QFL

Quioto Forest Legends

A B C

time up → loss
0 eggs → lois
100 eggs → win
must get 10 every 4 skirmishes

eggs	ticks
20	24
	~~23~~
25	21
27	18
→ 32	16
32	15
32	14
32	13
32	12
DEFENSE	WINS

$Q((1,1,1,10),0) = .9^{10}$

0 1 2 3

Images by eBay user furret_inc https://www.ebay.com/itm/OLD-Original-Vintage-Pokemon-10-Card-LOT-1st-Edition-Holo-Rare-Played/233176095585?hash=item364a60db61:g:3OQAAOSwG7NclWhm
Used under Fair Use Doctrine

OFFENSE

		Hitmonchan (3 kicks except as noted)	Machamp	Magneton	Mewtwo
Chansey	1-4	2	5	19	0
	5	8	12	-100	4
	6-10	5	5	22	-100
	11-12	-1	-100	47	47
	13-20	3	0	0	0
Charizard	1-4	1	4	11	48
	5	14	32	-100	-9
	6-10	2	5	14	41
	11-12	-4	0	-9	-100
	13-20	2	0	0	0
Clefairy	1-4	1	21	-7	53
	5	4	-4	8	-100
	6-10	2	0	15	-11
	11-12	1	46	-8	-100
	13-20	-3	5	18	0

DEFENSE

4 ticks
1 tick
otherwise 2 ticks

states encapsulate (yards to go , downs left , yards to go to 1st down , ticks left)

eggs needed to win , skirmishes left to get 10 eggs , eggs needed for current set of 4 skirmishes

possible values 101 , 5 , 99 , 25

total # of states ~1.25 million

$V(s)$ = P(offense wins in state s)

matrix for each state

	A	B	C
0	–	–	–
1	–	–	–
2	–	–	–
3	–	–	–

$$\sum_{s \to s'} P(s \to s') \cdot V(s')$$

value of equilibrium is $V(s)$

compute $V(s)$ in order of ↑ time left

Q Learning

$Q(s,a)$ = value of taking action a in state s
(expected reward from taking action a in state s
+ expected discounted future reward from resulting state)

$V(s) = \max_a Q(s,a)$

initialize $Q(s,a) = 0$ for all s,a (viewing reward as earned on transition into winning states)

while not done

$s \leftarrow s_0$ initial state

while s not terminal

choose action a — ε-greedy: $1-\varepsilon$ greedy: pick a to maximize $Q(s,a)$; ε pick random action

observe transition (s, a, r, s') — observed reward

update $Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \cdot \max_a Q(s',a) - Q(s,a)\right)$ — error

α: learning rate (should ↓ as episodes ↑)

episode

Function Approximators

— instead of learning $Q(s,a)$ for every (s,a), learn fxn to approximate them

Linear Approximator

Define features of states or (state, action) pairs

- time left
- yards left / ticks left
- yards-to-first / downs left

(fxns of state)

good features to start with

$f_1(s,a)$: on pace to earn upper bonus (1.0 have already earned … 0.0 no chance)

$f_2(s,a)$: is chance unused

$f_3(s,a)$: both LS, SS unused

$f_4(s,a)$: Y unused

let $f(s)$ be fxn from state to $[0,1]$

$$f_a(s,a') = \begin{cases} f(s) & \text{if } a'=a \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{Q}(s,a) = \underbrace{w_1 \cdot f_1(s,a) + w_2 \cdot f_2(s,a) + \cdots + w_n \cdot f(s,a)}_{\text{for nonterminal } s} + \underbrace{w_{n+1} \cdot 1}_{\text{constant term}}$$

learn w_i instead of $Q(s,a)$ directly

$\hat{Q}(s,a) = 0$ if s is terminal

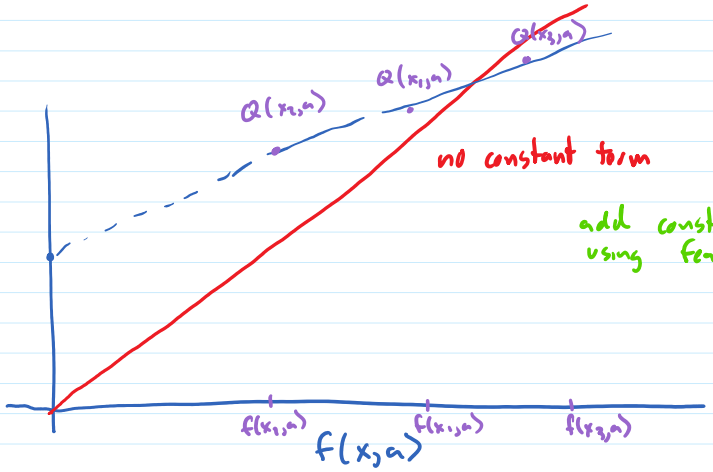
while not done
  $s \leftarrow s_0$
  while not terminal
    In state s

    Choose action a

    Observe transition (s, a, r, s')

    Update ~~$Q(s,a) \leftarrow Q(s,a) + \alpha(\max_{a'} Q(s',a') - Q(s,a))$~~

    for each feature $w_i \leftarrow w_i + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right) \cdot f_i(s,a)$



no constant term

add constant term using feature that's always 1

quantized features: $f_{\text{short-yardage}}(s) = \begin{cases} 1 & \text{if yards-to-first/downs-left} \le 2 \\ 0 & \text{otherwise} \end{cases}$

$$f_1(s) = \begin{cases} 0 & \text{if yards-to-first/downs-left} \le 2 \\ 1 & \text{if between 2 and 6} \\ 2 & \text{if " " } \ge 6 \end{cases}$$

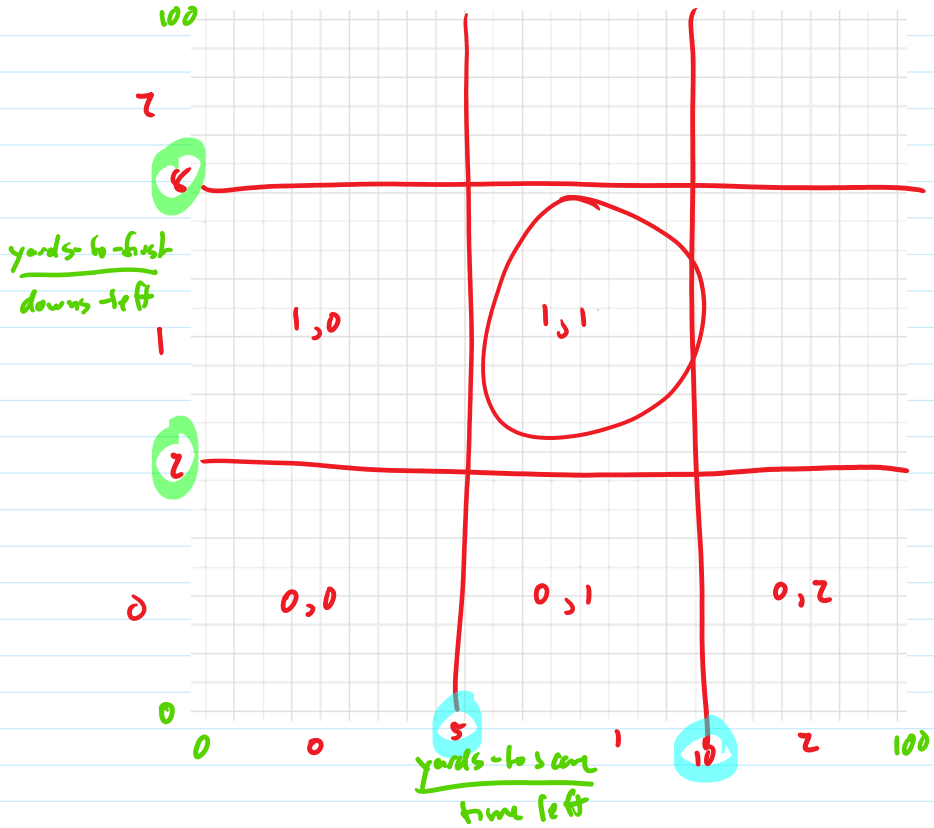
≥ 8

$f_2(s) = \begin{cases} 0 \\ 1 \\ 2 \end{cases}$ yards-to-score / time left ≤ 5

≥ 10

buckets

Buckets



$f_{0,0} = \begin{cases} 1 & \text{if } r_1 \le 2 \text{ and } r_2 \le 5 \\ 0 & \text{otherwise} \end{cases}$

$f_{1,1} = \begin{cases} 1 & \text{if } 2 < r_1 < 8 \text{ and } 5 < r_2 < 10 \\ 0 & \text{otherwise} \end{cases}$

⋮

equivalent to Q-learning except using superstates instead of states

(terminal states in their own bucket)

while not done

  $s \leftarrow s_0$

  while s not terminal

    choose action a

    observe transition (s, a, r, s')

    compute superstate (bucket) $\hat{s}$ for s, $\hat{s}'$ for s'

    update $Q(\hat{s}, a) \leftarrow Q(\hat{s}, a) + \alpha \left(r + \gamma \cdot \max_a Q(\hat{s}', a) - Q(\hat{s}, a) \right)$