Predictive Knowledge and Abstraction in Reinforcement Learning

Riashat Islam*

Thanks to ideas from Rich Sutton and Mark Ring

MPhil MLSALT Reading Group
University of Cambridge

* Email: ri258@cam.ac.uk
Reinforcement Learning and Artificial Intelligence (RLAI) Lab, University of Alberta, Edmonton, Alberta

Intelligence

Intelligence itself can be defined as the ability to maintain a very large body of knowledge and apply it effectively and flexibly to new problems

Grand Goal of Artificial Intelligence

Key Question : How do humans build up knowledge through experience and develop a rich understanding of itself and its surrounding environment.

$\rightarrow$ We need to capture the richness of human understanding and understand how to build a knowledge representation to justify our understanding

- Build knowledge intensive intelligent systems
- Learn through everyday experiences
- Agent to capture regularities between its actions and perceptions
- Be able to correct and learn knowledge autonomously

Predictive Knowledge

- As experiences are repeated - they become more predictable
- Build expectations of resulting observations from the actions
- We understand the world by predicting outcomes of actions
- By learning to predict we can represent and expand understanding of the world

Towards the goal of a continuous learning agent that can maintain its own knowledge and understanding of the world

General Artificial Intelligence

Humans build up layers of abstractions and learn through interaction with the environment

- AI agent to make use of experience to build knowledge.
- Continual learning process - key to the development of knowledge
- Experience oriented approach to AI based on RL ideas
- Experience is the data always available to the agent - key to solving AI
- Bridge the gap between low-level sensorimotor experience and human-level world knowledge

Knowledge Representation

- Describe knowledge in terms of senses and actions
- Incremental learning from experience
- Want to ground knowledge in experience
- Ground knowledge in terms of experience

How can predictions capture knowledge?

My keys are in my pocket === If I put my hand in my pocket, I will detect my keys”

→ Knowledge is fundamentally a set of predictions

- Current predictive models in reinforcement learning are too precise - they make single-step predictions of what the agent’s next observation will be
- Single step predictions cannot represent abstract human-level knowledge
- In normal life, it is not important to know what will happen exactly one time step from now
- Humans predict different events and quantities within a non-specific time frame!

We will look at *Forecasts* as non one-step predictions for Predictive Knowledge

Reinforcement Learning - Recap

- Agent-oriented learning by trial and error
- Learn a policy mapping states to actions
- Optimal policy to maximize long run cumulative reward
- Action-Value functions to measure how good it is to be in a state and take an action from there

$$q_\pi(s,a) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + ..|S_t = s, A_t = a] \quad (1)$$

- Find optimal policy from given optimal value function

$$\pi^*(s,a) = \arg\max_a q_*(s,a) \quad (2)$$

Reinforcement Learning - Recap

- Value functions are predictions - what's the expected reward the agent will receive being in the current state
- Policy improvement theorem to find a better policy greedily such that

$$q_{\pi'}(s,a) \geq q_{\pi}(s,a) \tag{3}$$

- Off-policy learning - exploration exploitation dilemma
- Behaviour policy is the off-policy generating the trajectory data, and target policy is the policy being learned about

Reinforcement Learning - Recap

- Parameterized function approximator for the action-value function

$$\hat{q}(s,a,w) \approx q(s,a) \tag{4}$$

- Bootstrapping in RL - learning estimates of value functions from other value estimates
- Sarsa(λ) for controlling the amount of bootstrapping
- Stochastic and Deterministic policy gradient methods - policy represented with own parameters θ independent of the value function

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q^w(s,a)] \tag{5}$$

- Actor-critic methods using a compatible function approximation

RL Ideas to Build Up on...

- Temporal-Difference Learning
- Temporal Difference Networks
- Temporally Abstract Options and Option Models
- General Value Functions (GVFs) or Forecasts

Temporal Difference Learning

Temporal-Difference Learning

- Learn directly from episodes of experience
- Learning from incomplete episodes via bootstrapping
- Updates value towards estimated return

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)) \tag{6}$$

- TD methods based on learning without knowing the final outcome - learn online after every step from incomplete sequences

TD(λ) for n-step predictions, where the λ-return G_t^λ combines all n-step returns where

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^n \tag{7}$$

Eligibility traces to assign credit to heuristics of most frequent states and most recent states

Temporal-Difference Learning

On-policy n-step SARSA or similarly SARSA(λ):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(q_t^n - Q(S_t, A_t)) \tag{8}$$

where

$$q_t^n = R_{t+1} + \gamma R_{t+2} + ..\gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}) \tag{9}$$

Off-policy learning

- Learn about optimal policy while following exploratory off-policy
- Q-learning : off-policy learning of Q(s,a)
- Allow both the behaviour policy μ and target policy π to improve

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R + \gamma \max_{a'} Q(s', a') - Q(s,a)) \tag{10}$$

TD Learning with Value Function Approximation

Estimate the value function with function approximation

$$\hat{v}(s,w) \approx v_\pi(s) \tag{11}$$

$$J(w) = \mathbb{E}_\pi[(v_\pi(S) - \hat{v}(s,w))^2] \tag{12}$$

$$\triangle w = \alpha(v_\pi(s) - \hat{v}(s,w))\nabla_w \hat{v}(s,w) \tag{13}$$

The TD target $R_{t+1} + \gamma\hat{v}(S_{t+1},w)$ is a biased sample of the true value function $v_\pi(S_t)$ such that

$$\triangle w = \alpha(R + \gamma\hat{v}(S',w) - \hat{v}(S,w))\nabla_w \hat{v}(S,w) \tag{14}$$

Recent Deep RL approaches mainly based on using a neural network for the function approximator

Temporal Difference Networks

Temporal Difference Networks

"If I open the fridge, will I see a bottle of beer?"

- What would happen if certain actions were taken?
- What would happen if the bottle was opened and turned upside down?

The target for the overall prediction is a composition in the mathematical sense of the first prediction with each of the other predictions

Temporal Difference Networks

Generalizes TD learning to interrelated predictions

Relate each prediction in a set of predictions to other predictions
$\rightarrow$ represent knowledge in entirely predicted grounded terms

- In conventional TD, the two guesses are of the same quantity at two different points in time
- TD Networks: Possibility of second guess being different from the first

A network of nodes each representing a single scalar prediction

- Predictions of predictions
- Each node is an answer to a question about future observations
- Approach towards learning predictive representations of state (PSR)

TD networks to represent the overall goals of predictive learning

Temporal Difference Networks

Each node in a TD network represents a specific question → something to be predicted

- Node 1 linked to node 2
- Node 1 represents a question incorporating node 2's question
- The value of the node is a prediction about node 2's prediction

The questions in TD network are conditional on a certain way of behaving - conditional on actions

- But opening the fridge is a complex high level action!
- There are other low level actions in this task - such as lifting the arm, hand shaped for grasping etc

This brings us to Temporally Extended Actions...

Temporal Abstraction and Options

Temporal Abstraction and Options

- We want to handle temporally extended courses of action
- High level decisions vs low level actions
- Options - make decisions about which actions to take for a period of time

Option is to pick an object

Actions are the muscle movements!

- Option's policy would now achieve subgoals
- Option will achieve the overall goal

Three classes of temporal abstraction

- Representing temporal abstractions using Options
- Learning temporal abstractions using off-policy methods
- Discovering temporal abstractions and selecting among them

Options

Option chooses among actions $\rightarrow o =< I, \pi, \beta >$

- I is the set of states
- π is the internal policy which determines the way option picks primitive actions
- β is the probability of terminating in each state

$\rightarrow$ If option is initiated, then actions are selected according to π until the option terminates

Hierarchical Options $\rightarrow o =< I, \mu, \beta >$

- μ is the internal policy over options
- The internal policy μ chooses among general options instead of among primitive actions

We now have experience over options as a sequence of states, options and rewards : $< s_t, o_t, r_{t+k}, s_{t+k}, ...s_T, o_T >$

Options

Models of options are a form of knowledge representation - we can learn more than just one policy, such as subgoals...

Options can be related to the RL framework by:

- Policies over options
- Value functions over options
- Planning methods such as value and policy iterations over options

Learning Options

We can generalise the action value function to option value function

$$Q^{\pi}(s,o) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + ..|E(o\pi, s, t)] \tag{15}$$

where $E(o\pi, s, t)$ denotes execution of $o\pi$ in state s.

Off-policy learning of options - learn many options in parallel

$$Q(s,o) \rightarrow Q(s,o) + \alpha[R + \gamma^{k} \max_{o' \in O_s} Q(s',o') - Q(s,o)] \tag{16}$$

Learning optimal options

$$o^{*} = \arg\max_{o \in O_s} Q^{*}(s,o) \tag{17}$$

Eligibility traces and TD learning for Option Models
Key idea: Represent abstract knowledge about course of action at different extended time scales

Learning Options

One step Q-learning for the option-value function

$$Q(s,o) \leftarrow Q(s,o) + \alpha[r + \gamma^k \max_{a \in O} Q(s',a) - Q(s,o)] \qquad (18)$$

Q(s,o) converges to the optinal value function over options $Q^*(s,o)$ for all $s \in S$ and $o \in O$.

- Importance of exploration-exploitation for options - how does this fit in?
- Key issues in options: Select good termination conditions and initiation states
- Intuition related to Eligibility Traces - if states occur freqeuntly on trajectories, these states may be important

Recent work $\rightarrow$ Option-Critic Architecture to extend the Policy Gradient Theorem to Options
Recent work $\rightarrow$ TD networks extended for temporally abstract options

General Value Functions

Value Functions → Knowledge

Knowledge can be represented as large number of approximate value functions learned in parallel, each with its own policy

- Value function asks a question - what will the cumulative future reward be → What it means for the knowledge to be accurate
- Approximate Value Function provides the answer to the question → knowledge

Extend the value function approach beyond reward to a theory of world knowledge

General Value Functions

"How full my stomach will be if I eat the entire cake?"

"How much snow will fall in January"

- Predict within non-specific time frame
- Move beyond one-step predictions
- Move beyond rewards

Instead of $V(S) \rightarrow$ a measure over the course of the future

GVFs or Forecasts - framework for representing general predictive knowledge

Hierarchical construction of various kinds of everyday knowledge

General Value Functions

Let's pretend anything is a reward and learn a value function for getting it

- Multiple value functions as a knowledge representation language
- Represent knowledge as a set of predictions - Agent maintains a set of forecasts
- Forecasts depend on temporally extended actions

Every forecast is a function of state defined by:

$$f(s) = f^{\pi,I,\beta,c,z}(s) \tag{19}$$

Value of forecast is the expected sum of all the cumulative values c the agent encounters while following option, plus the terminal value z when the option terminates

$$f(s) = \mathbb{E}[c_1 + c_2 + ...c_{k-1} + z_k | \pi, \beta, s_0 = s] \tag{20}$$

i,e expected sum of outcomes if the agent behaves as described by the option

General Value Functions

Value of forecast is a prediction about sum of cumulative outcomes c the agent will see if it follows option

- Think of forecasts as a question the agent might ask itself
- Question: if I act in this way, what will the outcome be

Example:

1. Option: Walk to the nearest wall
2. I=1 if there is a wall nearby, 0 otherwise
3. π = turn to face nearest wall, or take step forward
4. β = 1 when at wall

This forecasts the total number of elapsed time steps if I walk to the nearest doorway and then stop

High Level Knowledge → Layered Forecasts

Low Level knowledge

- Immediate knowledge about sensorimotor stream
- Raw actions and observation signals

High Level Knowledge

- Abstractions
- Agent has knowledge about arrangement of walls or rooms in vicinity

Abstraction - feature or function of the sensorimotor stream

Greater abstraction from deeper layering of features

- Proposition : Similar to DNNs, Forecasts can provide a framework for the representation of meaningful, useful abstractions from sensorimotor stream

Layered Forecasts

New forecasts can make predictions about future values of existing forecasts

Example:

- Option: Shoot basket
- Policy: Raise ball, throw towards hoop, watch ball trajectory
- A higher level forecast than above: Will above forecast A have higher value if I move closer to the hoop?

Series of layered forecasts can capture high level knowledge

- In traditional RL, there is little abstraction - no other knowledge than predicitons about agent's raw observations immediately following its actions

Layered Forecasts

Small experiment:

- Initially - agent has no knowledge - no understanding of relationships between senses and actions
- Attemp to build up layers of abstraction that can recognize, predict and utilize regularities of the sensorimotor stream
- Eventually arrive at knowledge of the world, rooms of different sizes, doorways etc $\rightarrow$ all regularities captured from raw sensorimotor stream

Layered Forecasts

- Layer 1 : Value of touch sensor if finger extended
- Layer 2 : Based on above forecast - can agent touch anything 30 degrees to its left?
- Layer 3 : Can agent expect to touch anything after some number of rotations to the left or right
- Layer 4 : Extend to Options - how to rotate until the agent predicts it can touch something by extending its finger
- Layer 5 : Two new options built from previous - policy for moving forward, and then rotate to something it can touch
- Layer 6 : Create map of related predictions - this forecast gives a sense of surroundings - how far it must travel in each direction to reach something it can touch

and so on ...

Represent Knowledge as Forecasts

Just as TD learning is learning a guess from a guess

- GVFs can be considered as building up knowledge in layers
- Each forecast can be part of a prediction of other forecasts $\rightarrow$ layered forecasts!
- New forecasts built to make predictions regarding future value of existing forecasts
- New forecasts governs new behaviour (policy)
- Layered forecasts eventually resulting in long term predictions $\rightarrow$ long term predictions $\rightarrow$ knowledge

Going Forward...

Some key ideas in RL:

- Extending TD learning beyond rewards
- Temporally extended predictions
- Importance of experience for knowledge representation
- General Value Functions or Forecasts from raw sensorimotor stream

Express agent knowledge as predictions $\rightarrow$ Learn predictive representations of abstract common knowledge from raw data!

Thank You

Everything we know that is specific to this world is a prediction of experience. All world knowledge must be translatable into statements about future experience. - Rich Sutton

References

- [1] Horde: A Scalable Real-time Architecture for Learning Knowledge from Unsupervised Sensorimotor Interaction
- [2] Temporal Abstraction and Options
- [3] Better generalisation with Forecasts
- [4] Continual Learning in Reinforcement Learning
- [5] Temporal Difference Networks
- [6] Grounding Abstractions in Predictive State Representations
- [7] Experience Oriented Artificial Intelligence
- [8] The Grand Challenge of Predictive Empirical Abstract Knowledge
- [9] Beyond Reward: The Problem of Knowledge and Data