POSTECH POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY Machine Learning Group

Stochastic Neural Networks For HRL

Carlos Florensa, Yan Duan, Pieter Abbeel

Conference paper at ICLR 2017

Youngseok Yoon

Before beginning…

- Meta Learning Shared Hierarchies
  - https://sites.google.com/site/mlshsupplementals/

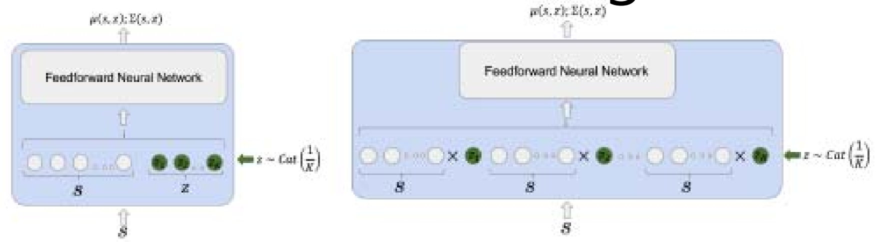
Contents

1.1) Constructing the pre-training env

- Letting the agent freely interact with the environment in a minimal setup.
- Rather than setting different reward to the desired skills, use generic single reward as the only reward signal to guide skill learning.
- But it is inefficient that training each policy from scratch.
- So,
  - First issue: using Stochastic Neural Networks as policies
  - Second issue: adding an information-theoretic regulrizer.

1.2) SNN (Stochastic Neural Networks)

- To learn several skills at the same time, propose to use Stochastic Neural Networks (SNNs).
- Use simple categorical distributions with uniform weights for the latent variables.
- K is the hyper parameter that upper bounds of #skills.
- Allows flexible weight-sharing.



(a) Concatenation (b) Bilinear integration

Figure 1: Different architectures for the integration of the latent variables in a FNN

- To further encourage the diversity of skills learned by the SNN, introduce an information theoretic regularizer.

1.3) Information-Theoretic Regularization

- Add an additional reward bonus:
  - Mutual information (MI) between latent variable and current state.
- Let current state as $c = (x, y)$, center of mass of mobile robot,
  - $MI: I(Z; C) = H(Z) - H(Z|C)$
  - $H(Z)$ is constant, since pdf of Z (latent variable) is fixed in training.
  - $H(Z|C) = -\mathbb{E}_{z,c} \log p(Z = z|C = c)$, so
  - $R_t^n \leftarrow R_t^n + \alpha_H \log \hat{p}(Z = Z^n|c_t^n)$, n denote estimating factor.
  - Also, to estimate $\hat{p}$, calculate $m_c(z)$ - how many time cell c is visited during latent code z is sampled.
  - $\hat{p}(Z = z|(x, y) \sim \hat{p}(Z = z|c) = \frac{m_c(z)}{\sum_{z\prime} m_c(z\prime)}$

1.4) Learning high-level policies

- How to use learned K skills that is learned during pre-training.
- Freeze them and training high-level policy, that operates by selecting a skill for a fixed step T.
- The factored representation of the state space S^M: S_{agent} and S^M_{rest}.

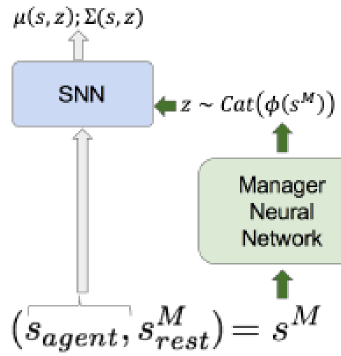


Figure 2: Hierarchical SNN architecture to solve downstream tasks

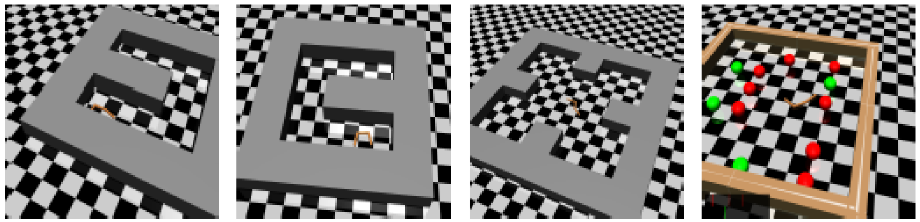
1.5) Policy Optimization

Algorithm 1: Skill training for SNNs with MI bonus

Initialize: Policy π_θ; Latent dimension K;
while *Not trained* **do**
  for $n \leftarrow 1$ **to** N **do**
    Sample $z_n \sim \text{Cat}(\frac{1}{K})$;
    Collect rollout with z_n fixed;
  end
  Compute $\hat{p}(Z = z|c) = \frac{m_c(z)}{\sum_{z'} m_c(z')}$;
  Modify $R_t^n \leftarrow R_t^n + \alpha_H \log \hat{p}(Z = z^n|c_t^n)$;
  Apply TRPO considering z part of the observation;
end

2. Experiments and Results

- Experiments
  - In benchmark by Duan et al. (2016)
    Benchmarking deep reinforcement learning for continuous control.
    - Locomotion + Maze and Locomotion + Food Collection
  - S_{agent}: the robot
  - S^{M}_{rest}: task specific attributes (walls, goals, and sensor readings)

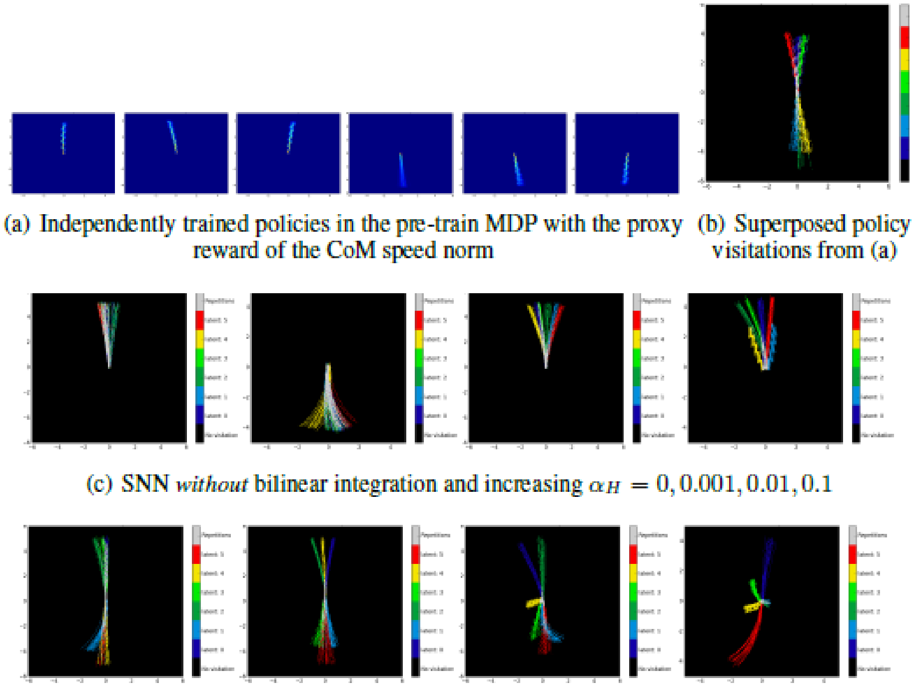


(a) Maze 0 (b) Maze 1 (c) Maze 2 or 3 (d) Food Gather

Figure 3: Illustration of the sparse reward tasks

2. Experiments and Results

- Locomotion Experiments
  - Solely reward speed.
- Skill learning in pre-train (visitation plots in swim learning environment)



(a) Independently trained policies in the pre-train MDP with the proxy reward of the CoM speed norm

(b) Superposed policy visitations from (a)

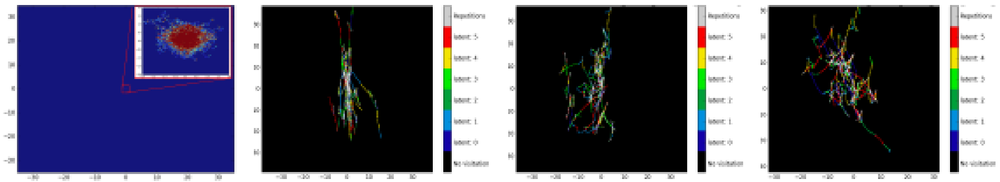
(c) SNN *without* bilinear integration and increasing $\alpha_H = 0, 0.001, 0.01, 0.1$

(d) SNN *with* bilinear integration and increasing $\alpha_H = 0, 0.001, 0.01, 0.1$

Figure 4: Span of skills learn by different methods and architectures

2. Experiments and Results

- Hierarchical use of skills
  a. Exploration drawn from Gaussian (Duan et al. (2016))
  b. It train six policies independently. The policies heavily concentrates on up-down ward.
  c. d. yields a wider coverage of the psace.



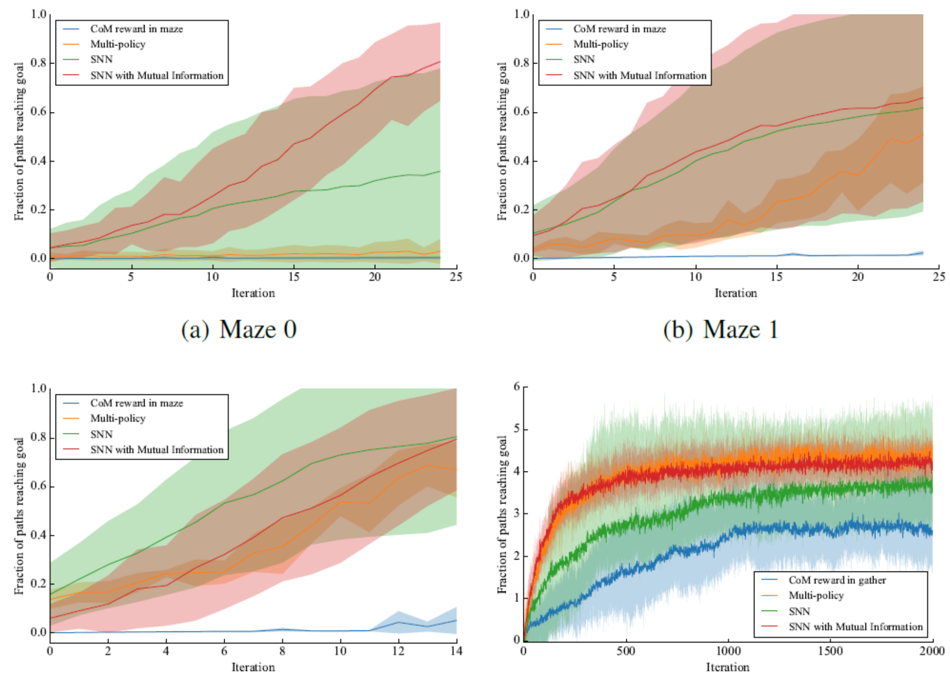
(a) Gaussian noise with covariance $\Sigma = I$ (b) Hierarchy with *Multi-policy* (c) Hierarchy with Bil-SNN $\alpha_H = 0$ (d) Hierarchy with Bil-SNN $\alpha_H = 0.01$

Figure 5: Visitation plots for different randomly initialized architectures (one rollout of 1M steps). All axis are in scale [-30,30] and we added a zoom for the Gaussian noise to scale [-2,2]

2. Experiments and Results

- Mazes and Gather tasks
  - To better baseline, adding to the CoM proxy reward in pre-training.

  a. b. c.
     poor performance, due to the long time-horizon needed to reach the goal. Furthermore, the proxy reward alone does not encourage diversity.



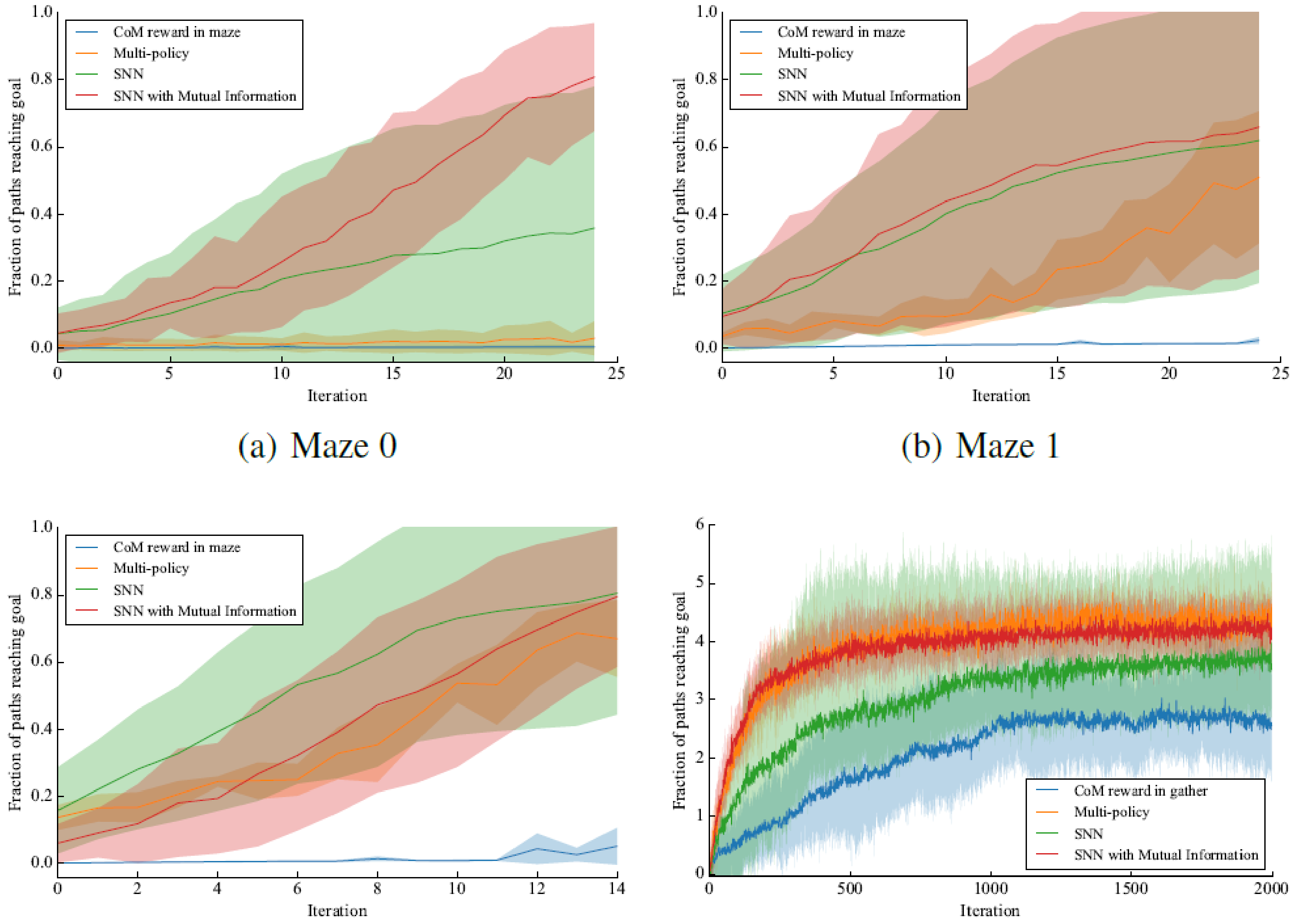
(a) Maze 0 (b) Maze 1

(c) Aggregated results for Mazes 2 and 3 (d) Gather task

Figure 6: Faster learning of the hierarchical architectures in the sparse downstream MDPs

2.

- Ma



(a) Maze 0 (b) Maze 1

(c) Aggregated results for Mazes 2 and 3 (d) Gather task

Figure 6: Faster learning of the hierarchical architectures in the sparse downstream MDPs

2. Experiments and Results

- Mazes and Gather tasks
  d. To fairly compare, experiment in exact set-ting, maximum path length of 500 and batch-size of 50k.

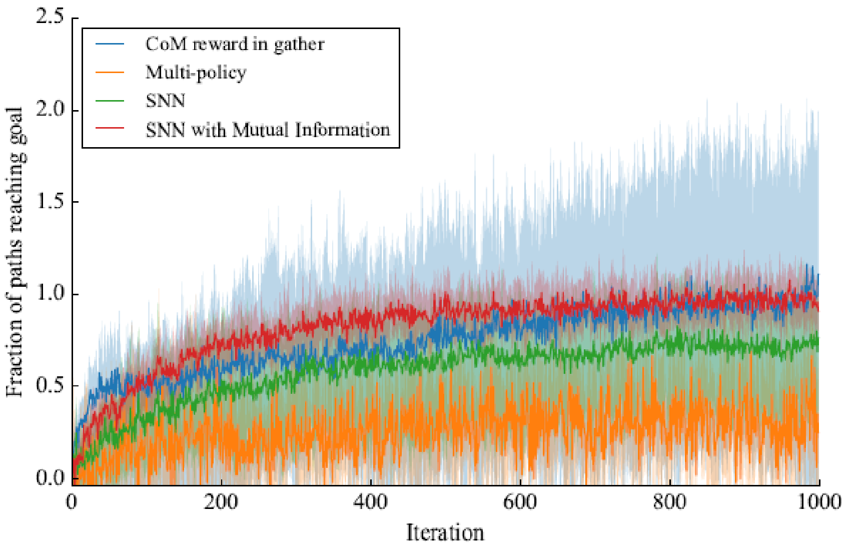


Figure 7: Results for Gather environment in the benchmark settings

2. Experiments and Results

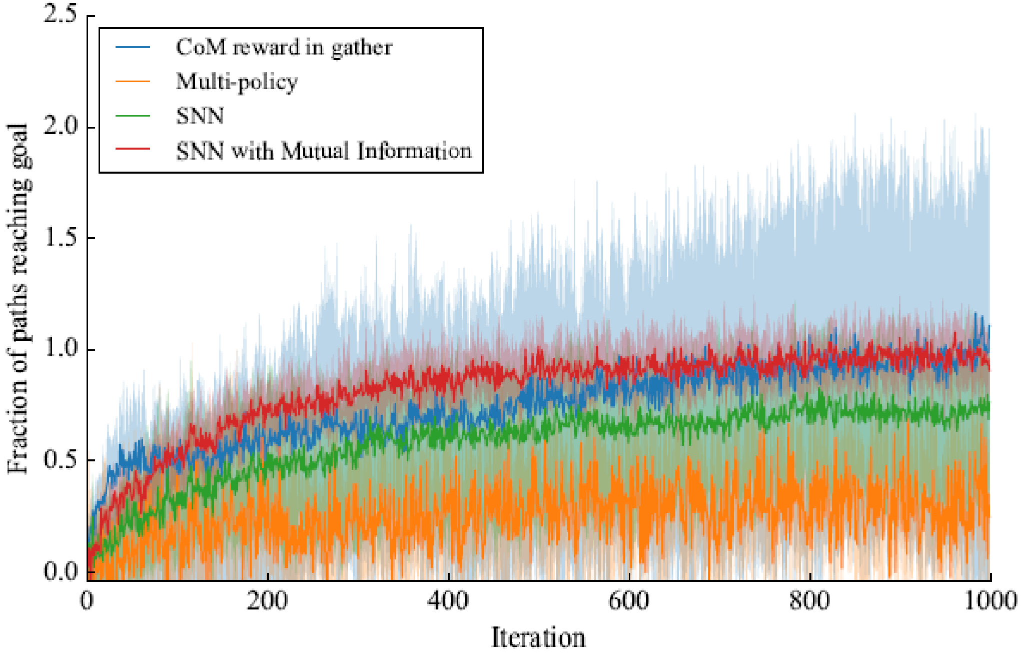


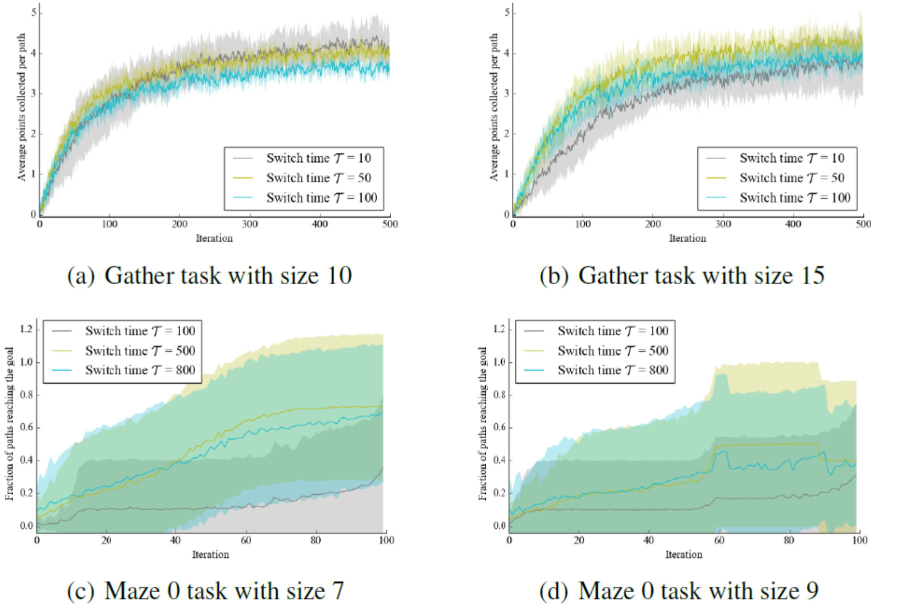
Figure 7: Results for Gather environment in the benchmark settings

2. Experiments and Results

- http://bit.ly/snn4hrl-videos

2. Experiments and Results

- Analysis of the switch time T
  - a. b.
    more frequently switch, more better in gather task.
  - c. d.
    no meaningful difference in maze task.



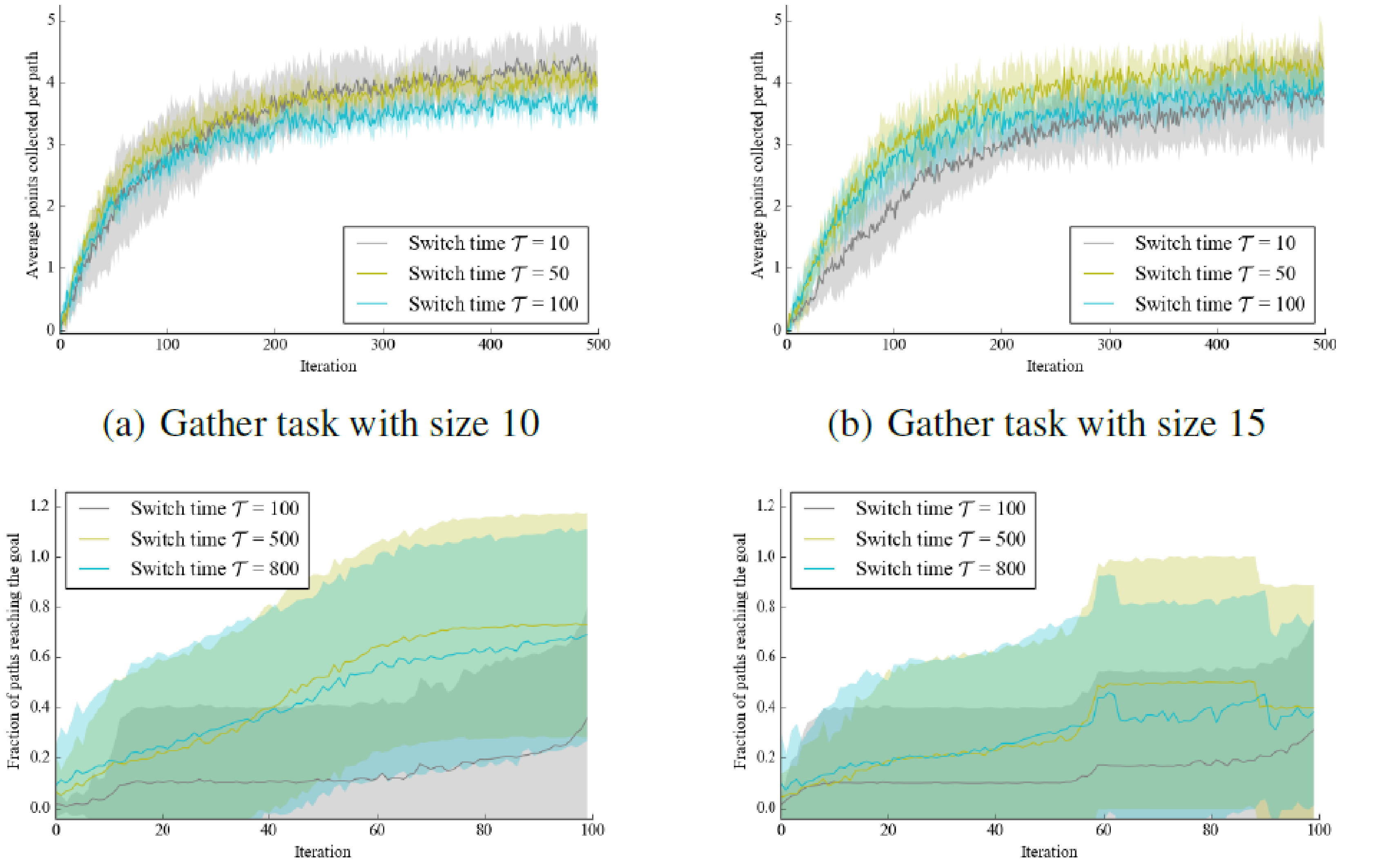
(a) Gather task with size 10 (b) Gather task with size 15

(c) Maze 0 task with size 7 (d) Maze 0 task with size 9

Figure 11: Mild effect of switch time $\mathcal{T}$ on different sizes of Gather and Maze 0

2.

• A



(a) Gather task with size 10

(b) Gather task with size 15

(c) Maze 0 task with size 7

(d) Maze 0 task with size 9

Figure 11: Mild effect of switch time $\mathcal{T}$ on different sizes of Gather and Maze 0

2. Experiments and Results

- Ant

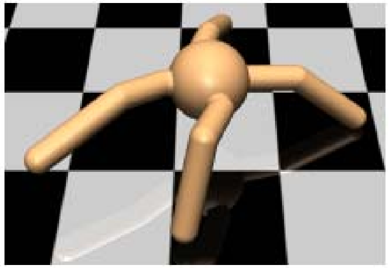


Figure 13: Ant

2. Experiments and Results

- Ant
  - Skill learning in pre-train

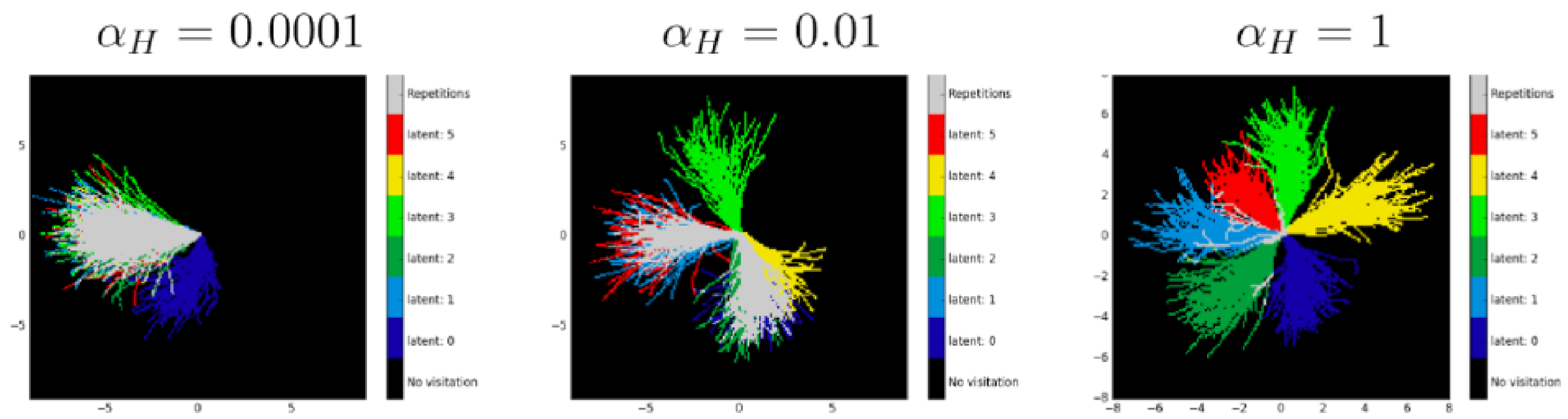


Figure 14: Pretrain visitation for Ant with different MI bonus

2. Experiments and Results

- Ant
  - Failure modes for unstable robots.

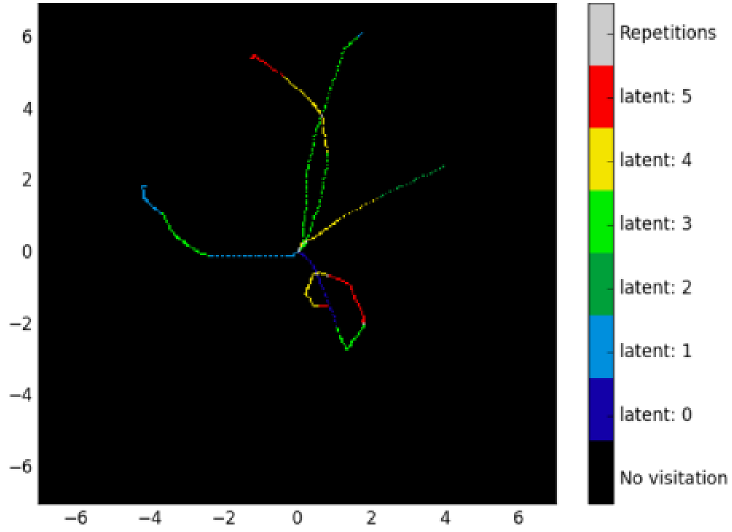


Figure 15: Failure mode of Ant: here 5 rollouts terminate in less than 6 skill switches.

3. Discussion and Future works

- The bilinear integration and the mutual information bonus are key to consistently yield a wide, interpretable span of skills.

- Limitations
  - the switching between skills for unstable agents (ant).
  - Having fixed sub-policies and a fixed switch time T.
  - Only use feedforward architectures.

Thank you!