OpenBrain: Backpropagation Free RL

Guss & Zhong et. al
Machine Learning at Berkeley

April 22, 2016

Overview

Background on Reinforcement Learning

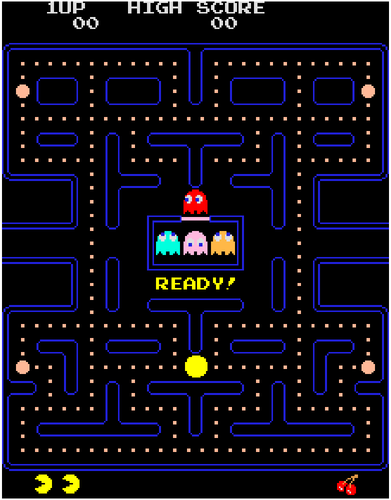
The setup.

1. Environment, $E = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \rho, r)$.
   1. State space, $\mathcal{S} = \mathbb{R}^n$
   2. Action space, $\mathcal{A} = \mathbb{R}^m$
   3. Reward space, $\mathcal{R} = \mathbb{R}$
   4. Transition function, $\rho(s' \mid s, a)$. Given a previous state s and action a, environment gives s'.
   5. Reward function $r(s, a) \in \mathcal{R}$.
2. Deterministic agent $\pi : \mathcal{S} \to \mathcal{A}$ acts in E.

$$s_1 \xrightarrow{\pi} a_1 \xrightarrow{\rho, r} s_2, r_2 \xrightarrow{\pi} a_2 \xrightarrow{\rho, r} \cdots$$

Eg. Pacman sees the screen, and decides to move $\uparrow, \downarrow, \rightarrow, \leftarrow$ and then gets a reward for eating food.

Background on Reinforcement Learning



The action-value function (simplified).

1. The future expected reward of an agent π is

$$Q^\pi(s_t, a_t) = \underbrace{r(s_t, a_t)}_{\text{reward for } a_t} + \sum_{n=t+1}^{\infty} \gamma^n r(s_n, \pi(s_n))$$

2. The Bellman equation gives us

$$Q^\pi(s_t, a_t) = r_t + \gamma Q^\pi(s_{t+1}, \pi(s_{t+1}))$$

''

3. Given some state s_t, the **best** agent, π^* is one that take action

$$a_t = \arg\max_a Q(s_t, a).$$

Background on Reinforcement Learning

The action-value function (simplified).

1. The Q function for π^* is

$$Q^*(s_t, a_t) = r_t + \gamma \arg\max_a Q^\pi(s_{t+1}, a).$$

2. We can *approximate* this with deep learning!
   1. Make a neural network $\mathcal{N} : \mathcal{S} \to \mathbb{R}^n$ which predicts the future reward of taking each possible action

$$\mathcal{N}(s_t) = \begin{pmatrix} Q^*(s_t, a_1) \\ Q^*(s_t, a_2) \\ \vdots \\ Q^*(s_t, a_n) \end{pmatrix}$$

Deep Q-Learning

$$s_t \longrightarrow \mathcal{N}$$

$$Q^*(s_t, a_1) = -12$$

$$Q^*(s_t, a_2) = 0.17$$

$$Q^*(s_t, a_3) = 0.22$$

$$Q^*(s_t, a_4) = 0.03$$

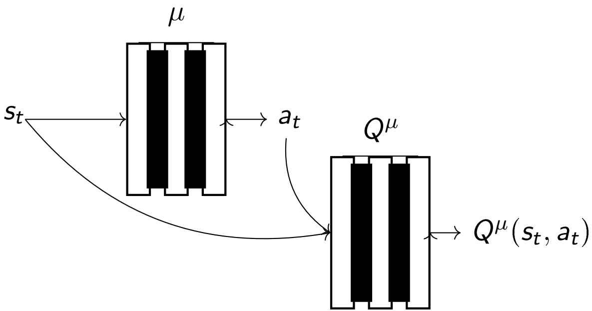
$$Q^*(s_t, a_5) = -3.2$$

$$\pi(s_t) = \arg\max_a \mathcal{N}(s_t)$$

Can we do better?

Deep Determisitic Policy Gradient

1. Actor neural network $\mu : \mathcal{S} \rightarrow \mathcal{A}$
2. Critic network $Q^{\mu} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$
3. Performance of μ is $Q^{\mu}(s_t, \mu(s_t))$. **Maximize performance!** $\nabla_W Q^{\mu}(s_t, a_t) = \nabla_a Q^{\mu}(s_t, a) \cdot \nabla_W \mu(s_t)$



Neuromorphically Local Agents

1. Every neuron in the brain is an agent!
2. Anterior neurons are the state that μ^n sees.
3. The action of each μ^n is its output:

$$\mu^n(s_t) = \sigma\left(\sum_{i=1}^{m} W_{in} s_t^{(i)}\right) = a_t$$

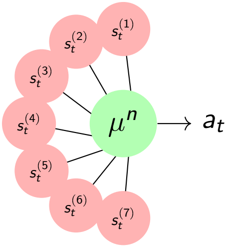


Figure: A neuron n and its environment, E^n.

Our Approach

Local Q Critics

1. Hypothesis: μ^n lives in an extremely **simple** environment.
2. Can we estimate Q^n without error backprop?
3. **Remark.** Training whole μ is equivalent to training μ^n simultaneously:

$$\nabla_{W^{(n)}} Q^n(\mu^n) = (\nabla_W Q^\mu(\mu))^{(n)}$$

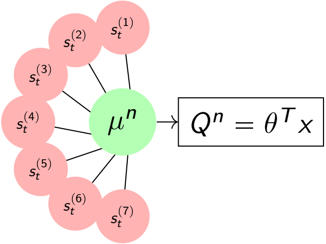


Figure: A linear critic for μ^n

The Training Regime

1. We can train every neuron **simultaneously** without BP.
2. There is no "extra" Q network, just $2n$ parameters!
3. Could be biologically plausible (certainly more reasonable than BP)
4. Linear critic $\iff$ *compatability* $\iff$ no bias in Q^n

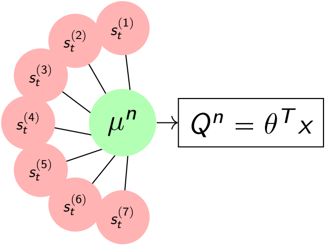


Figure: A linear critic for μ^n

Experiments

Experiment 1.

1. Test if training μ with the full $Q^n \implies$ each μ^n acting optimal to Q^n
2. Is it true in practice that $\nabla_{W^{(n)}} Q^n(\mu^n) = (\nabla_W Q^\mu(\mu))^{(n)}$?

Experiment 2.

1. Train μ^n using $Q^n \implies Q$ optimal?

Experiment 3.

1. Beat the state of the art in Atari 2600 environments!

Questions?

wguss@berkeley.edu

github.com/mlberkeley/openbrain