Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 7

Sep, 18, 2019

Course Announcements

<u>Assignment 1 has been posted (due Sept 27)</u>

- ValueOf Info and ValueOfControl
- MDPs: Value Iteration
- POMDPs: Belief State Update

422 big picture

StarAI (statistical relational AI)

Hybrid: Det +Sto

Prob CFG Parsing

Prob Relational Models

Markov Logics

	Deterministic	Stochastic
Query	*Logics* *First Order Logics* *Ontologies* • Full Resolution • SAT	***Belief Nets*** Approx. : Gibbs ***Markov Chains and HMMs*** Forward, Viterbi…. Approx. : Particle Filtering ***Undirected Graphical Models*** ***Markov Networks*** ***Conditional Random Fields***
Planning		***Markov Decision Processes and Partially Observable MDP*** • Value Iteration • Approx. Inference ***Reinforcement Learning***

Applications of AI

Representation

Reasoning Technique

Lecture Overview

- Start Reinforcement Learning
  - Start Q-learning
  - Estimate by Temporal Differences

MDP and Reinforcement Learning

Markov decision process

- Set of **states** S, set of **actions** A
- **Transition** probabilities to next states P(s'| s, a')
- **Reward** function R(s) or R(s, a) or R(s, a, s')

RL is based on MDPs, but

- Transition model is **not known**
- Reward model is **not known**

While for **MDPs** we can ***compute*** an optimal policy

RL ***learns*** an optimal policy

Search-Based Approaches to RL

Policy Search (stochastic local search)

- Start with an arbitrary policy
- To evaluate a policy, try it out in the world
- Generate some neighbours…..

Problems with evolutionary algorithms

- **Policy space can be huge**: with *n* states and *m* actions there are …… policies

- **Policies are evaluated as a whole**: cannot directly take into account locally good/bad behaviours

Q-learning

Contrary to search-based approaches**, Q-learning learns after every action**

Learns components of a policy, rather than the policy itself

Q(s,a) = expected value of doing action ***a*** in state ***s*** and then following the **optimal policy**

Q values

$$Q(s,a) \;=\; R(s) + \gamma \sum_{s'} P(s'|\,s,a) V^{\pi^*}(s') \qquad (1)$$

Q(s,a) are known as **Q-values**, and are related to the utility of state ***s*** as follows

$$V^{\pi^*}(s) = \qquad (2)$$

From (1) and (2) we obtain a constraint between the *Q* value in state *s* and the *Q* value of the states reachable from *a*

$$Q(s,a) \;=$$

Average Through Time

Suppose we have a sequence of values (your sample data):

$$v_1,\ v_2,\ ..,\ v_k$$

And want a running approximation of their expected value

- e.g., given sequence of grades, estimate expected value of next grade

A reasonable **estimate** is the average of the first *k* values:

$$A_k = \frac{v_1 + v_2 + \dots. + v_k}{k}$$

Average Through Time

$$A_k = \frac{v_1 + v_2 + \ldots. + v_k}{k}$$

$$kA_k = v_1 + v_2 + \ldots. + v_{k-1} + v_k$$ and equivalently for k-1:

$$(k-1)A_{k-1} = v_1 + v_2 + \ldots. + v_{k-1}$$ which replaced in the equation above gives

$$kA_k = (k-1)A_{k-1} + v_k$$ Dividing by k we get:

$$A_k = (1 - \frac{1}{k})A_{k-1} + \frac{v_k}{k}$$

and if we set $\alpha_k = 1/k$

$$A_k = (1 - \alpha_k)A_{k-1} + \alpha_k v_k = A_{k-1} + \alpha_k A_{k-1} + \alpha_k v_k$$

$$= A_{k-1} + \alpha_k (v_k - A_{k-1})$$

Estimate by Temporal Differences

$$A_k = A_{k-1} + \alpha_k (v_k - A_{k-1})$$

NEW ESTIMATE

PREVIOUS ESTIMATE

NEW VALUE

$(v_k - A_{k-1})$ is called a ***temporal difference error*** or ***TD-error***

- it specifies how different the new value v_k is from the prediction given by the previous running average A_{k-1}

The new estimate (average) is obtained by updating the previous average by α_k times the TD error

Q-learning: General Idea

- Learn from the ***history*** of interaction with the environment, *i.e.*, a sequence of state-action-rewards

  $$<s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, ...>$$

- History is seen as sequence of ***experiences***, i.e., tuples

  $<s, a, r, s'>$

  - agent doing action *a* in state *s*,
  - receiving reward *r* and ending up in *s'*

- These experiences are used to estimate the value of *Q (s,a)* expressed as

Q-learning: General Idea

But ..

$$Q(s,a) = r + \gamma \max_{a'} Q[s',a']$$

Is an **approximation**.

The real link between Q(s,a) and Q(s',a') is

$$Q(s,a) \;=\; r + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q(s',a')$$

Q-learning: Main steps

Store ***Q[S, A]***, for every state *S* and action *A* in the world

Start with **arbitrary estimates** in $Q^{(0)}[S, A]$,

Update them by using experiences

- Each **experience** <*s, a, r, s'*> provides one new data point on the actual value of *Q[s, a]*

$$Q[s,a] = r + \gamma \max_{a'} Q[s',a']$$

current *estimated* value of Q[s',a'], where s' is the state the agent arrives to in the current experience

New value of Q[s,a],

Q-learning: Update step

$$A_k = A_{k-1} + \alpha_k (v_k - A_{k-1})$$

NEW ESTIMATE

PREVIOUS ESTIMATE

NEW VALUE

- *TD* formula applied to Q[s,a]

$$Q^{(i)}[s,a] \leftarrow Q^{(i-1)}[s,a] + \alpha_k ((r + \gamma \max_{a'} Q^{(i-1)}[s',a']) - Q^{(i-1)}[s,a])$$

updated *estimated* value of Q[s,a]

New value for Q[s,a] from <s,a,r,s'>

Previous *estimated* value of Q[s,a]

Q-learning: algorithm

controller Q-learning(S,A)
inputs:
- S is a set of states
- A is a set of actions
- γ the discount
- α is the step size

internal state:
- real array $Q[S,A]$
- previous state s
- previous action a

begin
- initialize $Q[S,A]$ arbitrarily
- observe current state s
- **repeat forever:**
  - select and carry out an action a
  - observe reward r and state s'
  - $Q[s,a] \leftarrow Q[s,a] + \alpha\,(r + \gamma \max_{a'} Q[s',a'] - Q[s,a])$
  - $s \leftarrow s'$;
- **end-repeat**

end

Learning Goals for today's class

You can:

- Describe and criticize search-based approaches to RL
- Motivate Q-learning
- Justify Estimate by Temporal Differences
- Explain, trace and implement Q-learning

TODO for Fri

• **Do Practice Ex. On Reinforcement Learning:**

Exercise 11.A: Q-learning

•http://www.aispace.org/exercises.shtml

Keep working on assignment 1 !