

Hierarchical RL and Skill Discovery

CS 330

The Plan

Information-theoretic concepts

Skill discovery

Using discovered skills

Hierarchical RL

Why Skill Discovery?

What if we want to discover interesting behaviors?



[The construction of movement by the spinal cord,
Tresch et al., 1999]

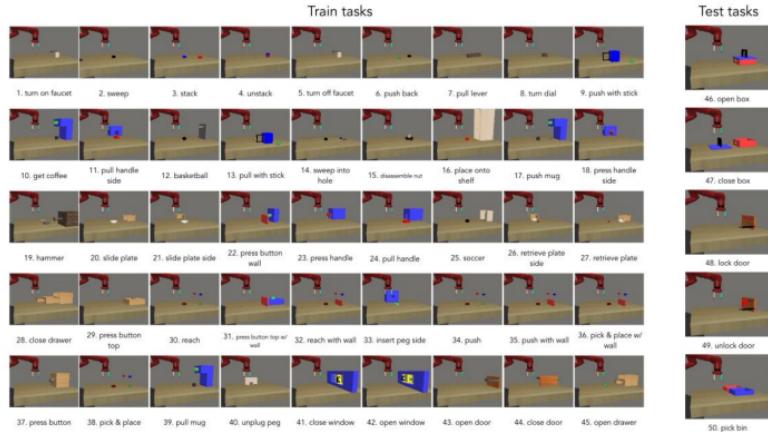


[Postural hand synergies for tool use, *Santello, et al., 1998*]

Why Skill Discovery? More practical version

Coming up with tasks is tricky...

Write down task ideas for a tabletop manipulation scenario



Why Hierarchical RL?

Performing tasks at various levels of abstractions

Exploration

Bake a cheesecake

Buy ingredients

Go to the store

Walk to the door

Take a step

Contract muscle X



The Plan

Information-theoretic concepts

Skill discovery

Using discovered skills

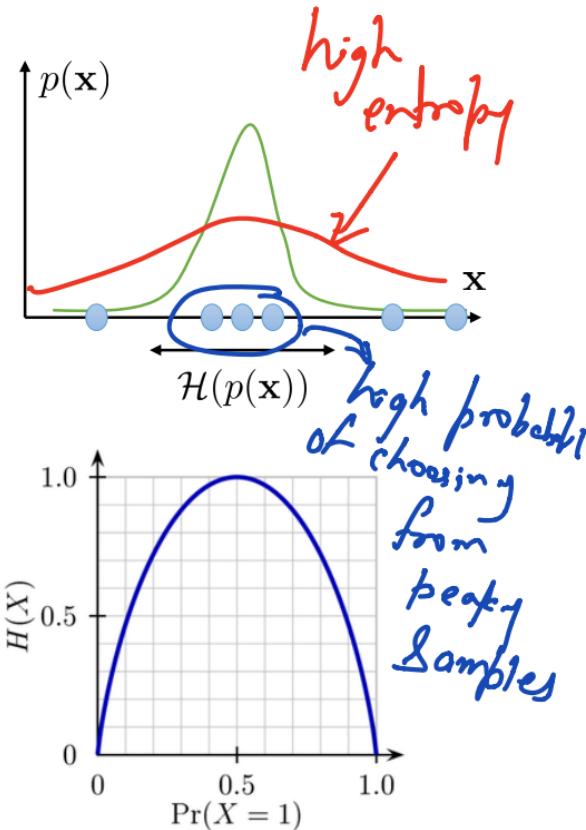
Hierarchical RL

Entropy

$p(\mathbf{x})$ distribution (e.g., over observations \mathbf{x})

$$\mathcal{H}(p(\mathbf{x})) = -E_{\mathbf{x} \sim p(\mathbf{x})}[\log p(\mathbf{x})]$$

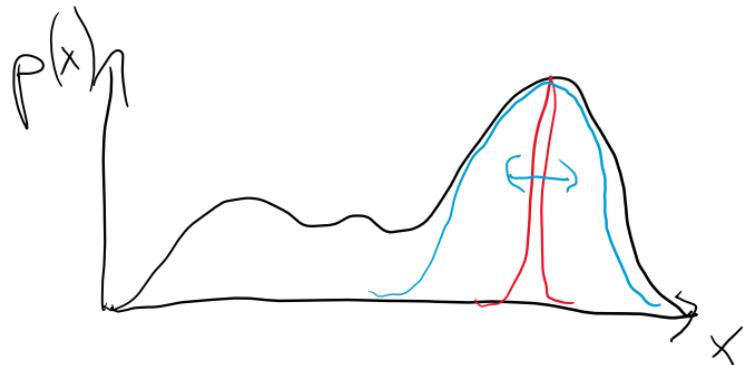
entropy – how “broad” $p(\mathbf{x})$ is



KL-divergence

Distance between two distributions

$$\mathbb{D}_{KL}(q||p) = \mathbb{E}_q \left[\log \frac{q(x)}{p(x)} \right] = \mathbb{E}_q \log q(x) - \mathbb{E}_q \log p(x) = \underbrace{-\mathbb{E}_q \log p(x)}_{\text{red}} - \underbrace{\mathcal{H}(q(x))}_{\text{blue}}$$



Mutual information

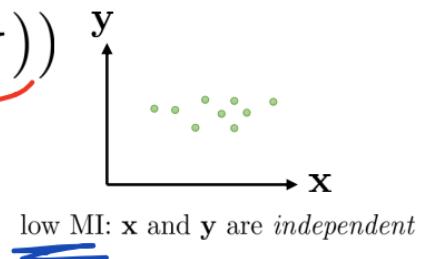
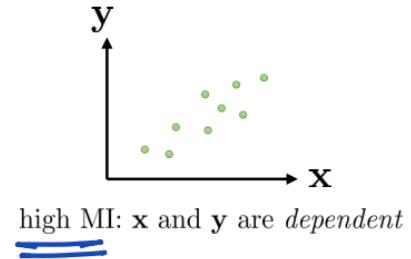
$$\begin{aligned}\mathcal{I}(\mathbf{x}; \mathbf{y}) &= D_{\text{KL}}(p(\mathbf{x}, \mathbf{y}) \| p(\mathbf{x})p(\mathbf{y})) \\ &= E_{(\mathbf{x}, \mathbf{y}) \sim p(\mathbf{x}, \mathbf{y})} \left[\log \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})p(\mathbf{y})} \right]\end{aligned}$$

$$= \mathcal{H}(p(\mathbf{y})) - \mathcal{H}(p(\mathbf{y}|\mathbf{x})) = \underbrace{\mathcal{H}(p(\mathbf{x}))}_{\text{high}} - \underbrace{\mathcal{H}(p(\mathbf{x}|\mathbf{y}))}_{\text{low}}$$

High MI?

x- it rains tomorrow, y – streets are wet tomorrow

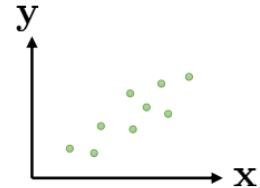
x- it rains tomorrow, y – we find life on Mars tomorrow



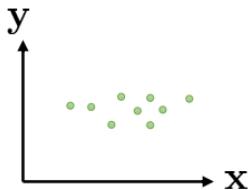
Mutual information

$$\begin{aligned}\mathcal{I}(\mathbf{x}; \mathbf{y}) &= D_{\text{KL}}(p(\mathbf{x}, \mathbf{y}) \| p(\mathbf{x})p(\mathbf{y})) \\ &= E_{(\mathbf{x}, \mathbf{y}) \sim p(\mathbf{x}, \mathbf{y})} \left[\log \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})p(\mathbf{y})} \right]\end{aligned}$$

$$= \mathcal{H}(p(\mathbf{y})) - \mathcal{H}(p(\mathbf{y}|\mathbf{x})) = \mathcal{H}(p(\mathbf{x})) - \mathcal{H}(p(\mathbf{x}|\mathbf{y}))$$



high MI: x and y are *dependent*



low MI: x and y are *independent*

example of mutual information: “empowerment” (Polani et al.)

$$\mathcal{I}(\mathbf{s}_{t+1}; \mathbf{a}_t) = \mathcal{H}(\mathbf{s}_{t+1}) - \mathcal{H}(\mathbf{s}_{t+1}|\mathbf{a}_t) \rightarrow \text{RL setting}$$

empowerment will be high if action, a_t , makes

an impact on the next state s_{t+1}

The Plan

Information-theoretic concepts

Skill discovery

Using discovered skills

Hierarchical RL

Soft Q-learning

Objective:

$$\sum_t E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q} [r(\mathbf{s}_t, \mathbf{a}_t) + \mathcal{H}(q(\mathbf{a}_t | \mathbf{s}_t))]$$

Value-, Q-functions, and the policy

$$V_t(\mathbf{s}_t) = \log \int \exp(Q_t(\mathbf{s}_t, \mathbf{a}_t)) \mathbf{a}_t$$

$$Q_t(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + E[(V_{t+1}(\mathbf{s}_{t+1})]$$

$$\pi(\mathbf{a}_t | \mathbf{s}_t) = \exp(Q_t(\mathbf{s}_t, \mathbf{a}_t) - V_t(\mathbf{s}_t)) = \exp(A_t(\mathbf{s}_t, \mathbf{a}_t))$$



Q-learning

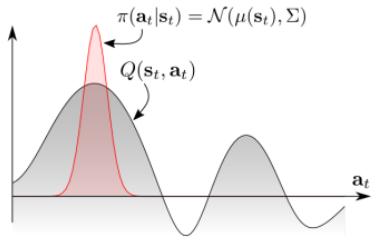
- 1. collect dataset $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$
- 2. set $\mathbf{y}_i \leftarrow r(\mathbf{s}_i, \mathbf{a}_i) + \gamma \max_{\mathbf{a}'_i} Q_\phi(\mathbf{s}'_i, \mathbf{a}'_i)$
- 3. set $\phi \leftarrow \arg \min_\phi \frac{1}{2} \sum_i \|Q_\phi(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{y}_i\|^2$

$$\pi(\mathbf{a} | \mathbf{s}) = \operatorname{argmax}_{\mathbf{a}} Q_\phi(\mathbf{s}, \mathbf{a})$$

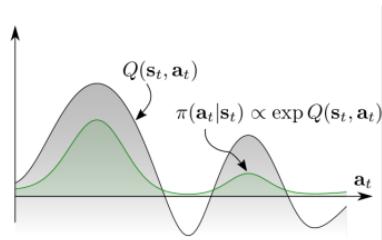
Soft Q-learning

- 1. collect dataset $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$
 - 2. set $\mathbf{y}_i \leftarrow r(\mathbf{s}_i, \mathbf{a}_i) + \gamma \max_{\mathbf{a}'_i} Q_\phi(\mathbf{s}'_i, \mathbf{a}'_i)$
 - 3. set $\phi \leftarrow \arg \min_\phi \frac{1}{2} \sum_i \|Q_\phi(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{y}_i\|^2$
- $\pi(\mathbf{a} | \mathbf{s}) = \operatorname{argmax}_{\mathbf{a}} Q_\phi(\mathbf{s}, \mathbf{a}) \exp(A_t(s_t, a_t))$

Soft Q-learning



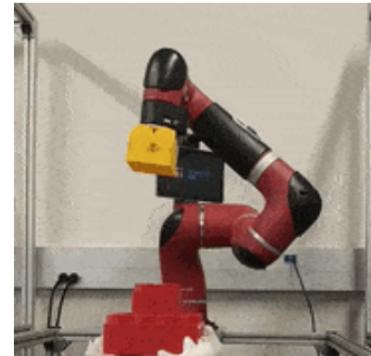
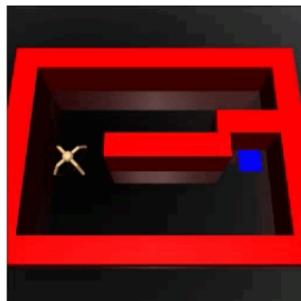
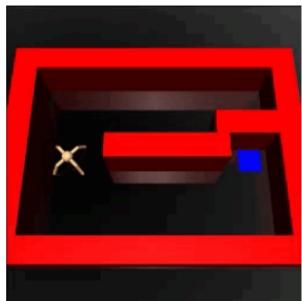
Exploration



Fine-tunability



Robustness



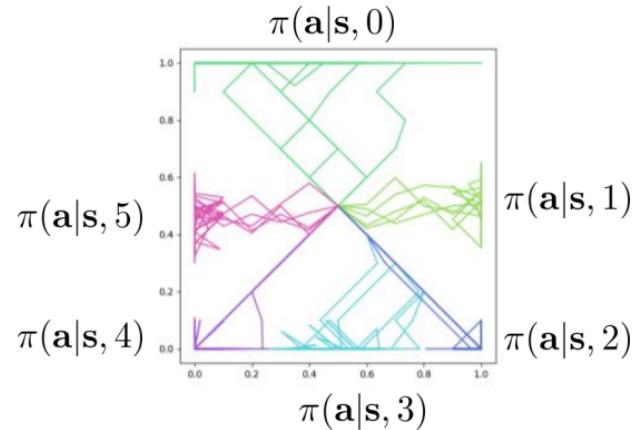
Learning diverse skills

$$\pi(\mathbf{a}|\mathbf{s}, z)$$

↑
task index

Why can't we just use MaxEnt RL

1. **action** entropy is not the same as **state** entropy
agent can take very different actions, but land in similar states
2. MaxEnt policies are stochastic, but not always **controllable**
intuitively, we want **low** diversity for a fixed z , high diversity *across* z 's



Intuition: different **skills** should visit different **state-space regions**

Diversity-promoting reward function

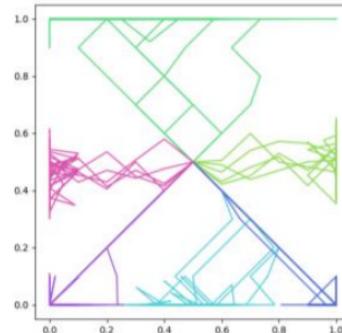
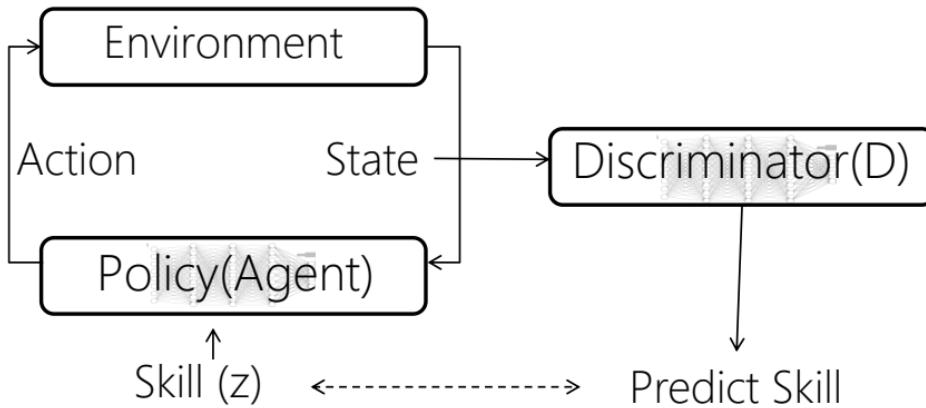
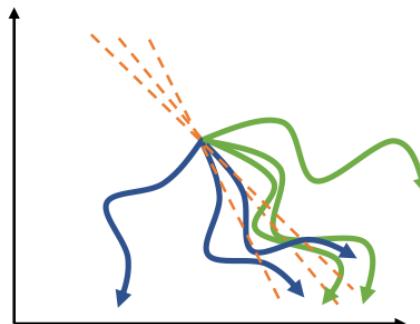
$$\pi(\mathbf{a}|\mathbf{s}, z) = \arg \max_{\pi} \sum_z E_{\mathbf{s} \sim \pi(\mathbf{s}|z)} [r(\mathbf{s}, z)]$$



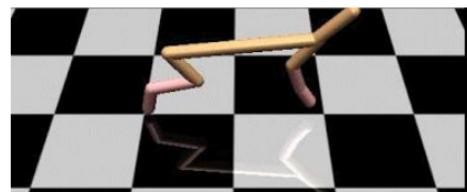
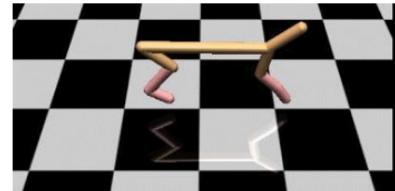
reward states that are unlikely for other $z' \neq z$

$$r(\mathbf{s}, z) = \log p(z|\mathbf{s})$$

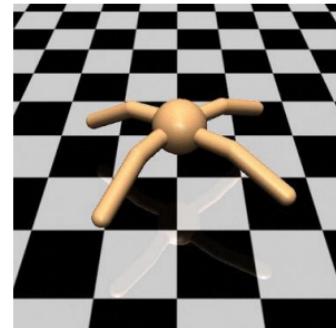
← classifier



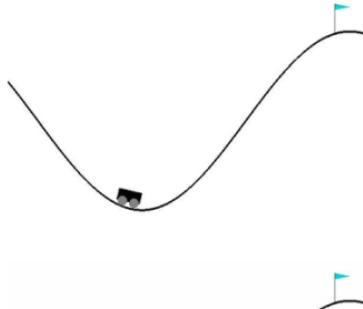
Examples of learned tasks



Cheetah



Ant



Mountain car

A connection to mutual information

$$\pi(\mathbf{a}|\mathbf{s}, z) = \arg \max_{\pi} \sum_z E_{\mathbf{s} \sim \pi(\mathbf{s}|z)}[r(\mathbf{s}, z)]$$

$$r(\mathbf{s}, z) = \log p(z|\mathbf{s})$$

$$I(z, \mathbf{s}) = H(z) - H(z|\mathbf{s})$$

maximized by using uniform prior $p(z)$

minimized by maximizing $\log p(z|\mathbf{s})$

The Plan

Information-theoretic concepts

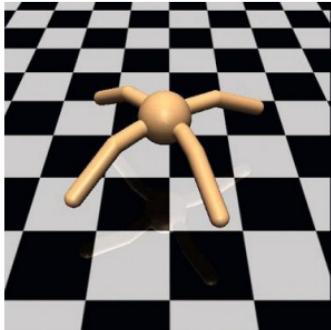
Skill discovery

Using discovered skills

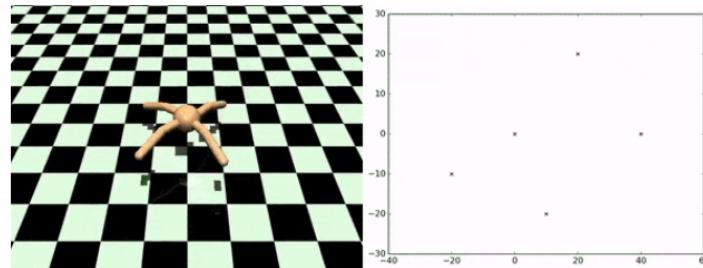
Hierarchical RL

How to use learned skills?

$$\pi(\mathbf{a}|\mathbf{s}, z)$$

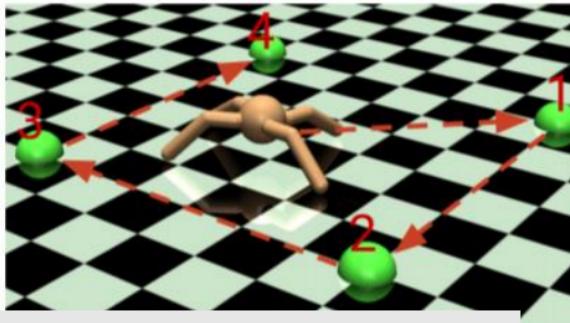
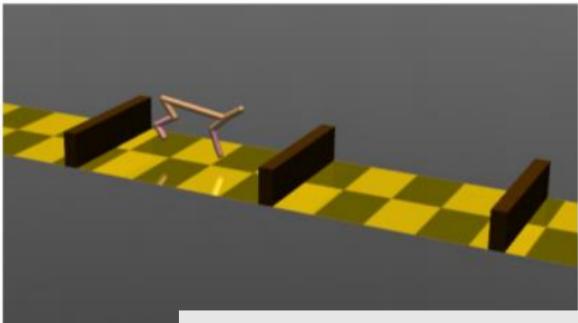


How can we use the learned skills to accomplish a task?

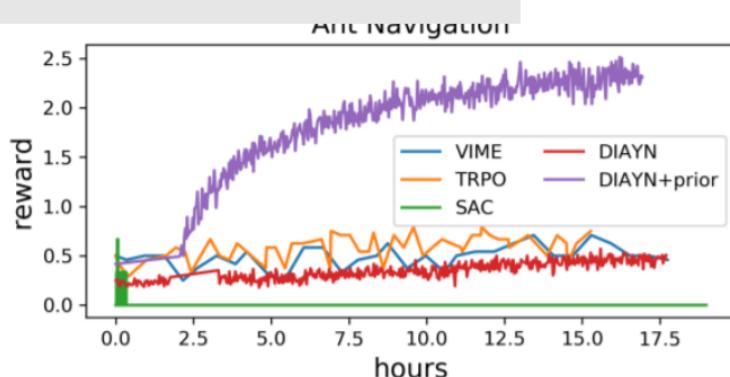
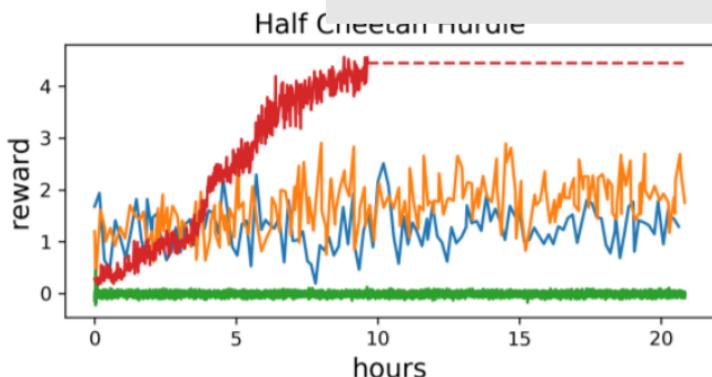


Learn a policy that operates on z 's

Results: hierarchical RL

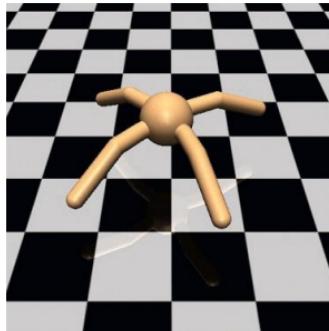
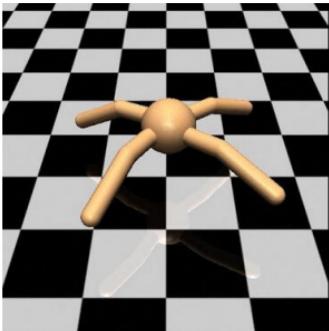


Can we do better?

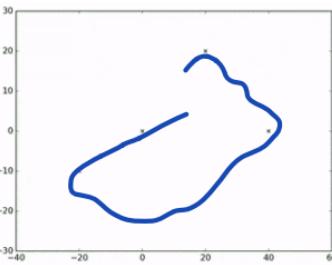
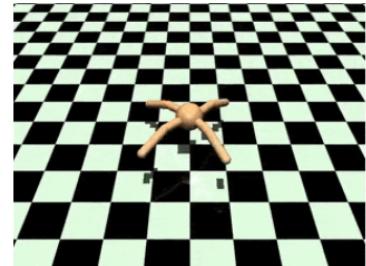


What's the problem?

Skills might not be particularly useful



It's not very easy to use the learned skills

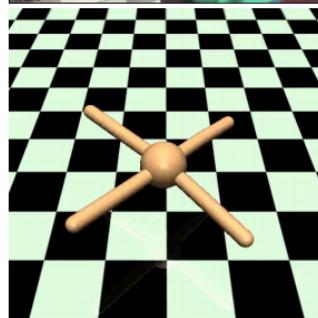


What makes a useful skill?

What's the problem?

Non-useful
skills

Consequences
are hard to
predict



Useful
skills

Consequences
are easy to
predict

Slightly different mutual information

Conditioned
mutual information

$$I(z, s) = H(z) - H(z|s)$$

$$\max \mathcal{I}(s', z | s) = \max \left(\mathcal{H}(s' | s) - \mathcal{H}(s' | s, z) \right)$$

$$\mathcal{I}(\mathbf{x}, \mathbf{y} | z) = D_{\text{KL}}(p(\mathbf{x}, \mathbf{y}) || p(\mathbf{x})p(\mathbf{y}))$$

Future hard to predict for different skills

Predictable future for a given skill

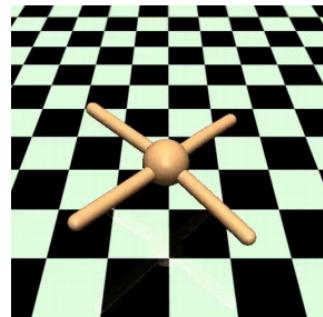
$$\begin{aligned} I(s'; z | s) &\geq \mathbb{E}_s \mathbb{E}_z \mathbb{E}_{p(s'|s, z)} [\log \frac{q_\phi(s'|s, z)}{p(s'|s)}] \\ &\approx \mathbb{E}_s \mathbb{E}_z \mathbb{E}_{p(s'|s, z)} [\log \frac{q_\phi(s'|s, z)}{\sum_{i=1}^L q_\phi(s'|s, z_i)} + \log L] \end{aligned}$$

Skill-dynamics model

We are learning a skill-dynamics model $q(s' | s, z)$

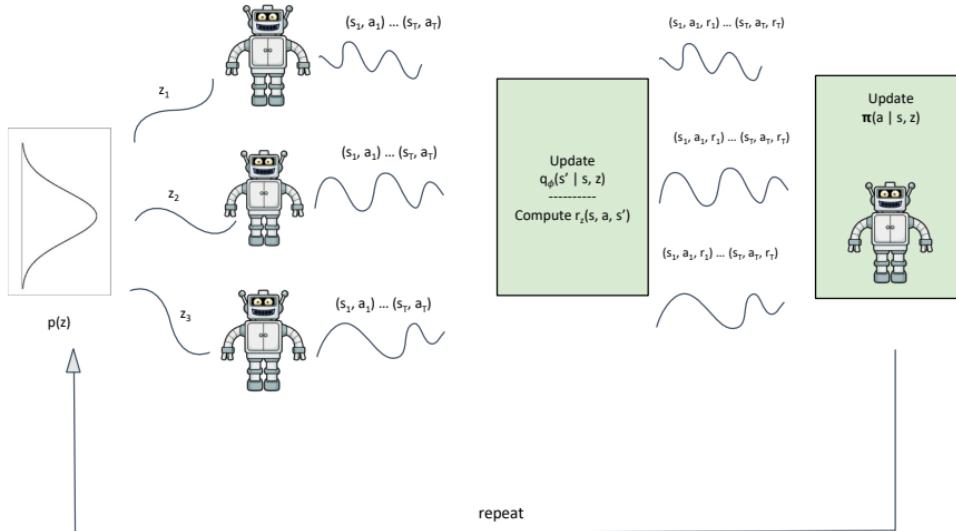
compared to conventional global dynamics $p(s' | s, a)$

Skills are optimized specifically to make skill-dynamics easier to model



skills

DADS algorithm



Algorithm 1: Dynamics-Aware Discovery of Skills (DADS)

Initialize π, q_ϕ ;

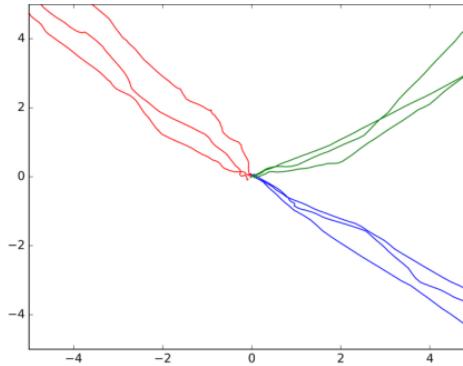
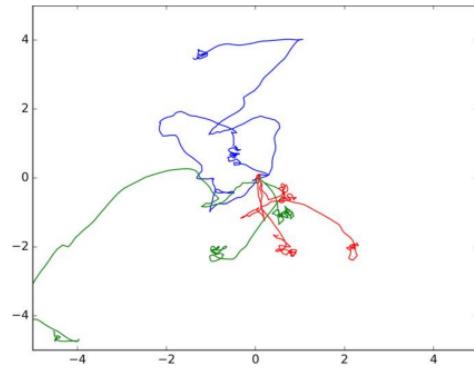
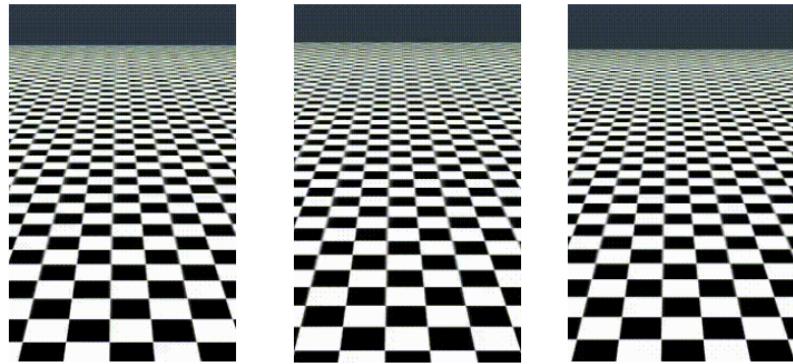
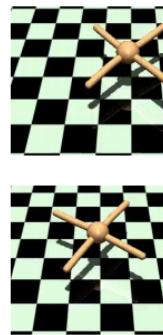
while not converged **do**

 Sample a skill $z \sim p(z)$ every episode;
 Collect new M on-policy samples;
 Update q_ϕ using K_1 steps of gradient descent on M transitions;

 Compute $r_z(s, a, s')$ for M transitions;
 Update π using any RL algorithm;

end

DADS results

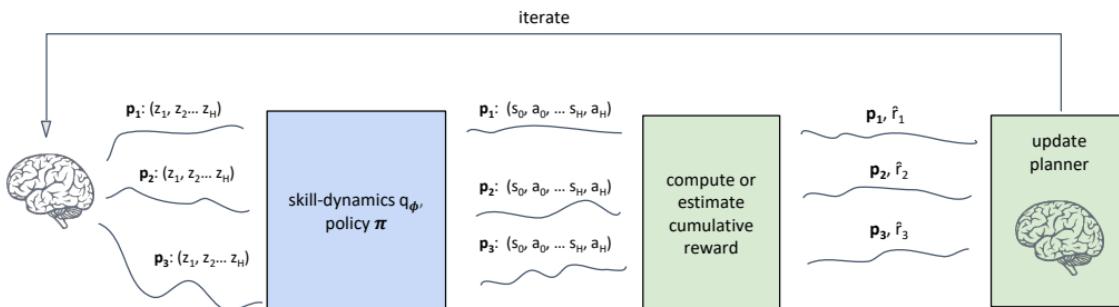
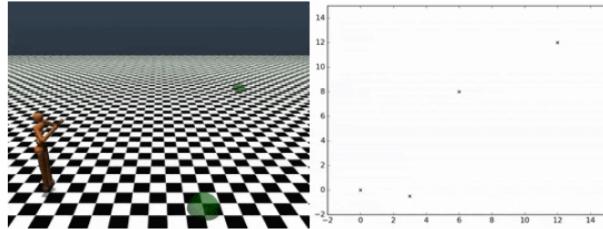
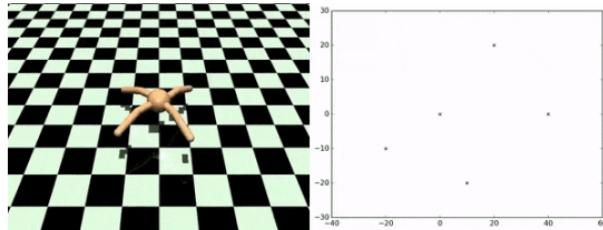


Using learned skills

Use skill-dynamics for model-based planning

Plan for skills not actions

Tasks can be learned zero-shot



Summary

- Two skill discovery algorithms that use mutual information
- Predictability can be used as a proxy for "usefulness"
- Method that optimizes for both, predictability and diversity
- Model-based planning in the skill space
- Opens new avenues such as unsupervised meta-RL
 - Gupta et al. *Unsupervised Meta-Learning for RL*, 2018



The Plan

Information-theoretic concepts

Skill discovery

Using discovered skills

Hierarchical RL

Why Hierarchical RL?

Performing tasks at various levels of abstractions

Exploration

Bake a cheesecake

Buy ingredients

Go to the store

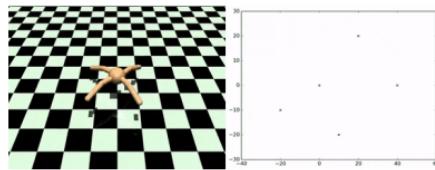
Walk to the door

Take a step

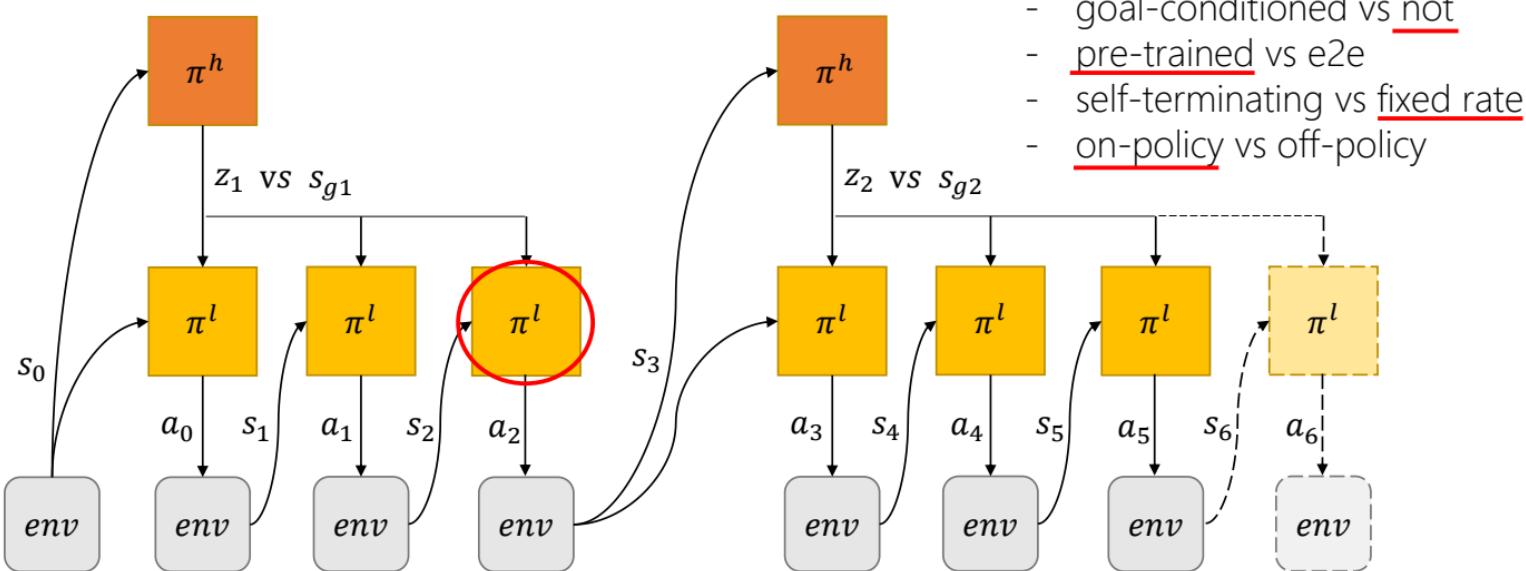
Contract muscle X



Hierarchical RL – design choices



Design choices:



Learning Locomotor Controllers

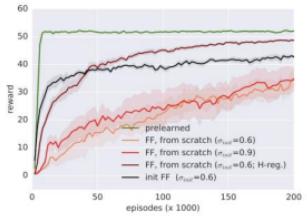
Command updated
every K steps

High-level
controller

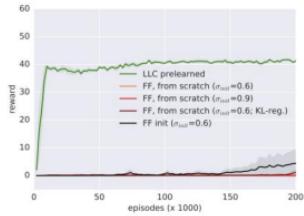
Low-level
controller

- HL and LL trained separately
- Trained with policy gradients
- Hierarchical noise

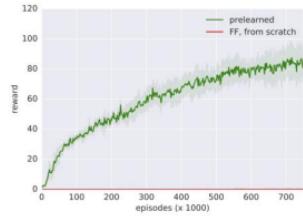
inform



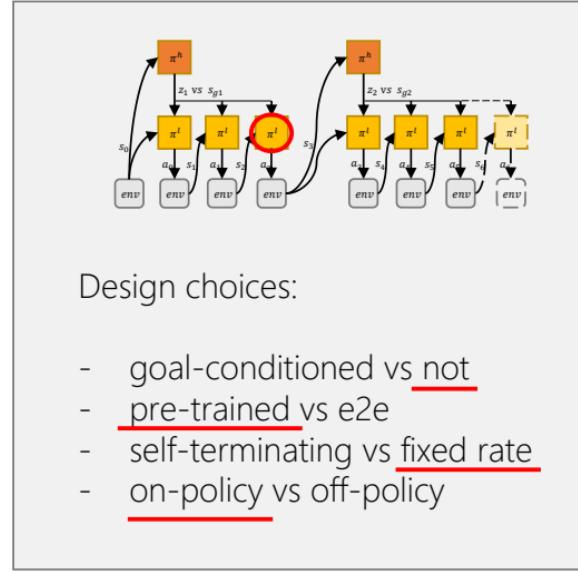
(a) target-seek (easy)



(b) target-seek (hard)



(c) soccer

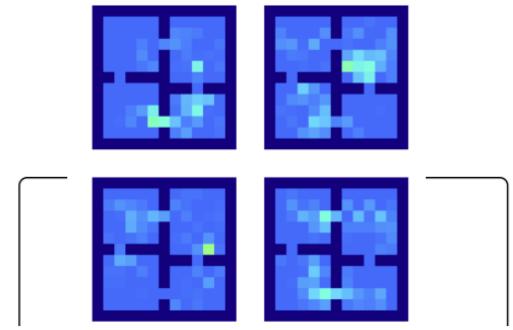


Design choices:

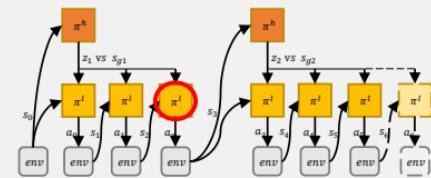
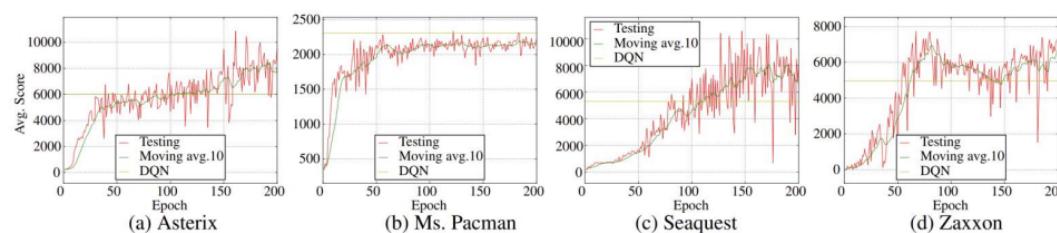
- goal-conditioned vs not
- pre-trained vs e2e
- self-terminating vs fixed rate
- on-policy vs off-policy

Option Critic

A Markovian option $\omega \in \Omega$ is a triple $(\mathcal{I}_\omega, \pi_\omega, \beta_\omega)$ in which $\mathcal{I}_\omega \subseteq \mathcal{S}$ is an initiation set, π_ω is an *intra-option* policy, and $\beta_\omega : \mathcal{S} \rightarrow [0, 1]$ is a termination function. We also assume that $\forall s \in \mathcal{S}, \forall \omega \in \Omega : s \in \mathcal{I}_\omega$ (i.e., all options are available everywhere)



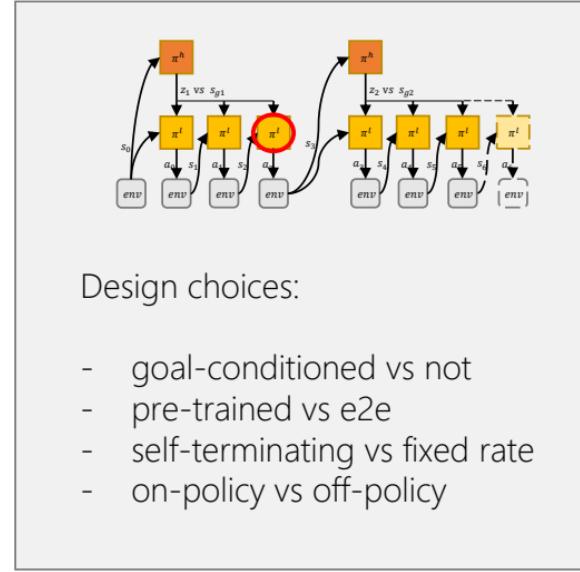
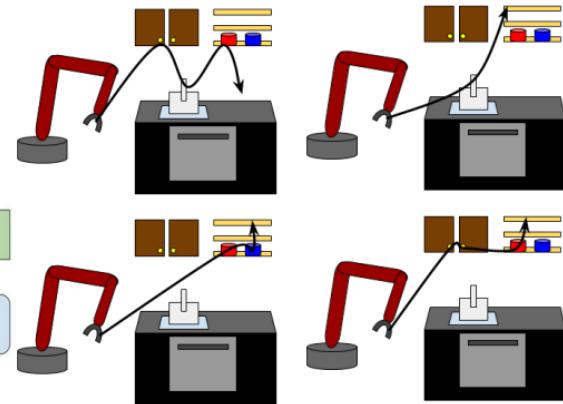
- Option is a self-terminating mini-policy
 - Everything trained together with policy gradient



Design choices:

- goal-conditioned vs not
 - pre-trained vs e2e
 - self-terminating vs fixed rate
 - on-policy vs off-policy

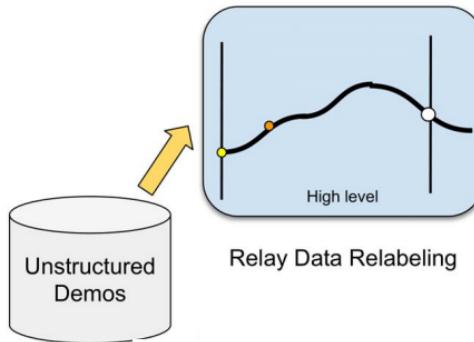
Relay Policy Learning



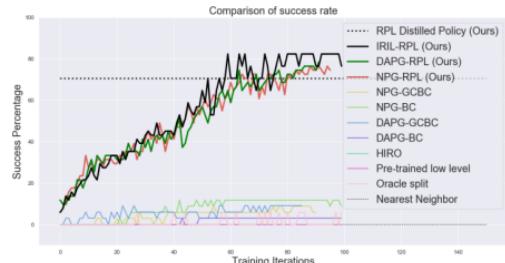
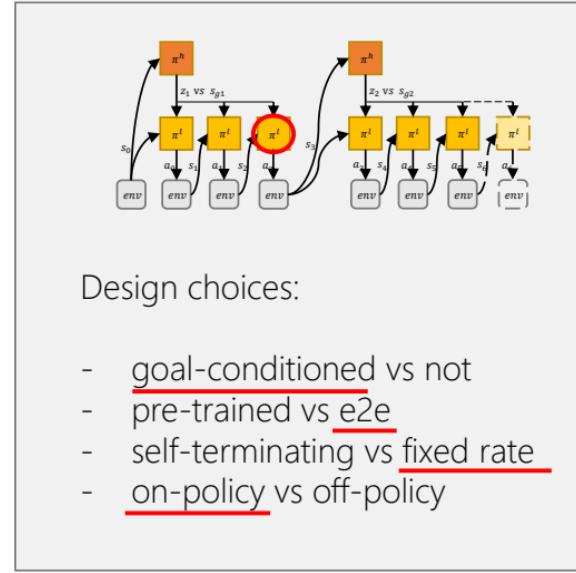
Design choices:

- goal-conditioned vs not
- pre-trained vs e2e
- self-terminating vs fixed rate
- on-policy vs off-policy

Relay Policy Learning



- Goal-conditioned policies with relabeling
- Demonstrations to pre-train everything
- On-policy



HIRO

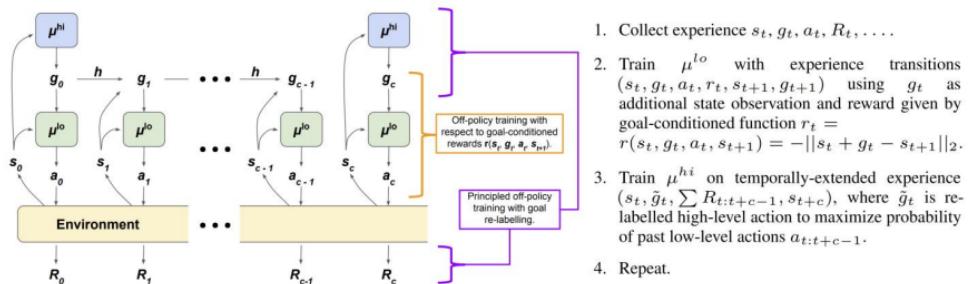
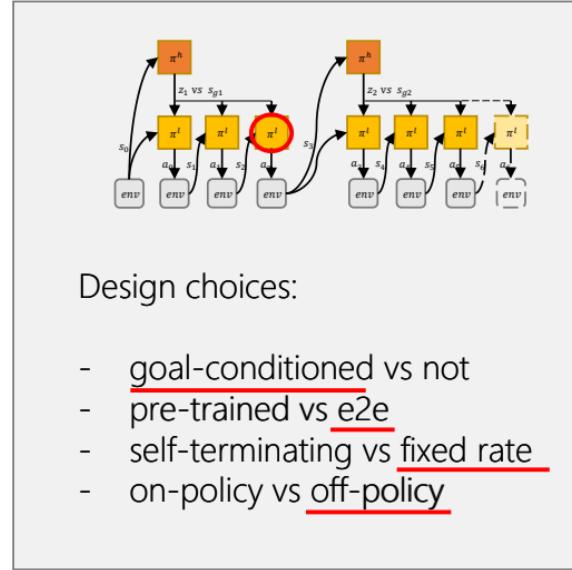
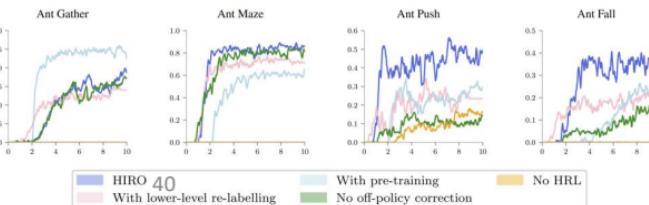


Figure 2: The design and basic training of HIRO. The lower-level policy interacts directly with the environment. The higher-level policy instructs the lower-level policy via high-level actions, or goals, $g_t \in \mathbb{R}^{d_g}$ which it samples anew every c steps. On intermediate steps, a fixed goal transition function h determines the next step's goal. The goal simply instructs the lower-level policy to reach specific states, which allows the lower-level policy to easily learn from prior off-policy experience.



- Goal-conditioned policies with relabeling
- Off-policy training through off-policy corrections

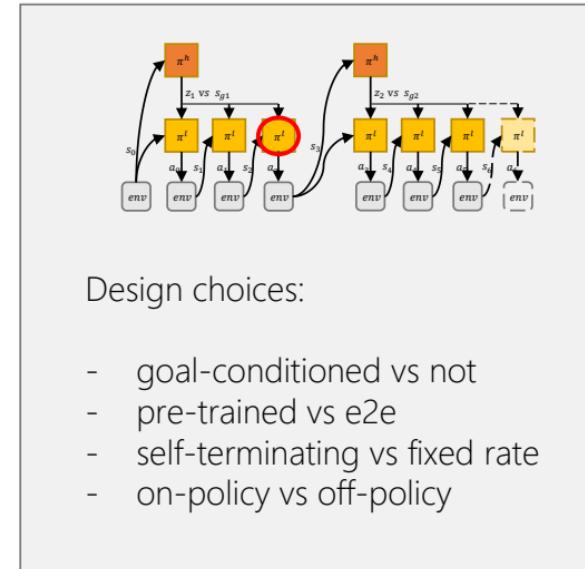


HRL Summary

- Multiple design choices and frameworks
- Helps with exploration and temporally extended tasks
- Can be difficult to get it to work
- Seems like a natural direction for harder RL problems

Hypothesis	Experiments	Important?
(H1) Temporal training	Figures 2, 3	Yes, but only for the use of multi-step rewards (n -step returns).
(H2) Temporal exploration	Figures 2, 4	Yes, and this is important even for non-hierarchical exploration.
(H3) Semantic training	Figure 3	No.
(H4) Semantic exploration	Figure 4	Yes, and this is important even for non-hierarchical exploration.

Figure 5: A summary of our conclusions on the benefits of hierarchy.



Design choices:

- goal-conditioned vs not
- pre-trained vs e2e
- self-terminating vs fixed rate
- on-policy vs off-policy

The Plan

Information-theoretic concepts

Skill discovery

Using discovered skills

Hierarchical RL

Next Week

Can the agent learn continuously over their life-time?

Lifelong learning – Nov 4