Deep Reinforcement Learning: An Overview

Patrick Emami

PhD student, CISE department

July 10, 2018

Motivation

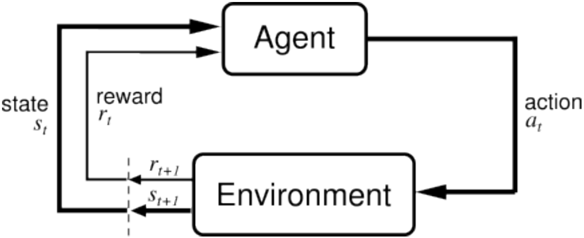
What is a good framework for studying intelligence?

Motivation

What is a good framework for studying intelligence?

What are the necessary and sufficient ingredients for building agents that learn and act like people?

Reinforcement Learning



Source: Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. Vol. 1. No. 1. Cambridge: MIT press, 1998.

Reinforcement Learning

My Perspective

Reinforcement Learning is necessary but not sufficient for general (strong) artificial intelligence

Source: Yann LeCun, NIPS 2016

Markov Decision Processes

Definition

A **Markov decision process** (MDP) is a formal way to describe the sequential decision-making problems encountered in RL.

Markov Decision Processes

Definition

A **Markov decision process** (MDP) is a formal way to describe the sequential decision-making problems encountered in RL.

In the simplest RL setting, an MDP is specified by states S, actions A, an episode length H, and a reward function $r(s, a)$.

Policies and Value Functions

- A policy $\pi(a|s)$ is a behavior function for selecting an action given the current state.
- The action-value function is the expected total reward accumulated from starting in state s, taking action a, and following policy π until the end of the length H episode:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{H} r_t \middle| s_0 = s, a_0 = a\right]$$

"What is the utility of doing action a when I'm in state s?"

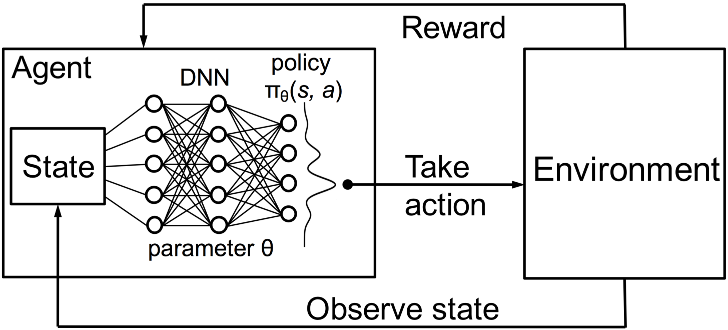
Big Picture

Find policy π^* that maximizes expected total reward, i.e.,

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, Q^{\pi}(s, a).$$

In particular, for any start state $s_0 \in S$, the agent can use π^* to select the action a_0 that will maximize its expected total reward.

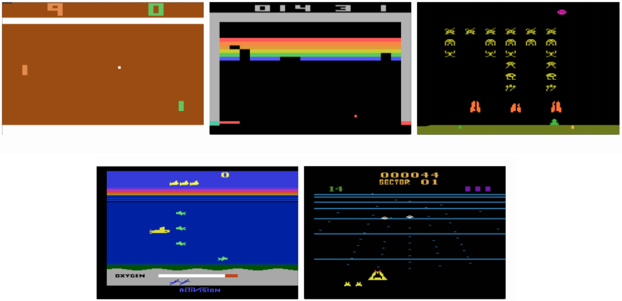
Deep Reinforcement Learning



Source:
http://people.csail.mit.edu/hongzi/

Deep Reinforcement Learning

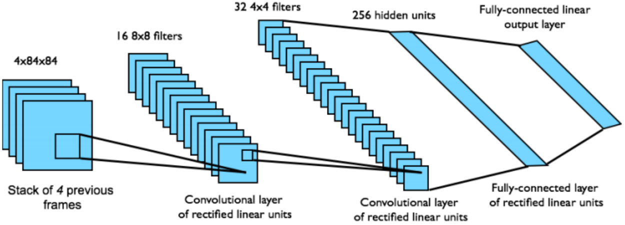
Playing Atari (2013)



Source: Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

Deep Q-Network (DQN)

- End-to-end learning of values $Q(s, a)$ from pixels s
- Input state s is stack of raw pixels from last 4 frames
- Output is $Q(s, a)$ for 18 joystick/button positions
- Reward is change in score for that step



Network architecture and hyperparameters fixed across all games

[Mnih et al.]

Just Apply Gradient Descent

- Represent $Q^{\pi}(s, a)$ by a deep Q-network with weights w

$$Q(s, a, w) \approx Q^{\pi}(s, a)$$

- Define objective function by mean-squared Bellman error

$$L(w) = \mathbb{E}\left[\left(r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w)\right)^2\right]$$

- Leading to the following gradient

$$\frac{\partial L}{\partial w} = \mathbb{E}\left[\left(r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w)\right)\frac{\partial Q(s, a, w)}{\partial w}\right]$$

- Optimize with stochastic gradient descent

Stability Issues with Deep RL

Naive Q-learning with non-linear function approximation oscillates or diverges

- Experiences from episodes generated during training are correlated, non-iid
- Policy can change rapidly with slight changes to Q-values
- Q-learning gradients can be large and unstable when backpropagated

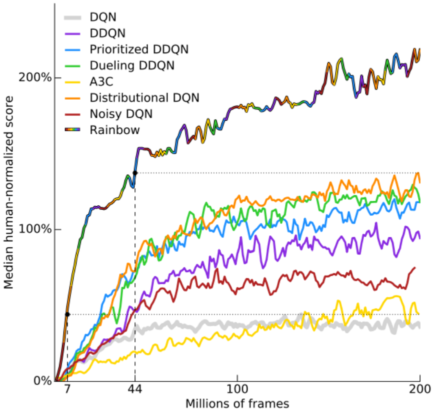
Stabilizing Deep RL

- Maintain a replay buffer of experiences to uniformly sample from to compute gradients for Q-network. This decorrelates samples and improves samples efficiency
- Hold the parameters of the target Q-values fixed in Bellman error with a target Q-network. Periodically update the parameters of the target network
- Clip rewards and potentially clip gradients as well

	B. Rider	Breakout	Enduro	Pong	Q*bert	Seaquest	S. Invaders
Random	354	1.2	0	−20.4	157	110	179
Sarsa [3]	996	5.2	129	−19	614	665	271
Contingency [4]	1743	6	159	−17	960	723	268
DQN	**4092**	**168**	**470**	**20**	**1952**	**1705**	**581**
Human	7456	31	368	−3	18900	28010	3690
HNeat Best [8]	3616	52	106	19	1800	920	**1720**
HNeat Pixel [8]	1332	4	91	−16	1325	800	1145
DQN Best	**5184**	**225**	**661**	**21**	**4500**	**1740**	1075

Source: Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

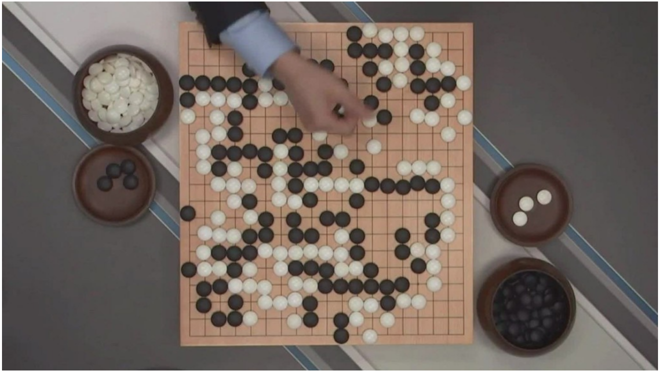
Rainbow DQN (2017)



Source: Hessel, Matteo, et al. "Rainbow: Combining Improvements in Deep Reinforcement Learning." arXiv:1710.02298 (2017).

Figure 1: **Median human-normalized performance** across 57 Atari games. We compare our integrated agent (rainbow-

AlphaGo (2015)

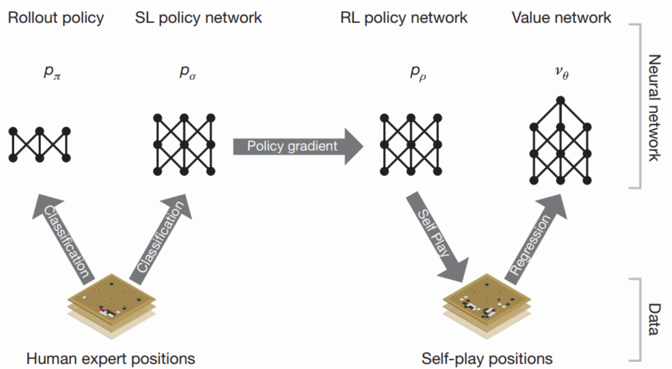


It was thought that AI was a decade away from beating humans at Go

AlphaGo

Key Ingredients

Tree search augmented with policy and value deep networks that intelligently control exploration and exploitation



Monte Carlo Tree Search

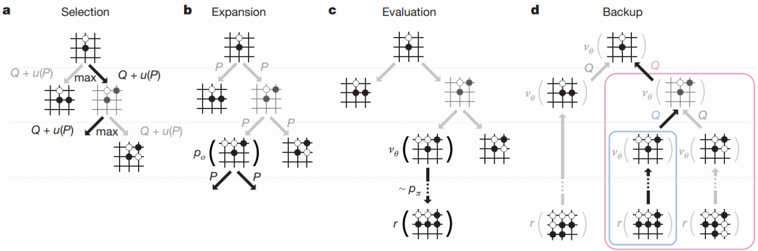


Figure 3 | Monte Carlo tree search in AlphaGo. a, Each simulation traverses the tree by selecting the edge with maximum action value Q, plus a bonus $u(P)$ that depends on a stored prior probability P for that edge. **b**, The leaf node may be expanded; the new node is processed once by the policy network p_σ and the output probabilities are stored as prior probabilities P for each action. **c**, At the end of a simulation, the leaf node is evaluated in two ways: using the value network v_θ; and by running a rollout to the end of the game with the fast rollout policy p_π, then computing the winner with function r. **d**, Action values Q are updated to track the mean value of all evaluations $r(\cdot)$ and $v_\theta(\cdot)$ in the subtree below that action.

Source: Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." nature 529.7587 (2016): 484-489.

AlphaZero (2017)

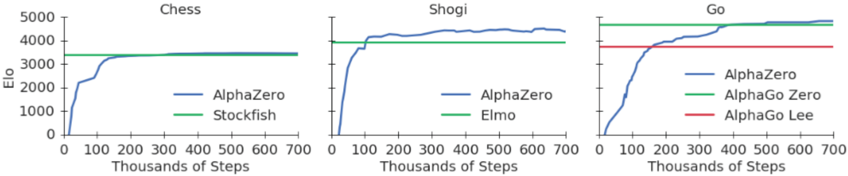


Figure 1: Training *AlphaZero* for 700,000 steps. Elo ratings were computed from evaluation games between different players when given one second per move. **a** Performance of *AlphaZero* in chess, compared to 2016 TCEC world-champion program *Stockfish*. **b** Performance of *AlphaZero* in shogi, compared to 2017 CSA world-champion program *Elmo*. **c** Performance of *AlphaZero* in Go, compared to *AlphaGo Lee* and *AlphaGo Zero* (20 block / 3 day) (*29*).

Source: arXiv:1712.01815

Continuous control

1. Locomotion Behaviors
   https://www.youtube.com/watch?v=g59nSURxYgk
2. Learning to Run
   https://www.youtube.com/watch?v=mbjuaRg__DI
3. Robotics
   https://www.youtube.com/watch?v=Q4bMcUk6pcw&t=56s

Continuous control

Can we learn Q^{π^*} by minimizing the expected Bellman error?

Continuous Action Spaces

For $A \in \mathbb{R}^n$,

$$\mathbb{E}\left[\left(r + \gamma \max_{a' \in A} Q(s', a', w) - Q(s, a, w)\right)^2\right]$$

Requires solving a non-convex optimization problem!

Policy Gradient Algorithms

REINFORCE

$$\nabla_w J(w) = \sum_{i=1}^{N} \log \pi_w(a_i|s_i)(R - b)$$

R can be the sum of rewards for the episode or the discounted sum of rewards for the episode. b is a baseline, or control variate, for reducing the variance of this gradient estimator.

Source: Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3-4 (1992): 229-256.

Policy Gradient Algorithms

REINFORCE

$$\nabla_w J(w) = \sum_{i=1}^{N} \log \pi_w(a_i|s_i)(R - b)$$

R can be the sum of rewards for the episode or the discounted sum of rewards for the episode. b is a baseline, or control variate, for reducing the variance of this gradient estimator.

How does this work? Ascend the policy gradient!

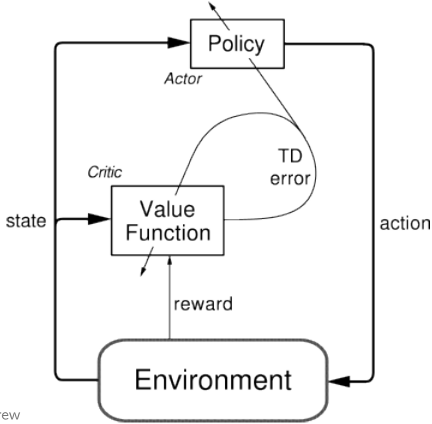
Source: Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." Machine learning 8.3-4 (1992): 229-256.

Deep Deterministic Policy Gradient (2016)

1. Let the policy be a deterministic function $\pi(s, \theta) : S \rightarrow A$, $S \in \mathbb{R}^m$, $A \in \mathbb{R}^n$, parameterized as a deep network
2. Still maximize expected total reward, except now need to compute the deterministic policy (actor) gradient and the (critic) action-value gradient
3. Train both the policy and action-value networks with an **actor-critic** approach

Source: Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).

Actor-Critic

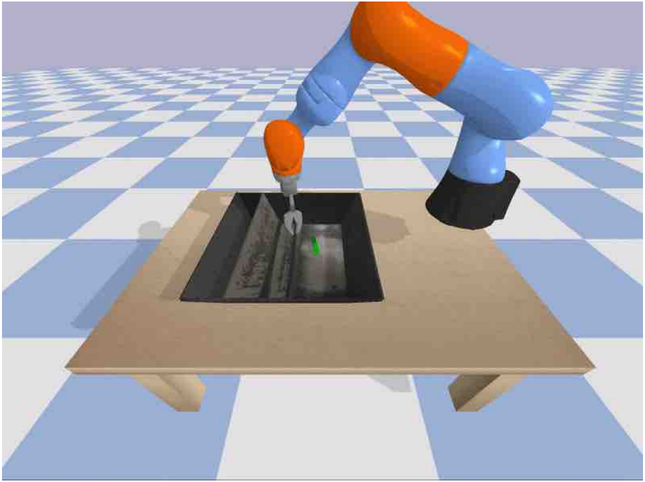


Source: Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. Vol. 1. No. 1. Cambridge: MIT press, 1998.

Future Research Directions

1. Sample-efficient learning by embedding priors about the world, e.g., intuitive physics
2. Low-variance, unbiased policy gradient estimators
3. Multi-agent RL (Dota2 and Starcraft)
4. Safe RL
5. Meta-learning and transfer learning
6. Reinforcement learning on combinatorial action spaces

Source: Emami, Patrick, and Sanjay Ranka. "Learning Permutations with Sinkhorn Policy Gradient." arXiv preprint arXiv:1805.07010 (2018).



Source: http://blog.otoro.net/2017/11/12/evolving-stable-strategies/

Fin.