Composable Learning with Sparse Kernel Representations

Ekaterina Tolstaya, Ethan Stump, Alec Koppel and Alejandro Ribeiro
eig@seas.upenn.edu

Penn Engineering | GRASP Laboratory
General Robotics, Automation, Sensing & Perception Lab

Reinforcement Learning

- In Markov Decision problems, the goal is to find the action sequence that maximizes the accumulated rewards at a given start state [Bel54]

$$V(\mathbf{s}, \{\mathbf{a}_t\}_{t=0}^{\infty}) := \mathbb{E}_{\mathbf{s}'}[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}'_t) \mid \mathbf{s}_0 = \mathbf{s}, \{\mathbf{a}_t\}_{t=0}^{\infty}] \quad (1)$$

- Action-value function, the accumulation of rewards given initial $\mathbf{s}, \mathbf{a}$

$$Q(\mathbf{s}, \mathbf{a}, \{\mathbf{a}_t\}_{t=1}^{\infty}) := \mathbb{E}_{\mathbf{s}'}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}'_t) \mid \mathbf{s}_0 = \mathbf{s}, \mathbf{a}_0 = \mathbf{a}, \{\mathbf{a}_t\}_{t=1}^{\infty}\right]$$

- Advantage Function, where $\max_{\mathbf{a}} A(\mathbf{s}, \mathbf{a}) = 0$ [Bai94]

$$Q(\mathbf{s}, \mathbf{a}) = V(\mathbf{s}) + A(\mathbf{s}, \mathbf{a}) \quad (2)$$

- Parameterizing the advantage function as a quadratic function yields computational savings [GLSL16]

$$A(\mathbf{s}, \mathbf{a}) = -\frac{1}{2}(\mathbf{a} - \pi(\mathbf{s}))L^T(\mathbf{s})L(\mathbf{s})(\mathbf{a} - \pi(\mathbf{s})) \quad (3)$$

Model to learn:
- $V(\mathbf{s})$ - value of state $\mathbf{s}$
- $\pi(\mathbf{s})$ - policy at state $\mathbf{s}$
- $L(\mathbf{s})$ - slope at $\mathbf{s}$

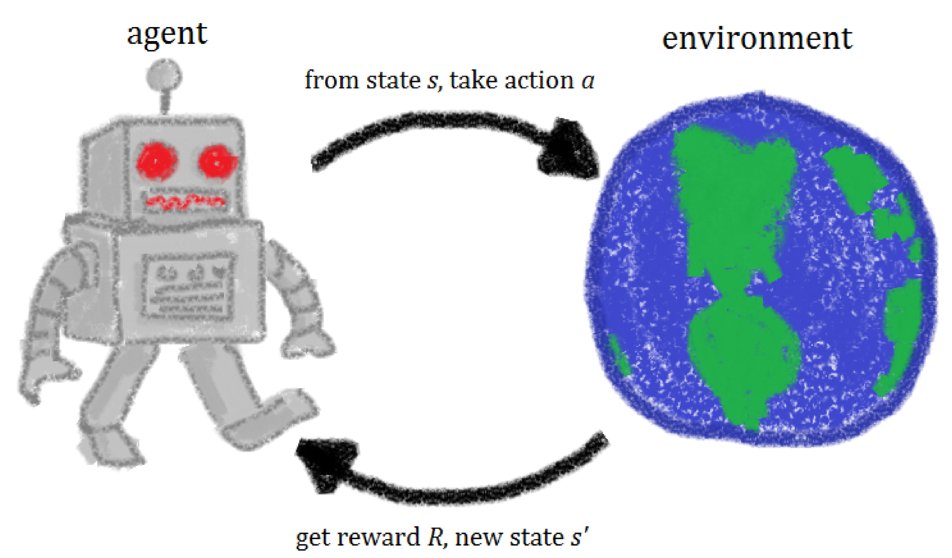

Illustration via Wikimedia

Optimizing the Bellman Error

- Bellman optimality equation [BS04]:

$$Q^*(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{\mathbf{s}'}[r(\mathbf{s}, \mathbf{a}, \mathbf{s}') + \gamma \max_{\mathbf{a}'} Q(\mathbf{s}', \mathbf{a}')] \quad (4)$$

- To find the optimal policy, we seek to satisfy (4) for all state-action pairs, yielding the cost functional:

$$J(V, \pi, L) = \mathbb{E}_{\mathbf{s},\mathbf{a}}(y(\mathbf{s}, \mathbf{a}) - Q(\mathbf{s}, \mathbf{a}))^2, \quad (5)$$

where $y(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{\mathbf{s}'}[r(\mathbf{s}, \mathbf{a}, \mathbf{s}') + \gamma V(\mathbf{s}')]$.
- Finding the Bellman fixed point reduces to the stochastic program:

$$V^*, L^*, \pi^* = \arg \min_{V, \pi, L \in \mathcal{B}(\mathcal{S})} J(V, \pi, L)\ . \quad (6)$$

Reproducing Kernel Hilbert Spaces

- We restrict $\mathcal{B}(\mathcal{S})$ to be a reproducing Kernel Hilbert space (RKHS) $\mathcal{H}$ to which V, π and L belong [KTSR17].
- An RKHS over $\mathcal{S}$ is a Hilbert space is equipped with a reproducing kernel, an inner product-like map $\kappa : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ [NK09, AMP09]:

$$(i)\langle \pi, \kappa(\mathbf{s}, \cdot)\rangle_{\mathcal{H}} = \pi(\mathbf{s}), \qquad (ii)\mathcal{H} = \text{span}\{\kappa(\mathbf{s}, \cdot)\} \quad (7)$$

- A continuous function over a compact set may be approximated uniformly by a function in a RKHS equipped with a universal kernel [MXZ06].
- We use the squared exponential kernel in our experiments:

$$\kappa(\mathbf{s}, \mathbf{s}') = \exp\{-\frac{1}{2}(\mathbf{s} - \mathbf{s}')\Sigma(\mathbf{s} - \mathbf{s}')^T\} \quad (8)$$

This work is supported by grants NSF DGE-1321851 and ARL DCIST CRA W911NF-17-2-0181.

Stochastic Gradient Descent in the RKHS

- **Goal**: Learn V, π and L using samples $(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t)$
- **Solution**: Stochastic semi-gradient descent [SB18] uses the directional derivative of the loss where the target value y_t is fixed:

$$y_t := r_t + \gamma V_t(\mathbf{s}'_t) \quad (9)$$

- Using the Reproducing Property of the RKHS, the optimal V, π and L functions in the RKHS are of the form:

$$V(\mathbf{s}) = \sum_{n=1}^{N} w_{Vn}\kappa(\mathbf{s}_n, \mathbf{s}), \quad \pi(\mathbf{s}) = \sum_{n=1}^{N} \mathbf{w}_{\pi n}\kappa(\mathbf{s}_n, \mathbf{s}), \quad L(\mathbf{s}) = \sum_{n=1}^{N} \mathbf{w}_{Ln}\kappa(\mathbf{s}_n, \mathbf{s})$$

Alg. 1: Q-Learning with Kernel Normalized Advantage Functions

Input: $l_0, \{\alpha_t, \beta_t, \zeta_t, \epsilon_t, \Sigma_t\}_{t=0,1,2\ldots}$

$V_0(\cdot) = 0, \pi_0(\cdot) = 0, L_0(\cdot) = l_0 I, \rho_0(\cdot) = 0$

for $t = 0, 1, 2, \ldots$ **do**

Obtain trajectory $(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t)$ where $\mathbf{a}_t \sim \mathcal{N}(\pi_t(\mathbf{s}_t), \Sigma_t)$

Compute the target value and temporal difference

$y_t = r_t + \gamma V_t(\mathbf{s}'_t), \quad \delta_t = y_t - Q_t(\mathbf{s}_t, \mathbf{a}_t)$

Compute the stochastic estimates of the gradients of the loss

$\hat{\nabla}_V J(Q_t) = -\delta_t \kappa(\mathbf{s}_t, \cdot), \ \hat{\nabla}_\pi J(Q_t) = -\delta_t L(\mathbf{s}_t) L(\mathbf{s}_t)^T (\mathbf{a}_t - \pi_t(\mathbf{s}_t))\kappa(\mathbf{s}_t, \cdot),$

$\hat{\nabla}_L J(Q_t) = \delta_t L(\mathbf{s}_t)^T (\mathbf{a}_t - \pi_t(\mathbf{s}_t))(\mathbf{a}_t - \pi_t(\mathbf{s}_t))^T \kappa(\mathbf{s}_t, \cdot)$

Update V, π, L, ρ:

$V_{t+1} = V_t - \alpha_t \hat{\nabla}_V J(Q_t), \quad \pi_{t+1} = \pi_t - \beta_t \hat{\nabla}_\pi J(Q_t),$

$L_{t+1} = L_t - \zeta_t \hat{\nabla}_L J(Q_t), \quad \rho_{t+1} = \rho_t + \kappa(\mathbf{s}_t)$

Obtain greedy compression of V_{t+1}, π_{t+1}, L_{t+1}, ρ_{t+1} via KOMP

end for

return V,π,L

Model Composition

- Our approach is motivated by multi-agent systems with infrequent communication.
- Models learned by separate systems are directly composed as a single model that combines the strengths of each.

Given: N models π_i each trained on $D_i = \{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t)\}_{t=1,\ldots,N_i}$

Goal: Fit Π, which performs as well as π trained on $\bigcup_{i=1}^{N} D_i$

- Interpolate among π_i to get Π by setting $\Pi(\mathbf{s}) = \pi_i(\mathbf{s}), \forall \mathbf{s}$

Challenge: Policies π_i can disagree for $\mathbf{s} \in \mathcal{S}$
- While training π_i, count the number of training samples around $\mathbf{s}$ to evaluate the support of the model at $\mathbf{s}$:

$$\rho_{i,t+1}(\mathbf{s}) = \rho_{i,t}(\mathbf{s}) + \kappa(\mathbf{s}_t, \mathbf{s}) \quad (10)$$

- For every $\mathbf{s} \in \mathcal{S}$, choose the policy with the highest density of samples, $\rho_i(\mathbf{s})$

Alg. 2: Composition with Conflict Resolution

Input: $\{\pi_i(\mathbf{s}) = \sum_j^{M_i} w_{ij}\kappa(\mathbf{s}, \mathbf{s}_{ij}),$

$\rho_i(\mathbf{s}) = \sum_j^{M_i} v_{ij}\kappa(\mathbf{s}, \mathbf{s}_{ij})\}_{i=1,2\ldots,N}, \epsilon$

Initialize $\Pi(\cdot) = 0$, append centers $D = [\mathbf{s}_{11}, \ldots, \mathbf{s}_{ij}, \ldots]$

for each $\mathbf{s}_{ij} \in D$ chosen uniformly at random **do**

if $\rho_i(\mathbf{s}_{ij}) > \max_{k \neq i} \rho_k(\mathbf{s}_{ij})$ **then**

$\Pi = \Pi(\cdot) + (\pi_i(\mathbf{s}_{ij}) - \Pi(\mathbf{s}_{ij}))\kappa(\mathbf{s}_{ij}, \cdot)$

end if

end for

Obtain compression of π using KOMP with ϵ

return f

Obstacle Avoidance Tasks

- **State**: 5 range readings from LIDAR at at an angular interval of 34° with a field of view of 170°
- **Action**: angular velocity of the Scarab robot, $a \in [-0.3, 0.3]$ rad/s
- **Reward**:

$$r(s) = \begin{cases} -200, & \text{if collision} \\ +1, & \text{otherwise} \end{cases}$$

- Sensor readings are received and controls are issued at 10 Hz
- Constant forward velocity of 0.15 m/s

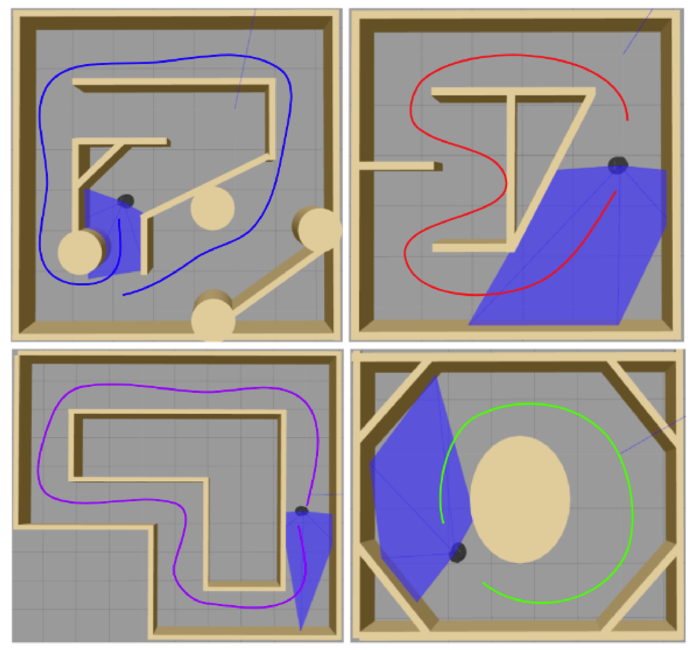

Figure: Four environments were simulated using Gazebo.

Simulation and Experiment Results

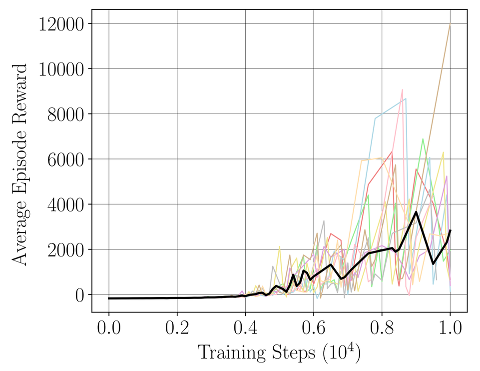

Figure: Reward averaged over 10 trials in the Round environment (black)

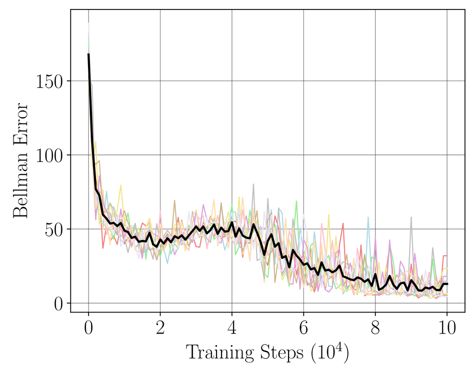

Figure: Training loss averaged over 10 trials in the Round environment (black)

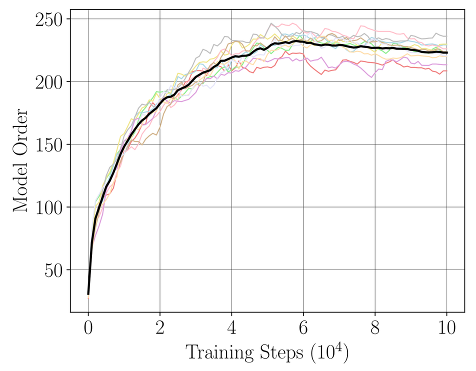

Figure: Model order averaged over 10 trials in the Round environment (black)

Policies / Reward	Round	Maze	Circuit 2	Circuit 1
1 - Round	**1000**	-11663	-608	-608
2 - Maze	**1000**	**1000**	-5	-407
3 - Circuit 2	**1000**	-11663	**1000**	196
4 - Circuit 1	**1000**	-11462	-407	**1000**
1 / 2	**1000**	**1000**	-5	-206
1 / 3	**1000**	-11663	799	-206
1 / 4	**1000**	-11261	-206	799
2 / 3	**1000**	**1000**	**1000**	-5
2 / 4	**1000**	**1000**	-5	799
3 / 4	**1000**	-11462	397	397
1 / 2 / 3	**1000**	**1000**	799	196
1 / 2 / 4	**1000**	**1000**	-5	**1000**
1 / 3 / 4	**1000**	-11663	397	799
2 / 3 / 4	**1000**	**1000**	799	-206
1 / 2 / 3 / 4	**1000**	**1000**	**1000**	598

Table: Composability results

Cross-validation was performed in simulation on all compositions of the 4 policies. We then validate our approach by testing these policies on a real robot. The policy trained only on the Round environment experienced 3 crashes over 1,000 testing steps. The composite 1/2/3/4 policy received a reward of 1,000 with no crashes.

Conclusions

Contributions:
- Stochastic gradient descent algorithm for RL in RKHS
- Formulation of the problem of composable learning
- Heuristic for policy composition

Shortcomings:
- Need to develop a theoretically justified metric of risk or uncertainty of the learned policy
- Using kernel methods in large state spaces is impractical without dimensionality reduction (Autoencoders, Sparse GPs using Pseudo-inputs)