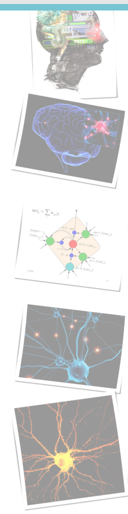
Attending Over Beliefs

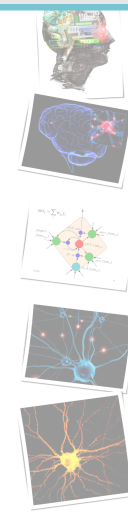
Spyros Samothrakis
Research Fellow, IADS
Univerisity of Essex

November 20, 2015



From Reinforcement Learning to Beliefs

From Beliefs to Attention Networks



What is Reinforcement Learning?

- *Reinforcement learning is the study of how animals and artificial systems can learn to optimize their behavior in the face of rewards and punishments* – Peter Dyan, Encyclopedia of Cognitive Science
- **Not** supervised learning - the animal/agent is not provided with examples of optimal behaviour, it has to be discovered!
- **Not** unsupervised learning either - we have more guidance than just observations

Links to other fields

- It subsumes most artificial intelligence problems
- Forms the basis of most modern “intelligent agent” frameworks
- Ideas drawn from a wide range of contexts, including psychology (e.g., Skinner’s “Operant Conditioning”), philosophy, neuroscience, operations research, **Cybernetics**
- This will form the main framework our talk - no need to understand everything
- Some of the things I am going to say might sound too introductory for someone familiar with RL

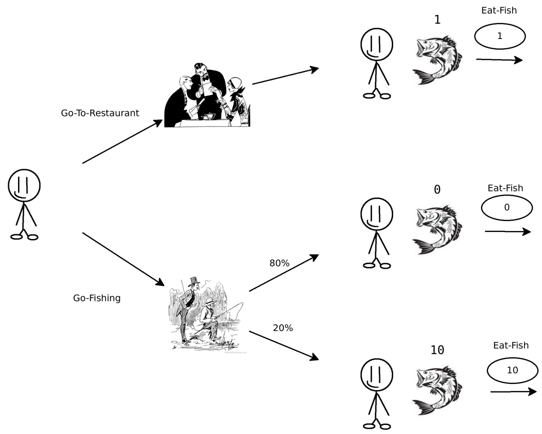
The Markov Decision Process

- The primary abstraction we are going to work with is the Markov Decision Process (MDP).
- MDPs capture the dynamics of a mini-world/universe/environment
- An MDP is defined as a tuple $< S, A, T, R, \gamma >$ where:
  - S, $s \in S$ is a set of states
  - A, $a \in A$ is a set of actions
  - $R : S \times A$, $R(s, a)$ is a function that maps state-actions to rewards
  - $T : S \times S \times A$, with $T(s'|s, a)$ being the probability of an agent moving from state s to state s' after taking a
  - γ is a discount factor - the impact of time on rewards

Policy

- The MDP (the world) is populated by an agent (an actor)
- You can take actions (e.g., move around, move blocks)
- The type of actions you take under a state is called the *policy*
- $\pi : S \times A$, $\pi(s, a) = P(a|s)$, a probabilistic mapping between states and actions
- Finding an optimal policy is *mostly* what the RL problem is all about
- Goal is to maximise long term reward
- $J(\theta) = E_\pi\{\sum_{t=0}^{\infty} R(s, a)\}$

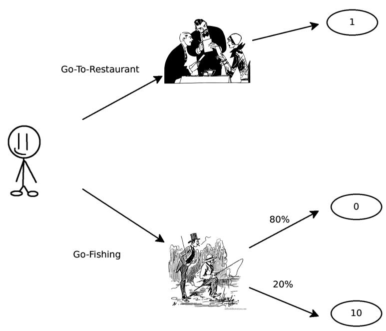
Fishing Toon: Pictorial Depiction



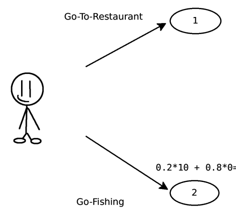
Expected Reward

- Our toon has to choose between two different actions
- `Go-To-Restaurant` or `Go-Fishing`
- We assume that toon is interested in maximising *the expected sum* of happiness/reward
- We can help the toon reason using the tree backwards

Reasoning Backwards (1)



Reasoning Backwards (2)



Correct Action

- Toon should go `Go-Fishing`
- **Would you do the same?**
- **Would a pessimist toon do the same?**
- We just went through the following equation:

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) \max_{a' \in A} Q^*(s',a')$$

- Looks intimidating - but it's really simple

No model?

- The existance of an a-priori model is extremely unrealistic
- Three major classes of methods with dealing with this:
  - **Model-free**
    - Learn policy directly by roaming around the world
  - **Model based**
    - Learn policy indirectly
    - Learn a model of the world
    - Learn a policy by "thinking hard"
  - **Some combination of both (e.g., Dyna)**
    - Learn both a model and a policy from interactions
    - Improve the policy based on fictitious replays

Examples of model free methods

- **Q-learning**: $Q(s,a) \leftarrow Q(s,a) + \eta \left[R(s,a) + \gamma \max_{a' \in A} Q(s',a') - Q(s,a)\right]$
- **SARSA(0)**: $Q(s,a) \leftarrow Q(s,a) + \eta \left[R(s,a) + \gamma Q(s',a') - Q(s,a)\right]$
- **SARSA(1)/MC**: $Q(s,a) \leftarrow Q(s,a) + \eta \left[\mathrm{v}_\tau - Q(s,a)\right]$
  $\mathrm{v}_\tau \leftarrow R(s,a) + \gamma R(s',a') + ... \gamma^2 R(s'',a'') + \gamma^{\tau-1} R(s^\tau, a^\tau)$
- η is a small learning rate, e.g., $\eta = 0.001$

Function Approximation

- States often have some kind of feature (e.g., distance, velocity, etc)
- Too many states in any problem
- Learning based on these features
- Approximate Q-values, policies...
- Neural Networks, Linear Regressors, n-tuple networks etc...
- Global vs Local function approximators

Some issues with model learning

- Errors Compound!
- If you try to think far away, minor errors in the model add up
- You end up with a completely broken future reward/transition structure
  - Unpublished
  - See here http://www.iclr.cc/lib/exe/fetch.php?media=iclr2015:silver-iclr2015.pdf
- Takes time to do

Why learn a model?

- I can't find any good reasons to learn a model
- Learning a model is most of the time MUCH harder than learning to act based on states directly
- Contradicts with our (my?) intuition
  - We make inferences about the future all time + logical reasoning, can't be that Q-values are enough!
- Doing any large-scale learning requires tons of tricks (see Deep-Q learning by deep mind)
  - Maybe we haven't found the right tricks for model-based stuff?

Partial Observability

- It's unintuitive
  - ... because the world is not fully observable
- The Markov property does NOT hold!
- An observation is not enough to recover the state
  - Can you think of an example?
- You cannot observe reality...

POMDPs

- The usual MDP stuff:
  - S, $s \in S$ is a set of states
  - A, $a \in A$ is a set of actions
  - $R : S \times A$, $R(s,a)$ is a function that maps state-actions to rewards
  - $T : S \times S \times A$, with $T(s'|s,a)$ being the probability of an agent moving from state s to state s' after taking a
  - γ is a discount factor - the impact of time on rewards
- Plus an observation model
  - O is a finite set of observations, $o \in O$
  - $\Omega : O \times S \rightarrow A$ is an observation function, with $\Omega(o|s',a)$ the probability observation o was emitted when the agent landed in state s after taking action a

The belief state

- Reality is unobservable
- We can keep track of observations and act on their histories

$$h(s) = \{o_0 ... o_n\} \tag{1}$$

*...Or we can find worthy stuff to believe in....

$$b'(s') = (1/\eta)\Omega(o|s', a) \sum_{s' \in S} T(s'|s, a)b(s) \tag{2}$$

$$\eta = \sum_{s' \in S} \Omega(o|s', a) \sum_{s \in S} T(s'|s, a)b(s) \tag{3}$$

$$\tag{4}$$

Acting on beliefs

- Now we can switch states, transitions, rewards with belief states, transitions, rewards
- We can try learning $Q(b(s), a)$, since s cannot be observed
- We have no clue what $b(s)$ is unless we have a model!
- But there is no way we can get this because we don't even have samples of $T(s'|s, a)$
- This is still possible (free energy learning schemes make this transparent)

Scaling up

- We need to encode the belief model is some kind of distributed form
- Obviously, Recurrent Neural Networks (more later...)
- Notice that the internal model has a relationship with the real world, but can have any form
- Game of Poker:
- External Model: The Game itself
- Internal Model: A probability distribution over possible opponent cards

Attending over beliefs

- Most of the things you believe in are unrelated to your current situation
- You must choose to believe in what is relevant
  - Your beliefs about Poker are irrelevant when driving
  - You need to focus only on relevant beliefs!
  - But how?
- Let's call this attention mechanism $\mathcal{T}$
- Possibly learning $Q(\mathcal{T}(b(s)), a)$ (attending over beliefs)??
- Or $Q(\mathcal{T}(o_0...o_n), a)$ (attending over observations)??

Research Questions

- What beliefs are worth having?
  - Not all beliefs are created the same
  - Some beliefs are counter-productive
  - What's the impact of reward on broken beliefs?
- **What beliefs are worth attending?**
  - **How do we find out which beliefs to focus on?**
  - **What does it mean to focus on different beliefs?**
- How are beliefs transmitted?
  - You can't possibly learn everything from experience
  - Someone has to transmit beliefs directly to you

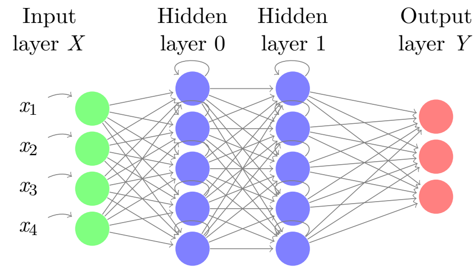
Neural Networks

- Non-linear global function approximators
- Can be learned iteratively through SGD
- Hard to get proofs of convergence for RL, work really well in practice
- Some issues:
  - Catastrophic forgetting
  - Global approximators
  - A new sample will affect all knowledge stored
  - Compare to updating a table (local approximator)

Modern tricks of the trade

- Better initialisation methods (e.g., unsupervisded pre-training, Glorot initialisation)
- Better training methods (e.g., ADAM, RMSPROP)
- Better activation functions/units (e.g., Rectifiers, Maxout)
- Better regularisation methods (e.g., Dropout)
- Better weight sharing/convolutional layers (e.g., Fractional Max-pooling)
- Better hardware (GPUs)

Recurrent Neural Networks



- NNs are ***NOT*** Turing Complete - no internal state
- Cannot handle inputs of arbitrary length
- Sometimes you can substitute *time* with *space*
- . . . but we can add recurrences. . .

Recurrent Neural Networks and Beliefs

- Recurrent networks have internal state
- Hence they form some kind of distributed belief
- But have no notion of attention/focus
- Everything gets updated when a new observation comes in
- In effect, spurious correlations (i.e. overfitting) come up all the time
- When acting one has to focus ONLY on the relevant task

Catastrophic forgetting (related)

- Effectively this means you can teach a network how to ride a bike
- Once you start teaching how to ride a car, it will practically forget everything
- Unless you mix, replay experiences
- But humans don't seem to need that
- You can choose which beliefs to attend over

Fixing Catastrophic forgetting

- Some work here and there
- We need to discover good mechanisms for doing this in artificial systems
- Nothing really good out there IMHO
- Rapid Context switching/focusing on relevant beliefs is key for general intelligence

BABI Dataset

- Language is strongly non-markov
- Words don't mean anything by themselves
- Context extremely important
- Not RL, beliefs are not necessary, but attention almost certainly is
- 20 Different types of questions

Babi Example (Question Type 3)

Mary moved to the bathroom. Sandra journeyed to the bedroom. Mary got the football there. John went back to the bedroom. Mary journeyed to the office. John journeyed to the office. John took the milk. Daniel went back to the kitchen. John moved to the bedroom. Daniel went back to the hallway. Daniel took the apple. John left the milk there. John travelled to the kitchen. Sandra went back to the bathroom. Daniel journeyed to the bathroom. John journeyed to the bathroom. Mary journeyed to the bathroom. Sandra went back to the garden. Sandra went to the office. Daniel went to the garden. Sandra went back to the hallway. Daniel journeyed to the office. Mary dropped the football. John moved to the bedroom.

Where was the football before the bathroom? office

Babi Example (Question Type 3)

Mary moved to the bathroom. Sandra journeyed to the bedroom. Mary got the football there. John went back to the bedroom. Mary journeyed to the office. John journeyed to the office. John took the milk. Daniel went back to the kitchen. John moved to the bedroom. Daniel went back to the hallway. Daniel took the apple. John left the milk there. John travelled to the kitchen. Sandra went back to the bathroom. Daniel journeyed to the bathroom. John journeyed to the bathroom. Mary journeyed to the bathroom. Sandra went back to the garden. Sandra went to the office. Daniel went to the garden. Sandra went back to the hallway. Daniel journeyed to the office. Mary dropped the football. John moved to the bedroom.

Where was the football before the bathroom? office

Distance Networks (my stuff)

- Encoding Paragraph
  - Embed Paragraph word by word
  - (Use RNN to get beliefs)
  - MaxPooling (get only most important features of each sentence)
- Embed Question
  - RNN (get only last state)
- Hop over data:
  - Subtract a the question vector from each sentence
  - Use a recurrent softmax mechanism to get the most relevant sentences

Intuition

- Goal is to find the most relevant sentence
- Focus on it - get results
- Same mechanism that would work for finding $Q(\mathcal{T}(b(s), a)$
- But this is just a mechanism I came up with
- Many more (see JC today)