Transfer learning using successor state features

Lucas Lehnert

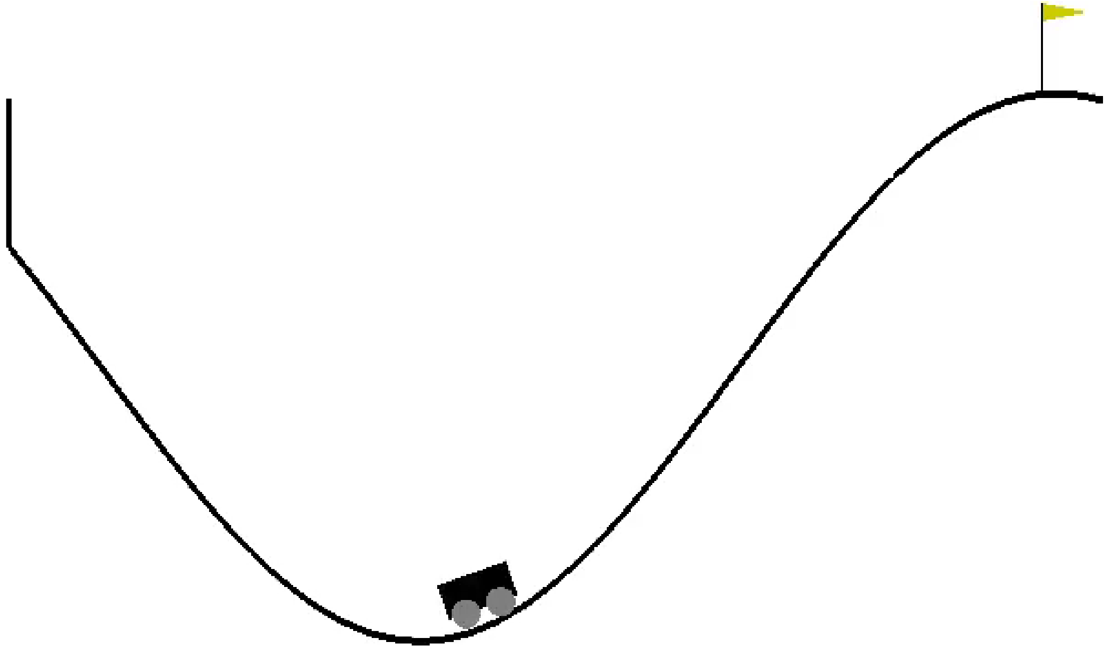
Joint work with Stefanie Tellex and Michael L. Littman

Brown University

Reinforcement Learning

Action $a_t = \pi(s_t)$

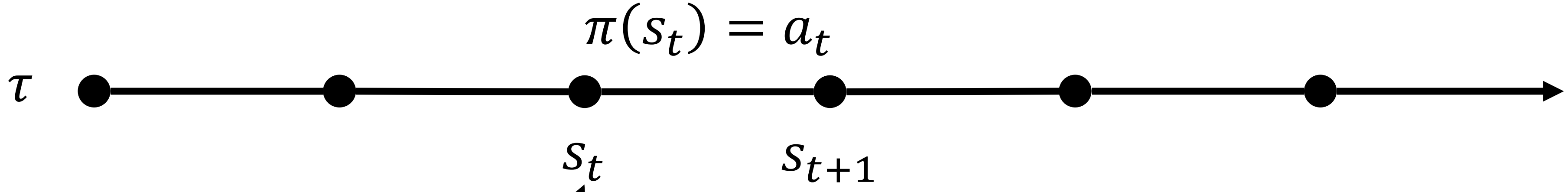
State $s_{t+1} \sim p(s_t, a_t, \cdot)$
Reward $r_t = r(s_t, a_t)$



This framework is called an MDP $M = \langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$.
The agent selects actions according to a policy $\pi: \mathcal{S} \rightarrow \mathcal{A}$.

Value Functions

Say we have a trajectory:



$$\boxed{Q^{\pi}(s_t, a_t)} = \mathbb{E}^{\pi}[r(s_t, a_t) + \gamma r(s_{t+1}, a_{t+1}) + \gamma^2 r(s_{t+2}, a_{t+2}) + \cdots]$$

The **discount factor** $\gamma \in [0,1)$ favours immediate rewards.

Value Functions

Q-functions encode information about

1. which rewards the agent receives, and
2. which next states the agent sees and in which order.

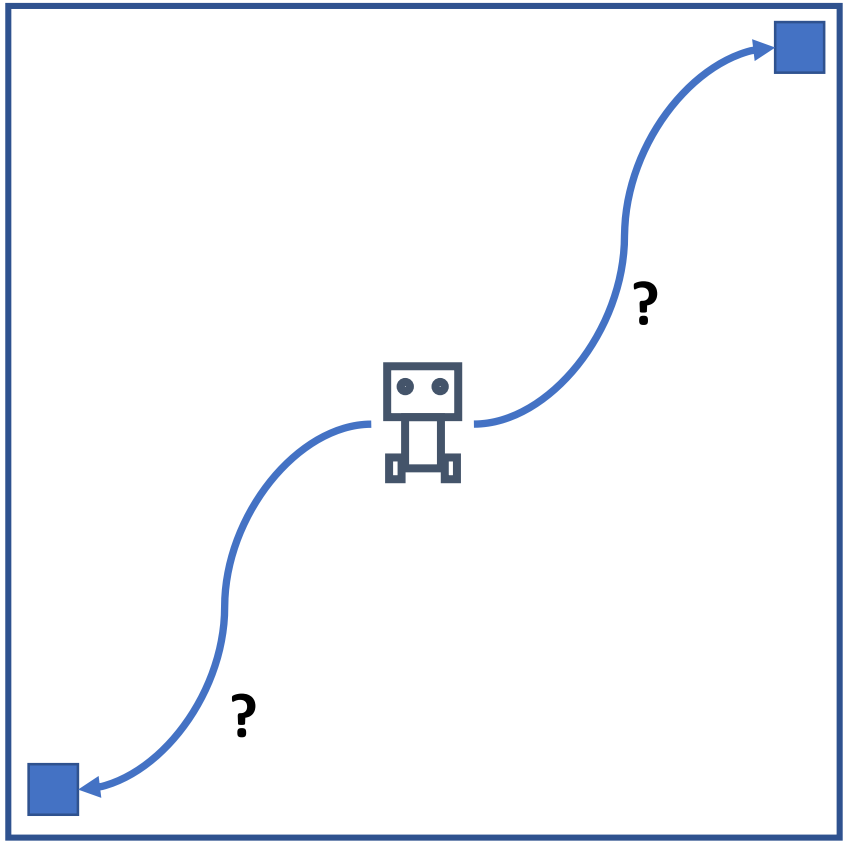
Using Successor Features for Transfer

Problem: Say we have a set of MDPs where only the reward function changes:

$$\{M_i = \langle \mathcal{S}, \mathcal{A}, p, r_i, \gamma \rangle\}$$

Example: Navigating between different goal locations. The goal location (reward function) might change.

Goal: Learn a feature representation so that we can adapt quickly to a new reward function.

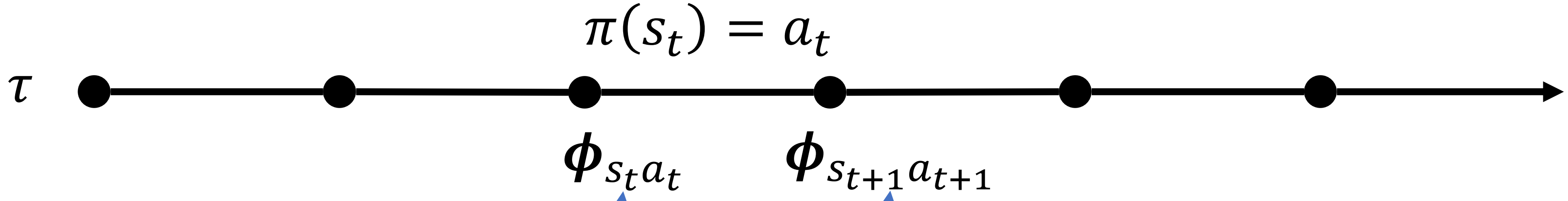


Successor Features (SF) [Baretto et al., 2016]

Every state-action pair has a feature $\boldsymbol{\phi}_{sa}$.

- These features are used to predict rewards: $\boldsymbol{\phi}_{sa}^T \boldsymbol{w} \approx r(s,a)$.

Say we have a trajectory:



Successor Features:

$$\boldsymbol{\psi}_{s_t a_t}^{\pi} = \mathbb{E}^{\pi}\left[\boldsymbol{\phi}_{s_t a_t} + \gamma \boldsymbol{\phi}_{s_{t+1} a_{t+1}} + \gamma^2 \boldsymbol{\phi}_{s_{t+2} a_{t+2}} + \gamma^3 \boldsymbol{\phi}_{s_{t+3} a_{t+3}} + \cdots\right]$$

Successor Features (SF) [Baretto et al., 2016]

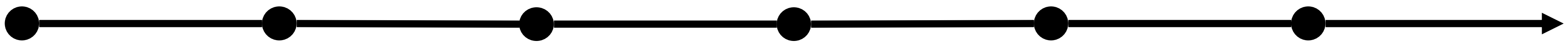
Every state-action pair has a feature $\boldsymbol{\phi}_{sa}$.

- These features are used to predict rewards: $\boldsymbol{\phi}_{sa}^T \boldsymbol{w} \approx r(s,a)$.

Say we have a trajectory:

$\pi(s_t) = a_t$

τ



$\boldsymbol{\phi}_{s_t a_t}$ $\quad$ $\boldsymbol{\psi}_{s_{t+1} a_{t+1}}$

Successor Features:

$$\boldsymbol{\psi}_{s_t a_t}^{\pi} = \mathbb{E}^{\pi}\left[\boxed{\boldsymbol{\phi}_{s_t a_t}} + \gamma \boxed{\boldsymbol{\psi}_{s_{t+1} a_{t+1}}^{\pi}}\right]$$

The same trick we use for value functions.

Successor Features (SF) [Baretto et al., 2016]

Theorem: Q-values can be computed using

1. the reward model $r(s,a) \approx \boldsymbol{\phi}_{sa}^T \boldsymbol{w}$
2. the Successor Features $\boldsymbol{\psi}_{sa}^{\pi}$

as

Reuse reward model to estimate Q-values

$$Q^{\pi}(s,a) = \underbrace{(\boldsymbol{\psi}_{sa}^{\pi})^T}_{\text{Next state prediction}} \underbrace{\boldsymbol{w}}_{\text{Reward prediction}}$$

The Fitted Successor Feature Algorithm

Similar to Fitted Q-iteration, we estimate a SF target:

$$\boldsymbol{y}_{s,a,s'} = \begin{cases} \boldsymbol{\phi}_{s,a} & \text{if } s' \text{ is terminal} \\ \boldsymbol{\phi}_{s,a} + \gamma \mathbb{E}_{a'}\left[\boldsymbol{\psi}_{s',a'}\right] & \text{otherwise} \end{cases}$$

The SFs $\boldsymbol{\psi}$ are then fitted against the target $\boldsymbol{y}_{s,a,s'}$ with the loss:

$$\mathcal{L}_{\text{SF}}(\boldsymbol{\psi}) = \mathbb{E}_{s,a,s'}\left[\left\|\boldsymbol{\psi}_{s,a} - \boldsymbol{y}_{s,a,s'}\right\|^2\right]$$

We use Adagrad (as implemented in Tensorflow) to minimize $\mathcal{L}_{\text{SF}}$ with respect to the parametrization of $\boldsymbol{\psi}$.

- Gradient update using every 100 transitions.

The Fitted Successor Feature Algorithm

Reward Model: $r(s, a) \approx \boldsymbol{\phi}_{s,a}^T \boldsymbol{w}$

The function $\boldsymbol{\phi}$ tabulates the state-action space.

Successor Feature Model: $\boldsymbol{\psi}_{s,a} = \boldsymbol{\Psi} \boldsymbol{\phi}_{s,a}$

The matrix $\boldsymbol{\Psi}$ is of size $|\mathcal{S} \times \mathcal{A}| \times |\mathcal{S} \times \mathcal{A}|$.

For finite MDPs, the true reward model and successor features can be captured.

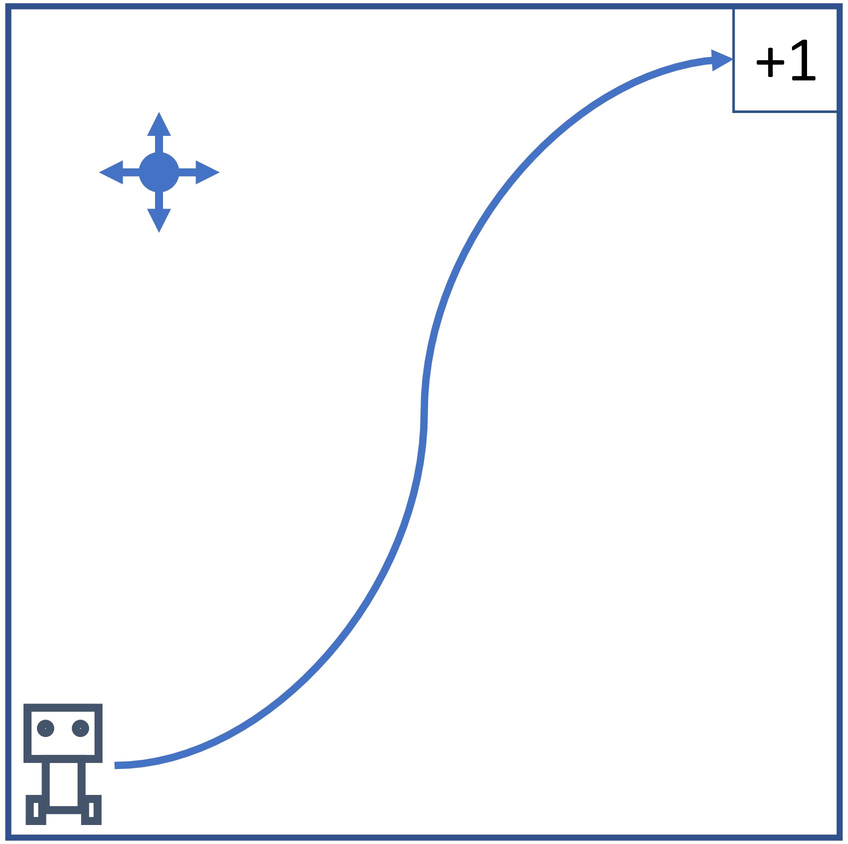
Although these models can be easily extended to use function approximation.

Grid World Navigation

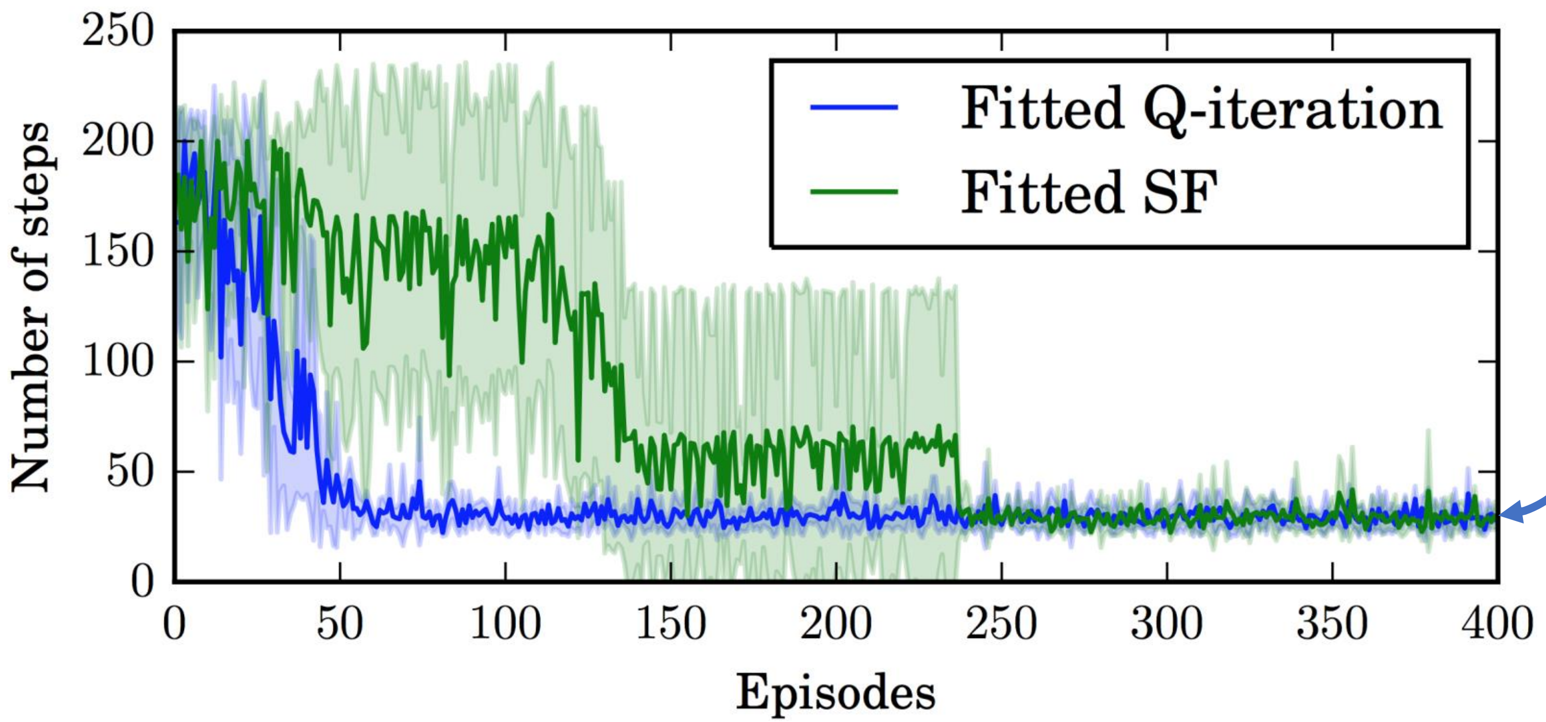
Robot has to navigate in on a 10-by-10 grid.

When robot reaches a goal it receives +1 reward.

Robot can move up, down, left, and right, and it might slip with small probability.



Single Task Navigation



The agent is constrained to select an random action with probability 0.3 (to ensure exploration).

Number of attempts to navigate.

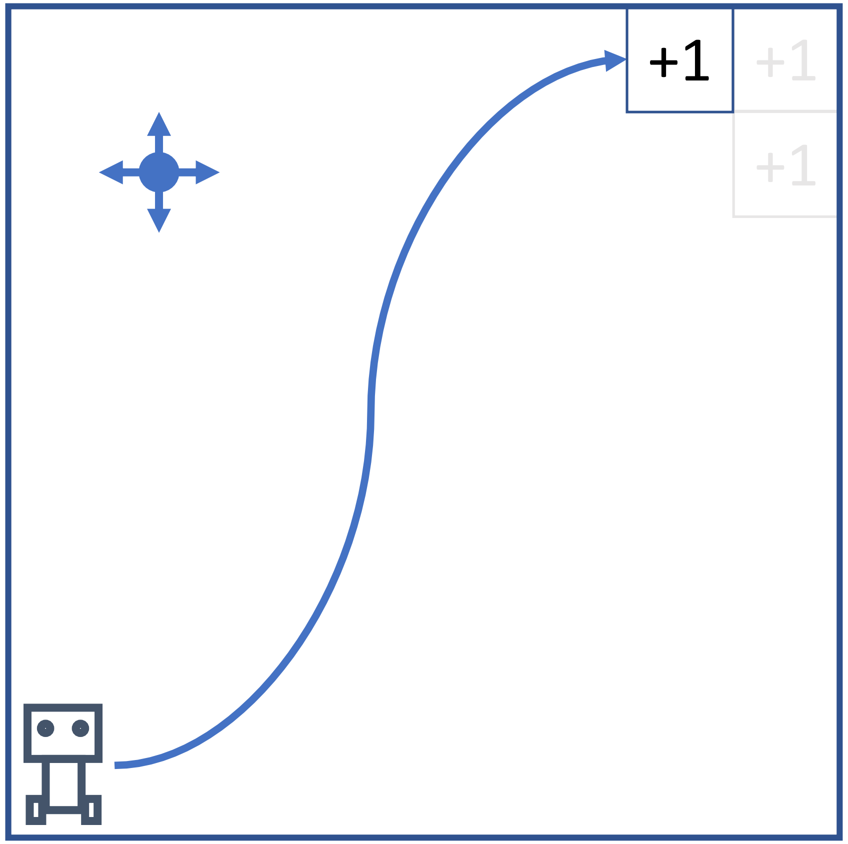
Multi Task Navigation: Slight Reward Changes

Same 10-by-10 grid world scenario as before.

The goal location is only moved by one cell.

The optimal policy changes only slightly.

Use an ε-greedy policy with $\varepsilon = 0.3$.



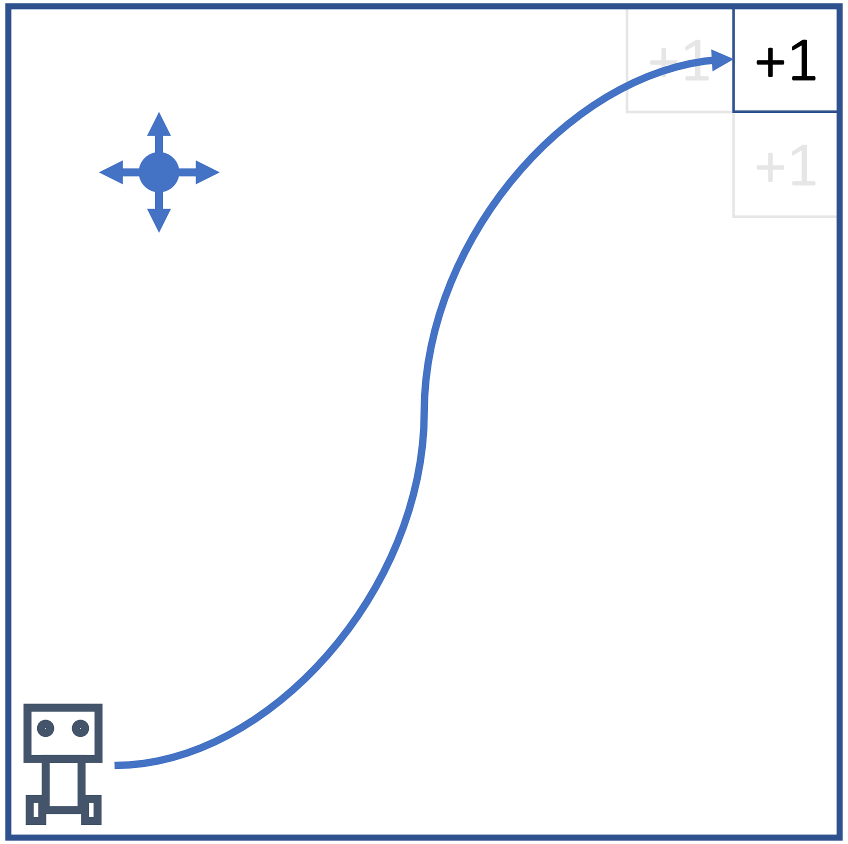
Multi Task Navigation: Slight Reward Changes

Same 10-by-10 grid world scenario as before.

The goal location is only moved by one cell.

The optimal policy changes only slightly.

Use an ε-greedy policy with $\varepsilon = 0.3$.



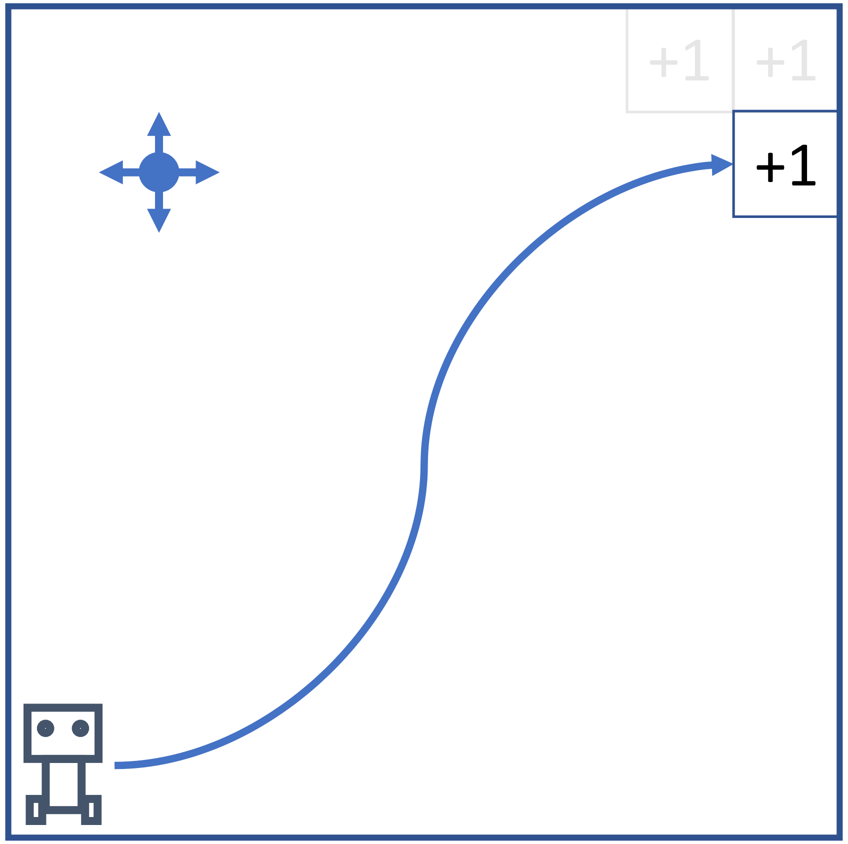
Multi Task Navigation: Slight Reward Changes

Same 10-by-10 grid world scenario as before.

The goal location is only moved by one cell.

The optimal policy changes only slightly.

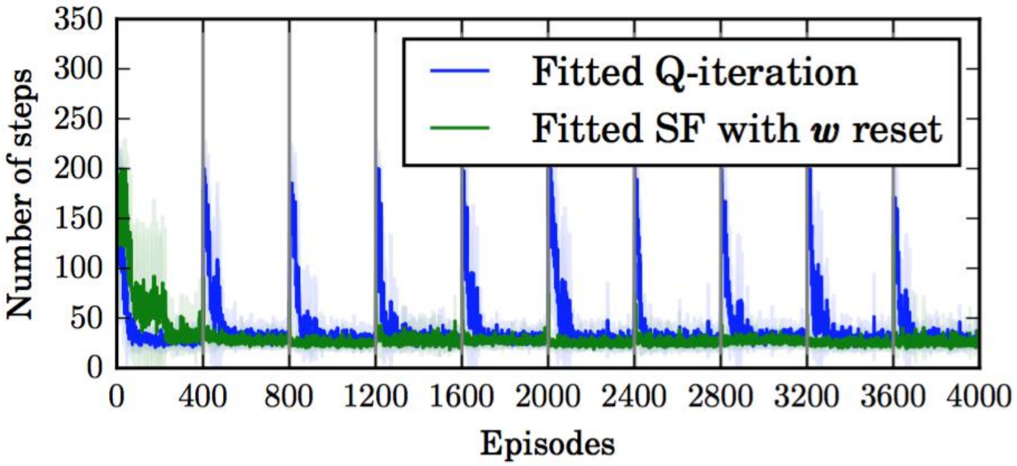
Use an ε-greedy policy with $\varepsilon = 0.3$.



SF Boost Transfer Performance

We change the goal location by one cell every 400 episodes.

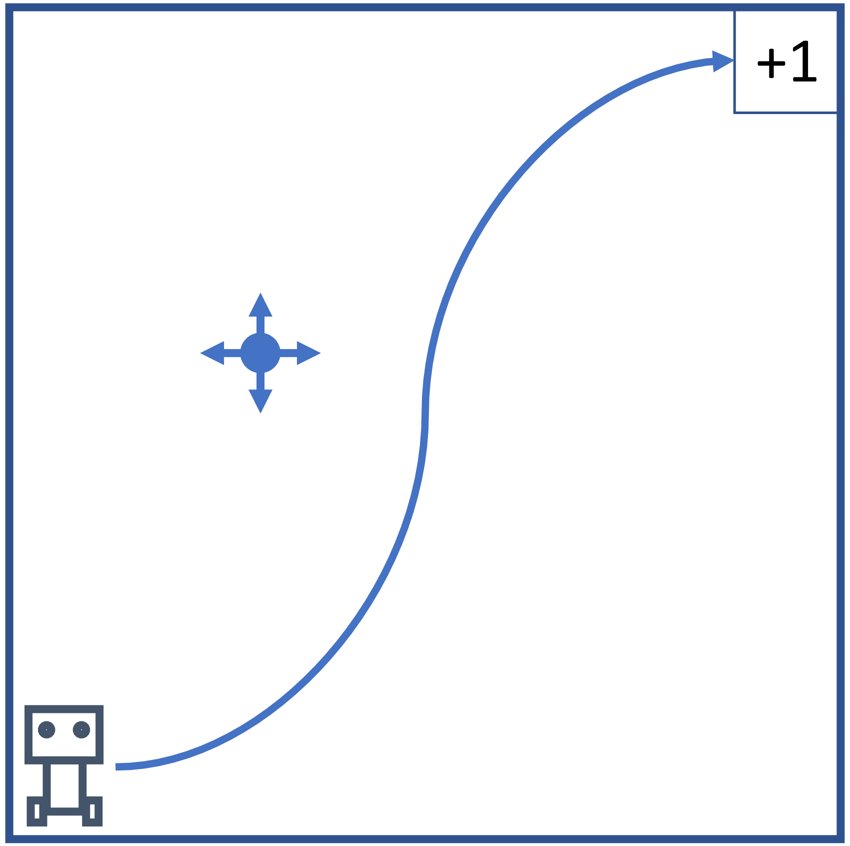
Different Reset Strategies for Fitted SF:



If SFs are not transferred, then the performance degrades and becomes similar to Fitted Q-iteration.

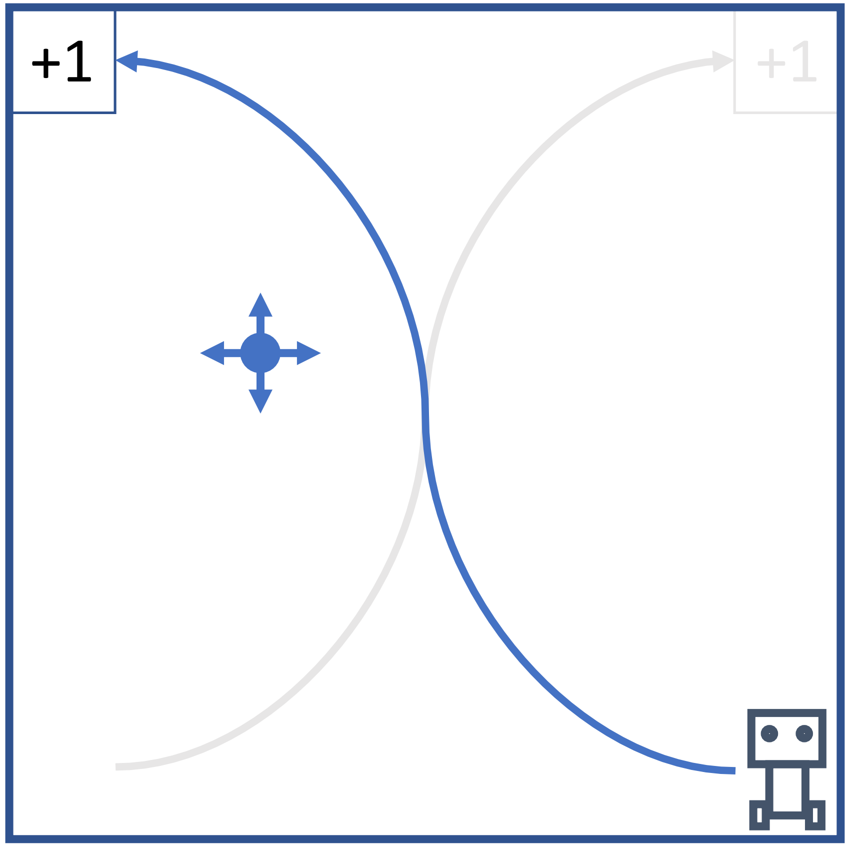
Multi Task Navigation: Significant Reward Changes

Every 100 episodes, change goal location to another corner, then repeat.



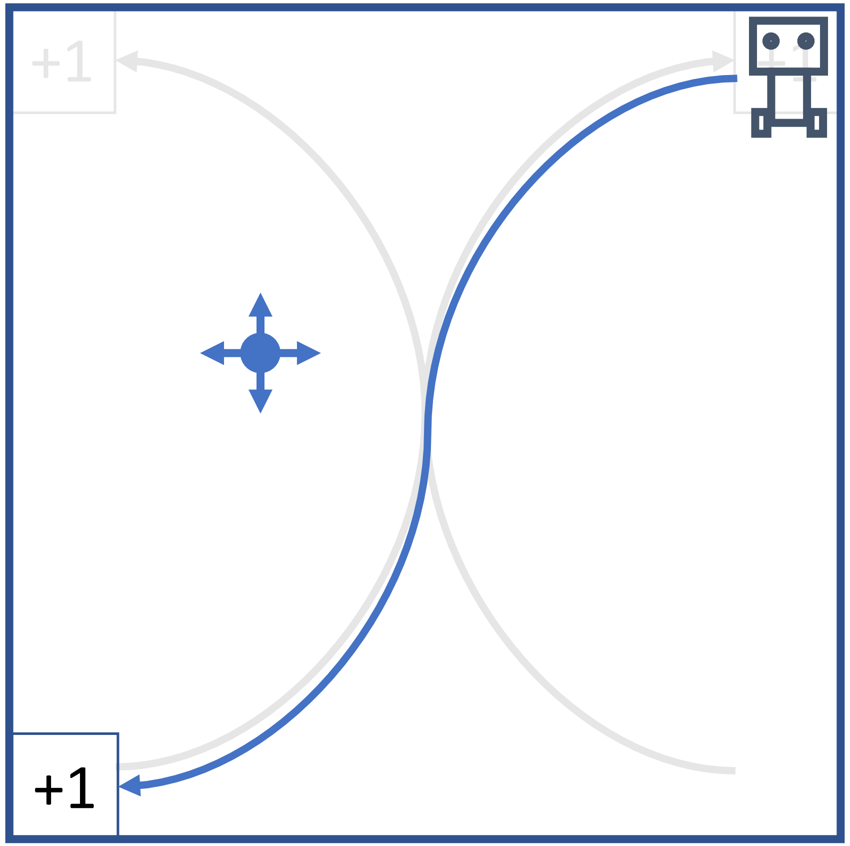
Multi Task Navigation: Significant Reward Changes

Every 100 episodes, change goal location to another corner, then repeat.



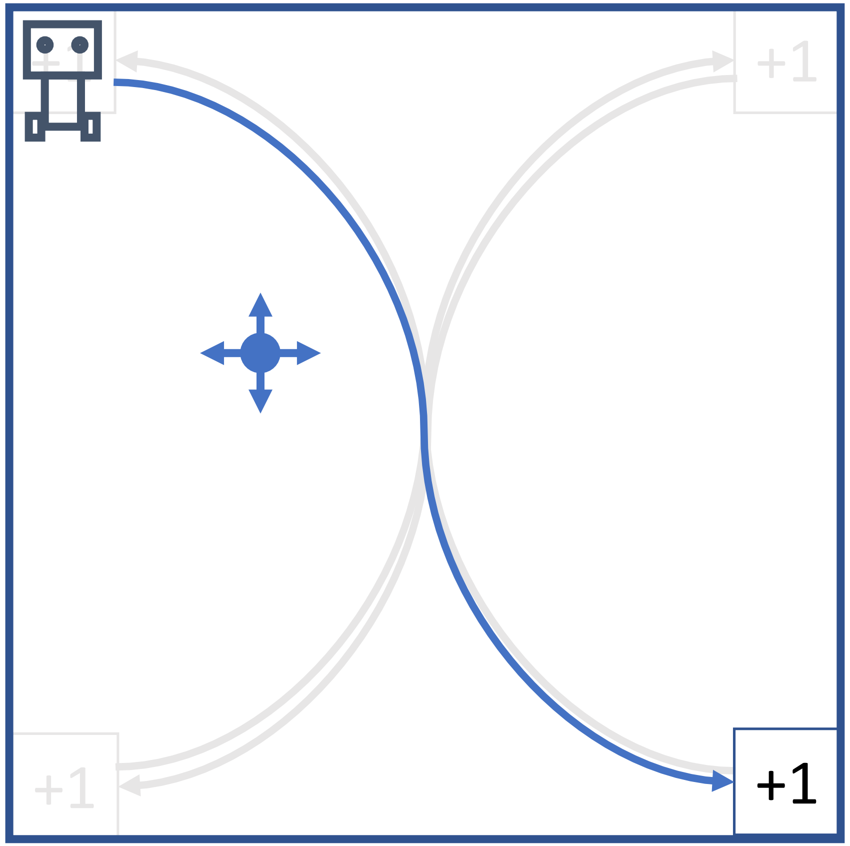
Multi Task Navigation: Significant Reward Changes

Every 100 episodes, change goal location to another corner, then repeat.



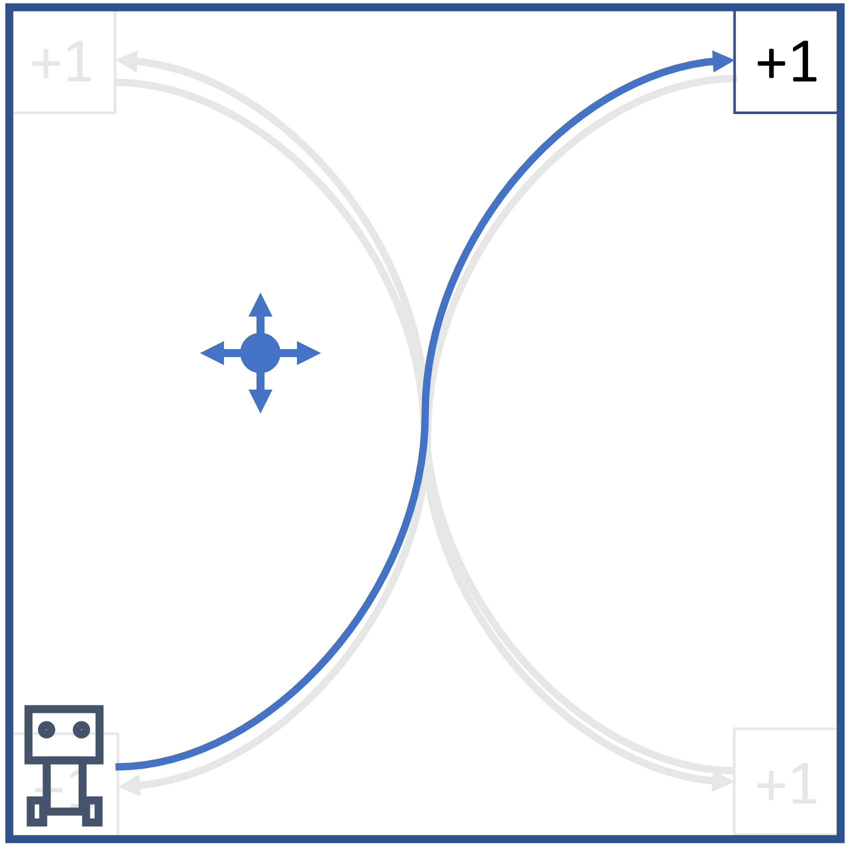
Multi Task Navigation: Significant Reward Changes

Every 100 episodes, change goal location to another corner, then repeat.

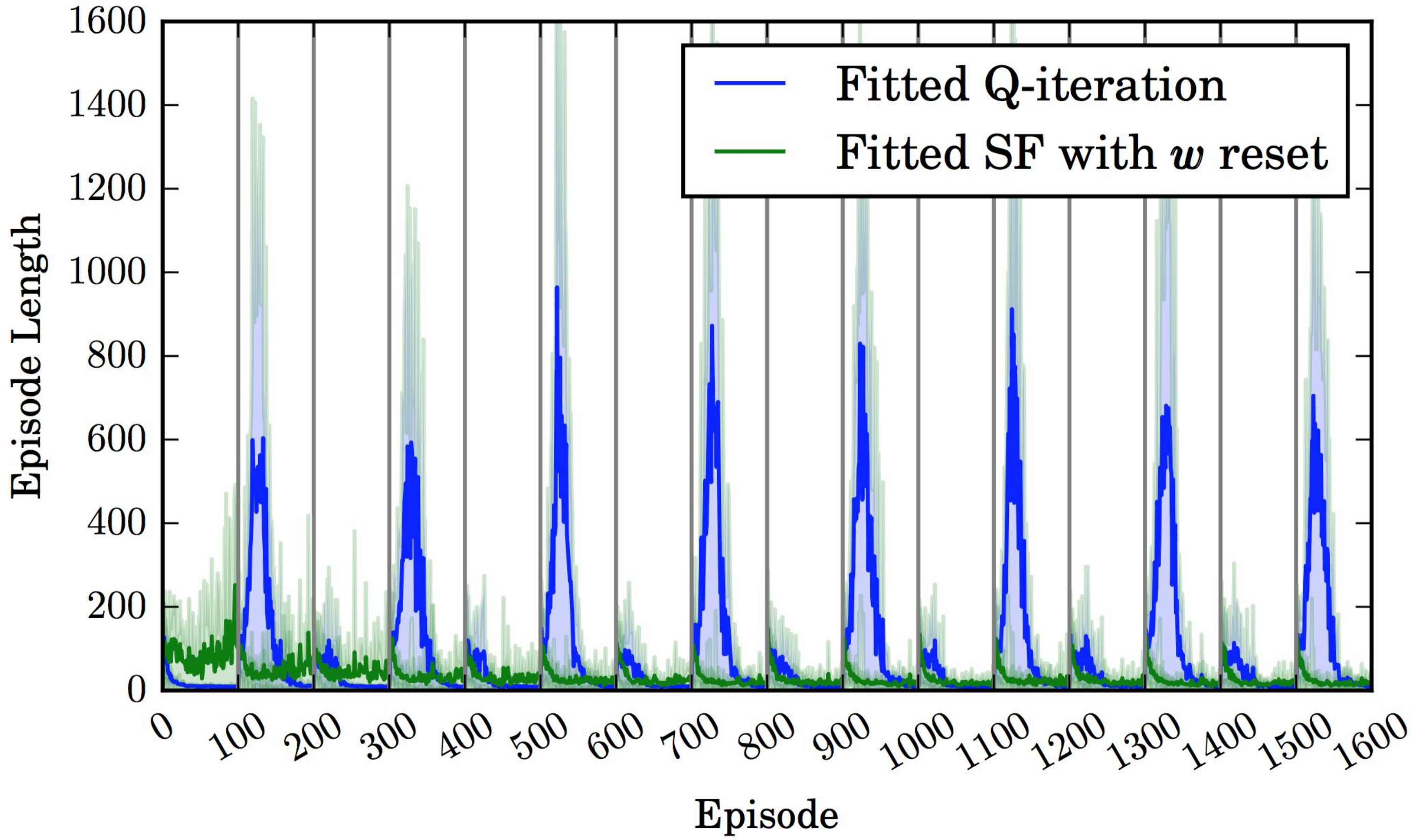


Multi Task Navigation: Significant Reward Changes

Every 100 episodes, change goal location to another corner, then repeat.



Multi Task Navigation: Significant Reward Changes



The significant reward function change requires to re-explore the environment.

The ε-greedy policy was changed from

$\varepsilon = 1.0$ down to 0.1.

Multi Task Navigation: Significant Reward Changes

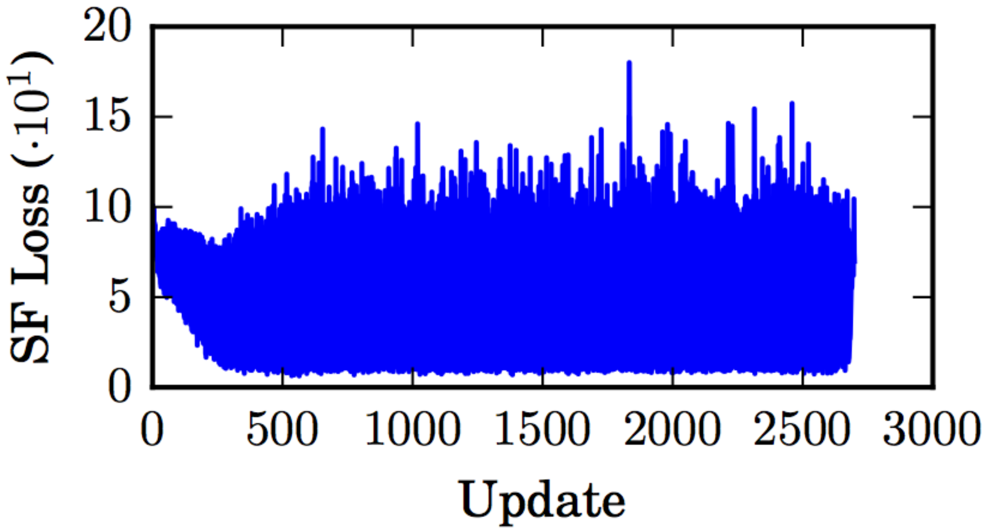
	Fitted Q-iteration	**Fitted SF**	**Welch's t-test *p*-value**
Average episode length	99.46 ± 10.43	34.50 ± 2.17	$\boxed{1.90 \cdot 10^{-17}}$

The probability of accidentally seeing different performance between the two methods.

Multi Task Navigation: Significant Reward Changes

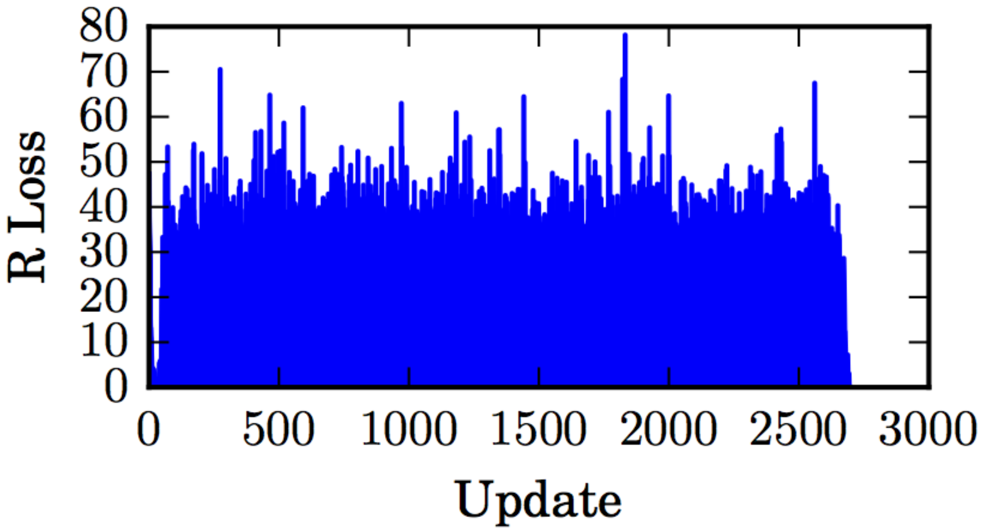
Successor Feature Loss

How badly do the feature satisfy the learning target?



The SF are adjusted for every task!
Good exploration (annealing ε) helps.

Reward Prediction Loss

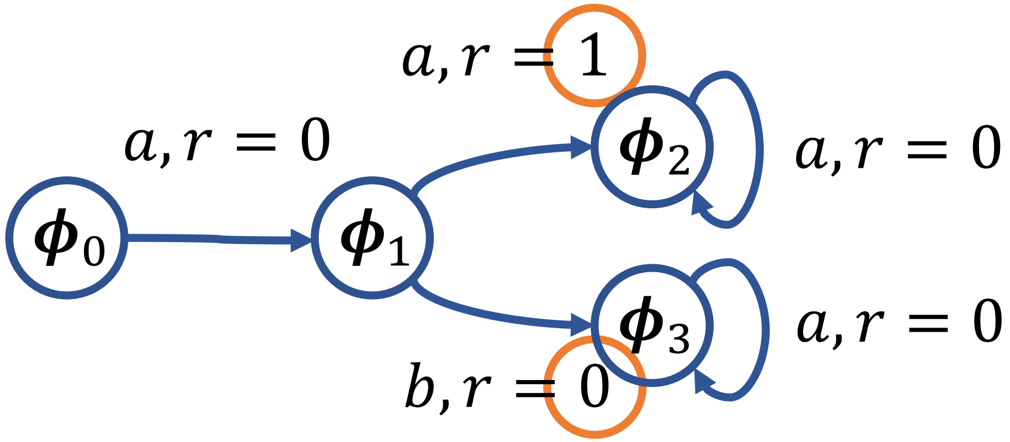


Oscillations are expected, because we keep changing the reward model.

Example: Transfer with SF

Task 1: Policy π_{aa}, select action a, then a again:

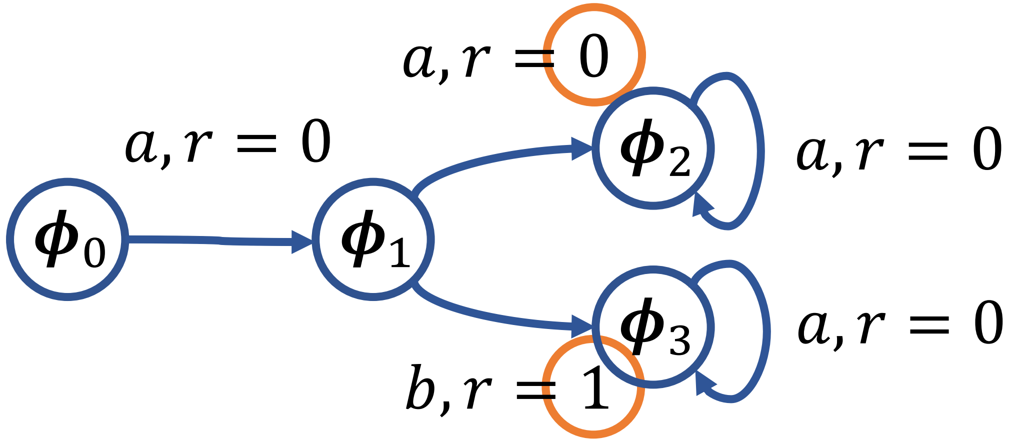
$$\boldsymbol{\psi}_{0a}^{\pi_{aa}} = \boldsymbol{\phi}_{0a} + \gamma\boldsymbol{\phi}_{1a} + \mathbb{E}^{\pi_{aa}}\left[\boldsymbol{\psi}_{2a}^{\pi_{aa}}\right]$$



$$\neq$$

Task 2: Policy π_{ab}, select action a, then b again:

$$\boldsymbol{\psi}_{0a}^{\pi_{ab}} = \boldsymbol{\phi}_{0a} + \gamma\boldsymbol{\phi}_{1b} + \mathbb{E}^{\pi_{ab}}\left[\boldsymbol{\psi}_{3a}^{\pi_{ab}}\right]$$



Conclusion

One task's **Successor Feature representation has to be re-learned** for another.

- The Successor Feature representation only initializes policy search.

However, we have shown that **Successor Features can significantly improve transfer** in RL across tasks with changing reward structure.

Thank you.

Related Paper:

Lucas Lehnert, Stefanie Tellex, and Michael L. Littman

Advantages and Limitations of using Successor Features for Transfer in Reinforcement Learning

Lifelong Learning: A Reinforcement Learning Approach Workshop @ICML, Sydney, Australia, 2017 [arXiv]