# SOFT OPTIONS CRITIC

WORK IN PROGRESS

October 13, 2019


## ABSTRACT

Hierarchical Reinforcement learning aims to solve complex tasks by learning reusable skills that exploit the compositional structure of the task to speed up learning. The option critic architecture successfully demonstrates a way to automate learning of these skills while solving a specific task. However, this framework uses on-policy updates which makes it highly sample inefficient and impractical for solving control tasks with high dimensional state and action space. Additionally, the framework uses entropy as a regularization which can sometimes degrade performance. In this work, we propose Soft Option Critic framework which learn maximum entropy options in an off-policy manner to improve sample efficiency. Experimental results show that our proposed framework enables the agent to learn to achieve higher rewards faster than vanilla options-critic and state of the art algorithms like PPO and Soft Actor-Critic in many control tasks.


## 1 Introduction

Enabling an agent to learn complex and long tasks has been a challenging problem in RL for several decades. Humans often approach such complex tasks by identifying simpler sub-tasks and learning skills to solve them. Hence, recent research in RL has focused on developing hierarchical frameworks that enable an agent to exploit the compositional structure of the task and learn reusable skills as opposed to learning with primitive actions. The options framework provides a way to learn and represent these skills as temporally extended actions also termed as options. Prior research has shown that these temporal abstractions significantly improves exploration and reduce the complexity of selecting actions. The options-critic architecture uses this framework to proposes a set of theorems that makes it feasible to parameterize and learn various aspects of options in an end-to-end manner using stochastic gradient descent method. However, this framework has two major limitations 1) The framework uses on-policy updates which require new samples to be generated for nearly every update and hence makes it highly sample inefficient. This makes it impractical for learning complex control tasks with high dimensional state and action space 2) The framework does not maximize entropy along with expected returns at every step in a constrained manner but instead uses entropy as a regularization which can sometimes degrade performance Haarnoja et al. (2018a). In this paper, we develop a new variant of options critic which does not bear these limitations. The recently introduced Soft Actor critic architecture provides an off-policy actor-critic algorithm based on the maximum-entropy framework which enables an agent to maximize rewards while acting as randomly as possible. Taking inspiration from this work, we develop an off-policy option-critic based on maximum entropy framework to address the problem of sample inefficiency. Using soft policy improvement theorem Haarnoja et al. (2018a), we derive intra-option policy gradient and intra-option termination gradient. We finally evaluate the proposed soft-option critic framework on OpenAI Roboschool tasks with continuous state and action space. Our experimental results show that the proposed framework enable an agent to achieves higher rewards much faster than vanilla options-critic and state of the art algorithms like PPO and Soft Actor-Critic in many control tasks.

### 1.1 Background

In Reinforcement learning, an agent generally does not have access to state-transition dynamics of the environment or reward function or both and hence learns by interacting with the environment. In this setting, the environment is modelled as a Markov Decision Process (MDP) which is defined as a tuple $< S, A, P, r, d_0, \gamma >$ where S is set of states of the environment as perceived by the agent, A is the set of actions that the agent can take, $P : S \times A \to (S \to [0, 1])$

is the state transition probability, $r : S \times A \rightarrow R$ is the reward function, $d_0$ is the initial state distribution ie- $Pr(s_0 = s)$ and $\gamma \in [0, 1]$ is the factor by which rewards are discounted over time. At any discrete time step $t \in \{0, 1, 2..T\}$, the agent observes state $s_t \in S$and takes action $a_t \in A$ which transitions the agent to state $s_{t+1}$ at time step $t + 1$ and the agent receives a reward $r_t$. In a finite horizon MDP, this process ends after T time steps or sooner if the agent reaches the termination state. When this process ends, we say that an episode is completed and t is reset to 0. In an infinite horizon MDP, $T \rightarrow \infty$. The returns of an episode is defined as $G = \sum_{t=0}^{\infty} r^t R_t$. A policy $\pi : S \times A \in [0, 1]$ is the probability distribution over actions conditioned on the given state. The value function of policy $\pi$ is defined as the discounted returns starting from state s ie- $V_\pi(s) = \mathbf{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s]$. Likewise, the action-value function of a policy is defined as the discounted returns starting from state s and taking action a - $Q_\pi(s_t, a_t) = \mathbf{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0, a_0]$. A policy is said to be greedy with respect to A policy $\pi$ is said to be greedy with respect to a given action value function Q if $\pi(s_t, a_t) > 0$ and $a_t = argmax_a Q(s_t, a)$. The value of V and Q can be learnt using one-step TD learning TD(0) which is a special case of $TD(\lambda)$(Sutton et al [1998]). The action value is updated using the equation $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\delta$ where $\alpha$ is the step size. We define $\delta$ is the TD(0) error $\delta = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$.

**Policy Gradient Methods**: These methods use stochastic gradient descent over maximizing discounted returns objective to get smooth updates for the policy.The policy gradient theorem introduced by Sutton et al. (1999b) gives the expression for the gradient of the expected discounted returns objective with respect to policy parameters $\theta$. The expected discounted returns objective is defined as $\rho(\theta, s_0) = \mathbf{E}_{\pi_\theta}[\sum_{t=0}^{\infty} r^t r_t \mid s_0]$ and its gradient with respect to the policy parameters is given as $\frac{\partial}{\partial\theta}\rho(\theta, s_0) = \sum_s \mu_{\pi_\theta}(s \mid s_0) \sum_a \frac{\partial}{\partial\theta}\pi_\theta(a \mid s) Q_{\pi_\theta}(s, a)$ where $\mu_{\pi_\theta}(s \mid s_0) = \sum_{t=0} \infty \gamma^t P(s_t = s \mid s_0)$ is the discounted state distribution starting from state $s_0$.

**Options Framework** The options framework (Sutton, Precup, and Singh 1999; Precup 2000) represents options as temporally extended actions. Options $\omega \in \Omega$ are markovian and are defined as a tuple$(I_\omega, \pi_\omega, \beta_\omega)$ where $I_\omega \subseteq S$ is the initiation set which comprises of states in which the option can be initiated, $\pi_\omega$ is the intra-option policy which gives the option's probability distribution over actions conditioned on states, and $\beta_\omega : S \rightarrow [0, 1]$ is a termination function which determines the probability of terminating the option in a given state. Options can be viewed as skills acquired by the agent to solve sub tasks within a task. An MDP with a set of options becomes a Semi-Markov Decision Process (SMDP) Sutton et al. (1999a) . This SMDP has an optimal value function over options $V_\Omega(s)$ and option-value function $Q_\Omega(s, \omega)$. Bellman equation can be used for updating the value of a state-option pair Sutton et al. (1999a) is given as $Q(s_t, \omega_t) \leftarrow Q(s_t, \omega_t) + \alpha[r_t + \gamma(1 - \beta_{\omega,v}(s_t))Q(s_{t+1}, \omega_t) + \gamma\beta_{\omega,v}(s_t) max_{\omega_{t+1} \in \Omega} Q(s_{t+1}, \omega_{t+1}) - Q(s_t, \omega_t)]$ where $\omega$ is obtained from the policy over options $\pi_\Omega$.

**Options critic framework** The options-critic architecture Bacon et al. (2016) provides a way to apply policy gradient algorithms to temporally extended actions to directly maximize the expected discounted returns. In this framework, the agent chooses an option $\omega$ according to inter-option policy $\pi_\Omega$ and executes actions according to policy $\omega$ until the option terminates according to the termination condition $\beta_{\omega,\nu}$. The next option is then chosen using $\pi_\Omega$ and this process continues until the episode terminates. $\pi_{\omega,\theta}$ represents the intra-option policy parameterised by $\theta$ and $\beta_{\omega,\nu}$ represents the option termination function parameterized by $\nu$. If option $\omega_t$ is executing at time t in state $s_t$, then the probability of transitioning to $(s_{t+1}, \omega_{t+1})$ in one step is given by: $P(s_{t+1}, \omega_{t+1} | s_t, \omega_t) = \sum_a \pi_{\omega_t,\theta}(a \mid s_t) P(s_{t+1} | s_t, a)((1 - \beta_{\omega_t,\mu}(s_{t+1}))\mathbf{1}\omega_t = \omega_{t+1} + \beta_{\omega_t,\mu}(s_{t+1})\pi_\Omega(\omega_{t+1} | s_{t+1}))$
The value of executing an action at a given state-option pair $s_t, w_t$ given by : $Q_U : S \times \Omega \times A \rightarrow R$ where $Q_U(s_t, \omega, a_t) = r(s_t, a_t) + \gamma \sum_{s_{t+1}} P(s_{t+1} \mid s_t, a_t) U(s_{t+1}, \omega_t)$ where U represents the value of executing an option $\omega$ at a state $s_{t+1} : U(s_{t+1}, \omega_t) = (1 - \beta_{\omega,v}(s_{t+1}))Q_\Omega(s_{t+1}, \omega_t) + \beta_{\omega,v}(s_{t+1})V_\Omega(s_{t+1})$.$Q_\Omega$ represents the optimal option - value function and can be represented as $Q_\Omega(s_t, \omega_t) = \sum_{a_t} \pi_{\omega,\theta} Q_U(s_t, w_t, a_t)$.
$V_\Omega$ represents the optimal value function for policy over options $\Omega$ and can be represented as $V_\Omega(s_t) = \sum_{\omega_t} \pi_\Omega(s_t, \omega_t) Q_\Omega(s_t, \omega_t)$. The expected discounted returns starting at a state $s_0$ and option $\omega_0$ is represented as $\rho(\Omega, \theta, \mu, s_0, \omega_0) = \mathbf{E}_{\mathbf{\Omega},\theta,\omega}[\sum_{t=0}^{\infty} \gamma^t r^t \mid s_0, \omega_0)]$. The gradient of the expected discounted returns objective with respect to intra-option policy parameters $\theta$ and initial conditions $(s_0, \omega_0)$ is given by the intra-option gradient theorem as $\frac{\partial}{\partial\theta}\rho(\pi, s_0, \omega_0) = \sum_{s_t,\omega_t} [\nu(s_t, \omega_t \mid s_0, \omega_0) \sum_a \frac{\partial}{\partial\theta}\pi_{\omega_t,\theta}(s_t, a_t) Q_U(s_t, \omega_t, a_t)]$ where $\mu(s_t, \omega_t \mid s_0, \omega_0)$ is the discounted state-option pair distribution with $\mu(s_t, \omega_t \mid s_0, \omega_0) = \sum_{t=0} \infty \gamma^t P(s_t = s, \omega_t = \omega | s_0, \omega_0)$. Likewise, the gradient of the expected discounted returns with respect to option termination parameters $\mu$ and the initial condition $(s_1, \omega_0)$ is given by the intra-option termination gradient theorem as: $\frac{\partial}{\partial\nu}\rho(\pi, s_1, \omega_0) = -\sum_{s_{t+1},\omega_t} \mu(s_{t+1}, \omega_t \mid s_1, \omega_0) \frac{\partial}{\partial\nu}\beta_{\omega,v}(s_t) A(s_{t+1}, \omega_t)$ where A is the advantage function given as $A_\Omega(s_t, \omega_t) = Q_\Omega(s_t, \omega_t) - V_\Omega(s_t)$

**Soft Actor Critic** The Soft Actor Critic architecture Haarnoja et al. (2018b) provides an off-policy actor

critic algorithm based on maximum entropy framework that outperforms policy gradient methods like PPO, TRPO in terms of sample efficiency. The algorithm is derived from the maximum entropy variant of policy iteration method. It optimizes the following objective.

$$J(\pi) = \sum_{t=0}^{\infty} \mathbf{E}_{s_t,a_t\sim\rho_\pi} \left[ \sum_{l=t}^{\infty} \gamma^{l-t} \mathbf{E}_{s_l\sim p, a_l\sim\pi} \left[ r(s_t,a_t) + \alpha H(\pi(. \mid s_t) \mid s_t, a_t) \right] \right]$$

The temperature parameter $\alpha$ indicates the relative importance of the entropy term against reward and determines the stochasticity of the optimal policy. This objective maximizes the expected discounted returns and entropy for future states starting from every state-action pair $(s_t, a_t)$ weighted by its probability $\rho_\pi$ under current policy.
The soft policy iteration alternates between soft policy evaluation and soft policy improvement. In policy evaluation , soft Q value is computed in an iterative manner, starting from any function Q:S $\times$ A $\rightarrow \mathbb{R}$ and repeatedly applying the modified Bellman operator $T^\pi$ given by:

$$T^\pi Q(s_t,a_t) \equiv R(s_t,a_t) + \gamma \mathbf{E}_{\mathbf{s_{t+1}}\sim\mathbf{p}}[V(s_{t+1})] \tag{4}$$

where

$$V(s_t) = \mathbf{E}_{\mathbf{a_t}\sim\pi}[Q(s_t,a_t) - \alpha \log \pi(a_t \mid s_t)] \tag{5}$$

In the policy improvement step, the policy is updated to follow the distribution of exponential of the soft Q function. Hence the policy is updated according to the following objective:

$$\pi_{new} = \arg\min_{\pi'\in\prod_{\omega,\theta}} D_{KL}\left( \pi'(\cdot \mid s_t, w_t) \middle| \frac{exp(Q^{\pi_{old}}(s_t,w_t,\cdot))}{Z^{\pi_{old}(s_t,w_t)}} \right) \tag{8}$$

The authors formulate the maximizing entropy objective as a constraint to maximizing returning objective and solve the same to automate tuning of the temperature hyperparameter.Stochastic gradient descent is used on the following objective to automatically tune the temperature hyperparameter to adjust the expected entropy over visited states to match the target entropy value.

$$\max_{\pi 0:T} \mathbf{E}_{\rho_\pi}[\sum_{t=0}^{\infty} r(s_t,a_t)]$$
$$\text{s.t } \mathbf{E}_{\mathbf{s_t},\mathbf{a_t}\sim\rho_\pi}[-\log(\pi_t(a_t \mid s_t))] >= H \,\forall t$$

## 2 Soft Option Critic

In this section, we modify the options framework to incorporate soft actor critic architecture and present a deep reinforcement learning algorithm for learning options. We use Soft Policy Iteration which alternates between soft policy evaluation and soft policy improvement to learn optimal maximum entropy option policies. We experiment learning of policy over options with Q learning and off-policy actor critic and discuss the effect of adding sparsity and variance regularization on policy over options. Here onwards, we would interchangeably refer to policy over options as inter-option policy and option policy as intra-option policy.

### 2.1 Soft Option Evaluation

For a fixed option policy, the Soft Q-value function is computed iteratively, starting from any function $Q : S \times \Omega \times A$ and repeatedly applying a modified Bellman backup operator $T^\pi$ given by:

$$T^\pi Q(s_t,w_t,a_t) \equiv R(s_t,a_t) + \gamma \sum_{s_{t+1}} \Pr(s_{t+1} \mid s_t,a_t) U(s_{t+1},w_t)$$

where

$$U(s_{t+1},\omega_t) = (1-\beta_{\omega,v}(s_{t+1}))Q_\Omega(s_{t+1},\omega_t) + \beta_{\omega,v}(s_{t+1})V_\Omega(s_{t+1}).Q_\Omega(s_t,\omega_t)$$

The Q-value function for inter-option policy is given as:

$$Q_\Omega(s_t,w_t) = \mathbf{E}_{\mathbf{a_t}\sim\pi}[Q(s_t,w_t,a_t) - \alpha_{\omega_t,\theta} \log \pi_{\omega,\theta}(a_t \mid s_t,w_t)]$$

where $\alpha_{\omega_t,\theta}$ is the temperature parameter corresponding to option $\omega_t$. We prove the convergence of the modified Bellman backup operator in the appendix section.

### 2.2 Soft Option Improvement

In the soft policy improvement step, we update the inter-option policy to minimize the KL divergence between the inter-option policy distribution and normalized form of exponential of the new inter-option Q value function. We derive the soft intra-option policy update and inter-option policy update by taking the gradient of the KL divergence objective with respect to intra-option policy parameters and inter-option parameters respectively.

$$\pi_{\omega}^{new} = argmin_{\pi'_{\Omega} \in \prod_{\Omega}} D_{KL}\left(\pi'_{\Omega}(\cdot \mid s_t) \middle| \frac{exp(Q^{\pi_{\Omega}^{old}}(s_t, \cdot))}{Z^{\pi_{\Omega}^{old}(s_t)}}\right) \tag{1}$$

This objective is equivalent to maximizing the discounted expected reward and entropy over actions for future states originating from every state-option-action tuple $(s_t, w_t, a_t)$ weighted by its probability $\rho_\pi$ under the current policy over options.

$$J(\pi) = \sum_{t=0}^{\infty} \mathbf{E}_{\mathbf{s_t}, \mathbf{w_t}, \mathbf{a_t} \sim \rho_\pi}[\sum_{l=t}^{\infty} \gamma^{l-t} \mathbf{E}_{\mathbf{s_l} \sim \mathbf{p}, \mathbf{w_l} \sim \pi_{\mathbf{\Omega}}, \mathbf{a_l} \sim \pi_{\omega_t, \theta}}[r(s_t, a_t) + \alpha_{\omega_t, \theta_t} H(\pi_{\omega_t, \theta}(. \mid s_t) \mid s_t, w_t, a_t) + \alpha_{\Omega} H(\pi_{\Omega}(. \mid s_t) \mid s_t, w_t)]]$$

### 2.3 Architecture

Since the proposed framework should be able to solve complex tasks with continuous state and action space, we use function approximators to approximate inter-option Q-value function $Q(s_t, w_t)$, intra-option Q-value function $Q(s_t, w_t, a_t)$, intra-option policy $\pi_{\omega,\theta}(s_t, a_t)$, inter-option policy $\pi_{\Omega}(s_t, w_t)$ and termination function $\beta_{\omega,\upsilon}(s_t)$ . $\psi$, $\phi$, $\theta$, $\Omega$ and $\upsilon$ represent the function approximators for these networks in the order specified above. The updates make use of target inter-option Q-value network whose weights are updated with exponential moving average of the inter-option Q-value network weights. We represent target inter-option Q-value function by $\bar{Q}(s_t, w_t)$. We represent the intra-option policy as a Gaussian function whose mean and covariance is given by the inter-option policy network. The inter-option network outputs probabilities over options for a given input state. We also use two inter-option Q-value networks and use the minimum of the two Q-values to mitigate positive bias which degrades performance in policy improvement step of value based methods. We maintain a replay buffer to store the agent's last N experiences. We randomly sample a mini-batch from the replay buffer and update all the function approximators in an off-policy manner. We now derive off-policy updates for each of these networks.

#### 2.3.1 Soft Option Evaluation and Update:

In the option evaluation step, we update the inter-option Q-value function approximator to minimize the squared Bellman residual error of the sampled mini-batch. We use the minimum of the two intra-option Q-values estimated using intra-option Q-value networks for the update.

$$J_Q(\psi) = E_{s,w}[\frac{1}{2}(Q_\psi(s_t, w_t) - E_{a_t \sim \theta}[Q_\phi(s_t, w_t, a_t) - \alpha_{\omega_t, \theta_t} \log \pi_{\omega,\theta}(a_t|s_t))^2]]$$

$$\frac{\partial}{\partial \psi} J_Q(\psi) = \frac{\partial}{\partial \psi} Q_\psi(s_t, w_t)(Q_\psi(s_t, w_t) - E_{a_t \sim \theta}[Q_\phi(s_t, w_t, a_t)])$$

$$\psi = \psi - \lambda_\psi \nabla J_Q(\psi)$$

Likewise, the intra-option Q-value function is trained to minimize the soft Bellman residual error. We found that using value function estimates derived from target inter-option Q-value network stabilized learning.

$$J_Q(\phi) = \mathbf{E}_{\mathbf{s_t}, \mathbf{w_t}, \mathbf{a_t}}[\frac{1}{2}(Q_\phi(s_t, w_t, a_t) - (R(s_t, a_t) + \gamma \mathbf{E}_{\mathbf{s_{t+1}}}[\sum_{w_{t+1}} (1 - \beta_{w,v}(s_{t+1})) I_{w_t = w_{t+1}} \bar{Q_\psi}(s_{t+1}, w_{t+1})$$
$$+ \beta_{w,v}(s_{t+1})(\pi_\Omega(w_{t+1}|s_{t+1}) \bar{Q_\psi}(s_{t+1}, w_{t+1}))]))^2]$$

Differentiating with respect to $\phi$ we get:

$$\frac{\partial}{\partial \phi} J_Q(\phi) = \frac{\partial}{\partial \phi} Q_\phi(s_t, w_t, a_t)(Q_\phi(s_t, w_t, a_t) - (R(s_t, a_t) + \gamma \mathbf{E}_{\mathbf{s_{t+1}}}[\sum_{w_{t+1}} ((1 - \beta_{w,v}(s)) I_{w_t = w_{t+1}} \bar{Q}_\psi(s_{t+1}, w_{t+1})$$
$$+ \beta_{w,v}(s_{t+1})(\pi_\Omega(w_{t+1}|s_{t+1})(\bar{Q}_\psi(s_{t+1}, w_{t+1})))]))$$

$$\phi = \phi - \lambda_\phi \nabla J_Q(\phi)$$

### 2.3.2 Inter-option and Intra-option policy update:

In the policy improvement step, we update inter-option policy distribution towards exponential of the inter-option Q value function. We derive the inter-option policy update by taking the gradient of the KL divergence objective in equation (1). Simplifying equation (1) we get,

$$J_\pi(\Omega) = \mathbf{E}_{\mathbf{s_t}\epsilon\sim\mathbf{N(0,1)}}[\log \pi_\Omega(w_t|s_t) - Q(s_t, w_t)]$$

The inter-option policy network outputs a discrete distribution from which option $\omega_t$ is sampled. Reparameterisation trick fails for discrete distributions so we use Gumbel Softmax reparameterisation trick as follows: Let $\omega_t$ be a discrete random variable with $P(\omega_t = k)$ where k $\in [1, 2, \dots |\omega|]$. $P(\omega_t = k) \propto \alpha_{\theta k}$ random variable where $\alpha_k$ is the probability assigned to option k by the inter-option network. We define $G_{k k\leq|w|}$ to be an iid sequence of standard Gumbel random variables. We can derive a differentiable sequence of simplex valued random variables ${\omega_t}^\tau$ which represents a nearly one hot-encoded form of option $\omega_t$.

$$\begin{aligned}
U &\sim \text{Unif}[0, 1] \\
G &= -log(-log(U)) \\
{\omega_t}^\tau = (\omega_t{}_k^\tau)_k = f_\tau(log\alpha_\theta + G) &= \left(\frac{e^{(\log \alpha_{\theta\, k}+G_k)/\mathcal{T}}}{\sum_{i=1}^{|w|} e^{(\log \alpha_{\theta\, i}+G_i)}}\right)_k \\
J_\pi(\Omega) &= \mathbf{E}_{\mathbf{s_t}}[-Q_\psi(s_t, {\omega_t}^\tau)] \\
\frac{\partial}{\partial\Omega}J_\pi(\Omega) &= -\frac{\partial}{\partial w_t}Q_\psi(s_t, \omega_t)\frac{\partial}{\partial\theta}f_\tau(log\alpha_\theta + G) \\
\Omega &\leftarrow \lambda_\Omega \nabla_\Omega J_\pi(\Omega)
\end{aligned}$$

where $\mathcal{T}$ is the temperature parameter. We derive the soft intra-option gradient by taking the gradient of the objective $J(\Omega)$ with respect to intra-option policy parameters $\theta$. Since the output of the intra-option policy network is a gaussian distribution over the space of actions, we use reparametrisation trick to obtain a low variance estimate of the gradient of loss with respect to inter-option policy parameters. We reparameterize the output of the inter-option policy network as follows:

$$\begin{aligned}
a_t &= f'_{\omega,\theta}(\epsilon; s_t) \\
\frac{\partial}{\partial\theta}J_\pi(\Omega) &= \frac{\partial}{\partial\theta}(\mathbf{E}_{\mathbf{s_t}\epsilon\sim\mathbf{N(0,1)}}[\log \pi_\Omega(w_t|s_t) - (Q(s_t, w_t, a_t) - \alpha_{\omega,\theta} \log \pi_{\omega,\theta}(a_t|s_t))]) \\
\frac{\partial}{\partial\theta}J(\omega,\theta) = \alpha_{\omega,\theta}\frac{\partial}{\partial\theta}\log \pi_{\omega,\theta}(s_t, w_t, a_t) &+ \left(\alpha_{\omega,\theta}\frac{\partial}{\partial a_t}\log \pi_{\omega,\theta}(s_t, a_t) - \frac{\partial}{\partial a_t}Q_\phi(s_t, w_t, a_t)\right)\frac{\partial}{\partial\theta}f'_{\omega,\theta}(\epsilon; s) \\
\theta &\leftarrow \lambda_\theta \nabla_\theta J_\pi(\omega,\theta)
\end{aligned}$$

### 2.3.3 Termination function update:

The option critic termination gradient theorem states that the goodness of the termination function can only be evaluated upon entering the next state. We derive the termination update by taking the gradient of inter-option Q-value function with respect to the termination function parameters $\upsilon$:

$$\begin{aligned}
\frac{\partial}{\partial\upsilon}(J_\pi(\upsilon)) = -\frac{\partial}{\partial\upsilon}(R(s_t, a_t) + \gamma\mathbf{E}_{\mathbf{s_{t+1}}}[\sum_{w_{t+1}}&(1 - \beta_{w,\upsilon}(s_{t+1}))I_{w_t=w_{t+1}}\bar{Q_\psi}(s_{t+1}, w_{t+1}) \\
&+\beta_{w,\upsilon}(s_{t+1})(\pi_\Omega(w_{t+1}|s_{t+1})\bar{Q_\psi}(s_{t+1}, w_{t+1}))]) \\
&= \frac{\partial}{\partial\upsilon}\beta_{w,\upsilon}(s_{t+1})(Q_\psi(s_{t+1}, w_t) - V_\psi(s_{t+1})) \\
&\upsilon \leftarrow \lambda_\upsilon \nabla_\upsilon J(\upsilon)
\end{aligned}$$

## 2.4 Automatic Temperature parameter tuning

We adopt Soft Actor Critic's automatic temperature parameter tuning method Haarnoja et al. (2018a) for tuning temperature parameters associated with intra-option policies. This method enables maximizing returns while maintaining

a minimum target entropy H for each option policy.

$$\max_{\pi 0:T} \mathbf{E}_{\rho_\pi}[\sum_{t=0}^{\infty} r(s_t, a_t)]$$
$$\text{s.t } \mathbf{E}_{\mathbf{s_t},\mathbf{a_t}\sim\rho_\pi}[-\log(\pi_{\omega,\theta}(a_t \mid s_t))] >= H \,\forall t$$

Solving this constrained optimization problem gives the following objective. We compute the gradient of this objective with respect to each option policy's temperature parameter $\alpha_{\theta,\omega}$ to derive corresponding updates:

$$J(\alpha_{\theta,\omega}) = \mathbf{E}_{w_t\sim\pi_{\omega,\theta}}[-\alpha_{\theta,\omega}\log\pi_{\omega,\theta}(a_t|s_t) - \alpha_{\theta,\omega}H]$$
$$\alpha_{\theta,\omega_i} \leftarrow \lambda_{\alpha_{\theta,\omega_i}}\nabla_{\alpha_{\theta,\omega_i}} J(,\alpha_{\theta,\omega})\forall i \in \{1, 2..|\omega|\}$$

### 2.5 Regularization Penalties:

#### 2.5.1 Sparsity and Variance Regularization

To encourage sparsity and diversity in options selected, we borrow sparsity and variance penalty from a study on conditional computation in neural networks Bengio et al. (2015) and modify them to fit the requirements of our framework.

$$L_b = \frac{1}{|\omega|}\sum_{\omega} ||\mathbf{E}_{\mathbf{s_t}}[\pi_\Omega(w_t|s_t)] - \tau||_2$$
$$L_v = -\frac{1}{|\omega|}\sum_{\omega} var_s\{\pi_\Omega(\omega|.)\}$$
$$L_e = \mathbf{E}_{\mathbf{s}}[var_\omega \pi_\Omega(.|s)]$$

where $\tau$ is the target sparsity which we set equal to $\frac{1}{|\omega|}$. $L_b$ encourages uniform activation of options in expectation over data Bengio et al. (2015). $L_e$ favors high variance in probability distribution over options for a given sample thus encouraging the option policy to be deterministic. $L_v$ encourages high variance in the activations of a given option within an episode. Hence, these three regularizations induce specialization of options in different regions of the state space there by improving temporal consistency.

### 2.6 Mutual Information Regularization

To further encourage specialization of options in different regions of the state space, we minimize the mutual information between the distribution over actions predicted by policy of any two options for a given state. We incorporate the mutual information penalty introduced by authors of OptionGan framework Henderson et al. (2017) in our framework as follows:

$$L_{mi} = \sum_{\omega}\sum_{\omega',\omega\neq\omega'} -0.5\log(1 - {\rho_{i,j}}^2) \tag{1}$$

where $\rho_{i,j}$ is the mutual information between policy of $option_i$ and $option_j$.

### 2.7 Off-Policy Correction

In order to be able to update inter-option policy and intra-option policies in an off-policy manner, we need to relabel options for each $(s, a, r, s')$ tuple but selecting the option with the highest probabilities of producing the given action.

### 2.8 Soft Option Critic Algorithm

**Input:** $\Omega, \phi_1, \phi_2, \psi, \theta, T, \upsilon$
$n \leftarrow n$
$D \leftarrow []$
$\bar{\phi_1} \leftarrow \phi_1, \bar{\phi_2} \leftarrow \phi_2$
$\alpha_{\theta,\omega_i} \leftarrow T$
**for** each iteration **do**
  $s \leftarrow s_0$
  done $\leftarrow$ False
  Choose $\omega$ according to inter-option policy $\pi_\Omega$
  **for** each environment step **do**
    $a_t \sim \pi_{w_t,\theta}(a_t \mid s_t)$
    Take action $a_t$ and observe state $s_{t+1}, done$
    $D \leftarrow D \bigcup \{s_t, a_t, r(s_t, a_t), s_{t+1}, done\}$
    **if** $beta_{\omega_t,\upsilon}$ terminates in $s_{t+1}$ **then**
      choose new $\omega$ according to inter-option policy $\pi_\Omega$
    **end if**
  **end for**
  **for** each gradient step **do**
    $\theta \leftarrow \lambda_\theta \nabla_\theta J_\pi(\theta, \omega)$
    $\Omega \leftarrow \lambda_\Omega \nabla_\Omega J_\pi(\Omega)$
    $\phi_i \leftarrow \lambda_\phi \nabla_\phi J(\phi_i) \forall i \in \{1, 2\}$
    $\psi \leftarrow \lambda_\psi \nabla_\psi J(\psi)$
    $\alpha_{\theta,\omega_i} \leftarrow \lambda_{\alpha_{\theta,\omega_i}} \nabla_{\alpha_{\theta,\omega_i}} J(, \alpha_{\theta,\omega}) \forall i \in \{1, 2..|\omega|\}$
    $\upsilon \leftarrow \lambda_\upsilon \nabla_\upsilon J(\upsilon)$
    $\bar{\phi_i} \leftarrow \tau \phi_i \ + (1 - \tau) \bar{\phi_i} \forall i \in \{1, 2\}$ Update target network weights.
  **end for**
**end for**

**Algorithm 1:** Soft Option Critic

## 3 Experiments

The goal of our experiments is to compare the sample complexity of the proposed framework with vanilla option critic and other state of the art algorithms like PPO and SAC. We use bullet Roboschool Benchmark Suite (BulletPhysics (2015)) which is a port of OpenAI Roboschool environments (Dhariwal et al. (2017)). This suite consist of a set of robotic control tasks with continuous state and action space. The state space represents the joint information of the robots and the action space represents the torque applied at various joints. We evaluate the options framework on 5 robotic locomotion tasks - HopperBulletEnv-v0, HalfCheetahBulletEnv-v0, AntBulletEnv-v0, Walker2DBulletEnv-v0 and ReacherBulletEnv-v0. We also try to understand the effect of adding sparsity and variance regularization on the skills generated by the proposed framework.

## 4 Experimental Setup and Hyperparameters

We use the PPO baseline implementation (Dhariwal et al. (2017)) provided by OpenAI for our experiments. For Soft Actor Critic, we use the implementation (Haarnoja (2018)) provided by the author. We use the same hyperparameters as that provided by the authors of SAC and PPO in their original papers for Mujoco tasks. We implement a deep version of options-critic framework which uses advantage Actor Critic as its underlying RL algorithm. The architecture comprises of a inter-option Q-value network and a intra-option policy network. The Q-value network is implemented using a deep neural network with two hidden layers and tanh activation on the output of the first two layers. This network takes state vector appended with one hot encoded option as input and outputs the value function estimate of the option at given state. The policy network shares the same architecture as Q-value network but takes state vector appended with one hot encoded option as input and outputs the mean and log of standard deviation of the normal distribution over actions. The soft option critic is implemented using deep neural network with 2 hidden layers as function approximators for inter-option Q value function, intra-option Q-value function, inter-option policy network, option termination function and intra-option Q value function. The values of the hyperparameters used for options-critic and soft options critic is provided in the Appendix section.

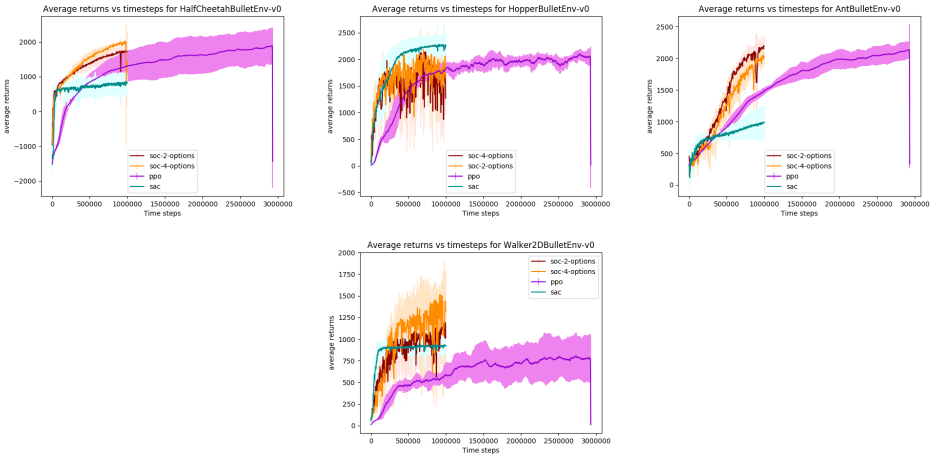


Figure 1: Average returns observed during training SOC and PPO framework

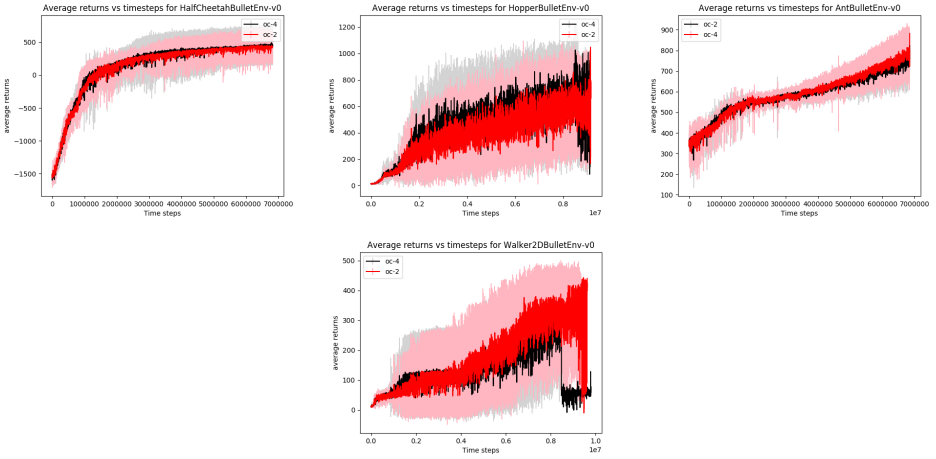


Figure 2: Average returns observed during training Option-critic framework

## 5 Evaluation

Figure 1 shows total average returns during training over 5 trials. The SOC framework, SAC and PPO are trained over 1 million steps and the policy is evaluated after every 2000 steps over 10 episodes. Results show that the SOC framework requires significantly less no of timesteps to achieve high returns as compared to options-critic framework. The framework also learns faster than soft actor critic and ppo in most tasks. Figure 2 shows that options-critic framework requires around 10 million steps to achieve the same order of returns as achieved by Soft Option critic in 1 million steps. Figure 3 shows a sample sequence of options used by SOC framework with and without regularization for solving the locomotion task in Ant environment after the training period. We observe temporal consistency in options selected by SOC framework with sparsity and variance regularization.

## 6 Conclusion

We have introduced a new off-policy variant of options-critic framework based on maximum entropy framework for solving complex control tasks with high dimensional state and action space. Our experiments show that this framework is significantly more sample efficient than options-critic and other baselines like PPO and SAC. We introduce additional sparsity and variance regularization in the options-critic framework to encourage state based specialization and diversity in options selected even in simple tasks. Our experiments demonstrate that these regularizations result in generating more interpretable options. We note that the weights for these regularizations require careful tuning in order to ensure

that it does not degrade performance. Alternatively, we can induce state based specialization of options by maximizing mutual information between state and option but we found that adding mutual information regularization penalty resulted in poor performance in practice. Hence, our future goal is to encourage sparsity and variance in options selected using constraints and automatically tune the weights for the same.

# 7 Appendix

**Lemma 1**
Consider the Bellman backup operator $T$ in Equation 6 and a mapping $Q^0 : S \times \Omega \times A \rightarrow R$ with $|A| < \infty$ and define $Q^{k+1} = TQ^k$. Then the sequence $Q^k$ will converge to the soft Q-value of $\pi$ as $k \rightarrow \infty$.
**Proof:**
We define the entropy augmented reward as

$$r_{\omega,\theta}(s_t, a_t) = r(s_t, a_t) + \mathbf{E}_{\mathbf{s_{t+1}} \sim \mathbf{p}}[H(\pi_{\omega,\theta}(\cdot \mid s_{t+1}))]$$

and rewrite the update rule as:

$$Q_U(w_t, s_t, a_t) = r_{\omega,\theta}(s_t, a_t) + \gamma \mathbf{E}_{\mathbf{s_{t+1}} \sim \mathbf{p}}[U(s_{t+1}, w_t)]$$

In the proof of intra-option q learning in Sutton et al 1998, it was proved that expected value of the update operator $r + \gamma U(s', w)$ yields a contraction ie-

$$[\mathbf{E}[r(s, a) + \gamma U(s', \omega)] - Q^*(s, w)] \leq \gamma \max_{s'', w''} \mid Q(s'', w'') - Q^*(s'', w'') \mid$$

Hence, we can define an evaluation operator $TQ$ over intra-option Q value function space $S \times \Omega \times A$, with similar metric as that used for $Q(s, \omega)$:

$$T(Q_U(w_t, s_t, a_t)) = r_{\omega,\theta}(s_t, a_t) + \gamma \mathbf{E}_{\mathbf{s_{t+1}} \sim \mathbf{p}}[U(s_{t+1}, w_t)]$$

Following analysis similar to that for $Q(s, \omega)$, we can show that the contraction holds.

# References


Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI*, 2016.

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *CoRR*, abs/1511.06297, 2015. URL `http://arxiv.org/abs/1511.06297`.

BulletPhysics. Bullet. `https://github.com/bulletphysics/bullet3`, 2015.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. Openai baselines. `https://github.com/openai/baselines`, 2017.

Tuomas Haarnoja. Soft actor critic. `https://github.com/haarnoja/sac`, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ArXiv*, abs/1801.01290, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *ArXiv*, abs/1812.05905, 2018b.

Peter Henderson, Wei-Di Chang, Pierre-Luc Bacon, David Meger, Joelle Pineau, and Doina Precup. Optiongan: Learning joint reward-policy options using generative adversarial inverse reinforcement learning. *CoRR*, abs/1709.06683, 2017. URL `http://arxiv.org/abs/1709.06683`.

Richard Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999a.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, 1999b.