A Learning Note on Near-Optimal Representation Learning For Hierarchical Reinforcement Learning

Chun-Yu Chai
Department of Computer Science
National Chiao Tung University
cychai.cs06g@nctu.edu.tw

1 Introduction

In this report, we address the problem of representation learning in goal-conditioned hierarchical reinforcement learning (HRL) from work by Nachum et al..

Among the hierarchical reinforcement learning problems, how to represent a high-level goal for a low-level policy to proceed is very crucial. A direct approach is to assign a goal state (or observation) to the low-level policy. The low-level policy is rewarded once it reaches closer to the assigned goal state.

For environments with low-dimensional observations, e.g., position of a target object or angular velocity of an inverted pendulum, we might be possible to learn a high-level policy to generate these states. However, for environments with high-dimensional observations, e.g., images, it's challenging to generate the exact goal state (observations) for these environments. Therefore, we can instead learn a representation to transform a high-level goal state into an embedding space and let the low-level policy reach the goal in this space. However, since the high-level policy is no longer communicating to the low-level policy by the full state, some performance loss may happen if the embedding space is not represented well.

In this work, the authors are trying to model this loss, namely the sub-optimality of a given representation. This measure aims to answer the question: How much does using the learned representation in place of the full representation cause us to lose, in terms of expected reward, against the optimal policy? We expect the learned representation to compress the state somehow to make the policy learning easier. However, we also care how lossy a learned representation is, not in terms of reconstruction, but in terms of the ability to represent near-optimal policies on top of this representation.

This paper's main contribution is the theoretical analysis of the bounded sub-optimality and an improved algorithm for better representation learning. The HRL framework is based on the previous work by Nachum et al. [2018] with temporal abstraction.

2 Problem Formulation

2.1 Two-level hierarchical reinforcement learning

We consider the HRL problem on MDP $\mathcal{M} = (S, A, R, T)$. A full hierarchical policy π_{hier} is a two-level policy that contains π_{hi} and π_{low} as the high-level policy and the low-level policy, respectively.

The high-level policy $\pi_{hi}(g|s_t)$ is a policy that generate a high-level goal g every fixed c steps (temporal abstraction). We denote $g_t \sim \pi_{hi}(g|s_t)$.

The low-level policy $\pi_{low}(a|s_t, g_t, s_{t+k}, k)$ is a goal-conditioned policy that translates a high-level goal into low-level actions $a_{t+k} \in A$ for every time step $k \in [0, c-1]$. The low policy at each time

Preprint. Under review.

step can be seen as a policy $\pi_{low,t}(\Psi(s_t, g_t), s_{t+k}, k)$, where $\Psi(s_t, g_t)$ is a mapping from high-level goal to low-level goal and fixed in the following c steps. k can be considered the time steps in the low-policy view.

For every c steps, the high-level policy chooses a goal that is expected to be reached in the next c steps by the low-level policy. This process is repeated until the episode is finished. Therefore, at time step t, the high-level policy is at s_t and generate a goal $g_t \sim \pi_{hi}(g_t|s_t)$. The low-level policy then generates actions $a_{t+k} \sim \pi_{low,t}(a|\Psi(s_t, g_t), s_{t+k}, k)$. The low-level policy is trained using a goal-conditioned reward: $-D(f(s_{t+k}), g_t)$, where D is a distance function on the transformed representation space defined by a function $f : S \to \mathbb{R}^d$

The HRL framework is shown in Figure 1.

We can see that the mapping function f affects the reward in each step, and thus affects the behavior of the low-level policy if we define the goal mapping function Ψ as

$$\Psi(s_t, g) = \arg\max_{\pi \in \Pi} \sum_{k=1}^{c} \gamma^{k-1} \mathbb{E}_{P_\pi(s_{t+k}|s_t)}[-D(f(s_{t+k}), g)], \tag{1}$$

where $P_\pi(s_{t+k}|s_t)$ is the probability of being in state s_{t+k} after following a low-level policy π for k steps starting from s_t.

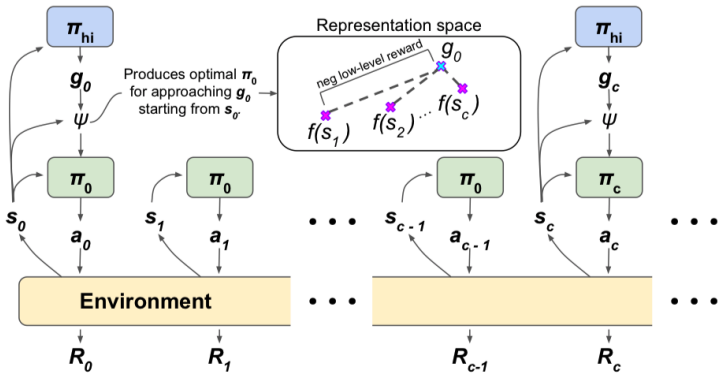

Figure 1: The HRL framework used in this paper.

2.2 Sub-Optimality

To quantify the performance loss on a learned policy due to the representation learning, we first assume that there exists some optimal high-level policy with and without the goal mapping Ψ. The notation is summarized in Table 1.

	Optimal high-level policy	Low-level policy	Full(hierarchical) policy
with Ψ	$\pi^*_{hi}(g_t|s_t, \Psi)$	$\pi_{low}(\Psi, s_{t+k}, k)$	π^*_{hier}
without Ψ	$\pi^{**}_{hi}(\pi|s_t)$	$\pi_{low} \sim \pi^{**}_{hi}(\pi|s_t)$	π^*

Table 1: Notation for policies.

We then introduce the sub-optimality in terms of the state value difference w.r.t. the form of Ψ.

$$\text{SubOpt}(\Psi) = \sup_{s \in S} V^{\pi^*}(s) - V^{\pi^*_{hier}}(s) \tag{2}$$

The state values $V^{\pi^*_{hier}}$ are determined by the form of Ψ which is in turn determined by the choice of representation f. However, it is still unclear how a change in f will result in a change to the sub-optimality. The contribution of this paper is to translate this sub-optimality of Ψ to a practical representation learning objective for f.

3 Theoretical Analysis

In this section, we provide the main theoretical result of this paper. The goal is to connect the policy optimization objective (Equation1) and representation learning objective for f.

3.1 Single-steps ($c = 1$) and deterministic policies

We first present the restricted case $c = 1$(no temporal abstraction) with deterministic lower-level policies. At each time step, the high-level policy produces a new goal g_t and induces a low-level deterministic policy $a = \pi_t = \Psi(s_t, g_t)$ for some $a \in A$. We introduce the notation $\tilde{g} = \varphi(s, a)$ to represent a inverse goal mapping function, which aims to predict which goal $\tilde{g}$ will cause Ψ to yield an action $\tilde{a} = \Psi(s, \tilde{g})$ that induces a next state distribution $P(s'|s, \tilde{a})$ similar to $P(s'|s, a)$.

φ is a model that produces the goal representation for a low-level policy. If φ is a good mapping, then we expect the total variation divergence between $P(s'|s, \tilde{a})$ and $P(s'|s, a)$ should be small. This idea leads to Theorem 1.

Theorem 1. *If there exists $\varphi : S \times A \to G$ such that,*

$$\sup_{s \in S, a \in A} D_{TV}(P(s' \mid s, a) \| P(s' \mid s, \Psi(s, \varphi(s, a)))) \leq \epsilon \tag{3}$$

then SubOpt$(\Psi) \leq C\epsilon$, where $C = \frac{2\gamma}{(1-\gamma)^2} R_{max}$, $R_{max} = \sup_s |R(s)|$

Theorem 1 allows us to bound the sub-optimality of Ψ in terms of how recoverable the effect of any action in A is, in terms of transition to the next state. If φ is exactly the inverse of Ψ, such that $\Psi(s, \varphi(s, a)) = a$ for all s, a, then the sub-optimality of Ψ is 0. Since an invertible Ψ may not be feasible to learn, we can introduce another distribution $K(s'|s, a)$ to model the dynamics and lead to bounded sub-optimality.

Claim 1. *Let $\rho(s)$ be a prior and f, φ be so that, for $K(s' \mid s, a) \propto \rho(s') \exp(-D(f(s'), \varphi(s, a)))$*

$$\sup_{s \in S, a \in A} D_{KL}(P(s' \mid s, a) || K(s' \mid s, a)) \leq \epsilon^2 / 8. \tag{4}$$

If the low-level objective is defined as

$$\Psi(s, g) = \arg\max_{a \in A} \mathbb{E}_{P(s'|s,a)}[-D(f(s'), g) + \log \rho(s') - \log P(s' \mid s, a)] \tag{5}$$

then the sub-optimality of Ψ is bounded by $C\epsilon$

$K(s'|s, a)$ in Equation4 can be interpreted as a dynamics model determined by f and φ. By bounding the difference between the true dynamics $P(s'|s, a)$ and the dynamics $K(s'|s, a)$ implied by f and φ, Equation4 states that the representation f should be chosen in such a way that dynamics in representation space are roughly given by $\varphi(s, a)$

3.2 Temporal abstraction($c \geq 1$) and general policies

For cases with $c \geq 1$, a high-level goal g_t is selected by the high-level policy every c steps, and the corresponding low-level actions $a_{t+k} \sim \pi(a|\Psi(s_t, g_t), s_{t+k}, k)$, for $k = [0, c-1]$. The inverse goal model is revised as $\tilde{g} = \varphi(s_t, \pi)$, which aims to predict which goal $\tilde{g}$ will cause Ψ to yield a policy $\tilde{\pi} = \Psi(s_t, \tilde{g})$ that induces future state distributions $P_{\tilde{\pi}}(s_{t+k} \mid s_t)$ similar to $P_\pi(s_{t+k} \mid s_t)$ for $k \in [1, c]$. The general form to Theorem 1 is as follows:

Theorem 2. *Consider a mapping $\varphi : S \times \Pi \to G$ and define $\epsilon_k : S \times \Pi \to \mathbb{R}$ for $k \in [1, c]$ as,*

$$\epsilon_k(s_t, \pi) = D_{\mathrm{TV}}\left(P_\pi(s_{t+k} \mid s_t) \| P_{\Psi(s_t, \varphi(s_t, \pi))}(s_{t+k} \mid s_t)\right). \tag{6}$$

If

$$\sup_{s_t \in S, \pi \in \Pi} \frac{1}{\bar{w}} \sum_{k=1}^{c} \gamma^{k-1} w_k \epsilon_k(s_t, \pi) \leq \epsilon \tag{7}$$

then SubOpt$(\Psi) \leq C\epsilon$, where $C = \frac{2\gamma}{1-\gamma^c} R_{\max} \bar{w}$

The divergence is weighted by $w_k = 1$ for $k < c$ and $w_k = (1-\gamma)^{-1}$ for $k = c$. $\bar{w} = \sum_{k=1}^{c} \gamma^{k-1} w_k$.

The analogue to Claim 1 simply replacing the single-step KL divergence with weighted sum of multi-step KL divergence.

Claim 2. *Let $\rho(s)$ be a prior over S. Let f, φ be such that*

$$\sup_{s_t \in S, \pi \in \Pi} \frac{1}{\bar{w}} \sum_{k=1}^{c} \gamma^{k-1} w_k D_{\mathrm{KL}} \left(P_\pi \left(s_{t+k} \mid s_t \right) || K \left(s_{t+k} \mid s_t, \pi \right) \right) \leq \epsilon^2 / 8, \tag{8}$$

where $K(s_{t+k} \mid s_t, \pi) \propto \rho(s_{t+k}) \exp(-D(f(s_{t+k}), \varphi(s_t, \pi)))$.

If the low-level objective is defined as

$$\Psi \left(s_t, g \right) = \arg\max_{\pi \in \Pi} \sum_{k=1}^{c} \gamma^{k-1} w_k \mathbb{E}_{P_\pi(s_{t+k}|s_t)} \left[-D \left(f \left(s_{t+k} \right), g \right) + \log \rho \left(s_{t+k} \right) - \log P_\pi \left(s_{t+k} \mid s_t \right) \right], \tag{9}$$

then the sub-optimality of Ψ is bounded by $C\epsilon$.

Claim 2 is the main theoretical contribution of this paper. If the low-level objective is defined as in Equation 9, then minimizing the sub-optimality can be equivalently done by optimizing a representation learning objective based on Equation 8. We should leverage Equation 8 for our learning as a way to minimize the sub-optimality. [1]

3.3 Practical usage for learning

We now turn our attention on Equation 8. Two separate networks $f_{\theta_1}, \varphi_{\theta_2}$ are used to represent the state representation function $f_\theta : S \to \mathbb{R}^d$ and inverse goal mapping function $\varphi_\theta : S \times \Pi \to \mathbb{R}^d$. The learning parameters are $\theta = [\theta_1, \theta_2]$. The low-level policy π used in Equation 8 is approximated by uniformly sampling a s_t from the replay buffer, which stores low-level action sequences that last c steps, and using $a_{t:t+c-1}$ to represent the low-level policy given s_t. The learning objective corresponding to the Equation 8 is thus,

$$J(\theta) = \mathbb{E}_{s_t, a_{t:t+c-1} \sim \text{ replay}} \left[J \left(\theta, s_t, a_{t:t+c-1} \right) \right], \tag{10}$$

where

$$\begin{aligned} J \left(\theta, s_t, \pi \right) &= J \left(\theta, s_t, a_{t:t+c-1} \right) \\ &= \sum_{k=1}^{c} \gamma^{k-1} w_k D_{\mathrm{KL}} \left(P_\pi \left(s_{t+k} \mid s_t \right) || K_\theta \left(s_{t+k} \mid s_t, \pi \right) \right) \\ &= B + \sum_{k=1}^{c} -\gamma^{k-1} w_k \mathbb{E}_{P_\pi(s_{t+k}|s_t)} \left[\log K_\theta \left(s_{t+k} \mid s_t, \pi \right) \right] \\ &= B + \sum_{k=1}^{c} -\gamma^{k-1} w_k \mathbb{E}_{P_\pi(s_{t+k}|s_t)} \left[\log E_\theta \left(s_{t+k}, s_t, \pi \right) \right] + \gamma^{k-1} w_k \log \mathbb{E}_{\tilde{s} \sim \rho} \left[E_\theta \left(\tilde{s}, s_t, \pi \right) \right], \end{aligned} \tag{11}$$

where $E_\theta \left(s', s, \pi \right) = \exp \left(-D \left(f_\theta \left(s' \right), \varphi_\theta(s, \pi) \right) \right)$ and B is a constant. Note that $K(s_{t+k}|s_t, \pi)$ can be written as

$$K(s_{t+k}|s_t, \pi) = \frac{1}{\int \rho \left(s_{t+k} \right) E_\theta \left(s_{t+k}, s_t, \pi \right) ds_{t+k}} \rho \left(s_{t+k} \right) E_\theta \left(s_{t+k}, s_t, \pi \right). \tag{12}$$

$\nabla_\theta J(\theta, s_t, \pi)$ is therefore

$$\sum_{k=1}^{c} -\gamma^{k-1} w_k \mathbb{E}_{P_\pi(s_{t+k}|s_t)} \left[\nabla_\theta \log E_\theta \left(s_{t+k}, s_t, \pi \right) \right] + \gamma^{k-1} w_k \frac{\mathbb{E}_{\tilde{s} \sim \rho} \left[\nabla_\theta E_\theta \left(\tilde{s}, s_t, \pi \right) \right]}{\mathbb{E}_{\tilde{s} \sim \rho} \left[E_\theta \left(\tilde{s}, s_t, \pi \right) \right]}. \tag{13}$$

[1] The proofs of Theorem 2 and Claim 2 are attached in the original paper. They are not presented here since I cannot justify well on some intermediate steps.

We treat ρ to be the replay buffer distribution. Therefore all terms can be approximated by sampling a batch of experience from the replay buffer. The authors suggest that the denominator of the second term can be estimated as

$$\mathbb{E}_{\tilde{s}\sim\rho}\left[E_\theta\left(\tilde{s}, s_t, \pi\right)\right] \approx |\widetilde{S}|^{-1} \sum_{\tilde{s}\in\widetilde{S}} E_\theta\left(\tilde{s}, s_t, \pi\right) \tag{14}$$

with a batch of size S from the replay buffer to get better estimate of this normalization term of K.

To leverage the result in Equation 9 for learning the low-level policy, we can treat the inner terms as the low-level reward with weights outside the expectation:

$$r_{t+k} = w_k(-D\left(f\left(s_{t+k}\right), g\right) + \log \rho\left(s_{t+k}\right) - \log P_\pi\left(s_{t+k} \mid s_t\right)). \tag{15}$$

However, the second and last term in Equation 15 are generally unknown. The authors proposed to use $K(s_{t+k}|s_t, \pi)$ to approximate $P_\pi(s_{t+k}|s_t)$ since we have optimized on Equation 8 (we expect the KL divergence of this two distributions should be small). Therefore,

$$\begin{aligned}
r_{t+k} &= w_k(-D(f(s_{t+k}), g) + \log \rho(s_{t+k}) - \log P_\pi(s_{t+k} \mid s_t)) \\
&\approx w_k(-D(f(s_{t+k}), g) + \log \rho(s_{t+k}) - \log K_\pi(s_{t+k} \mid s_t, \pi)) \\
&= w_k(-D(f(s_{t+k}), g) + \log \rho(s_{t+k}) - \log \rho(s_{t+k}) + \log D(f(s_{t+k}), \varphi(s_t, \pi) + \log \mathbb{E}_{\tilde{s}\sim\rho}[E(\tilde{s}, s_t, \pi)]) \\
&= w_k(-D(f(s_{t+k}), g) + \log D(f(s_{t+k}), \varphi(s_t, \pi)) + \log \mathbb{E}_{\tilde{s}\sim\rho}[E(\tilde{s}, s_t, \pi)])
\end{aligned} \tag{16}$$

Now we can use this approximated reward to train the low-level policy. The full algorithm is shown in Figure 2.

Algorithm 1 Representation learning for hierarchical RL.

Input: Replay buffer $\mathcal{D}$, number of training steps N, batch size B, parameterizations $f_\theta, \varphi_\theta, \pi_{\text{hi}}^\phi, \pi_{\text{lo}}^\phi$

function $EstLogPart(\widetilde{S}, s_t, a_{t:t+c-1})$
Equation 14
Return $\log \frac{1}{|\widetilde{S}|} \sum_{\tilde{s}\in\widetilde{S}} \exp(-D(f_\theta(\tilde{s}), \varphi_\theta(s_t, a_{t:t+c-1})))$.
end function

function $CompLowReward(g, s_t, s', a_{t:t+c-1}, L)$
Equation 16
Return $-D(f_\theta(s'), g) + D(f_\theta(s'), \varphi_\theta(s_t, a_{t:t+c-1})) + L$.
end function

function $CompReprLoss(s_t, s', a_{t:t+c-1}, \tilde{s}, L)$
Equation 13
Compute attractive term $J_{\text{att}} = D(f_\theta(s'), \varphi_\theta(s_t, a_{t:t+c-1}))$.
Compute repulsive term $J_{\text{rep}} = \exp(-D(f_\theta(\tilde{s}), \varphi_\theta(s_t, a_{t:t+c-1})) - \text{stopgrad}(L))$.
Return $J_{\text{att}} + J_{\text{rep}}$.
end function

for $T = 1$ **to** N **do**

Sample experience g, s, a, r, s' and add to replay buffer $\mathcal{D}$ (Nachum et al., 2018).
Sample batch of c-step transitions $\{(g^{(i)}, s_{t:t+c}^{(i)}, a_{t:t+c-1}^{(i)}, r_{t:t+c-1}^{(i)})\}_{i=1}^B \sim \mathcal{D}$.
Sample indices into transition: $\{k^{(i)}\}_{i=1}^B \sim [1, c]^B$.
Sample batch of states $\widetilde{S} = \{\tilde{s}^{(i)}\}_{i=1}^B \sim \mathcal{D}$.
Estimate log-partitions $\{L^{(i)}\}_{i=1}^B = \{EstLogPart(\widetilde{S}, s_t^{(i)}, a_{t:t+c-1}^{(i)})\}_{i=1}^B$.

// Reinforcement learning
Compute low-level rewards $\{\tilde{r}^{(i)}\}_{i=1}^B = \{CompLowReward(g^{(i)}, s_t^{(i)}, s_{t+k^{(i)}}^{(i)}, a_{t:t+c-1}^{(i)}, L^{(i)})\}_{i=1}^B$.
Update π_{lo}^ϕ (Nachum et al., 2018) with experience $\{(g^{(i)}, s_{t+k^{(i)}-1}^{(i)}, a_{t+k^{(i)}-1}^{(i)}, \tilde{r}^{(i)}, s_{t+k^{(i)}}^{(i)})\}_{i=1}^B$.
Update π_{hi}^ϕ (Nachum et al., 2018) with experience $\{(g^{(i)}, s_{t:t+c}^{(i)}, a_{t:t+c-1}^{(i)}, r_{t:t+c-1}^{(i)})\}_{i=1}^B$.

// Representation learning
Compute loss $J = \frac{1}{B}\sum_{i=1}^B w_{k^{(i)}} \gamma^{k^{(i)}-1} CompReprLoss(k^{(i)}, s_t^{(i)}, s_{t+k^{(i)}}^{(i)}, a_{t:t+c-1}^{(i)}, \tilde{s}^{(i)}, L^{(i)})$.

Update θ based on $\nabla_\theta J$.

end for

Figure 2: The full HRL algorithm with representation learning.

4 Conclusion

This paper is interesting, but the proofs are elegant and not easy. I spent much time to understand their ideas and proofs. Many approximation tricks are used in the final learning algorithm. I think HRL is a very interesting direction for research, especially in environments with only sparse reward signals. The goal mapping and state representation are the keys to communicating with high-level and low-level policies. However, under so many approximations in the implementations, I will also doubt that if the state space is very large (e.g., images), some approximations, e.g., Equation 14 may not perform as expected without very large $|S|$. (Evaluation tasks in this paper are all environments with low dimensional state space).

References

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Near-optimal representation learning for hierarchical reinforcement learning. In *ICLR*.

Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *NIPS*, pages 3303–3313, 2018.