Reinforcement Learning
Lecture 2

Gillian Hayes

11th January 2007

School of informatics

Reinforcement Learning: How Does It Work?

We detect a **state**

We choose an **action**

We get a **reward**

Our aim is to learn a **policy** – what action to choose in what state to get maximum reward

Maximum reward over the *long term*, not necessarily *immediate* maximum reward – watch TV now, panic over homework later vs. do homework now, watch TV while all your pals are panicking...

Bandit Problems

N-armed bandits – as in slot machines

– action selection

– evaluation

- Action-values – Q: how good (in the long term) it is to do this action in this situation, Q(s,a)
- Estimating Q
- How to select an action
- Evaluation vs. instruction
  - Evaluation tells you how well you did after choosing an action
  - Instruction tells you what the right thing to do was – make your action more like that next time!

Evaluation vs Instruction

RL – Training information *evaluates* the *action*. Doesn't say whether it was best or correct. Relative to all other actions – must try them all and compare to see which is best

Supervised – Training *instructs* – it gives the *correct answer* regardless of the action chosen. So there is no search in the action space in supervised learning (though may need to search parameters, e.g. neural network weights)

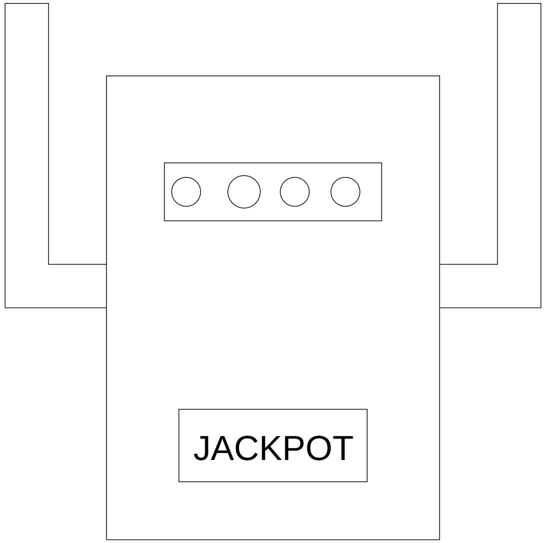
- So RL needs trial-and-error search
- must try all actions
- feedback is a scalar – other actions could be better (or worse)
- learning by selection – selectively choose those actions that prove to be better

What about GAGP?

What Is a Bandit Problem?

Just one state, always the same

Non-associative, not mapping $S \rightarrow A$ (since just one $s \in S$)



N-armed bandit:

- N levers (actions) – choose one
- Each has scalar reward (coins – or not) which is...
- Chosen from probability distribution

Aim

- Maximise expected total reward over time T, e.g. some number of plays

Which lever is best?

The Action Value Q

- Q $=$ value of an action – the Expected or Mean reward from that action
- If Q-value known exactly, always choose that action with highest Q

BUT, only have estimates of Q – build up these estimates from experience of rewards

- **Greedy** action(s): have highest estimated Q: EXPLOITATION
- Other actions: lower estimated Qs: EXPLORATION

Maximise expected reward on 1 play vs. over long time?

Uncertainty in our estimates of values of Q

EXPLORATION VS. EXPLOITATION TRADEOFF

Can't exploit all the time; must sometimes explore to see if an action that currently looks bad eventually turns out to be good

How Do We Estimate Q?

True value $Q^*(a)$ of action a

Estimated value $Q_t(a)$ at play/time t

Suppose we choose action a k_a times and observe a reward r_i on play i:

Then we can estimate Q^* from running mean: $Q_t(a) = \frac{r_1+r_2+r_3+\cdots+r_{k_a}}{k_a}$

If $k_a = 0, r_0 = 0$

As $k_a \to \infty,\ Q_t(a) \to Q^*(a)$

Sample-average method of calculating Q.

* in this case means "true value": $Q^*(a)$. Sometimes write $\hat{Q}$ as estimated value

Action Selection

Greedy: select the action a^* for which Q is highest:

$$Q_t(a^*) = \max_a Q_t(a)$$

So $a^* = \arg\max_a Q_t(a)$ – and * means "best"

Example: 10-armed bandit

Snapshot at time t for actions 1 to 10

$Q_t(a) \rightarrow$

0	0.3	0.1	0.1	0.4	0.05	0	0	0.05	0

$Q_t(a^*) = 0.4$ and $a^* = ?$

Maximises reward

ϵ-**greedy**: Select *random* action ϵ of the time, else select greedy action

Sample all actions infinitely many times

So as $k_a \to \infty$, Qs converge to Q^*

Can reduce ϵ over time

NB: Difference between $Q^*(a)$ and $Q(a^*)$ (but we are following the Sutton and Barto notation)

ϵ-Greedy vs. Greedy

- What if reward variance is larger?
- What if reward variance is very small, e.g. zero?
- What if task is nonstationary?

Which would be better in each of these cases?

Exploration and Exploitation again

Softmax Action Selection

ϵ-greedy: even if worst action is very bad, it will still be chosen with same probability as second-best – we may not want this. So:

Vary selection probability as a function of estimated goodness

Choose a at time t from among the n actions with probability

$$\frac{\exp(Q_t(a)/\tau)}{\sum_{b=1}^{n}\exp(Q_t(b)/\tau)}$$

Gibbs/Boltzmann distribution, τ is temperature (from physics)

Softmax Action Selection

Drawback of softmax? What if our estimate of the value of $Q(a^*)$ is initially very low?

Effect of $\mid \tau \mid$

As $\tau \rightarrow \infty$, probability $\rightarrow 1/n$

As $\tau \rightarrow 0$, probability $\rightarrow$ greedy

Incremental Update Equations

Estimate Q^* from running mean: $Q(a) = \frac{r_1+r_2+r_3+\cdots+r_{k_a}}{k_a}$ if we've tried action a k_a times

Incremental calculation:

$$Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i \tag{1}$$

$$= Q_k + \frac{1}{k+1}[r_{k+1} - Q_k] \tag{2}$$

NewEstimate $=$ OldEstimate $+$ StepSize [Target - OldEstimate]

Incremental Update Equations

This general form will be met often:

NewEstimate = OldEstimate + StepSize [Target - OldEstimate]

Step size α depends on k in incremental equation: $\alpha_k = 1/k$

But is often kept constant, e.g. $\alpha = 0.1$

(gives more weight to recent rewards – why might this be useful?)

Effect of Initial Values of Q

We arbitrarily set the initial values of Q to be zero.

Our estimates are biassed by initial estimate of Q

Can use this to include domain knowledge

Example Set all Q values very high – optimistic

Initial actual rewards are disappointing compared to estimate, so switch to another action – exploration

Temporary effect

Policy

Once we've learnt the Q values, our policy is the greedy one: choose the action with the highest Q

Application

Drug trials. You have a limited number of trials, several drugs, and need to choose the best of them. Bandit arm $\approx$ drug

Define a measure of success/failure – the reward

Measure how well the patients do on each drug – estimating the Q values

Ethical clinical trials – how do we allocate patients to drug treatments? During the trial we may find that some drugs work better than others.

- Fixed allocation design: allocate $1/k$ of the patients to each of the k drugs

- Adaptive allocation design: if the patients on one drug appear to be doing worse, switch them to the other drugs – equivalent to removing one of the arms of the bandit

See: http://www.eecs.umich.edu/∼qstout/AdaptSample.html

And: J.Hardwick, R.Oehmke,Q.Stout: A program for sequential allocation of three Bernoulli populations, Computational Statistics and Data Analysis 31, 397–416, 1999. (just scan this one)

Reading: Sutton and Barto Chapter 2.

Next: Reinforcement Learning with more than one state.