

What is RL: The Value Function (Cont.)

- Look familiar? In a typical Economic problem:

$$\max_{\{k_t\}} \sum_{t=1}^{n} \gamma^t \mathbb{E}[u(c_t)] \quad \text{s.t.} f(k_t, c_t) = 0$$

- We can rewrite it in a Value Function format:

$$V_*(k) = \max_{k'} \mathbb{E}[u(c) + \gamma V_*(k')]$$

- We learnt Dynamic Programming (DP) to solve for the Value Function

RL, DP and Econ Problems

- So Far:
  - RL is closely related to DP
  - If we use DP to solve Econ problems, we can potentially use RL as well

RL, DP and Econ Problems

- So Far:
  - RL is closely related to DP
  - If we use DP to solve Econ problems, we can potentially use RL as well
- Two questions arise naturally:
  - difference between DP and RL?
  - Why not use RL in Econ?

from DP to RL

- Recall our Bellman Equation:

$$V_*(s) = \max_a \mathbb{E}\left[R_t + \gamma V_*\left(S_{t+1}\right) \mid S_t = s, A_t = a\right]$$

from DP to RL

- Recall our Bellman Equation:

$$V_*(s) = \max_a \mathbb{E}\left[R_t + \gamma V_*\left(S_{t+1}\right) \mid S_t = s, A_t = a\right]$$

- WLOG let's rewrite as

$$V_*(s) = \max_a R_t + \gamma \mathbb{E}[V_*(S_{t+1})] = \max_a R_t + \gamma \sum_{s'} P(s'|s,a) V_*(s')$$

from DP to RL

- Recall our Bellman Equation:

$$V_*(s) = \max_a \mathbb{E}\left[R_t + \gamma V_*\left(S_{t+1}\right) \mid S_t = s, A_t = a\right]$$

- WLOG let's rewrite as

$$V_*(s) = \max_a R_t + \gamma \mathbb{E}[V_*(S_{t+1})] = \max_a R_t + \gamma \sum_{s'} P(s'|s,a) V_*(s')$$

- In practice DP may suffer from:
  - P is not known or hard to write down
  - $\mathcal{S}, \mathcal{A}$ is continuous/high-dimensional
  - the max operator is computationally expensive

from DP to RL cont.

- Simulation is used to solve for unknown P

from DP to RL cont.

- Simulation is used to solve for unknown P
- rewrite the Bellman Equation for State Value Function

$$V_*(s) = \max_a R_t + \gamma \mathbb{E}[V_*(S_{t+1})] = \max_a R_t + \gamma \sum_{s'} P(s'|s,a) V_*(s')$$

- to State-Action Value Function

$$Q_*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q_*(s',a')$$

from DP to RL cont.

- Simulation is used to solve for unknown P
- rewrite the Bellman Equation for State Value Function

$$V_*(s) = \max_a R_t + \gamma \mathbb{E}[V_*(S_{t+1})] = \max_a R_t + \gamma \sum_{s'} P(s'|s,a) V_*(s')$$

- to State-Action Value Function

$$Q_*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q_*(s',a')$$

- the celebrated Q-learning algorithm:

$$Q^{i+1}(s,a) = (1-\alpha) Q^i(s,a) + \alpha(r + \gamma \max_{a'} Q^i(s',a'))$$

- key: r, s' is simulated

from DP to RL cont.

- The *Frozen-Lake* Environment:
  "the ice is slippery, so you won't always move in the direction you intend."

```
SFFF       (S: starting point, safe)
FHFH       (F: frozen surface, safe)
FFFH       (H: hole, fall to your doom)
HFFG       (G: goal, where the frisbee is located)
```

Figure: Frozen-Lake

from DP to RL cont.

- In practice DP may suffer from:
  - P is not known or hard to write down
  - $\mathcal{S}, \mathcal{A}$ is continuous/high-dimensional
  - the max operator is computationally expensive
- To Solve for Problem 2 & 3, we use Neural Network and go to the modern Deep RL
  - Critic: A Value Network $Q_\theta(s, a)$
  - Actor: A Policy Network $\pi_\phi(s)$

Three Components of a RL project

- An Interactive Environment
  - The problem you want to solved
  - generate an initial state, take an action as input and return the reward and next state as feedback

Three Components of a RL project

- An Interactive Environment
  - The problem you want to solved
  - generate an initial state, take an action as input and return the reward and next state as feedback
- An Agent
  - the "brain": e.g. the value function, the policy function
  - Some behavior patterns: the *exploration-exploitation trade-off*
  - Some other structures, e.g. the "memory for experiences"

Three Components of a RL project

- An Interactive Environment
  - The problem you want to solved
  - generate an initial state, take an action as input and return the reward and next state as feedback
- An Agent
  - the "brain": e.g. the value function, the policy function
  - Some behavior patterns: the *exploration-exploitation trade-off*
  - Some other structures, e.g. the "memory for experiences"
- A Training Algorithm
  - hyper-parameters to pin down, e.g. the learning rate scheme, training epochs

GymAI Environments

- RL researchers test on various benchmark Environments to verify the performance of their latest learning algorithms

GymAI Environments

- RL researchers test on various benchmark Environments to verify the performance of their latest learning algorithms
- GymAI: one of the Open-Source collections of Environments

GymAI Environments

- RL researchers test on various benchmark Environments to verify the performance of their latest learning algorithms
- GymAI: one of the Open-Source collections of Environments
- When we DIY our Environment (e.g. an Econ model) we follow the gymAI Environment structure

GymAI Environments

- RL researchers test on various benchmark Environments to verify the performance of their latest learning algorithms
- GymAI: one of the Open-Source collections of Environments
- When we DIY our Environment (e.g. an Econ model) we follow the gymAI Environment structure
- The "Hello World" Environment in DRL: The CartPole Problem
  - State: A 4-d tuple, continuous
  - Action: 0 or 1 denoting Left or Right
  - reward: $+1$ unless game finish
  - Link: CartPole

GymAI Environments

- RL researchers test on various benchmark Environments to verify the performance of their latest learning algorithms
- GymAI: one of the Open-Source collections of Environments
- When we DIY our Environment (e.g. an Econ model) we follow the gymAI Environment structure
- The "Hello World" Environment in DRL: The CartPole Problem
  - State: A 4-d tuple, continuous
  - Action: 0 or 1 denoting Left or Right
  - reward: $+1$ unless game finish
  - Link: CartPole
- Link: The Atari Games

GymAI Environments

```
In [19]: import gym

         # create the Cart-Pole env
         env = gym.make('CartPole-v0')

         # initialize
         state = env.reset()
         state

Out[19]: array([-0.03184797,  0.03824757,  0.00570897,  0.01185016])

In [20]: # take an action
         action = 0
         env.step(action)

Out[20]: (array([-0.03108302, -0.15695578,  0.00594598,  0.30632885]), 1.0, False, {})
```

Figure: Code Demo

Programming your RL project

- We are not RLers so to create new algorithms
- Yet implementing RL/DL is code-demanding
  - build your own Environments
  - monitor your performance
  - testing, debugging, tuning
  - run your code on High-Performance-Computing Cluster

Programming your RL project

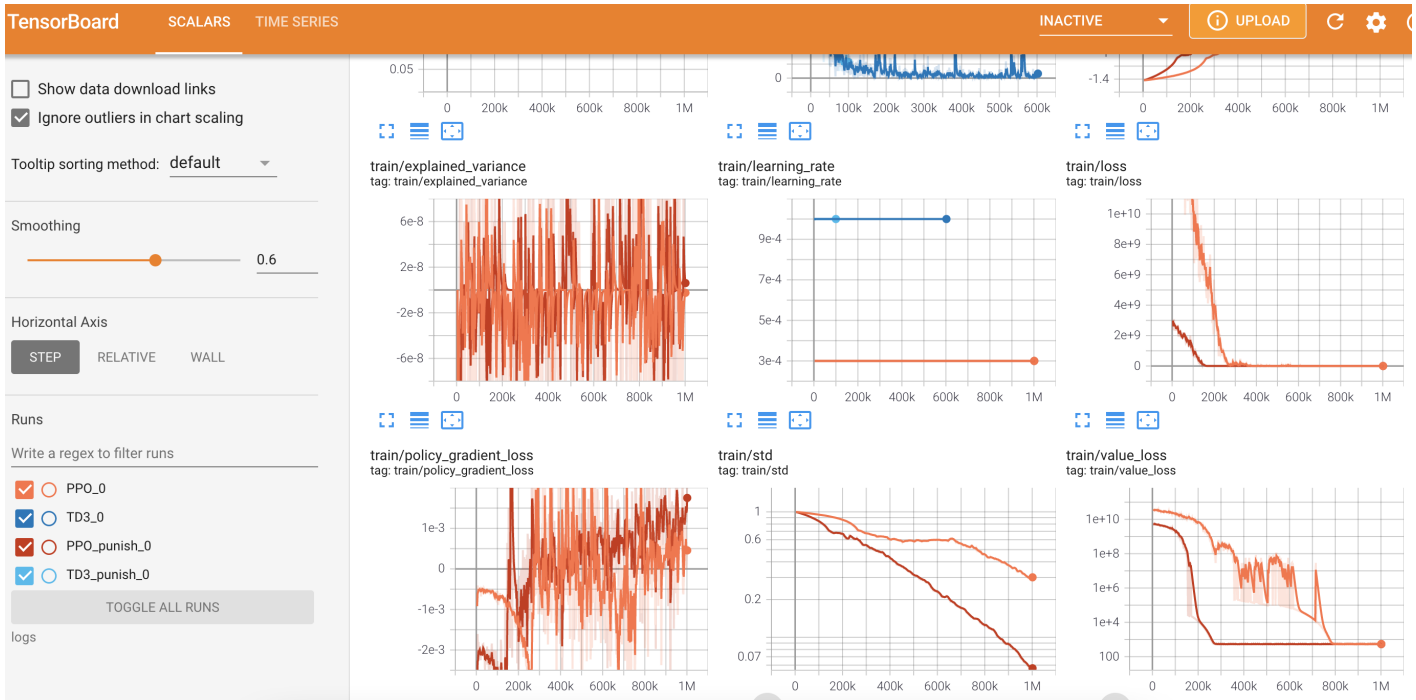


Figure: Tensorboard for monitoring

RL in Economics

- RL may be unnecessary if your problem can be solved by in an easier way
- BUT: economic models are more and more complex now
- A small field: Agent-Based Computational Economics
- Related topics: Bounded Rational Agents, Non-Equilibrium, ..., see the research by W. Brian Arthur in Santa Fe Institute

RL in Economics: Literature

- DRL in a Monetary Model (Chen, et al 2020)
- AI, algorithmic pricing and collusion (AER, 2020)
- AI as structural estimation: Deep Blue, Bonanza, and AlphaGo (2020)
- RL for Optimization of COVID-19 Mitigation policies (2020)
- High-Performance Computing Implementations of Agent-Based Economic Models for Realizing 1:1 Scale Simulations of Large Economies(IEEE, 2020)

Multi-Agent Learning and Game Theory

- Link:Multi-Agent Hide and Seek

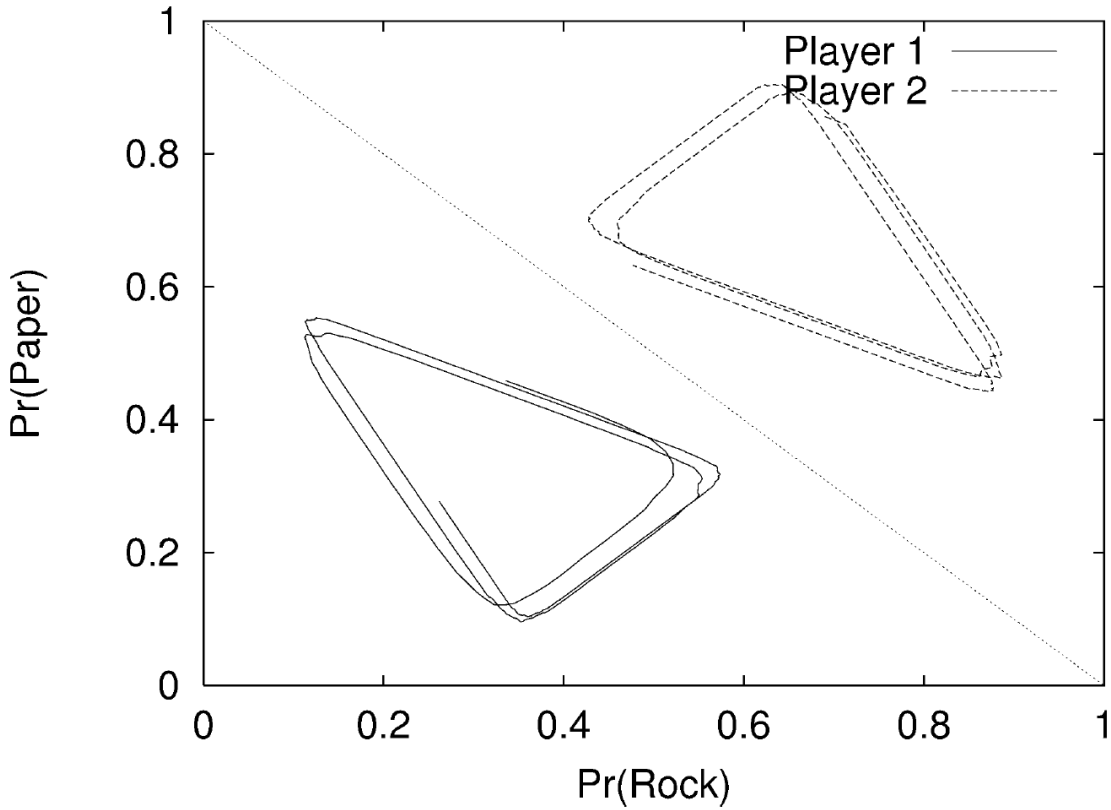
Multi-Agent Learning and Game Theory

- Link:Multi-Agent Hide and Seek
- The learning of other agents would make the Environment non-stationary

Multi-Agent Learning and Game Theory

- Link:Multi-Agent Hide and Seek
- The learning of other agents would make the Environment non-stationary
- Many game-theory settings have been studied previously for Multi-Agent learning, "Evolutionary Game Theory"
- it is non-trivial to build up learning algorithms even for those trivial matrix games

Multi-Agent Learning and Game Theory



(a) Policy Hill-Climbing

Figure: Non-Convergence in Rock-Paper-Scissor

Multi-Agent Learning and Game Theory

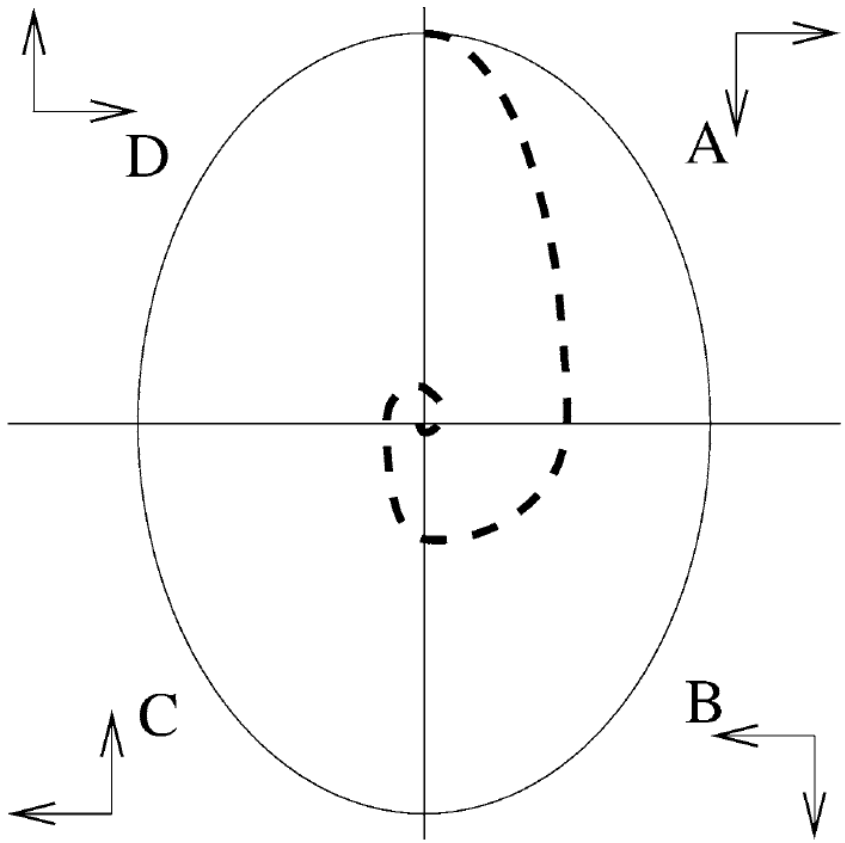
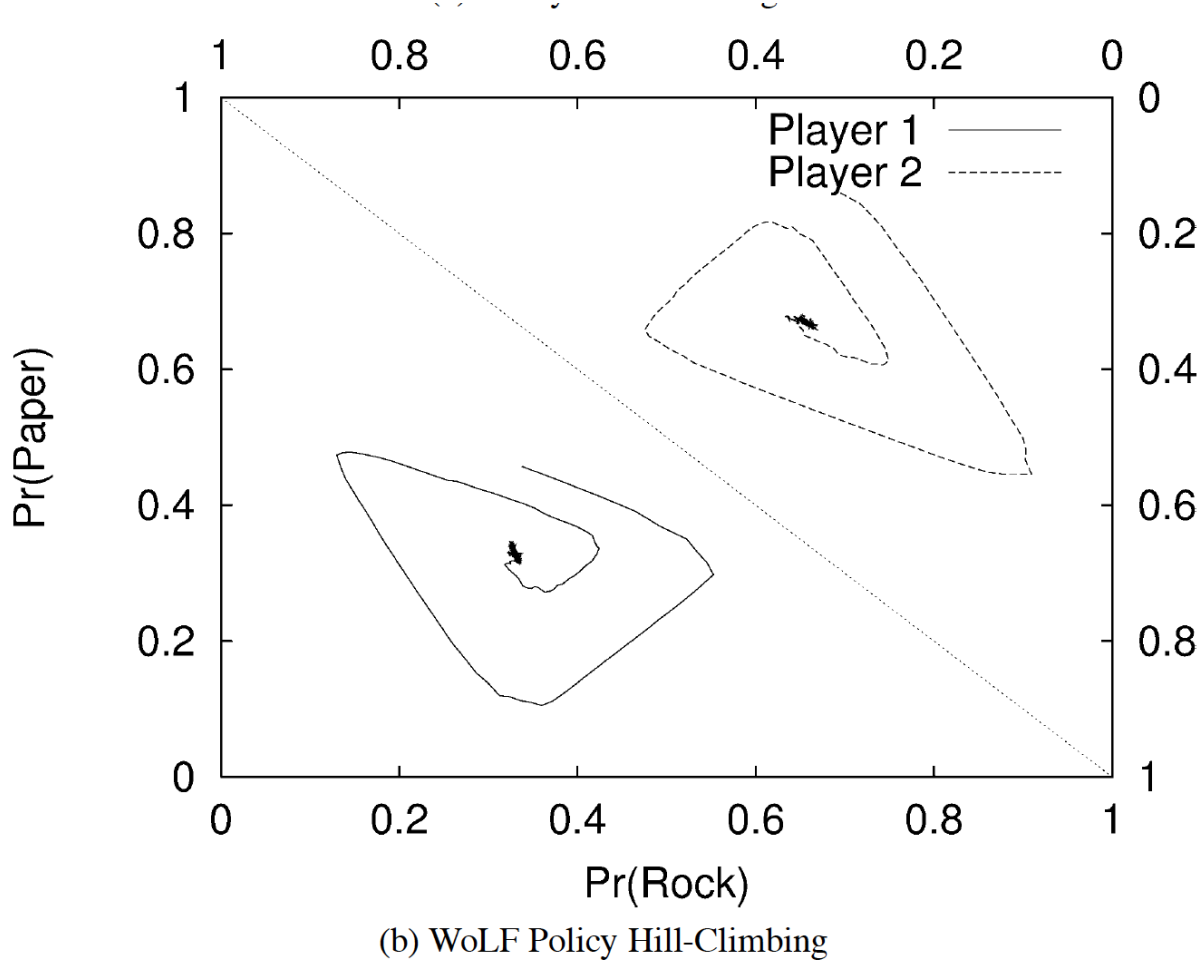


Figure: The "Win-or-Learn-Fast" Algorithm

Multi-Agent Learning and Game Theory



(b) WoLF Policy Hill-Climbing

Figure: Convergence in Rock-Paper-Scissor with WoLF

Multi-Agent Reinforcement Learning

- Link:AI-Economist with tax policies

Conclusion

- RL is nothing far away from what we learn in economics

Conclusion

- RL is nothing far away from what we learn in economics
- RL could potentially help us to solve some complex settings where we should rely on simulations to solve agents' decision-makings

Conclusion

- RL is nothing far away from what we learn in economics
- RL could potentially help us to solve some complex settings where we should rely on simulations to solve agents' decision-makings
- MARL could even go further to study more interactive settings

Conclusion

- RL is nothing far away from what we learn in economics
- RL could potentially help us to solve some complex settings where we should rely on simulations to solve agents' decision-makings
- MARL could even go further to study more interactive settings
- policy-makers' problem in macro, strategic plays in game theory, firms' interaction in IO...

Conclusion

- RL is nothing far away from what we learn in economics
- RL could potentially help us to solve some complex settings where we should rely on simulations to solve agents' decision-makings
- MARL could even go further to study more interactive settings
- policy-makers' problem in macro, strategic plays in game theory, firms' interaction in IO...
- We believe this is a promising research direction!