Introduction to Deep (actually quite shallow...) Reinforcement Learning

An Odyssey

Yuxin Fong

School of Engineering Science
Huazhong University of Science and Technology

12.2018

Outline

School of Engineering Sciences

School of Engineering Sciences

A demo: RL agent playing Atari games

Reinforcement learning (RL) is a general-purpose framework for artificial intelligence.

Definition

RL is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.

Some remarks:

- RL is for an agent with the capacity to act.
- Each action influences the agent's future state.
- Success is measured by a scalar reward signal.

School of Engineering Sciences

To formlize a little bit:

State

A state S is a complete description of the state of the world. There is no information about the world which is hidden from the state.

The set of all valid states is denoted as state $\mathcal{S}$.

Action

Different environments allow different kinds of actions. The set of all valid actions in a given environment is often called the action space $\mathcal{A}$.

School of Engineering Sciences

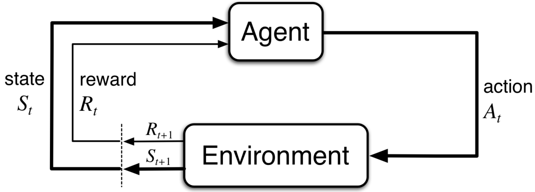


Figure: The RL Universe.

At each step(period) t the agent:

- Receives state s_t.
- Receives scalar reward r_t.
- Executes action a_t.

At each step(period) t the agent:

- Receives action a_t.
- Emit state s_{t+1}.
- Emit reward r_{t+1}.

School of Engineering Sciences

Random Variable

A random variable is a function: $\Omega \to \mathbb{R}$, Ω is the sample space.

- What's the meaning of "$X = x$"?

Conditional Expectation

- Def: $\mathbb{E}[X|Y=y] = \sum_x x\mathbb{P}[X=x|Y=y]$
- $\mathbb{E}[X|Y=y]$ is a function of y, not x any more.
- Adam's law:

$$\begin{aligned}\mathbb{E}[\mathbb{E}[X|Y]] &= \sum_y y(\sum_x x\mathbb{P}[X=x|Y=y]) \\ &= \sum_x x(\sum_y y\mathbb{P}[X=x|Y=y]) \\ &= \sum_x x\mathbb{P}[X=x] = \mathbb{E}[X]\end{aligned}$$

So far, we've RL in an informal way. The standard mathematical formalism for this setting is Markov Decision Processes (MDPs).

The name Markov Decision Process refers to the fact that the system obeys the Markov property.

Markov property

In probability theory and statistics, the term Markov property refers to the memoryless property of a stochastic process.

Transitions only depend on the most recent state and action, and no prior history.

For example: A state S_t is Markov if and only if

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

$\mathcal{MDP}$ in a netshell:

"The future is independent of the past given the present."

Policies:

- A policy is the agent's behaviour.
- It is a map from state to action.
- It can be deterministic, in which case it is usually denoted by μ:

Deterministic Policy

$$A_t = \mu(S_t).$$

- Or it may be stochastic, in which case it is usually denoted by π:

Stochastic Policy

$$A_t \sim \pi(\cdot|S_t), \ \textit{where } \pi(a|s) := \mathbb{P}[A_t = a|S_t = s].$$

Usually, stochastic policy is more common.

Trajectories(Episodes)

A trajectory τ is a sequence of states, actions and rewards in the world:

$$\tau = (S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, S_3, ...)$$

$$\tau_t = (S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, A_{t+2}, R_{t+3}, S_{t+3}, ...)$$

Dynamics of the MDP

The function p, called transitions function, defines the dynamics of the MDP:

$$p(s', r|s, a) := \mathbb{P}[S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a]$$

Important Fact

A RL problem is characterized by two import functions: Its policy π and its transitions function p.

Two Facts

This two facts will be frequently used:

- $$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r|s, a) = 1, \ \textit{for all } s \in \mathcal{S}, a \textit{ in } \mathcal{A}.$$

- $$\begin{aligned} p(s'|s, a) &= \mathbb{P}[S_t = s' | S_{t-1} = s, A_{t-1} = a] \\ &= \sum_{r \in \mathcal{R}} p(s', r|s, a) \\ (&= \int_{r \in \mathcal{R}} p(s', r|s, a) dr) \end{aligned}$$

One Def: Reward Function

To deal with the randomness of the reward, we define Reward Function $r(s, a)$, which is the expected reward:

$$
\begin{aligned}
r(s, a) &= \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a] \\
&= \sum_{r \in \mathcal{R}} r \mathbb{P}[R_t = r | S_{t-1} = s, A_{t-1} = a] \\
&= \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r | s, a)
\end{aligned}
$$

School of Engineering Sciences

The goal of the agent is to maximize some notion of cumulative expected reward(also called expected return) over a trajectory.

Cumulative Reward(Return): $G(\tau)$

The cumulative reward, or return, is a function of trajectories τ. There are two kinds of return:

- Receives action Finite-horizon undiscounted return:

$$G(\tau) = \sum_{t=1}^{T} R_t$$

- Infinite-horizon discounted return:

$$G(\tau) = \sum_{t=1}^{\infty} \gamma^{t-1} R_t, \ \gamma \in (0, 1)$$

The goal of the agent is to maximize some notion of cumulative expected reward(also called expected return) over a trajectory.

So, let's first deal with the probability distributions over trajectories.

Probability Distributions over Trajectories $\tau : \mathcal{P}_\pi[\tau]$

$$
\begin{aligned}
\mathcal{P}_\pi[\tau] &= \mathbb{P}[S_0, A_0, S_1, A_1, ...] \\
&= \rho(S_0)\mathbb{P}[A_0, S_1, A_1, ...|S_0] \\
&= \rho(S_0)\mathbb{P}[A_0|S_0]\mathbb{P}[S_1|A_0, S_0]\ \mathbb{P}[A_1, S_2, A_2, S_3, ...|S_0, A_0, S_1] \\
&= \rho(S_0)\pi(A_0|S_0)p(S_1|A_0, S_0)\mathbb{P}[A_1, S_2, A_2, , S_3, ...|S_1]\ \text{(Markov Property)} \\
&= \rho(S_0)\pi(A_0|S_0)p(S_1|A_0, S_0)\pi(A_1|S_1)p(S_2|A_1, S_1)\mathbb{P}[A_2, S_3, ...|S_2] \\
&= \rho(S_0)\prod_t \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)
\end{aligned}
$$

Probability Distributions over Trajectories τ: $\mathcal{P}_\pi(\tau)$

$$\mathcal{P}_\pi(\tau) = \rho(S_0) \prod_t \pi(A_t|S_t) p(S_{t+1}|S_t, A_t)]$$

Sence we have defined the probability distributions over trajectories, and the cumulative reward(also called return), we can now define the cumulative expected reward(also called expected return) over a trajectory.

Cumulative Expected Reward(Expected Return: $\mathcal{J}(\pi)$)

$$\mathcal{J}(\pi) = \mathbb{E}_{\tau \sim \mathcal{P}_\pi(\tau)}[G(\tau)] = \sum_\tau G(\tau)\mathcal{P}_\pi(\tau)$$

The central optimization problem in RL can then be expressed by:

The RL Problem

The RL Problem: Find π^* s.t.

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathcal{J}(\pi)$$

with π^* being the optimal policy.

School of Engineering Sciences

Start from what we wish to maximize — $\mathcal{J}(\pi)$

$$\mathcal{J}(\pi) = \mathbb{E}_{\tau\sim\mathcal{P}_\pi(\tau)}[G_1(\tau)] = \mathbb{E}_{\tau\sim\mathcal{P}_\pi(\tau)}\Big[\sum_t R_t\Big]$$

$$= \mathbb{E}_{\tau\sim\mathcal{P}_\pi(\tau)}[R_1 + G_2(\tau)]$$

$$= \underbrace{\mathbb{E}_{(S_0,A_0)\sim\mathcal{P}_\pi(S_0,A_0)}\Big[\overbrace{\mathbb{E}_{(S_1,A_1,\dots)\sim\mathcal{P}(S_1,A_1,\dots)}\big[R_1 + G_2(\tau)|S_0, A_0\big]}^{Def.:\ Q_\pi(S_0,A_0)}\Big]}_{\mathbb{E}_y[\mathbb{E}_x[X|Y]]=\mathbb{E}_x[X]}$$

$$= \mathbb{E}_{(S_0,A_0)\sim\mathcal{P}_\pi(S_0,A_0)}\Big[\underbrace{Q_\pi(S_0, A_0)}_{Def.}\Big]$$

On-Policy Action-Value Function: $Q^{\pi}(s, a)$

The On-Policy Action-Value Function, $Q^{\pi}(s, a)$, which gives the expected return if you start in state s, take an arbitrary action a, and then forever after act according to policy π:

$$Q_{\pi}(s_t, a_t) = \mathbb{E}_{\tau \sim \pi}\Big[G_t(\tau) | S_t = s_t, A_t = a_t\Big]$$

And (s, a) is called a state-action pair.

Optimal Action-Value Function, $Q^{*}(s, a)$

The Optimal Action-Value Function, $Q^{*}(s, a)$, which gives the expected return if you start in state s, take an arbitrary action a, and then forever after act according to the *optimal policy* in the environment:

$$Q^{*}(s_t, a_t) = \mathbb{E}_{\tau \sim \pi}\Big[G_t(\tau) | S_t = s_t, A_t = a_t\Big]$$

Bellman Equation

The Bellman equations for the on-policy value functions are:

$$Q_\pi(s_t, a_t) = \mathbb{E}_{(S_{t+1}, R_{t+1}, A_{t+1}) \sim \mathcal{P}(S_{t+1}, R_{t+1}, A_{t+1})} \left[R_{t+1} + Q_\pi(s_{t+1}, a_{t+1}) | s_t, a_t \right]$$

The Bellman equations for the *optimal* value functions are:

$$Q^*(s_t, a_t) = \mathbb{E}_{S_{t+1} \sim \mathcal{P}(S_{t+1})} \left[r(s_t, a_t) + \max_{a_{t+1}} Q_\pi(S_{t+1}, a_{t+1}) | s_t, a_t \right]$$

School of Engineering Sciences

Q-Function

$$
\begin{aligned}
Q_\pi(s_0, a_0) &= \mathbb{E}_{(S_1,R_1,A_1,\dots)\sim\mathcal{P}(S_1,R_1,A_1,\dots)}\left[R_1 + G_2(\tau)|s_0, a_0\right] \\
&= \mathbb{E}_{(S_1,A_1,R_1)\sim\mathcal{P}(S_1,A_1,R_1)}\left[R_1|s_0, a_0\right] \\
&+ \sum_{s_1,r_1,a_1,\dots\in\mathcal{S},\mathcal{R},\mathcal{A}} p(S_1 = s_1, R_1 = r_1, A_1 = a_0, \dots|s_0, s_0)G_2(\tau) \\
&= \mathbb{E}_{R_1\sim\mathcal{P}(R_1)}\left[R_1|s_0, a_0\right] + \sum_{s_1,r_1} p(s_1, r_1|s_0, a_0)\sum_{a_1} p(a_1|\underbrace{s_0, a_0, s_1, r_1}_{s_1}) \\
&\sum_{s_2,r_2,a_2,\dots} p(s_2, r_2, a_2, \dots|\underbrace{s_0, a_0, s_1, r_1, a_1}_{s_1,a_1})G_2(\tau) \\
&= r(s_0, a_0) \\
&+ \sum_{s_1,r_1} p(s_1, r_1|s_0, a_0)\sum_{a_1}\pi(a_1|s_1)\sum_{s_2,r_2,a_2,\dots} p(s_2, r_2, a_2, \dots|s_1, a_1)G_2(\tau)
\end{aligned}
$$

$$Q_\pi(s_0, a_0) = \mathbb{E}_{(S_1,R_1,A_1,\dots)\sim\mathcal{P}(S_1,R_1,A_1,\dots)}\Big[G_1(\tau)|s_0, a_0\Big]$$
$$= \mathbb{E}_{(S_1,R_1,A_1,\dots)\sim\mathcal{P}(S_1,R_1,A_1,\dots)}\Big[R_1 + G_2(\tau)|s_0, a_0\Big] = r(s_0, a_0)$$
$$+ \sum_{s_1,r_1} p(s_1, r_1|s_0, a_0) \sum_{a_1} \pi(a_1|s_1) \sum_{s_2,r_2,a_2,\dots} p(s_2, r_2, a_2, \dots|s_1, a_1) G_2(\tau)$$

Q Function

$$\sum_{s_1,r_1} p(s_1, r_1|s_0, a_0) \sum_{a_1} \pi(a_1|s_1) \sum_{s_2,r_2,a_2,\dots} p(s_2, r_2, a_2, \dots|s_1, a_1) G_2(\tau)$$
$$= \sum_{s_1,r_1} p(s_1, r_1|\underbrace{s_0, a_0}_{***}) \sum_{a_1} \pi(a_1|s_1) \mathbb{E}_{(S_2,R_2,A_2,\dots)\sim\mathcal{P}(S_2,R_2,A_2,\dots)}\Big[G_2(\tau)|s_1, a_1\Big]$$
$$= \mathbb{E}_{(S_1,R_1,A_1)\sim\mathcal{P}(S_1,R_1,A_1)}\Big[Q_\pi(s_1, a_1)|\underbrace{s_0, a_0}_{***}\Big]$$

Q Function

$$
\begin{aligned}
r(s_t, a_t) &= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\left[R_{t+1}|S_t, A_t\right] \\
&= \mathbb{E}_{(R_{t+1})\sim\mathcal{P}(R_{t+1})}\left[R_{t+1}|S_t, A_t\right]
\end{aligned}
$$

Q Function

$$
\begin{aligned}
&Q_\pi(s_t, a_t) \\
&\quad= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\left[G_{t+1}(\tau_{t+1})|s_t, a_t\right] \text{(By definition)} \\
&\quad= r(s_t, a_t) + \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\left[Q_\pi(s_{t+1}, a_{t+1})|s_t, a_t\right] \\
&\quad= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\left[R_{t+1} + Q_\pi(s_{t+1}, a_{t+1})|s_t, a_t\right]
\end{aligned}
$$

Right or Wrong?

$$
\begin{aligned}
Q_\pi(s_t, a_t) &= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\Big[G_{t+1}(\tau_{t+1})|s_t, a_t\Big] \text{(By definition)}\\
&= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\Big[R_{t+1} + R_{t+2} + ...|s_t, a_t\Big]\\
&= \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\Big[R_{t+1} + G_{t+2}(\tau_{t+2})|s_t, a_t\Big]\\
&= \mathbb{E}\Big[R_{t+1} + \mathbb{E}\Big[G_{t+2}(\tau_{t+2})|s_t, a_t\Big]|s_t, a_t\Big]\\
&\text{(By: } \mathbb{E}_x[\mathbb{E}_x[X|Y]|Y] = \mathbb{E}_x[X|Y])\\
&= \mathbb{E}\Big[R_{t+1} + Q_\pi(s_{t+1}, a_{t+1})|s_t, a_t\Big] \text{ (Right or Wrong ???)}
\end{aligned}
$$

$$
\begin{aligned}
&Q_\pi(s_t, a_t) \\
&\quad = \mathbb{E}_{(S_{t+1},R_{t+1},A_{t+1})\sim\mathcal{P}(S_{t+1},R_{t+1},A_{t+1})}\Big[G_{t+1}(\tau_{t+1})|s_t, a_t\Big] \text{(By definition)} \\
&\quad = \sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1}) \sum_{s_{t+1},r_{t+1}} p(s_{t+1}, r_{t+1}|s_t, a_t)\Big[R_{t+1} + Q_\pi(s_{t+1}, a_{t+1})\Big] \\
&\quad = \underbrace{\sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1})}_{=\,{\color{red}1}} \underbrace{\sum_{s_{t+1}} p(s_{t+1}|\overbrace{r_{t+1}}^{\color{red}\text{Junk}}, s_t, a_t) \overbrace{\sum_{r_{t+1}} p(r_{t+1}|s_t, a_t)R_{t+1}}^{=\ r(s_t,a_t)}}_{\text{Not a function of } s_{t+1}} \\
&\quad + \sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1}) \sum_{s_{t+1},r_{t+1}} p(s_{t+1}, \overbrace{r_{t+1}}^{Junk}|s_t, a_t)\ \underbrace{Q_\pi(s_{t+1}, a_{t+1})}_{\text{Not a function of } r_{t+1}}
\end{aligned}
$$

$$Q_\pi(s_t, a_t) = \underbrace{\sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1})}_{=\,1} \underbrace{\overbrace{\sum_{s_{t+1}} p(s_{t+1}|\overbrace{r_{t+1}}^{\text{Junk}}, s_t, a_t)}^{=\,1} \overbrace{\sum_{r_{t+1}} p(r_{t+1}|s_t, a_t) R_{t+1}}^{=\,r(s_t,a_t)}}_{\text{Not a function of } s_{t+1}}$$

$$+ \sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1}) \sum_{s_{t+1}, r_{t+1}} p(s_{t+1}, \overbrace{r_{t+1}}^{Junk}|s_t, a_t) \underbrace{Q_\pi(s_{t+1}, a_{t+1})}_{\text{Not a function of } r_{t+1}}$$

$$= \underbrace{\mathbb{E}_{S_{t+1}}\Big[r(s_t, a_t)|s_t, a_t\Big]}_{=\,r(s_t,a_t)} + \mathbb{E}_{S_{t+1}}\Big[\mathbb{E}_{A_{t+1}}\Big[Q_\pi(S_{t+1}, A_{t+1})\Big]|s_t, a_t\Big]$$

$$= \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\Big[r(s_t, a_t) + \mathbb{E}_{A_{t+1}\sim\pi(\cdot|S_{t+1})}\Big[Q_\pi(S_{t+1}, A_{t+1})\Big]|s_t, a_t\Big]$$

Max

$$Q_{\pi}(s_t, a_t) = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\Big[r(s_t, a_t)+\mathbb{E}_{A_{t+1}\sim\pi(\cdot|S_{t+1})}\Big[Q_{\pi}(S_{t+1}, A_{t+1})\Big]\Big|s_t, a_t\Big]$$

How to maximize $Q_{\pi}(s_t, a_t)$?

Max

$$Q^{*}(s_t, a_t) \overset{Def}{=} \max_{\pi} Q_{\pi}(s_t, a_t)$$

$Q^{*}(s_t, a_t)$ is not a function of π.

An optimal policy can be found by maximising over $Q^*(s, a)$

$$\pi^*(A = a|S = s) = \begin{cases} 1 & \text{if } a = \text{argmax}_{a \in A} Q^*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

- There is always a deterministic optimal policy for any MDP.
- If we know $Q^*(s, a)$, we immediately have the optimal policy.

Max

$$Q_{\pi}(s_t, a_t) = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\Big[r(s_t, a_t) + \mathbb{E}_{A_{t+1}\sim\pi(\cdot|S_{t+1})}\Big[Q_{\pi}(S_{t+1}, A_{t+1})\Big]\Big|s_t, a_t\Big]$$

Max

$$Q^*(s_t, a_t) = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\Big[r(s_t, a_t) + 1 \times \max_{a_{t+1}} Q_{\pi}(S_{t+1}, A_{t+1} = a_{t+1}) + 0 \times Q_{\pi}(S_{t+1}, A_{t+1} \neq a_{t+1})\Big|s_t, a_t\Big]$$

$$\rightarrow Q^*(s_t, a_t) = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\Big[r(s_t, a_t) + \max_{a_{t+1}} Q_{\pi}(S_{t+1}, a_{t+1})\Big|s_t, a_t\Big]$$

Bellman Equation

$$Q^*(s_t, a_t) = \mathbb{E}_{S_{t+1} \sim \mathcal{P}(S_{t+1})}\left[r(s_t, a_t) + \max_{a_{t+1}} Q_\pi(S_{t+1}, a_{t+1}) \middle| s_t, a_t \right]$$

School of Engineering Sciences

- If we know $Q^*(s, a)$, we immediately have the optimal policy...
- Actually, it's very hard to get Q-function...

Solution

Use a neural network as an universal approximator to estimate Q-function.

Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of *n* real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our results settle an open question about representability in the class of single hidden layer neural networks. In particular, we show that arbitrary decision regions can be arbitrarily well approximated by continuous feedforward neural networks with only a single internal, hidden layer and any continuous sigmoidal nonlinearity. The paper discusses approximation properties of other possible types of nonlinearities that might be implemented by artificial neural networks.

Key words. Neural networks, Approximation, Completeness.

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky, University of Toronto, kriz@cs.utoronto.ca
Ilya Sutskever, University of Toronto, ilya@cs.utoronto.ca
Geoffrey E. Hinton, University of Toronto, hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

School of Engineering Sciences

- NN as an universal approximator (1989)
- AlexNet: First "modern" CNN(2012)

Q-learning

Use a function approximator(e.g. a neural network) to estimate the action-value function:

$$\underbrace{(s, a)}_{\text{state-action pair}} \xrightarrow{input} \text{Neural Network} \xrightarrow{output} Q_{\theta}(s, a) \xrightarrow{approximate} Q^{*}(s, a)$$

- θ: function parameters (e.g. neural network's weights)
- If the function approximator is a deep neural network $\rightarrow$ deep Q-learning(DQN).

School of Engineering Sciences

We want to find a Q-Function that satisfies the Bellman Equation:

Bellman Equation

$$Q^*(s_t, a_t) = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\left[r(s_t, a_t) + \max_{a_{t+1}} Q_\pi(S_{t+1}, a_{t+1}) | s_t, a_t\right]$$

We use a (deep) neural network as a Q-Function approximator.

Forward Pass

Loss Function: $\mathcal{L}_i(\theta_i) = \mathbb{E}_{s,a\sim\rho(\cdot)}\left[\left(y_i - Q_{\theta_i}(s_t, a_t)\right)^2\right]$

where: $y_i = \mathbb{E}_{S_{t+1}\sim\mathcal{P}(S_{t+1})}\left[r(s_t, a_t) + \max_{a_{t+1}} Q_\pi(S_{t+1}, a_{t+1}) | s_t, a_t\right]$

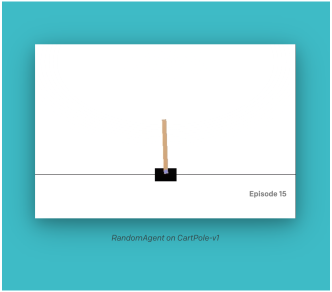
Backward Pass

Gradient Ascent: $Q_{\theta_{i+1}}(s_t, a_t) = Q_{\theta_i}(s_t, a_t) + \alpha\nabla_{\theta_i}\mathcal{L}_i(\theta_i)$

Outline

School of Engineering Sciences

- Objective: Applying forces to a cart moving along a track so as to keep a pole hinged to the cart from falling over.
- State: Raw pixel inputs of the game state.
- Action: Move left or right.
- Reward:
    - +1 for every time step on which failure did not occur.
    - -1 on each failure and 0 at all other times.



School of Engineering Sciences

School of Engineering Sciences

School of Engineering Sciences

From Zero to AlphaGo Zero

An Odyssey.

School of Engineering Sciences