Meta-Learning with Multi-Level Hierarchies via Context Variables

Willie McClinton, Andrew Levy, George Konidaris
Department of Computer Science, Brown University Providence, RI

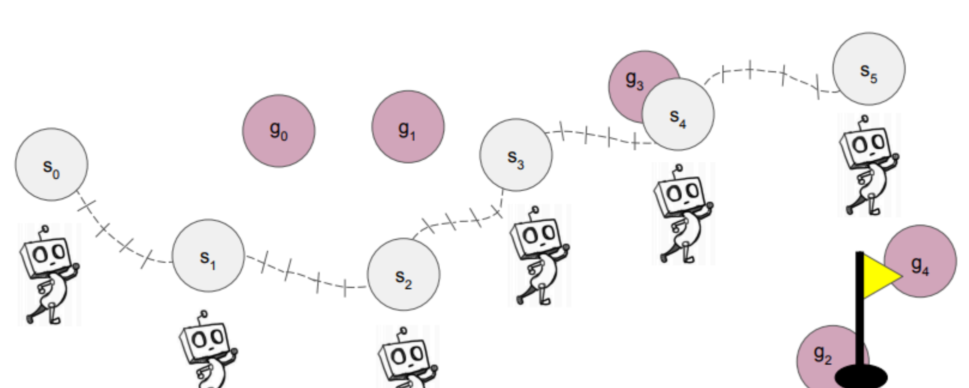
Motivation:

- Sample efficiency is a big problem for *Deep Reinforcement Learning* (DRL):
  - Large amounts of experience needed to learn an individual task, especially in continuous spaces [1]
  - No general method of transfer to new tasks
- *Meta-Reinforcement Learning* (MetaRL) tries to enable agents to learn new tasks from small amounts of experience [2-4]:
  - Leverages the data of past tasks and experiences to learn informative priors
- *Hierarchical Reinforcement Learning* (HRL) is a framework for learning temporally extended actions to accelerate learning by dividing a problem into a set of shorter horizon subproblems [4-6].
- We believe that combining these approaches can lead to the benefits of both in one framework, further improving the sample efficiency over DRL tasks.

Contribution:

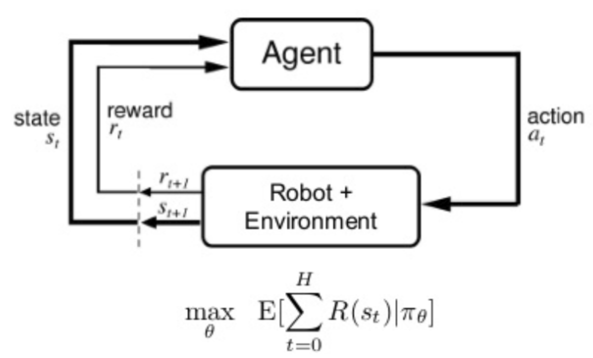
Probabilistic Embeddings for Actor Critic with Hierarchies (PEACH) 🍑:

- Leverages information about the dynamics and rewards of our task to infer a latent variable representing the task (Context Variable) [3]
- Uses Context Variable to generate a high level goal given the task to direct a hierarchy of policies similar to a Hierarchical Actor Critic [6]
- Decouples the **task-learning** of the multiple levels of the hierarchical policy to achieve the high level goal via sub-goals and the **meta-learning** of the context variables (tasks representations) for the high level policy proposing high level goals.

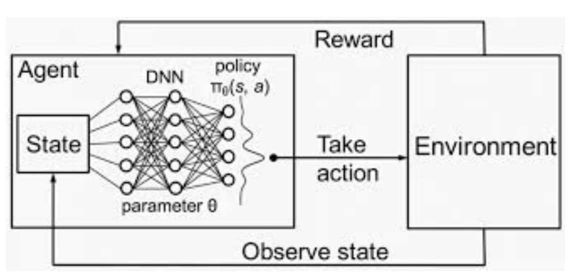


Background:

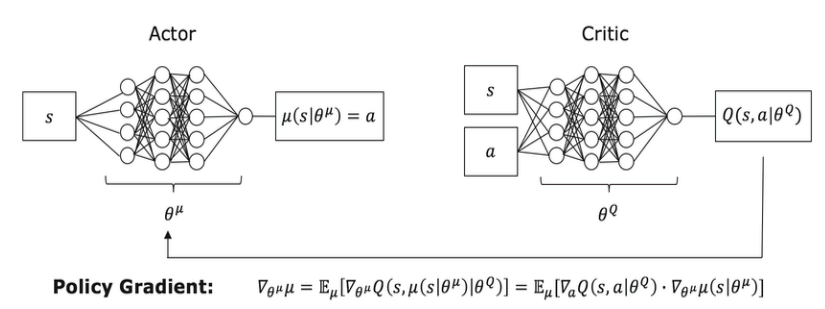
(a)



(b)



(c)



a. Reinforcement Learning is a mathematical framework for reasoning about how to learn a suitable policy (state to action mapping) to maximize a reward signal in an environment [7].

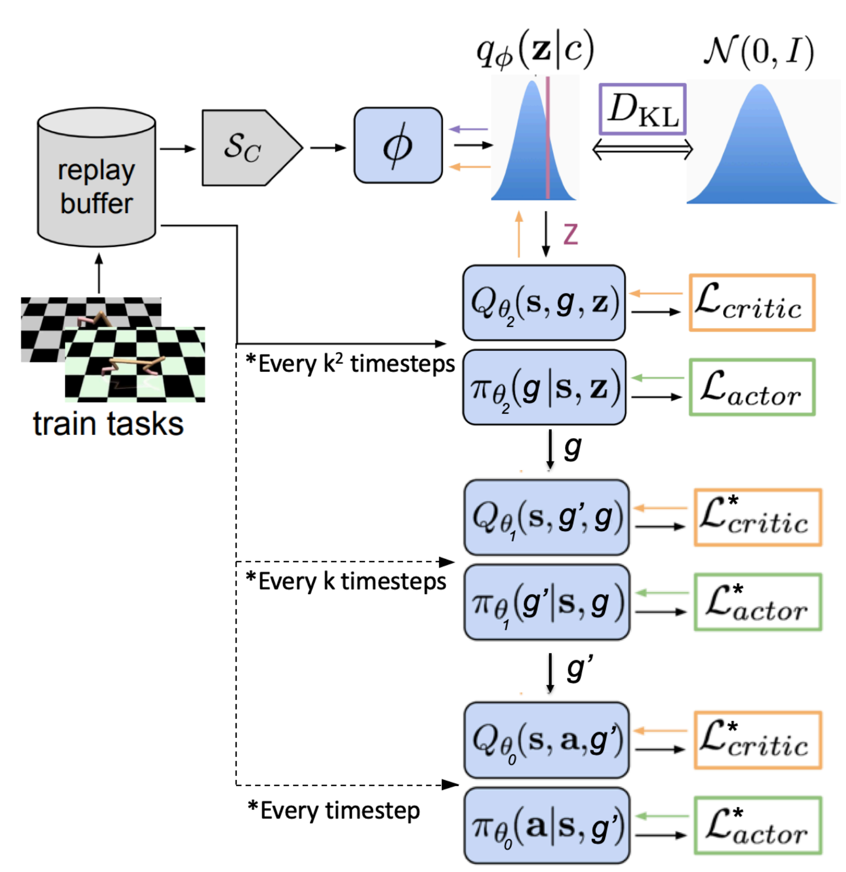
b. Deep Reinforcement Learning models our RL agents policy with a Deep Neural Network.

c. Actor-Critic algorithms combine value-based and policy-gradient methods to learn a policy in RL settings [8-9].

Problem Formulation:

Example PEACH Hierarchy *(Right)*:

- While training on a task, Z is selected by posterior sampling with updated context and given to hierarchies which produce sub-goals for lower policies, all reasoning at different timescales (dotted line).
- $q_\phi(z|c)$ is meta-trained off-policy over a batch of tasks (See Algorithm 1), along with high level actor and critic. The other levels are trained off-policy too with *Hindsight Experience Replay* and goal-based reward function (*).



Computing z:

$c^{0:t} = \{(s_i, a_i, r_i, s_{i+1}) : i \in \{0,1,...t\}\}$ this represents the state transitions sampled from the environment during a task.

$q_\phi(\mu, \sigma|c)$ where $c = (s, a, r, s)$ is a transition in the environment and μ is the mean and σ is the std dev of Gaussian distribution representing the probability that transition is from a context $z \in \mathbb{R}$

$q_\phi(\mu, \sigma|c^{0:t}) = \Gamma_{i=0}^{t} q(\mu, \sigma|c^i)$ the joint probability the trajectory is from a context $z \in \mathbb{R}$

*Γ is a product of Gaussians and can be thought of as updating the prior given data to get a posterior

PEACH Implementation:

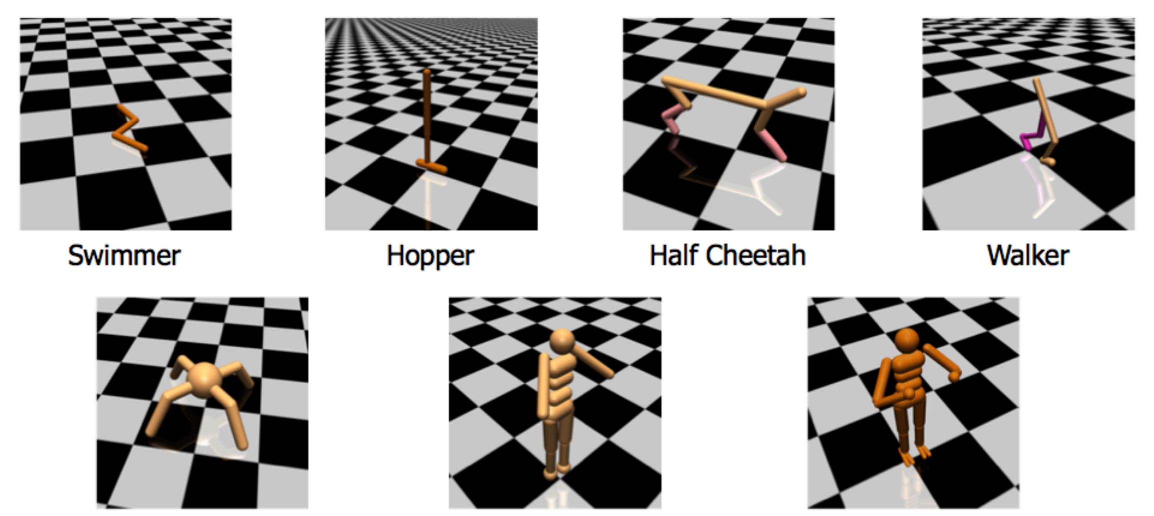
- Gather transitions for Context Variable meta-training batch + train hierarchy of policies.
- Compute training loss for high level goal actor, critic, and Context Variable KL Divergence Loss.
- Update high level actor, critic, and Context Variable function via gradient step.

Algorithm 1: PEACH Meta-training

```
Bath of training tasks {T_i}_{i=1...T} from p(T)
learning rates = α_1, α_2, α_3
Initialize replay Buffers B^i for each training task
while not done do
    for each T_i do
        Initialize context c^i = {}
        for l = 1,...,L do
            Sample z ~ q_φ(z|c^i)
            TRAIN-LEVEL(k-1, s, z) and add every {(s, a, s', r)} to B^i
            Update lower actor and critic networks θ_{π_{H-1}}, ..., θ_{π_0} and θ_{Q_{H-1}}, ..., θ_{Q_0}
            Update c^i = {(s_j, a_j, s'_j, r_j)}_{j:1...N} ~ B^i
        end
    end
    for each gradient step do
        for each T_i do
            Sample context c^i ~ S_c(B^i) and RL batch b^i ~ B^i
            Sample z ~ q_φ(z|c^i)
            L^i_actor = L_actor(b^i, z)
            L^i_critic = L_critic(b^i, z)
            L^i_KL = β D_KL(q(z|c^i)||r(z))
        end
        Update context variable encoder:
        φ → φ − α_1 ∇_φ Σ_i (L^i_critic + L^i_KL)
        Update top actor critic:
        θ_{π_H} → θ_{π_H} − α_2 ∇_θ Σ_i L^i_actor
        θ_{Q_H} → θ_{Q_H} − α_3 ∇_θ Σ_i L^i_critic
    end
end
```

Ongoing Work:

- Testing Framework on Mujoco environments and comparing results to non-hierarchical approaches
- Understanding if hierarchies have direct transfer benefits to new environment with the same transition dynamics, but different reward function
- Finding the sample efficiency of using context variables to differentiate task, especially in sparse reward settings


Swimmer Hopper Half Cheetah Walker
Ant Simplified Humanoid Full Humanoid

References:

[1] Y. Duan, X. Chen, R. Houthooft, J. Schulman, P. Abbeel. *Benchmarking Deep Reinforcement Learning for Continuous Control*. ICML, 2016.
[2] C. Finn, P. Abbeel, S. Levine. *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. ICML, 2017.
[3] K. Rakelly, A. Zhou, D. Quillen, C. Finn, S. Levine. *Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables*. ICLR, 2019.
[4] K. Frans, J. Ho, X. Chen, P. Abbeel, J. Schulman. *Meta Learning Shared Hierarchies*. ICLR, 2018.
[5] P. Bacon, J. Harb, D. Precup. *The Option-Critic Architecture*. AAAI, 2017.
[6] A. Levy, G. Konidaris, R. Platt, K. Saenko. *Learning Multi-Level Hierarchies with Hindsight*. ICLR 2019.
[7] R. Sutton, A. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1999.
[8] V. Mnih, A. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu. *Asynchronous Methods for Deep Reinforcement Learning*. ICML, 2016.
[9] R. Sutton, D. McAllester, S. Singh, Y. Mansour. *Policy Gradient Methods for Reinforcement Learning with Function Approximation*. NeurIPS, 1999.

Research funded by the Brown CS Startup Fund, under Dr. George Konidaris