Q function Approximation

Wednesday, 28 September 2022 2:55 PM

Problems with Q-Tables

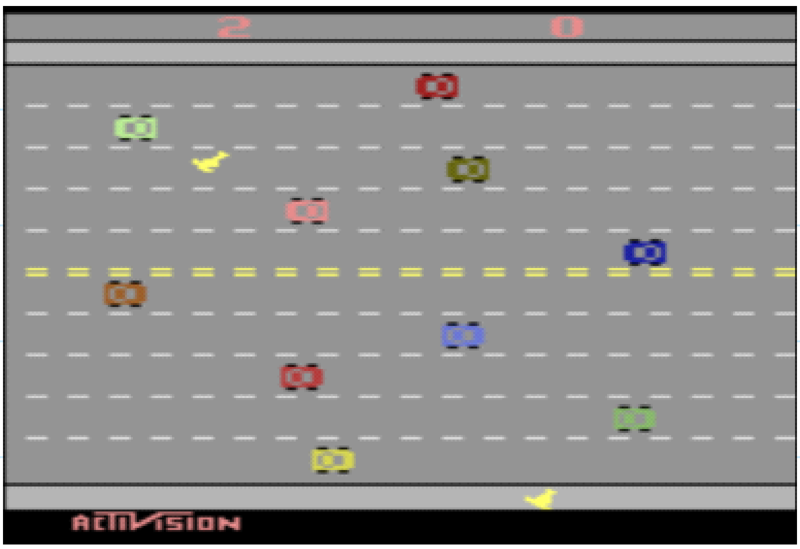
- Size = # states × # actions

  Can get too large very quickl

- Learning Q values for all (s,a) requires multiple visits of all (s,a)

  So it can take a long time to update

e.g : Freeway game



- 12 rows × 40 cols = 480 positions
- each position can either have a car or not = 2_{480} car configurations
- ∴ State space size = 480×2^{480} Too big!

Q function approximation

- Doesn't require a Q-table

  (Does require extra space but much cheaper)

- All Q values get updated each step, even for

(s,a) we've never seen before

Feature representation of states

Feature: Represents some meaningful info about

states

e.g.) In freeway we can use 3 features to represent states

- # rows above kangaroo f_0

- # cols left or right closest car in above row is f_1

- # cols left or right closest car in below row is f_2

Ideally features are efficiently computable

$$\underset{\sim}{f}(s) = [f_0(s), f_1(s), f_2(s), \ldots, f_n(s)]$$

Aside: Math notes

x_0
x_1

Aside: Math notes

Vector: $\underset{\sim}{x} = (x_0 \; x_1 \; x_2 \ldots x_n)$ OR $\begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$

Essentially a 1-D array

Vector multiplication aka dot product: $\underset{\sim}{x} \cdot \underset{\sim}{y}$

Must be same size

$$= \sum_{i=0}^{n} x_i \times y_i = x_0 y_0 + x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$$

Linear equations with vectors: $y = \underset{\sim}{w} \cdot \underset{\sim}{x}$

$\underset{\sim}{w}$: weight vector $\quad \underset{\sim}{x}$: data features

Linear Q-function Representation

Can calculate Q values as a linear representation of state features

$Q(s,a) = \underset{\sim}{w_a} \cdot \underset{\sim}{f}(s)$

$\underset{\sim}{w_a}$ is the weight vector for action a (have to learn)

$\underset{\sim}{w_a} = (w_{0,a} \; w_{1,a} \; w_{2,a} \ldots w_{n,a})$ Initialise arbitrarily i.e all 0

$|\underset{\sim}{w_a}| = \#$ features

However! In this subject we don't consider $\underset{\sim}{w_a}$ separately for each action. We only consider $\underset{\sim}{w}$

Set an order of actions. eg) a_0 = Up a_1 = Down a_2 = left a_3 = right

$\underset{\sim}{w}$ = concatenation of all $\underset{\sim}{w}_a$

$$\underset{\sim}{w} = (w_{0,a_0}\ w_{1,a_0} \cdots w_{m,a_0}\ w_{0,a_1}\ w_{2,a_1} \ldots w_{1,a_n} \ldots w_{m,a_n})$$

$|\underset{\sim}{w}|$ = size of $\underset{\sim}{w}$ = # features × # actions

Note: can't do $Q(s,a) = \underset{\sim}{w} \cdot \underset{\sim}{f}(s)$ since $|\underset{\sim}{w}| \neq |\underset{\sim}{f}(s)|$

Need to define a feature vector $\underset{\sim}{f}(s,a)$ with the same size as $|\underset{\sim}{w}|$

To construct $\underset{\sim}{f}(s,a)$:

- Initialise a vector of size $|\underset{\sim}{w}|$ full of zeroes
- Replace indices associated with action a with $\underset{\sim}{f}(s)$

eg) $\underset{\sim}{w} = (w_{0,up}\ w_{1,up}\ \ w_{0,down}\ w_{1,down})$ 2 features 2 actions

$\underset{\sim}{f}(s) = (13 \quad 24)$

$\underset{\sim}{f}(s, down) = (0 \ 0 \ 13 \ 24)$

Finally: $Q(s,a) = \underset{\sim}{w} \cdot \underset{\sim}{f}(s,a)$

- Only need to store $\underset{\sim}{w}$ of size #actions × #features
- $f(s,a)$ calculated on the fly

$Q(s, down) = w_{0,down} \times 13 + w_{1,down} \times 24$

Other weights multiply by 0

Weight update

$$w_i \leftarrow w_i + \alpha \cdot [r + \gamma Q(s', a') - Q(s,a)] \cdot f_i(s,a)$$

$Q(s', a')$ same as Q-learning / SARSA

- Only weights associated with action are updated