Steps Towards Continual Learning

Satinder Singh[1]

[1]University of Michigan

DLSS, 2017
Presenter: Xueying Bai

Outline

Motivation

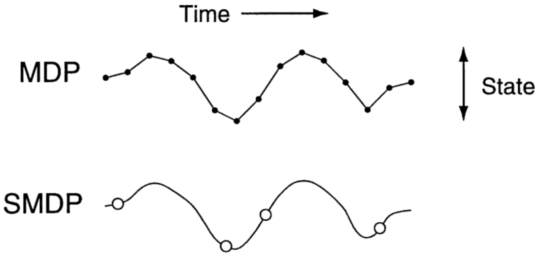
Different from traditional reinforcement learning, continual learning emphasizes on knowledge and intrinsic motivations.

- **Knowledge**: Reuse the knowledge overtime. Agents learn skills (optioins) from experience overtime, store skills using data structure, integrate previous skills to learn new knowledge (optional-conditional predictions).
- Reinforcement learning comes with a task while continual learning doesn't. Where are agents' motivations come from?
  - **Extrinsic Motivation**: It is the outside demand, obligation, or reward that requires the achievement of a particular goal.
  - **Intrinsic Motivation**: Do something because it's inherently enjoyable.

Outline

Semi-Markov Decision Process

- **Markov Decision Process**: Defined by 5-tuple (S, A, P, R, γ), based on a discrete time step: the unitary action taken at time t affects the state and reward at time $t+1$.
- **Semi-Markov Decision Process**: The actions in SMDPs take variable amounts of time and are intended to model temporally-extended courses of action.



Options

- **Options**: Temporary extended behavior. An option is a triple $o =< I, \pi, \beta >$: I is the initiation set of states.
  π is the policy followed during o. $S \times A \to [0,1]$.
  β is termination conditions: probability of terminating in each state. $S_+ \to [0,1]$.
- **Execution of a markov option**: $s_t \xrightarrow{\pi(s_t,\cdot)} a_t \to$ environment transit to $s_{t+1} \to \beta(s_{t+1})$. If the option terminates, it has some opportunity to select another option. Otherwise repeat the process.
  Policy over options: $S \times O \to [0,1]$.

Outline

Option Models

Planning with options requires a model of their consequences. An option model is a probabilistic description of the effects of executing an option.

- Probability with which the option will terminate at any other state.
- Total amount of reward expected over the options execution. Option models can be learned from experience (usually only approximately) using standard methods.

Outline

1. Introduction of Continual Learning
2. Key Elements in Continual Learning
   - Options
   - Option Models
   - Intra-option Learning Methods
   - Reward in Continual Learning
3. A Continual Learning Process
   - Some Concepts
   - A Childs Playroom Domain
   - The Learning Algorithm
4. Deep Learning for Reward Design
   - UCT with Intrinsic Rewards

Intra-option Learning Methods

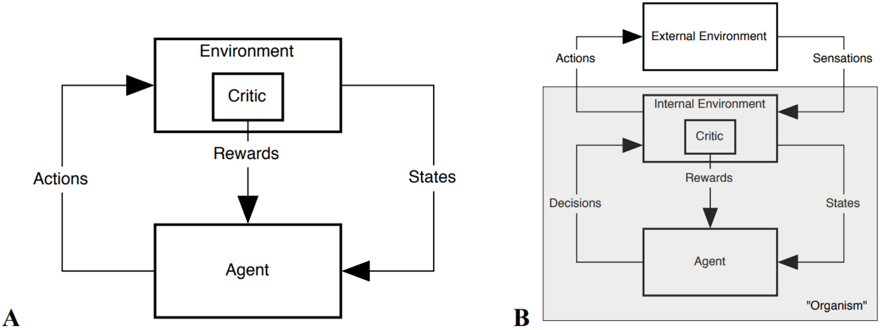
- For a semi-markov model learning, the process is to execute the option to termination many times in each state, record resultant state and reward. Use these outcomes to approximate parameters in the option model. (Keep updating $\rightarrow$ improving efficiency.)
- Two options o_1, o_2 with same actions, just o_2 has one step later to terminate. The experience from o_1 is also helpful for the update of o_2's option model.
- Intra-option learning methods allow the policies of many options to be updated simultaneously during an agents interaction with the environment.

  If an option could have produced a primitive action in a given state, its policy can be updated on the basis of the observed consequences even though it was not directing the agents behavior at the time.

Outline

Intrinsic Reward

Continual learning not necessarily comes with specific tasks, so not only relies on the external reward.



- **Intrinsic reward**: Primary reward.
- **Extrinsic reward**: Guide the internal to generate appropriate level of primary reward.

Outline

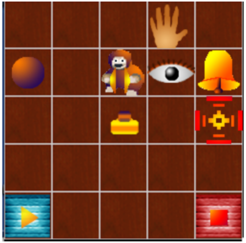
Some Concepts

- **Behavior**: Take action a in state s. Decided by Q_B.
- **Salient Event**: Things the agent has interest in. Independent of specific tasks and applicable to many environments.
- **Skill_KB**: Agent's knowledge base. A set of options.

Once a salient event occurs, learn an option that achieves that salient event. In this part, the intrinsic reward for each salient event is proportional to the error in the prediction of the salient event according to the learned option model for that event.

Outline

A Childs Playroom Domain



A Child's Playroom Domain (NIPS 2004)
(An ancient Continual Learning Demonstration)

Agent has: hand, eye, marker
Primitive Actions: 1) move hand to eye, move eye to hand, move eye to marker
move eye N, S, E, W, move eye to random object, move marker to eye,
move marker to hand. If both eye and hand are on object, *operate* on object
(e.g., push ball to marker, toggle light switch)

Objects: Switch controls room lights; Bell rings and moves one square if ball hits it;
Pressing blue/red block turns music on and off; Lights have to be on to see colors;
Can push blocks; Money cries out if bell and music both sound in dark room

Skills: (example)
To make monkey cry out: Move eye to switch, move hand to eye, turn lights on, move eye to blue block, move hand to eye, turn music on, move eye to switch, move hand to eye, turn light off, move eye to bell, move marker to eye, move eye to ball, move hand to ball, kick ball to make bell ring
Uses skills (options): turn lights on, turn music on, turn lights off, ring bell

Singh, Barto & Chentanez

Outline

The Learning Algorithm

Loop forever

Current state s_t, current primitive action a_t, current option o_t,
extrinsic reward r_t^e, intrinsic reward r_t^i

Obtain next state s_{t+1}

//— *Deal with special case if next state is salient*
If s_{t+1} is a salient event e
If option for e, o_e, does not exist in O (skill-KB)
Create option o_e in skill-KB;
Add s_t to I^{o_e} // *initialize initiation set*
Set $\beta^{o_e}(s_{t+1}) = 1$ // *set termination probability*
//— *set intrinsic reward value*
$r_{t+1}^i = \tau[1 - P^{o_e}(s_{t+1}|s_t)]$ // *τ is a constant multiplier*
else
$r_{t+1}^i = 0$

//— *Update all option models*
For each option $o \neq o_e$ in skill-KB (O)
If $s_{t+1} \in I^o$, then add s_t to I^o // grow initiation set
If a_t is greedy action for o in state s_t
//— *update option transition probability model*
$P^o(x|s_t) \overset{\alpha}{\leftarrow} [\gamma(1 - \beta^o(s_{t+1})P^o(x|s_{t+1}) + \gamma\beta^o(s_{t+1})\delta_{s_{t+1}x}]$
//— *update option reward model*
$R^o(s_t) \overset{\alpha}{\leftarrow} [r_t^e + \gamma(1 - \beta^o(s_{t+1}))R^o(s_{t+1})]$

The Learning Algorithm

//— *Q-learning update of behavior action-value function*
$Q_B(s_t, a_t) \stackrel{\alpha}{\leftarrow} [r_t^e + r_t^i + \gamma \max_{a \in A \cup O} Q_B(s_{t+1}, a)]$

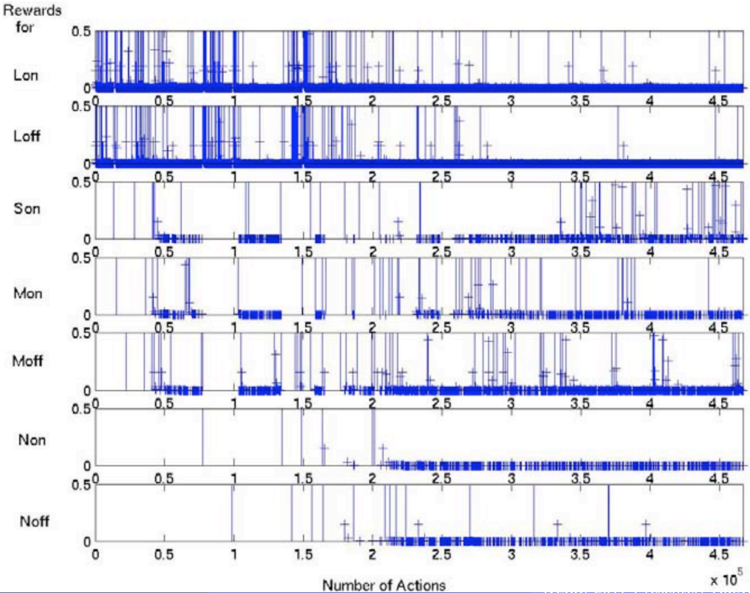
//— *SMDP-planning update of behavior action-value function*
For each option o in skill-KB
$Q_B(s_t, o) \stackrel{\alpha}{\leftarrow} [R^o(s_t) + \sum_{x \in S} P^o(x|s_t) \max_{a \in A \cup O} Q_B(x, a)]$

//— *Update option action-value functions*
For each option $o \in O$ such that $s_t \in I^o$
$Q^o(s_t, a_t) \stackrel{\alpha}{\leftarrow} [r_t^e + \gamma\ (\beta^o(s_{t+1}) \times \text{terminal value for option o})$
$+\gamma(1 - \beta^o(s_{t+1})) \times \max_{a \in A \cup O} Q^o(s_{t+1}, a)]$
For each option $o' \in O$ such that $s_t \in I^{o'}$ and $o \neq o'$
$Q^o(s_t, o') \stackrel{\alpha}{\leftarrow} R^{o'}(s_t) + \sum_{x \in S} P^{o'}(x|s_t)[\beta^o(x) \times \text{terminal val for option o}$
$+((1 - \beta^o(x)) \times \max_{a \in A \cup O} Q^o(x, a))]$

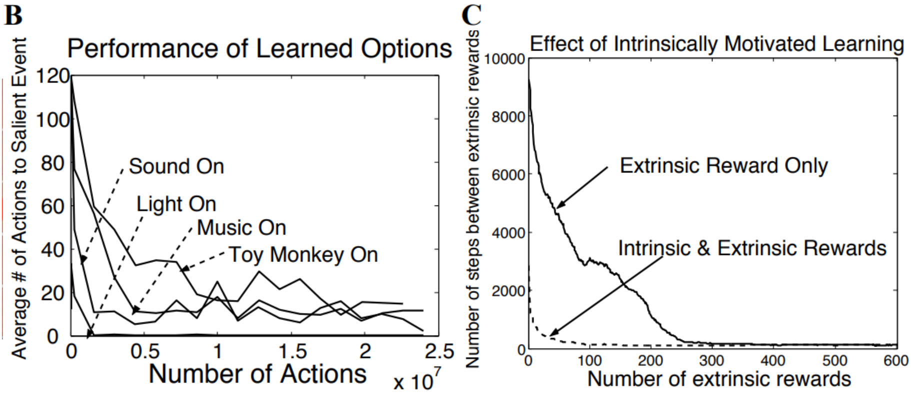
Choose a_{t+1} using ϵ-greedy policy w.r.to Q_B // — *Choose next action*
//— *Determine next extrinsic reward*
Set r_{t+1}^e to the extrinsic reward for transition $s_t, a_t \rightarrow s_{t+1}$

Set $s_t \leftarrow s_{t+1}$; $a_t \leftarrow a_{t+1}$; $r_t^e \leftarrow r_{t+1}^e$; $r_t^i \leftarrow r_{t+1}^i$

Experimental Results



Experimental Results



Outline

UCT with Intrinsic Rewards

UCT is a planning method. It generate trajectories of state-action pairs in a greedy way. Each state-art pair is defined by (s, a, d). Things go through UCT is like a look at search.

- The return after experiencing the tuple:

$$Q(s, a, d) = \sum_{i=1}^{N} \frac{I_i(s, a, d)}{n(s, a, d)} \sum_{h=d}^{H-1} \gamma^{h-d} R(s_h^i, a_h^i)$$

  N is the number of trajectories, H is the trajectory length. d starts from 0.

- Internal reward for UCT with intrinsic reward:

$$R^I(s, a; \theta) = CNN(s, a; \theta) + R^O(s, a)$$

UCT with Intrinsic Rewards

- When UCT planning finishes, the greedy action is:

$$a = \arg\max_b Q^I(s, b, 0; \theta)$$

- The softmax version for gradient:

$$\mu(a|s;\theta) = \frac{\exp Q^I(s, a, 0; \theta)}{\sum_b \exp Q^I(s, b, 0; \theta)}$$

- Objective function:

$$u(h_T) = \sum_{t=0}^{T-1} R^O(s_t, a_t)$$

- Parameters for intrinsic reward:

$$\theta^* = \arg\max_\theta E\{u(h_T)|\theta\}$$