Successor Feature Sets: Generalizing Successor Representations Across Policies

Kianté Brantley[1], Soroush Mehri[2], Geoffrey J. Gordon[2]

[1]University of Maryland College Park, [2]Microsoft Research

Contributions

For any *control problem*, we define three sets of embeddings

- a convex set of possible state vectors q
- a convex set of reward function representations r
- a convex set of policy embedding vectors π

such that the value of a policy is a *(simple) multilinear* function of r, π, and q.

Using these embeddings we get the "best of all worlds" from well-understood ideas:

- predictive state representations (PSRs) (generalize over states)
- successor features (generalize over tasks or rewards)
- POMDP value iteration (generalize over polices)

New Dynamic Programming method

- "Bellman-like" consistency equation is a contraction
- generalizes the value iteration algorithm for POMDPs or PSRs
- once computed, embeddings can be used for either planning or imitation

Background

World Model: state $q_t \overset{\text{act } a_t}{\Longrightarrow} P_t \underset{\text{obs } o_t}{\overset{\text{actual}}{\Longrightarrow}} q_{t+1}$

(P_t predicted observation probabilities)

For example, POMDP: *Tiger Problem*

$$P_t(o) = u^T T_{a_t o} q_t$$
$$q_{t+1} = T_{a_t o_t} q_t / P_t(o_t) \quad (*)$$

$$q_1 = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}, u = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, T_{W\ell} = \begin{pmatrix} \frac{1}{2}+\epsilon & 0 & 0 \\ 0 & \frac{1}{2}-\epsilon & 0 \\ 0 & 0 & 0 \end{pmatrix} T_{Wr} = \begin{pmatrix} \frac{1}{2}-\epsilon & 0 & 0 \\ 0 & \frac{1}{2}+\epsilon & 0 \\ 0 & 0 & 0 \end{pmatrix},$$
$$T_{L\ell} = T_{R\ell} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, T_{Lr} = T_{Rr} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, T_{L\omega} = T_{R\omega} = T_{W\omega} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Belief state q_t and update (*) satisfy:

1. $\forall a, o, t : u^T T_{ao} q_t \geq 0$
2. $\forall a, t : \sum_o u^T T_{ao} q_t = 1$

Our approach: Successor Feature Sets

- **How do we embed states? Predictive states**
  We can use a PSR directly or convert an MDP or POMDP like Tiger to a PSR. Whenever the state update satisfies *(*)* the state vector is compatible with our task and policy embeddings.

- **How do we embed rewards? Successor features**
  Reward: $r(q, a) = r^T f(q, a)$ for some vector r and feature function f. Suppose wlog that f is linear in q for each a: for matrices F_a, $f(q, a) = F_a q$. Then we can write the state-action value function as:

$$Q(q, a) = \mathbb{E}_\pi \left[\sum_{t=1}^{H} \gamma^{t-1} r^T F_{a_t} q_t \mid \text{do } q_1 = q, a_1 = a \right]$$

  We can pull out r^T and write as $Q^\pi(q, a) = r^T \phi^\pi(q, a)$, where the *successor feature* function ϕ is defined as:

$$\phi^\pi(q, a) = \mathbb{E}_\pi \left[\sum_{t=1}^{H} \gamma^{t-1} F_{a_t} q_t \mid \text{do } q_1 = q, a_1 = a \right]$$

- **[New Idea]: How do we embed policies?**
  The successor feature vector $\phi^\pi(q, a)$ is linear in q so there exists a matrix A^π such $\phi^\pi = A^\pi q$. These *successor feature matrices* satisfy a dynamic programming equation:

$$A^\pi = F_a + \gamma \sum_o A^{\pi(o)} T_{ao}$$

  ($\pi(o)$ = how the policy π continues on step $t+1$ after seeing o)

  Define the *Successor Feature Sets* as:

$$\Phi^{(H)} = \{A^\pi \mid \pi \text{ a policy with horizon } H\}$$

  which satisfies Bellman equations

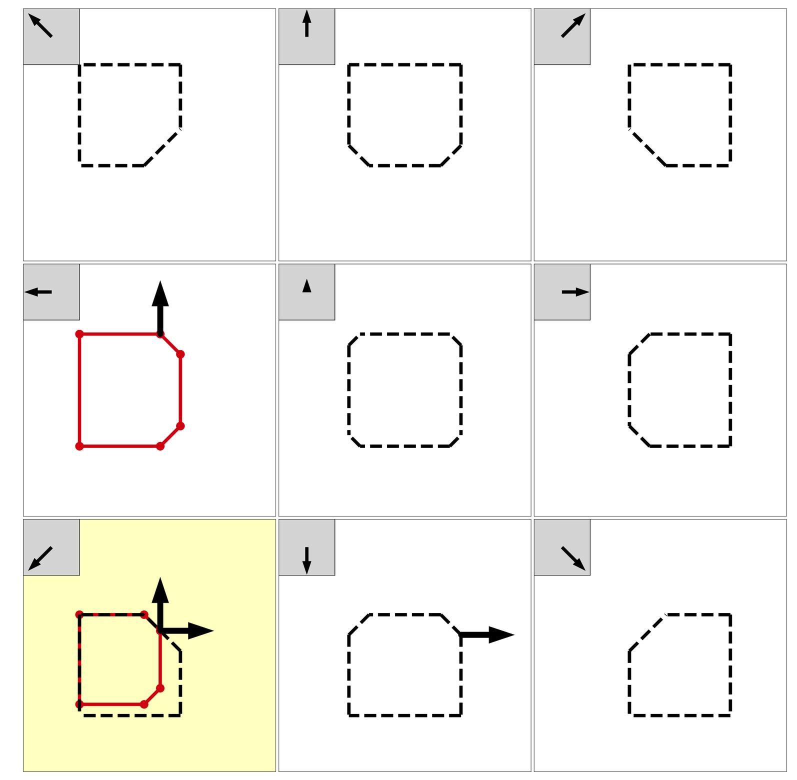
$$\Phi_a^{(H)} = F_a + \gamma \sum_o \Phi^{(H-1)} T_{ao}$$
$$\Phi^{(H)} = \text{conv} \bigcup_a \Phi_a^{(H)}$$

  Corresponding update is a contraction and converges to a unique fixed point.

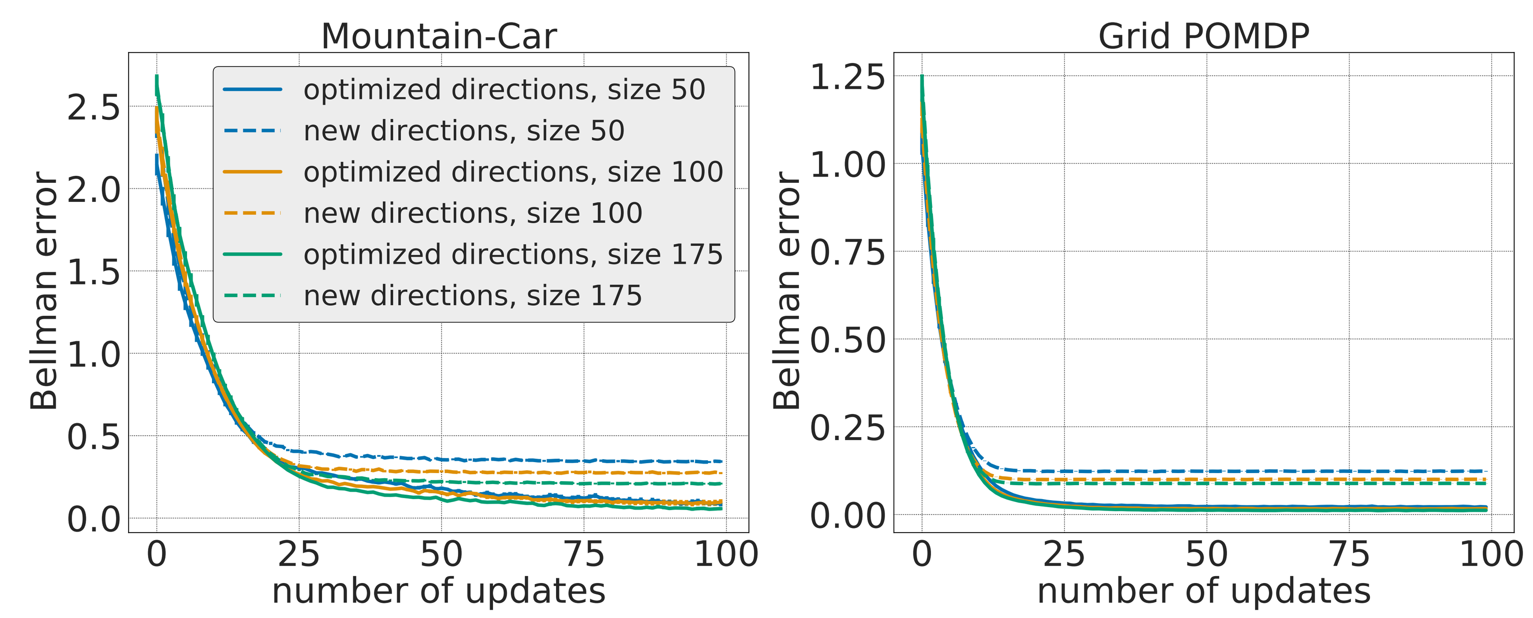
- **Given the above embeddings**, the value of any policy π for any task r starting from any state q is $r^T \pi q$.

Successor feature set example

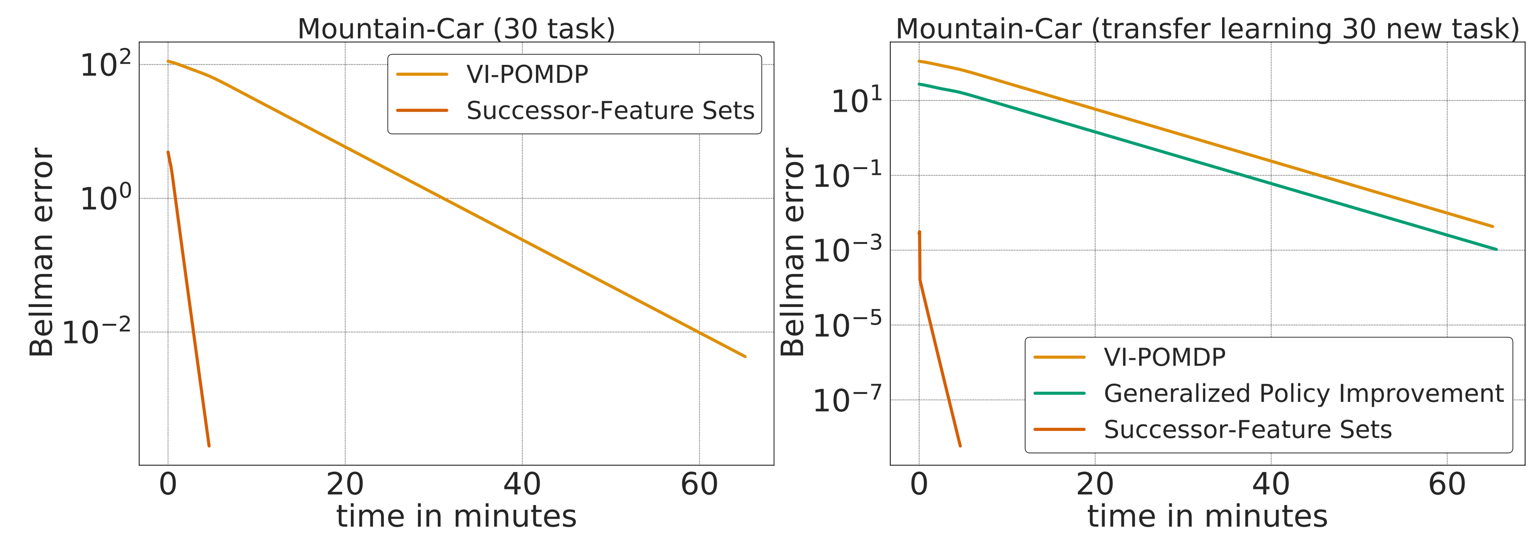
Projections of successor feature sets for 3x3 grid MDP. Red outlines illustrate a step of dynamic programming.



Experiments



We show error separately in directions we have optimized over and in new random directions. ***Left***: The Mountain-car domain. ***Right***: Random 18 × 18 POMDP gridworld where actions and observations are noisy.



Comparing Successor Feature Sets, value iteration and General Policy Improvement. We see that Successor Feature Sets improves over both baselines.