Options of Interest: Temporal Abstraction with Interest Functions

Khimya Khetarpal,[1,2] **Martin Klissarov,**[1,2] **Maxime Chevalier-Boisvert,**[2,3]
Pierre-Luc Bacon,[4] **Doina Precup**[1,2,5]
[1]McGill University, [2]Mila, [3]Universite de Montreal,
[4]Stanford University, [5]Google DeepMind
khimya.khetarpal@mail.mcgill.ca, martin.klissarov@mail.mcgill.ca
maxime.chevalier-boisvert@mila.quebec, plbacon@cs.stanford.edu, dprecup@cs.mcgill.ca

Abstract

Temporal abstraction refers to the ability of an agent to use behaviours of controllers which act for a limited, variable amount of time. The options framework describes such behaviours as consisting of a subset of states in which they can initiate, an internal policy and a stochastic termination condition. However, much of the subsequent work on option discovery has ignored the initiation set, because of difficulty in learning it from data. We provide a generalization of initiation sets suitable for general function approximation, by defining an interest function associated with an option. We derive a gradient-based learning algorithm for interest functions, leading to a new interest-option-critic architecture. We investigate how interest functions can be leveraged to learn interpretable and reusable temporal abstractions. We demonstrate the efficacy of the proposed approach through quantitative and qualitative results, in both discrete and continuous environments.

Introduction

Humans have a remarkable ability to acquire skills, and knowing when to apply each skill plays an important role in their ability to quickly solve new tasks. In this work, we tackle the problem of learning such skills in reinforcement learning (RL). AI agents which aim to achieve goals are faced with two difficulties in large problems: the depth of the lookahead needed to obtain a good solution, and the breadth generated by having many choices. The first problem is often solved by providing shortcuts that skip over multiple time steps, for example, by using macro-actions (Hauskrecht et al. 1998). The second problem is handled by restricting the agent's attention at each step to a reasonable number of possible choices. Temporal abstraction methods aim to solve the first problem, and a lot of recent literature has been devoted to this topic (2011; 2015; 2016; 2016; 2017; 2017). We focus specifically on the second problem: learning how to reduce the number of choices considered by an RL agent.

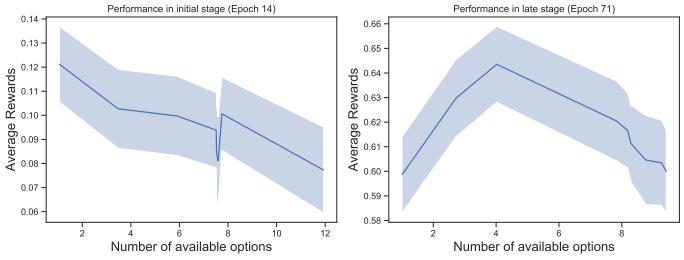
In classical planning, the early work on STRIPS (Fikes, Hart, and Nilsson 1972) used preconditions that had to be satisfied before applying a certain action. Similar ideas can also be found in later work on macro-operators (Korf 1983) or the Schema model (Drescher 1991). In RL, the framework of options (Sutton, Precup, and Singh 1999) uses a similar concept, *initiation sets*, which limit the availability of options (i.e. temporally extended actions) in order to deal with the possibly high cost of choosing among many options. Moreover, initiation sets can also lead to options that are more *localized* (Konidaris and Barto 2007), which can be beneficial in transfer learning. For example, in continual learning (1997), specialization is key to both scaling up learning in large environments, as well as to "protecting" knowledge that has already been learned from forgetting due to new updates.

The option-critic architecture (Bacon, Harb, and Precup 2017) is a gradient-based approach for learning options in order to optimize the usual long-term return obtained by an RL agent from the environment. However, the notion of initiation sets originally introduced in Sutton, Precup, and Singh (1999) was omitted from Bacon, Harb, and Precup (2017) due to the difficulty of learning sets with gradient-based methods. We propose a generalization of initiation sets to *interest functions* (Sutton, Mahmood, and White 2016; White 2017). We build from the fact that a set can be represented through its membership function. Interest functions are a generalization of membership functions which allows smooth parameterization. Without this extension, determining suitable initiation sets would necessitate a non-differentiable, search-based approach.

Key Contributions: We generalize initiation sets for options to *interest functions*, which are differentiable, and hence easier to learn. We derive a gradient-based learning algorithm capable of learning all components of options end-to-end. The resulting interest-option-critic architecture generates options that are specialized to different regions of state space. We demonstrate through experiments in both discrete and continuous problems that our approach generates options that are useful in a single task, interpretable and reusable in multi-task learning.

Preliminaries

Markov Decision Processes (MDPs). A finite, discrete-time MDP (Puterman 1995; Sutton and Barto 1998) is a



(a) (b)

Figure 1: **Interest functions and the branching factor**. During the initial stages of the learning process, allowing fewer options helps improve learning speed, whereas in the later stages, good solutions can still be obtained with a reasonable number of choices at each decision point.

tuple $\langle \mathcal{S}, \mathcal{A}, r, P, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $P : \mathcal{S} \times \mathcal{A} \to Dist(\mathcal{S})$ is the environment transition probability function, and $\gamma \in [0, 1)$ is the discount factor. At each time step, the learning agent perceives a state $S_t \in \mathcal{S}$, takes an action $A_t \in \mathcal{A}$ drawn from a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, and with probability $P(S_{t+1}|S_t, A_t)$ enters next state S_{t+1}, receiving a numerical reward $R_{t+1} = r(S_t, A_t)$ from the environment. The value function of policy π is defined as: $V_\pi(s) = E_\pi[\sum_{t=0}^{\infty} \gamma^t R_{t+1}|S_0 = s]$ and its action-value function is: $Q_\pi(s, a) = E_\pi[\sum_{t=0}^{\infty} \gamma^t R_{t+1}|S_0 = s, A_0 = a]$.

The Options framework. A Markovian option (Sutton, Precup, and Singh 1999) $\omega \in \Omega$ is composed of an *intra-option policy* π_ω, a termination condition $\beta_\omega : \mathcal{S} \to [0, 1]$, where $\beta_\omega(s)$ is the probability of terminating the option upon entering state s, and an initiation set $I_\omega \subseteq \mathcal{S}$. In the *call-and-return* option execution model, when an agent is in state s, it first examines the options that are available, i.e., for which $s \in I_\omega$. Let $\Omega(s)$ denote this set of available options. The agent then chooses $\omega \in \Omega(s)$ according to the policy over options $\pi_\Omega(s)$, follows the internal policy of ω, π_ω, until it terminates according to β_ω, at which point this process is repeated. Note that Ω is the union of all sets $\Omega(s), \forall s$. The option-value function of $\omega \in \Omega(s)$ is defined as:

$$Q_\Omega(s, \omega) = \sum_a \pi_\omega(a|s) Q_U(s, \omega, a) \ ,$$

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

$$Q_U(s, \omega, a) = r(s, a) + \gamma \sum_{s'} P(s'|s, a) \times \left((1 - \beta_\omega(s')) Q_\Omega(s', \omega) + \beta_\omega(s') \max_{\omega' \in \Omega(s')} Q_\Omega(s', \omega') \right)$$

Note that if an option cannot initiate in a state s, its value is considered undefined.

Interest-Option-Critic

In Sutton, Precup, and Singh (1999), the focus is on discrete environments, and the notion of initiation set provides a direct analog of preconditions from classical planning. In large problems, options would be applicable in parts of the state space described by certain features. For example, an option of the form *stop if the traffic light is red* would only need to be considered in states where a traffic light is detected. Let $\mathbf{1}_\omega : \mathcal{S} \to \{0, 1\}$ be the indicator function corresponding to set I_ω: $\mathbf{1}_\omega(s) = 1$ iff $s \in I_\omega$ and 0 otherwise.

An *interest function* $I_\omega : \mathcal{S} \times \Omega \to \mathbb{R}^+$ generalizes the set indicator function, with $I_\omega(s) > 0$ iff ω can be initiated in s. A bigger value of $I_\omega(s)$ means the interest in executing ω in s is larger. Note that, depending on how I_ω is parameterized, one could view the interest as a prior on the likelihood of executing ω in s. However, we will not use this perspective here, because our goal is to learn I_ω. So, instead, we will choose a parameterized form for I_ω which is differentiable, in order to leverage the power of gradients in the learning process.

The value of I_ω modulates the probability of option ω being sampled in state s by a policy over options π_Ω, resulting in an *interest policy over option* defined as:

$$\pi_{I_\omega}(\omega|s) \propto I_\omega(s) \pi_\Omega(\omega|s) \tag{1}$$

Note that this specializes immediately to usual initiation sets (where the interest is the indicator function).

We will now describe an approach to learning options which includes interest functions. We propose a policy gradient algorithm, in the style of option-critic (2017), based on the following result:

Theorem 1. *Given a set of Markov options with differentiable interest functions $I_{\omega,z}$, where z is the parameter vector, the gradient of the expected discounted return with respect to z at (s, ω) is:*

$$\sum_{s', \omega'} \hat{\mu}_\Omega(s', \omega'|s, \omega) \beta_\omega(s') \frac{\partial \pi_{I_{\omega,z}}(\omega'|s')}{\partial z} Q_\Omega(s', \omega')$$

where $\hat{\mu}_\Omega(s', \omega'|s, \omega)$ is the discounted weighting of the state-option pairs along trajectories starting from (s, ω) sampled from the distribution determined by $\pi_{I_{\omega,z}}$, β_ω is the termination function and Q_Ω is the value function over options corresponding to $\pi_{I_{\omega,z}}$.

The proof is in Appendix A.2.1 available on the project page[1]. We can derive policy gradients for intra-option policies and termination functions as in option-critic (Bacon,

[1] https://sites.google.com/view/optionsofinterest

Harb, and Precup 2017) (see Appendix A.2.2, A.2.3) with the difference that the discounted weighting of state-option pairs is now according to the new option sampling distribution determined by $\pi_{I_{\omega,z}}(s)$. This is natural, as the introduction of the interest function should only impact the choice of option in each state. Pseudo-code of the interest-option-critic (IOC) algorithm using intra-option Q-learning is shown in Algorithm 1.

Intuitively, the gradient update to z can be interpreted as increasing the interest in an option which terminates in states with good value. It links initiation and termination, which is natural. It is to be noted that the proposed gradient works at the level of the augmented chain; and not at the SMDP level. Implementing policy gradient at the SMDP level for the policy over options would entail performing gradient updates only upon termination, whereas using the augmented chain allows for updates throughout. Note that this approach does not appear in previous work to the best of our knowledge.

Algorithm 1 IOC with tabular intra-option Q-learning

Initialize policy over options π_Ω
Initialize $I_{\omega,z}$ parameterized by z such that all options are available everywhere to some extent
Initialize $\pi_{I_{\omega,z}}(\omega|s)$ as in Eq.(1)
Set $s \longleftarrow s_0$ and ω at s according to $\pi_{I_{\omega,z}}$
repeat
  Choose a according to $\pi_{\omega,\theta}(a|s)$
  Take action a in s, observe s', r
  Sample termination from $\beta_\omega, \nu(s')$
  if ω terminates in s' **then**
    Sample ω' according to $\pi_{I_{\omega,z}}(\cdot|s')$
  else
    $\omega' = \omega$
  end if
  1. Evaluation step:
  $\delta \leftarrow r - Q_U(s,\omega,a)$
  $\delta \leftarrow r + \gamma(1 - \beta_{\omega,\nu}(s'))Q_\Omega(s',\omega) + \gamma\beta_{\omega,\nu}(s')\max_{\omega'} Q_\Omega(s',\omega')$
  $Q_U(s,\omega,a) \leftarrow Q_U(s,\omega,a) + \alpha\delta$
  2. Improvement step
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log \pi_{\omega,\theta}(a|s)}{\partial\theta} Q_U(s,\omega,a)$
  $\nu \leftarrow \nu - \alpha_\nu \frac{\partial \beta_{\omega,\nu}(s')}{\partial\nu}(Q_\Omega(s',\omega) - V_\Omega(s'))$ where $V_\Omega(s') = \sum'_\omega \pi_{I_{\omega,z}}(\omega'|s')Q_\Omega(s',\omega')$
  $z \leftarrow z + \alpha_z \beta_{\omega,\nu}(s') \frac{\partial \pi_{I_{\omega,z}}(\omega'|s')}{\partial z} Q_\Omega(s',\omega')$
  $s \leftarrow s'$
until s' is a terminal state

Illustration

In order to elucidate the way in which interest functions can help regulate the complexity of learning how to make decisions, we provide a small illustration of the idea. Consider a point mass agent in a continuous 2-D maze, which starts in a uniformly random position and must reach a fixed goal state. Consider a scalar threshold $k \in [0, 1]$, so that at any choice point, only options whose interest is at least k can be initiated by the interest policy over options $\pi_{I_\omega}(\omega|s)$. The agent uses 16 options in total. Intuitively, an agent which has fewer option choices at a decision point should learn faster, since it has fewer alternatives to explore, but in the long run, this limits the space of usable policies for the agent. Fig. 1 confirms this trade-off between speed of learning and ultimate quality. Note that this trade-off holds the same way in planning as well (as discussed extensively in classical planning works).

Experimental Results

We now study the empirical behavior of IOC in order to answer the following questions: (1) are options with interest functions useful in a single task; (2) do interest functions facilitate learning reusable options and, (3) do interest functions provide better interpretability of the skills of an agent. A link to the source code for all experiments is provided on the project page.

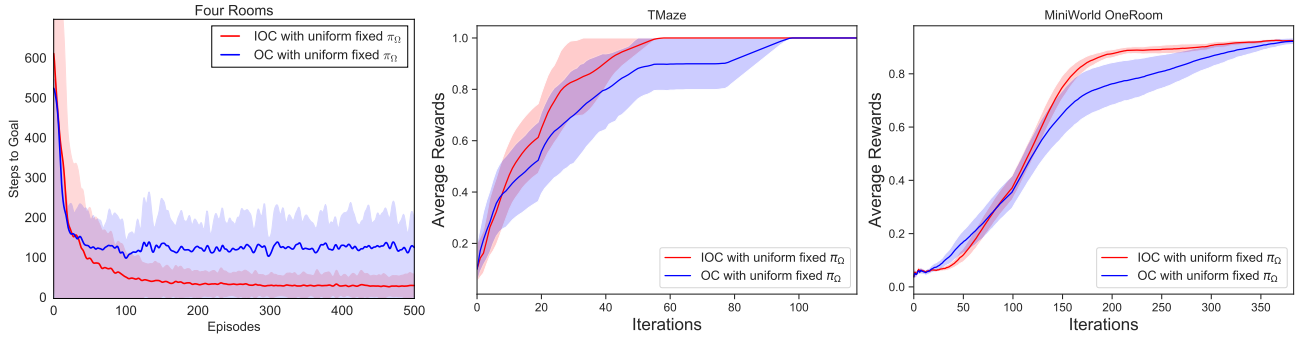
Learning in a single task

To analyze the utility of interest functions when learning in a single task, consider a given, fixed policy over options, either specified by a just-in-time planner or via human input. This setup allows us to understand the impact of interest functions alone in the learning process.

Four rooms (FR) We first consider the classic FR domain (1999) (Fig. 3(a)). The agent starts at a uniformly random state and there is a *goal* state in the bottom right hallway ($\gamma = 0.9$). With probability $1/3$, the agent transitions randomly to one of the empty adjacent cells instead of the desired movement. The reward is $+50$ at the goal and 0 otherwise. We used 4 options, whose intra-option policies were parameterized with Boltzmann distributions, and termination and interest functions represented as linear-sigmoid functions. Options were learned using either IOC or OC with tabular intra-option Q-learning, as described in Algorithm 1. Learning proceeds for 500 episodes, with a maximum of 2000 time steps allowed per episode. Additional details are provided in Appendix A.3.1.

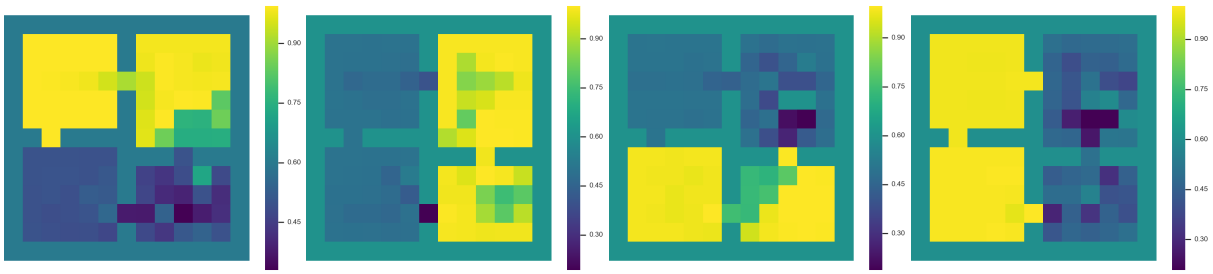
Results: Fig. 2(a) shows the steps to goal for both OC and IOC, averaged over 70 independent runs. The IOC agent performs better than OC agent by roughly 100 steps. One potential reason for the improvement in IOC is that options become specialized to different regions of the state-space, as can be seen in Fig. 3. We also observe that the termination functions (which were initialized to 0) naturally become coherent with the interest functions learned, and are mostly room specific for each option (see appendix Fig. A1). On the other hand, options learned by OC do not show such specialization and terminate everywhere (see appendix Fig. A1). These results demonstrate that the IOC agent is not only able to correct for the given higher level policy, but also, leads to more understandable options as a side effect.

TMaze Next, we illustrate the learning and use of interest functions in the non-linear function approximation setting, using simple continuous control tasks implemented in Mujoco (Todorov, Erez, and Tassa 2012). A point mass agent (blue) is located at the bottom end of a T-shaped maze (0, -0.1) and must navigate to within 0.1 of the goal posi-



(a) (b) (c)

Figure 2: **Learning options with interest reshapes a uniform fixed policy over options** in a single task, in 3 different domains: (a) tabular Four rooms, (b) continuous control in TMaze, and (c) 3D visual Miniworld task. IOC outperforms OC, indicating the utility of interest functions.



(a) I_{ω_1} (b) I_{ω_2} (c) I_{ω_3} (d) I_{ω_4}

Figure 3: **Visualization of Interest Functions** (I_ω) at the end of 500 episodes with the goal in the east hallway. Brighter colors represent higher values. Options learned with interest functions emerge with specific interest in different regions of the state.

tion (0.3, 0.3) at the right end of the maze (green location) (Fig. 4(e)). The state space consists of the x, y coordinates of the agent and the action space consists of force applied in the x, y directions. We use a uniform fixed policy over options for both IOC and OC. We reuse the Proximal Policy Option-Critic (PPOC) algorithm (Klissarov et al. 2017)[2] and add a 2-layer network with sigmoid outputs to compute the interest functions. However, we do correct the implementation of the gradient of the policy over options which has been overlooked in that work. The remaining update rules are consistent with Algorithm 1. Complete details about the implementation and hyper-parameter search are provided in Appendix A.3.2.

Results: We report the average performance over 10 independent runs. The IOC agent is able to converge in almost half the time steps needed by the OC agent. Potentially, interest functions in the IOC agent provide an attention mechanism and thus facilitates learning options which are more diverse (see Fig. 5 for evidence). A deeper analysis of interest functions learned in this domain is deferred to subsequent sections.

MiniWorld We also explore learning in more complex 3D first-person visual environment from the MiniWorld framework (Chevalier-Boisvert 2018). We use the *Oneroom* task where the agent has to navigate to a randomly placed red block in a closed room (Fig. 4(f)). This requires the agent to turn around and scan the room to find the red box, then navigate to it.

The observation space is a 3-channel RGB image and the action space consists of 8 discrete actions. At the start of each episode, the red box is placed randomly. The episode terminates if the agent reaches the box or max of 180 time steps is reached. We used the DQN architecture of (Mnih et al. 2015). See Appendix A.3.3 for details about implementation and hyper-parameters.

Results: The IOC agent is able to converge much faster (100 iterations) than the OC agent with a given uniform policy over option (Fig. 2(c)). The performance is averaged across 10 runs.

Based on these experiments, IOC provides improvement in performance consistently across a range of tasks, from the simple four-rooms domain to complex visual navigation tasks such as MiniWorld, indicating the utility of learning interest functions.

Option reusability

One of the primary reasons for an agent to autonomously learn options is the ability to generalize its knowledge quickly in new tasks. We now evaluate our approach in settings where adaptation to changes in the task is vital.

[2]In this work we name this algorithm OC for option-critic

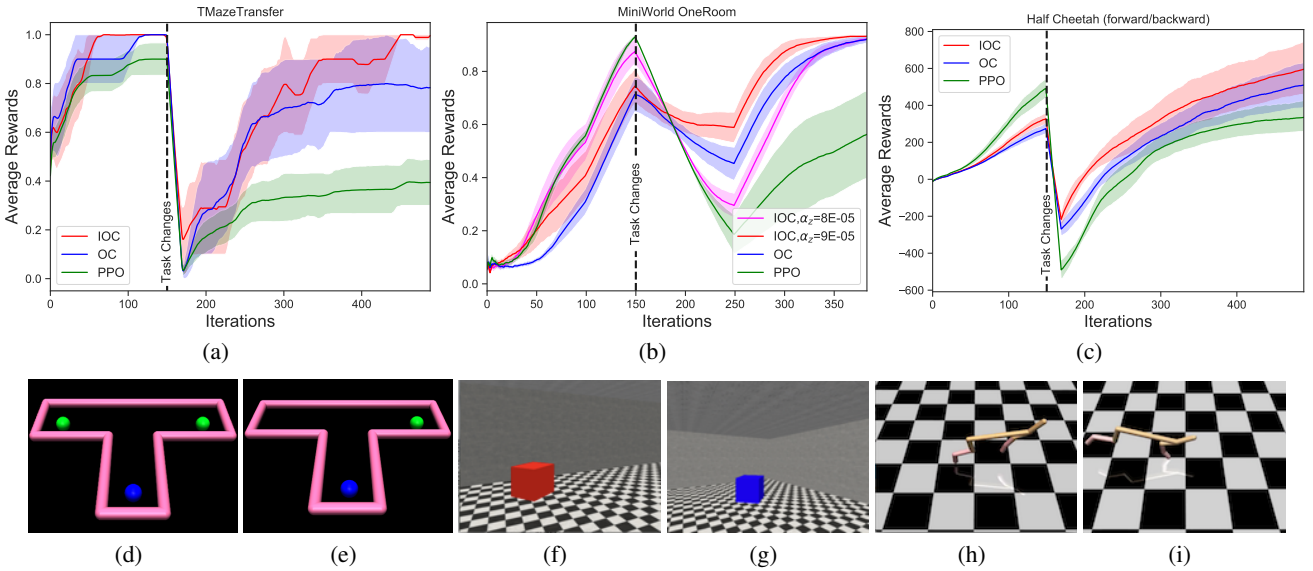


Figure 4: **Transfer in continual learning setting:** *Continuous Control in TMaze:* (d) The point mass agent (blue) has two goals (green) both resulting in a reward of $+1$. After 150 iterations, the goal that is most visited is removed (e). (a) illustrates that IOC converges fastest in the first few iterations. After the task change, IOC suffers the least in terms of immediate loss in performance and gets the best final score. *Visual navigation in Miniworld:* requires the agent to go to a randomly placed red box in a closed room. After 150 episodes the color of the box changes and the agent has to navigate to a unseen blue box (g). IOC quickly adapts to the change (b) indicating that harnessing options learned from the old task speeds up learning in the new task. *Locomotion in HalfCheetah:* The cheetah is rewarded for moving forward as fast as possible during the first 150 iterations, after which it is rewarded for going backwards as fast as possible.

TMaze The point mass agent starts at the bottom of the maze, with two goal locations (Fig. 4(d)), both giving a reward of $+1$. After 150 episodes, the goal that has been visited the most is removed and the agent has to adapt its policy to the remaining goal available (Fig. 4(e)). Both OC and IOC learn 2 options. We use a softmax policy over options for both IOC and OC, which is also learned at the same time.

Results: In the initial phase, the difference in performance between IOC and the other two agents (OC and PPO) is striking (Fig. 4(a)): IOC converges twice as fast. Moreover, when the most visited goal is removed and adaptation to the task change is required, the IOC agent is able to explore faster and its performance declines less. This suggests that the structure learned by IOC provides more generality. At the end of task 2, IOC recovers its original performance, whereas PPO fails to recover during thee allotted learning trials.

MiniWorld Initially, the agent is tasked to search and navigate to a randomly placed red box in one closed room (Fig. 4(f)). After 150 episodes, the agent has to adapt its skills to navigate to a randomly located blue box (Fig. 4(g)) which it has never seen before. Here, the policy over options as well as all the option components are being learned at the same time.

Results: The IOC agent outperforms both OC and PPO agents when required to adapt to the new task (Fig. 4(b)). This result indicates that the options learned with interest functions are more easily transferable. The IOC agent is able to adapt faster to unseen scenarios.

HalfCheetah We also study adaptation in learning a complex locomotion task for a planar cheetah. The initial configuration of this environment follows the standard HalfCheetah-v1 from OpenAI's Gym: the agent is rewarded for moving forward as fast as possible. After 150 iterations, we modify the reward function so that the agent is now encouraged to move backward as fast as possible (Finn, Abbeel, and Levine 2017).

Results: PPO outperforms both OC and IOC in the initial task. However, as soon as the task changes, IOC reacts in the most efficient way and converges to the highest score after 500 iterations (Fig. 4(c)). As seen consistently in all the environments, IOC generalizes much better over tasks, whereas PPO seems to overfit to the first task and generalizes poorly when the task changes.

In all our experiments, we notice that interest functions result in option specialization, which leads to both reusability and *adaptability* (i.e. an option may get slightly tweaked), especially in the complex tasks.

Option interpretability

To gain a better understanding of the agent's behavior, we visualize different aspects of the learning process in several tasks.

TMaze We visualize the interest functions learned in TMaze (Fig. 5). Initially, the interest functions are random-

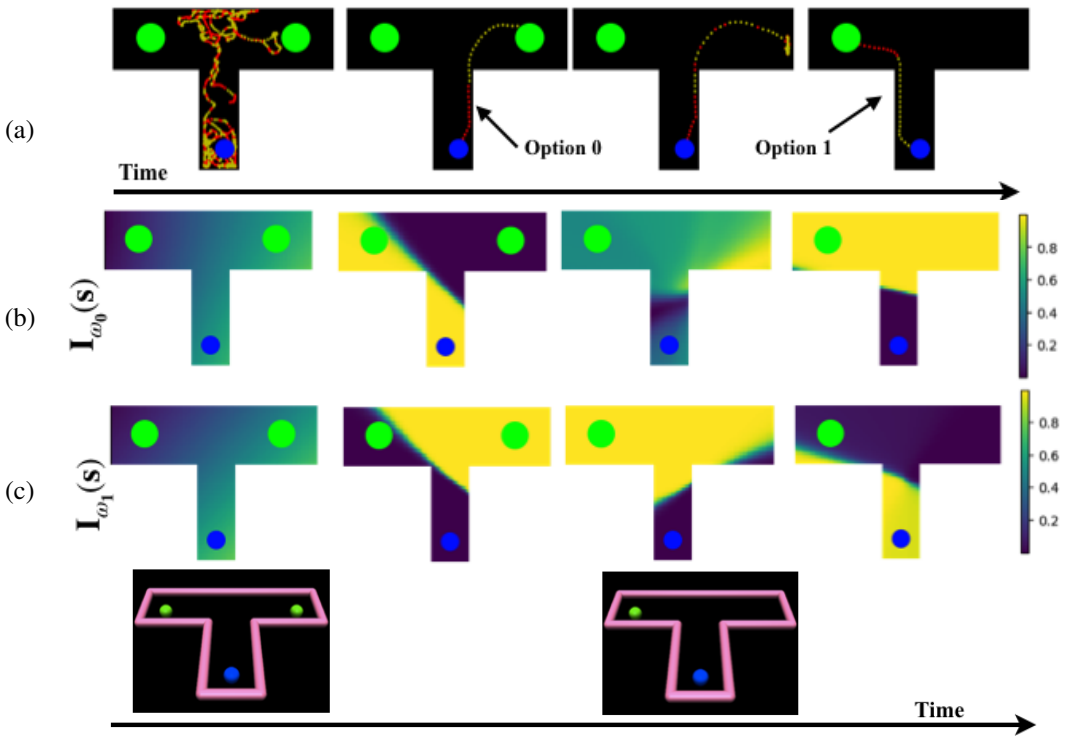


Figure 5: **Qualitative analysis of IOC in TMaze** illustrates the options and their interest over time. **Row a** depicts sampled trajectories with option 0 indicated by red dots and option 1 by yellow dots. **Row b & c** depict the interest of each option. At the end of task 1, the interest of option 0 emerges in the lower diagonal (row b) whereas option 1 is interested in a different region (row c). As the task changes, the interest functions adapt (col 3).

ized. At the end of the first task, the interest function for option 0 specializes in the lower diagonal of the state-space (Fig. 5(b)), whereas option 1's interest function is completely different (Fig. 5(c)). When the task changes, the options readjust their interest. Eventually, the interest functions for the two options automatically specialize in *different* regions of the state space (last column of Fig. 5(c) & 5(b)). Fig. 5(a) illustrates the agent trajectories at different time instances, where the yellow and red dots indicate the two different options during the trajectory. A visualization of the emergence of interest functions during the learning process is also available on the project page (see 1). In contrast, the options learned by the OC agent are employed everywhere and have not specialized as much (see Appendix Fig. A2).

HalfCheetah We analyze the skills learned in HalfCheetah. During the task of moving forward as fast as possible, the IOC agent employs option 0 to move forward by dragging its limbs, and option 1 to take much larger hopped steps (Fig. 6). Fig. 6 demonstrates the emergence of these very distinct skills and the agent's switching between them across time. Additionally, we analyzed each option at the end of task 2 in which the agent was rewarded for moving backward. Option 0 now specializes in moving forward while option 1 focuses on moving backward. This is nice, as the agent preserves some ability to now solve both tasks. OC doesn't learn options which are as distinct, and both options end up going backward and overfitting to the new task (see accompanying videos (1)).

MiniWorld We visualize the skills acquired by inspecting the agent's behavior at the end of first task. The IOC agent has learned two distinct options: option 0 scans the surroundings, whereas option 1 is used to directly navigate towards the block upon successfully locating it (Fig. 7). During task 2, option 1 is being harnessed primarily to move forward, whereas option 0 is employed when jittery motion is involved, such as turning and scanning.

Related Work

Temporal abstraction in RL has a rich history (Parr and Russell 1998; Thrun and Schwartz 1995; Dayan and Hinton 1993; Dietterich 2000; McGovern and Barto 2001; Menache, Mannor, and Shimkin 2002; Stolle and Precup 2002). Options in particular have been shown to speed up convergence both empirically (Precup 2000) and theoretically (Mann and Mannor 2014). Constructing such temporal abstractions automatically from data has also been tackled extensively, and with some success (Konidaris et al. 2011; Mann, Mannor, and Precup 2015; Kulkarni et al. 2016; Mankowitz, Mann, and Mannor 2016; Bacon, Harb, and Precup 2017; Machado, Bellemare, and Bowling 2017). While some of the approaches require prior knowledge, have a fixed time horizon for partial policies (Vezhnevets et al.

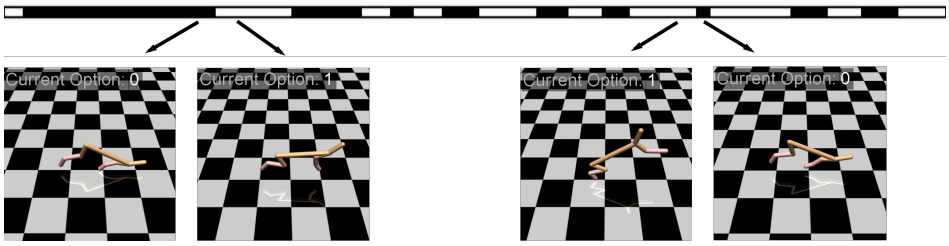


Figure 6: **Timeline of options used by IOC agent in HalfCheetah** where each option is represented by a distinct color (black & white). Two distinct options are learned during the task of moving forward; option 0 moves forward by dragging the limbs whereas option 1 takes larger hopped steps; see accompanying videos.

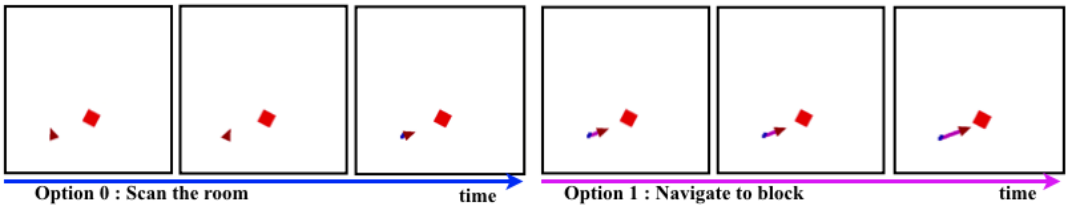


Figure 7: **3D Navigation in MiniWorld** with the top view of the environment and the agent as the red arrow. We show sequence of frames from a sampled trajectory. Option 0 scans the environment to locate the block. Option 1 learns to directly navigate to the block, once located. Please see accompanying videos.

2017), or use intrinsic rewards (Kulkarni et al. 2016), Bacon, Harb, and Precup (2017) provides an end-to-end differentiable approach without needing any sub-goals or intrinsic motivation. We generalize their policy gradient approach to learn interest functions. While we use rewards in our gradient-based algorithm, our qualitative analysis also indicates some clustering of states in which a given option starts, as in (Lakshminarayanan et al. 2016; Bacon 2013; Mannor et al. 2004). Our approach is closely related in motivation to Mankowitz, Mann, and Mannor (2016). However, our method does not make assumptions about a particular structure for the initiation and termination functions (except smoothness).

Initiation sets were an integral part of Sutton, Precup, and Singh (1999) and provide a way to control the complexity of the process of exploration and planning with options. This aspect of options has been ignored since, including in recent works (Bacon, Harb, and Precup 2017; Harb et al. 2017; Harutyunyan et al. 2019a; 2019b) because there was no elegant way to learn initiation sets. We address this *open problem* by generalizing initiation sets to differentiable interest functions. Since an interest function is a component of an option, it can be transferred once learned.

Discussion and Future Directions

We introduced the notion of interest functions for options, which generalize initiation sets, with the purpose of controlling search complexity. We presented a policy gradient-based method to learn options with interest functions, using general function approximation. Because the learned options are specialized, they are able to both learn faster in a single task and adapt to changes much more efficiently than options which initiate everywhere. Our qualitative results suggest that the interest function could be interpreted as an *attention mechanism* (see Appendix Fig. A4). To some extent, the interest functions learnt are able to override termination degeneracy as well (only one option being active all the time, or options switching often) although our approach was not meant to tackle that problem directly. Exploring further the interaction of initiation and termination functions, and imposing more coordination between the two, is an interesting topic for future work.

In our current experiments, the agent optimizes a single external reward function. However, the same algorithm could be used with intrinsic rewards as well.

We did not explore in this paper the impact of interest functions in the context of planning. However, given the intuitions from classical planning, learning models for options with interest functions could lead to better and faster planning, which should be explored in the future.

Finally, other ways of incorporating interest functions into the policy over options would be worth considering, in order to consider only choices over few options at a time.

Acknowledgments

The authors would like to thank NSERC & CIFAR for funding this research; Emmanuel Bengio, Kushal Arora for useful discussions throughout this project; Michael Littman, Zafarali Ahmed, Nishant Anand, and the anonymous reviewers for providing critical and constructive feedback.

References

Bacon, P.-L.; Harb, J.; and Precup, D. 2017. The option-critic architecture. In *AAAI*, 1726–1734.

Bacon, P.-L. 2013. On the bottleneck concept for options discovery.

Chevalier-Boisvert, M. 2018. gym-miniworld environment for openai gym. https://github.com/maximecb/gym-miniworld.

Dayan, P., and Hinton, G. E. 1993. Feudal reinforcement learning. In *Advances in neural information processing systems*, 271–278.

Dietterich, T. G. 2000. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of Artificial Intelligence Research* 13:227–303.

Drescher, G. L. 1991. *Made-up Minds: A Constructivist Approach to Artificial Intelligence*. Cambridge, MA, USA: MIT Press.

Fikes, R. E.; Hart, P. E.; and Nilsson, N. J. 1972. Learning and executing generalized robot plans. *Artificial Intelligence* 3:251–288.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1126–1135. JMLR. org.

Harb, J.; Bacon, P.-L.; Klissarov, M.; and Precup, D. 2017. When waiting is not an option: Learning options with a deliberation cost. *arXiv preprint arXiv:1709.04571*.

Harutyunyan, A.; Dabney, W.; Borsa, D.; Heess, N.; Munos, R.; and Precup, D. 2019a. The termination critic. *arXiv preprint arXiv:1902.09996*.

Harutyunyan, A.; Vrancx, P.; Hamel, P.; Nowe, A.; and Precup, D. 2019b. Per-decision option discounting. In *International Conference on Machine Learning*, 2644–2652.

Hauskrecht, M.; Meuleau, N.; Kaelbling, L. P.; Dean, T.; and Boutilier, C. 1998. Hierarchical solution of markov decision processes using macro-actions. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, 220–229. Morgan Kaufmann Publishers Inc.

Klissarov, M.; Bacon, P.; Harb, J.; and Precup, D. 2017. Learnings options end-to-end for continuous action tasks. *CoRR* abs/1712.00004.

Konidaris, G., and Barto, A. G. 2007. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, volume 7, 895–900.

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

Korf, R. E. 1983. *Learning to Solve Problems by Searching for Macro-operators*. Ph.D. Dissertation, Pittsburgh, PA, USA. AAI8425820.

Kulkarni, T. D.; Narasimhan, K.; Saeedi, A.; and Tenenbaum, J. 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, 3675–3683.

Lakshminarayanan, A. S.; Krishnamurthy, R.; Kumar, P.; and Ravindran, B. 2016. Option discovery in hierarchical reinforcement learning using spatio-temporal clustering. *arXiv preprint arXiv:1605.05359*.

Machado, M. C.; Bellemare, M. G.; and Bowling, M. 2017. A laplacian framework for option discovery in reinforcement learning. *arXiv preprint arXiv:1703.00956*.

Mankowitz, D. J.; Mann, T. A.; and Mannor, S. 2016. Adaptive skills adaptive partitions (asap). In *Advances in Neural Information Processing Systems*, 1588–1596.

Mann, T., and Mannor, S. 2014. Scaling up approximate value iteration with options: Better policies with fewer iterations. In *International Conference on Machine Learning*, 127–135.

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

Mannor, S.; Menache, I.; Hoze, A.; and Klein, U. 2004. Dynamic abstraction in reinforcement learning via clustering. In *Proceedings of the twenty-first international conference on Machine learning*, 71. ACM.

McGovern, A., and Barto, A. G. 2001. Automatic discovery of subgoals in reinforcement learning using diverse density. In *ICML*, volume 1, 361–368.

Menache, I.; Mannor, S.; and Shimkin, N. 2002. Q-cut-dynamic discovery of sub-goals in reinforcement learning. In *European Conference on Machine Learning*, 295–306. Springer.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529.

Parr, R., and Russell, S. J. 1998. Reinforcement learning with hierarchies of machines. In *Advances in neural information processing systems*, 1043–1049.

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

Puterman, M. L. 1995. Markov decision processes: Discrete stochastic dynamic programming. *Journal of the Operational Research Society* 46(6):792–792.

Ring, M. B. 1997. Child: A first step towards continual learning. *Machine Learning* 28(1):77–104.

Stolle, M., and Precup, D. 2002. Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, 212–223. Springer.

Sutton, R. S., and Barto, A. G. 1998. *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1st edition.

Sutton, R. S.; Mahmood, A. R.; and White, M. 2016. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research* 17:73:1–73:29.

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

Thrun, S., and Schwartz, A. 1995. Finding structure in reinforcement learning. In *Advances in neural information processing systems*, 385–392.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 5026–5033. IEEE.

Vezhnevets, A. S.; Osindero, S.; Schaul, T.; Heess, N.; Jaderberg, M.; Silver, D.; and Kavukcuoglu, K. 2017. Feudal networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1703.01161*.

White, M. 2017. Unifying task specification in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 3742–3750.