Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

The Option-Critic Architecture

Pierre-Luc Bacon, Jean Harb, Doina Precup
Reasoning and Learning Lab, School of Computer Science
McGill University
{pbacon, jharb, dprecup}@cs.mcgill.ca

Abstract

Temporal abstraction is key to scaling up learning and planning in reinforcement learning. While planning with temporally extended actions is well understood, creating such abstractions autonomously from data has remained challenging. We tackle this problem in the framework of options [Sutton, Precup & Singh, 1999; Precup, 2000]. We derive policy gradient theorems for options and propose a new *option-critic* architecture capable of learning both the internal policies and the termination conditions of options, in tandem with the policy over options, and without the need to provide any additional rewards or subgoals. Experimental results in both discrete and continuous environments showcase the flexibility and efficiency of the framework.

Introduction

Temporal abstraction allows representing knowledge about courses of action that take place at different time scales. In reinforcement learning, *options* (Sutton, Precup, and Singh 1999; Precup 2000) provide a framework for defining such courses of action and for seamlessly learning and planning with them. *Discovering* temporal abstractions autonomously has been the subject of extensive research efforts in the last 15 years (McGovern and Barto 2001; Stolle and Precup 2002; Menache, Mannor, and Shimkin 2002; Şimşek and Barto 2009; Silver and Ciosek 2012), but approaches that can be used naturally with continuous state and/or action spaces have only recently started to become feasible (Konidaris et al. 2011; Niekum 2013; Mann, Mannor, and Precup 2015; Mankowitz, Mann, and Mannor 2016; Kulkarni et al. 2016; Vezhnevets et al. 2016; Daniel et al. 2016).

The majority of the existing work has focused on finding *subgoals* (useful states that an agent should reach) and subsequently learning policies to achieve them. This idea has led to interesting methods but ones which are also difficult to scale up given their "combinatorial" flavor. Additionally, learning policies associated with subgoals can be expensive in terms of data and computation time; in the worst case, it can be as expensive as solving the entire task.

We present an alternative view, which blurs the line between the problem of *discovering options* from that of *learning options*. Based on the policy gradient theorem (Sutton et al. 2000), we derive new results which enable a gradual learning process of the intra-option policies and termination functions, simultaneously with the policy over them. This approach works naturally with both linear and non-linear function approximators, under discrete or continuous state and action spaces. Existing methods for learning options are considerably slower when learning from a single task: much of the benefit comes from re-using the learned options in similar tasks. In contrast, we show that our approach is capable of successfully learning options within a single task without incurring any slowdown and while still providing benefits for transfer learning.

We start by reviewing background related to the two main ingredients of our work: policy gradient methods and options. We then describe the core ideas of our approach: the intra-option policy and termination gradient theorems. Additional technical details are included in the appendix. We present experimental results showing that our approach learns meaningful temporally extended behaviors in an effective manner. As opposed to other methods, we only need to specify the number of desired options; it is not necessary to have subgoals, extra rewards, demonstrations, multiple problems or any other special accommodations (however, the approach can take advantage of pseudo-reward functions if desired). To our knowledge, this is the first end-to-end approach for learning options that scales to very large domains at comparable efficiency.

Preliminaries and Notation

A Markov Decision Process consists of a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, a transition function $\mathrm{P} : \mathcal{S} \times \mathcal{A} \to (\mathcal{S} \to [0, 1])$ and a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. For convenience, we develop our ideas assuming discrete state and action sets. However, our results extend to continuous spaces using usual measure-theoretic assumptions (some of our empirical results are in continuous tasks). A (Markovian stationary) *policy* is a probability distribution over actions conditioned on states, $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$. In discounted problems, the value function of a policy π is defined as the expected return: $V_\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \mid s_0 = s\right]$ and its action-value function as $Q_\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \mid s_0 = s, a_0 = a\right]$, where $\gamma \in [0, 1)$ is the *discount factor*. A policy π is *greedy* with respect to a given action-value function Q if

$\pi(s,a) > 0$ iff $a = \text{argmax}_{a'} Q(s,a')$. In a discrete MDP, there is at least one optimal policy which is greedy with respect to its own action-value function.

Policy gradient methods (Sutton et al. 2000; Konda and Tsitsiklis 2000) address the problem of finding a good policy by performing stochastic gradient descent to optimize a performance objective over a given family of parametrized stochastic policies, π_θ. The policy gradient theorem (Sutton et al. 2000) provides expressions for the gradient of the average reward and discounted reward objectives with respect to θ. In the discounted setting, the objective is defined with respect to a designated start state (or distribution) s_0: $\rho(\theta, s_0) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \mid s_0\right]$. The policy gradient theorem shows that: $\frac{\partial \rho(\theta, s_0)}{\partial \theta} = \sum_s \mu_{\pi_\theta}(s \mid s_0) \sum_a \frac{\partial \pi_\theta(a|s)}{\partial \theta} Q_{\pi_\theta}(s,a)$, where $\mu_{\pi_\theta}(s \mid s_0) = \sum_{t=0}^{\infty} \gamma^t \mathrm{P}(s_t = s \mid s_0)$ is a discounted weighting of the states along the trajectories starting from s_0. In practice, the policy gradient is estimated from samples along the on-policy stationary distribution. (Thomas 2014) showed that neglecting the discount factor in this stationary distribution makes the usual policy gradient estimator biased. However, correcting for this discrepancy also reduces data efficiency. For simplicity, we build on the framework of (Sutton et al. 2000) and discuss how to extend our results according to (Thomas 2014).

The options framework (Sutton, Precup, and Singh 1999; Precup 2000) formalizes the idea of temporally extended actions. A Markovian option $\omega \in \Omega$ is a triple $(\mathcal{I}_\omega, \pi_\omega, \beta_\omega)$ in which $\mathcal{I}_\omega \subseteq \mathcal{S}$ is an initiation set, π_ω is an *intra-option* policy, and $\beta_\omega : \mathcal{S} \to [0,1]$ is a termination function. We also assume that $\forall s \in \mathcal{S}, \forall \omega \in \Omega : s \in \mathcal{I}_\omega$ (i.e., all options are available everywhere), an assumption made in the majority of option discovery algorithms. We will discuss how to dispense with this assumption in the final section. (Sutton, Precup, and Singh 1999; Precup 2000) show that an MDP endowed with a set of options becomes a Semi-Markov Decision Process (Puterman 1994, chapter 11), which has a corresponding optimal value function over options $V_\Omega(s)$ and option-value function $Q_\Omega(s,\omega)$. Learning and planning algorithms for MDPs have their counterparts in this setting. However, the existence of the underlying MDP offers the possibility of learning about many different options in parallel : this is the idea of *intra-option learning*, which we leverage in our work.

Learning Options

We adopt a continual perspective on the problem of learning options. At any time, we would like to distill all of the available experience into every component of our system: value function and policy over options, intra-option policies and termination functions. To achieve this goal, we focus on learning option policies and termination functions, assuming they are represented using differentiable parameterized function approximators.

We consider the *call-and-return* option execution model, in which an agent picks option ω according to its policy over options π_Ω , then follows the intra-option policy π_ω until termination (as dictated by β_ω), at which point this procedure is repeated. Let $\pi_{\omega,\theta}$ denote the intra-option policy of option ω parametrized by θ and $\beta_{\omega,\vartheta}$, the termination function of ω parameterized by ϑ. We present two new results for learning options, obtained using as blueprint the policy gradient theorem (Sutton et al. 2000). Both results are derived under the assumption that the goal is to learn options that maximize the expected return in the current task. However, if one wanted to add extra information to the objective function, this could readily be done so long as it comes in the form of an additive differentiable function.

Suppose we aim to optimize directly the discounted return, expected over all the trajectories starting at a designated state s_0 and option ω_0, then: $\rho(\Omega, \theta, \vartheta, s_0, \omega_0) = \mathbb{E}_{\Omega,\theta,\omega}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \mid s_0, \omega_0\right]$. Note that this return depends on the policy over options, as well as the parameters of the option policies and termination functions. We will take gradients of this objective with respect to θ and ϑ. In order to do this, we will manipulate equations similar to those used in *intra-option* learning (Sutton, Precup, and Singh 1999, section 8). Specifically, the definition of the option-value function can be written as:

$$Q_\Omega(s,\omega) = \sum_a \pi_{\omega,\theta}(a \mid s)\, Q_U(s,\omega,a) \ , \qquad (1)$$

where $Q_U : \mathcal{S} \times \Omega \times \mathcal{A} \to \mathbb{R}$ is the value of executing an action in the context of a state-option pair:

$$Q_U(s,\omega,a) = r(s,a) + \gamma \sum_{s'} \mathrm{P}(s' \mid s,a)\, U(\omega, s') \ . \quad (2)$$

Note that the (s,ω) pairs lead to an augmented state space, cf. (Levy and Shimkin 2011). However, we will not work explicitly with this space; it is used only to simplify the derivation. The function $U : \Omega \times \mathcal{S} \to \mathbb{R}$ is called the option-value function *upon arrival*, (Sutton, Precup, and Singh 1999, equation 20). The value of executing ω upon entering a state s' is given by:

$$U(\omega, s') = (1 - \beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \ \beta_{\omega,\vartheta}(s')V_\Omega(s') \quad (3)$$

Note that Q_U and U both depend on θ and ϑ, but we do not include these in the notation for clarity. The last ingredient required to derive policy gradients is the Markov chain along which the performance measure is estimated. The natural approach is to consider the chain defined in the augmented state space, because state-option pairs now play the role of regular states in a usual Markov chain. If option ω_t has been initiated or is executing at time t in state s_t, then the probability of transitioning to (s_{t+1}, ω_{t+1}) in one step is:

$$\mathrm{P}(s_{t+1}, \omega_{t+1} \mid s_t, \omega_t) = \sum_a \pi_{\omega_t,\theta}(a \mid s_t)\, \mathrm{P}(s_{t+1} \mid s_t, a)($$
$$(1 - \beta_{\omega_t,\vartheta}(s_{t+1}))\mathbf{1}_{\omega_t = \omega_{t+1}} + \ \beta_{\omega_t,\vartheta}(s_{t+1})\pi_\Omega(\omega_{t+1} \mid s_{t+1})) \quad (4)$$

Clearly, the process given by (4) is homogeneous. Under mild conditions, and with options available everywhere, it is in fact ergodic, and a unique stationary distribution over state-option pairs exists.

We will now compute the gradient of the expected discounted return with respect to the parameters θ of the intra-option policies, assuming that they are stochastic and differentiable. From (1 , 2), it follows that:

$$\frac{\partial Q_\Omega(s,\omega)}{\partial\theta} = \left(\sum_a \frac{\partial \pi_{\omega,\theta}(a \mid s)}{\partial\theta} Q_U(s,\omega,a)\right) + \sum_a \pi_{\omega,\theta}(a \mid s) \sum_{s'} \gamma \, \mathrm{P}(s' \mid s,a) \frac{\partial U(\omega,s')}{\partial\theta}.$$

We can further expand the right hand side using (3) and (4), which yields the following theorem:

Theorem 1 (Intra-Option Policy Gradient Theorem)**.** *Given a set of Markov options with stochastic intra-option policies differentiable in their parameters θ, the gradient of the expected discounted return with respect to θ and initial condition (s_0,ω_0) is:*

$$\sum_{s,\omega} \mu_\Omega(s,\omega \mid s_0,\omega_0) \sum_a \frac{\partial \pi_{\omega,\theta}(a \mid s)}{\partial\theta} Q_U(s,\omega,a) \ ,$$

where $\mu_\Omega(s,\omega \mid s_0,\omega_0)$ is a discounted weighting of state-option pairs along trajectories starting from (s_0,ω_0): $\mu_\Omega(s,\omega \mid s_0,\omega_0) = \sum_{t=0}^{\infty} \gamma^t \, \mathrm{P}(s_t = s, \omega_t = \omega \mid s_0,\omega_0)$.

The proof is in the appendix. This gradient describes the effect of a local change at the primitive level on the global expected discounted return. In contrast, subgoal or pseudo-reward methods assume the objective of an option is simply to optimize its own reward function, ignoring how a proposed change would propagate in the overall objective.

We now turn our attention to computing gradients for the termination functions, assumed this time to be stochastic and differentiable in ϑ. From (1, 2, 3), we have:

$$\frac{\partial Q_\Omega(s,\omega)}{\partial\vartheta} = \sum_a \pi_{\omega,\theta}(a \mid s) \sum_{s'} \gamma \, \mathrm{P}(s' \mid s,a) \frac{\partial U(\omega,s')}{\partial\vartheta}.$$

Hence, the key quantity is the gradient of U. This is a natural consequence of the call-and-return execution, in which the "goodness" of termination functions can only be evaluated upon entering the next state. The relevant gradient can be further expanded as:

$$\frac{\partial U(\omega,s')}{\partial\vartheta} = -\frac{\partial \beta_{\omega,\vartheta}(s')}{\partial\vartheta} A_\Omega(s',\omega) + \gamma \sum_{\omega'} \sum_{s''} \mathrm{P}(s'',\omega' \mid s',\omega) \frac{\partial U(\omega',s'')}{\partial\vartheta} \ , \quad (5)$$

where A_Ω is the advantage function (Baird 1993) over options $A_\Omega(s',\omega) = Q_\Omega(s',\omega) - V_\Omega(s')$. Expanding $\frac{\partial U(\omega',s'')}{\partial\vartheta}$ recursively leads to a similar form as in theorem (1) but where the weighting of state-option pairs is now according to a Markov chain shifted by one time step: $\mu_\Omega(s_{t+1},\omega_t \mid s_t,\omega_{t-1})$ (details are in the appendix).

Theorem 2 (Termination Gradient Theorem)**.** *Given a set of Markov options with stochastic termination functions differentiable in their parameters ϑ, the gradient of the expected discounted return objective with respect to ϑ and the initial condition (s_1,ω_0) is:*

$$-\sum_{s',\omega} \mu_\Omega(s',\omega \mid s_1,\omega_0) \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial\vartheta} A_\Omega(s',\omega) \ ,$$

where $\mu_\Omega(s',\omega \mid s_1,\omega_0)$ is a discounted weighting of state-option pairs from (s_1,ω_0): $\mu_\Omega(s,\omega \mid s_1,\omega_0) = \sum_{t=0}^{\infty} \gamma^t \, \mathrm{P}(s_{t+1} = s, \omega_t = \omega \mid s_1,\omega_0)$.

The advantage function often appears in policy gradient methods (Sutton et al. 2000) when forming a *baseline* to reduce the variance in the gradient estimates. Its presence in that context has to do mostly with algorithm design. It is interesting that in our case, it follows as a direct consequence of the derivation and gives the theorem an intuitive interpretation: when the option choice is suboptimal with respect to the expected value over all options, the advantage function is negative and it drives the gradient corrections up, which increases the odds of terminating. After termination, the agent has the opportunity to pick a better option using π_Ω. A similar idea also underlies the *interrupting* execution model of options (Sutton, Precup, and Singh 1999) in which termination is forced whenever the value of $Q_\Omega(s',\omega)$ for the current option ω is less than $V_\Omega(s')$. (Mann, Mankowitz, and Mannor 2014) recently studied interrupting options through the lens of an *interrupting Bellman Operator* in a value-iteration setting. The termination gradient theorem can be interpreted as providing a gradient-based interrupting Bellman operator.

Algorithms and Architecture

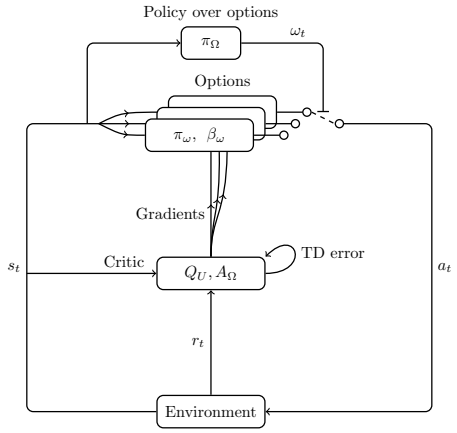

Figure 1: Diagram of the option-critic architecture. The option execution model is depicted by a *switch* $\perp$ over the *contacts* $-\circ$. A new option is selected according to π_Ω only when the current option terminates.

Based on theorems 1 and 2, we can now design a stochastic gradient descent algorithm for learning options. Using a two-timescale framework (Konda and Tsitsiklis 2000), we propose to learn the values at a *fast* timescale while updating the intra-option policies and termination functions at a *slower* rate.

We refer to the resulting system as an *option-critic architecture*, in reference to the actor-critic architectures (Sutton 1984). The intra-option policies, termination functions and policy over options belong to the *actor* part of the system while the *critic* consists of Q_U and A_Ω. The option-critic architecture does not prescribe how to obtain π_Ω since a variety of existing approaches would apply: using policy gradient methods at the SMDP level, with a planner over the options models, or using temporal difference updates. If π_Ω is the greedy policy over options, it follows from (2) that the corresponding one-step off-policy update target $g_t^{(1)}$ is:

$$g_t^{(1)} = r_{t+1} + \gamma\Big((1-\beta_{\omega_t,\vartheta}(s_{t+1}))\sum_a \pi_{\omega_t,\theta}(a \mid s_{t+1})\, Q_U(s_{t+1},\omega_t,a) + \beta_{\omega_t,\vartheta}(s_{t+1})\max_\omega \sum_a \pi_{\omega,\theta}(a \mid s_{t+1})\, Q_U(s_{t+1},\omega,a)\Big)\ ,$$

which is also the update target of the *intra-option Q-learning* algorithm of (Sutton, Precup, and Singh 1999). A prototypical implementation of option-critic which uses intra-option Q-learning is shown in Algorithm 1. The tabular setting is assumed only for clarity of presentation. We write α, α_θ and α_ϑ for the learning rates of the critic, intra-option policies and termination functions respectively.

Algorithm 1: Option-critic with tabular intra-option Q-learning

$s \leftarrow s_0$
Choose ω according to an ϵ-soft policy over options $\pi_\Omega(s)$
repeat
  Choose a according to $\pi_{\omega,\theta}(a \mid s)$
  Take action a in s, observe s', r

  1. Options evaluation:
  $\delta \leftarrow r - Q_U(s,\omega,a)$
  if s' *is non-terminal* **then**
    $\delta \leftarrow \delta + \gamma(1-\beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \gamma\beta_{\omega,\vartheta}(s')\max_{\bar{\omega}} Q_\Omega(s',\bar{\omega})$
  end
  $Q_U(s,\omega,a) \leftarrow Q_U(s,\omega,a) + \alpha\delta$

  2. Options improvement:
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log \pi_{\omega,\theta}(a \mid s)}{\partial\theta} Q_U(s,\omega,a)$
  $\vartheta \leftarrow \vartheta - \alpha_\vartheta \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial\vartheta}\,(Q_\Omega(s',\omega) - V_\Omega(s'))$

  if $\beta_{\omega,\vartheta}$ *terminates in* s' **then**
  choose new ω according to ϵ-soft$(\pi_\Omega(s'))$
  $s \leftarrow s'$
until s' *is terminal*

Learning Q_U in addition to Q_Ω is computationally wasteful both in terms of the number of parameters and samples. A practical solution is to only learn Q_Ω and derive an estimate of Q_U from it. Because Q_U is an expectation over next states, $Q_U(s,\omega,a) = \mathbb{E}_{s'\sim \mathrm{P}}\left[r(s,a) + \gamma U(\omega,s') \mid s,\omega,a\right]$, it follows that $g_t^{(1)}$ is an appropriate estimator. We chose this approach for our experiment with deep neural networks in the Arcade Learning Environment.

Experiments

We first consider a navigation task in the four-rooms domain (Sutton, Precup, and Singh 1999). Our goal is to evaluate the ability of a set of options learned fully autonomously to recover from a sudden change in the environment. (Sutton, Precup, and Singh 1999) presented a similar experiment for a set of pre-specified options; the options in our results have not been specified a priori.

Initially the goal is located in the east doorway and the initial state is drawn uniformly from all the other cells. After 1000 episodes, the goal moves to a random location in the lower right room. Primitive movements can fail with probability $1/3$, in which case the agent transitions randomly to one of the empty adjacent cells. The discount factor was 0.99, and the reward was $+1$ at the goal and 0 otherwise. We chose to parametrize the intra-option policies with Boltzmann distributions and the terminations with sigmoid functions. The policy over options was learned using intra-option Q-learning. We also implemented primitive actor-critic (denoted AC-PG) using a Boltzmann policy. We also compared option-critic to a primitive SARSA agent using Boltzmann exploration and no eligibility traces. For all Boltzmann policies, we set the temperature parameter to 0.001. All the weights were initialized to zero.

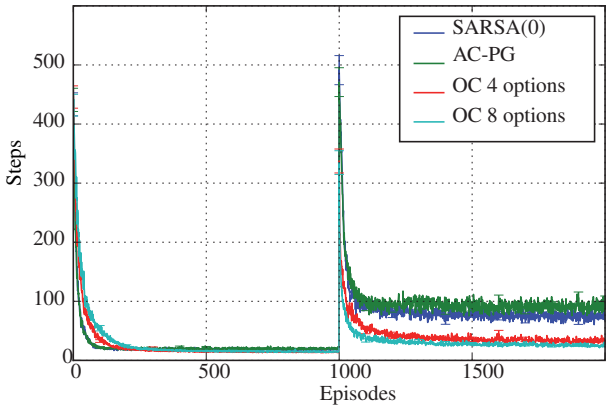

Figure 2: After a 1000 episodes, the goal location in the four-rooms domain is moved randomly. Option-critic ("OC") recovers faster than the primitive actor-critic ("AC-PG") and SARSA(0). Each line is averaged over 350 runs.

As can be seen in Figure 2, when the goal suddenly changes, the option-critic agent recovers faster. Furthermore, the initial set of options is learned *from scratch* at a rate comparable to primitive methods. Despite the simplicity of the domain, we are not aware of other methods which could have solved this task without incurring a cost much larger than when using primitive actions alone (McGovern and Barto 2001; Şimşek and Barto 2009).

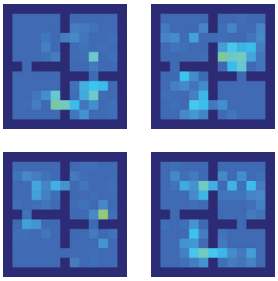

Figure 3: Termination probabilities for the option-critic agent learning with 4 options. The darkest color represents the *walls* in the environment while lighter colors encode higher termination probabilities.

In the two temporally extended settings, with 4 options and 8 options, termination events are more likely to occur near the doorways (Figure 3), agreeing with the intuition that they would be good subgoals. As opposed to (Sutton, Precup, and Singh 1999), we did not encode this knowledge ourselves but simply let the agents find options that would maximize the expected discounted return.

Pinball Domain

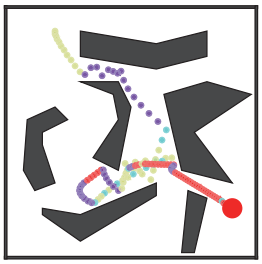

Figure 4: Pinball: Sample trajectory of the solution found after 250 episodes of training using 4 options All options (color-coded) are used by the policy over options in successful trajectories. The initial state is in the top left corner and the goal is in the bottom right one (red circle).

In the Pinball domain (Konidaris and Barto 2009), a ball must be guided through a maze of arbitrarily shaped polygons to a designated target location. The state space is continuous over the position and velocity of the ball in the x-y plane. At every step, the agent must choose among five discrete primitive actions: move the ball faster or slower, in the vertical or horizontal direction, or take the *null* action. Collisions with obstacles are elastic and can be used to the advantage of the agent. In this domain, a drag coefficient of 0.995 effectively stops ball movements after a finite number of steps when the null action is chosen repeatedly. Each thrust action incurs a penalty of -5 while taking no action costs -1. The episode terminates with $+10000$ reward when the agent reaches the target. We interrupted any episode taking more than 10000 steps and set the discount factor to 0.99.

We used intra-option Q-learning in the critic with linear function approximation over Fourier bases (Konidaris et al. 2011) of order 3. We experimented with 2, 3 or 4 options. We used Boltzmann policies for the intra-option policies and linear-sigmoid functions for the termination functions. The learning rates were set to 0.01 for the critic and 0.001 for both the intra and termination gradients. We used an epsilon-greedy policy over options with $\epsilon = 0.01$.

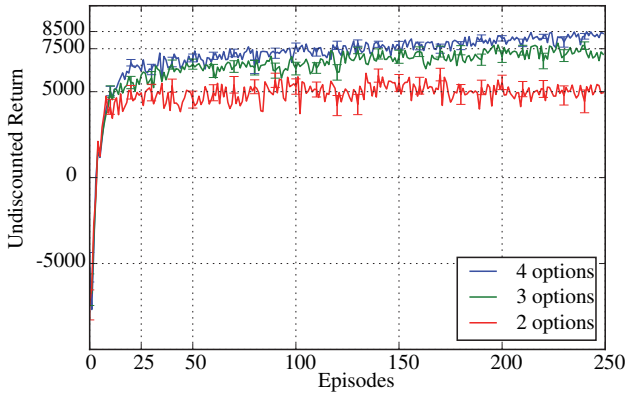

Figure 5: Learning curves in the Pinball domain.

In (Konidaris and Barto 2009), an option can only be used and updated after a *gestation* period of 10 episodes. As learning is fully integrated in option-critic, by 40 episodes a near optimal set of options had already been learned in all settings. From a qualitative point of view, the options exhibit temporal extension and specialization (fig. 4). We also observed that across many successful trajectories the *red* option would consistently be used in the vicinity of the goal.

Arcade Learning Environment

We applied the option-critic architecture in the Arcade Learning Environment (ALE) (Bellemare et al. 2013) using a deep neural network to approximate the critic and represent the intra-option policies and termination functions. We used the same configuration as (Mnih et al. 2013) for the first 3 convolutional layers of the network. We used 32 convolutional filters of size 8×8 and stride of 4 in the first layer, 64 filters of size 4×4 with a stride of 2 in the second and 64 3×3 filters with a stride of 1 in the third layer. We then fed the output of the third layer into a dense shared layer of 512 neurons, as depicted in Figure 6. We fixed the learning rate for the intra-option policies and termination gradient to 0.00025 and used RMSProp for the critic.

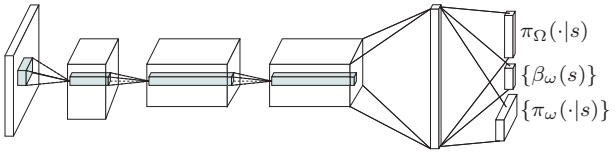

Figure 6: Deep neural network architecture. A concatenation of the last 4 images is fed through the convolutional layers, producing a dense representation shared across intra-option policies, termination functions and policy over options.

We represented the intra-option policies as linear-softmax

of the fourth (dense) layer, so as to output a probability distribution over actions conditioned on the current observation. The termination functions were similarly defined using sigmoid functions, with one output neuron per termination.

The critic network was trained using intra-option Q-learning with experience replay. Option policies and terminations were updated on-line. We used an ϵ-greedy policy over options with $\epsilon = 0.05$ during the *test* phase (Mnih et al. 2013).

As a consequence of optimizing for the return, the termination gradient tends to *shrink* options over time. This is expected since in theory primitive actions are sufficient for solving any MDP. We tackled this issue by adding a small $\xi = 0.01$ term to the advantage function, used by the termination gradient: $A_\Omega(s,\omega)+\xi = Q_\Omega(s,\omega) - V_\Omega(s)+\xi$. This term has a regularization effect, by imposing an ξ-margin between the value estimate of an option and that of the "optimal" one reflected in V_Ω. This makes the advantage function positive if the value of an option is near the optimal one, thereby *stretching* it. A similar regularizer was proposed in (Mann, Mankowitz, and Mannor 2014).

As in (Mnih et al. 2016), we observed that the intra-option policies would quickly become deterministic. This problem seems to pertain to the use of policy gradient methods with deep neural networks in general, and not from option-critic itself. We applied the regularizer prescribed by (Mnih et al. 2016), by penalizing for low-entropy intra-option policies.

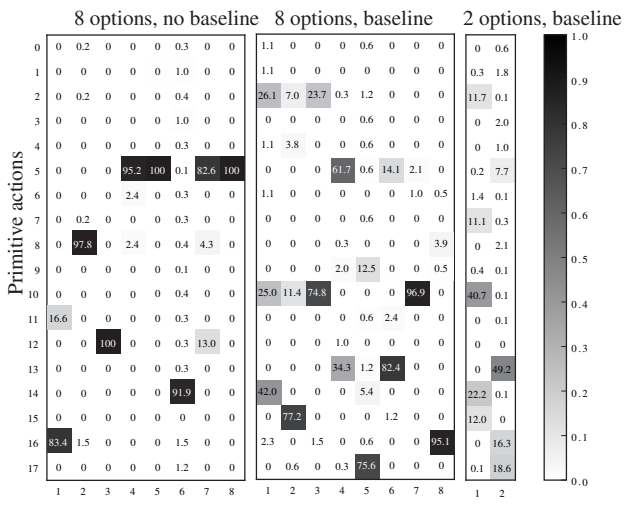

Figure 7: Seaquest: Using a baseline in the gradient estimators improves the distribution over actions in the intra-option policies, making them less deterministic. Each column represents one of the options learned in Seaquest. The vertical axis spans the 18 primitive actions of ALE. The empirical action frequencies are coded by intensity.

Finally, the baseline Q_Ω was added to the intra-option policy gradient estimator to reduce its variance. This change provided substantial improvements (Harb 2016) in the quality of the intra-option policy distributions and the overall agent performance as explained in Figure 7.

We evaluated option-critic in *Asterisk*, *Ms. Pacman*, *Seaquest* and *Zaxxon*. For comparison, we allowed the system to learn for the same number of episodes as (Mnih et al. 2013) and fixed the parameters to the same values in all four domains. Despite having more parameters to learn, option-critic was capable of learning options that would achieve the goal in all games, from the ground up, within 200 episodes (Figure 8). In Asterisk, Seaquest and Zaxxon, option-critic surpassed the performance of the original DQN architecture based on primitive actions. The eight options learned in each game are learned fully end-to-end, in tandem with the feature representation, with no prior specification of a subgoal or pseudo-reward structure.

The solution found by option-critic was easy to interpret in the game of Seaquest when learning with only two options. We found that each option specialized in a behavior sequence which would include either the up or the down button. Figure 9 shows a typical transition from one option to the other, first going upward with option 0 then switching to option 1 downward. Options with a similar structure were also found in this game by (Krishnamurthy et al. 2016) using an option discovery algorithm based on graph partitioning.

Related Work

As option discovery has received a lot of attention recently, we now discuss in more detail the place of our approach with respect to others. (Comanici and Precup 2010) used a gradient-based approach for improving only the termination function of semi-Markov options; termination was modeled by a logistic distribution over a cumulative measure of the features observed since initiation. (Levy and Shimkin 2011) also built on policy gradient methods by constructing explicitly the augmented state space and treating stopping events as additional control actions. In contrast, we do not need to construct this (very large) space directly. (Silver and Ciosek 2012) dynamically chained options into longer temporal sequences by relying on compositionality properties. Earlier work on linear options (Sorg and Singh 2010) also used compositionality to plan using linear expectation models for options. Our approach also relies on the Bellman equations and compositionality, but in conjunction with policy gradient methods.

Several very recent papers also attempt to formulate option discovery as an optimization problem with solutions that are compatible with function approximation. (Daniel et al. 2016) learn return-optimizing options by treating the termination functions as hidden variables, and using EM to learn them. (Vezhnevets et al. 2016) consider the problem of learning options that have open-loop intra-option policies, also called *macro-actions*. As in classical planning, action sequences that are more frequent are cached. A mapping from states to action sequences is learned along with a *commitment module*, which triggers re-planning when necessary. In contrast, we use closed-loop policies throughout, which are reactive to state information and can provide better solutions. (Mankowitz, Mann, and Mannor 2016) propose a gradient-based option learning algorithm, assuming a particular structure for the initiation sets and termination functions. Under this framework, exactly one option is active in any partition of the state space. (Kulkarni et al. 2016)

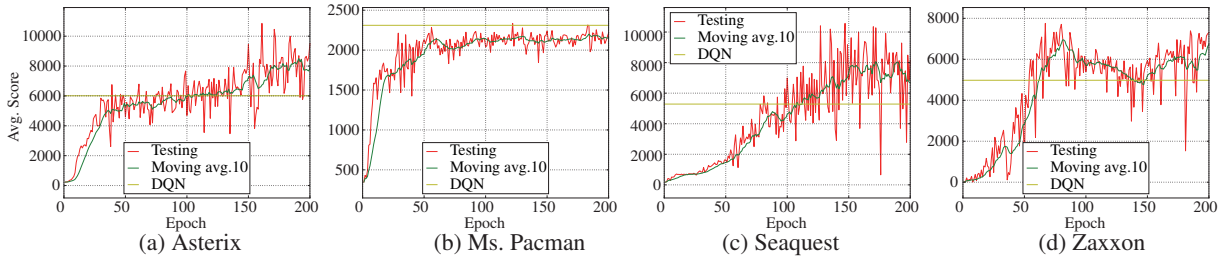

Figure 8: Learning curves in the Arcade Learning Environment. The same set of parameters was used across all four games: 8 options, 0.01 termination regularization, 0.01 entropy regularization, and a baseline for the intra-option policy gradients.

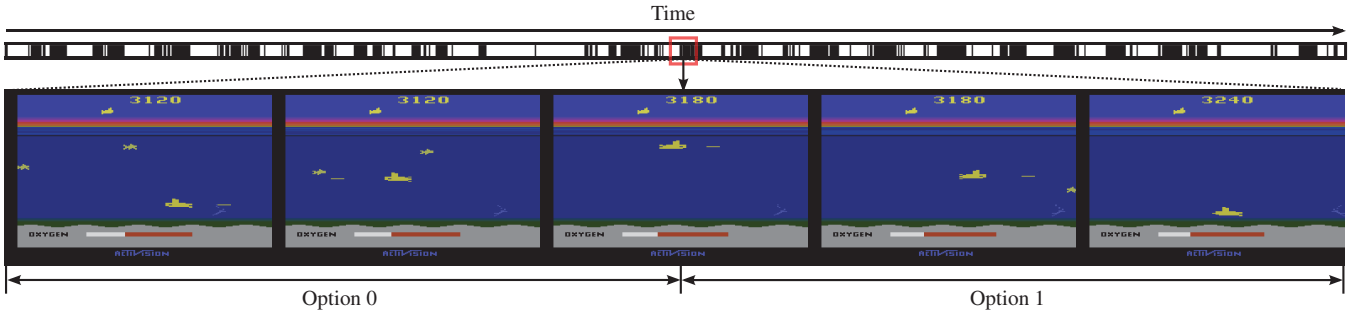

Figure 9: Up/down specialization in the solution found by option-critic when learning with 2 options in Seaquest. The top bar shows a trajectory in the game, with "white" representing a segment during which option 1 was active and "black" for option 2.

use the DQN framework to implement a gradient-based option learner, which uses intrinsic rewards to learn the internal policies of options, and extrinsic rewards to learn the policy over options. As opposed to our framework, descriptions of the subgoals are given as inputs to the option learners. Option-critic is conceptually general and does not require intrinsic motivation for learning the options.

Discussion

We developed a general gradient-based approach for learning simultaneously the intra-option policies and termination functions, as well as the policy over options, in order to optimize a performance objective for the task at hand. Our ALE experiments demonstrate successful end-to-end learning of options in the presence of nonlinear function approximation. As noted, our approach only requires specifying the number of options. However, if one wanted to use additional pseudo-rewards, the option-critic framework would easily accommodate it. In this case, the internal policies and termination function gradients would simply need to be taken with respect to the pseudo-rewards instead of the task reward. A simple instance of this idea, which we used in some of the experiments, is to use additional rewards to encourage options that are indeed temporally extended by adding a penalty whenever a switching event occurs. Our approach can work seamlessly with any other heuristic for biasing the set of options towards some desirable property (e.g. compositionality or sparsity), as long as it can be expressed as an additive reward structure. However, as seen in the results, such biasing is not necessary to produce good results.

The option-critic architecture relies on the policy gradient theorem, and as discussed in (Thomas 2014), the gradient estimators can be biased in the discounted case. By introducing factors of the form $\gamma^t \prod_{i=1}^{t}(1-\beta_i)$ in our updates (Thomas 2014, eq (3)), it would be possible to obtain unbiased estimates. However, we do not recommend this approach since the sample complexity of the unbiased estimators is generally too high and the biased estimators performed well in our experiments.

Perhaps the biggest remaining limitation of our work is the assumption that all options apply everywhere. In the case of function approximation, a natural extension to initiation sets is to use a classifier over features, or some other form of function approximation. As a result, determining which options are allowed may have similar cost to evaluating a policy over options (unlike in the tabular setting, where options with sparse initiation sets lead to faster decisions). This is akin to eligibility traces, which are more expensive than using no trace in the tabular case, but have the same complexity with function approximation. If initiation sets are to be learned, the main constraint that needs to be added is that the options and the policy over them lead to an ergodic chain in the augmented state-option space. This can be expressed as a flow condition that links initiation sets with terminations. The precise description of this condition, as well as sparsity regularization for initiation sets, is left for future work.

Acknowledgements

The authors gratefully acknowledge financial support for this work by the National Science and Engineering Research Council of Canada (NSERC) and the Fonds de recherche du Quebec - Nature et Technologies (FRQNT).

Appendix

Augmented Process

If ω_t has been initiated or is executing at time t, then the discounted probability of transitioning to (s_{t+1}, ω_{t+1}) is:

$$\mathrm{P}_\gamma^{(1)}\left(s_{t+1}, \omega_{t+1} \middle| s_t, \omega_t\right) = \sum_a \pi_{\omega_t}\left(a \middle| s_t\right) \gamma\, \mathrm{P}(s_{t+1} \middle| s_t, a) \big((1 - \beta_{\omega_t}(s_{t+1})) \mathbf{1}_{\omega_t = \omega_{t+1}} + \beta_{\omega_t}(s_{t+1}) \pi_\Omega\left(\omega_{t+1} \mid s_{t+1}\right) \big) \ .$$

When conditioning the process from (s_t, ω_{t-1}), the discounted probability of transitioning to s_{t+1}, ω_t is:

$$\mathrm{P}_\gamma^{(1)}\left(s_{t+1}, \omega_t \mid s_t, \omega_{t-1}\right) = \big((1 - \beta_{\omega_{t-1}}(s_t)) \mathbf{1}_{\omega_t = \omega_{t-1}} + \beta_{\omega_{t-1}}(s_t) \pi_\Omega\left(\omega_t \mid s_t\right) \big) \sum_a \pi_{\omega_t}\left(a \mid s_t\right) \gamma\, \mathrm{P}\left(s_{t+1} \mid s_t, a\right) \ .$$

More generally, the k-steps discounted probabilities can be expressed recursively as follows:

$$\mathrm{P}_\gamma^{(k)}\left(s_{t+k}, \omega_{t+k} \mid s_t, \omega_t\right) = \sum_{s_{t+1}} \sum_{\omega_{t+1}} \Big(\mathrm{P}_\gamma^{(1)}\left(s_{t+1}, \omega_{t+1} \mid s_t, \omega_t\right) \mathrm{P}_\gamma^{(k-1)}\left(s_{t+k}, \omega_{t+k} \mid s_{t+1}, \omega_{t+1}\right) \Big) ,$$

$$\mathrm{P}_\gamma^{(k)}\left(s_{t+k}, \omega_{t+k-1} \mid s_t, \omega_{t-1}\right) = \sum_{s_{t+1}} \sum_{\omega_t} \Big(\mathrm{P}_\gamma^{(1)}\left(s_{t+1}, \omega_t \mid s_t, \omega_{t-1}\right) \mathrm{P}_\gamma^{(k-1)}\left(s_{t+k}, \omega_{t+k-1} \mid s_{t+1}, \omega_t\right) \Big) .$$

Proof of the Intra-Option Policy Gradient Theorem

Taking the gradient of the option-value function:

$$\begin{aligned}
\frac{\partial Q_\Omega(s,\omega)}{\partial \theta} &= \frac{\partial}{\partial \theta} \sum_a \pi_{\omega,\theta}\left(a \mid s\right) Q_U(s,\omega,a) \\
&= \sum_a \left(\frac{\partial \pi_{\omega,\theta}\left(a|s\right)}{\partial \theta} Q_U(s,\omega,a) + \pi_{\omega,\theta}\left(a|s\right) \frac{\partial Q_U(s,\omega,a)}{\partial \theta} \right) \\
&= \sum_a \left(\frac{\partial \pi_{\omega,\theta}\left(a \mid s\right)}{\partial \theta} Q_U(s,\omega,a) + \pi_{\omega,\theta}\left(a \mid s\right) \sum_{s'} \gamma\, \mathrm{P}\left(s' \mid s, a\right) \frac{\partial U(\omega, s')}{\partial \theta} \right) , \quad (6) \\
\frac{\partial U(\omega, s')}{\partial \theta} &= \\
&(1 - \beta_{\omega,\vartheta}(s')) \frac{\partial Q_\Omega(s',\omega)}{\partial \theta} + \beta_{\omega,\vartheta}(s') \frac{\partial V_\Omega(s')}{\partial \theta} \\
&= (1 - \beta_{\omega,\vartheta}(s')) \frac{\partial Q_\Omega(s',\omega)}{\partial \theta} + \beta_{\omega,\vartheta}(s') \sum_{\omega'} \pi_\Omega\left(\omega' \mid s'\right) \frac{\partial Q_\Omega(s',\omega')}{\partial \theta} \\
&= \sum_{\omega'} \big((1 - \beta_{\omega,\vartheta}(s')) \mathbf{1}_{\omega' = \omega} + \beta_{\omega,\vartheta}(s') \pi_\Omega\left(\omega' \mid s'\right) \big) \frac{\partial Q_\Omega(s',\omega')}{\partial \theta} \ . \quad (7)
\end{aligned}$$

where (7) follows from the assumption that θ only appears in the intra-option policies. Substituting (7) into (6) yields a recursion which, using the previous remarks about augmented process can be transformed into:

$$\begin{aligned}
\frac{\partial Q_\Omega(s,\omega)}{\partial \theta} &= \sum_a \frac{\partial \pi_{\omega,\theta}\left(a \mid s\right)}{\partial \theta} Q_U(s,\omega,a) + \\
&\sum_a \pi_{\omega,\theta}\left(a \mid s\right) \sum_{s'} \gamma\, \mathrm{P}\left(s' \mid s, a\right) \sum_{\omega'} \Big(\beta_{\omega,\vartheta}(s') \pi_\Omega\left(\omega' \mid s'\right) \\
&\qquad + (1 - \beta_{\omega,\vartheta}(s')) \mathbf{1}_{\omega' = \omega} \Big) \frac{\partial Q_\Omega(s',\omega')}{\partial \theta} \\
&= \sum_a \frac{\partial \pi_{\omega,\theta}\left(a \mid s\right)}{\partial \theta} Q_U(s,\omega,a) + \\
&\qquad \sum_{s'} \sum_{\omega'} \mathrm{P}_\gamma^{(1)}\left(s', \omega' \mid s, \omega\right) \frac{\partial Q_\Omega(s',\omega')}{\partial \theta} \\
&= \sum_{k=0}^{\infty} \sum_{s',\omega'} \mathrm{P}_\gamma^{(k)}\left(s', \omega' | s, \omega\right) \sum_a \frac{\partial \pi_{\omega',\theta}\left(a|s'\right)}{\partial \theta} Q_U(s',\omega',a).
\end{aligned}$$

The gradient of the expected discounted return with respect to θ is then:

$$\begin{aligned}
&\frac{\partial Q_\Omega(s_0,\omega_0)}{\partial \theta} = \\
&\sum_{s,\omega} \sum_{k=0}^{\infty} \mathrm{P}_\gamma^{(k)}\left(s, \omega \mid s_0, \omega_0\right) \sum_a \frac{\partial \pi_{\omega,\theta}\left(a \mid s\right)}{\partial \theta} Q_U(s,\omega,a) \\
&= \sum_{s,\omega} \mu_\Omega(s,\omega|s_0,\omega_0) \sum_a \frac{\partial \pi_{\omega,\theta}\left(a \mid s\right)}{\partial \theta} Q_U(s,\omega,a) \ .
\end{aligned}$$

Proof of the Termination Gradient Theorem

The expected sum of discounted rewards starting from (s_1, ω_0) is given by:

$$U(\omega_0, s_1) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \,\middle|\, s_1, \omega_0 \right] \ .$$

We start by expanding U as follows:

$$\begin{aligned}
U(\omega, s') &= (1 - \beta_{\omega,\vartheta}(s')) Q_\Omega(s', \omega) + \beta_{\omega,\vartheta}(s') V_\Omega(s') \\
&= (1 - \beta_{\omega,\vartheta}(s')) \sum_a \pi_{\omega,\theta}\left(a \mid s'\right) \Big(r(s', a) + \sum_{s''} \gamma\, \mathrm{P}\left(s'' \mid s', a\right) U(\omega, s'') \Big) \\
&+ \ \beta_{\omega,\vartheta}(s') \sum_{\omega'} \pi_\Omega\left(\omega' \mid s'\right) \sum_a \pi_{\omega',\theta}\left(a \mid s'\right) \Big(r(s', a) + \sum_{s''} \gamma\, \mathrm{P}\left(s'' \mid s', a\right) U(\omega', s'') \Big) \ .
\end{aligned}$$

The gradient of U is then:

$$\frac{\partial U(\omega, s')}{\partial \vartheta} = \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} \underbrace{\left(V_\Omega(s') - Q_\Omega(s', \omega)\right)}_{-A_\Omega(s',\omega)} + (1 - \beta_{\omega,\vartheta}(s')) \sum_a \pi_{\omega,\theta}\left(a|s'\right) \sum_{s''} \gamma\, \mathrm{P}\left(s''|s', a\right) \frac{\partial U(\omega, s'')}{\partial \vartheta} .$$

Using the structure of the augmented process:

$$\frac{\partial U(\omega, s')}{\partial \vartheta} = -\frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} A_\Omega(s', \omega) + \sum_{\omega'} \sum_{s''} \mathrm{P}_\gamma^{(1)}(s'', \omega' \mid s', \omega) \frac{\partial U(\omega', s'')}{\partial \vartheta}$$

$$= -\sum_{\omega', s''} \sum_{k=0}^{\infty} \mathrm{P}_\gamma^{(k)}(s'', \omega' \mid s', \omega) \frac{\partial \beta_{\omega',\vartheta}(s'')}{\partial \vartheta} A_\Omega(s'', \omega') \ .$$

We finally obtain:

$$\frac{\partial U(\omega_0, s_1)}{\partial \vartheta} = -\sum_{\omega, s'} \sum_{k=0}^{\infty} \mathrm{P}_\gamma^{(k)}(s', \omega \mid s_1, \omega_0) \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} A_\Omega(s', \omega)$$

$$= -\sum_{\omega, s'} \mu_\Omega(s', \omega | s_1, \omega_0) \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} A_\Omega(s', \omega) \ .$$

References

Baird, L. C. 1993. Advantage updating. Technical Report WL–TR-93-1146, Wright Laboratory.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47:253–279.

Comanici, G., and Precup, D. 2010. Optimal policy switching algorithms for reinforcement learning. In *AAMAS*, 709–714.

Şimşek, O., and Barto, A. G. 2009. Skill characterization based on betweenness. In *NIPS 21*, 1497–1504.

Daniel, C.; van Hoof, H.; Peters, J.; and Neumann, G. 2016. Probabilistic inference for determining options in reinforcement learning. *Machine Learning, Special Issue* 104(2):337–357.

Harb, J. 2016. Learning options in deep reinforcement learning. Master's thesis, McGill University.

Konda, V. R., and Tsitsiklis, J. N. 2000. Actor-critic algorithms. In *NIPS 12*, 1008–1014.

Konidaris, G., and Barto, A. 2009. Skill discovery in continuous reinforcement learning domains using skill chaining. In *NIPS 22*, 1015–1023.

Konidaris, G.; Kuindersma, S.; Grupen, R. A.; and Barto, A. G. 2011. Autonomous skill acquisition on a mobile manipulator. In *AAAI*.

Krishnamurthy, R.; Lakshminarayanan, A. S.; Kumar, P.; and Ravindran, B. 2016. Hierarchical reinforcement learning using spatio-temporal abstractions and deep neural networks. *CoRR* abs/1605.05359.

Kulkarni, T.; Narasimhan, K.; Saeedi, A.; and Tenenbaum, J. 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NIPS 29*.

Levy, K. Y., and Shimkin, N. 2011. Unified inter and intra options learning using policy gradient methods. In *EWRL*, 153–164.

Mankowitz, D. J.; Mann, T. A.; and Mannor, S. 2016. Adaptive skills, adaptive partitions (ASAP). In *NIPS 29*.

Mann, T. A.; Mankowitz, D. J.; and Mannor, S. 2014. Time-regularized interrupting options (TRIO). In *ICML*, 1350–1358.

Mann, T. A.; Mannor, S.; and Precup, D. 2015. Approximate value iteration with temporally extended actions. *Journal of Artificial Intelligence Research* 53:375–438.

McGovern, A., and Barto, A. G. 2001. Automatic discovery of subgoals in reinforcement learning using diverse density. In *ICML*, 361–368.

Menache, I.; Mannor, S.; and Shimkin, N. 2002. Q-cut - dynamic discovery of sub-goals in reinforcement learning. In *ECML*, 295–306.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. A. 2013. Playing atari with deep reinforcement learning. *CoRR* abs/1312.5602.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T. P.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *ICML*.

Niekum, S. 2013. *Semantically Grounded Learning from Unstructured Demonstrations*. Ph.D. Dissertation, University of Massachusetts, Amherst.

Precup, D. 2000. *Temporal abstraction in reinforcement learning*. Ph.D. Dissertation, University of Massachusetts, Amherst.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.

Silver, D., and Ciosek, K. 2012. Compositional planning using optimal option models. In *ICML*.

Sorg, J., and Singh, S. P. 2010. Linear options. In *AAMAS*, 31–38.

Stolle, M., and Precup, D. 2002. Learning options in reinforcement learning. In *Abstraction, Reformulation and Approximation, 5th International Symposium, SARA Proceedings*, 212–223.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS 12*. 1057–1063.

Sutton, R. S.; Precup, D.; and Singh, S. P. 1999. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112(1-2):181–211.

Sutton, R. S. 1984. *Temporal Credit Assignment in Reinforcement Learning*. Ph.D. Dissertation.

Thomas, P. 2014. Bias in natural actor-critic algorithms. In *ICML*, 441–448.

Vezhnevets, A. S.; Mnih, V.; Agapiou, J.; Osindero, S.; Graves, A.; Vinyals, O.; and Kavukcuoglu, K. 2016. Strategic attentive writer for learning macro-actions. In *NIPS 29*.