Meta Learning Shared Hierarchies

고려대학교
박영준

Paper

Under review as a conference paper at ICLR 2018

META LEARNING SHARED HIERARCHIES

Kevin Frans
Henry M. Gunn High School
Work done as an intern at OpenAI
kevinfrans2@gmail.com

Jonathan Ho, Xi Chen, Pieter Abbeel
UC Berkeley, Department of Electrical Engineering and Computer Science

John Schulman
OpenAI

ABSTRACT

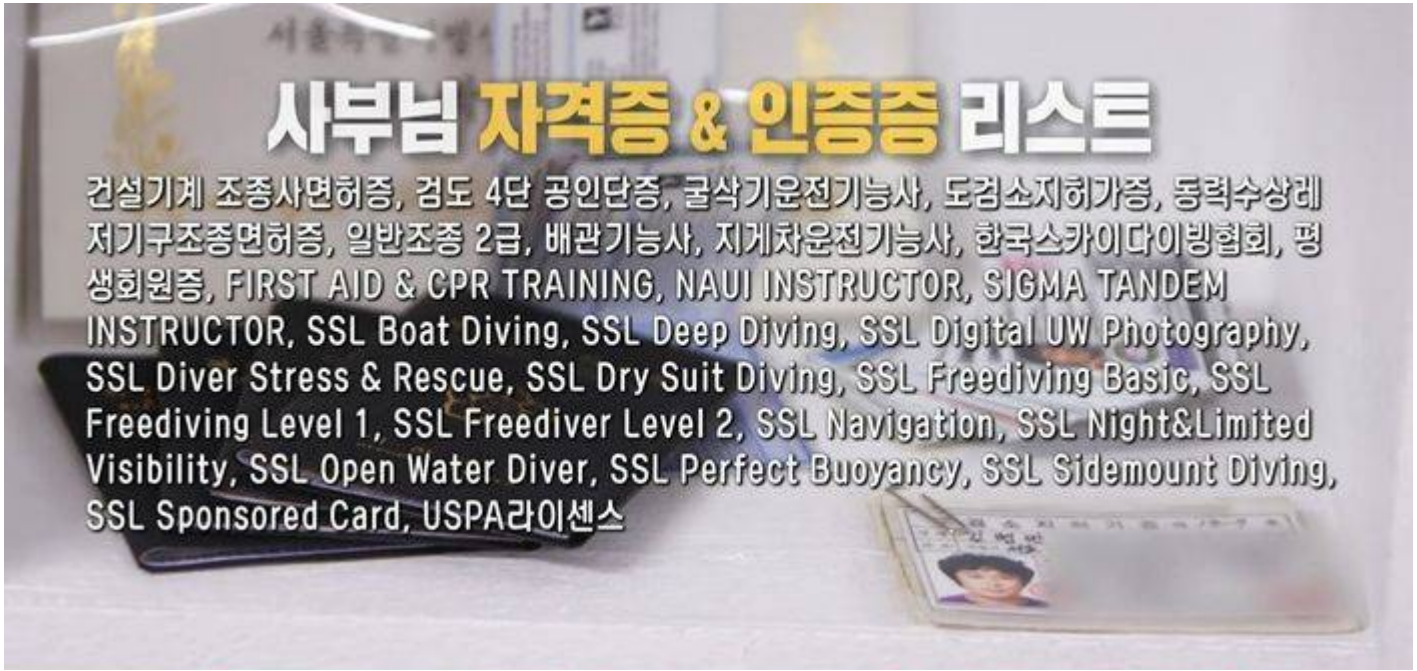
We develop a metalearning approach for learning hierarchically structured policies, improving sample efficiency on unseen tasks through the use of shared primitives—policies that are executed for large numbers of timesteps. Specifically, a set of primitives are shared within a distribution of tasks, and are switched between by task-specific policies. We provide a concrete metric for measuring the strength of such hierarchies, leading to an optimization problem for quickly reaching high reward on unseen tasks. We then present an algorithm to solve this problem end-to-end through the use of any off-the-shelf reinforcement learning method, by repeatedly sampling new tasks and resetting task-specific policies. We successfully discover[1] meaningful motor primitives for the directional movement of four-legged robots, solely by interacting with distributions of mazes. We also demonstrate the transferability of primitives to solve long-timescale sparse-reward obstacle courses, and we enable 3D humanoid robots to robustly walk and crawl with the same policy.

Contents

- Meta Learning
- Meta Learning in Reinforcement Learning
- Problem Statement
- Proposed method
- Experiments
- Conclusions

Meta Learning

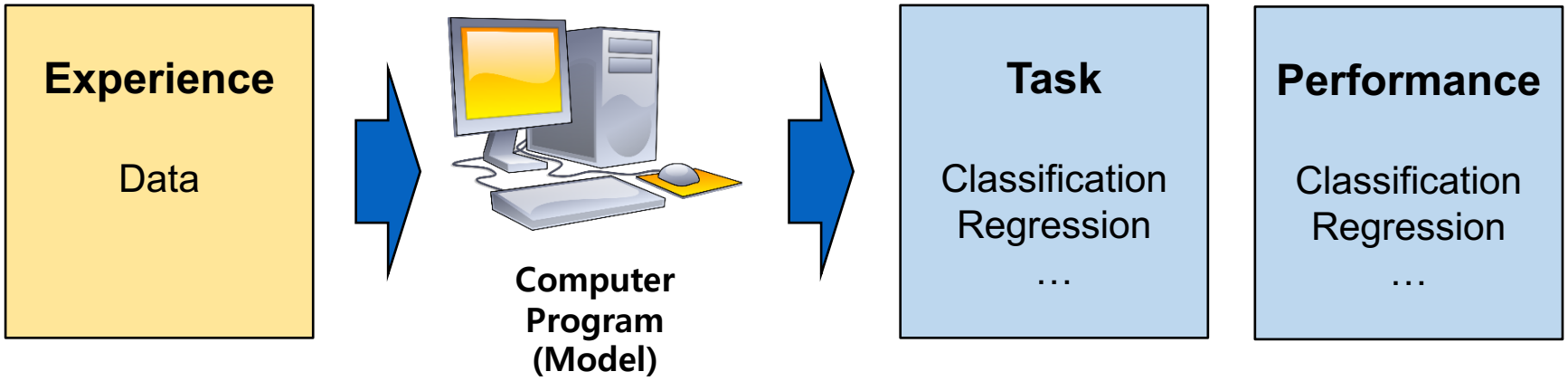
- In psychology, learning about one's own learning and learning processes



김병만 자격증 보유 리스트

Meta Learning in Machine Learning

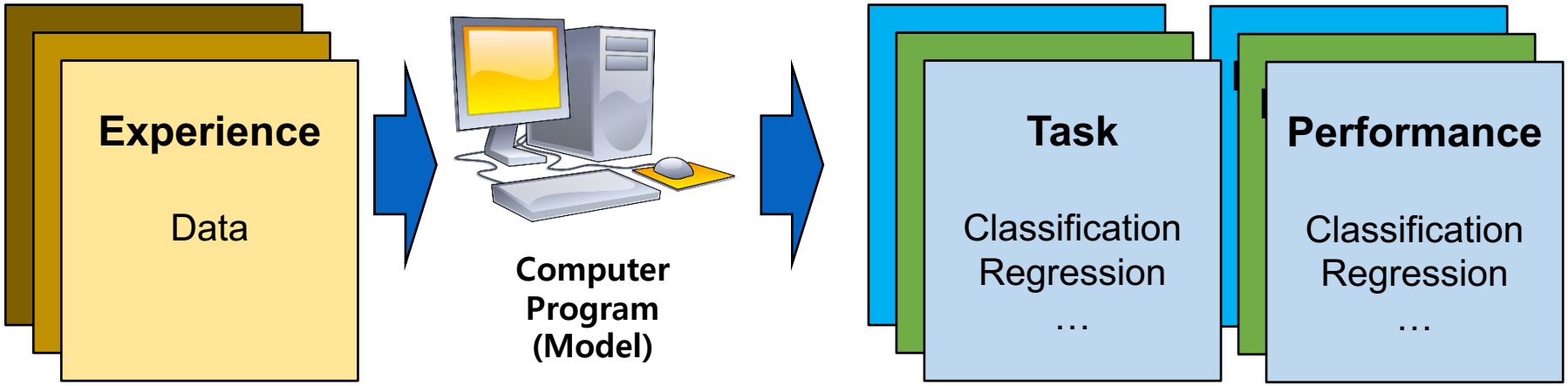
- It is also known as “learning to learn”, intends to design models that can learn new skills or adapt to new environments rapidly with a few training examples.



Definition of Machine Learning

Meta Learning in Machine Learning

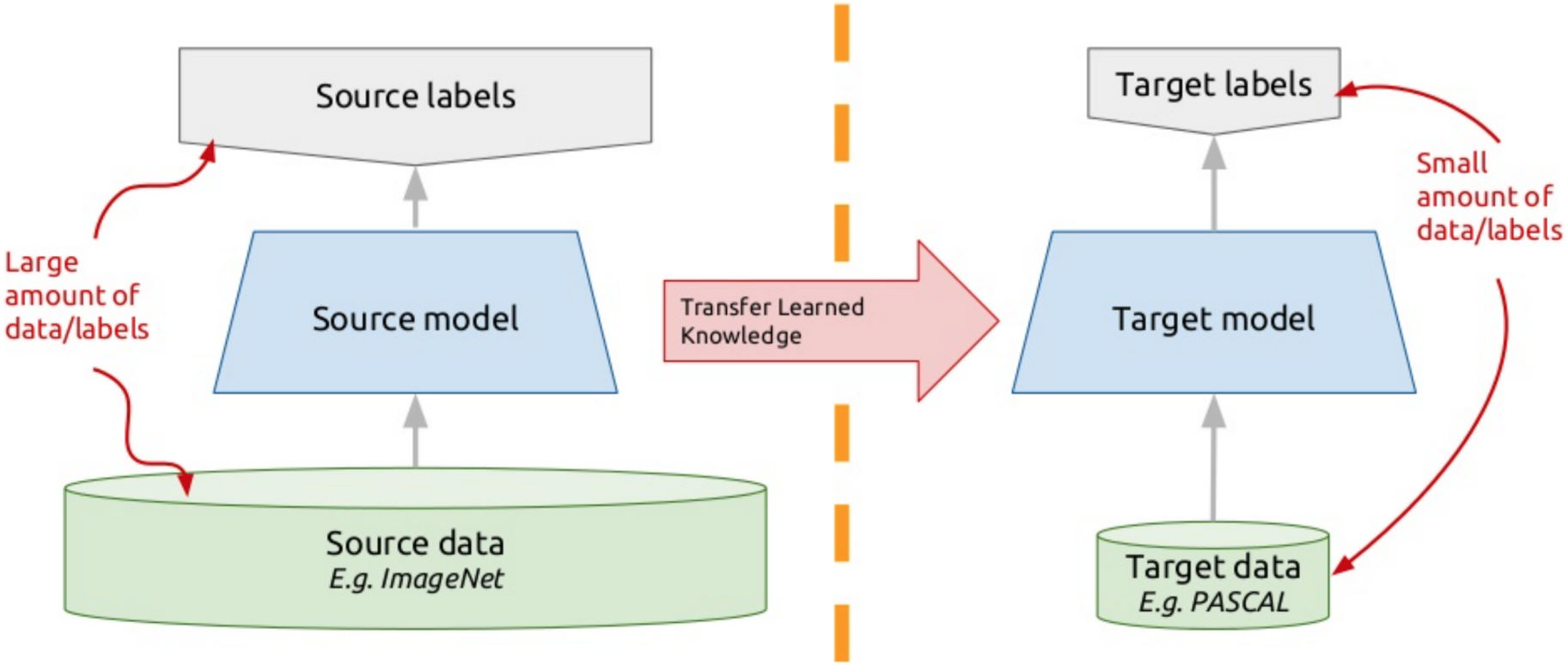
- It is also known as “learning to learn”, intends to design models that can learn new skills or adapt to new environments rapidly with a few training examples.



Definition of Machine Learning

Relation to Transfer Learning

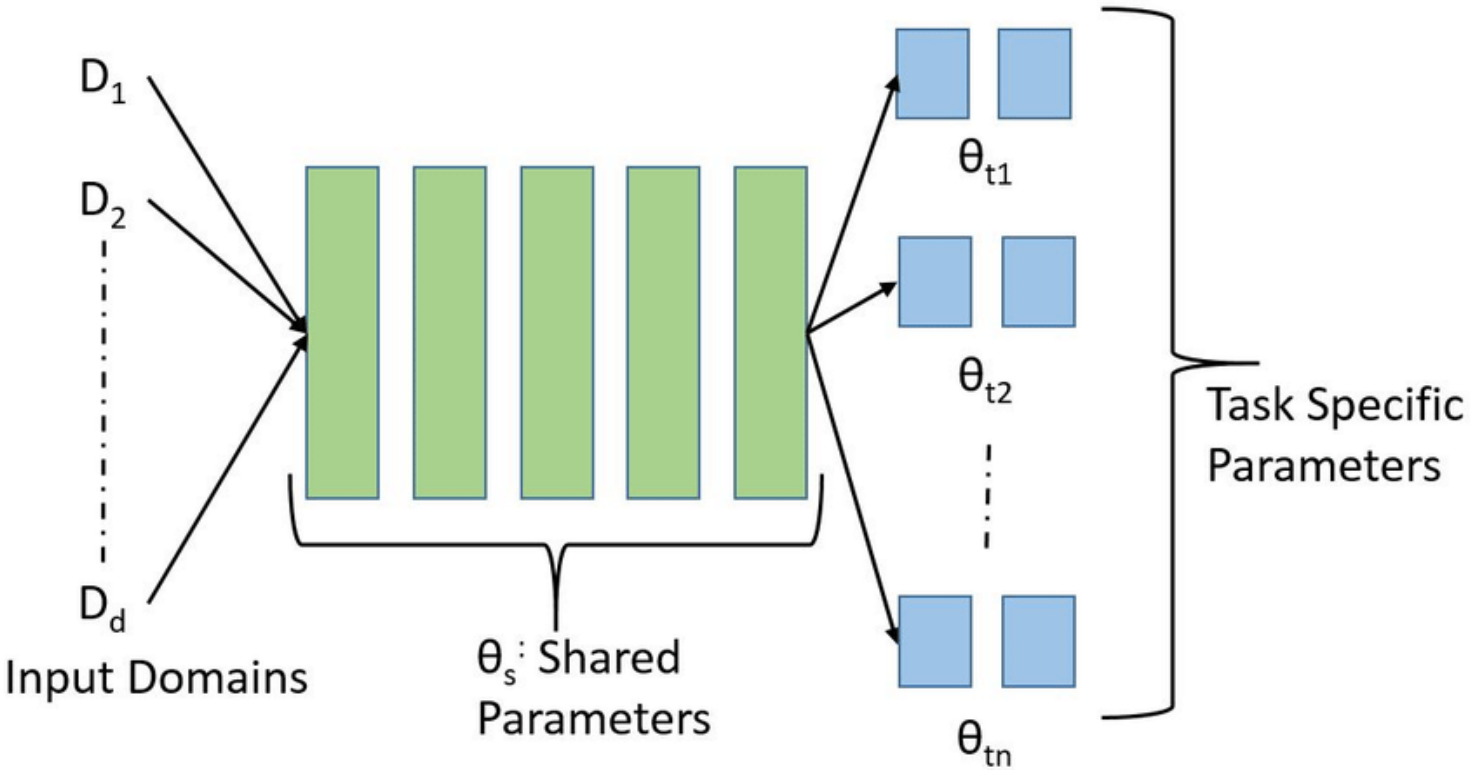
- Storing knowledge gained while solving one problem and applying it to a different but related problem



Transfer Learning Idea

Relation to Multi-Task Learning

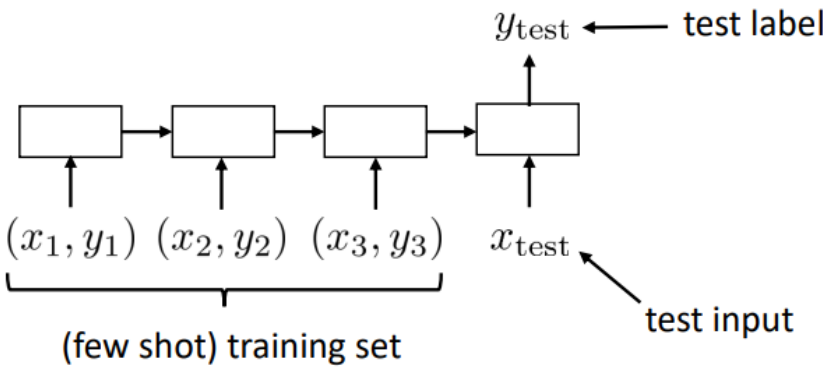
- Multiple learning tasks are solved at the same time, while exploiting commonalities and differences across tasks.



Multi-Task Learning Idea

Meta Learning in Supervised Learning

- Different performance evaluation scheme from transfer learning or multi-task learning



supervised learning: $f(x) \rightarrow y$

input (e.g., image) output (e.g., label)

supervised meta-learning: $f(\mathcal{D}_{\text{train}}, x) \rightarrow y$

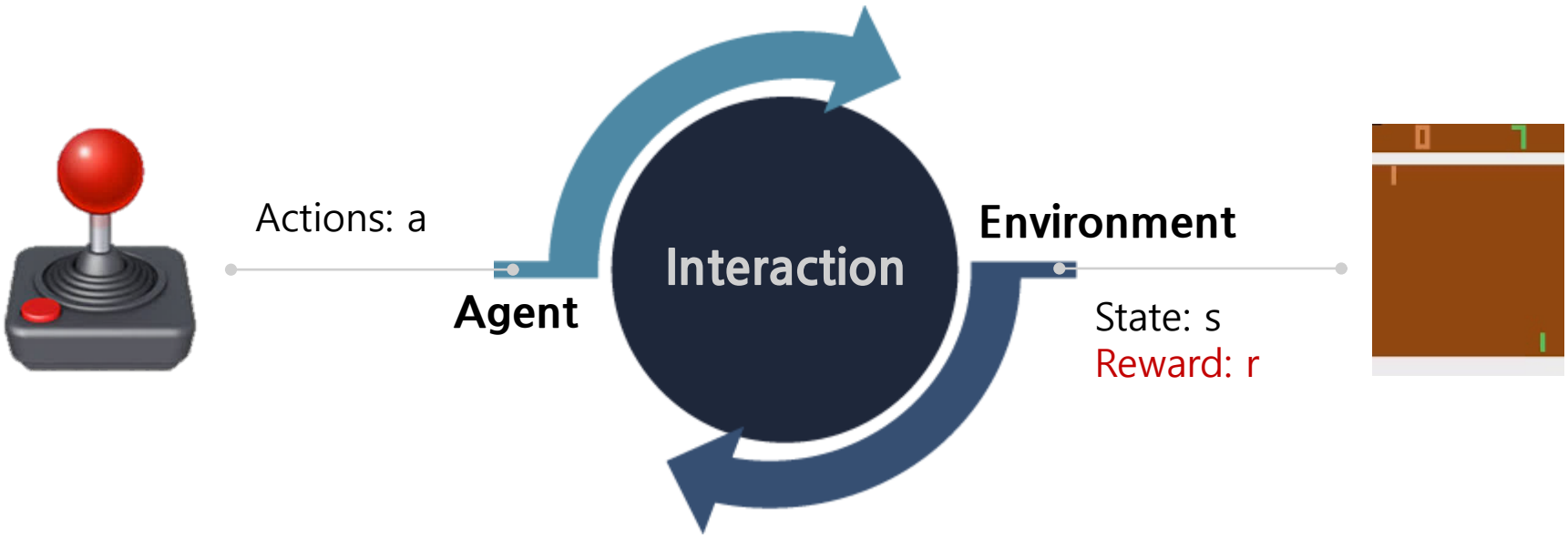
training set

- How to read in training set?
  - Many options, RNNs can work

Contents

Reinforcement Learning (RL)

- Find policy $\pi(a|s)$ while maximize cumulative rewards
- It is called *approximate dynamic programming*

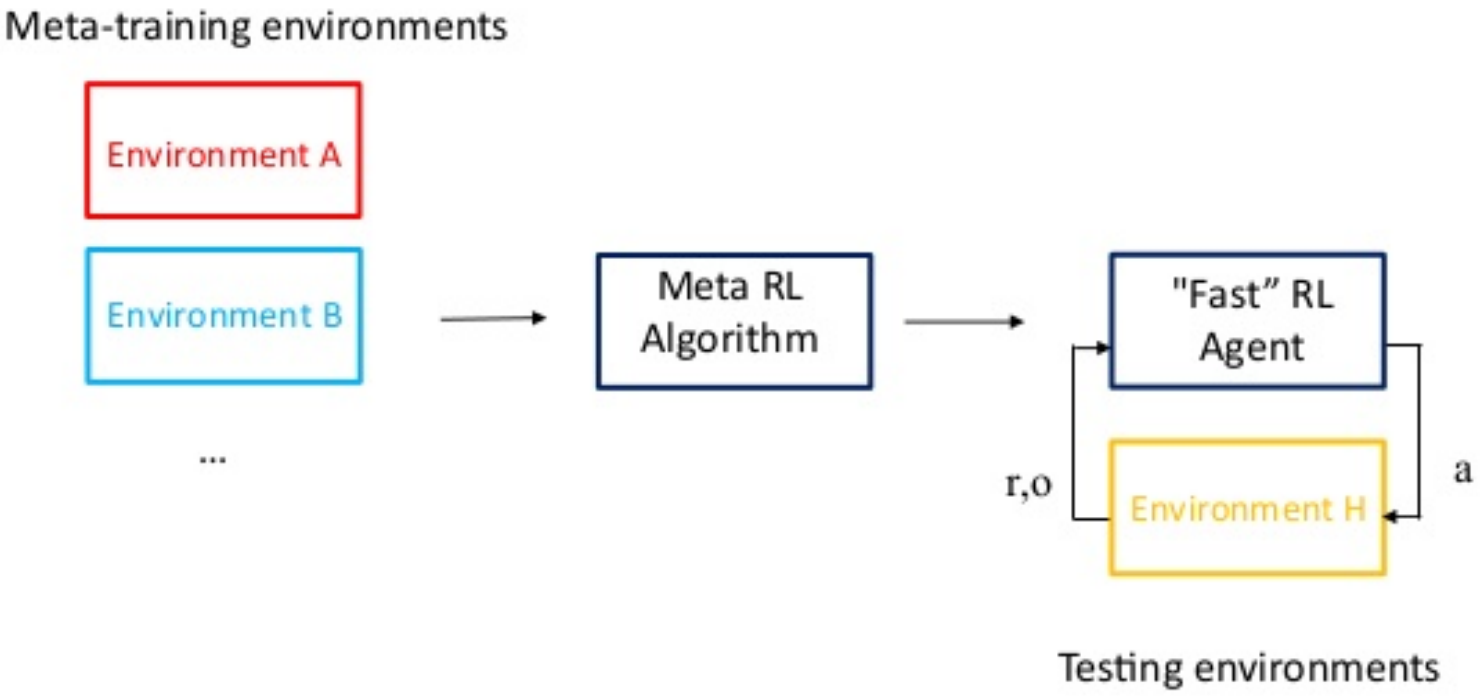


Meta Learning in Reinforcement Learning

- Fast learning (=sample efficiency) on unseen tasks

Meta Learning in Reinforcement Learning

- Fast learning (=sample efficiency) on unseen tasks

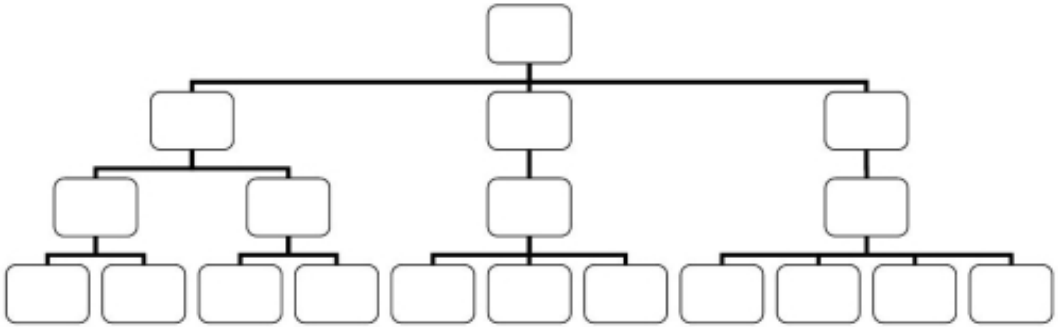


Contents

Research Objective

- Learning **hierarchically structured policies**, improving sample efficiency on unseen tasks
- Hierarchical reinforcement learning (HRL) aims to decompose large problems into smaller ones to address scalability issues.

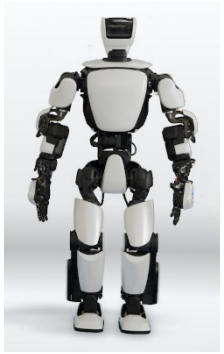
High-Level Policy: macro action

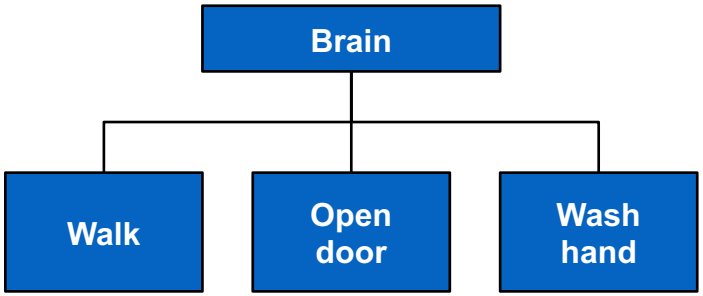


Low-Level Policy: micro action

HRL: Example1

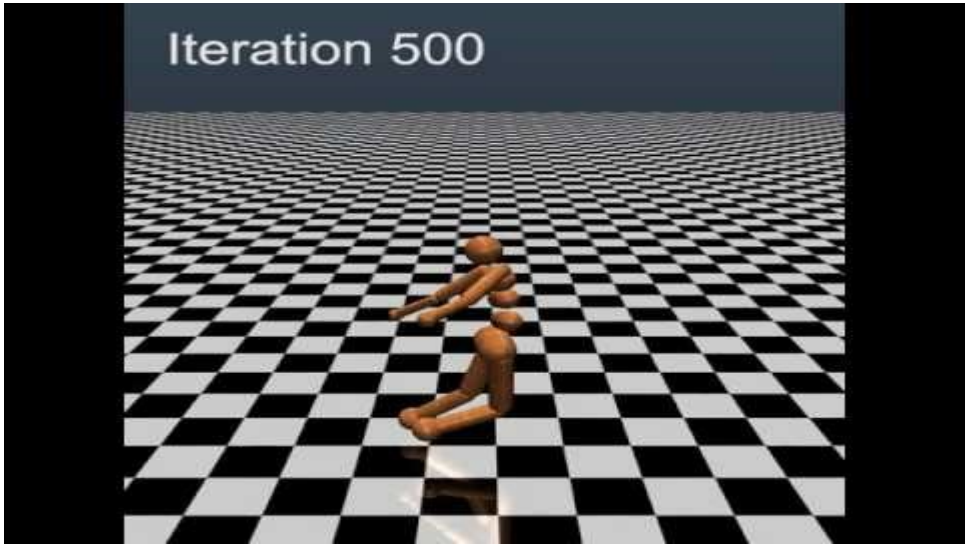
- Hierarchical reinforcement learning (HRL) aims to decompose large problems into smaller tasks to address scalability issues.
- Task: go to bath room & wash hands





HRL: Example1

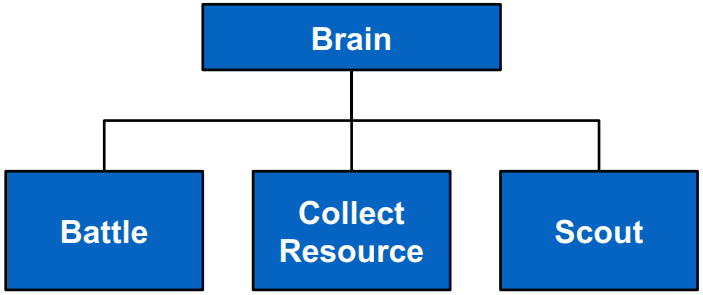
- Robot has high dimensional action space (e.g. control every joint torque)



MUJOCO Walking with PPO

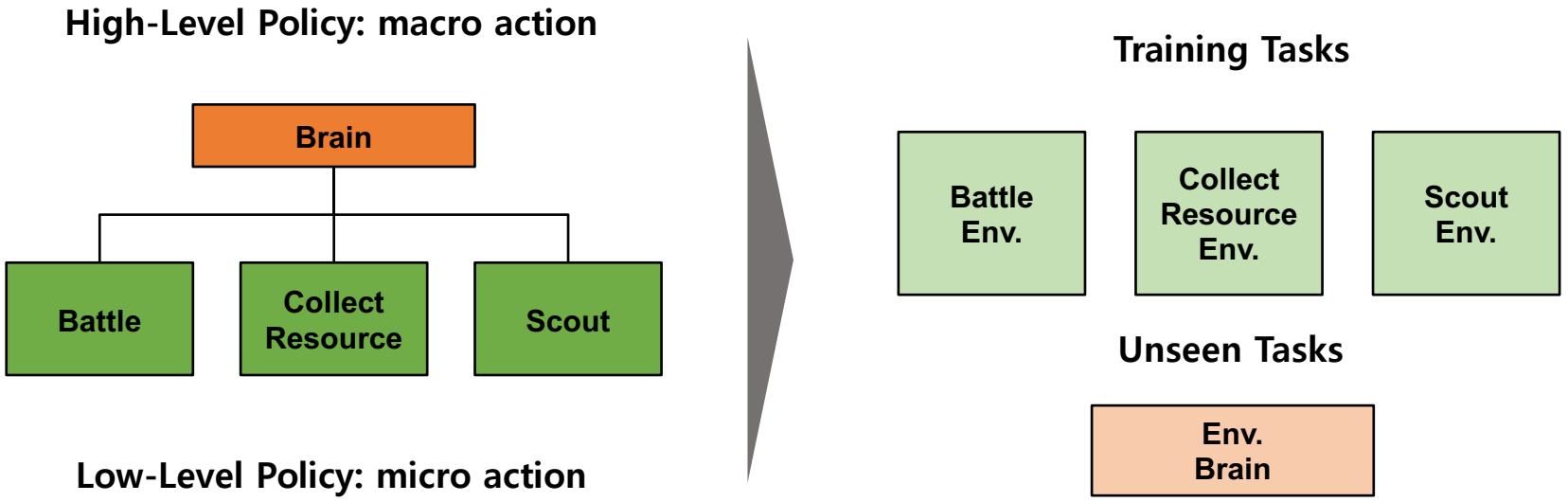
HRL: Example2

- Hierarchical reinforcement learning (HRL) aims to decompose large problems into smaller tasks to address scalability issues.
- Task: Starcraft



Research Objective

- Learning **hierarchically structured policies**, improving sample efficiency on unseen tasks
- Hierarchical reinforcement learning (HRL) aims to decompose large problems into smaller ones to address scalability issues



Simple Example: 2D Moving Bandits

- We consider the setting where agents solve distributions of related tasks, with the goal of learning new task quickly. One challenge is that while we want to share information between the different tasks, these tasks have different optimal policies, so it is suboptimal to learn a single shared policy for all tasks.

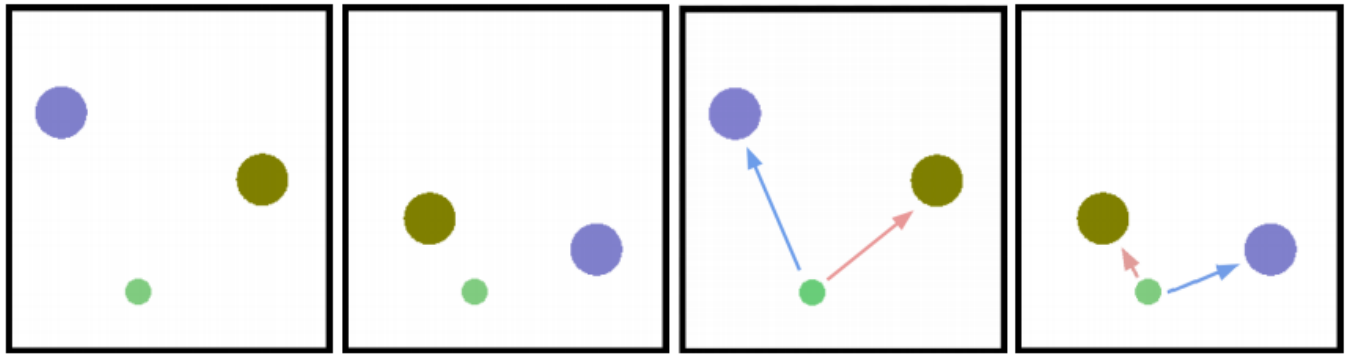


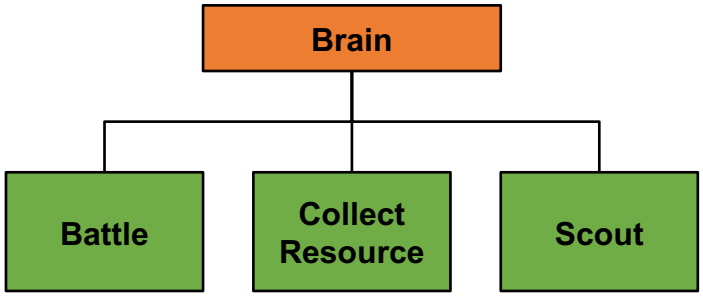
Figure 3: Sampled tasks from 2D moving bandits. Small green dot represents the agent, while blue and yellow dots represent potential goal points. Right: Blue/red arrows correspond to movements when taking sub-policies 1 and 2 respectively.

Contents

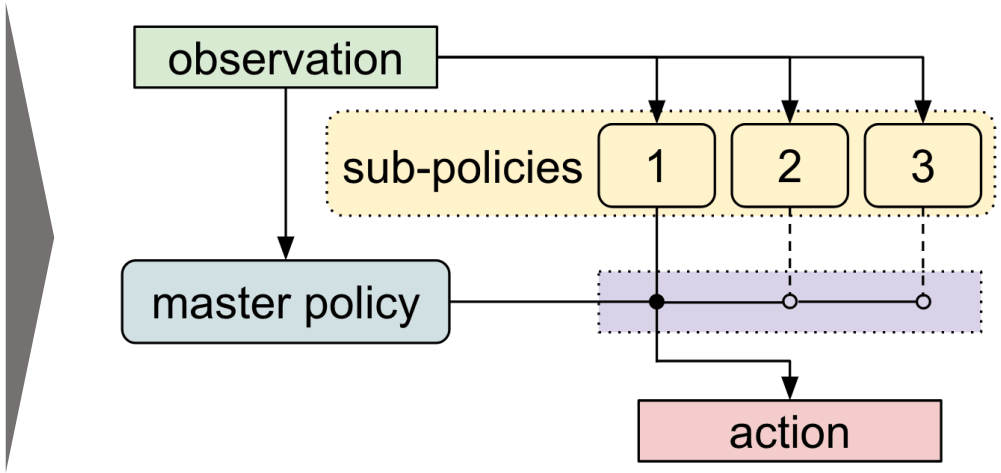
Meta Learning Shared Hierarchies (MLSH)

- Meta learning: small-size tasks (environments)
- Hierarchical RL: complex task
- Learn sub-policies automatically without hand engineering

High-Level Policy: macro action



Low-Level Policy: micro action



Training of MLSH

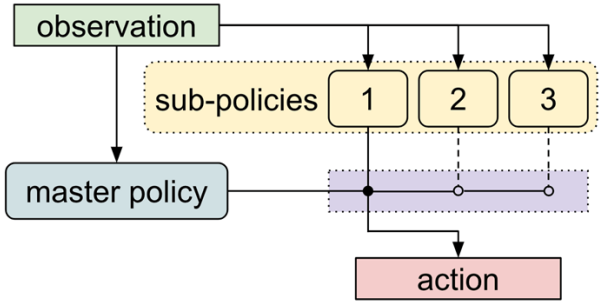
- Skip

Algorithm 1 Meta Learning Shared Hierarchies

Initialize ϕ
repeat
  Initialize θ
  Sample task $M \sim P_M$
  for $w = 0, 1, ...W$ (warmup period) **do**
    Collect D timesteps of experience using $\pi_{\phi,\theta}$
    Update θ to maximize expected return from $1/N$ timescale viewpoint
  end for
  for $u = 0, 1, ...U$ (joint update period) **do**
    Collect D timesteps of experience using $\pi_{\phi,\theta}$
    Update θ to maximize expected return from $1/N$ timescale viewpoint
    Update ϕ to maximize expected return from full timescale viewpoint
  end for
until convergence

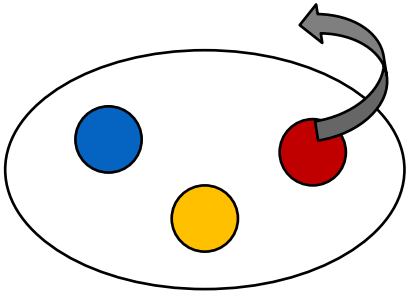
Training of MLSH

- θ: parameters of master policy
- ϕ: parameters of sub-policies
- Repeat steps until convergence



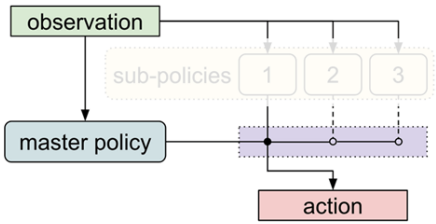
Step1: Task Sampling

- Sample task $M \sim P_M$
- Initialize θ



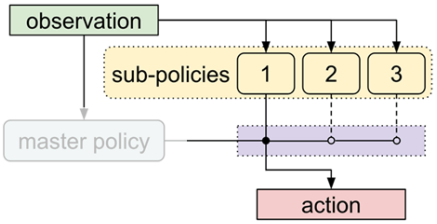
Step2: Warmup

- Update master policy
- 10~30 iterations



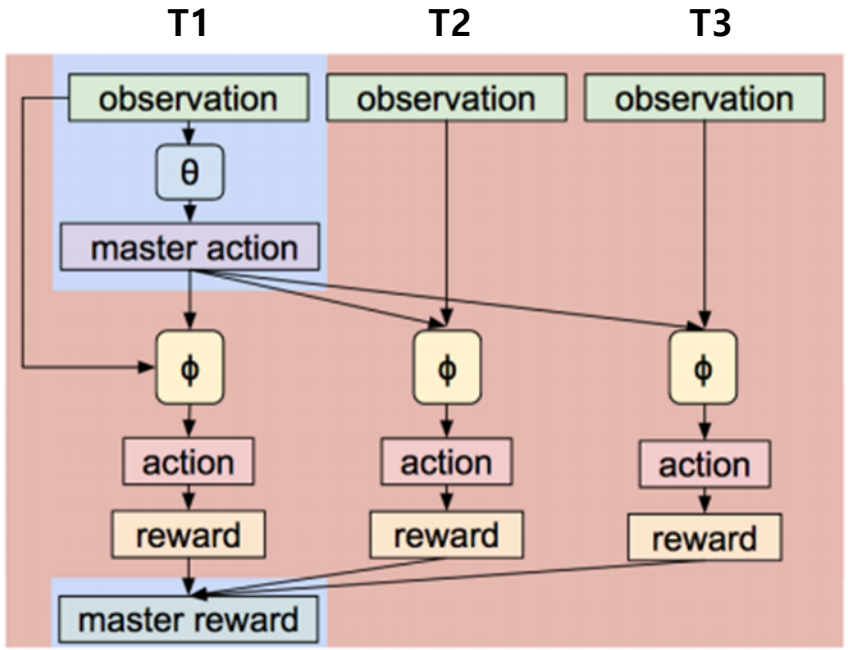
Step3: Joint Update

- Update master & sub policy
- 30~40 iterations



Training Master Policy

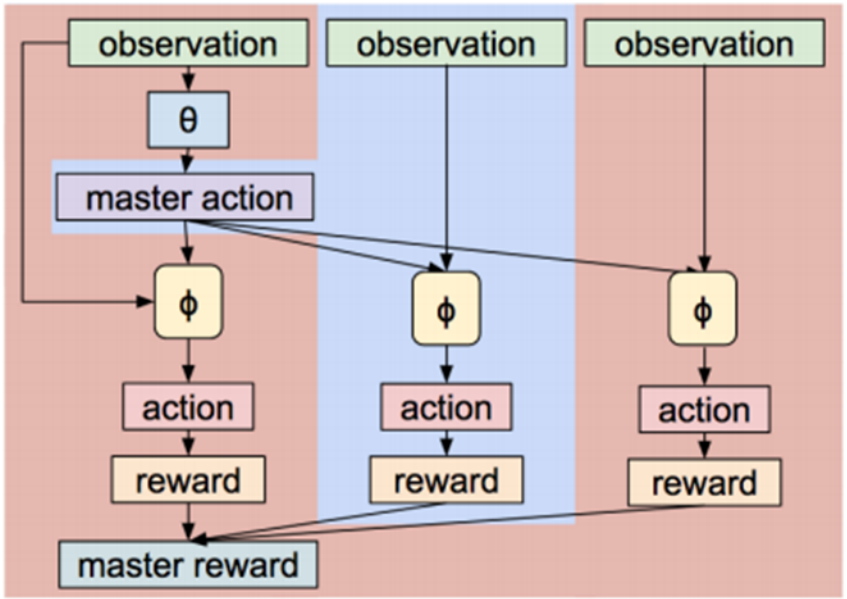
- Master policy actions last for N time steps
- **State**: N observations
- **Action**: sub-policy selection
- **Reward**: mean of rewards for N time steps



Time Scale N=3 Example

Training Sub-Policies

- **State**: observation & master action
- **Action**: low-level actions
- **Reward**: reward for each time step



How to Calculate Gradients

- $\nabla\theta$, $\nabla\phi$
- Use existing reinforcement learning algorithms
- DQN, A3C, TRPO, **PPO**, …

$$\theta \leftarrow \theta + \alpha_\theta \nabla\theta \ , \qquad \alpha_\theta = 0.01$$

$$\phi \leftarrow \phi + \alpha_\phi \nabla\phi \ , \qquad \alpha_\phi = 0.0003$$

Contents

- Meta Learning
- Meta Learning in Reinforcement Learning
- Problem Statement
- Proposed Method
- **Experiments**
- Conclusions

Simple Environment

- "Can meaningful sub-policies be learned over a distribution of tasks, and do they outperform a shared policy?"
- **Movement Bandits Environment**: agent starts at a specific spot in the gridworld, and is randomly assigned a goal position. A reward of 1 is awarded for being in the goal state.

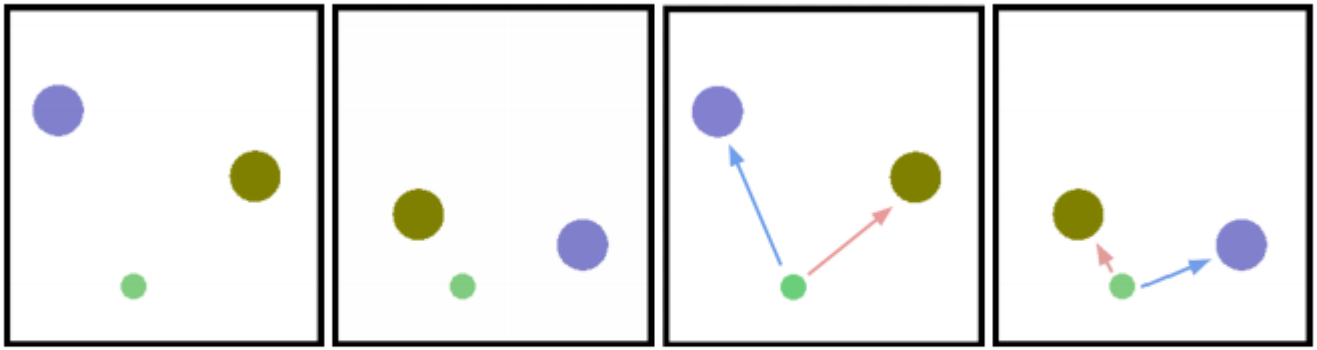
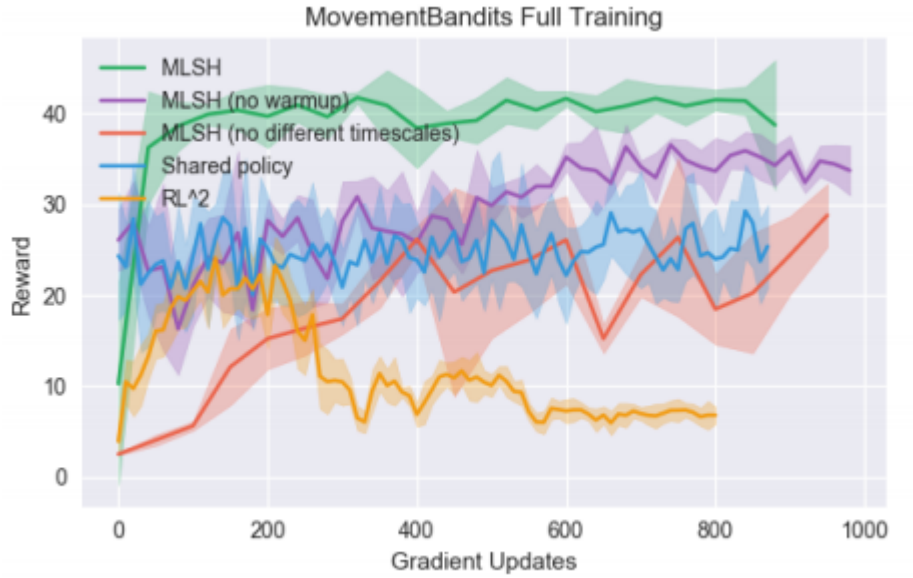
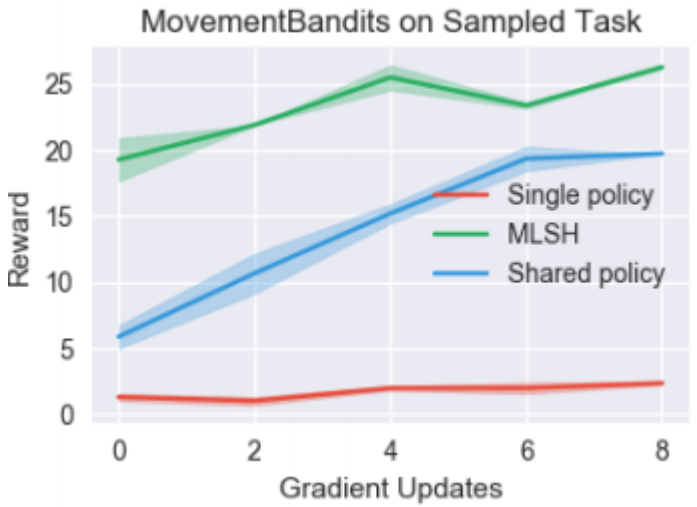


Figure 3: Sampled tasks from 2D moving bandits. Small green dot represents the agent, while blue and yellow dots represent potential goal points. Right: Blue/red arrows correspond to movements when taking sub-policies 1 and 2 respectively.

Comparison to Other Approach

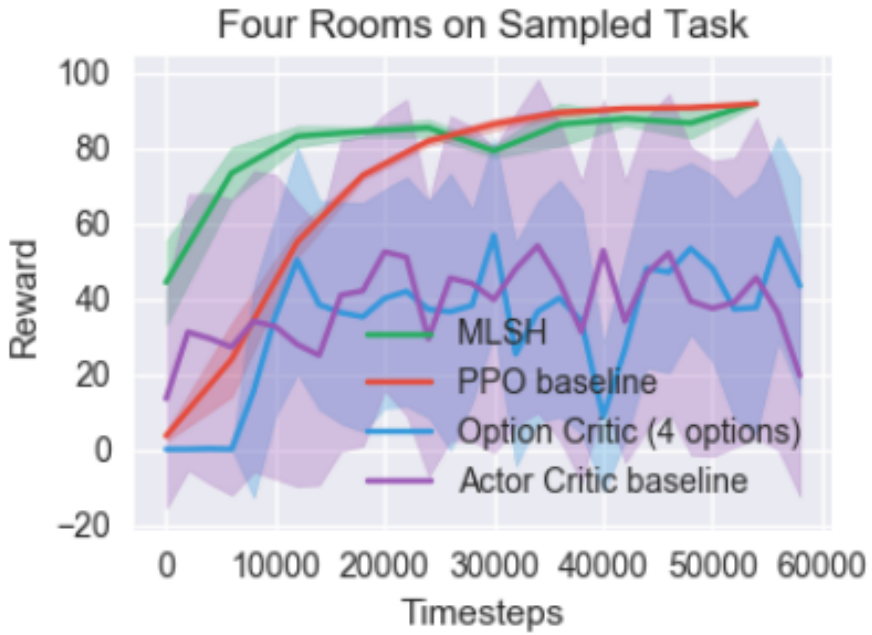
- MLSH: two sub-policies + master policy
- Shared policy over various tasks
- Single policy from scratch





Comparison to Existing HRL

- Environment: four room
- Existing method: Option Critic



Complex Environments

- Mujoco based environments
- Actions: angle of multiple joint

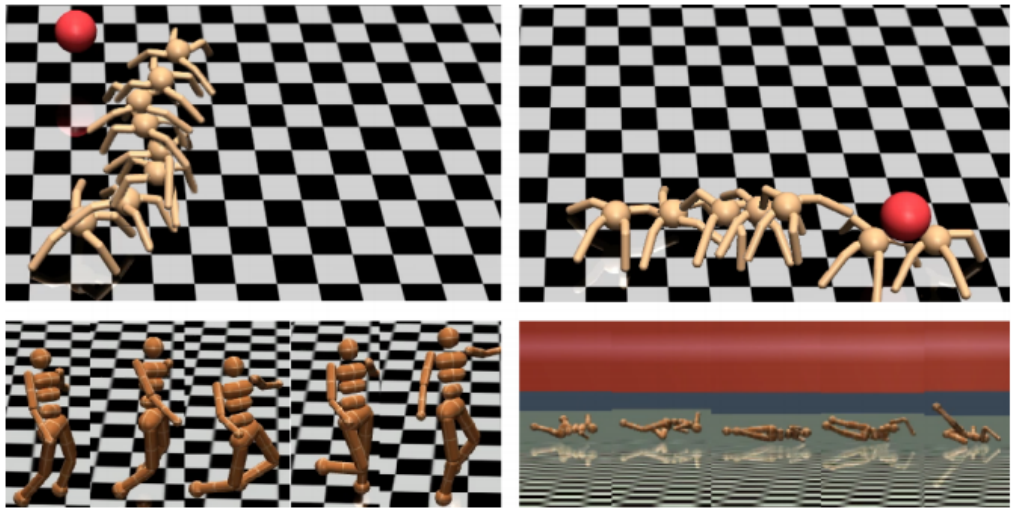
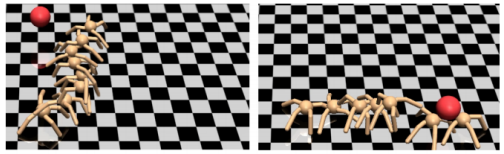
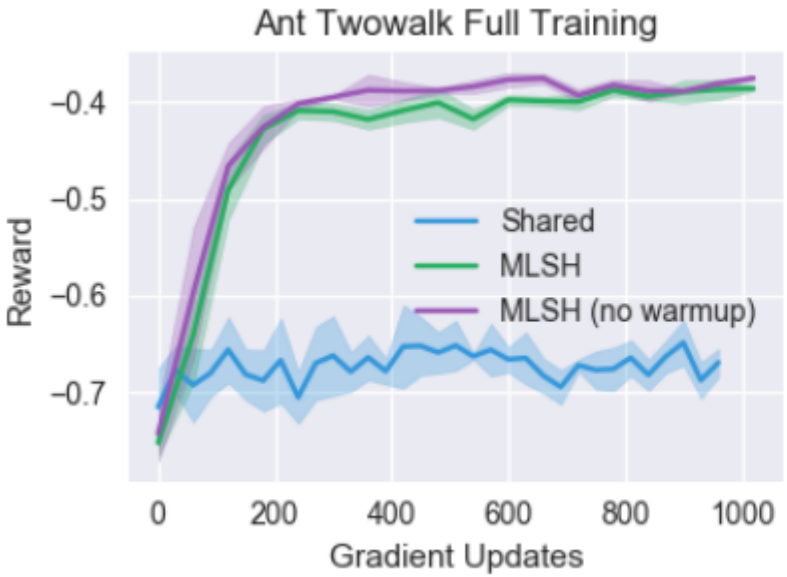


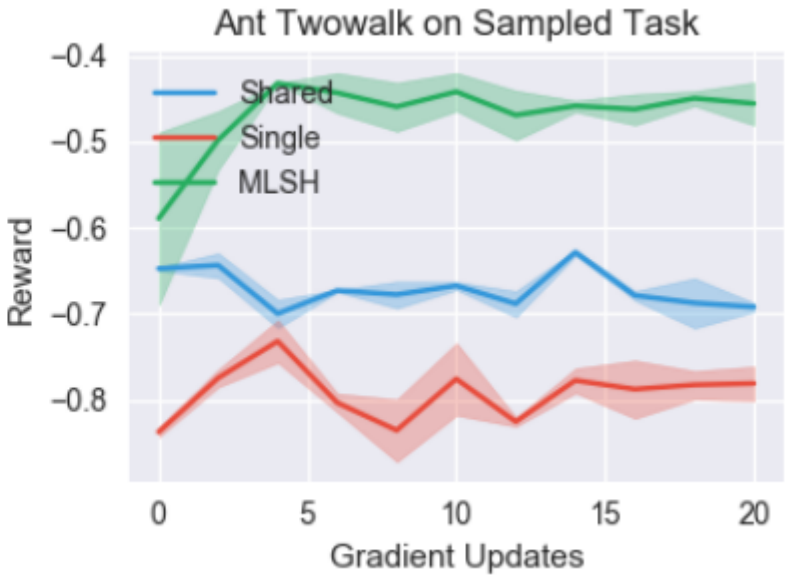
Figure 5: Top: Ant Twowalk. Ant must maneuver towards red goal point, either towards the top or towards the right. Bottom Left: Walking. Humanoid must move horizontally while maintaining an upright stance. Bottom Right: Crawling. Humanoid must move horizontally while a height-limiting obstacle is present.

Comparison to Other Approach

- **Task: Twowalk (ant bandits)**
- MLSH: two sub-policies + master policy
- Shared policy over various tasks
- Single policy from scratch

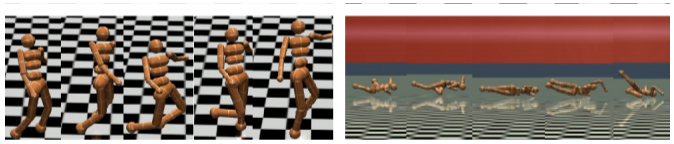


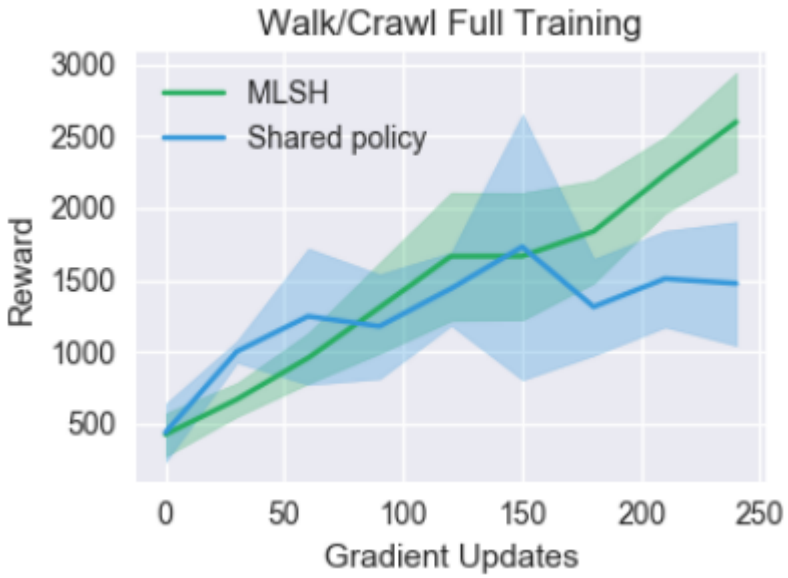




Comparison to Other Approach

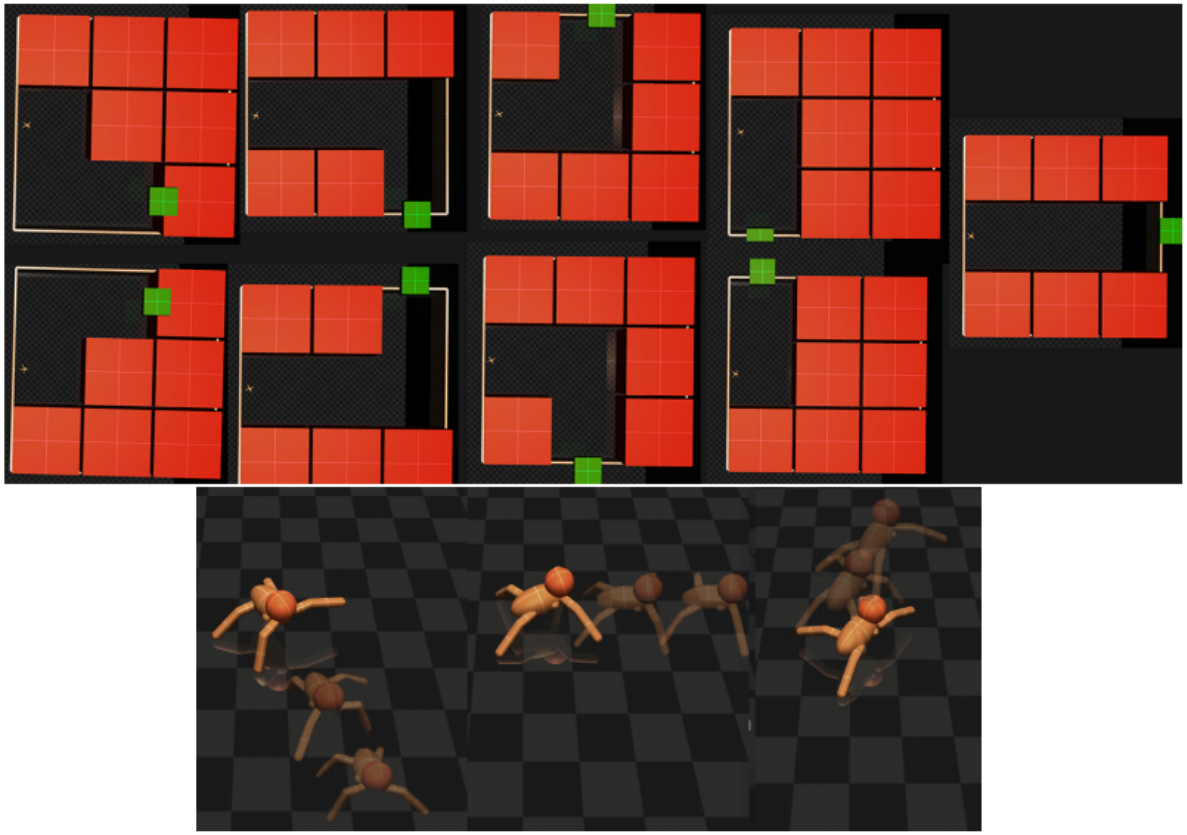
- **Task: Walk/Crawl**
- MLSH: two sub-policies + master policy
- Shared policy over various tasks





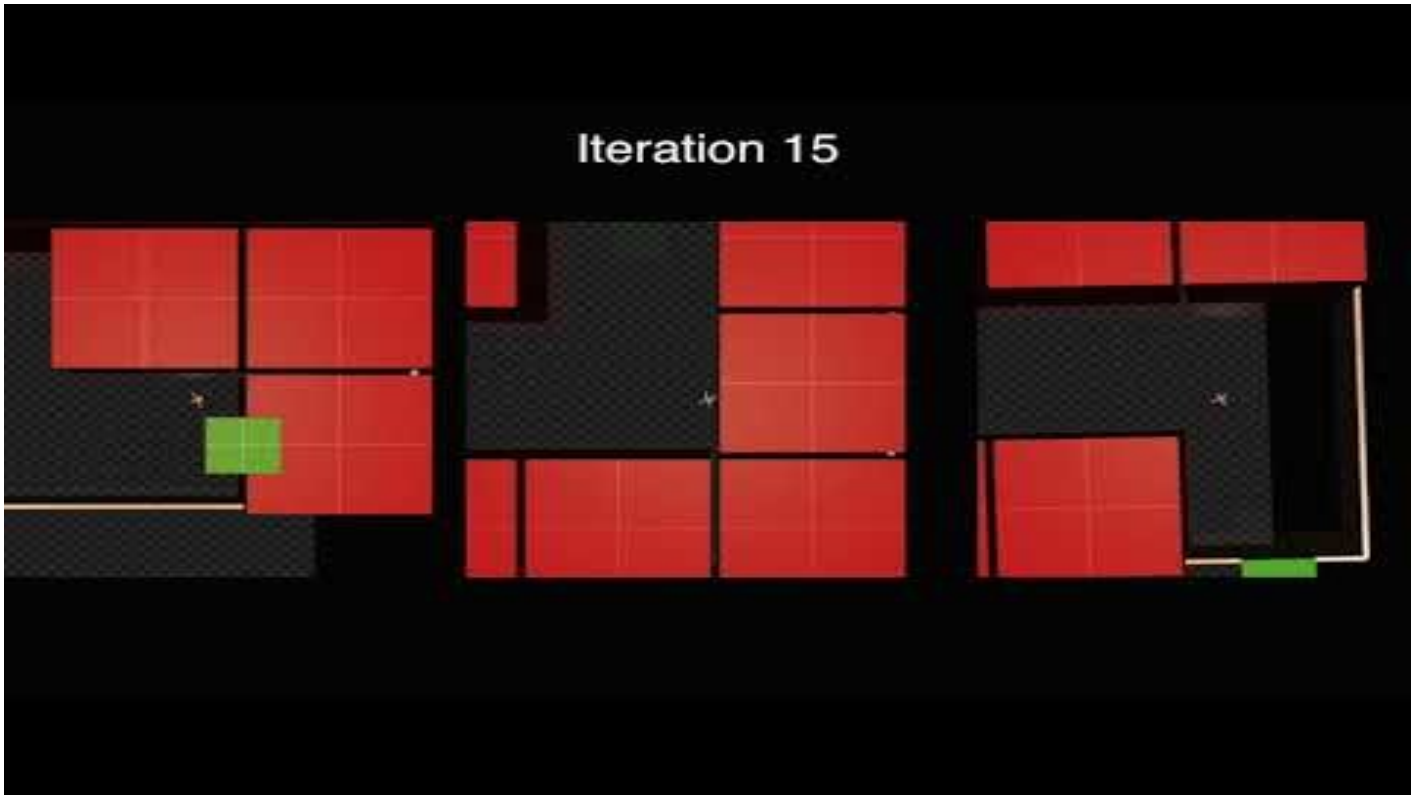
Maze Environment Example

- **Task: Escape maze**
- Shared policy over various tasks



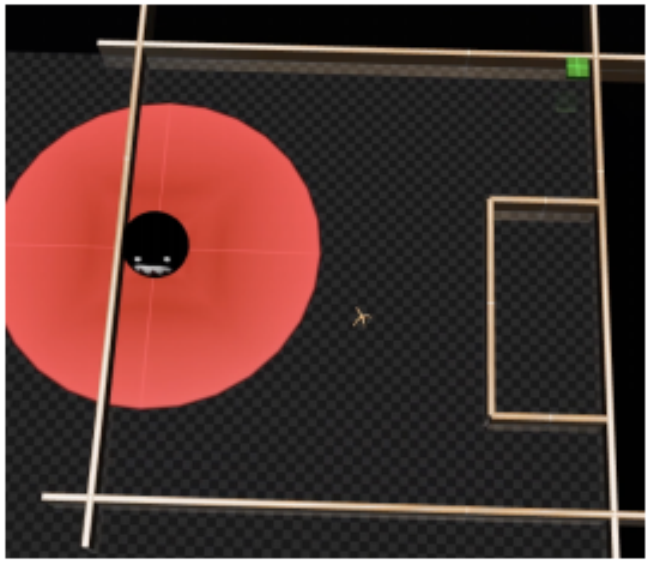
Maze Environment Example

- **Task: Escape maze**
- Shared policy over various tasks



Sparse Reward Envrionment

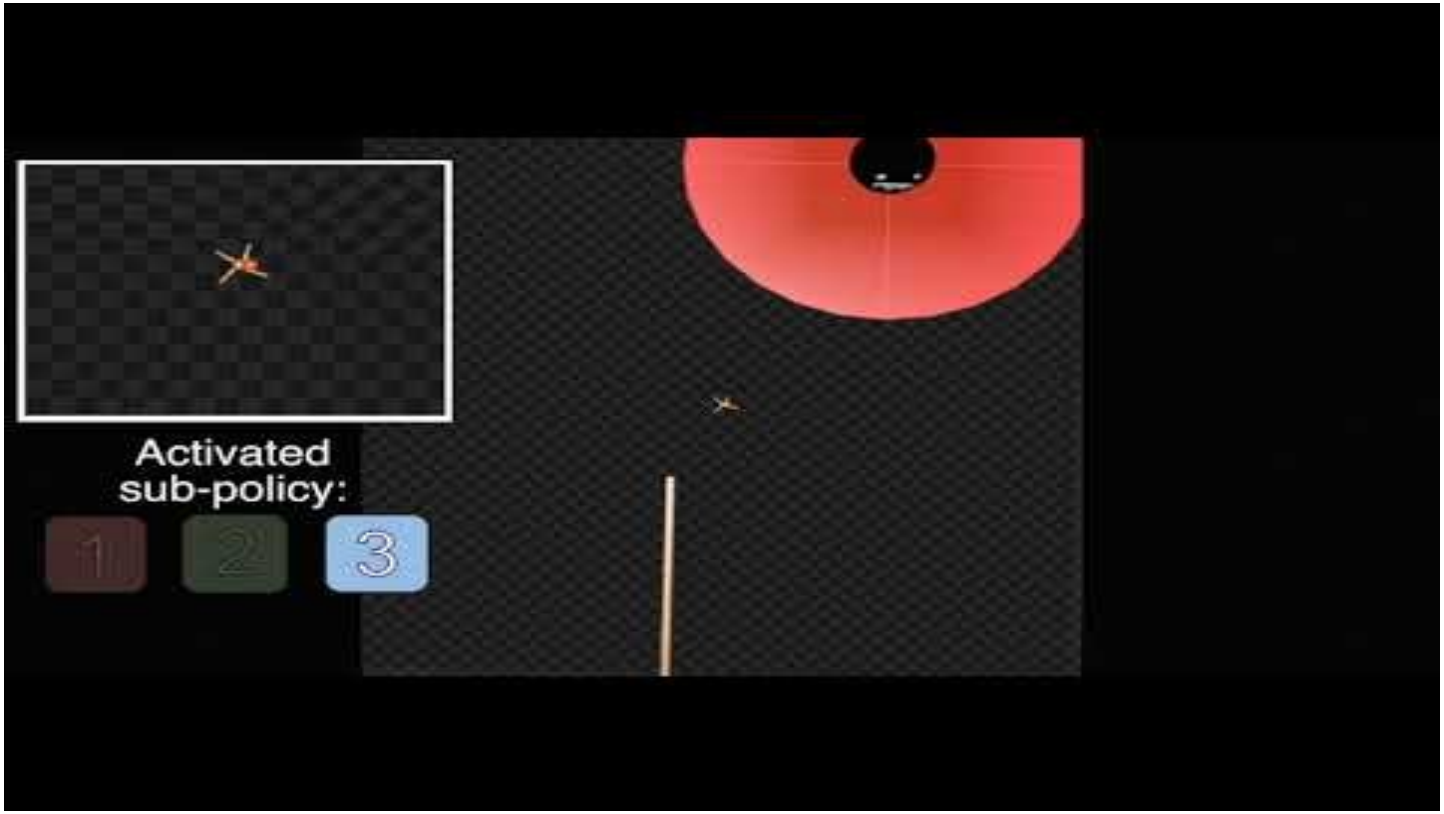
- **Task: Ant Obstacle**
- Agent must navigate to the green square in the top right corner. Entering the red circle causes an enemy to attack the agent, knocking it back.
- Reward: 1 if success, 0 otherwise



Sparse Reward Envrionment

- Transfer from TwoWalk task

Reward on Ant Obstacle task	
MLSH Transfer	193
Single Policy	0



Contents

- Meta Learning
- Meta Learning in Reinforcement Learning
- Problem Statement
- Proposed Method
- Experiments
- **Conclusions**

Conclusions

Summary

- Metalearning approach for learning hierarchically structured policies, improving sample efficiency on unseen tasks
- Transferability of primitives to solve long-timescale sparse-reward obstacle courses

Critic

- No constant evaluations
- Is really meta learning? Strong relation to hierarchical reinforcement learning
- Two stage training scheme is not nice (why warmup?)
- Many hyper parameters. Especially the number of sub-policies

Research Idea

- θ: parameters of master policy
- ϕ: parameters of sub-policies
- Add constraint to encourage diversification of sub-policies (i.e. $\max \sum_{i \neq j} \|\phi_i - \phi_j\|$)