1 Preliminaries

1.1 Notations and definitions

1. The setting is transfer learning RL, with separate pre-training tasks and transfer tasks.
2. An agent is trained from scratch on the pre-training tasks, but it may then apply any skills learned during pre-training to the subsequent transfer tasks.
3. Their objective is to use the pre-training tasks to acquire a set of reusable skills.
4. Each task is defined by its state space, action space, dynamics model, goal space $g \in \mathcal{G}$, goal distribution $g \sim p(g)$ and a reward function $r_t = r\left(s_t, a_t, g\right)$.
5. All tasks they consider share the state, action space and dynamics model.
6. The different between tasks include the goal space, goal distribution and reward function.
7. The objective is expected return over the distribution of goals from the task $J(\pi) = \mathbb{E}_{g \sim p(g), \tau \sim p_\pi(\tau|g)}\left[\sum_{t=0}^{T} \gamma^t r_t\right]$ where $p_\pi(\tau|g) = p\left(s_0\right) \prod_{t=0}^{T-1} p\left(s_{t+1}|s_t, a_t\right) \pi\left(a_t|s_t, g\right)$.

1.2 Hierarchical policies

1. Hierarchical policies are a common model for reusing and composing previously learned skills.
2. One approach is to use a mixture-of-experts model:
3. The composite's policy action distribution $\pi(a|s,g)$ is a weighted sum of a set of primitives $\pi_i(a|s,g)$.
4. A gating function determines the weights w:

$$\pi(a|s,g) = \sum_{i=1}^{k} w_i(s,g) \pi_i(a|s,g), \quad \sum_{i=1}^{k} w_i(s,g) = 1, \quad w_i(s,g) \geq 0$$

5. This is referred to as a additive model.
6. To sample from the composite policy, a primitive π_i is first selected according to w, then an action is sampled from the primitive's distribution. (Note: this does not seem to agree with the gating function formulation).
7. The limitation is that the additive model can only sample one primitive at a time.
8. They hypothesize that this works for system with low-degree of freedom, but as the complexity of the task grows, the agent might need to use multiple primitive at once (person speaking, walking at the same time as an example).
9. Also, the primitives can be combined, leading to combinatorial explosion, which the additive model does not handle.

2 Multiplicative Compositional Policies

1. The idea is to factor the agent's behavior not with respect to time (one primitive at a time), but to factor it with respect to the action space.
2.

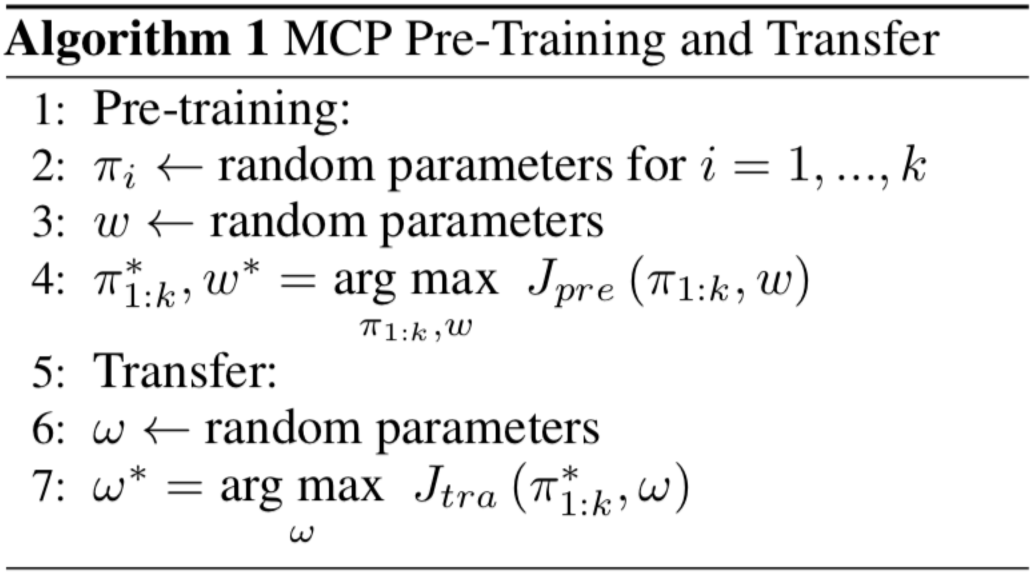
$$\pi(a|s,g) = \frac{1}{Z(s,g)} \prod_{i=1}^{k} \pi_i(a|s,g)^{w_i(s,g)}, \quad w_i(s,g) \geq 0$$

3. Each primitive is modeled using a diagonal Gaussian. The composite policy is thus also a Gaussian.

2.1 Pre-training and transfer

1. The same set of primitives is used to solve all pre-training tasks, which results in a collection of primitives that captures the range of behaviors needed to solve the sets of tasks.

2.



Algorithm 1 MCP Pre-Training and Transfer

```
Pre-training:
$\pi_i \leftarrow$ random parameters for $i = 1, ..., k$
$w \leftarrow$ random parameters
$\pi^*_{1:k}, w^* = \arg\max_{\pi_{1:k}, w} \; J_{pre}\left(\pi_{1:k}, w\right)$
Transfer:
$\omega \leftarrow$ random parameters
$\omega^* = \arg\max_{\omega} \; J_{tra}\left(\pi^*_{1:k}, \omega\right)$
```

3. During pre-training, in order to force the primitive to specialize in distince skills, they use an asymmetric model, where only the gating function receives the goal as input and the primitive only receives the state

$$\pi(a|s,g) = \frac{1}{Z(s,g)} \prod_{i=1}^{k} \pi_i(a|s)^{w_i(s,g)}, \quad \pi_i(a|s) = \mathcal{N}\left(\mu_i(s), \Sigma_i(s)\right)$$

4. They hypothesize that the asymmetric model prevents the degeneracy of a single primitive becoming responsible for all goals, and instead encourages the primitives to learn distinct skills.

5. Also, since the primitive depends only on states, they can be transferred to tasks with a new goal space.

3 Experiments

3.1 Pre-training tasks

1. The pre-training tasks consists of imitating reference motions.

2. Each reference motions specifies a sequence of target states $\{\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_T\}$ that the agent should track at each timestep.

3. A single composite policy is trained to imitate different motions.

4. To imitate multiple motions, the goal $g_t = (\hat{s}_{t+1}, \hat{s}_{t+2})$ provides the policy with the target states for the next two timesteps.

5. To encourage the policy to learn to transition between different motions, the reference motion is also switched randomly to another motion within each episode.

6. The reference motion consists of walking and turning motions.