Computer Engineering Department

Reinforcement Learning:
Unsupervised RL (Skill Discovery) and Hierarchical RL, Goal-Conditioned RL

Mohammad Hossein Rohban, Ph.D.

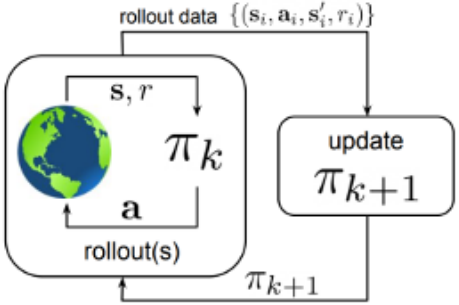
Hosein Hasani

Spring 2023

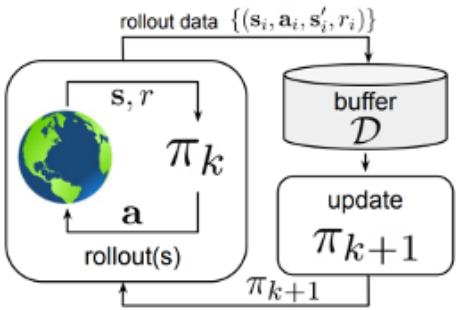
Courtesy: Most of slides are adopted from CS 330 Stanford.

Recall: RL so far

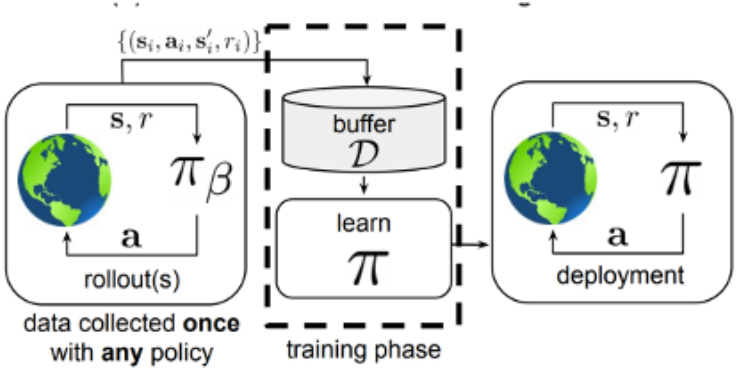
Online Reinforcement Learning

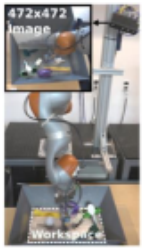


Off Policy Reinforcement Learning



Offline Reinforcement Learning





Grasping

Manipulation

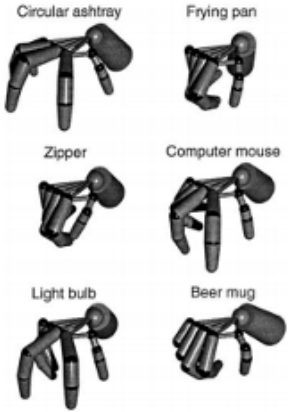
So far:

- We knew what we wanted
- Short-horizon behaviors
- Well defined tasks/rewards

Why Skill Discovery?

What if we want our agent to discover interesting behaviors?

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



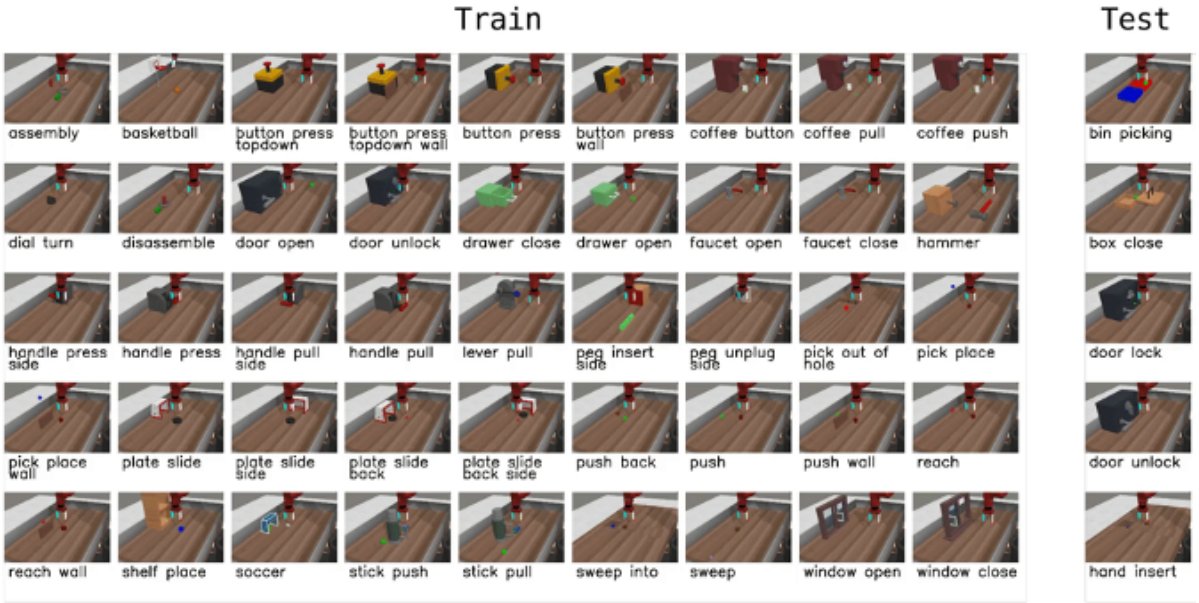
[Postural hand synergies for tool use, Santello, et al., 1998]

[Learning Agile Soccer Skills for a Bipedal Robot with Deep Reinforcement Learning, Tuomas Haarnoja, et al., 2023]

Why Skill Discovery?

Coming up with tasks is tricky…

Task ideas for a tabletop manipulation scenario



[Meta-World, Yu, Quillen, He, Julian, et al., 2019]

Why Hierarchical RL?

Performing tasks at various levels of abstractions

Bake a cheesecake

Buy ingredients

Go to the store

Walk to the door

Take a step

Contract muscle X

Exploration

Outline

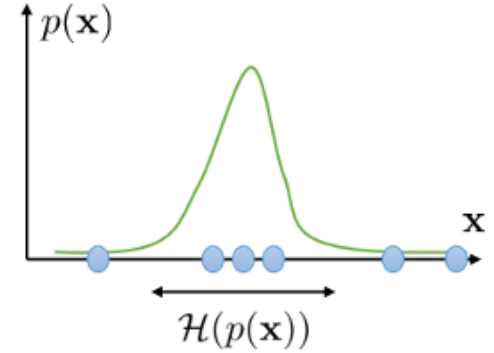
- **Information-theoretic concepts**
- Skill Discovery (Unsupervised Reinforcement Learning)
- Hierarchical Reinforcement Learning
- Goal Conditioned RL

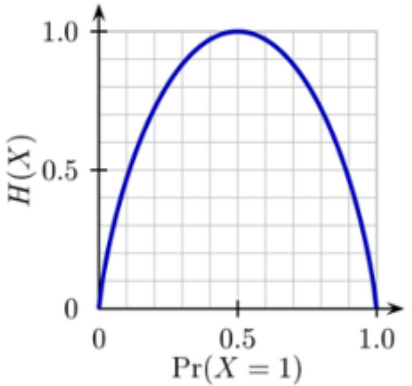
Entropy

$p(\mathbf{x})$ 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

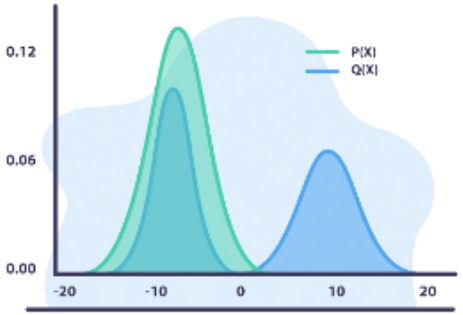




KL-divergence

Distance between two distributions

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



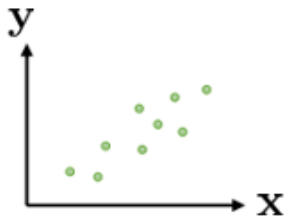
Kullback–Leibler Divergence

Mutual information

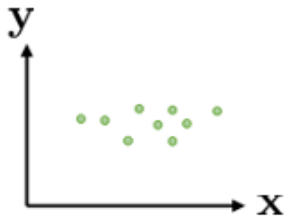
$$\mathcal{I}(\mathbf{x};\mathbf{y}) = D_{\mathrm{KL}}(p(\mathbf{x},\mathbf{y}) \| p(\mathbf{x})p(\mathbf{y}))$$

$$= E_{(\mathbf{x},\mathbf{y})\sim p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})p(\mathbf{y})}\right]$$

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



high MI: $\mathbf{x}$ and $\mathbf{y}$ are *dependent*



low MI: $\mathbf{x}$ and $\mathbf{y}$ are *independent*

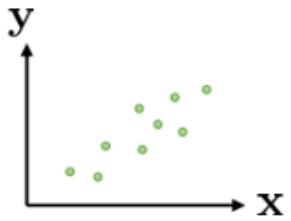
High MI?
x- it rains tomorrow, y – streets are wet tomorrow
x- it rains tomorrow, y – we find life on Mars tomorrow

Mutual information

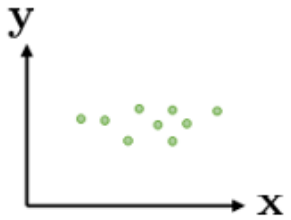
$$\mathcal{I}(\mathbf{x};\mathbf{y}) = D_{\mathrm{KL}}(p(\mathbf{x},\mathbf{y}) \| p(\mathbf{x})p(\mathbf{y}))$$

$$= E_{(\mathbf{x},\mathbf{y})\sim p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})p(\mathbf{y})}\right]$$

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



high MI: $\mathbf{x}$ and $\mathbf{y}$ are *dependent*



low MI: $\mathbf{x}$ and $\mathbf{y}$ are *independent*

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

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

Outline

- Information-theoretic concepts
- **Skill Discovery (Unsupervised Reinforcement Learning)**
- Hierarchical Reinforcement Learning
- Goal Conditioned RL

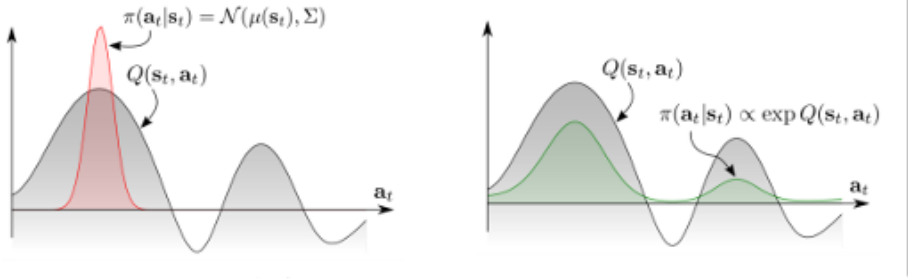
What is Skill?

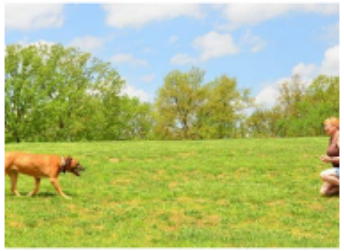
- A skill is a latent-conditioned policy that alters that state of the environment in a consistent way.
- DRL has been demonstrated to effectively learn a wide range of reward-driven skills, including playing games, controlling robots, and navigating complex environments
- Learning skills without reward has several practical applications.
  - e.g. For long horizon tasks, skills discovered without reward can serve as primitives for hierarchical RL, effectively shortening the episode length.

Soft Q-Learning

Objective:

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





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 $\phi \leftarrow \arg\min_\phi \frac{1}{2}\sum_i \|Q_\phi(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{y}_i\|^2$

($K\times$)

$$\pi(\mathbf{a}|\mathbf{s}) = \arg\max_{\mathbf{a}} Q_\phi(\mathbf{s}, \mathbf{a})$$

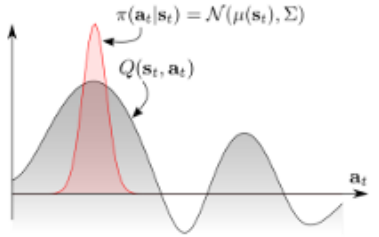
Soft Q-learning

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

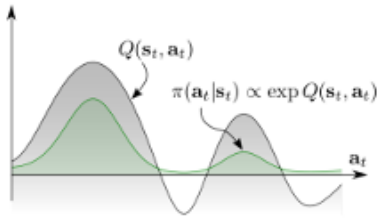
($K\times$)

$$\pi(\mathbf{a}|\mathbf{s}) = \arg\max_{\mathbf{a}} Q_\phi(\mathbf{s}, \mathbf{a})$$ ~~$\propto \exp\left(A_t(s_t, a_t)\right)$~~

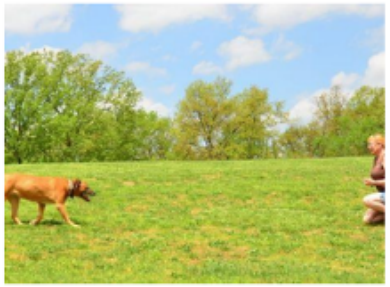
Soft Q-Learning



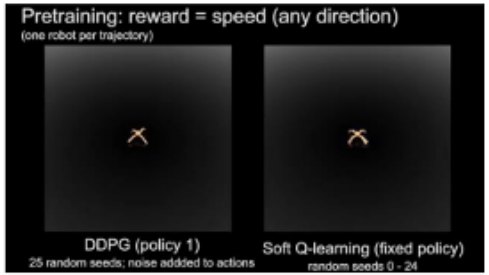
Exploration

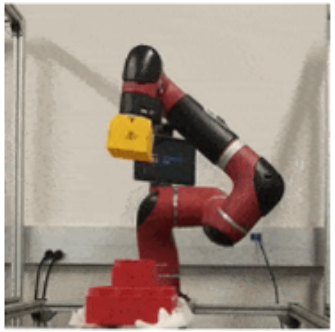


Fine-tunability



Robustness



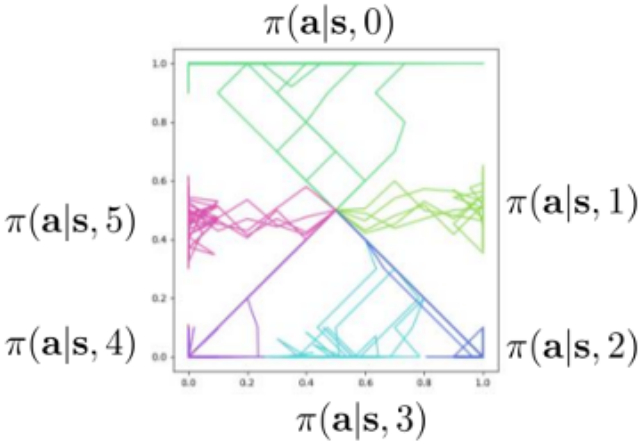


Haarnoja et al. RL with Deep Energy-Based Policies, 2017

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

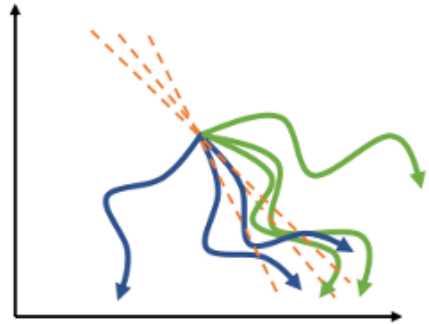
Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need.

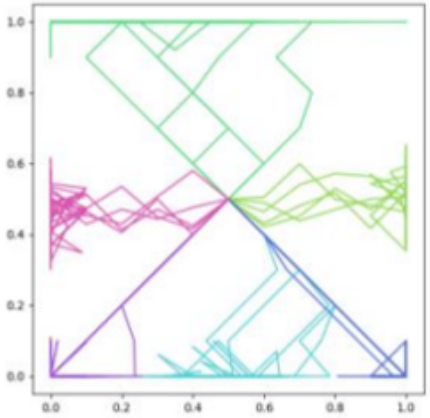
Diversity-promoting reward function

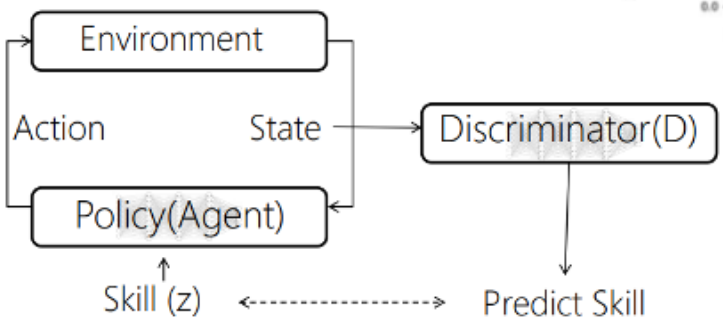
$$\pi(\mathbf{a}|\mathbf{s}, z) = \arg\max_{\pi} \sum_{z} E_{\mathbf{s}\sim\pi(\mathbf{s}|z)}[r(\mathbf{s}, z)]$$

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

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

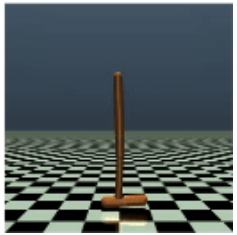
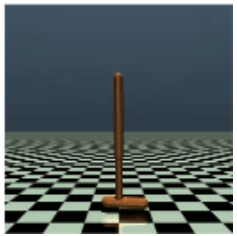
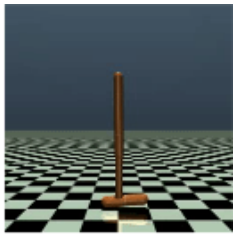
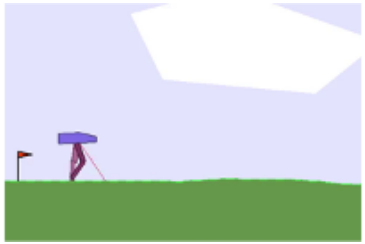
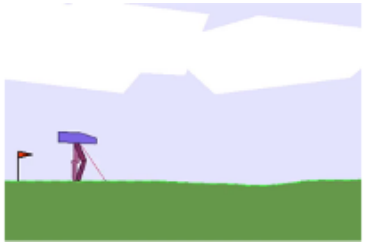






Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need

Examples of learned tasks



Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need.

A connection to mutual information

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

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

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

maximized by using uniform prior $p(z)$

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

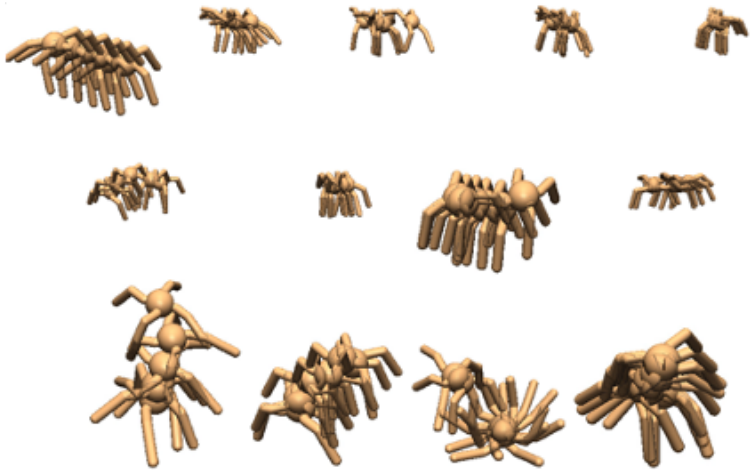
Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need.

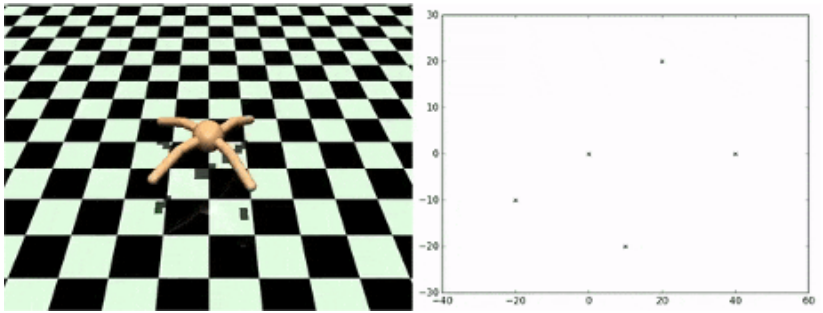
See also: Gregor et al. Variational Intrinsic Control. 2016

How to use learned skills?

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

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

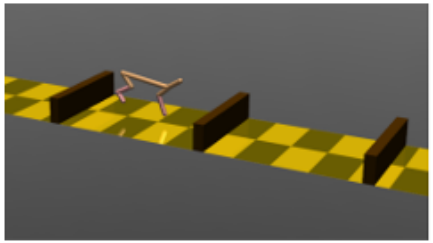




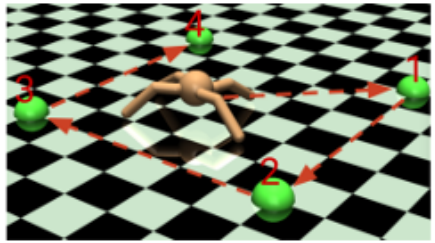
Learn a policy that operates on z's

Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need

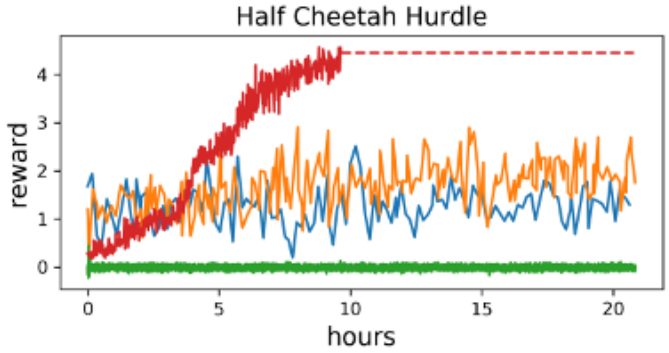
Results: hierarchical RL

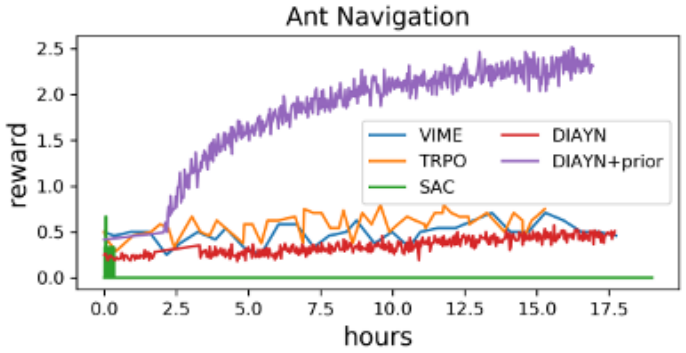


Cheetah Hurdle



Ant Navigation

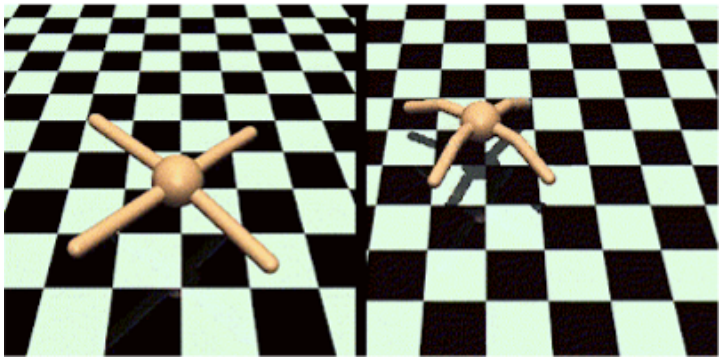




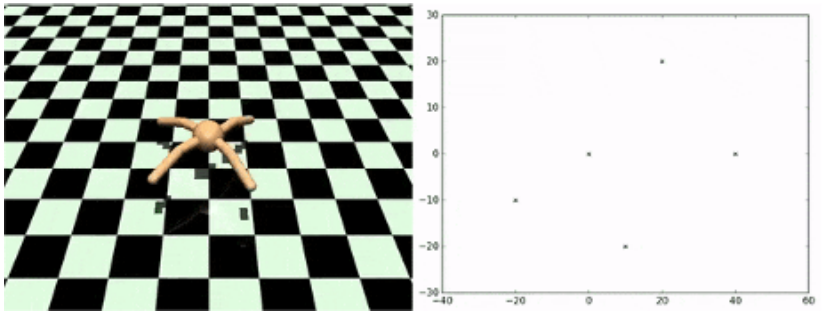
Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need

What's the problem?

Skills might not be particularly useful



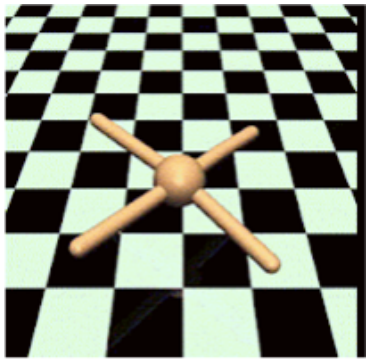
It's not very easy to use the learned skills

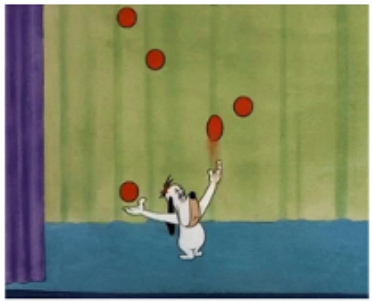


What makes a useful skill?

What's the problem?

Consequences are hard to predict





Consequences are easy to predict

Slightly different mutual information

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

$$\max \mathcal{I}(s', z \mid s) = \max \Big(\mathcal{H}(s' \mid s) - \mathcal{H}(s' \mid s, z) \Big)$$

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

Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need.

Slightly different mutual information

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

$$\max \mathcal{I}(s', z \mid s) = \max \Big(\mathcal{H}(s' \mid s) - \mathcal{H}(s' \mid s, z) \Big)$$

Future hard to predict for different skills

Predictable future for a given skill

$$I(s'; z|s) \geq \mathbb{E}_s \mathbb{E}_z \mathbb{E}_{p(s'|s,z)} [\log \frac{q_\phi(s'|s,z)}{p(s'|s)}]$$

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

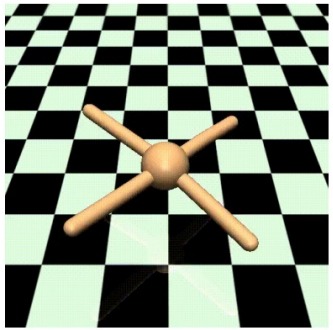
Eysenbach, Gupta, Ibarz, Levine. Diversity is All You Need.

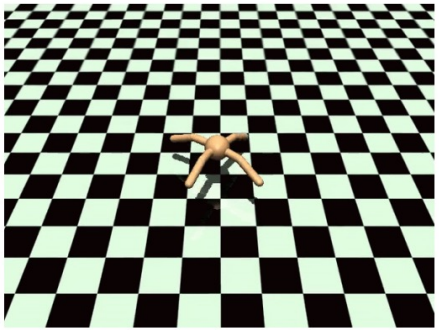
Skill-dynamics model

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

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

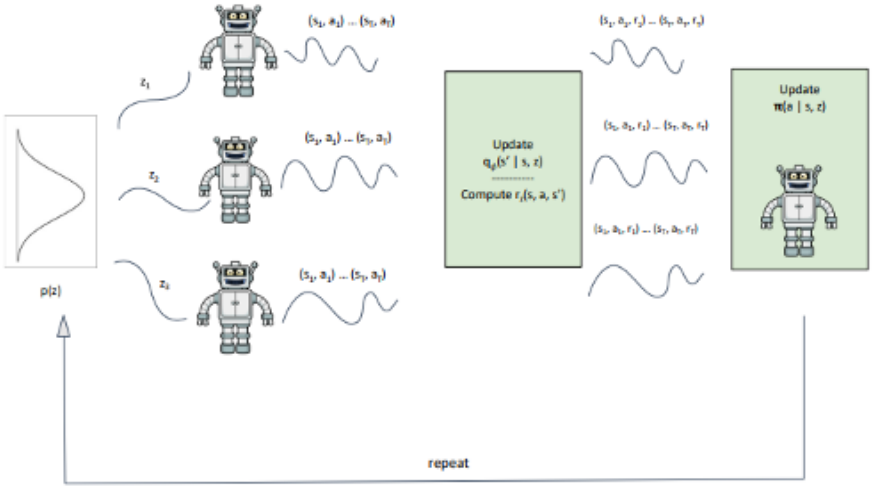
Skills are optimized specifically to make skill-dynamics easier to model





Sharma, Gu, Levine, Kumar, Hausman, DADS, 2019.

DADS algorithm

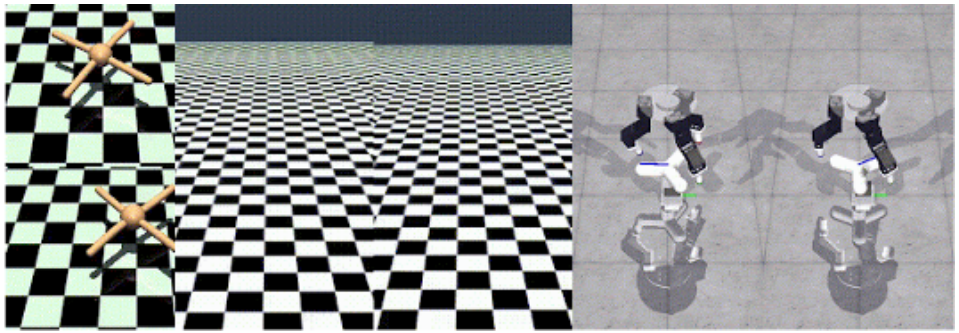


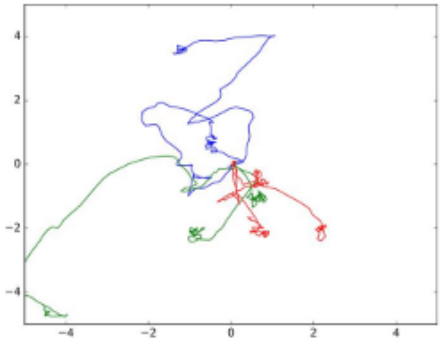
Algorithm 1: Dynamics-Aware Discovery of Skills (DADS)

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

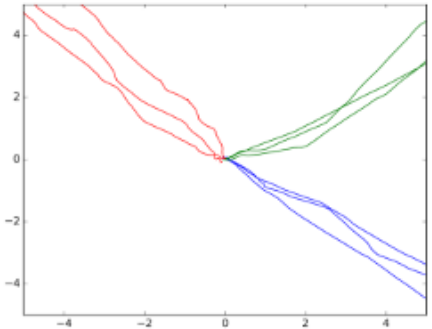
Sharma, Gu, Levine, Kumar, Hausman, DADS, 2019.

DADS results





DIAYN

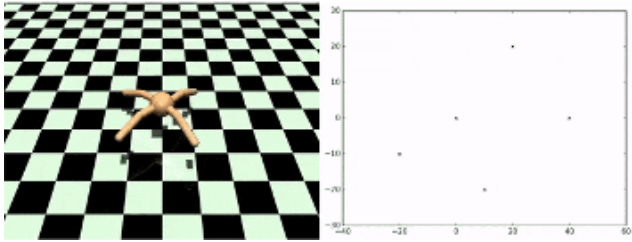


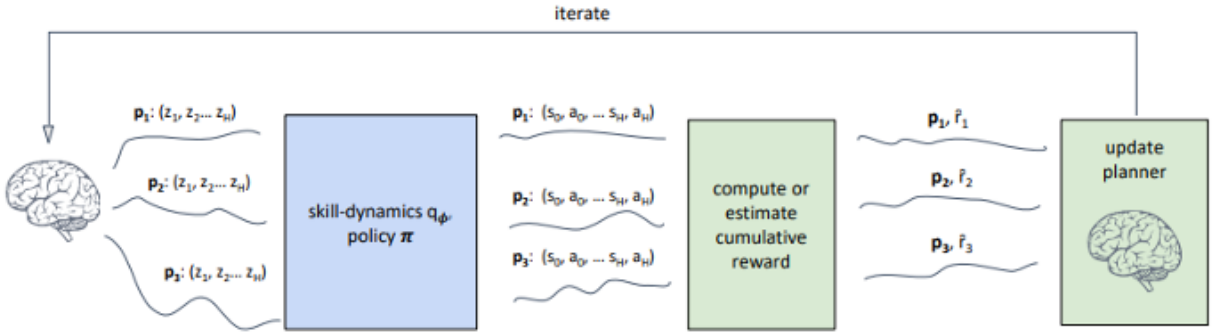
DADS

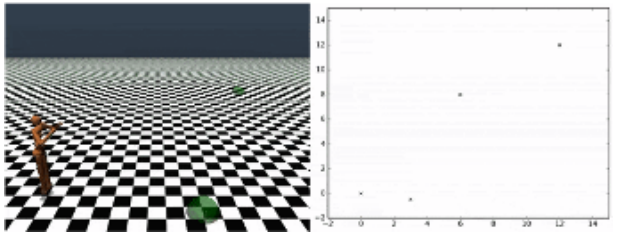
Sharma, Gu, Levine, Kumar, Hausman, DADS, 2019.

Using learned skills

Use skill-dynamics for model-based planning Plan for skills not actions
Tasks can be learned zero-shot







Sharma, Gu, Levine, Kumar, Hausman, DADS, 2019.

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

Additional Readings

- Eysenbach, B., Gupta, A., Ibarz, J., & Levine, S. (2018). Diversity is All You Need: Learning skills without a reward function. In arXiv [cs.AI]. http://arxiv.org/abs/1802.06070

- Sharma, A., Gu, S., Levine, S., Kumar, V., & Hausman, K. (2019). Dynamics-aware unsupervised discovery of skills. In arXiv [cs.LG]. http://arxiv.org/abs/1907.01657

- Sharma, A., Ahn, M., Levine, S., Kumar, V., Hausman, K., & Gu, S. (2020). Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. In arXiv [cs.RO]. http://arxiv.org/abs/2004.12974

Outline

- Information-theoretic concepts
- Skill Discovery (Unsupervised Reinforcement Learning)
- **Hierarchical Reinforcement Learning**
- Goal Conditioned RL

Hierarchical Reinforcement Learning

- Performing tasks at **various levels of abstractions**.
- Helps with exploration and temporally extended tasks
- Seems like a natural direction for harder RL problems
- Can be difficult to get it to work

Courtesy: Most of slides of this section are adopted from Doina Precup Presentation, “Introduction to Hierarchical Reinforcement Learning”, McGill University.

What is Temporal Abstraction? - Motivating Example

- Consider an activity such as cooking
  - **High-level**: Choose a recipe, make grocery List
  - **Medium-level**: get a pot, put ingredients in the Pot, stir until smooth
  - **Low-level**: wrist and arm movement, muscle Contraction
- All have to be seamlessly integrated.

Temporal Abstraction in AI

- Temporal Abstractions is not specific to RL, it has a long root in AI.
- It has been shown to:
  - Generate shorter plans
  - Reduce the complexity of choosing actions
  - Provide robustness against model misspecification
  - Allow taking shortcuts in the environment
  - Improves interpretability

Advantages in Complex RL Tasks

Advantages to planning

- Need to generate shorter plans
- Improves robustness to model errors
- Might need to look at fewer states, since the abstract actions have pre-defined termination conditions
- Discretize the action space in continuous problems

Advantages to learning

- Improves exploration (can travel in larger leaps)
- Gives a natural way of using a single stream of data to learn many things (off-policy learning)

Advantages to interpretability

- Focusing attention: Sub-plans ignore a lot of information - Improves readability of both models and resulting plans - Reduces the problem size

Procedural, Temporally Abstract knowledge: Options

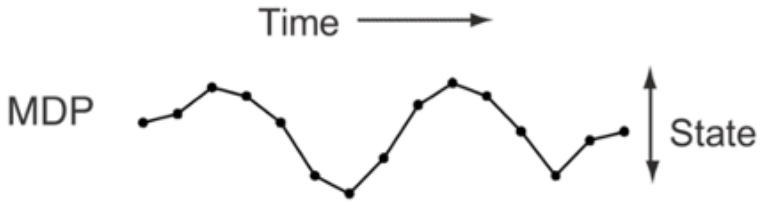
- Generalize actions to include temporally extended courses of actions.
- An option ω = (I, π, β) has three components:
  - An initiation set $I \subseteq S$
  - A policy π: $S \times \mathcal{A} \rightarrow [0,1]$
  - A terminations condition $\beta : S \rightarrow [0,1]$
- If the option (I, π, β) is taken at $s \in I$, then actions are selected according to π until the option terminates stochastically according to β.

Options - Example

- **Robot Navigation**: If there is no obstacle in front (I), go forward (π) until you get too close to another object (β).

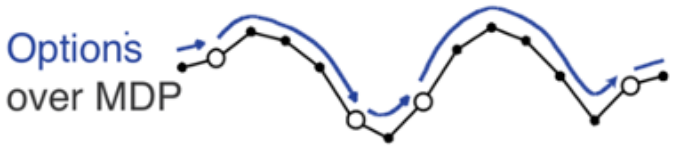
- **Open-the-door**:
  - I: all states in which a closed door is within reach
  - π: pre-defined controller for reaching, grasping, and turning the door knob
  - β: terminate when the door is open

Decision-Making with Options: Semi-MDPs



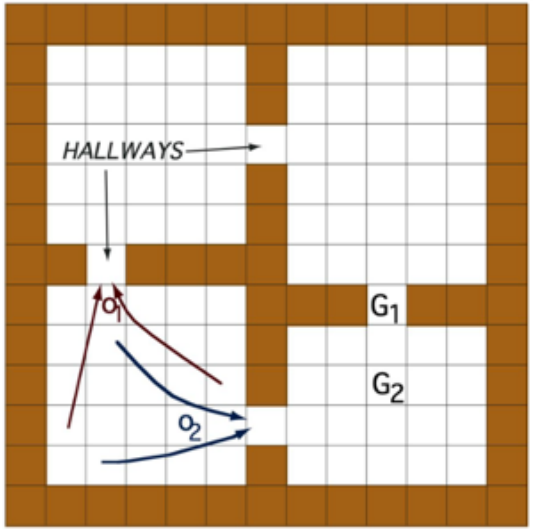
Discrete time
Homogeneous discount

Continuous time
Discrete events
Interval-dependent discount



Discrete time
Overlaid discrete events
Interval-dependent discount

Example: Navigation



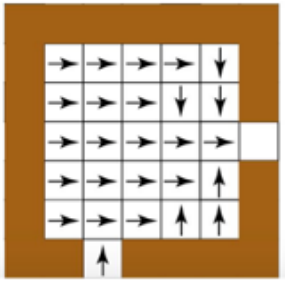
4 stochastic primitive actions

Fail 33% of the time

8 multi-step options

(to each room's 2 hallways)

Example of one option's policy:

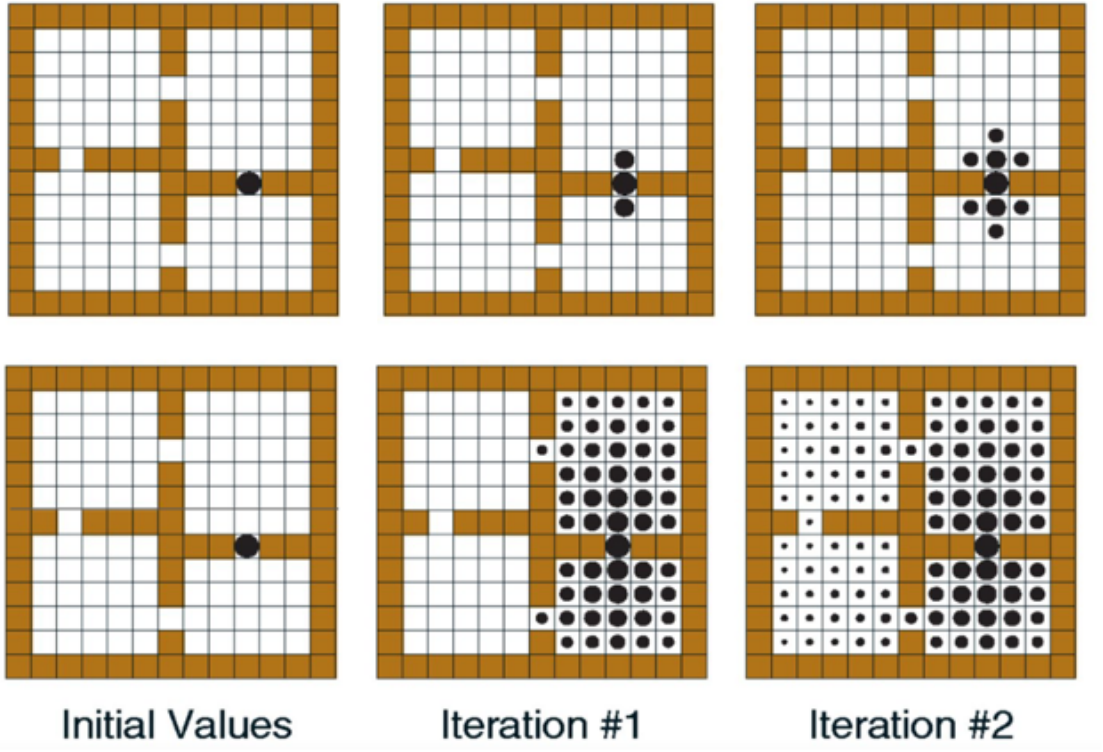


Target Hallway

Example: Navigation

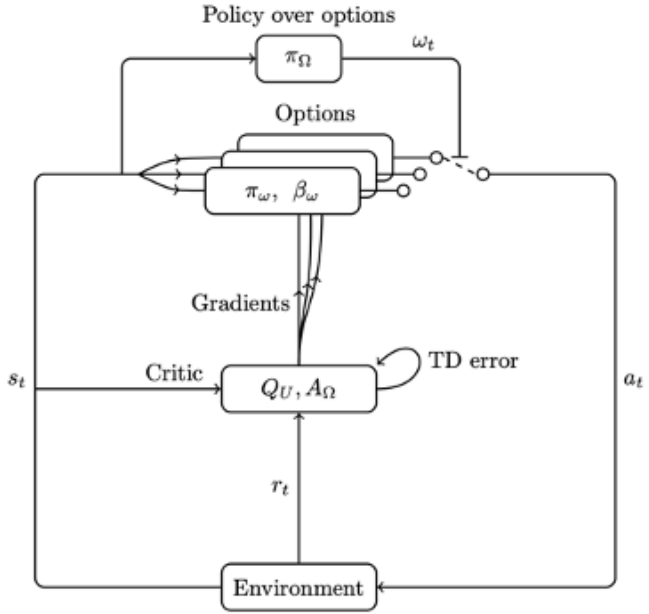
With cell-to-cell primitive actions

With room-to-room options



Option-critic

Architecture



Algorithm

Algorithm 1: Option-critic with tabular intra-option Q-learning

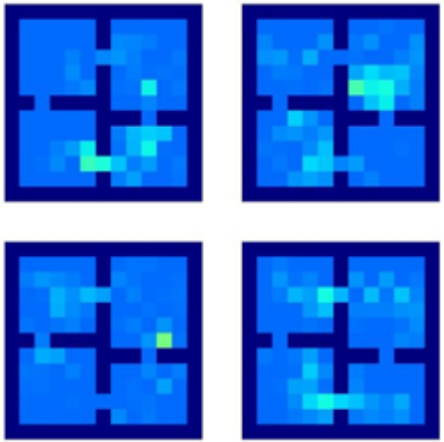
$s \leftarrow s_0$
Choose ω according to an ϵ-soft policy over options $\pi_\Omega(s)$
repeat
  Choose a according to $\pi_{\omega,\theta}(a \mid s)$
  Take action a in s, observe s', r

  1. Options evaluation:
  $\delta \leftarrow r - Q_U(s,\omega,a)$
  if s' *is non-terminal* **then**
    $\delta \leftarrow \delta + \gamma(1-\beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \gamma\beta_{\omega,\vartheta}(s')\max_{\bar{\omega}} Q_\Omega(s',\bar{\omega})$
  end
  $Q_U(s,\omega,a) \leftarrow Q_U(s,\omega,a) + \alpha\delta$

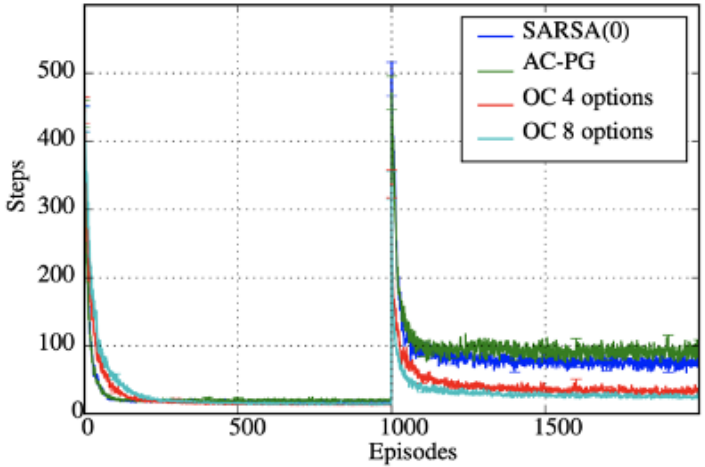
  2. Options improvement:
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log \pi_{\omega,\vartheta}(a \mid s)}{\partial\theta} Q_U(s,\omega,a)$
  $\vartheta \leftarrow \vartheta - \alpha_\vartheta \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial\vartheta}\left(Q_\Omega(s',\omega) - V_\Omega(s')\right)$

  if $\beta_{\omega,\vartheta}$ *terminates in* s' **then**
  choose new ω according to ϵ-soft$(\pi_\Omega(s'))$
  $s \leftarrow s'$
until s' *is terminal*

Experiment 1: Four-Room Navigation

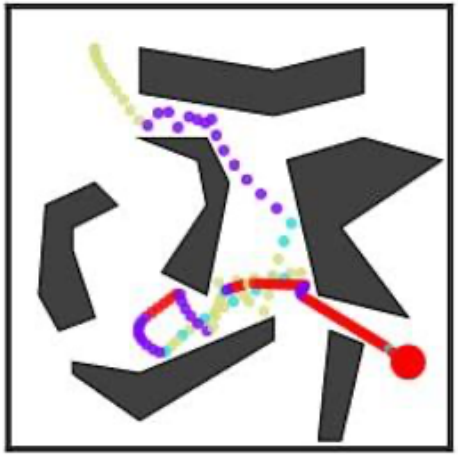


Termination probabilities for the option-critic agent learning with 4 options. The darkest color represents the walls in the environment while lighter colors encode higher termination probabilities.

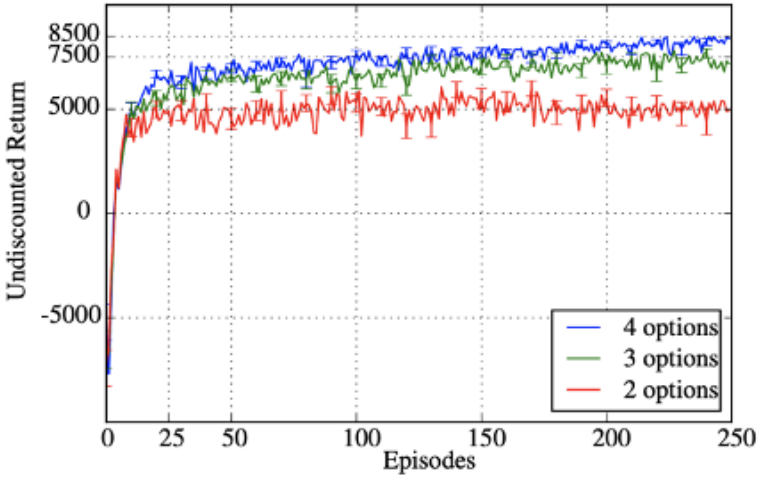


After a 1000 episodes, the goal location in the four-room domain is moved randomly. Option-critic (“OC”) recovers faster than the primitive actor-critic (“AC-PG”) and SARSA(0). Each line is averaged over 350 runs.

Experiment 2: Pinball



Pinball: Sample trajectory of the solution found after 250 episodes of training using 4 options All options (color-coded) are used by the policy over options in successful trajectories. The initial state is in the top left corner and the goal is in the bottom right one (red circle).



Learning curves in the Pinball domain.

Outline

- Information-theoretic concepts
- Skill Discovery (Unsupervised Reinforcement Learning)
- Hierarchical Reinforcement Learning
- **Goal Conditioned RL**

Goal-Conditioned RL (GCRL)

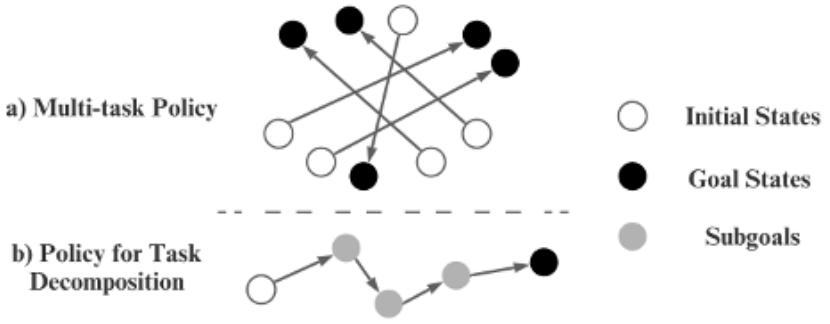
- standard RL solutions: learn a policy solely depending on the states or observations.

- GCRL additionally requires the agent to make decisions according to different goals.
  - The agent can achieve a wide diversity of tasks of varying horizons in its environment.

- This approach can mitigate limitations of Hierarchical RL
  - Designing appropriate temporal choices.
  - More complicated to deploy on real world tasks.

Goal-Conditioned RL (GCRL)

- Agent can learn different goals:
  - simultaneously (multi-task policy)
  - Sequentially by task decomposition



Reward Function in Goal-Conditioned RL: $r(\mathbf{s}) = r(\mathbf{s}, \mathbf{s}g) = -d(\mathbf{s}, \mathbf{s}g)$

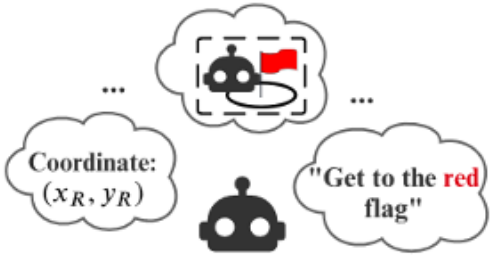
Distance function d examples:
- Euclidean ℓ2
- sparse 0/1

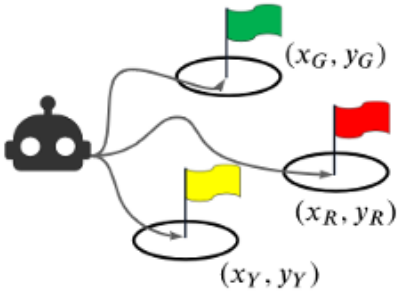
What are Goals?

there is no unique answer for the structure of goals

It is highly depends on the task specifications.

Typical representations of goals in GCRL:
vectors, images and languages.



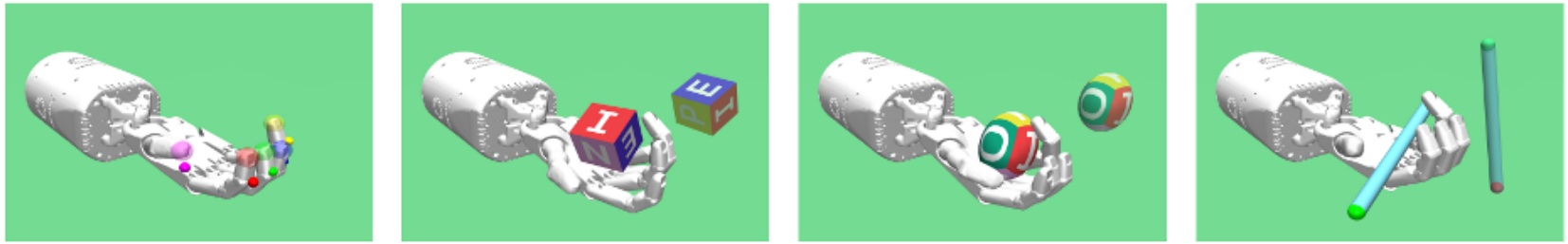


Vectors

- Goals are defined as desired properties or features of an agent
- The representation of the goal is a set of vectors of the desired location

  - Asking the robotic arm to push, pick, or slide a block to a specific location
  - Setting the goal as a desired moving speed
  - A vector of the target position to reach

- The goal space is usually a subspace of the state space
  - A target state, or a dimension reduction function that choosing specific dimensions from the state.



[Plappert et al., 2018]

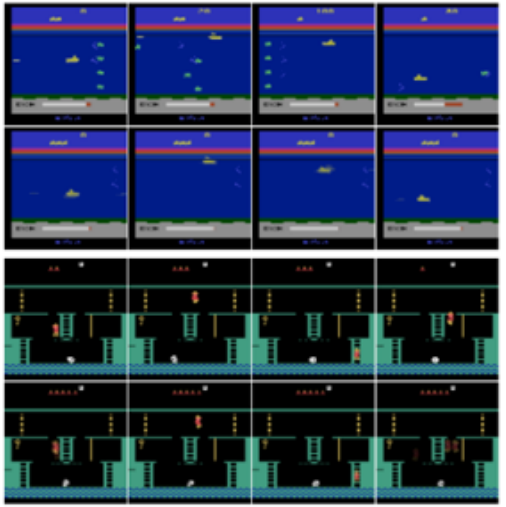
Images

- more straightforward representation for complex tasks
- The goal is either:
  - the image of the target position to approach
  - a imagined image of winning the game.

The state and goal images can be directly fed into the policy and value networks and trained in an end-to-end manner

Due to the high dimensionality of images, a more general way is to work on a compact vector representation

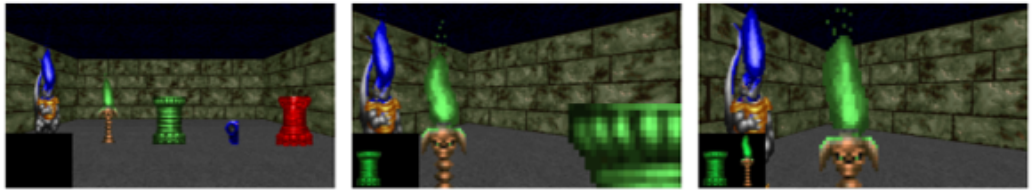
- variational Auto-Encoder (VAE)
- context-conditioned VAE
- via training a weak supervised model for learning disentangled representation
- an end-to-end supervised imitation learning objective



[Warde-Farley et al., 2019]

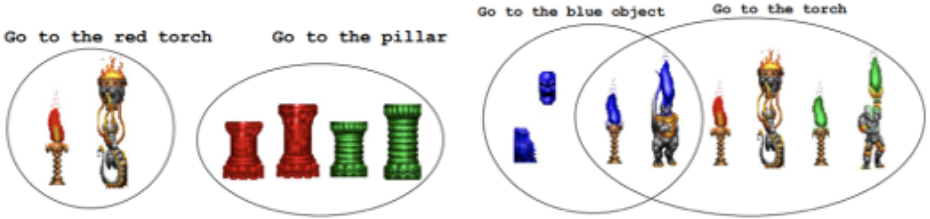
Languages

They can easily be understood by human



The goal is:

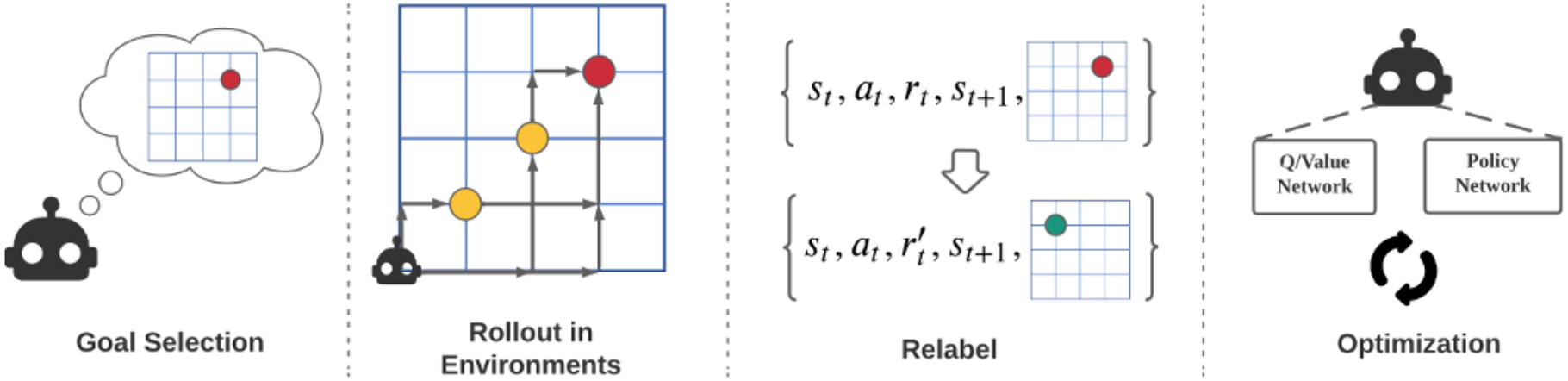
- an instruction sentence containing explicit verbs and objects: “go to blue torch”
- question words: “there is a red ball; are there any matte cyan sphere right of it?”
- simply, a predefined set of semantic predicates: “close”, “above”



These words must be either translated into embeddings:

- Via pre-training models
- Taken into a recurrent neural network (RNN) as learning the embedding jointly with the RL agent
- Can be learned to translate the handcrafted encodings into vectors

Algorithms to GCRL



- Optimization is learning a conditioned policy towards a set of goals.
- To generate or select sub-goals to help the agent reach the target goals.
- Relabeling focuses on replacing the history data in the experience buffer before training the agent

Relabeling: Idea

Task 1: passing

Task 2: shooting goals

What if you accidentally perform a good pass when trying to shoot a goal? Store experience as normal. *and* Relabel experience with task 2 ID & reward and store.

“hindsight relabeling” "hindsight experience replay” (HER)

Multi-task RL with relabeling

1. Collect data $\mathcal{D}_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_i, r_{1:T})\}$ using some policy
2. Store data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
3. Perform hindsight relabeling:
   a. Relabel experience in $\mathcal{D}_k$ for task $\mathcal{T}_j$:
      $\mathcal{D}'_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_j, r'_{1:T}\}$ where $r'_t = r_j(\mathbf{s}_t)$
   b. Store relabeled data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}'_k$
4. Update policy using replay buffer $\mathcal{D}$

$k++$

Which task $\mathcal{T}j$ to choose?
- randomly
- task(s) in which the trajectory gets high reward
- other

Eysenbach et al. Rewriting History with Inverse RL
Li et al. Generalized Hindsight for RL
Kalashnikov et al. MT-Opt
Yu et al. Conservative Data-Sharing

When can we apply relabeling?
- reward function form is known, evaluatable
- dynamics consistent across goals/tasks
- using an off-policy algorithm*

Kaelbling. Learning to Achieve Goals. IJCAI ‘93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS ‘17

Goal-conditioned RL with hindsight

1. Collect data $\mathcal{D}_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{s}_g, r_{1:T})\}$ using some policy
2. Store data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_k$
3. Perform hindsight relabeling:
   a. Relabel experience in $\mathcal{D}_k$ using last state as goal: $\mathcal{D}'_k = \{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, \mathbf{s}_T, r'_{1:T}\}$ where $r'_t = -d(\mathbf{s}_t, \mathbf{s}_T)$
   b. Store relabeled data in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}'_k$
4. Update policy using replay buffer $\mathcal{D}$

$k++$

Other relabeling strategies?
use any state from the trajectory

Result: exploration challenges alleviated

Kaelbling. Learning to Achieve Goals. IJCAI ‘93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS ‘17

Hindsight relabeling for goal-conditioned

Example: goal-conditioned RL, simulated robot manipulation

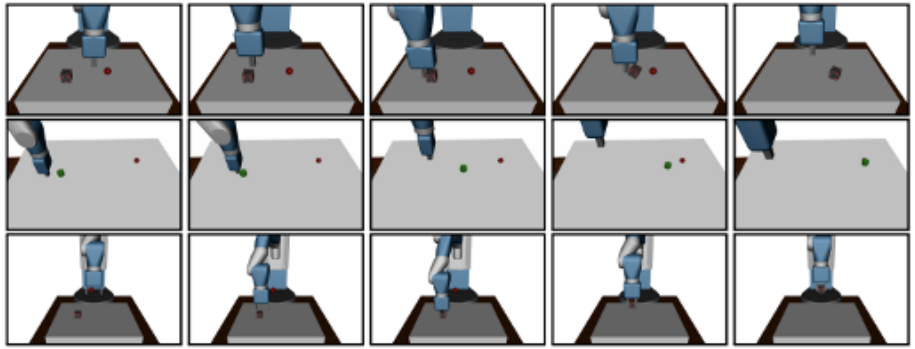
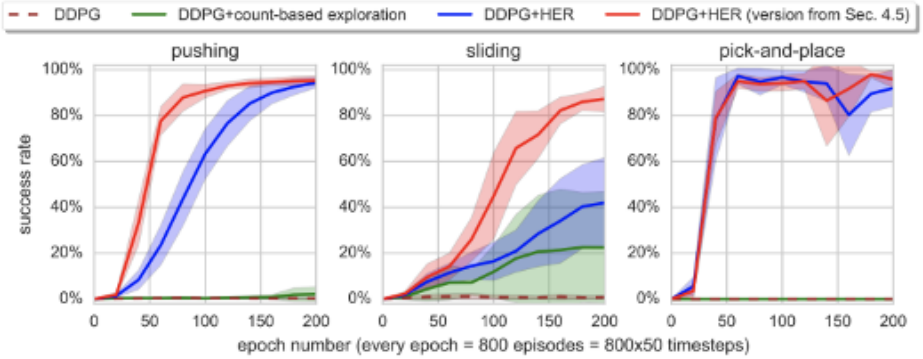


Figure 2: Different tasks: *pushing* (top row), *sliding* (middle row) and *pick-and-place* (bottom row). The red ball denotes the goal position.



Kaelbling. Learning to Achieve Goals. IJCAI ‘93
Andrychowicz et al. Hindsight Experience Replay. NeurIPS ‘17

Additional Readings

- Liu, M., Zhu, M., & Zhang, W. (2022). Goal-conditioned reinforcement learning: Problems and solutions. In arXiv [cs.AI]. http://arxiv.org/abs/2201.08299
- Nachum, O., Gu, S., Lee, H., & Levine, S. (2018). Data-efficient hierarchical reinforcement learning. In arXiv [cs.LG]. http://arxiv.org/abs/1805.08296
- Gupta, A., Kumar, V., Lynch, C., Levine, S., & Hausman, K. (2019). Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In arXiv [cs.LG]. http://arxiv.org/abs/1910.11956
- Eysenbach, B., Salakhutdinov, R., & Levine, S. (2019). Search on the replay buffer: Bridging planning and reinforcement learning. In arXiv [cs.AI]. http://arxiv.org/abs/1906.05253
- Nasiriany, S., Pong, V. H., Lin, S., & Levine, S. (2019). Planning with goal-conditioned policies. In arXiv [cs.LG]. http://arxiv.org/abs/1911.08453