UMassAmherst
The Commonwealth's Flagship Campus

Deep RL Intro

Chaitra Gopalappa

Reference

- Chapters 9, 10, 11 of Sutton and Barto
  - Provide general understanding of function approximation
- Practical advances over past decade
  - Research articles best source
- Chapters 6 and 7, Lapan
  - Focusses on problems needing high-computational resources (1-2 days to converge)
  - Review it for computational efficiency
    - Use of wrappers to simplify code writing
    - Use of PTAN libraries for simplifying functionality
  - For this class: I will instead use simple example (computationally doable)

Recollect Q-learning
(also called off-policy temporal difference (TD) control)

- Set step size: $\alpha \in (0,1], small\ \epsilon > 0$
- Set episode length; If there is absorbing state, episode ends if reaches terminal state or reaches episode length
- Initialize $Q(s,a), \forall (s,a) pairs$; except $Q(terminal\ state, .) = 0$
- Loop for each episode
  - Initialize state, say s
  - Loop for each step of episode (until episode length or terminal state)
    - Choose action a given state s, using **action selection method**
    - Take action a and observe r, s'
    - Update $Q(s,a) = Q(s,a) + \alpha \left[r(s,a,s') + \lambda \max_{a'} Q(s',a') - Q(s,a) \right]$
    - Set $s \leftarrow s'$
    - **Update α**

Q-Learning uses Q-table

	a=0	a=1	a=2
s=0	13774.6	13760.6	14085
s=1	14187.7	14183.6	14478.7
s=2	14558.7	14522.4	14824.2
s=3	14950.3	14886.4	15095.4
s=4	15307.8	15218.8	15229.7

Overview TabularRL v DeepRL

function approximation

Tabular methods

Q-Learning

	a=0	a=1	a=2
s=0	13774.6	13760.6	14085
s=1	14187.7	14183.6	14478.7
s=2	14558.7	14522.4	14824.2
s=3	14950.3	14886.4	15095.4
s=4	15307.8	15218.8	15229.7

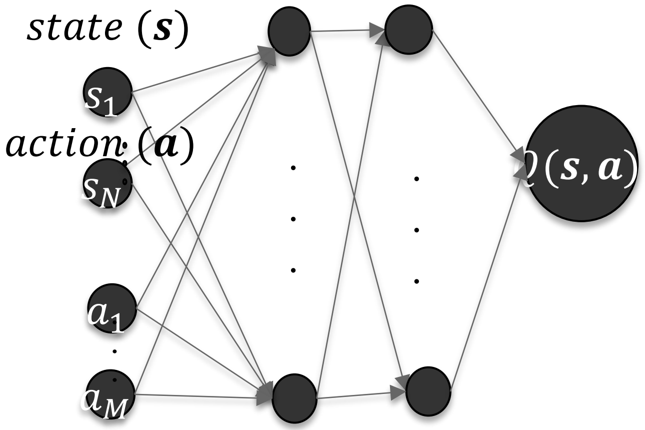
Q-Learning with function approximation

If analytical form is known, solve for coefficients ($\mathbf{w}$) using Widrow-Hoff

$$\tilde{Q}(s, A = a; \mathbf{w}) = w_1 s^2 + w_2 s^3$$

DeepRL

If analytical form is not known, use deep neural network (DNN) for function approximation of either **state-value function (v), action value function (Q) or policy function (π)**

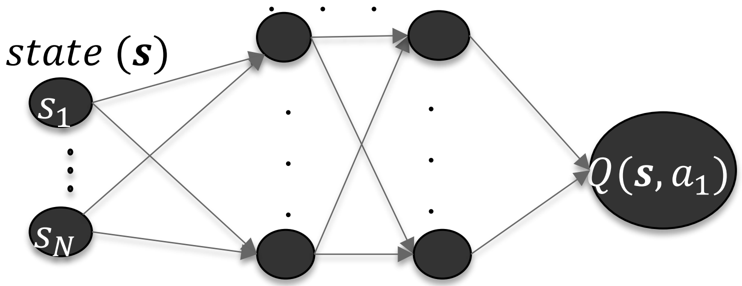


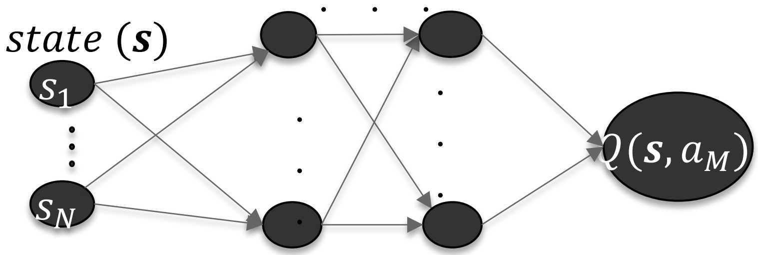
Q-learning with function approximation using W-H

- Suppose we know functional form
  - $\tilde{Q}(s, a = 1; \boldsymbol{w}) = w_1 s^2 + w_2 s^3$
  - **Note: addition of *w*** in Q-value to represent coefficient of the function approximation
  - Apply incremental Widrow-Hoff
  - Recollect W-H uses steepest descent (SD)
    - **Objective (loss function E)**
    - **E** $= \boldsymbol{Min_w}\left[\tilde{Q}(s, a = 1; \boldsymbol{w}) - Q(s, a = 1)\right]^2$ **for every** $a \in A$
      - $\tilde{Q}(s, a = 1; \boldsymbol{w})$ **estimated Q**
      - $Q(s, a = 1)$ **actual Q**
    - Bellman equation: $Q(s,a) = (1 - \alpha)Q(s,a) + \alpha\left[r + \lambda \max_{a'} Q(s', a')\right]$
    - **Main transformation:** $\boldsymbol{w} \leftarrow \boldsymbol{w} - \mu \frac{\partial E}{\partial \boldsymbol{w}}$

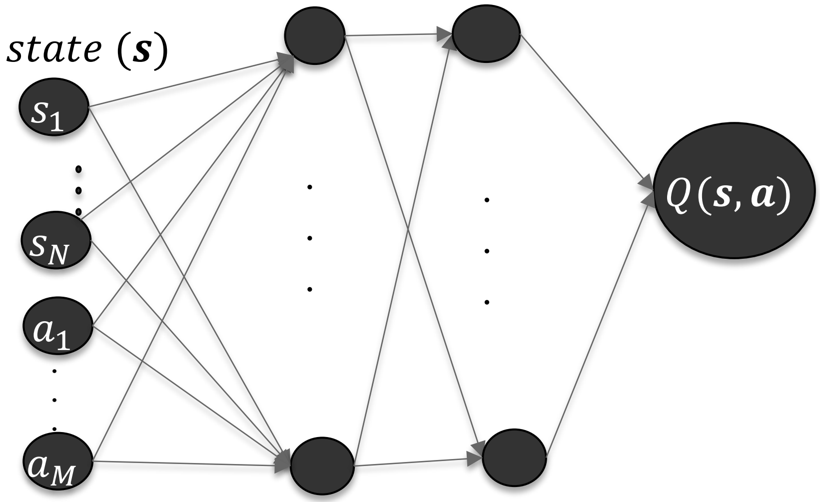
Deep RL- earlier architectures

One network for each action





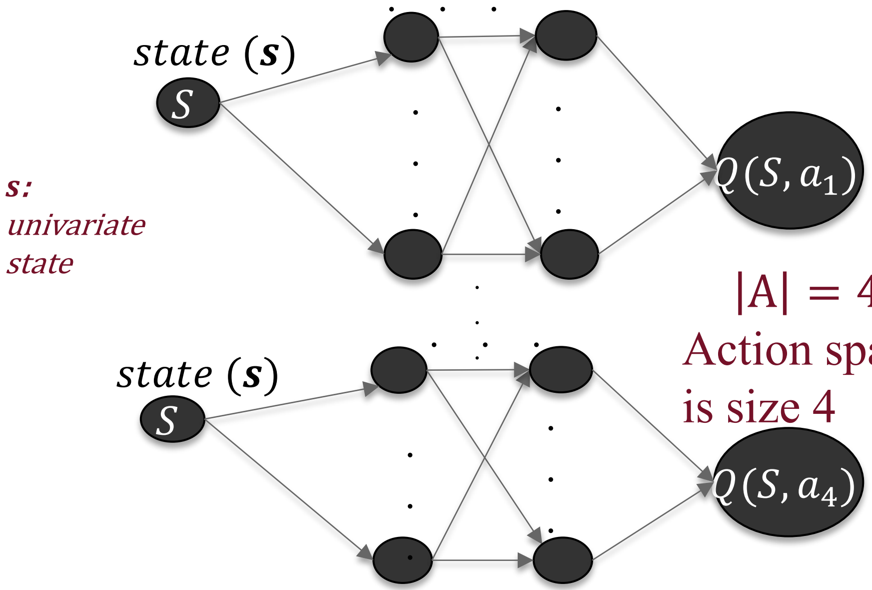
Include action in input layer



$\boldsymbol{s}$: state vector $\quad$ $\boldsymbol{a}$: action vector

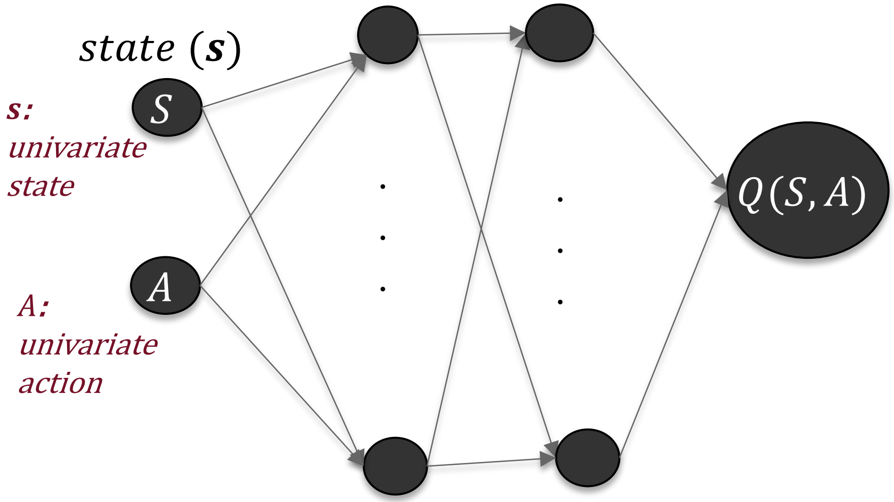
$\boldsymbol{s} = [s_1, \dots s_N]$ $\quad$ $\boldsymbol{a} = [a_1, \dots a_M]$

Example of uni-variate state and action



$|\mathrm{A}| = 4$
Action space is size 4

Include action in input layer

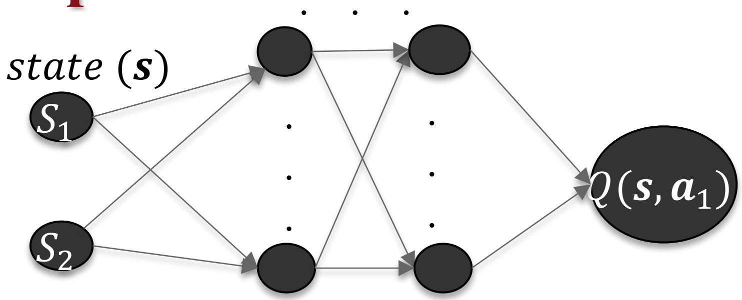


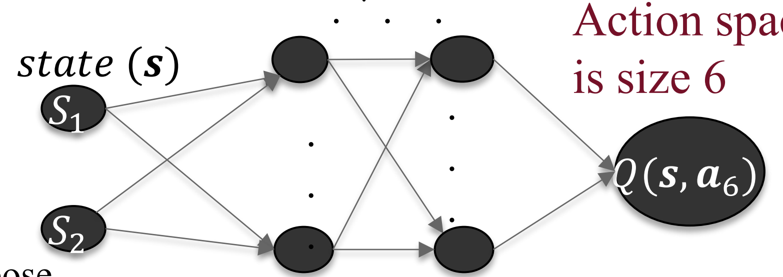
Suppose,
X_t =proportion of people with active infection
D_t =how often to test
State space: $\mathrm{S} \in \mathbb{R}^1; \mathrm{S} \in [0,1]$
Action space:A ={test once a week, twice a week, three times a week, daily}

Action space:$\mathrm{A} \in \mathbb{R}^1; A \in [1,30]$

Example of multi-variate state and action

***s*: vector of 2 variates**





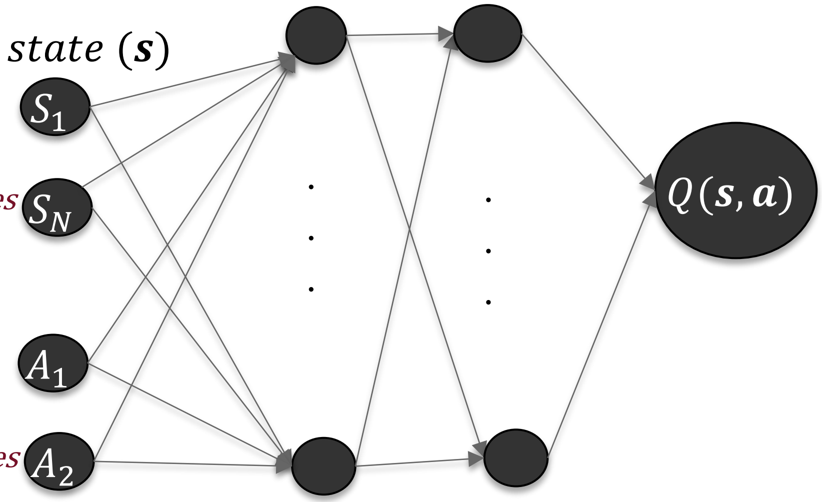
$|\mathrm{A}| = 6$
Action space is size 6

Suppose,
X_t =[proportion with active infection, proportion recovered]
D_t =[how often to test in a week, what %lockdown]
State space: S ∈ $\mathbb{R}^2$; (vector of 2 real random variables)
Action space:A ={[once, 25%], [twice,25%], [thrice, 25%], [once, 50%], [twice,50%], [thrice, 50%] }

Include action in input layer

***s*: vector of 2 variates**

***a*: vector of 2 variates**



Action space: $\boldsymbol{a} \in \mathbb{R}^2$; (vector of 2 real random variables)

$\boldsymbol{s}$*: state vector*
$\boldsymbol{s} = [S_1, \ldots S_N]$

$\boldsymbol{a}$*: action vector*
$\boldsymbol{a} = [A_1, \ldots A_M]$

Notations

- Random variables in Capital
- Vector in bold small

$\boldsymbol{s}$: *state vector*

$\boldsymbol{s} = [S_1, \ldots S_N]$

$\boldsymbol{a}$: *action vector*

$\boldsymbol{a} = [A_1, \ldots A_M]$

Challenges with these earlier architectures of Deep RL

- Disadvantages of earlier architectures?
- If action space is large, computationally burdensome
  - Train a separate network for each action
  - Need to do a forward pass for each action

UMassAmherst
The Commonwealth's Flagship Campus