Synthesized Policies for Transfer and Adaptation across Tasks and Environments

Hexiang Hu*[1], Liyu Chen*[1], Boqing Gong[2], Fei Sha[1,3]

[1]University of Southern California, [2]Tencent AI Lab, [3]Netfilx

Poster #155

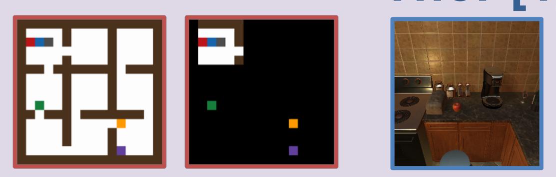
NeurIPS

NETFLIX

Highlights:

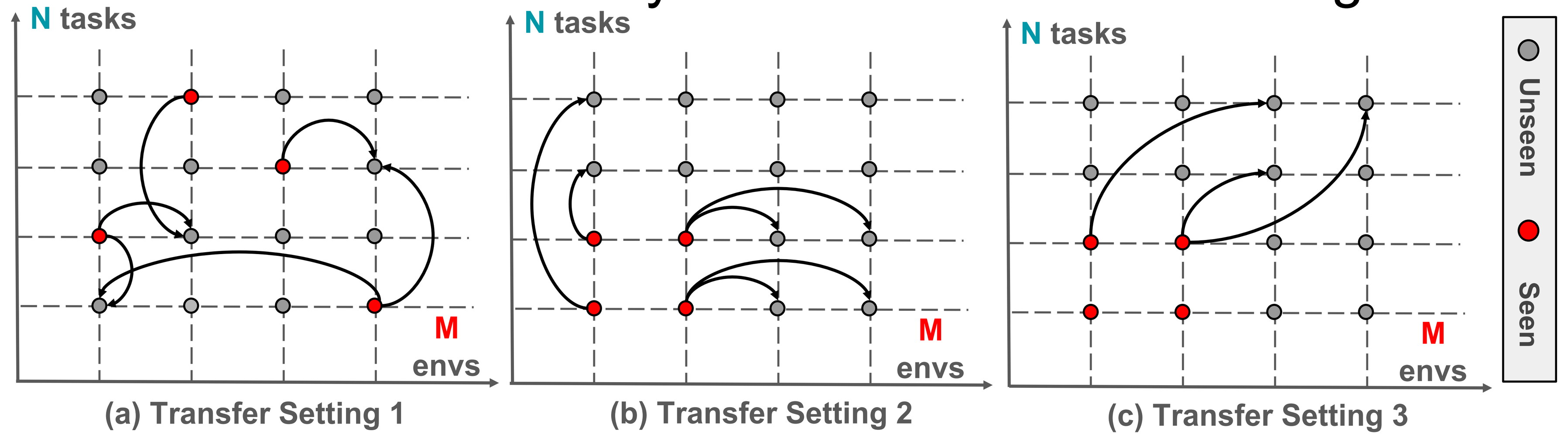
- Study three progressively difficult transfer settings where an agent needs to transfer and adapt across both environments ($\boldsymbol{\varepsilon}$) and tasks ($\boldsymbol{\tau}$) simultaneously.
- Novel architecture of policy and reward factorization and disentanglement objective.
- We experiment on two simulators, and our approach has achieved superior performances under different transfer settings

GridWorld Thor [1]



Problem Setting:

We consider adaption and transfer across environments and tasks simultaneously in Reinforcement Learning.



(a) Transfer Setting 1 (b) Transfer Setting 2 (c) Transfer Setting 3

Goal: Learn $\mathcal{O}(|\mathcal{E}| + |\mathcal{T}|)$ combos and generalize to $|\mathcal{E}| \times |\mathcal{T}|$

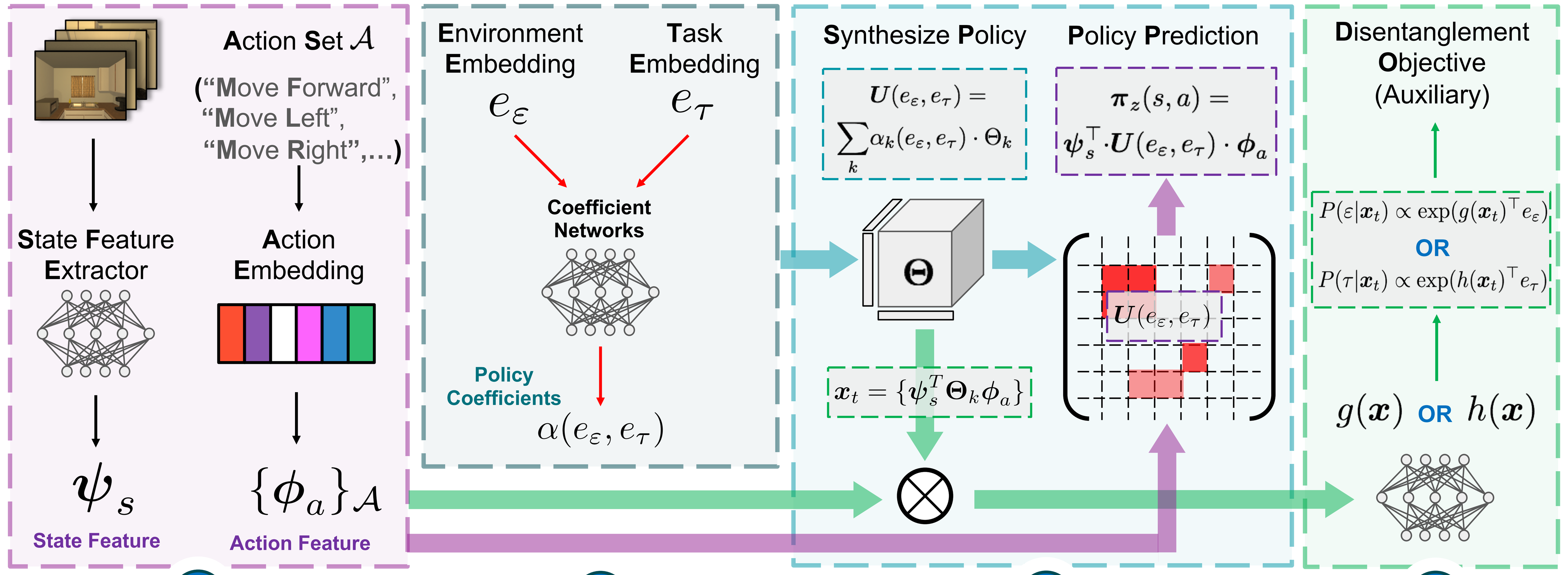
(a): Compositional generalization to novel combos

(b) & (c): "Incremental transfer" or "Learn a big jump"

References:

[1] Kolve, Eric, et al. "AI2-THOR: An interactive 3d environment for visual AI." arXiv preprint arXiv:1712.05474 (2017).
[2] Barreto, André, et al. "Successor features for transfer in reinforcement learning." Advances in neural information processing systems. 2017.
[3] Devin, Coline, et al. "Learning modular neural network policies for multi-task and multi-robot transfer." Robotics and Automation (ICRA), 2017, 2017 IEEE International Conference on. IEEE, 2017.

More about SynPo!



1 Extract State-Action Features 2 Compose Policy Coefficients 3 Compute State-Action Score 4 Disentanglement

Main Idea:

(1) Learn a policy basis Θ to compose (ε, τ) specific policies $U(e_\varepsilon, e_\tau)$

(2) Learn low dimensional embeddings e_ε or e_τ for novel $\boldsymbol{\varepsilon}$ and $\boldsymbol{\tau}$

Policy Synthesis: $U(e_\varepsilon, e_\tau) = \sum_k \alpha_k(e_\varepsilon, e_\tau)\cdot\boldsymbol{\Theta}_k$

$V(e_\varepsilon, e_\tau) = \sum_k \beta_k(e_\varepsilon, e_\tau)\cdot\boldsymbol{\Theta}_k$

Policy Prediction: $\pi_z(a,s) \propto \exp(\boldsymbol{\psi}_s^T U(e_\varepsilon, e_\tau)\boldsymbol{\phi}_a + b_\pi)$

Reward Prediction: $\tilde{r}_z(s,a) = \boldsymbol{\psi}_s^T V(e_\varepsilon, e_\tau)\boldsymbol{\phi}_a + b_r$

Disentanglement Objective:

$l_\epsilon = -\sum_t \log P(\epsilon|\boldsymbol{x}_t), \text{with } P(\epsilon|\boldsymbol{x}_t) \propto \exp(g(\boldsymbol{x}_t)^T e_\epsilon)$

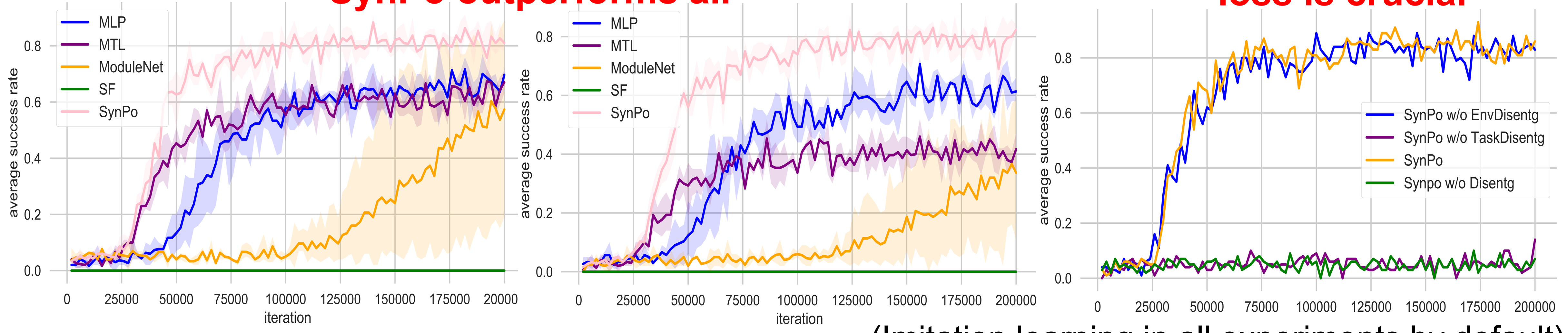
$l_\tau = -\sum_t \log P(\tau|\boldsymbol{x}_t), \text{with } P(\tau|\boldsymbol{x}_t) \propto \exp(h(\boldsymbol{x}_t)^T e_\tau)$

Experiments on GridWorld:

Results for Transfer Setting 1

SynPo outperforms all

Task disentanglement loss is crucial



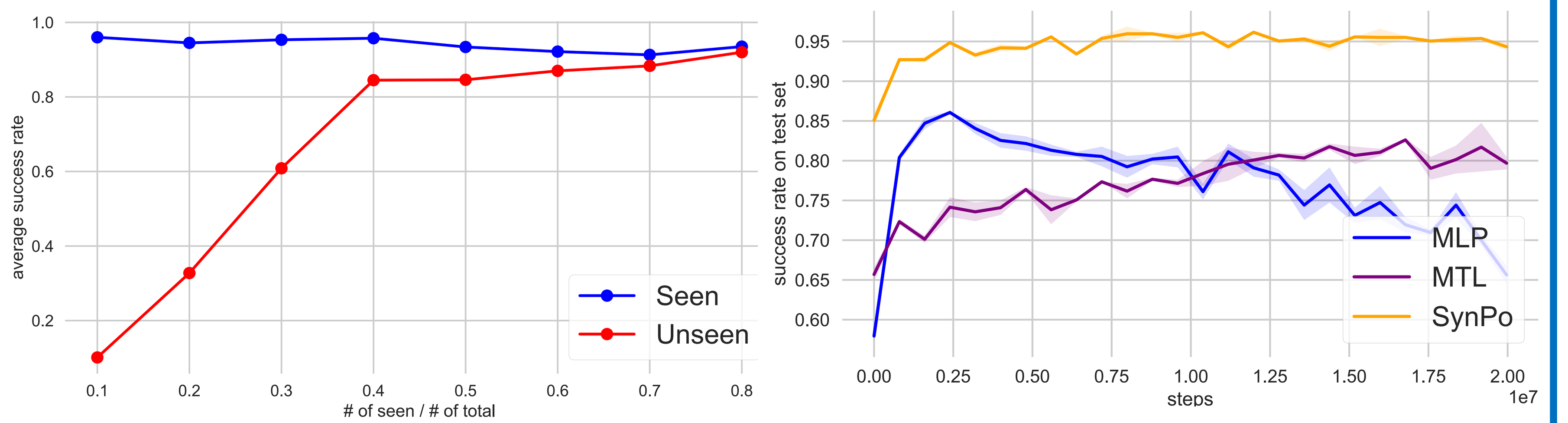
(Imitation learning in all experiments by default)

How many seen $(\boldsymbol{\varepsilon}, \boldsymbol{\tau})$ pairs are needed to transfer well?

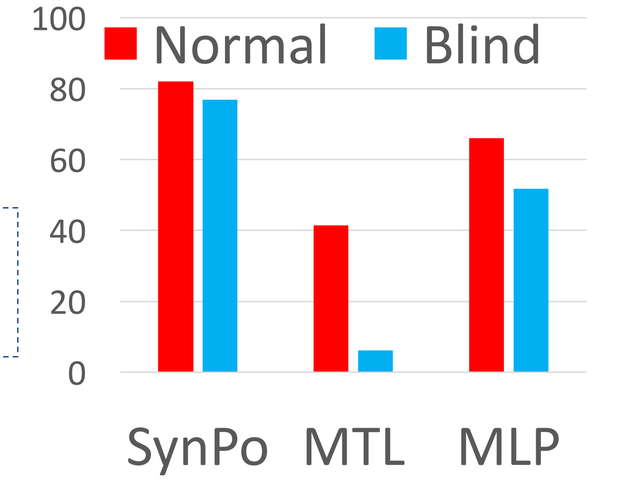
40% of the training $(\boldsymbol{\varepsilon}, \boldsymbol{\tau})$ pairs can generalize

Does reinforcement learning help transfer?

Fine tuning with RL on training set helps transfer in test set!



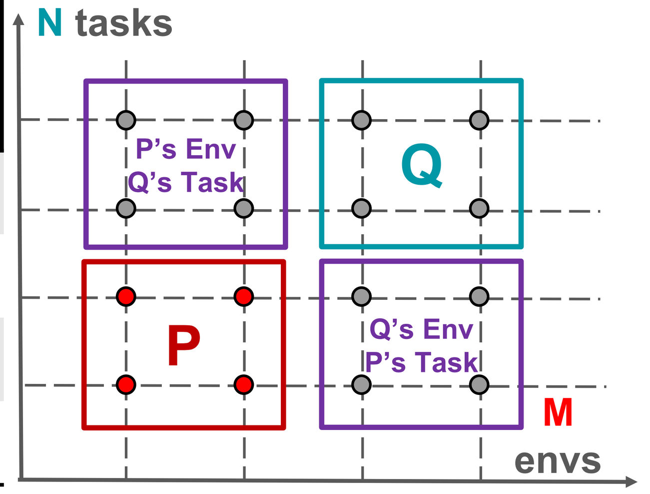
SynPo is less sensitive to Imperfect perception.



Blind Agent: the agent can only see its own and treasures' position, **but not the maze.**

Results for Transfer Setting 2 & 3

Setting	Method	Q's ε, P's τ	P's ε, Q's τ	Q Pairs
Setting 2	MLP	13.8%	20.7%	6.3%
	SynPo	**50.5%**	**21.5%**	**13.5%**
Setting 3	MLP	14.6%	18.3%	7.2%
	SynPo	**42.7%**	**19.4%**	**12.9%**



Experiments on THOR [1]:

Results for Transfer Setting 1

Split	ModuleNet	MLP	MTL	SynPo
SEEN	51.5%	47.5%	52.2%	**55.6%**
UNSEEN	14.4%	25.8%	33.3%	**35.4%**