1 Motivation

1. RL tasks can be hard to solved because
2. Solutions are unlikely to be found when no prior knowledge is available
3. Solution space is dominated by local minima
4. Properties of the behaviors are not captured by the reward functions
5. Previous approaches to deal with these issues rely on transferring between tasks, which include mechanisms to extract knowledge about the structure of existing solutions and injecting this knowledge into a RL solution
6. Two main ideas: hierarchical policies and a learnable default policy

2 Background on RL as probabilistic modeling and Notation

2.1 Notation

State and action at time t are s_t, a_t
r(s,a) is the reward
History up to time t is $x_t = (s_1, a_1, \ldots, s_t)$
The whole trajectory is $\tau = (s_1, a_1, s_2, a_2, \ldots)$
$\pi(a_t|x_t)$ is the agent's policy
$\pi_0(a_t|x_t)$ denotes a default policy
$KL(Y|X)$ denotes $E_{\pi(Y|X)}[\log \frac{\pi(Y|X)}{\pi_0(Y|X)}] = KL(\pi(.|X)||\pi_0(.|X))$

2.2 RL as probabilistic modeling

1. The KL-regularized RL objective is

$$\mathcal{L}(\pi, \pi_0) = \mathbb{E}_\tau \left[\sum_{t \geq 1} \gamma^t r(s_t, a_t) - \alpha \gamma^t \mathrm{KL}(a_t|x_t) \right]$$

   where

$$\mathrm{KL}(a_t|x_t) = \mathbb{E}_{\pi(a_t|x_t)} \left[\log \frac{\pi(a_t|x_t)}{\pi_0(a_t|x_t)} \right]$$

2. This objective can be optimized with respect to only π or jointly for both π and π_0
3. If the objective is optimized with respect to the default policy, then

$$\pi_0^*(a_t|x_t) = \arg\max_{\pi_0} \mathbb{E}_{\pi(a|x_t)} [\log \pi_0(a|x_t)]$$

4. Thus, learning π_0 can be seen as supervised learning where π_0 is trained to match the history-conditional action sequence produced by π

3 Hierarchically structured policies

1. The controller is decomposed into a high-level policy π^H and a low-level policy π^L.
2. Intuitively, the high-level policy should be concerned with task objectives but largely agnostic to details of low-level actuation. The opposite applies for the low-level policy.
3. Conceptually, the high and low level policies interact via a latent variable z.

4. Let z_t be a latent variable for each timestep. The controller is formalized as $\pi(a_t, z_t|x_t) = \pi^H(z_t|x_t)\,\pi^L(a_t|z_t, x_t)$ and likewise for the default policy.

5. If z_t is continuous, then KL-regularized RL objective becomes intractable as the marginal distribution $\pi(a_t|x_t)$ and $\pi_0(a_t|x_t)$ can not be computed in closed form.

6. Assume the latent variable for both π and π_0 have the same dimension and semantics, we can construct an upper bound for the KL

$$\mathrm{KL}(a_t|x_t) \leq \mathrm{KL}(z_t|x_t) + \mathbb{E}_{\pi(z_t|x_t)}[\mathrm{KL}(a_t|z_t, x_t)]$$

where

$$\mathrm{KL}(z_t|x_t) = \mathrm{KL}\left(\pi^H(z_t|x_t) \,\|\pi_0^H(z_t|x_t)\right)$$
$$\mathrm{KL}(a_t|z_t, x_t) = \mathrm{KL}\left(\pi^L(a_t|z_t, x_t) \,\|\pi_0^L(a_t|z_t, x_t)\right)$$

7. The resulting lower bound for the objective is

$$\mathcal{L}(\pi, \pi_0) \geq \mathbb{E}_\tau\left[\sum_{t\geq 1}\gamma^t r(s_t, a_t) - \alpha\gamma^t\,\mathrm{KL}(z_t|x_t) - \alpha\gamma^t\mathrm{KL}(a_t|z_t, x_t)\right]$$

8. This is the main objective that is optimized in this paper.

3.1 Sharing low-level controllers

1. The paper argues that an advantage of the hierarchical structure of the policies is that it enables parameter sharing, which when used in appropriate context, can make learning more statistically efficient.

2. They consider sharing the low-level policies in both the low-level and high-level policies, that is:

$$\pi^L(a_t|z_t, x_t) = \pi_0^L(a_t|z_t, x_t)$$

3. This results in a new lower bound for the objective function

$$\mathcal{L}(\pi, \pi_0) \geq \mathbb{E}_\tau\left[\sum_{t\geq 1}\gamma^t r(s_t, a_t) - \alpha\gamma^t\,\mathrm{KL}(z_t|x_t)\right]$$

3.2 Regularizing via information asymmetries

1. Restricting the information available to different policies is a powerful tool to force regularization and generalization.

2. Generally, they provide full information to only π^H and partial information to π^L (body-specific (proprioceptive) and some environment-related (exteroceptive) information).

3. This is so that the task is only known to pi^H and makes it easier to transfer π^L across tasks.

4. They also limit the information available to π_0^H

5. In experiments where they consider transferring the high-level policy across bodies (the abstract tasks remain the same but the body changes), they hide the body-specific information from π^H to force it to learn body-agnostic behavior.

3.3 Parameterizing the default policies

1. Independent isotropic Gaussian $\pi_0^H(z_t|x_t) = \mathcal{N}(z_t|0, 1)$

2. AR(1) process $\pi_0^H(z_t|x_t) = \mathcal{N}\left(z_t|\alpha z_{t-1}, \sqrt{1-\alpha^2}\right)$. α is chosen to ensure a normal marginal distribution.

3. Learned AR prior $\pi_0^H(z_t|x_t) = \mathcal{N}\left(z_t|\mu_\phi(z_{t-1}), \sigma_\phi^2(z_{t-1})\right)$

4 Algorithm

1. The proposed method has different instantiations based on design choices.
2. If π_0^H has learnable parameters or π_0^L is not shared, then they jointly optimize the default policy and the agent's policy.
3. The agent's policy is regularized by a target default policy, which is updated at an interval to match the default policy.
4. To optimize the hierarchical policy, they reparameterize $z_t \sim \pi^H(z_t|x_t)$ as $z_t = f^H(x_t, \epsilon_t)$, where $\epsilon_t \sim \rho(\epsilon_t)$ is a fixed noise distribution and f is a deterministic function.
5. In practice, this means that $\pi(a_t|\epsilon_t, x_t) = \pi^L\left(a_t|f^H(x_t, \epsilon_t), x_t\right)$

5 Experiments

5.1 Bodies

1. Ball: a body with 2 actuators for moving forward-backward and turning left-right
2. Ant: a body with 4 legs and 8 actuators
3. Quadruped: a body with 4 legs and 12 actuators
4. Each body is characterized by a different set of proprioceptive features

5.2 Tasks

1. Go to 1 of K targets: the agent receives a sparse reward upon reaching one of K locations.
2. Move K boxes to K targets: the goal is to move one of K boxes to one of K targets, whose positions are randomized, as indicated by the environments.
3. Move heavier boxes: same as above but the boxes are heavier.
4. Gather boxes: the agent needs to move two boxes such that they are in contact with each other.
5. Move box and go to target: the agent is required to move the box to a target and then go to a different target in a single episode.

5.3 Learning from scratch

1. They demonstrate that in the learning from scratch setting, the KL regularized objective significantly speed up learning and the hierarchical structure provides an advantage over a flat formulation.
2. They also demonstrate that sharing the parameter for the low-level policies between the agent's policy and the default policy speeds up learning.

6 Transfer Learning

6.1 Task Transfer

1. They consider transfer between task distributions whose solutions exhibit significant shared structure.
2. If the default policy can capture and transfer this structure, it will facilitate learning similar tasks.
3. In this setting, they re-use the pretrained goal agnostic components, including the high level default policy π_0^H and the lower level default policy π_0^L
4. They learn a new high level controller π^H and optionally a new low level controller π^L.

5. They demonstrate experimentally that transferring the pre-trained default policy brings significant benefits when learning related tasks

6. The hierarchical structure which facilitates parameter-sharing performs better than a non-hierarchical baseline.

6.2 Body Transfer

1. In this setting, the task distribution does not change, but the body changes

2. They focus on transferring the high-level structure of a goal-specific behavior

3. They transfer the high-level controller π^H and the high-level default policy π_0^H

4. They learn a new body-specific low-level controller π^L, which is assumed to share parameters with the low-level default controller.

5. They hypothesize that the main challenge of transferring task knowledge across bodies is how to interpret the latent actions.

6. The high level controller provides abstract instructions on how to solve a task through the latent variable z, but it is unclear how the instruction should be instantiated on the new body.

7. Rewarding the low-level controller to learn the semantics of the latent code is problematic, because the true label of the semantics is not available.

8. To address this challenge, they optimize the following lower bound during transfer, which provides a dense reward signal based on the KL divergence.

$$\mathcal{L}(\pi, \pi_0) \geq \mathbb{E}_\tau \left[\sum_{t \geq 1} \gamma^t r(s_t, a_t) - \alpha \gamma^t \operatorname{KL}(z_t | x_t) \right]$$

9. The negative KL encourages the high level controller to be close to the default high level controller.

10. They interpret this as: the negative KL is an intrinsic reward to encourage the low level controller to executed the latent action properly so that the high level controller can behave in a habitual way.