Hierarchical RL with Option Learning

Chunlok Lo, Eric Liu, Matthew Gombolay

04/24/19

Outline

- **Introduction**
- Background
  - Hierarchical Reinforcement Learning
  - Unsupervised Option Learning
- Experiments
  - Our approach
  - Results

Introduction

We want robot to make hamburgers.

Things agent can do:
{up, down, left, right, pick up, drop off, interact}
No idea whether any of these things are useful at all in achieving the goal.

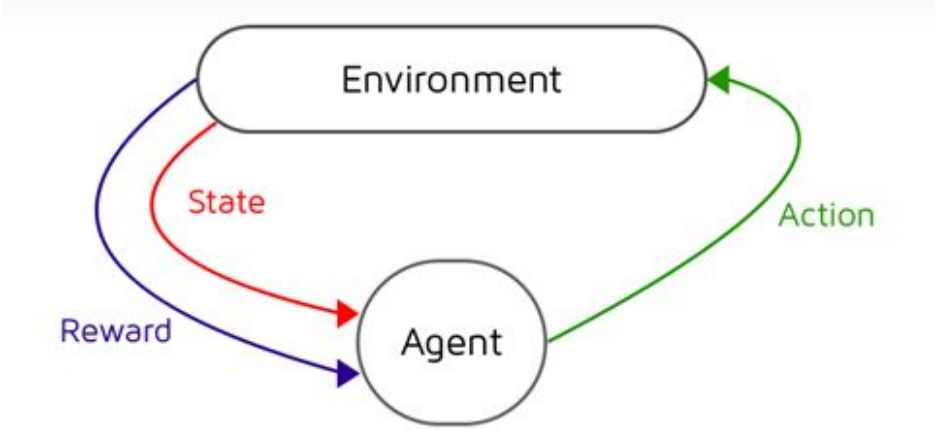
Markov Decision Process (S, A, T, R)

S - Set of states
A - Set of actions
T - Transition probabilities P(s' | s, a)
R - Reward R(s, a)

$\pi_\theta(s) = a$

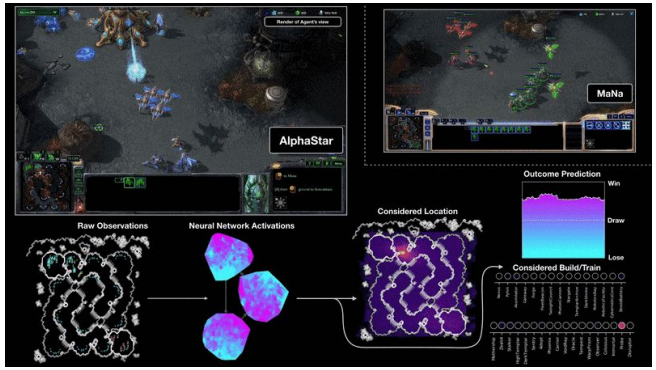
Solution: Optimal policy

$$\theta^\star = \arg\max_\theta E_{(\mathbf{s},\mathbf{a})\sim p_\theta(\mathbf{s},\mathbf{a})}[r(\mathbf{s},\mathbf{a})]$$



State of the Art

Starcraft 2



AlphaStar beats professional Starcraft II player 10-0

Dota 2

OpenAI Five beats professional Dota 2 team 2-0

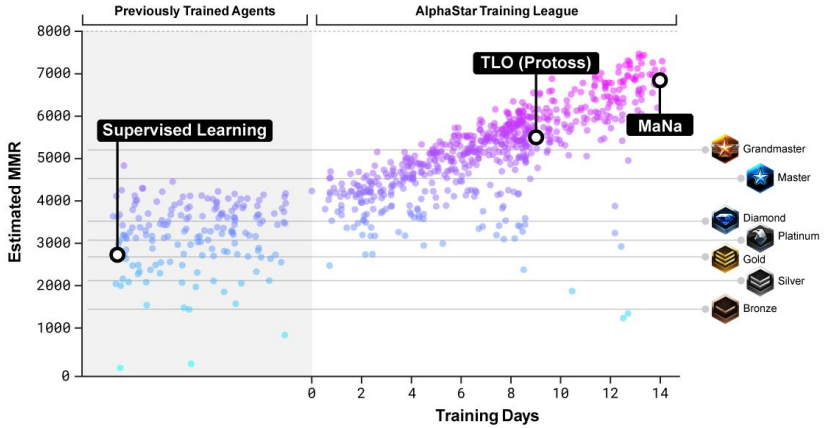
Objective: Win the Game

Only thing that matters is that you defeat the opponent. The agent need to make many decisions one can make along the way that is not always clear how it contributes to the final victory.

State of the art reinforcement learning agents are able to beat human professionals in complex games where thousands of steps are needed and steps to win is uncertain.

Learning With Prior Knowledge

Imitation Learning (AlphaStar)



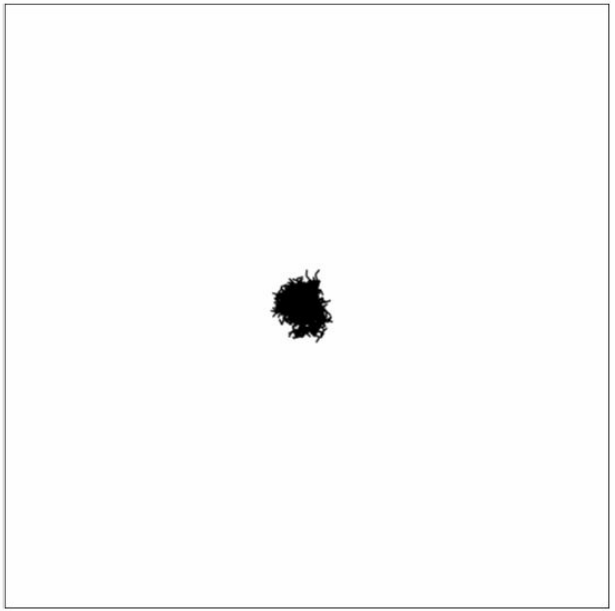
Reward Shaping (OpenAI Five)

Individual	Weight	Awarded for
Experience	0.002	Per unit of experience.
Gold	0.006	Per unit of gold gained[1].
Mana	0.75	Mana (fraction of total).
Hero Health	2.0	Gaining (or losing) health[2].
Last Hit	0.16	Last Hitting an enemy creep[3].
Deny	0.2	Last Hitting an allied creep[3].
Kill	-0.6	Killing an enemy hero[3].
Death	-1.0	Dying.

It is common for state of the art results to rely on providing prior knowledge on how to complete the task and encoding that into the agent. However, this is sometimes difficult to do.

Learning Without Prior Knowledge

Naive Random Walk (Howie Choset)



Common exploration strategies:
ε-greedy
Entropy maximization

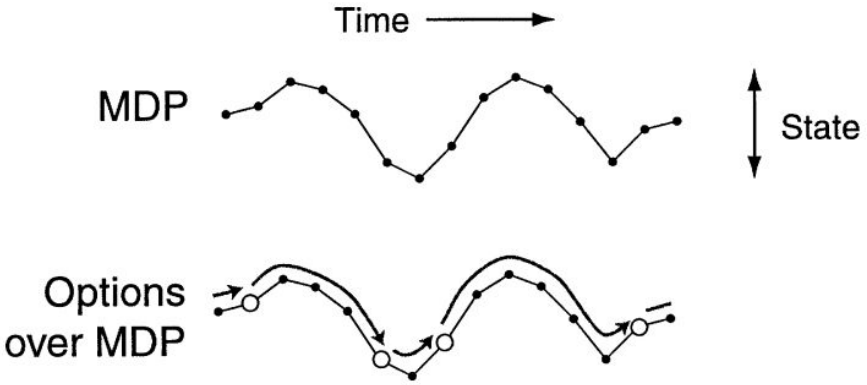
Exploration through randomness does not work in a sparse reward environment where it is often too rare for the agent to stumble upon the reward by performing random actions

Outline

- Introduction
- **Background**
  - Hierarchical Reinforcement Learning
  - Unsupervised Option Learning
- Experiments
  - Our approach
  - Results

Proposed Approach

- Action space is too **granular** for sufficient random exploration
- Can we operate at a higher **temporal scale**? Options
- Less “steps” to goal mean easier problem.



Sutton et. al Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning

Having larger and meaningful sub-policies and exploring over those policies should be an easier task, with less random decisions needed to encounter the goal.

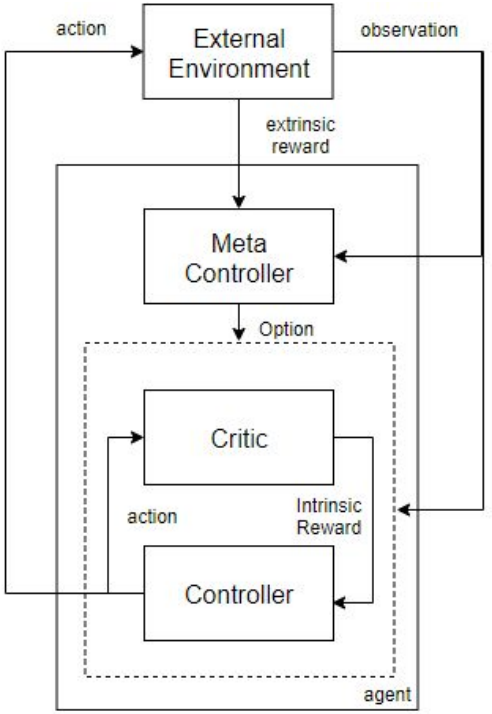
Hierarchical Reinforcement Learning

h-DQN(hierarchical DQN) framework proposed by Kulkarni et al.

Meta Controller - High level policy over **options,** rewarded by environment
Controller - Low level policy over **actions,** rewarded by internal critic.

Internal Critic provides intrinsic reward to the low level controller: {1, 0} for goal achieved or not.

Problem: Options are hand specified. Can we learn an internal critic instead that will give us a diverse set of goals?



Unsupervised Option Learning

Gregor et al. (2016) - Variational Intrinsic Control (VIC)
Eysenbach et al. (2018) - Diversity Is All You Need (DIAYN)

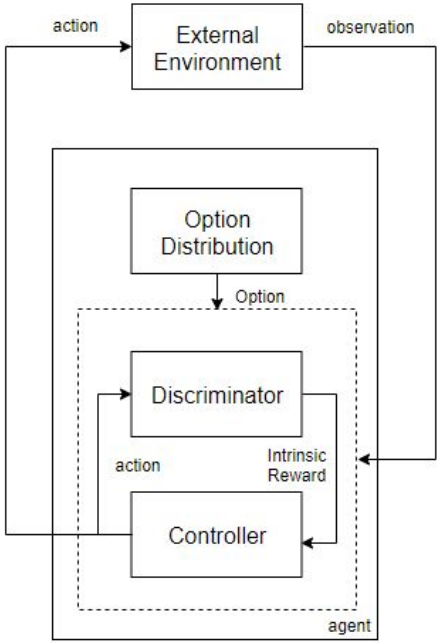
Learn **automatically** a **diverse** set of **options** without extrinsic reward
Develop options which create trajectories that are distinguishable by a **discriminator**

There are existing work in learning an internal critic to produce a diverse set of options.

Details on Diversity Is All You Need (DIAYN)

Algorithm 1: DIAYN

while *not converged* **do**
  Sample skill $z \sim p(z)$ and initial state $s_0 \sim p_0(s)$
  for $t \leftarrow 1$ **to** *steps_per_episode* **do**
    Sample action $a_t \sim \pi_\theta(a_t \mid s_t, z)$ from option.
    Step environment: $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$.
    Compute $q_\phi(z \mid s_{t+1})$ with discriminator.
    Set skill reward $r_t = \log q_\phi(z \mid s_{t+1}) - \log p(z)$
    Update policy (θ) to maximize r_t with SAC.
    Update discriminator (ϕ) with SGD.



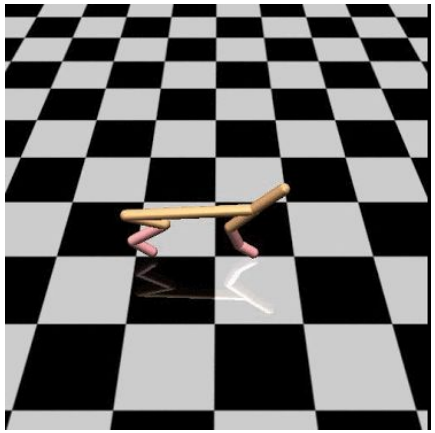
VIC: only rewards the final state with

$$r_f = \log q(z|s_o, s_f) - \log p(z)$$

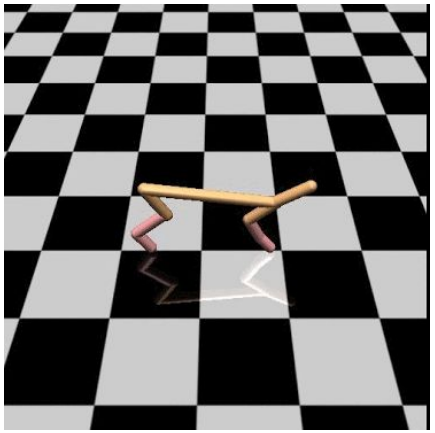
DIAYN and VIC uses a discriminator to rewards trajectory based on their distinguishability between skills, leading to a diverse set of options.

DIAYN Results

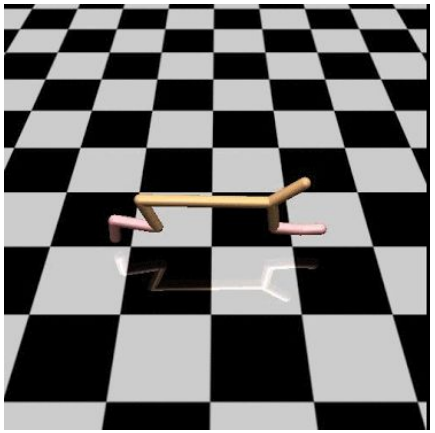
Running



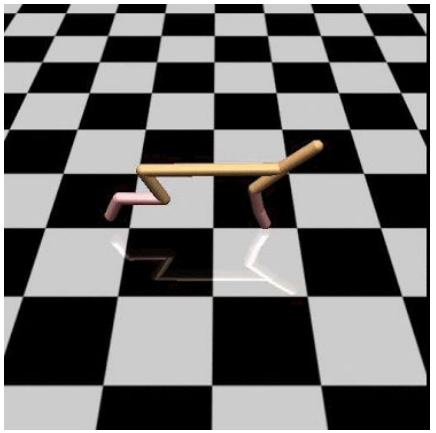
Walking



Backwards



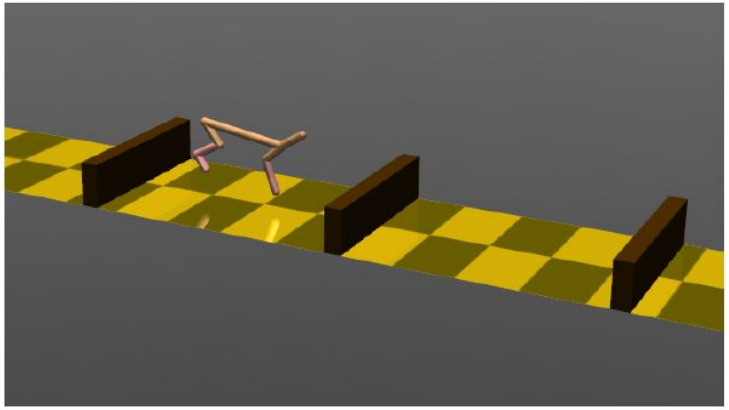
Flip

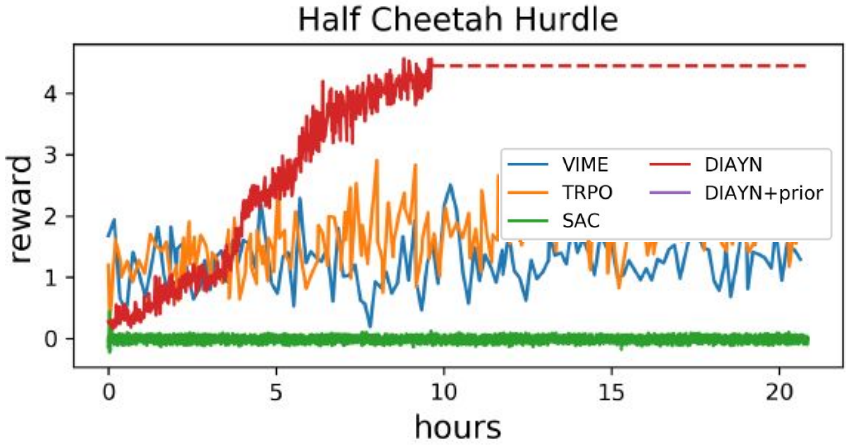


DIAYN is able to obtain learn qualitatively different options without external rewards.

DIAYN with HRL

Eysenbach et al. tested simple extension to HRL by learning a meta-controller over fixed policies learned by DIAYN

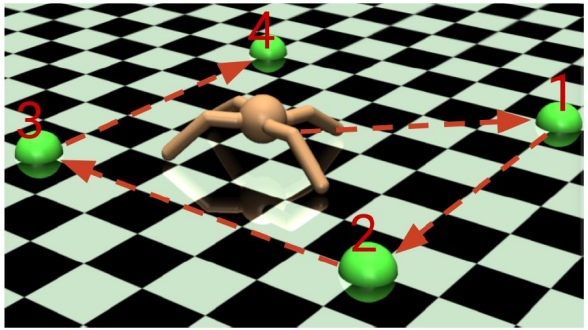


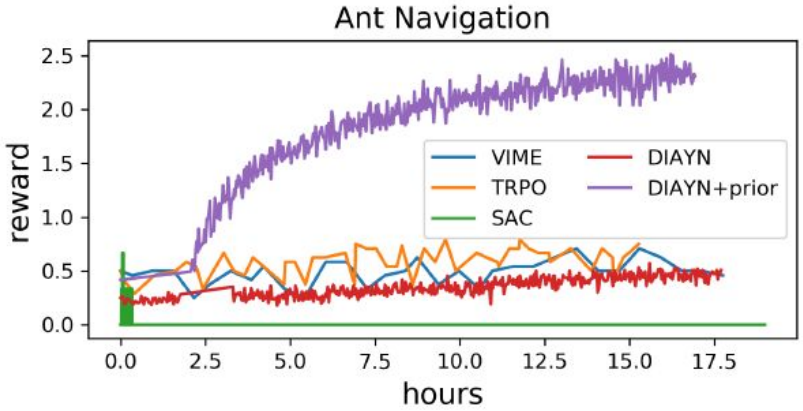


DIAYN outperforms various SOTA reinforcement learning algorithms in half cheetah hurdle

DIAYN with HRL

Eysenbach et al. tested simple extension to HRL by learning a meta-controller over fixed policies learned by DIAYN



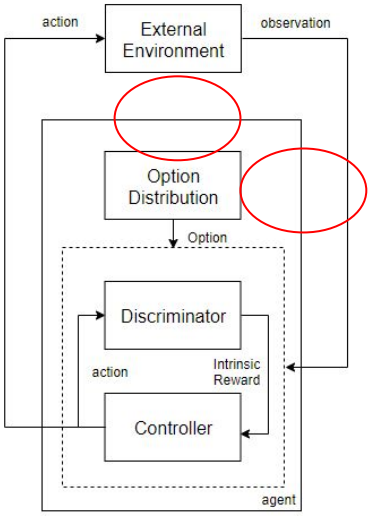


DIAYN does not do so well on ant navigation without prior knowledge.
The results of DIAYN with HRL is mixed but generally positive.

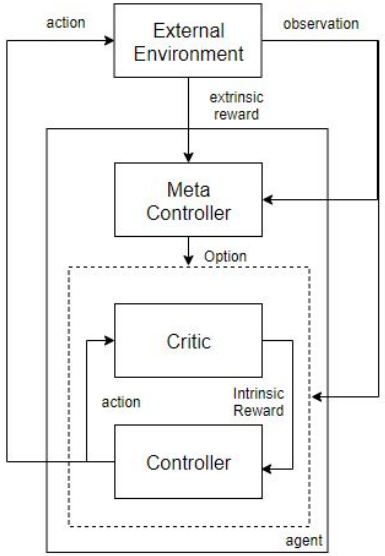
Unsupervised Option Learning and HRL

Option learning as formulated in VIC and DIAYN and the HRL framework is very similar.

Option Learning



HRL



Hypothesis: We should be able to introduce extrinsic reward back into the loop of DIAYN and learn the meta controller at the same time as the discriminator and controller.

Outline

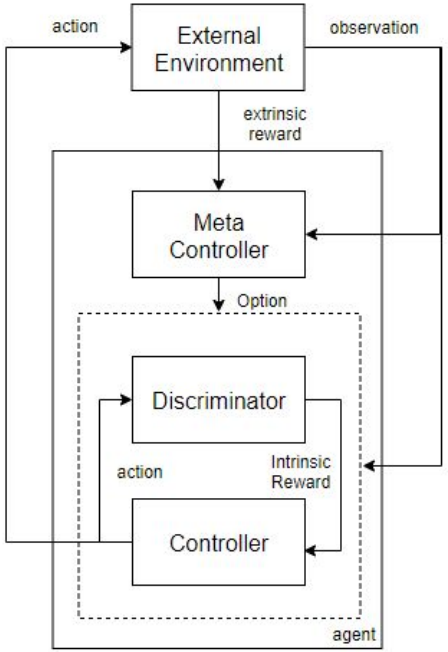
- Introduction
- Background
  - Hierarchical Reinforcement Learning
  - Unsupervised Option Learning
- **Experiments**
  - Our approach
  - Results

Our Approach

Learn the meta controller **alongside** the discriminator and policy
Terminate option after finite steps and rechose option at that state.

Take VIC's intrinsic reward formulation (inference based only on **starting** and **final** state) as it matches up more with h-DQN's goal formulation

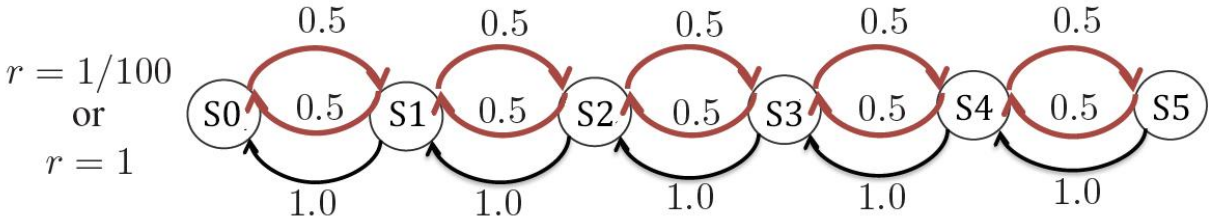
Use policy gradient implementation of meta controller and controller as baseline



Experimental Setup

Take baseline example from Kulkarni et al. with hand specified goals.

Reward:
1 if s_6 then s_1
0.01 if s_1 but no s_6



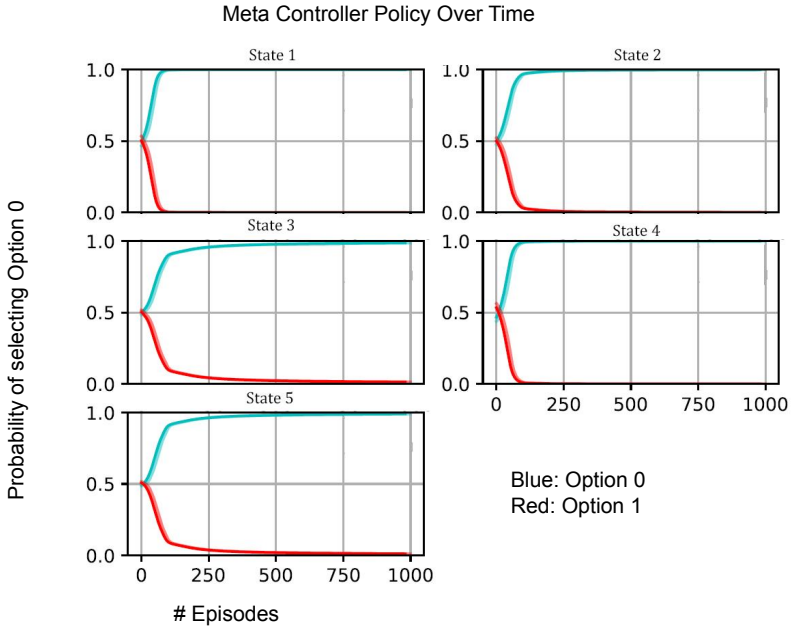
Terminate on S0
Probability of going right when the agent selects the red direction is adjustable as 0.0 < p < 1.0

The agent needs to learn the option of visiting s_6 despite the closer but suboptimal reward of going straight to s_1

Simultaneous Learning

Next experiment: train the meta controller with discriminator and controller.

- Meta controller quickly prefers one option

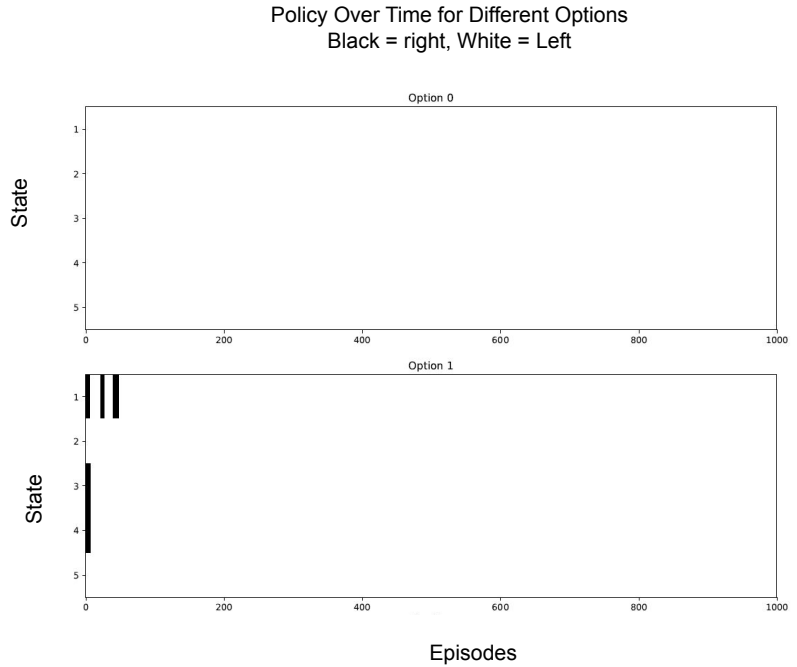


The combined training approach is unable to learn diverse options because the meta controller can prefer the sub-optimal action and never train the other options

Simultaneous Learning

Next experiment: train the meta controller with discriminator and controller.

- Meta controller quickly prefers one option
- Option 0 is optimal in obtaining sub-optimal reward

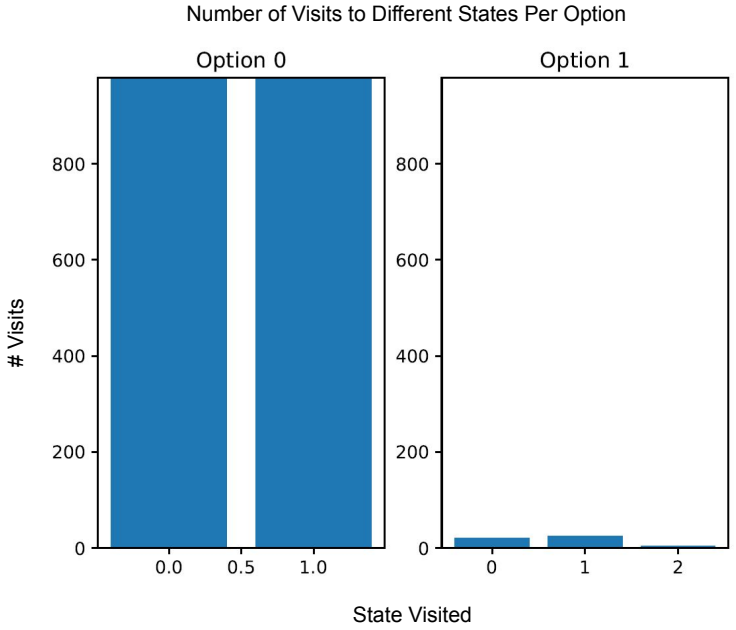


The combined training approach is unable to learn diverse options because the meta controller can prefer the sub-optimal action and never train the other options

Simultaneous Learning

Next experiment: train the meta controller with discriminator and controller.

- Meta controller quickly prefers one option
- Option 0 is optimal in obtaining sub-optimal reward
- Option 1 is never trained

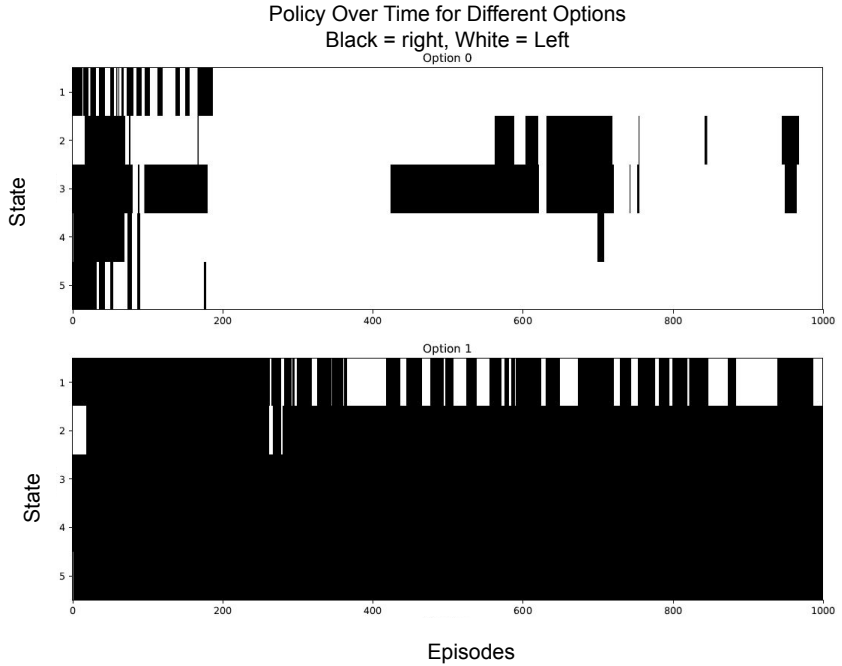


The combined training approach is unable to learn diverse options because the meta controller can prefer the sub-optimal action and never train the other options

Learning Options with DIAYN

As a baseline we tested our approach without learning the meta controller by fixing the skill distribution to uniform random.

Show one with change in policy over time

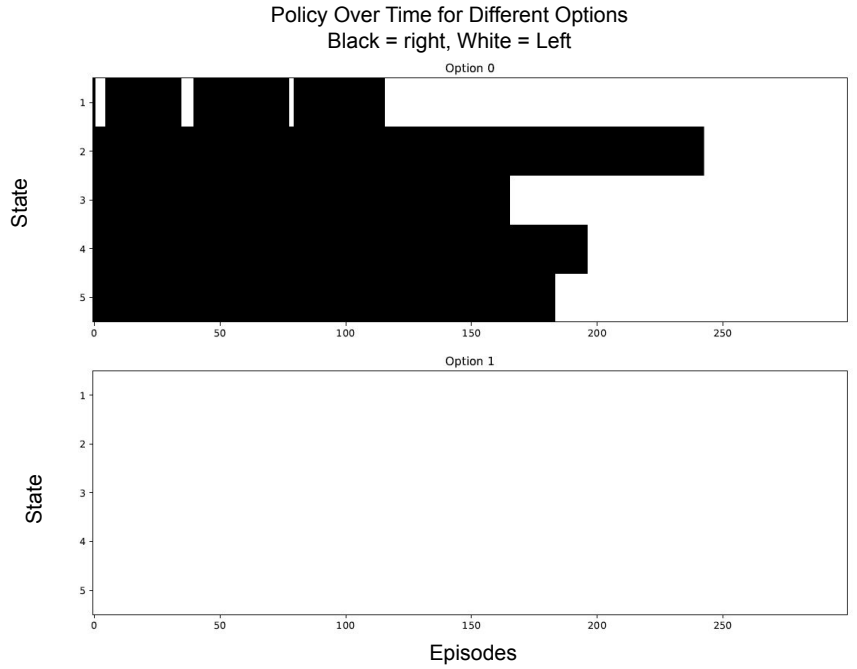


As a baseline, VIC/DIAYN is able to learn qualitatively different options with only 2 options. One that head towards state 6, and another that doesn't.

Simultaneous Learning with Pre-Trained Options

Pre-trained with optimal options, one going right and one going left.

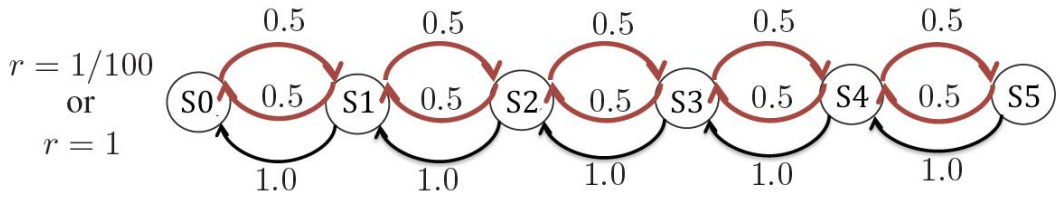
We see that optimal action maintains optimal behavior for a while then converge to a sub-optimal policy.

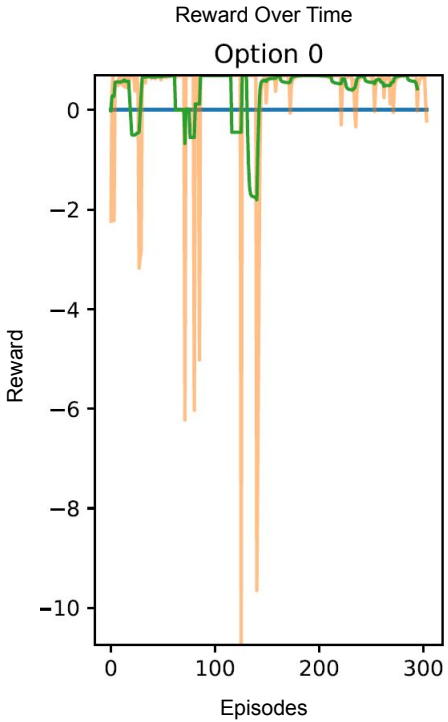


The optimal pretrained options becomes sub-optimal as we train simultaneously.

Why did action policy change in the first place?

Option receives negative reward when it reaches s_1 due to the stochastic transition probabilities



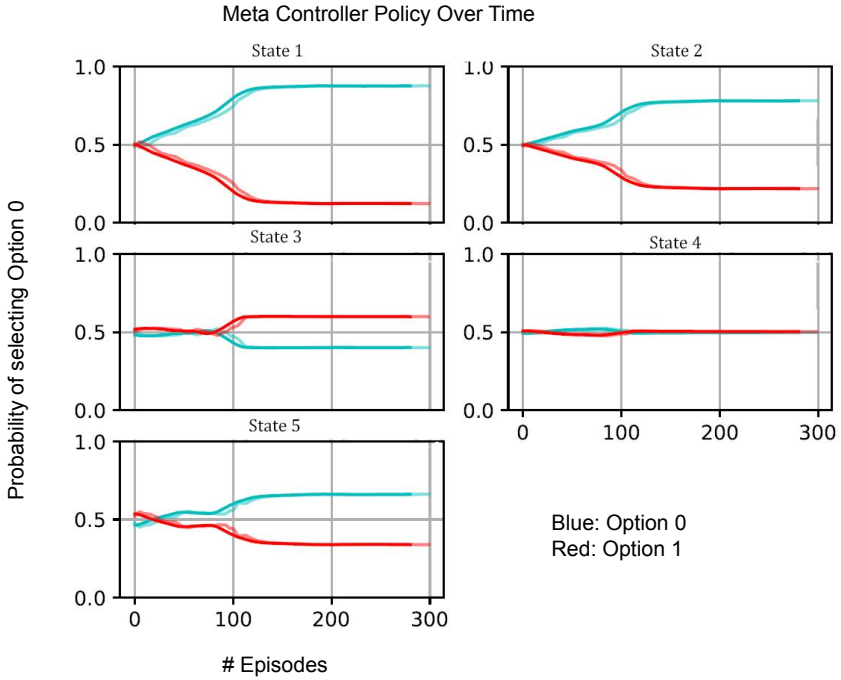


Problem is not fixed with less stochasticity because discriminator becomes better at inferring options

Action policy is prone to change when the options can probabilistically overlap with another option

Why did option become stuck as suboptimal?

The meta controller learns to choose the "correct" option more often.

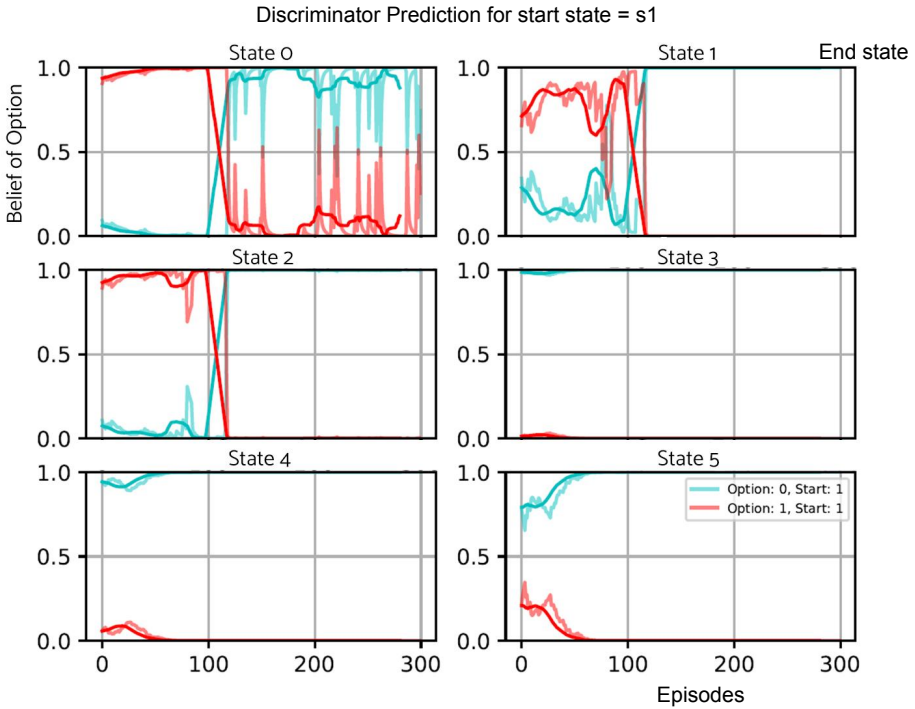


As the meta controller converges to an optimal option, it selects mostly a single option

Why did option become stuck as suboptimal?

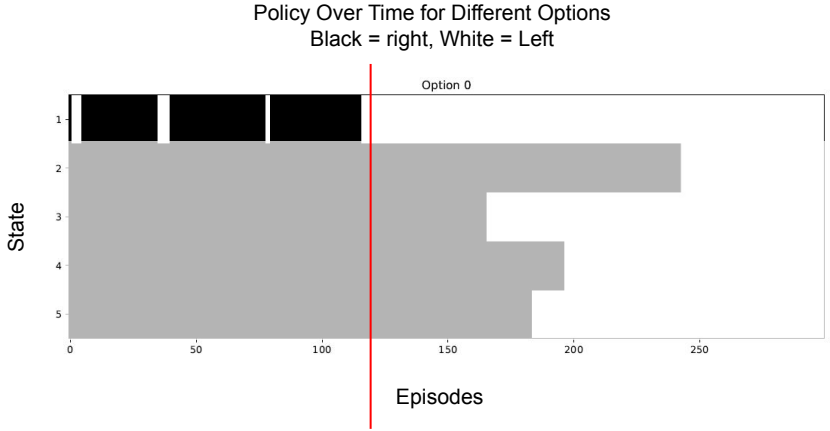
The meta controller learns to choose the “correct” option more often.

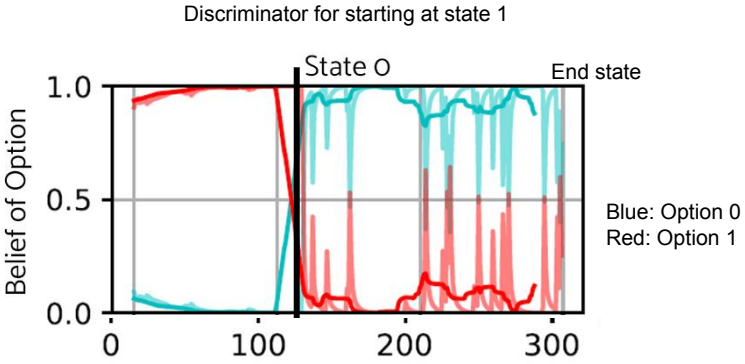
Because overlaps in state distribution of options, discriminator becomes useless.



Convergence of the meta controller creates a useless discriminator as it guesses the chosen option for all final states.

Why did option become stuck as suboptimal?



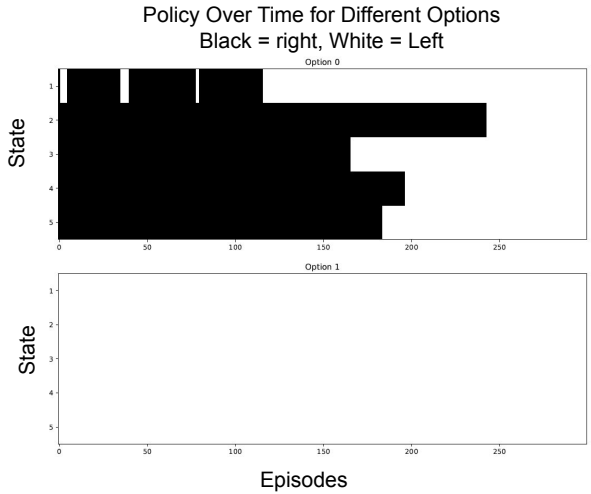


Note: Don't worry about policies for states other than state 1 after option decides to go left. No signals are obtained for the other states other than state 1 and state 0 after the policy decides to go left

When the discriminator sufficiently rewards option 0 for ending in s_0, option does not return to its original distribution.

Contributions

- We experimented with combining HRL with unsupervised skill learning approaches
- We showed that combining discriminator based option learning approaches can lead to an irreversible collapse of options when trained with a high level policy over options.



Future Steps

- Consider intrinsic rewards not based on disciminator.
- State visitation-based rewards (ex. RND, pseudo-counts).
- Dynamic prediction-based rewards (ex. ICM)
  Pros: Generalization to unseen environments
  Cons: Predicting dynamics for stochastic environment

Questions?