Frontiers

Monday, June 29, 2020 8:36 PM

General Value Function:

$$v_{\pi,\gamma,C(s)} = E\left[\sum_{k=t}^{\infty}\left(\prod_{i=T+1}^{k}\gamma(S_i)\right)C_{k+1}\,|S_t = s, A_{t:\infty}\sim\pi\right]$$

Where $\gamma(S_i)$ is the discount for state S_i and C_{k+1} is a generalized reward called a cumulant signal intended to be a general signal that is added up or accumulated

Using signals other than long term reward may allow for learning auxiliary tasks that can help on the main task but may not be incentivized on their own

Option:

$\omega = \langle \pi_\omega, \gamma_\omega \rangle$

A generalized form of action consisting of a policy and state-dependent termination function

Executing an option at time t involves following the policy starting from time t then terminating at time $t+1$ with probability $1-\gamma_\omega(S_{t+1})$ and generally terminating at time i with probability $1-\gamma_\omega(S_i)$ until termination

Action value functions can be generalized to option value functions

Policies can be generalized to hierarchical policies that select from options instead of actions

Option Reward:

$r(s,\omega) \equiv E\left[R_1 + \gamma R_2 + \gamma^2 R_3 + \gamma^{\tau-1}R_\tau \middle| S_0 = s, A_{0:\tau-1}\sim\pi_\omega, \tau\sim\gamma_\omega\right]$

Option State-Transitions:

$$p(s'|s,\omega) \equiv \sum_{k=1}^{\infty}\gamma^k \Pr\{S_k = s', \tau = k | S_0 = s, A_{0:k-1}\sim\pi_w, \tau\sim\gamma_w\}$$

Option Bellman Equation:

$$v_\pi(s) = \sum_{\omega\in\Omega(s)}\pi(\omega|s)\left[r(s,\omega) + \sum_{s'}p(s'|s,\omega)v_\pi(s')\right]$$

Where $\Omega(s)$ denotes the set of options available in state s

Option Value Iteration Algorithm:

$$v_{k+1}(s) \equiv \max_{\omega\in\Omega(s)}\left[r(s,\omega) + \sum_{s'}p(s'|s,\omega)v_k(s')\right]$$

If $\Omega(s)$ includes all the low-level (normally defined) options available in each state, then this value converges to the conventional optimal value function v_*

Recall that function approximations that are parameterized to depend on a certain state variable are the same as if that state variable were unobservable

This means that the below section can be avoided by simply using function approximation methods

Explicit Treatment of Partial Observability:

Instead of states, the environment should be considered as outputting observations

Instead of states, the agent should consider a history composed of all past actions and observations

A state would then reasonable be represented by a compact summary/representation of the history

In this formalism a test is any specific sequence of alternating states and observations

Markov Condition:

$f(h) = f(h') \rightarrow p(\tau|h) = p(\tau|h')$

Where $p(\tau|h) \equiv$

probability of a test $\tau = a_1 o_1 a_2 o_2 a_3 o_3$ given a specific history h

A Markov condition summarizes all the information in the history necessary for determining any test's probability/making any predictions

Want a Compact State Representation:

$S_{t+1} \equiv u(S_t, A_t, O_{t+1})$

Where u is called the state-update function

This ensures that the state can be represented compactly since it can be updated incrementally

Markov Condition for Incrementally Updated States:

$f(h) = f(h') \rightarrow \Pr\{O_{t+1} = o | H_t = h, A_t = a\} = \Pr\{O_{t+1} = o | H_t = h', A_t = a\}$

This is just saying that the same actions in the same state should lead to the same probability distribution of observations

This means that one-step predictions are enough (although if they are not exact errors will propagate)

Use an approximate version of state (States that are non-Markov)

Partially Observable MDP (POMDP):

A MDP where the state is assumed to have some latent state X_t which underlies the environment's observations, but is never available to the agent

Belief State:

$s_t[i] \equiv \Pr\{X_i = i | H_t\}$

This is the distribution over latent states, given the history in an POMDP

This is the equivalent to the Markov state S_t for a POMDP

Belief State Update:

$$u(\vec{s}, a, o)[i] \equiv \frac{\sum_{x=1}^{d} \vec{s}[x] p(i, o | x, a)}{\sum_{x=1}^{d} \sum_{x'=1}^{d} \vec{s}[x] p(x', o | x, a)}$$

Latent State Transition Probabilities:

$p(x', o | x, a) \equiv \Pr\{X_t = x', O_t = o | X_{t-1} = x, A_{t-1} = a\}$

Note that the update here is Bayes Rule

Predictive State Representations:

Another way to handle a partially observable state that puts the state in terms of predictions about future observations and actions

One way to perform shaping is to change the reward signal over time

This can help avoid sparse rewards (where receiving the reward is too infrequent to effectively guide learning)

Another way to perform shaping is to start on easier tasks then proceed to harder tasks

Inverse Reinforcement Learning:

Starting from an agent and trying to learn the reward that the agent is using

Future Directions the Authors Outline:

Powerful parametric function approximation methods that work in fully incremental and online settings

They see the issue with neural nets as requiring too much training data and not being able to train well without forgetting old experiences

Methods for learning features such that subsequent learning generalizes well

AKA how to use experience not just to learn a desired function, but to learn inductive biases such that future learning generalizes better and is thus faster (meta-learning)

This can potentially be though of as learning the state update function

Scalable models for planning with learned environment models

Learned models need to focus on the key consequences of the most important options so that planning can be fast and effective

The learned model should also be continually monitored to keep planning efficiency high

The Dyna model extended to learn nonlinear environment models may be a starting point?

Automating the choice of tasks that an agent works on and uses to structure its developing competence

- Agents that set their own goals
- Chosen tasks might be subtasks of an overall tasks or intended to create building blocks for more efficient learning later

Interaction between behavior and learning through an analog of curiosity

- This analog would be a form on intrinsic reward in their formulation

Methods to safely embed reinforcement learning into physical systems