LEARNING DIVERSE OPTIONS VIA INFOMAX TERMINATION CRITIC

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of autonomously learning reusable temporally extended actions, or options, in reinforcement learning. While options can speed up transfer learning by serving as reusable building blocks, learning reusable options for unknown task distribution remains challenging. Motivated by the recent success of mutual information (MI) based skill learning, we hypothesize that more diverse options are more reusable. To this end, we propose a method for learning termination conditions of options by maximizing MI between options and corresponding state transitions. We derive a scalable approximation of this MI maximization via gradient ascent, yielding the InfoMax Termination Critic (IMTC) algorithm. Our experiments demonstrate that IMTC significantly improves the diversity of learned options without rewards, combined with an intrinsic option learning method. Moreover, we test the reusability of learned options by transferring options into various tasks, confirming that IMTC helps quick adaptation, especially in complex domains where an agent needs to manipulate objects.

1 INTRODUCTION

Behavior learning from environmental interaction is a fundamental problem in artificial intelligence and robotics. In recent years, combined with deep neural networks (NN), reinforcement learning (RL) has successfully learned complex behaviors from environmental interactions and human-designed tasks represented by reward functions (Heess et al., 2017; OpenAI et al., 2019). To reuse trained agents across multiple tasks, abstracting a course of action as a higher-level building block (Barto and Mahadevan, 2003; Barto et al., 2013) would be useful. The options framework (Sutton et al., 1999) is a representative formulation of action abstraction in RL, where each option consists of an intra-option policy and its termination function. Options have the potential of accelerating learning in new RL tasks (Brunskill and Li, 2014), serving as reusable behavioral blocks. For example, if a wheeled robot already had learned to lean and turn left and right, it can learn to drive on unfamiliar roads faster than learning from scratch. However, discovering reusable options remains challenging due to the difficulty of define and measure reusability a priori. Although we can measure the reusability of options when we know the task distribution is known (e.g., in a Bayesian way (Solway et al., 2014)), it is difficult to measure it without prior knowledge. Hence, in such cases, we need a reasonable heuristic assumption about future tasks.

In this paper, we hypothesize that diversifying options improves the reusability of options, assuming that tasks are uniformly distributed over the state space. We argue that this is a reasonable heuristic because diverse options are expected to work well for any task to some degree. The next problem is in measuring the diversity of options. The most intuitive strategy would be visiting all states as equally as possible. However, in environments with continuous or large state spaces, targetting all states could be computationally impossible or does not make much sense. As an alternative, in the context of autonomous skill acquisition that considers learning a set of policies without conditions, mutual information (MI) maximization has been widely applied for learning diverse policies (Eysenbach et al., 2019; Baumli et al., 2021; Choi et al., 2021). MI maximization is advantageous in that it has tractable approximations with probabilistic models learned by NN, enabling it to scale up to more complex domains.

To this end, we propose maximizing MI between options and their terminating states conditioned by a starting state for learning diverse options. Maximizing this MI diversifies the destinations of an agent

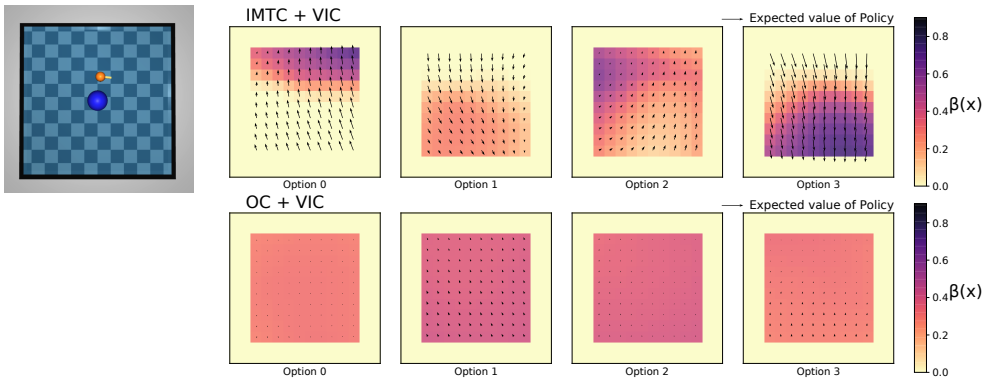


Figure 1: **Left:** `PointBilliard` environment. **Right:** Learned options by IMTC (upper row) and OC (lower row). Arrows show policies, and heatmaps show termination conditions of options.

when it chooses different options at a certain state while keeping the transition as deterministic as possible, leading to diverse and meaningful options. The same objective is employed by Gregor et al. (2016) for learning sub-policies, but we aim to train termination conditions of options as well. We derive an unbiased estimator of the gradient of this MI with respect to termination conditions using the termination gradient theorem (Harutyunyan et al., 2019). As a key contribution, we reformulate the estimated gradient using Bayes' rule and derive a tractable approximation of this MI maximization with a leaned classification model, yielding the InfoMax Termination Critic (IMTC) algorithm. We implement IMTC on the Option Critic Architecture (OC, Bacon et al. (2017)) and Proximal Policy Optimization (PPO, Schulman et al. (2017)) with a newly proposed advantage estimation for option learning. In experiments, we qualitatively demonstrate that IMTC successfully learns diverse and meaningful options in reward-free RL (Jin et al., 2020) setting, combined with VIC (Gregor et al., 2016). Figure 1 shows the learned options by IMTC and OC in `PointBilliard` domain. We can see that options learned by IMTC are clearly separated and different. Moreover, we test the reusability of learned options in task adaptation experiments: we first pre-train options without rewards and then transfer them on various tasks. We show that IMTC helps quick adaptation to specific tasks, especially in complex domains where an agent needs to manipulate objects.

2 Background and Notation

We assume the standard RL setting in the Markov decision process (MDP), following Sutton and Barto (2018). MDP $\mathcal{M}$ consists of a tuple $(\mathcal{X}, \mathcal{A}, p, r)$, where $\mathcal{X}$ is the finite set of states, $\mathcal{A}$ is the finite set of actions, $p : \mathcal{X} \times \mathcal{A} \times X \to [0, 1]$ is the state transition function, $r : \mathcal{X} \times \mathcal{A} \to [r_{\text{min}}, r_{\text{max}}]$ is the reward function, where $r_{\text{min}}, r_{\text{max}}$ are minimal and maximal rewards. We use X_t and A_t for denoting random variables of the state and action experienced at time t. We define $G \stackrel{\text{def}}{=} \sum_{t=0}^{\infty} \gamma^t R_t$ as discounted cumulative return, where $R_t = r(X_t, A_t)$ is the reward received at time t and $0 < \gamma < 1$ is the discount factor. We consider maximizing expected discounted return $\mathbb{E}[G|\pi]$. As an agent, we consider a memoryless policy $\pi : \mathcal{X} \times \mathcal{A} \to [0, 1]$. π induces two value functions: action-value function $Q^\pi(x, a) \stackrel{\text{def}}{=} \mathbb{E}_\pi[G|X_0 = x, A_0 = a, \pi]$ and state-value function $V^\pi(x) \stackrel{\text{def}}{=} \sum_a \pi(a|x) Q^\pi(x, a)$. We use d^π for denoting an agent's undiscounted occupancy measure $d^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \mathbb{I}_{X_t = x}]$, where $\mathbb{I}$ is the indicator function.

Assuming that π is differentiable by the policy parameters θ_π, the policy gradient (PG) method (Williams, 1992) maximizes G^π by updating θ_π via gradient ascent. A common formulation of PG estimates the gradient by $\nabla_{\theta_\pi} G^\pi = \mathbb{E}_{x,a,\pi}\left[\nabla_{\theta_\pi} \log \pi(a|x) \hat{A}(x, a)\right]$, where $\hat{A}(x, a)$ is an estimation of the advantage function $A^\pi(x, a) \stackrel{\text{def}}{=} Q^\pi(x, a) - V^\pi(x)$. Among many variantes of PG methods, we implemented our method on PPO (Schulman et al., 2017). At each gradient step, PPO updates θ_π to maximize $\text{clip}(\frac{\pi(a|x)}{\pi_{\text{old}}(a|x)} \hat{A}, -\epsilon, \epsilon)$, where $\text{clip}(x, -\epsilon, \epsilon) = \max(-\epsilon, \min(\epsilon, x))$ and π_{old} is the policy before being updated. This clipping heuristics prevents π from updating too rapidly.

Options Framework Options (Sutton et al., 1999) provide a framework for formulating temporally abstracted actions in RL. An option $o \in \mathcal{O}$ consists of a tuple $(\mathcal{I}^o, \beta^o, \pi^o)$, where $\mathcal{I}^o \subseteq \mathcal{X}$ is the initiation $\beta^o : \mathcal{X} \to [0, 1]$ is a termination function, and π^o is an *intra-option* policy. Following related studies (Bacon et al., 2017; Harutyunyan et al., 2019), we assume that $\mathcal{I}^o = \mathcal{X}$ focuses on learning β^o and π^o. We let $\mu : \mathcal{X} \times \mathcal{O} \to [0, 1]$ denote a policy over options. A typical RL agent sample A_t from $\pi(\cdot|X_t)$. Analogously, at time t, an RL agent with options first samples a termination T_t from $\beta^{O_t}(X_t)$. If $T_t = 1$, O_{t+1} is sampled from $\mu(\cdot|X_t)$ and if not, the current option remains the same. The next action A_t is sampled from $\pi^{O_{t+1}}(\cdot|X_t)$. For option learning methods, we use π to denote the resulting policy induced by μ, $\cup_{o\in\mathcal{O}}\pi^o$, and $\cup_{o\in\mathcal{O}}\beta^o$.

Option value functions With options, we have three option-value functions $Q_\mathcal{O}$, $V_\mathcal{O}$, and $U_\mathcal{O}$. $Q_\mathcal{O}$ is the option-value function denoting the value of selecting an option o at state x defined by $Q_\mathcal{O}(x, o) \stackrel{\text{def}}{=} \mathbb{E}[G|X_0 = x, O_0 = o]$. Similar to the relationship between Q^π and V^π, we let $V_\mathcal{O}$ denote the marginalized option-value function $V_\mathcal{O}(x) \stackrel{\text{def}}{=} \sum_o \mu(o|x)Q_\mathcal{O}(x, o)$. $U_\mathcal{O}(x, o) \stackrel{\text{def}}{=} (1 - \beta^o(x))Q_\mathcal{O}(x, o) + \beta^o(x)V_\mathcal{O}(x)$ is called the option-value function *upon arrival* (Sutton et al., 1999) and denotes the value of reaching a state x with o and not having selected the new option. We use these notations in Section 4.2.

Termination gradient theorem Analogously to p, we let $P^o : \mathcal{X} \times \mathcal{O} \times X \to [0, 1]$ denote the state transition probability induced by options. When an agent is at x_s and having an option o, the probability that o ends at x_f is given by:

$$P^o(x_f|x_s) = \beta^o(x_f)\mathbb{I}_{x_f=x_s} + (1 - \beta^o(x_s)) \sum_x p^{\pi^o}(x|x_s)P^o(x_f|x), \tag{1}$$

where p^{π^o} is the *policy-induced* transition function $p^{\pi^o}(x'|x) \stackrel{\text{def}}{=} \sum_{a\in\mathcal{A}} \pi^o(a|x)P(x'|x, a)$. Here we assume that all options eventually terminate, such that P^o is a valid probability distribution over x_f. Interestingly, P^o is differentiable with respect to the parameter of β^o. Harutyunyan et al. (2019) introduced the termination gradient theorem:

Theorem 1. *Let β^o be parameterized by θ_β, and let ℓ_{β^o} denote the logit of β^o, i.e., $\ell_{\beta^o} = \log(\frac{\beta^o(x)}{1-\beta^o(x)})$. We have*

$$\nabla_{\theta_\beta} P^o(x_f|x_s) = \sum_x P^o(x|x_s)\nabla_{\theta_\beta}\ell_{\beta^o}(x)(\mathbb{I}_{x_f=x} - P^o(x_f|x)). \tag{2}$$

We use this theorem to derive an unbiased estimator of our target gradient.

3 InfoMax Termination Critic

We now present the InfoMax Termination Critic (IMTC) algorithm. To learn diverse options, we propose to maximizes the following MI at each state x_s:

$$I(X_f; O|x_s) = H(X_f|x_s) - H(X_f|x_s, O) = H(O|x_s) - H(O|X_f, x_s), \tag{3}$$

where I denotes the MI $I(A; B|c) = H(A|c) - H(A|B, c)$, H denotes entropy, and O is a random variable denoting options experienced by an agent. Precisely, we let η denote the probability of having an option o when leaving a state x: $\eta(o|x) = \beta^o(x)\mu(o|x) + (1 - \beta^o(x)) \sum_{x'\in\mathcal{X}} p^{\pi^o}(x|x')\eta(o|x')$ and let η^π denote η marginalized over d_π: $\eta^\pi(o) = \sum_{x\in\mathcal{X}} d_\pi(x)\eta(o|x)$. Then, we define O as a random variable with η^π. By decomposing this MI as $I(X_f; O|x_s) = H(X_f|x_s) - H(X_f|x_s, O)$, we can interprete this MI maximization as maximization of $H(X_f|x_s)$ and minimization of $H(X_f|x_s, O)$. Maximizing $H(X_f|x_s)$ diversifies possible destinations of an agent. Thus, we expect the resulting options to be diverse because choosing a different option is likely to lead to a different destination. On the other hand, minimizing $H(X_f|x_s, O)$ makes the option state transition more deterministic, making the options more meaningful.

To build an intuition, we searched options that maximizes this MI by a simple blute-force method from a limited set of π^o and β^o in a 3×3 gridworld environment. Specifically, we considered only deterministic intra-option polciies and 0.1 or 0.8 as the value of $\beta^o(x)$. We set the number of options as 3. In this environment, an agent can take four actions: `Up`, `Down`, `Left`, and `Right`.

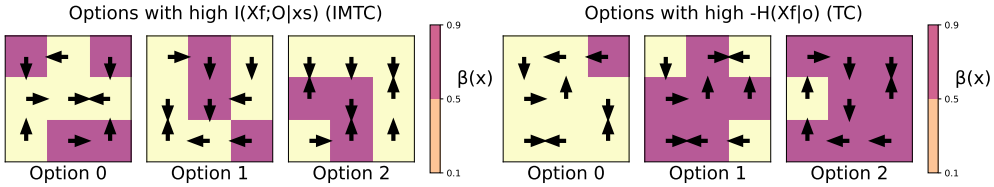


Figure 2: IMTC options (left) and TC options (right).

With probability 0.1 the taken action fails and it takes an uniformly random action. Figure 2 shows the result. Arrows represent intra-option policies and colors of the room represent termination probabilities. Here, we can see the tendency that IMTC prefers options that have almost separated termination regions and different policies per option. We also visualized the options that maximize $-H(X_f|o)$ (TC options) proposed by Harutyunyan et al. (2019). For TC options, we can see a similar tendency as IMTC options, but option 0 and 1 of TC share the intra-option policies on the left side of the room. Thanks to the diversity term $H(X_f|x_s)$, IMTC successfully avoids such a policy overlap. In addition, we can see that IMTC has a smaller termination regions than TC, which can enhance long-range abstraction.

We propose to maximize the MI (3) by updating β^o via gradient ascent w.r.t. θ_β. For this purpose, we now derive an unbiased estimator of the gradient. First, we write the gradient of the MI using the option transition model P^o and marginalized option-transition model $P(x_f|x_s) = \sum_o \eta^\pi(o)P^o(x_f|x_s)$.

Proposition 1. *Let β^o be parameterized using a sigmoid function. Given a trajectory $\tau = x_s, \dots, x, \dots, x_f$ sampled by π^o and β^o, we can obtain unbiased estimations of $\nabla_{\theta_\beta} H(X_f|x_s)$ and $\nabla_{\theta_\beta} H(X_f|x_s, O)$ by*

$$\nabla_{\theta_\beta} H(X_f|x_s) = \mathbb{E}_{d^\pi,\eta}\left[-\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\Big(\log P(x|x_s) - \log P(x_f|x_s)\Big)\right] \tag{4}$$

$$\nabla_{\theta_\beta} H(X_f|x_s, O) = \mathbb{E}_{d^\pi,\eta}\left[-\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\Big(\log P^o(x|x_s) - \log P^o(x_f|x_s)\Big)\right], \tag{5}$$

where $\ell_{\beta^o}(x)$ denotes the logit of $\beta^o(x)$.

Note that the additional term β^o is necessary because x is not actually a terminating state. The proof is based on Section 4 in Harutyunyan et al. (2019) and is given in Appendix A.1.

Now the targetting gradient can be written as:

$$\begin{aligned}\nabla_{\theta_\beta} I(X_f; O|x_s) &= \nabla_{\theta_\beta} H(X_f|x_s) - \nabla_{\theta_\beta} H(X_f|x_s, O)\\ &= \mathbb{E}_{d^\pi,\eta}\left[-\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\left(\log P(x|x_s) - \log P(x_f|x_s) - (\log P^o(x|x_s) - \log P^o(x_f|x_s))\right)\right],\end{aligned} \tag{6}$$

which means that we can approximate MI maximization by estimating P^o and P. However, in RL, learning a density model over the state space in RL is often difficult, especially with large or continuous state spaces. For example, it has been tackled with data compression methods (Bellemare et al., 2016) and generative models (Ostrovski et al., 2017). Hence, we reformulate the gradient using Bayes' rule to avoid estimating P^o and P. The resulting term consists of the inverse option transition function $p_{\mathcal{O}}(o|x_s, x_f)$, which denotes the probability of having an o given a state transition x_s, x_f.

Proposition 2. *We now have*

$$\begin{aligned}\nabla_{\theta_\beta} I(X_f; O|x_s) &= \nabla_{\theta_\beta} H(X_f|x_s) - \nabla_{\theta_\beta} H(X_f|x_s, O)\\ &= \mathbb{E}_{d^\pi,\eta}\left[\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\Big(\log p_{\mathcal{O}}(o|x_s, x) - \log p_{\mathcal{O}}(o|x_s, x_f)\Big)\right].\end{aligned} \tag{7}$$

The proof is provided in Appendix A.2. Equation (7) requires estimating of $p_{\mathcal{O}}$ per updating π^o and β^o, which is computational quite expensive. Thus, we approximate approximate the gradient (7) by regressing a classification model over options $\hat{p_{\mathcal{O}}}(o|x_s, x_f)$ on sampled option transition

4 Implementation

Since IMTC can be combined with any on-policy RL methods, we choose PPO (Schulman et al., 2017) as a base algorithm because of its stability and ease of implementation. As notable implementation

details, this section explains the estimation of $p_{\mathcal{O}}$, advantage estimation, and our VIC implementation. We provide further details and the full description of the algorithm in Appendix B.

4.1 Estimating $p_{\mathcal{O}}$

To estimate $p_{\mathcal{O}}$, we employ a classification model over options $\hat{p_{\mathcal{O}}}(o|x_s, x_f)$ and regress it on sampled option transitions, as per Gregor et al. (2016). However, our preliminary experiments observed that this online regression could be unstable because the supply of transition data depends on the termination probability and can drastically increase or decrease during training. To address this problem, we maintain a replay buffer $B_{\mathcal{O}}$, which stores option state transitions $\{(o, x_s, x_f)\}$ to stabilize the regression of $\hat{p_{\mathcal{O}}}$. Note that using older option state transitions can introduce bias to $\hat{p_{\mathcal{O}}}$ because it depends on the current policy. However, we found that this is not harmful when the capacity of the replay buffer is reasonably small.

4.2 Advantage Estimation

The original PPO implementation employs GAE (Schulman et al., 2015b) for estimating the advantage, which is important for learning performance (Andrychowicz et al., 2021). Therefore, we employed two variants of GAE in experiments for option learning according to the experimental setup. In the following paragraphs, we let N denote the rollout length used for advantage estimation and let $t + k$ denote the time step at which the current option o_t terminates. Thus, we need to consider the effect of option-switching in advantage estimation When $k < N$. Furthermore, we use two variants of the option-specific TD errors $\delta(o_t) = R_t + \gamma Q_{\mathcal{O}}(x_{t+1}, o_t) - Q_{\mathcal{O}}(x_t, o_t)$ and $\delta_{\mathrm{U}}(o_t) = R_t + \gamma U_{\mathcal{O}}(x_{t+1}, o_t) - Q_{\mathcal{O}}(x_t, o_t)$.

Independent GAE for Reward-Free RL For no-reward experiments with VIC, we used the following variant of GAE:

$$\hat{A}^o_{\text{ind}} = -Q_{\mathcal{O}}(x_t, o_t) + \sum_{i=0}^{\min(k,N)} (\gamma\lambda)^i \delta(o_{t+i}) \tag{8}$$

Here, we ignore the future rewards produced by other options after the current option o_t terminates. This formulation enhances learning diverse intra-option policies per option.

Upgoing GAE for learning from rewards For single-task learning, increasing the rollout step N often helps speeding up learning (Sutton and Barto, 2018). However, future rewards after option termination heavily depend on the selected option and have high variance, especially when learning diverse options. This high variance of future rewards slows advantage learning and causes underestimation of $\hat{A}^o$. Thus, to prevent underestimation, we introduce an *upgoing* GAE (UGAE) for estimating advantage with options:

$$\hat{A}^o_{\text{upg}} = -Q_{\mathcal{O}}(x_t, o_t) + \begin{cases} \sum_{i=0}^{k} (\gamma\lambda)^i \delta^o_{t+i} + \underbrace{\max\left(\sum_{i=k+1}^{N} (\gamma\lambda)^i \delta(o_{t+i}), 0\right)}_{\text{upgoing estimation}} & (k < N) \\ \sum_{i=0}^{N-1} (\gamma\lambda)^i \delta^o_{t+i} + (\gamma\lambda)^N \delta_{\mathrm{U}}(o_{t+N}) & (\text{otherwise}). \end{cases} \tag{9}$$

Like the upgoing policy update (Vinyals et al., 2019), the idea is optimistic regarding future rewards after option termination by taking the maximum over 0. We use $\hat{A}^o_{\text{upg}}$ for task adaptation experiments in Section 5.2.

4.3 VIC Implementation

In experiments, we show that IMTC helps an agent learn diverse options without reward signals. For this purpose, we employ VIC (Gregor et al., 2016) as a method for providing intrinsic rewards. Here we explain our VIC implementation. VIC updates intra-option policies to maximize the lower bound of MI (3) $H(O|x_s) - H(O|x_s, X_f) \geq H(O|x_s) + \mathbb{E}_{o,x_f}[\log q(o|x_s, x_f)]$ as rewards, where q can be any distribution and called an *option inference model*. We use $\hat{p_{\mathcal{O}}}$ as q. We also learn $\hat{\eta}(o|xs)$ from sampled option transitions and approximate $H(O|x_s)$ by $H(O|x_s) = -\sum_{o\in\mathcal{O}} \eta(o|x_s) \log \eta(o|x_s) = -\mathbb{E}[\log \eta(o|x_s)] \approx \log \hat{\eta}(o|x_s)$. To this end, we giving an agent

$R_{\text{VIC}} = c_{\text{VIC}} \left(\log \hat{p_{\mathcal{O}}}(o|x_s, x_f) - \log \hat{\eta}(o|x_s)\right)$ as an immediate reward when an option o terminates, where c_{VIC} is a scaling coefficient. This is different from the original VIC implementation where $H(O|x_s)$ is treated as a constant. However, we empirically found this formulation helps diversity options in our preliminary experiments.

5 EXPERIMENTS

As presented in this section, we conducted two series of experiments to analyze the property of IMTC. First, we qualitatively evaluated the diversity of options learned by IMTC in a reward-free setting. We use VIC (Gregor et al., 2016) for intrinsic rewards in these experiments. Second, we quantitatively test the reusability of learned options by task adaptation learned options on a specific task. Through experiments, we compared our method with OC (Bacon et al., 2017) and vanilla VIC with fixed β^o. We used $\forall_x \beta^o(x) = 0.1$ for VIC since it performed the best in task adaptation tasks, while 0.05 was used in Gregor et al. (2016). In order to check only the effect of the different methods for learning beta β^o, the rest of the implementation is the same for these three methods. That is, OC and vanilla VIC also use PPO and advantage estimation methods in Section 4.2 for learning P^o. We used four options for all option-learning methods, and the effect of the number of options is further investigated in Appendix C. We did not compare our method (IMTC) with TC (Harutyunyan et al., 2019) because our TC implementation failed to learn options with relatively small termination regions as reported in the paper, and there is no official public code for TC. All environments that we used for experiments are implemented on the MuJoCo (Todorov et al., 2012) physics simulator. We further describe the detail in Appendix C.

5.1 INTRINSIC OPTION LEARNING WITH VIC

We now compare the options learned by IMTC with options of other methods. Generally, learned options depend on the reward structure in the environment. Thus, to make sure that rewards are not designed intentionally to learn good options, we employ a reward-free RL setting where no reward is given to agents. Instead, for all methods, we used VIC (Gregor et al., 2016) as intrinsic rewards for training each intra-option policy π^o. We noted our implementation of VIC in Section 4.3. We fix μ as $\mu(o|x) = \frac{1}{|\mathcal{O}|}$ in this experiment. Intra-option policies are trained by PPO (Schulman et al., 2017) and independent GAE (8). We show network architectures and hyperparameters in appendix C. We set the episode length to 1×10^4, so an agent is reset to its starting position after 1×10^4 steps. For all visualizations, we chose the best one from five independent runs with different random seeds.

We visualized learned options in `PointReach` environment shown in Figure 4a. In this environment, an agent controls the ball initially placed at the center of the room. The state space consists of positions $(\mathbf{x}, \mathbf{y})$ and velocities $(\Delta\mathbf{x}, \Delta\mathbf{y})$ of an agent, and the action space consists of acceralations $(\frac{\Delta\mathbf{x}}{\Delta\mathbf{t}}, \frac{\Delta\mathbf{y}}{\Delta\mathbf{t}})$. Figure 3 shows the options learned in this environment after 4×10^6 steps. Each arrow represents the mean value of intra-option policies, and the heatmaps represent β^o. In this experiment, we observed the effect of IMTC clearly, for both termination regions and intra-option policies. Termination regions are almost separated over options: option 0 terminates at the lower, option 1 at the upper-center, option 2 at the left, and option 3 at the upper right. Only termination regions of option 0 and 2 overlap each other. Contrary, OC failed to learn meaningful termination regions: option 0 and 1 never terminate, and option 3 terminates almost everywhere. Hence, we can say that IMTC certainly helps to learn diverse termination regions. We also observed that the intra-option policies of IMTC are the most diverse. While intra-option policies of OC and vanilla VIC point out roughly two directions (lower left and upper right), all intra-option policies of IMTC point in different directions. Moreover, the magnitude of intra-option policies (shown by the length of arrows) is larger with IMTC. This result suggests that IMTC diversifies and clarifies intra-option policies by diversifying termination regions. Figure 1 also show options learned by IMTC and OC in `PointBilliard` domain, where we can see the same tendency.

5.2 TRANSFERRING SKILLS VIA TASK ADAPTATION

Now we quantitatively test the reusability of learned options by task adaptation them with specific reward functions. Specifically, we first trained agents with VIC rewards without external rewards as per the previous section. Then we transferred agents to an environment with the same state and action space but with some external reward functions. We prepared multiple reward functions, which we call tasks, for each domain and evaluated the averaged performance over tasks. We compare

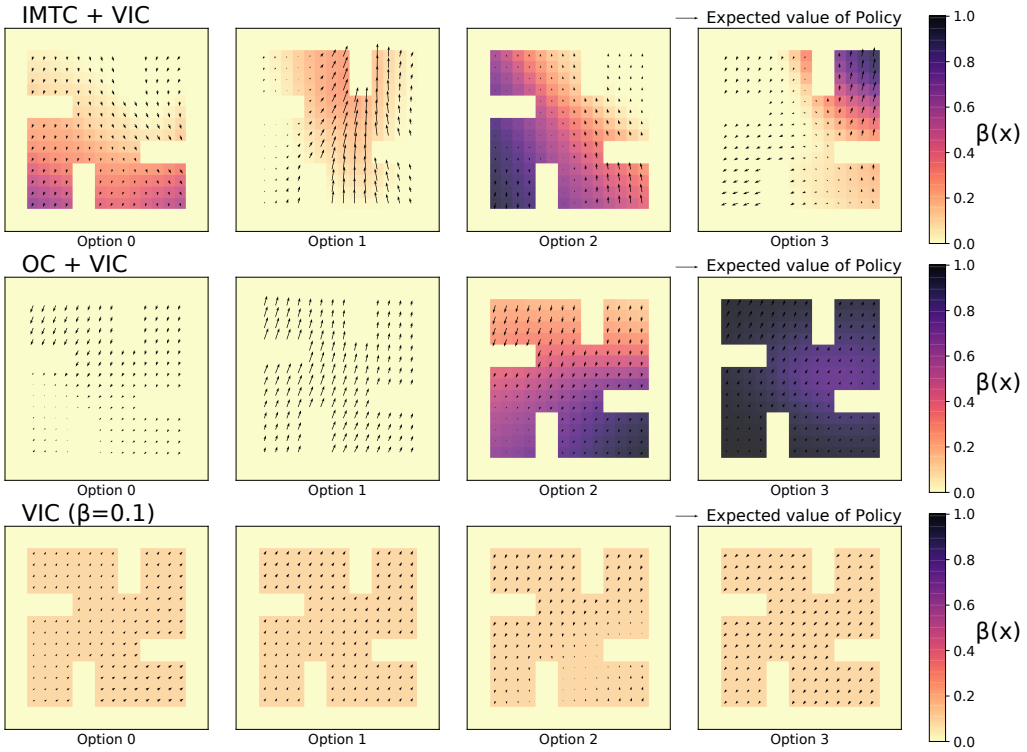
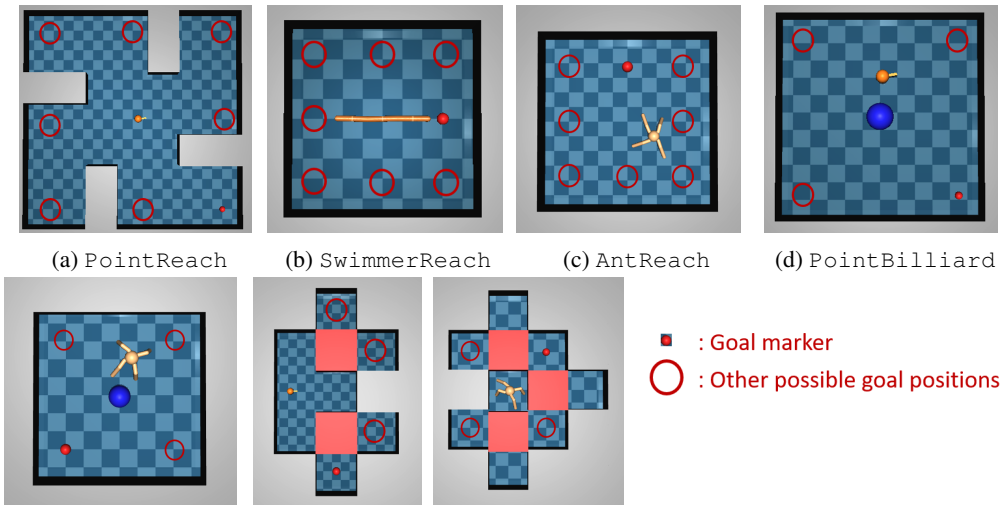


Figure 3: Learned intra-option policies (π^o) and termination probabilities (β^o) for each option in PointMaze after training 4×10^6 steps. Arrows show the expected value of action, and heatmaps show probabilities of each β^o. **First row:** Options learned by IMTC and VIC rewards. Intra-option policies have various directions, e.g., option 0 points from lower left to lower right and option 1 points from upper right to left. Also, termination regions are clearly separated across options. For example, option 2 terminates at the lower left, and contrary, option 3 terminates at the upper right. **Second row:** Options learned by OC and VIC rewards. Intra-option policies have two directions. Option 0 and 2 point to the lower left, and option 1 and 3 point to the upper right. Option 3's termination region is close to 1.0 at all states, and thus, this option is rarely used. Contrary, termination probabilities of option 0 and 1 are close to 0 at all states. Thus these options never terminate. **Third row:** Options learned by vanilla VIC with fixed β^o. Similar to OC, Intra-option policies have basically two directions. Option 0 and 1 point to the lower left, and option 2 and 3 point to the upper right.

IMTC with OC, vanilla VIC, and PPO without pre-training. For a fair comparison, all of IMTC, OC, and vanilla use UGAE (9) and PPO-style policy updates for training intra-option policies. We found UGAE is very effective for task adaptation and show the result in Appendix C.5. For vanilla VIC, termination probability is fixed to 0.1 through pre-training and task adaptation. ϵ-greedy based on $Q_{\mathcal{O}}$ with $\epsilon = 0.1$ is used as the option selection policy μ. We hypothesize that diverse options learned by IMTC can help quickly adapt to given tasks, supposing the diversity of tasks.

Figure 4 shows all domains used for task adaptation experiments. For simplicity, all tasks have goal-based sparse reward functions. I.e., an agent receives $R_t = 1.0$ when it satisfies a goal condition, and otherwise the control cost -0.0001 is given. Red circles show possible goal locations for each task. When the agent fails to reach the goal after 1×10^3 steps, it is reset to a starting position. `PointReach`, `SwimmerReach`, and `AntReach` are simple navigation tasks where an agent aim to just navigate itself to the goal. We also prepared tasks with some complex tool use: in `PointBilliard` and `AntBilliard` an agent aims to kick the blue ball to the goal position, and in `PointPush` and `AntPush`, it has to push the block out of the way to the goal. We pre-trained IMTC, OC, and vanilla VIC agents for 4×10^6 environmental steps and additionally trained them for 1×10^6 steps for each task. Figure 5 shows learning curves and scatter plots drawn from five independent runs with different random seeds per domain. [1] We observed that IMTC performed the best or was compatible with baselines except for `SwimmerReach`, where vanilla VIC and

[1] We used seaborn (Waskom, 2021)'s `lmplot` with `order=3` method to draw plots.



(a) PointReach (b) SwimmerReach (c) AntReach (d) PointBilliard

(e) AntBilliard (f) PointPush (g) AntPush

Figure 4: Domains used in task adaptation experiments.

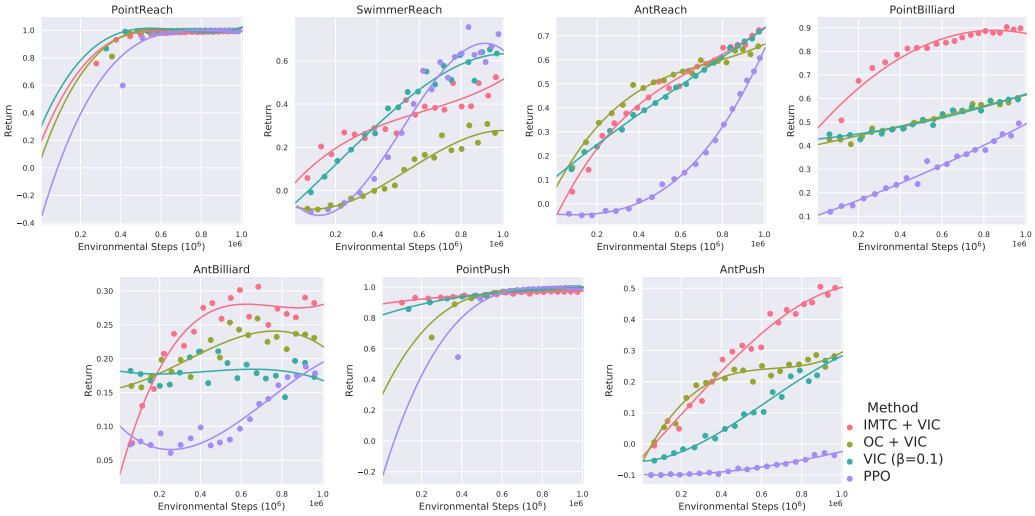


Figure 5: Learning curves for transfer learning experiments.

PPO finally outperformed IMTC. Notably, IMTC outperforms other methods with a large margin in complex PointBilliard, AntBilliard, and AntPush. This result suggests options learned by IMTC speed up adaptation to specific tasks, especially in complex domains where an agent needs to manipulate objects.

6 Related Work

Discovering diverse options Since Sutton et al. (1999) first formulated the options, discovering good options has been challenging. A classic concept for the goodness of options is a bottleneck region (McGovern and Barto, 2001), which refers to important states for reaching diverse areas, such as a passage between two rooms. Once an important region is discovered, we can construct an option that guides an agent to that area. Various approaches have been proposed to define bottleneck regions concisely and compute bottleneck options efficiently, including betweenness (Simsek and Barto, 2008), Eigen options (Machado et al., 2017), and covering options (Jinnai et al., 2019). Barto et al. (2013) discussed the importance of diverse options for exploration, referring to an option construction method with a graph-based decomposition of an MDP Vigorito and Barto (2010). We share the same motivation with these methods but have a different approach. While these methods construct *point options* that bridge two states, we capture a set of states by learning β^o directly, making it easy to

scale up with function approximation. To our knowledge, (Jinnai et al., 2020) only succeeded in scaling up point options to continuous options using approximated computation of the Laplacian.

End-to-end learning of options As described in the previous paragraph, many studies have attempted to construct options and then train intra-option policies. However, motivated by the recent success of RL with deep NN, Bacon et al. (2017) proposed OC to train intra-option policies and termination functions in parallel using a neural network as a function approximator. OC updates β^o by gradient ascent, so that maximize $Q_{\mathcal{O}}$. Thus, the resulting terminating regions heavily depend on the reward function. OC also proposed a PG-style method for learning intra-option policies. We used a similar method to OC for learning policies and values, but we employed a different method that does not depend on rewards for learning termination functions. Also, we proposed UGAE (9) for enhancing OC-style policy learning. While OC maximizes the option-value function directly, many heuristic objectives have been proposed with similar architectures, including deliberation cost (Harb et al., 2018), interest (Khetarpal et al., 2020), and safety (Jain et al., 2018). Notably, (Harutyunyan et al., 2019) proposed the termination critic (TC) that maximizes an information-theoretic objective referred to as *predictability* $-H(X_f|o)$. Our method is inspired by TC and maximizes a similar information-theoretic objective for diversity rather than predictability. In addition, TC requires estimating $P^o(x_f|x_s)$ and a marginal distribution of P^o, which can be quite difficult in environments with large or continuous state spaces. While Harutyunyan et al. (2019) proposed to use approximated dynamic programming for estimating P^o, it is even more difficult than estimating Q^π or V^π due to the lack of discounting and conditioning by x_s. We avoid such difficult approximations using Bayes' rule, making IMTC more scalable.

Mutual Information, Empowerment, and Skill Acquisition MI also appears in the literature regarding *intrinsically motivated* RL (Singh et al., 2004), as a driver of goal-directed behavior. A well-known example is the *empowerment* (Klyubin et al., 2005; Salge et al., 2014), obtained by maximizing MI between sequential k actions and the resulting state $I(a_t, ..., a_{t+k}; x_{t+k}|x_t) = H(x_{t+k}|x_t) - H(x_{t+k}|a_t, ..., a_{t+k}, x_t)$. Empowerment represents both large degree of freedom and good preparedness: i.e., larger $H(x_{t+k}|x_t)$ implies that there can be more diverse future states, while smaller $H(x_{t+k}|a_t, ..., a_{t+k}, x_t)$ indicates that an agent can realize its intention with greater certainty. In RL literature, empowerment is often implemented by maximizing the variational lower bounds as intrinsic rewards (Mohamed and Rezende, 2015; Zhao et al., 2020). We can interpret our objective $I(X_f; O|x_s)$ as a variant of empowerment where a fixed number of options represents action sequences. Gregor et al. (2016) also employed this interpretation and introduced VIC for training intra-option policies with fixed termination probabilities in a no reward RL setting. Our experiments observed that IMTC helps VIC to learn meaningful intra-option policies. As a variant of VIC, Baumli et al. (2021) proposed to use the reversed MI $I(X_s; O|x_f)$. MI has been used for discovering diverse skills without termination functions. Eysenbach et al. (2019) proposed maximizing MI between skills and states $I(O; X)$, and Sharma et al. (2020) extended the objective to a conditional MI $I(O; X'|x)$. These methods also maximize variational lower bounds of MI as rewards.

7 Conclusion

In this paper, we considered the problem of learning diverse options in RL. To learn diverse termination regions of options in a scalable way, we proposed to maximize MI between options and terminating states per starting state. We derived an unbiased gradient estimator to approximately maximize this MI, yielding the InfoMax Termination Critic (IMTC) algorithm. Also, we proposed a practical implementation of IMTC with enhanced advantage estimation section 4.2. In reward-free experiments, we visualized that IMTC helped learn diverse and clear options combined with VIC. We also showed that options learned by IMTC help an agent to quickly adapt to a specific reward function by transferring learned options.

Although our experiments observed that IMTC can learn clear and meaningful options, a problem is that learning of a classification model $\hat{p_{\mathcal{O}}}$ heavily depends on exploration. For example, an agent cannot explore a room well, it would not be able to learn sufficiently diverse options. Investigating the relationship between diversity and exploration would be an interesting approach. Also, our analysis in Figure 2 suggests that IMTC options can fall into small loops, forming uninteresting options. To prevent this, considering distances between states, e.g., by using bisimulation metric (Castro and Precup, 2010), is a plausible research direction. Another limitation of our method is that it requires

on-policy data for training, making it difficult to use with off-policy RL methods. Thus, updating termination regions in an off-policy way is an interesting future challenge.

REPRODUCIBILITY STATEMENT

We publish anonymized source code used for all our experiments on https://anonymous.4open.science/r/imtc-anonymized-code-E5D1/.

REFERENCES

M. Andrychowicz, A. Raichuk, P. Stanczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem. What matters for on-policy deep actor-critic methods? a large-scale study. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=nIAxjsniDzg&.

P. Bacon, J. Harb, and D. Precup. The option-critic architecture. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1726–1734, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14858.

A. G. Barto and S. Mahadevan. Recent advances in hierarchical reinforcement learning. *Discret. Event Dyn. Syst.*, 13(1-2):41–77, 2003. doi: 10.1023/A:1022140919877. URL https://doi.org/10.1023/A:1022140919877.

A. G. Barto, G. D. Konidaris, and C. M. Vigorito. Behavioral hierarchy: Exploration and representation. In G. Baldassarre and M. Mirolli, editors, *Computational and Robotic Models of the Hierarchical Organization of Behavior*, pages 13–46. Springer, 2013. doi: 10.1007/978-3-642-39875-9_2. URL https://doi.org/10.1007/978-3-642-39875-9_2.

K. Baumli, D. Warde-Farley, S. Hansen, and p Volodymyr Mnih. Relative variational intrinsic control. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, February 2-9, 2021*, 2021.

M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1471–1479, 2016. URL http://papers.nips.cc/paper/6383-unifying-count-based-exploration-and-intrinsic-motivation.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

E. Brunskill and L. Li. Pac-inspired option discovery in lifelong reinforcement learning. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 316–324. JMLR.org, 2014. URL http://proceedings.mlr.press/v32/brunskill14.html.

P. S. Castro and D. Precup. Using bisimulation for policy transfer in mdps. In M. Fox and D. Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press, 2010. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1907.

J. Choi, A. Sharma, H. Lee, S. Levine, and S. S. Gu. Variational empowerment as representation learning for goal-conditioned reinforcement learning. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 1953–1963. PMLR, 2021. URL http://proceedings.mlr.press/v139/choi21b.html.

R. Coulom. *Reinforcement Learning Using Neural Networks, with Applications to Motor Control. (Apprentissage par renforcement utilisant des réseaux de neurones, avec des applications au contrôle moteur)*. PhD thesis, Grenoble Institute of Technology, France, 2002. URL https://tel.archives-ouvertes.fr/tel-00003985.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1329–1338. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/duan16.html.

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

A. Geramifard, C. Dann, R. H. Klein, W. Dabney, and J. P. How. Rlpy: a value-function-based reinforcement learning framework for education and research. *J. Mach. Learn. Res.*, 16:1573–1578, 2015. URL http://dl.acm.org/citation.cfm?id=2886799.

K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *CoRR*, abs/1611.07507, 2016. URL http://arxiv.org/abs/1611.07507.

J. Harb, P. Bacon, M. Klissarov, and D. Precup. When waiting is not an option: Learning options with a deliberation cost. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3165–3172, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17421.

A. Harutyunyan, W. Dabney, D. Borsa, N. Heess, R. Munos, and D. Precup. The termination critic. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 2231–2240, 2019. URL http://proceedings.mlr.press/v89/harutyunyan19a.html.

N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. A. Riedmiller, and D. Silver. Emergence of locomotion behaviours in rich environments. *CoRR*, abs/1707.02286, 2017. URL http://arxiv.org/abs/1707.02286.

A. Jain, K. Khetarpal, and D. Precup. Safe option-critic: Learning safety in the option-critic architecture. *CoRR*, abs/1807.08060, 2018. URL http://arxiv.org/abs/1807.08060.

C. Jin, A. Krishnamurthy, M. Simchowitz, and T. Yu. Reward-free exploration for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4870–4879. PMLR, 2020. URL http://proceedings.mlr.press/v119/jin20d.html.

Y. Jinnai, J. W. Park, D. Abel, and G. D. Konidaris. Discovering options for exploration by minimizing cover time. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3130–3139. PMLR, 2019. URL http://proceedings.mlr.press/v97/jinnai19b.html.

Y. Jinnai, J. W. Park, M. C. Machado, and G. D. Konidaris. Exploration in reinforcement learning with deep covering options. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SkeIyaVtwB.

K. Khetarpal, M. Klissarov, M. Chevalier-Boisvert, P. Bacon, and D. Precup. Options of interest: Temporal abstraction with interest functions. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 4444–4451. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/5871.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

A. S. Klyubin, D. Polani, and C. L. Nehaniv. All else being equal be empowered. In M. S. Capcarrère, A. A. Freitas, P. J. Bentley, C. G. Johnson, and J. Timmis, editors, *Advances in Artificial Life, 8th European Conference, ECAL 2005, Canterbury, UK, September 5-9, 2005, Proceedings*, volume 3630 of *Lecture Notes in Computer Science*, pages 744–753. Springer, 2005. doi: 10.1007/11553090_75. URL https://doi.org/10.1007/11553090_75.

M. C. Machado, M. G. Bellemare, and M. H. Bowling. A laplacian framework for option discovery in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2295–2304, 2017. URL http://proceedings.mlr.press/v70/machado17a.html.

A. McGovern and A. G. Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 361–368. Morgan Kaufmann, 2001.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1928–1937, 2016. URL http://jmlr.org/proceedings/papers/v48/mniha16.html.

S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2125–2133, 2015. URL http://papers.nips.cc/paper/5668-variational-information-maximisation-for-intrinsically-motivated-reinforcement-learning.

O. Nachum, S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3307–3317, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/e6384711491713d29bc63fc5eeb5ba4f-Abstract.html.

OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving rubik's cube with a robot hand. *CoRR*, abs/1910.07113, 2019. URL http://arxiv.org/abs/1910.07113.

G. Ostrovski, M. G. Bellemare, A. van den Oord, and R. Munos. Count-based exploration with neural density models. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2721–2730. PMLR, 2017. URL http://proceedings.mlr.press/v70/ostrovski17a.html.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. URL http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.

C. Salge, C. Glackin, and D. Polani. *Empowerment–An Introduction*, pages 67–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-642-53734-9. doi: 10.1007/978-3-642-53734-9_4. URL https://doi.org/10.1007/978-3-642-53734-9_4.

A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6120.

J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In F. R. Bach and D. M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1889–1897. JMLR.org, 2015a. URL http://proceedings.mlr.press/v37/schulman15.html.

J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *CoRR*, abs/1506.02438, 2015b. URL http://arxiv.org/abs/1506.02438.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

A. Sharma, S. Gu, S. Levine, V. Kumar, and K. Hausman. Dynamics-aware unsupervised discovery of skills. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgLZR4KvH.

Ö. Simsek and A. G. Barto. Skill characterization based on betweenness. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 1497–1504. Curran Associates, Inc., 2008. URL http://papers.nips.cc/paper/3411-skill-characterization-based-on-betweenness.

S. P. Singh, A. G. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pages 1281–1288, 2004. URL http://papers.nips.cc/paper/2552-intrinsically-motivated-reinforcement-learning.

A. Solway, C. Diuk, N. Córdova, D. Yee, A. G. Barto, Y. Niv, and M. Botvinick. Optimal behavioral hierarchy. *PLoS Computational Biology*, 10(8), 2014. doi: 10.1371/journal.pcbi.1003779. URL https://doi.org/10.1371/journal.pcbi.1003779.

R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

R. S. Sutton, D. Precup, and S. P. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1-2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1. URL https://doi.org/10.1016/S0004-3702(99)00052-1.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109. URL https://doi.org/10.1109/IROS.2012.6386109.

C. M. Vigorito and A. G. Barto. Intrinsically motivated hierarchical skill learning in structured environments. *IEEE Trans. Auton. Ment. Dev.*, 2(2):132–143, 2010. doi: 10.1109/TAMD.2010.2050205. URL https://doi.org/10.1109/TAMD.2010.2050205.

O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama,

D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019. doi: 10.1038/s41586-019-1724-z. URL https://doi.org/10.1038/s41586-019-1724-z.

M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5279–5288, 2017. URL http://papers.nips.cc/paper/7112-scalable-trust-region-method-for-deep-reinforcement-learning-using-kronecker-factored-approximation.

R. Zhao, P. Abbeel, and S. Tiomkin. Efficient online estimation of empowerment for reinforcement learning. *CoRR*, abs/2007.07356, 2020. URL https://arxiv.org/abs/2007.07356.

A Omitted Proofs

A.1 Proof of Proposition 1

First, we make an assumption on the dependence of η from β.

Assumption 1. *The empirical distribution of options $\eta(o|x_s)$ is independent of termination conditions $\cup_{o\in\mathcal{O}}\beta^o$.*

This assumption does not strictly hold, and we can consider using, e.g., two-time scale optimization to suppress the distribution shift of η by updating β^o. We employed a replay buffer to mitigate this issue in Section 4.1.

Lemma 1. *The following equations hold.*

$$\nabla_{\theta_\beta} H(X_f|x_s) = -\sum_o \eta(o|x_s) \sum_x P^o(x|x_s) \nabla_{\theta_\beta} \ell_{\beta^o}(x) \Big[\log P(x|x_s) + 1 - \sum_{x_f} P^o(x_f|x)\Big(\log P(x_f|x_s) + 1\Big)\Big] \tag{10}$$

$$\nabla_{\theta_\beta} H(X_f|x_s, O) = -\sum_o \eta(o|x_s) \sum_x P^o(x|x_s) \nabla_{\theta_\beta} \ell_{\beta^o}(x) \Big[\log P^o(x|x_s) + 1 - \sum_{x_f} P^o(x_f|x)\Big(\log P^o(x_f|x_s) + 1\Big)\Big] \tag{11}$$

Then, sampling x, x_f from d^π and o from η,

$$\nabla_{\theta_\beta} H(X_f|x_s) = \mathbb{E}_{d^\pi,\eta}\Big[-\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\Big(\log P(x|x_s) - \log P(x_f|x_s)\Big)\Big] \tag{12}$$

$$\nabla_{\theta_\beta} H(X_f|x_s, O) = \mathbb{E}_{d^\pi,\eta}\Big[-\nabla_{\theta_\beta}\ell_{\beta^o}(x)\beta^o(x)\Big(\log P^o(x|x_s) - \log P^o(x_f|x_s)\Big)\Big]. \tag{13}$$

Proof of Lemma 1

Proof. First, we prove Equation (10). We have:

$$\begin{aligned}
\nabla_{\theta_\beta} H(X_f|x_s) &= -\nabla_{\theta_\beta} \sum_{x_f} P(x_f|x_s) \log P(x_f|x_s) \\
&= -\sum_{x_f} \Big(\nabla_{\theta_\beta} P(x_f|x_s) \log P(x_f|x_s) + P(x_f|x_s) \frac{\nabla_{\theta_\beta} P(x_f|x_s)}{P(x_f|x_s)}\Big) \\
&= -\sum_{x_f} \nabla_{\theta_\beta} P(x_f|x_s)\Big(\log P(x_f|x_s) + 1\Big) \\
&= -\sum_{x_f}\sum_o \eta(o|x_s) \underbrace{\nabla_{\theta_\beta} P^o(x_f|x_s)}_{\text{Apply eq. (2)}} \Big(\log P(x_f|x_s) + 1\Big) \\
&= -\sum_{x_f}\sum_o \eta(o|x_s) \sum_x P^o(x|x_s) \nabla_{\theta_\beta}\ell_{\beta^o}(x)(\mathbb{I}_{x_f=x} - P^o(x_f|x))\Big(\log P(x_f|x_s) + 1\Big) \\
&= -\sum_o \eta(o|x_s) \sum_x P^o(x|x_s) \nabla_{\theta_\beta}\ell_{\beta^o}(x) \sum_{x_f} (\mathbb{I}_{x_f=x} - P^o(x_f|x))\Big(\log P(x_f|x_s) + 1\Big) \\
&= -\underbrace{\sum_o \eta(o|x_s)}_{\text{sample}} \underbrace{\sum_x P^o(x|x_s)}_{\text{sample}} \nabla_{\theta_\beta}\ell_{\beta^o}(x) \times \Big[\log P(x|x_s) + 1 - \underbrace{\sum_{x_f} P^o(x_f|x)}_{\text{sample}} \Big(\log P(x_f|x_s) + 1\Big)\Big].
\end{aligned}$$

Sampling x, x_f, o, we get (12). Then we prove Equation (11).

$$\nabla_{\theta_\beta} H(X_f|x_s, O) = -\nabla_{\theta_\beta} \sum_o \eta(o|x_s) \sum_{x_f} P^o(x_f|x_s) \log P^o(x_f|x_s)$$

$$= -\sum_o \eta(o|x_s) \sum_{x_f} \Big(\nabla_{\theta_\beta} P^o(x_f|x_s) \log P^o(x_f|x_s) + P^o(x_f|x_s) \frac{\nabla_{\theta_\beta} P^o(x_f|x_s)}{P^o(x_f|x_s)} \Big)$$

$$= -\sum_o \eta(o|x_s) \sum_{x_f} \underbrace{\nabla_{\theta_\beta} P^o(x_f|x_s)}_{\text{Apply eq. (2)}} \Big(\log P^o(x_f|x_s) + 1 \Big)$$

$$= -\sum_o \eta(o|x_s) \sum_{x_f} \sum_x P^o(x|x_s) \nabla_{\theta_\beta} \ell_{\beta^o}(x) (\mathbb{I}_{x_f = x} - P^o(x_f|x)) \Big(\log P^o(x_f|x_s) + 1 \Big)$$

$$= -\sum_o \eta(o|x_s) \sum_x P^o(x|x_s) \nabla_{\theta_\beta} \ell_{\beta^o}(x) \sum_{x_f} (\mathbb{I}_{x_f = x} - P^o(x_f|x)) \Big(\log P^o(x_f|x_s) + 1 \Big)$$

$$= -\underbrace{\sum_o \eta(o|x_s)}_{\text{sample}} \underbrace{\sum_x P^o(x|x_s)}_{\text{sample}} \nabla_{\theta_\beta} \ell_{\beta^o}(x) \times \Big[\log P^o(x|x_s) + 1 - \underbrace{\sum_{x_f} P^o(x_f|x)}_{\text{sample}} \Big(\log P^o(x_f|x_s) + 1 \Big) \Big]$$

Sampling x, x_f, o, we get eq. (13). □

A.2 Proof of Proposition 2

Proof. First, using Bayes' rule, we have:

$$P^o(x_f|x_s) = \frac{\Pr(o|x_f, x_s) \Pr(x_f|x_s)}{\Pr(o|x_s)} = \frac{p_{\mathcal{O}}(o|x_f, x_s) P(x_f|x_s)}{\eta(o|x_s)}$$

Then, we have:

$$\begin{aligned} \log P^o(x_f|x_s) - \log P(x_f|xs) &= \log \frac{P^o(x_f|x_s)}{P(x_f|xs)} \\ &= \log \frac{\frac{p_{\mathcal{O}}(o|x_s, x_f) P(x_f|x_s)}{\eta(o|x_s)}}{P(x_f|xs)} \\ &= \log \frac{p_{\mathcal{O}}(o|x_s, x_f)}{\eta(o|x_s)} \end{aligned}$$

Applying this equation to the eq. (6), we have:

$$\begin{aligned} \nabla_{\theta_\beta} I(X_f; O|x_s) &= \nabla_{\theta_\beta} H(X_f|x_s) - \nabla_{\theta_\beta} H(X_f|x_s, O) \\ &= \mathbb{E}_{d^\pi, \eta} \Big[-\nabla_{\theta_\beta} \ell_{\beta^o}(x) \beta^o(x) \Big(\log P(x|x_s) - \log P(x_f|x_s) - \log P^o(x|x_s) + \log P^o(x_f|x_s) \Big) \Big] \\ &= \mathbb{E}_{d^\pi, \eta} \Big[\nabla_{\theta_\beta} \ell_{\beta^o}(x) \Big(\Big(\log P^o(x|x_s) - \log P(x|x_s) \Big) - \Big(\log P^o(x_f|x_s) - \log P(x_f|x_s) \Big) \Big) \Big] \\ &= \mathbb{E}_{d^\pi, \eta} \Big[\nabla_{\theta_\beta} \ell_{\beta^o}(x) \Big(\log \frac{p_{\mathcal{O}}(o|x_s, x)}{\eta(o|x_s)} - \log \frac{p_{\mathcal{O}}(o|x_s, x_f)}{\eta(o|x_s)} \Big) \Big] \\ &= \mathbb{E}_{d^\pi, \eta} \Big[\nabla_{\theta_\beta} \ell_{\beta^o}(x) \Big(\log p_{\mathcal{O}}(o|x_s, x) - \log p_{\mathcal{O}}(o|x_s, x_f) \Big) \Big] \end{aligned}$$

□

B Implementation Details

Clipped β loss Common PPO implementation updates π_θ multiple times. However, our preliminary experiments observed that performing multiple updates for β^o led to destructively large updates and

Algorithm 1 InfoMax Termination Critic with VIC, PPO Style

1: **Given:** Initial option-value $Q_{\mathcal{O}}$, option-policy π^o, and termination function β^o.
2: Let $B_{\mathcal{O}}$ be a replay buffer for storing option-transitions.
3: **for** $k = 1, ...$ **do**
4: **for** $i = 1, 2, ..., N$ **do** ▷ Collect experiences from environment
5: Sample termination variable b_i from $\beta^{o_i}(x_i)$
6: **if** $b_i = 1$ **then**
7: Store (x_s, x_f, o_i), (x_{s+1}, x_f, o_i), ..., (x_{s+h}, x_f, o_i) to the replay buffer $B_{\mathcal{O}}$
8: **end if**
9: Choose next option o_{i+1} by ϵ-Greedy
10: Receive reward R_i and state x_{i+1}, taking $a_i \sim \pi_{o_{i+1}}(x_i)$
11: **end for**
12: **for** $k = 1, 2, ...,$ Num. of PPO epochs **do** ▷ Optimize π^o, $Q_{\mathcal{O}}$, and β^o
13: **for all** x_i in the trajectory **do**
14: Compute R_{VIC} from the target network
15: Compute $\hat{A}^o_{\text{ind}}$ by (8) using R_{VIC}
16: Update $\pi^o(a_i|x_i)$ via PPO using $\hat{A}^o$
17: Update $Q_{\mathcal{O}}(x_i, o)$ to regress to $\hat{A}^o$
18: **if** o_i has already terminated **then**
19: Update $\beta^o(x_i)$ via (7)
20: **end if**
21: **end for**
22: **end for**
23: Train $\hat{p}$ and $\hat{\mu}$ by option-transitions sampled from $B_{\mathcal{O}}$
24: **if** $k \mod K_{\text{VIC}}$ **then**
25: Update the target network for VIC
26: **end if**
27: **end for**

resulted in the saturation of β^o to zero or one. Hence, to perform PPO-style multiple updates, we introduce a clipped objective of eq. (7):

$$L^{\text{CLIP}}(\theta_\beta) = \text{clip}(\ell_{\beta^o}(x) - \ell_{\beta^o{}_{\text{old}}}(x), -\epsilon_\beta, \epsilon_\beta)\beta^o{}_{\text{old}}(x)\Big(\log p_{\mathcal{O}}(o|x_s, x) - \log p_{\mathcal{O}}(o|x_s, x_f)\Big), \tag{14}$$

where ϵ_β is a small coefficient, and $\beta^o{}_{\text{old}}$ is an old β^o before updating. We also add maximization of the entropy of β^o for preventing the termination probability saturating on zero or one. To this end, we maximize $L^{\text{CLIP}}(\theta_\beta) + c_{H_\beta} H(\beta^o(x))$ via gradient ascent w.r.t. θ_β, where c_{H_β} is a weight of the entropy bonus.

Full description of the algorithm Algorithm 1 shows a full description of our implementation of IMTC on PPO when combined with VIC rewards. As of the original PPO, it is built on the A2C-style (Mnih et al., 2016; Wu et al., 2017) architecture with multiple synchronous actors and a single learner. First, we collect N-step experiences interacting with environments. At line 7, we append tuples corresponding to option transitions $(x_s, x_f, o_i), ..., (x_{s+h}, x_f, o_i)$ to $B_{\mathcal{O}}$. Here, we do not use all transitions and store first h_{max} options to prevent memory shortage. We used $h_{\text{max}} = 10$ or $h_{\text{max}} = 20$. Then we update π^o, $Q_{\mathcal{O}}$, and β^o. Line 13 is done via minibatch sampling in the actual implementation. We also update $\hat{p_{\mathcal{O}}}$ for estimating the gradient (7), sampling from the replay buffer $B_{\mathcal{O}}$. We empirically found that rapidly changing R_{VIC} leads to unstable learning, especially when IMTC is used in parallel. Thus, we employ a target network to compute R_{VIC} and periodically update it at line 26. We used $K_{\text{VIC}} = 10$ or $K_{\text{VIC}} = 20$ in the experiments.

C Experimental Details

Our anonymized code used for all our experiments is on https://anonymous.4open.science/r/imtc-anonymized-code-E5D1/.

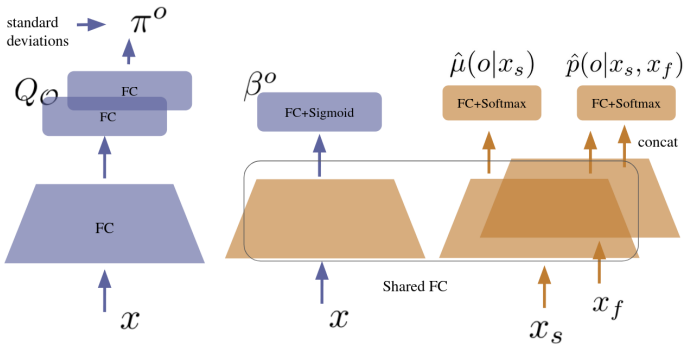


Figure 6: Neural Network architecture used

Description	Value
γ	0.99
Adam Learning Rate	3×10^{-4}
Adam ϵ	1×10^{-4}
Clip parameter for ℓ_{β^o} (ϵ_β)	0.05
Num. timesteps per rollout	256
Num. actors	16
GAE λ	0.95
Num. epochs for PPO	10
Minibatch size for PPO	1024
Weight of $H(\pi^o)$ (c_H)	0.001
Weight of $H(\beta^o)$ (c_{H_β})	0.01
Gradient clipping	0.5
Capacity of $B_{\mathcal{O}}$	8192
Max num. option transitions to store (h_{max})	20
Num. epochs for training $\hat{p}$ and $\hat{\mu}$	4
Minibatch size for training $\hat{p}$ and $\hat{\mu}$	2048
Scaling of R_{VIC} (c_{VIC})	0.005
Synchronizing interval of the target network for VIC (K_{VIC})	20

Table 1: Hyperparameters used in continuous control experiments

C.1 Network Architecture

Figure 6 illustrates the neural network architecture used in our experiments.

In continuous control experiments, we encode the state by two fully connected layers with 64 units. π^o is parameterized as a Gaussian distribution with separated networks for standard derivations per option, similar to Schulman et al. (2015a). We used ReLU as an activator for all hidden layers and initialized networks by the orthogonal (Saxe et al., 2014) initialization in all experiments. Note that the last layer for intra-option policy π^o was initialized with small values, following the standard practice (Andrychowicz et al., 2021). We used the Adam (Kingma and Ba, 2015) optimizer in all experiments. Unless otherwise noted, we used the default parameters in PyTorch (Paszke et al., 2019) 1.8.1.

C.2 Hyperparameters

Table 1 shows all hyperparameters used in IMTC + VIC experiments on MuJoCo continuous control tasks.

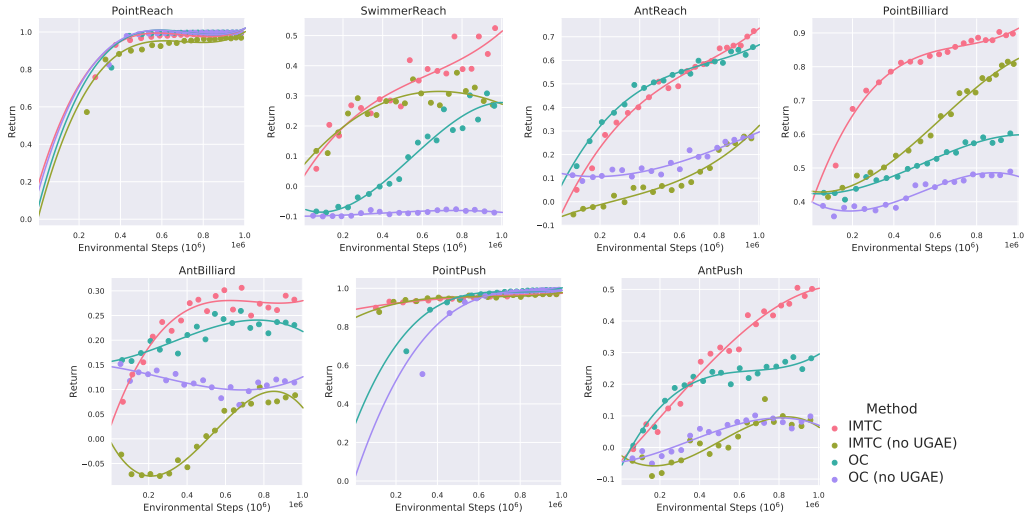


Figure 7: Learning curves for transfer learning experiments with

C.3 Environment Implementation

Gridworld is based on RLPy (Geramifard et al. (2015), BSD3 License). We constructed continuous control environments on the MuJoCo (commercial license, Todorov et al. (2012)), using OpenAI Gym (MIT license, Brockman et al. (2016)). Especially, point environments are implemented based on "PointMaze" in rllab (MIT license, Duan et al. (2016)) with some modifications, mainly around collision detection. We also refered to the modified PointMaze code[2] (Apache 2.0 license) relased by Nachum et al. (2018). The swimmer robot is originally used in Coulom (2002).

C.4 Computational Resources

All experiments are conducted on a private cluster with NVIDIA V100 and P100 GPUs. On the cluster, training IMTC with VIC on MuJoCo PointMaze domain for 4×10^6 steps takes about 27 minutes.

C.5 Effect of UGAE

To investigate the effect of UGAE (9), we compared IMTC and OC with and without UGAE in task adaptation experiments. `IMTC (no UGAE)` and `OC (no UGAE)` estimate the adavantage ignoring the future rewards after switching options, similar to advantage estimation proposed by Bacon et al. (2017). Figure 7 shows the result. We can see that UGAE improves the performance in all domains for both IMTC and OC.

C.6 Number of Options

Figure 8 shows learning curves of IMTC agents with variable number of options $(2, 4, 6, 8)$ in task adaptation experiments. Increasing the number of options sometimes improves the performance (e.g., in `SwimmerReach`), but the relationship is not obvious. However, we can confirm that increasing the number of options does not have a bad effect on the performance, while it reduces training steps per each intra-option policy, thanks to network sharing Appendix C.1.

[2] https://github.com/tensorflow/models/tree/v2.3.0/research/efficient-hrl

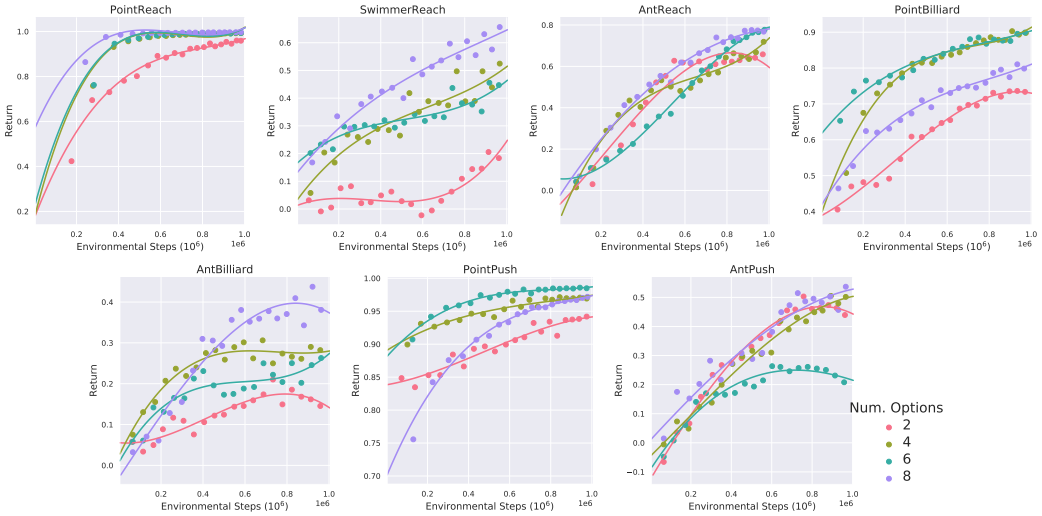


Figure 8: Learning curves for transfer learning experiments.