强化学习原理及应用
Reinforcement Learning (RL): Theories & Applications

DCS6289 Spring 2022

Lijun Xia（夏礼俊）

School of Computer Science and Engineering
Sun Yat-Sen University

Lecture 14：Hierarchical RL

14th June. 2022

Hierarchical RL

- Why Hierarchical Approaches?
  - To deal with the problem of sparse reward
  - To solve the sequential decision-making problem with long horizon
  - Many large problems have hierarchical structure that allows them to be broken down into sub-problems. The sub-problems, being smaller, are often solved more easily

- What is key of Hierarchical RL?
  - Temporally extended actions
  - Policy on the abstract action and policy in the abstract action

Hierarchical RL

Basic model – Two levels of hierarchy

- High level:
  - High level policy: $g_t = \pi^h(s_t)$
  - The high level policy receives state s_t, then chooses an abstracted action $g_t \in G$, where G denotes the set of all possible current abstracted actions (e.g., skills/sub-policies/options/goals)
  - The high level aims to maximize the rewards from environment directly, i.e., extrinsic rewards

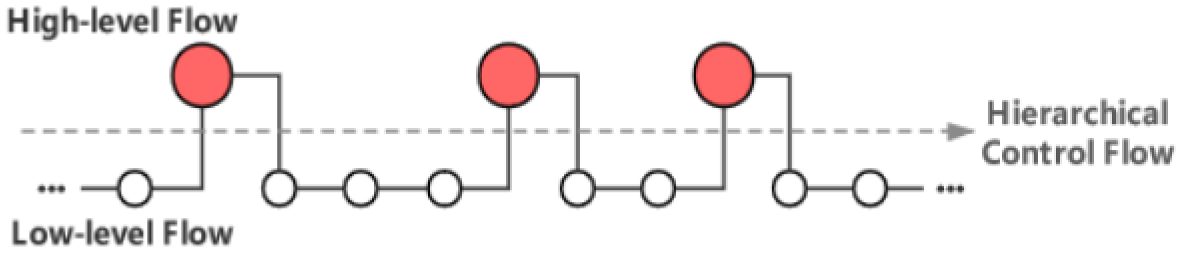
- Low level:
  - Low level policy: $a_t = \pi^l(s_t, g_t)$
  - The low level policy receives state s_t and g_t then takes a primitive action a_t, while results in a new state s_{t+1}
  - The low level is expected to accomplish subtasks or achieve goals from high level

Hierarchical RL

Basic model – Two levels of hierarchy

- Both high level and low level can use RL algorithms to realize (e.g. DQN, PPO, DDPG, TD3)
- The hierarchy can be deeper (i.e., more than 2 levels)

$$g_t = \pi^h(s_t)$$



$$a_t = \pi^l(s_t, g_t)$$

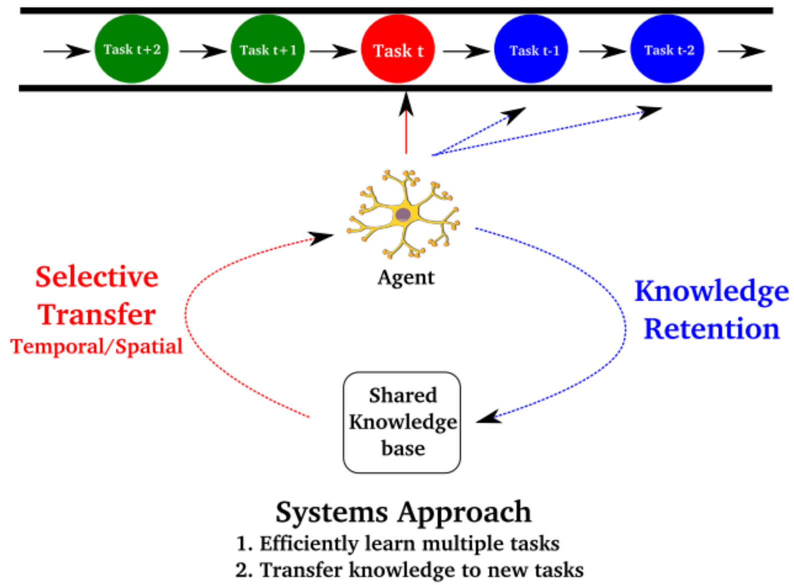
Hierarchical RL

- Discrete abstracted action
  - H-DRLN
  - OC
- Continual abstracted action
  - FuN
  - HIRO

Hierarchical RL

H-DRLN

- Motivation
  - In Minecraft, the task of building a wooden house can be decomposed into sub-tasks (a.k.a skills) such as chopping trees, sanding the wood, cutting the wood into boards and finally nailing the boards together.
  - The knowledge gained from 'building a house' task can also be partially reused when building a small city
- Main idea
  - Predefined and reusable skills

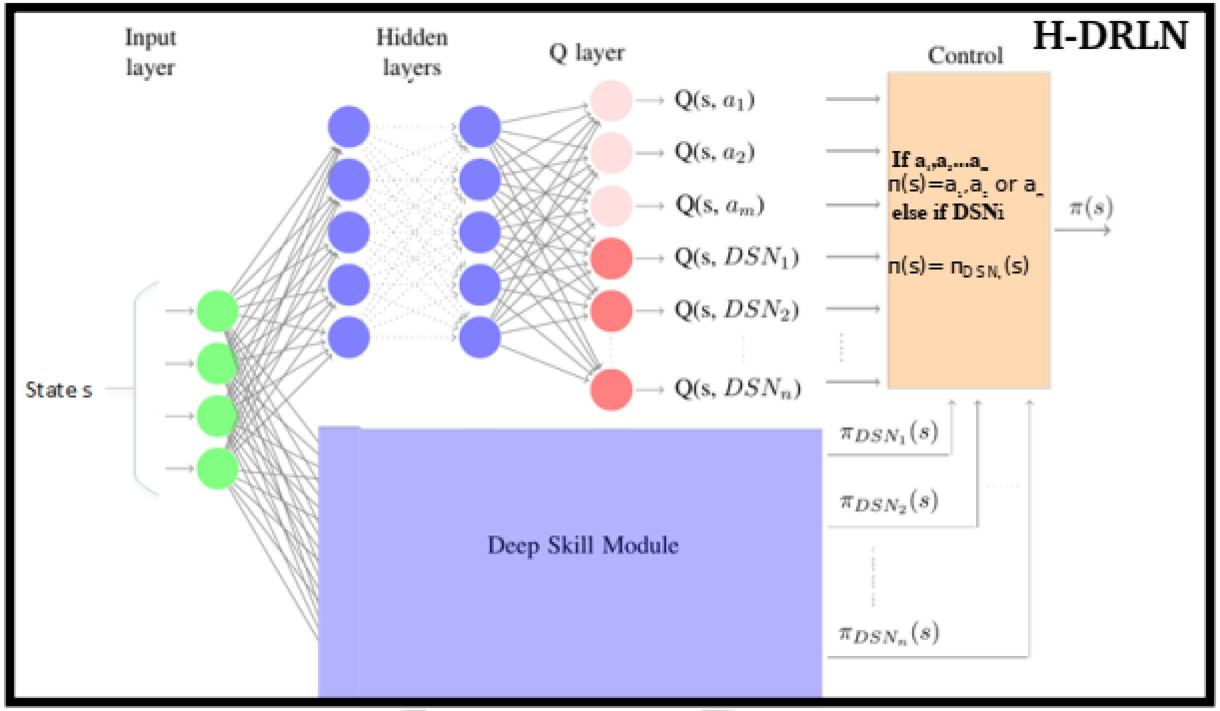


Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

H-DRLN

- The outputs of the H-DRLN consist of primitive actions as well as skills. The H-DRLN learns a policy that determines when to execute primitive actions and when to reuse pre-learned skills.

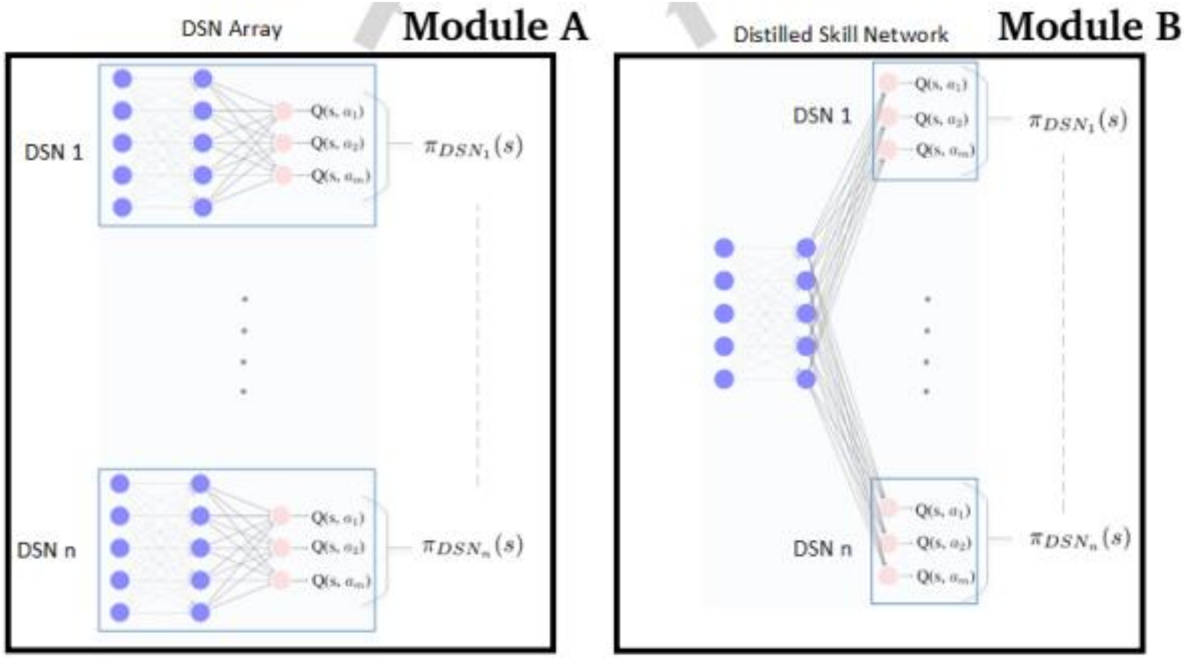


Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

H-DRLN

- The architecture of the deep skill module can be either a DSN array or a Distilled Multi-Skill Network.



Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

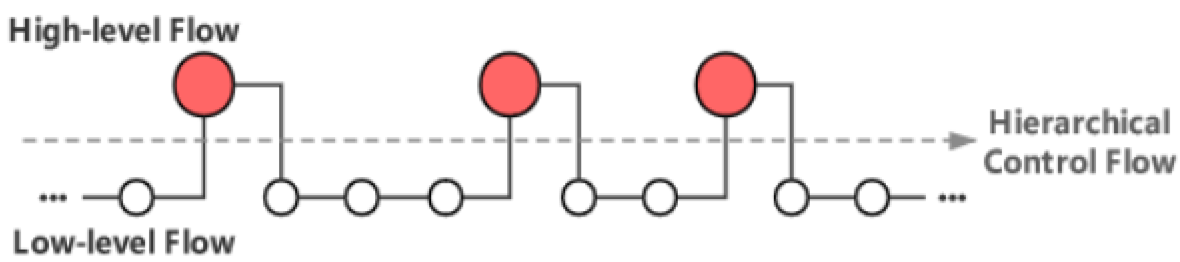
The training process of H-DRLN

- High level:
  - For a skill σ_t initiated in state s_t at time t that has executed for a duration k, the H-DRLN target function is given by:

$$y_t = \begin{cases} \sum_{j=0}^{k-1} \left[\gamma^j r_{j+t}\right] & \text{if } s_{t+k} \text{ terminal} \\ \sum_{j=0}^{k-1} \left[\gamma^j r_{j+t}\right] + \gamma^k \max_{\sigma'} Q_{\theta_{target}} \left(s_{t+k}, \sigma'\right) & \text{else} \end{cases}$$

  - Transition tuple $(s_t, \sigma_t, \sum_{j=0}^{k-1} \gamma^j r_{t+j}, s_{t+k})$

- Low level:
  - Pre-trained with manually defined scenarios

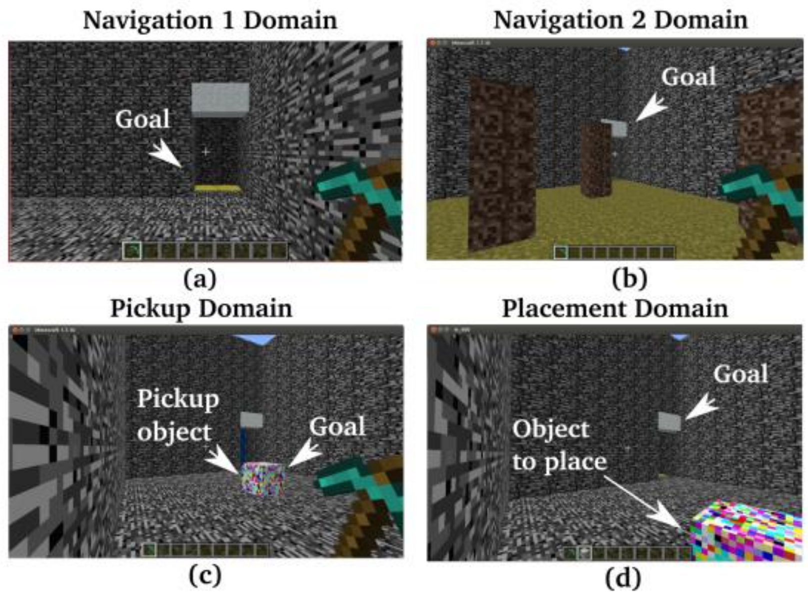


Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

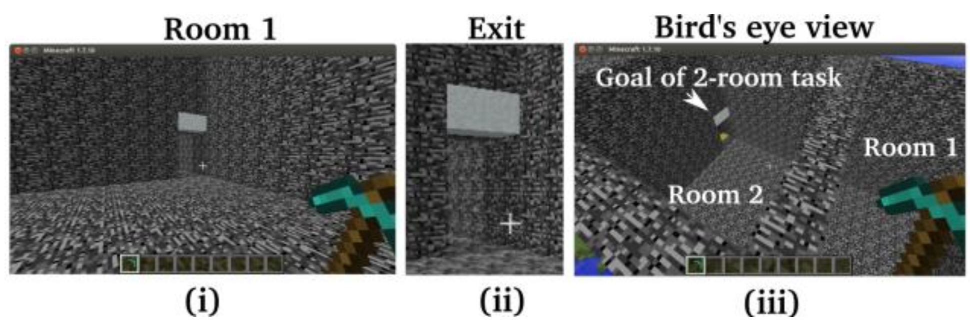
Hierarchical RL

H-DRLN

- Experiments
  - The manually defined scenarios:



  - The Two-room scenario:

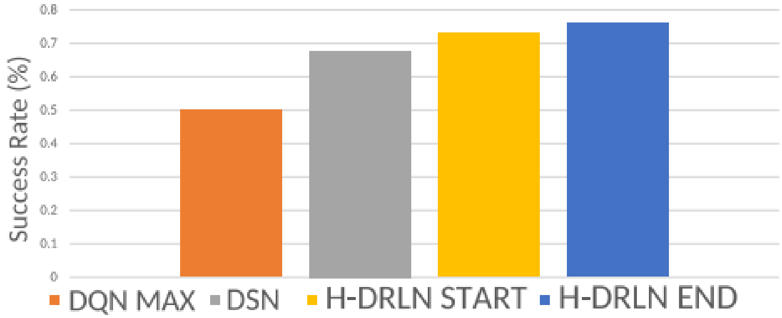


Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

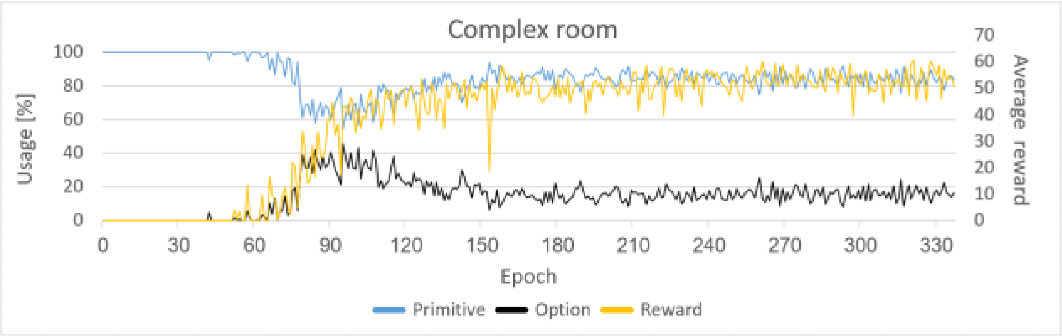
Hierarchical RL

H-DRLN

- Experiments
  - The success percentages of the two-room scenario



  - Information in the training process

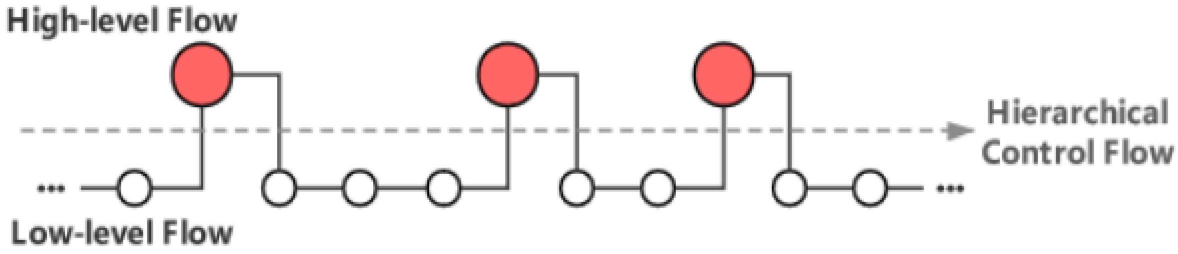


Tessler, Chen, et al. "A Deep Hierarchical Approach to Lifelong Learning in Minecraft". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

OC

- Motivation
  - Create temporally extended actions autonomously
  - No additional rewards or subgoals are required
- Method
  - Using differentiable parameterized function approximators
    - policy over options π_{Ω}
    - intra-option policy $\pi_{W,\theta}$ for option W
    - Termination policy $\beta_{\mathrm{W},\vartheta}$ for option W
  - Calculate the derivative of the cumulative return reward function with respect to the parameters



Bacon, Pierre-Luc, et al. "The Option-Critic Architecture". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

OC

- Define the option-value function

$$Q_\Omega(s,\omega) = \sum_a \pi_{\omega,\theta}(a \mid s)\, Q_U(s,\omega,a)$$

- Where $Q_U: S \times \Omega \times A \rightarrow R$ is the value of executing an action in the context of a state-option pair:

$$Q_U(s,\omega,a) = r(s,a) + \gamma \sum_{s'} \mathrm{P}(s' \mid s,a)\, U(\omega,s')$$

- The function $U: \Omega \times S \rightarrow R$ is called the option-value function upon arrival

$$U(\omega,s') = (1-\beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \beta_{\omega,\vartheta}(s')V_\Omega(s')$$

Bacon, Pierre-Luc, et al. "The Option-Critic Architecture". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

OC

- Theorem 1 (Intra-Option Policy Gradient Theorem). Given a set of Markov options with stochastic intra-option policies differentiable in their parameters θ, the gradient of the expected discounted return with respect to θ and initial condition (s_0, w_0) is:

$$\sum_{s,\omega} \mu_\Omega(s, \omega \mid s_0, \omega_0) \sum_a \frac{\partial \pi_{\omega,\theta}(a \mid s)}{\partial \theta} Q_U(s, \omega, a)$$

- Theorem 2 (Termination Gradient Theorem). Given a set of Markov options with stochastic termination functions differentiable in their parameters ϑ, the gradient of the expected discounted return objective with respect to ϑ and the initial condition (s_1, w_0) is:

$$-\sum_{s',\omega} \mu_\Omega(s', \omega \mid s_1, \omega_0) \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} A_\Omega(s', \omega)$$

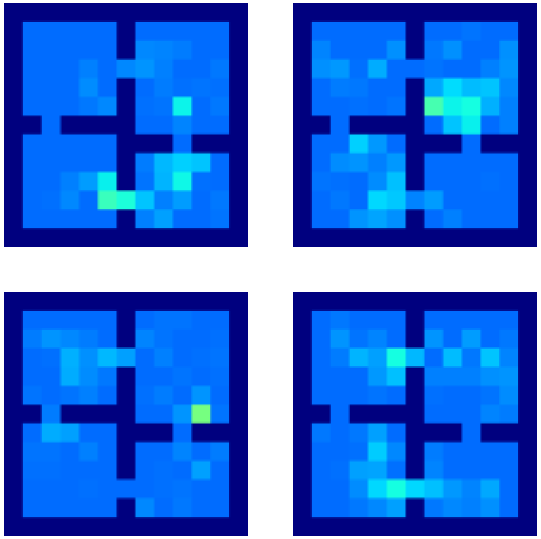
Bacon, Pierre-Luc, et al. "The Option-Critic Architecture". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

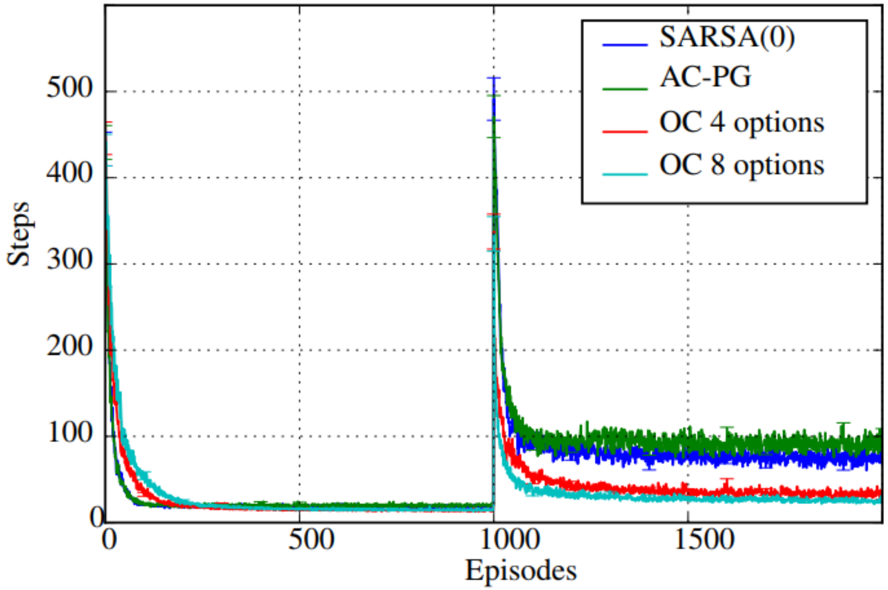
OC

- four-rooms domain

Termination probabilities for the option-critic agent learning with 4 options



After a 1000 episodes, the goal location in the four-rooms domain is moved randomly

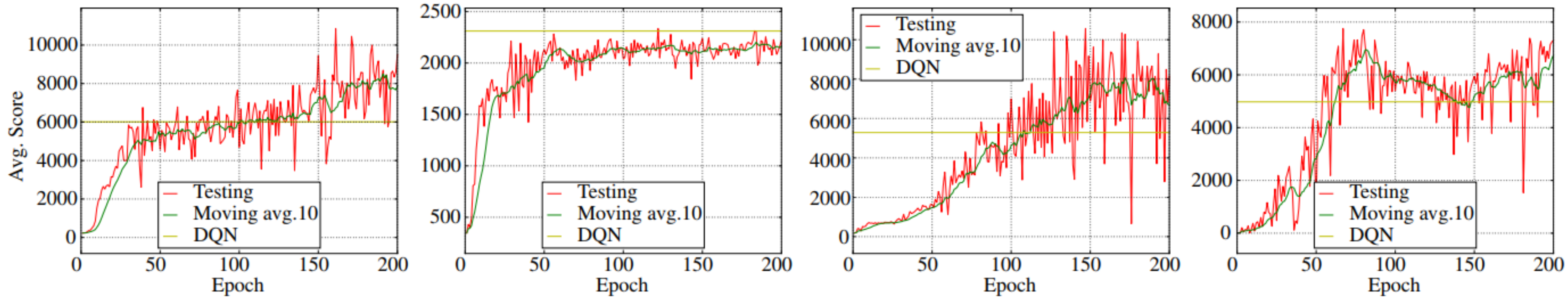


Bacon, Pierre-Luc, et al. "The Option-Critic Architecture". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

Hierarchical RL

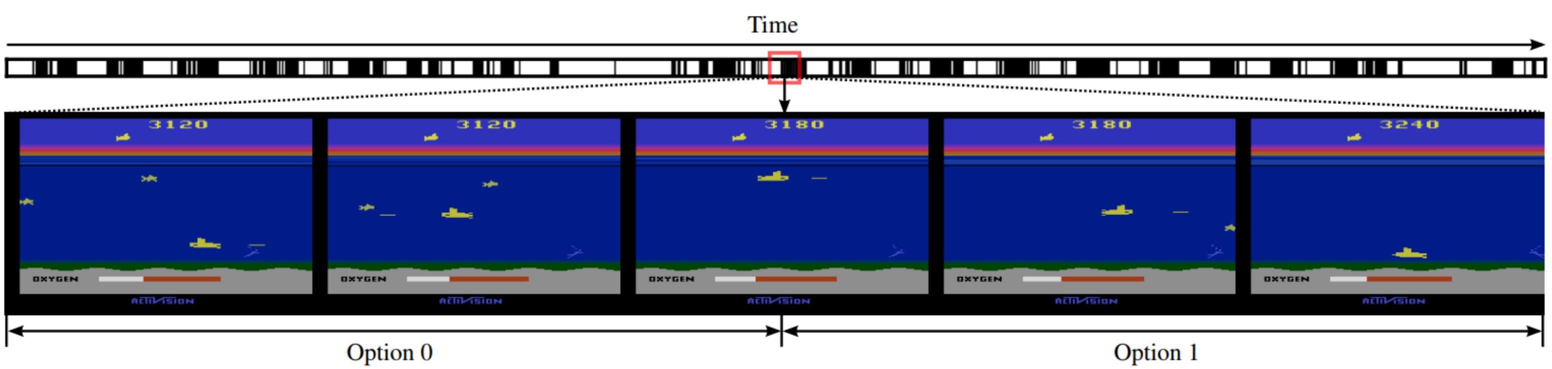
OC

- Arcade Learning Environment
  - Performance



(a) Asterix (b) Ms. Pacman (c) Seaquest (d) Zaxxon

  - Up/down specialization in the solution found by option-critic when learning with 2 options in Seaquest.



Bacon, Pierre-Luc, et al. "The Option-Critic Architecture". *Proceedings of the AAAI Conference on Artificial Intelligence* 31(2017).

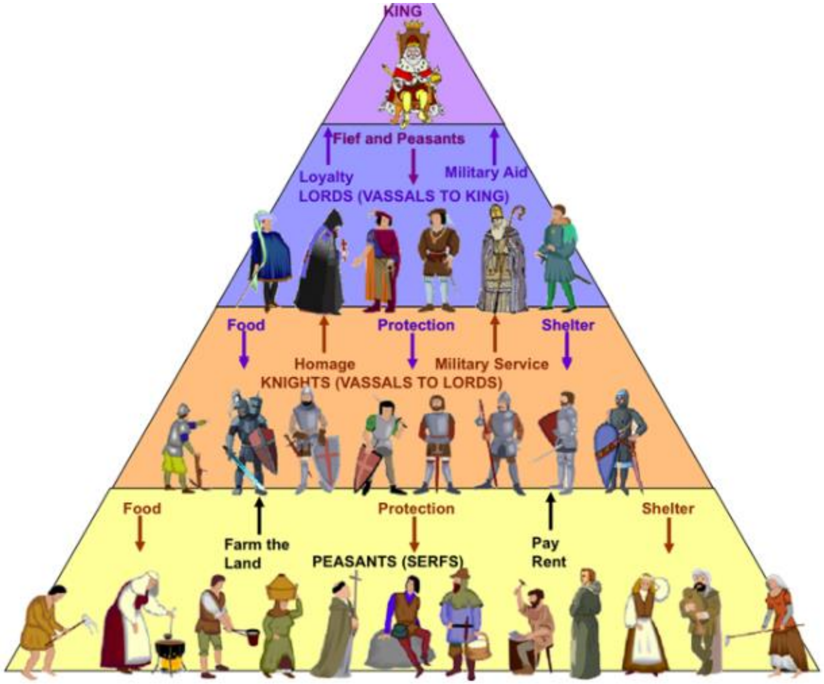
Hierarchical RL

- Discrete abstracted action
    - H-DRLN
    - OC
- Continual abstracted action
    - FuN
    - HIRO

Hierarchical RL

FuN

- Motivation
  - Goals can be generated in a top-down fashion
  - Goal setting can be decoupled from goal achievement

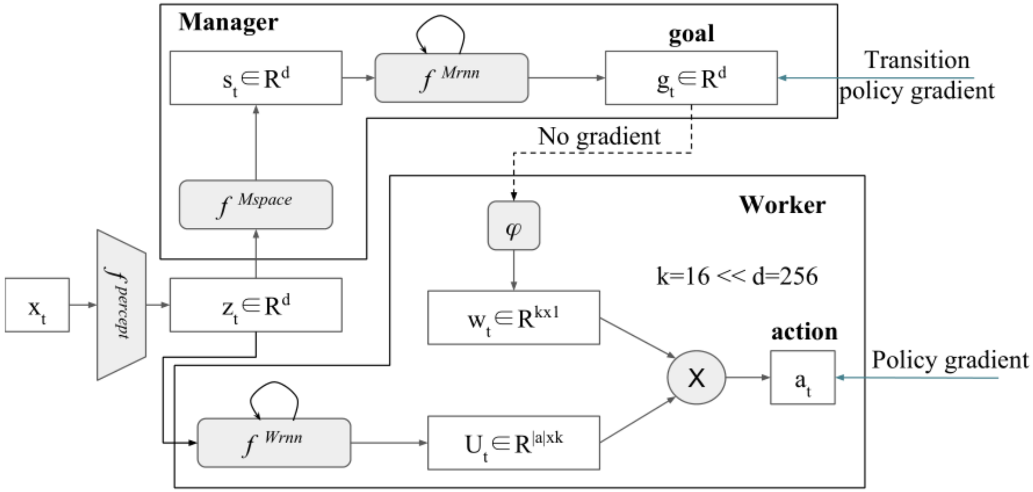


Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning ". *International Conference on Machine Learning* (2017).

Hierarchical RL

FuN

- The Manager and the Worker share a perceptual module
- The Manager's goals g t are trained using an approximate transition policy gradient
- The Worker is then trained via intrinsic reward to produce actions that cause these goal directions to be achieved



$$z_t = f^{\text{percept}}(x_t)$$

$$s_t = f^{Mspace}(z_t)$$

$$h_t^M, \hat{g}_t = f^{Mrnn}(s_t, h_{t-1}^M); g_t = \hat{g}_t / ||\hat{g}_t||;$$

$$w_t = \phi(\sum_{i=t-c}^{t} g_i)$$

$$h^W, U_t = f^{Wrnn}(z_t, h_{t-1}^W)$$

$$\pi_t = SoftMax(U_t w_t)$$

Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning ". *International Conference on Machine Learning* (2017).

Hierarchical RL

FuN

- High level:
  - The update rule:

$$\nabla g_t = A_t^M \nabla_\theta d_{\cos}(s_{t+c} - s_t, g_t(\theta)),$$

Where $A_t^M = R_t - V_t^M(x_t, \theta)$, $d_{cos}(\alpha, \beta) = \alpha^T \beta / (|\alpha||\beta|)$

- Low level:
  - Using intrinsic reward to encourage the Worker to follow the goals

$$r_t^I = 1/c \sum_{i=1}^{c} d_{\cos}(s_t - s_{t-i}, g_{t-i})$$

  - Total reward: $R_t + \alpha R_t^I$
  - The Workers policy is trained by traditional reinforcement learning algorithms.

$$\nabla \pi_t = A_t^D \nabla_\theta \log \pi(a_t | x_t; \theta)$$

Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning ". *International Conference on Machine Learning* (2017).

Hierarchical RL

FuN

- Transition Policy Gradients
  - The high-level policy can be composed with the transition distribution to give a 'transition policy'

$$\pi^{TP}(s_{t+c}|s_t) = p(s_{t+c}|s_t, \mu(s_t, \theta))$$

  - The policy gradient theorem can be applied to the transition policy π^{TP}, so as to find the performance gradient with respect to the policy parameters

$$\nabla_\theta \pi_t^{TP} = \mathbb{E}\left[(R_t - V(s_t))\nabla_\theta \log p(s_{t+c}|s_t, \mu(s_t, \theta))\right]$$

  - FuN assumes a particular form for the transition model: that the direction in state-space, $s_{t+c} - s_t$, follows a von Mises-Fisher distribution

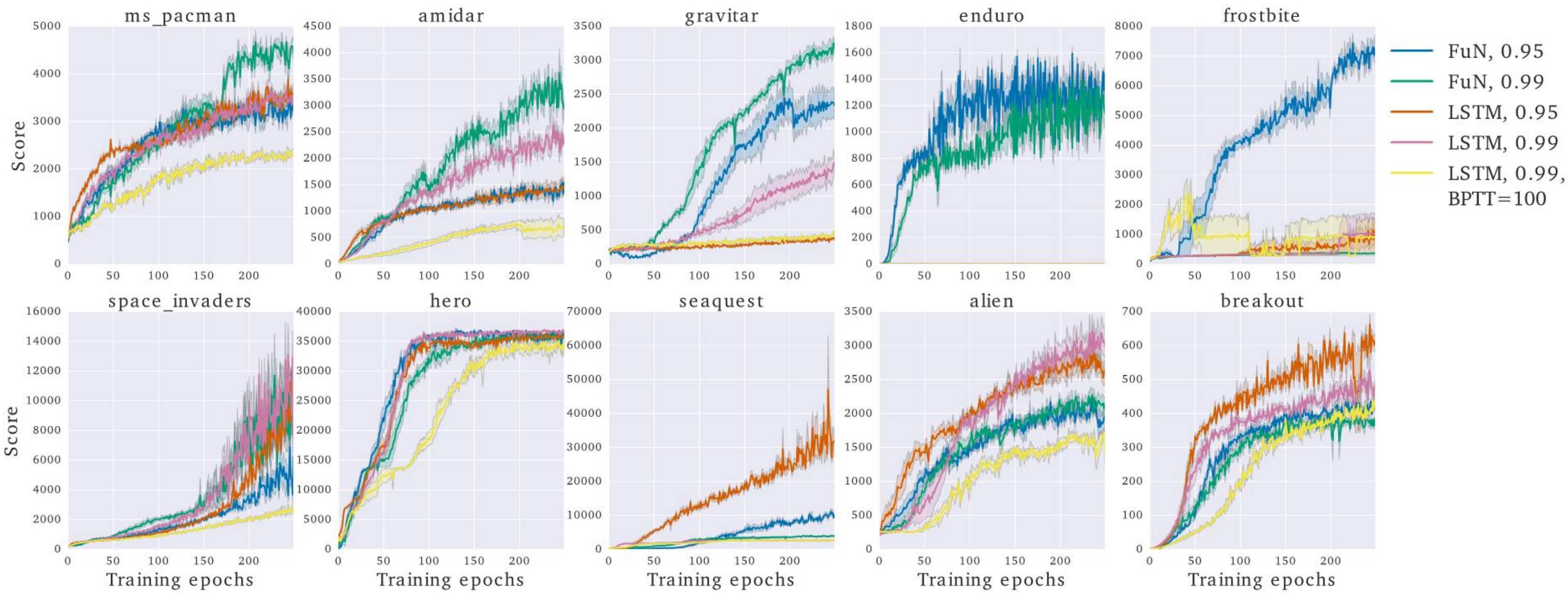
$$p(s_{t+c}|s_t, o_t) \propto e^{d_{\cos}(s_{t+c}-s_t, g_t)}$$

  - If this functional form were indeed correct, then the proposed update heuristic for the Manager, is in fact the proper form for the transition policy gradient

Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning". *International Conference on Machine Learning* (2017).

FuN

- ATARI training curves

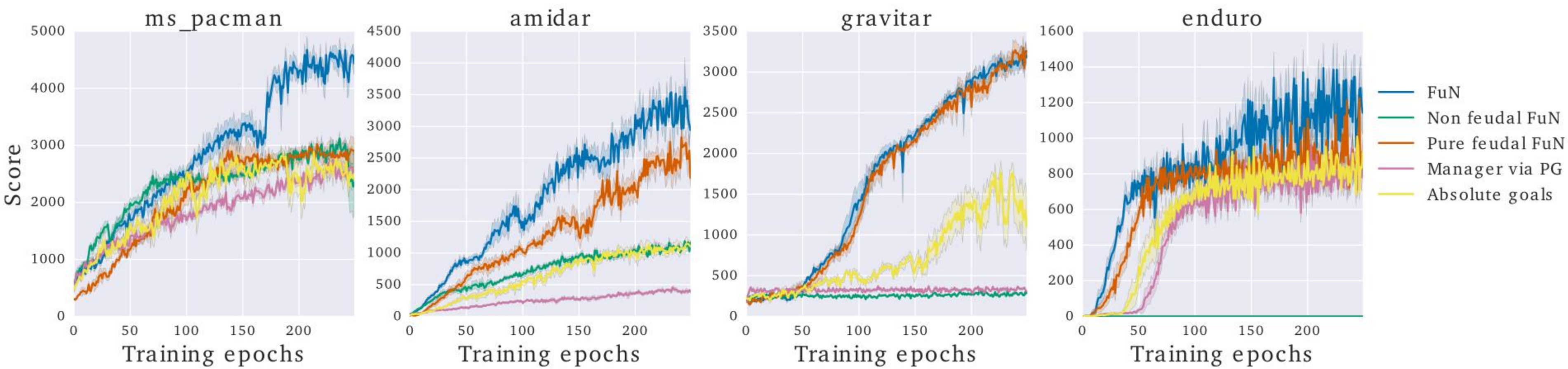


Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning ". *International Conference on Machine Learning* (2017).

Hierarchical RL

FuN

- Ablative analysis
  - **Non feudal FuN**: the Managers output g is trained with gradients coming directly from the Worker and no intrinsic reward is used
  - **Manager via PG:** g is learnt using a standard policy gradient approach
  - **Absolute goals**: g specifies absolute, rather than relative/directional, goals
  - **Pure feudal FuN**: the Worker is trained from the intrinsic reward alone

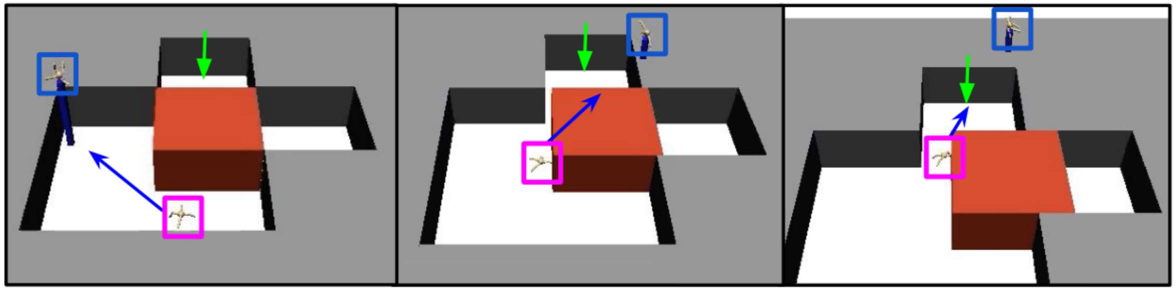


Vezhnevets, Alexander Sasha, et al. "FeUdal Networks for Hierarchical Reinforcement Learning ". *International Conference on Machine Learning* (2017).

Hierarchical RL

HIRO

- Motivation
  - A successful policy must perform a complex sequence of directional movement



  - Off-policy methods are generally more efficient than off-policy methods, but also face another challenge that is unique to HRL
- Main ideas
  - Use states as goals directly, which allows for simple and fast training
  - Use off-policy training with novel off-policy correction, which is extremely sample-efficient

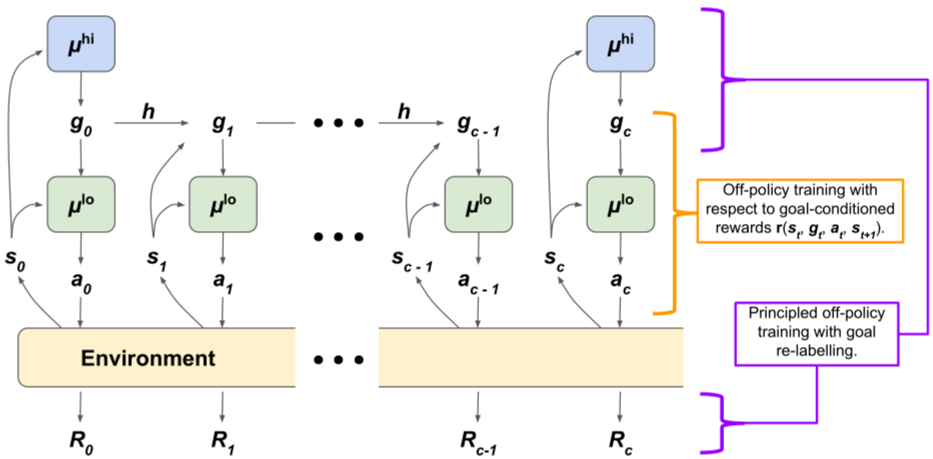
Nachum, Ofir, et al. " Data-Efficient Hierarchical Reinforcement Learning". *32nd Conference on Neural Information Processing Systems* (2018).

HIRO

- Low level
  - The rewards for training the lower-level policy is defined as:

$$r(s_t, g_t, a_t, s_{t+1}) = -||s_t + g_t - s_{t+1}||_2.$$

  - All positional observations are used as the representation for g_t, without distinguishing between the (x,y,z) root position or the joints, making for a generic and broadly applicable choice of goal space



1. Collect experience $s_t, g_t, a_t, R_t, \ldots$.
2. Train μ^{lo} with experience transitions $(s_t, g_t, a_t, r_t, s_{t+1}, g_{t+1})$ using g_t as additional state observation and reward given by goal-conditioned function $r_t = r(s_t, g_t, a_t, s_{t+1}) = -||s_t + g_t - s_{t+1}||_2$.
3. Train μ^{hi} on temporally-extended experience $(s_t, \tilde{g}_t, \sum R_{t:t+c-1}, s_{t+c})$, where $\tilde{g}_t$ is re-labelled high-level action to maximize probability of past low-level actions $a_{t:t+c-1}$.
4. Repeat.

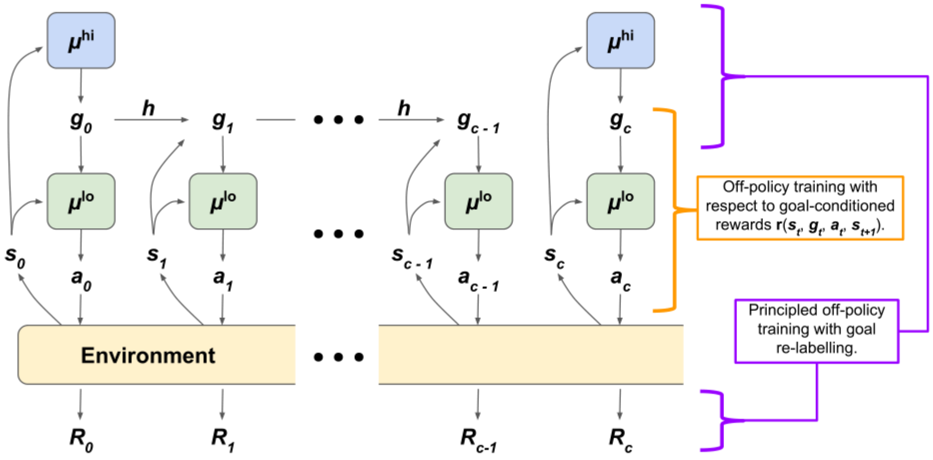
Nachum, Ofir, et al. " Data-Efficient Hierarchical Reinforcement Learning". *32nd Conference on Neural Information Processing Systems* (2018).

Hierarchical RL

HIRO

- High level
  - Re-labeling the high-level transition $(s_t, g_t, \sum R_{t:t+c-1}, s_{t+c})$ with a different high-level action $\tilde{g}_t$ chosen to maximize the probability $\mu^{lo}(a_{t:t+c-1}|s_{t:t+c-1}, \tilde{g}_{t:t+c-1})$
  - Most RL algorithms will use random action-space exploration to select actions, and the log probability may be computed as:

$$\log \mu^{lo}(a_{t:t+c-1}|s_{t:t+c-1}, \tilde{g}_{t:t+c-1}) \propto -\frac{1}{2}\sum_{i=t}^{t+c-1} ||a_i - \mu^{lo}(s_i, \tilde{g}_i)||_2^2 + \text{const.}$$



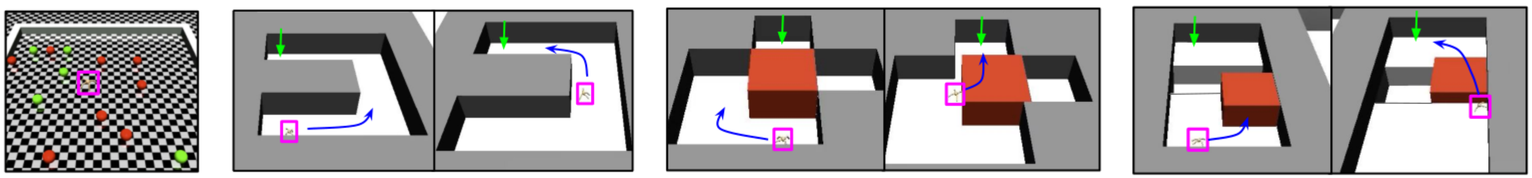
1. Collect experience $s_t, g_t, a_t, R_t, \ldots$.
2. Train μ^{lo} with experience transitions $(s_t, g_t, a_t, r_t, s_{t+1}, g_{t+1})$ using g_t as additional state observation and reward given by goal-conditioned function $r_t = r(s_t, g_t, a_t, s_{t+1}) = -||s_t + g_t - s_{t+1}||_2$.
3. Train μ^{hi} on temporally-extended experience $(s_t, \tilde{g}_t, \sum R_{t:t+c-1}, s_{t+c})$, where $\tilde{g}_t$ is re-labelled high-level action to maximize probability of past low-level actions $a_{t:t+c-1}$.
4. Repeat.

Nachum, Ofir, et al. " Data-Efficient Hierarchical Reinforcement Learning". *32nd Conference on Neural Information Processing Systems* (2018).

Hierarchical RL

HIRO

- Experiments
  - Scene screenshot



  - performance

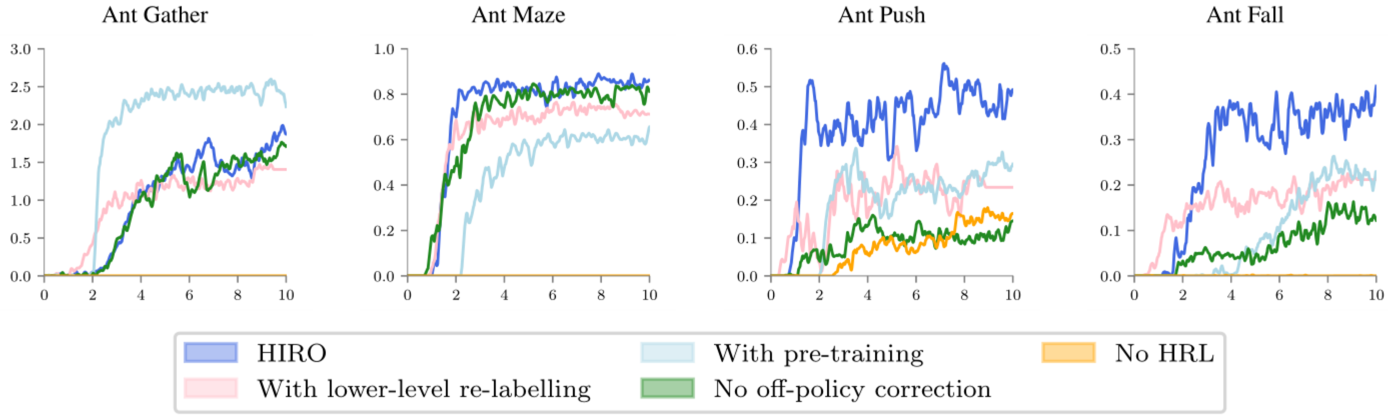
	Ant Gather	Ant Maze	Ant Push	Ant Fall
HIRO	**3.02±1.49**	**0.99±0.01**	**0.92±0.04**	**0.66±0.07**
FuN representation	0.03 ± 0.01	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
FuN transition PG	0.41 ± 0.06	0.0 ± 0.0	0.56 ± 0.39	0.01 ± 0.02
FuN cos similarity	0.85 ± 1.17	0.16 ± 0.33	0.06 ± 0.17	0.07 ± 0.22
FuN	0.01 ± 0.01	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SNN4HRL	1.92 ± 0.52	0.0 ± 0.0	0.02 ± 0.01	0.0 ± 0.0
VIME	1.42 ± 0.90	0.0 ± 0.0	0.02 ± 0.02	0.0 ± 0.0

Nachum, Ofir, et al. " Data-Efficient Hierarchical Reinforcement Learning". *32nd Conference on Neural Information Processing Systems* (2018).

Hierarchical RL

HIRO

- Ablative Analysis
  - **With lower-level re-labelling**: increase the amount of data available to an agent trained using a parameterized reward (the lower-level policy in our setup) by re-labeling experiences with randomly sampled goals.
  - **With pre-training**: pre-train the lower-level policy for 2M steps (using goals sampled from a Gaussian) before freezing it and training the higher-level policy alone
  - **No off-policy correction**
  - **No HRL**



Nachum, Ofir, et al. " Data-Efficient Hierarchical Reinforcement Learning". *32nd Conference on Neural Information Processing Systems* (2018).