Lifelong Inverse Reinforcement Learning


Jorge Mendez
University of Pennsylvania


Shashank Shivkumar
University of Pennsylvania


Eric Eaton
University of Pennsylvania

Penn Engineering

GRASP Laboratory

Summary

We introduce the novel problem of lifelong imitation learning and develop the first algorithm for lifelong inverse reinforcement learning (IRL).

Capabilities of our approach:

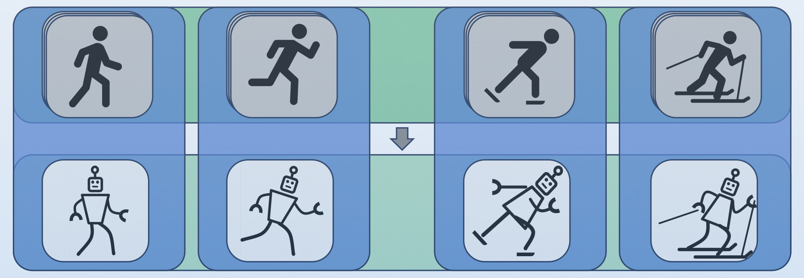
- Learns multiple tasks consecutively
- Transfers knowledge to accelerate learning of new tasks
- Supports a variety of base learning algorithms
- Has lower computational cost than current multi-task learning algorithms
- Supports varying feature spaces across tasks

We demonstrate the effectiveness of ELIRL in lifelong learning settings.

Introduction

Goal: Develop intelligent systems that

- Rapidly learn to imitate demonstrated tasks
- Re-use knowledge relevant to different tasks
- Learn to imitate varied tasks

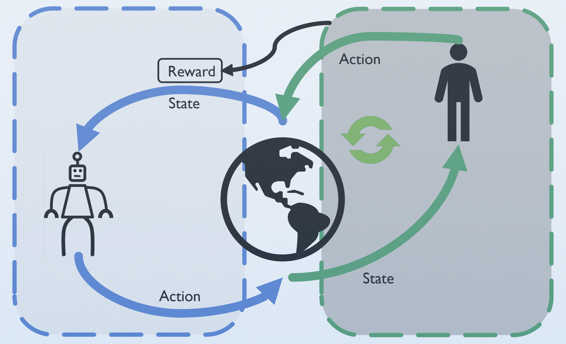


We frame lifelong learning from demonstration as online multi-task learning of inverse reinforcement learning tasks.

Our lifelong learning algorithm wraps around existing IRL methods and performs lifelong function approximation on the reward functions.

Background: Inverse Reinforcement Learning

Given an environment $\mathrm{MDP}\backslash\mathbf{r} : \langle \mathcal{S}, \mathcal{A}, T, \gamma \rangle$ and a set of expert trajectories $\mathcal{Z} = \{\zeta_1, \ldots, \zeta_n\}$, with $\zeta_j = [\mathbf{s}_{0:H}, \mathbf{a}_{0:H}]$, output a reward $\mathbf{r}$ that explains the expert's behavior.



MaxEnt IRL [Ziebart et al., AAAI '08]

Assumptions:

- Linear reward: $\mathbf{r}_{s_i} = \mathbf{r}(\mathbf{x}_{s_i}, \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x}_{s_i}$
- Maximum entropy trajectories: $P(\zeta_j \mid \boldsymbol{\theta}) = \dfrac{\exp(\boldsymbol{\theta}^\top \mathbf{x}_{\zeta_j})}{Z(\boldsymbol{\theta})}$

Goal: Maximize $\log P(\mathcal{Z} \mid \boldsymbol{\theta})$

Lifelong Inverse Reinforcement Learning Problem

Given a sequence of tasks $\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(N_{max})}$, each of them an $\mathrm{MDP}\backslash\mathbf{r}$: $\mathcal{T}^{(t)} = \left\langle \mathcal{S}^{(t)}, \mathcal{A}^{(t)}, T^{(t)}, \gamma^{(t)} \right\rangle$

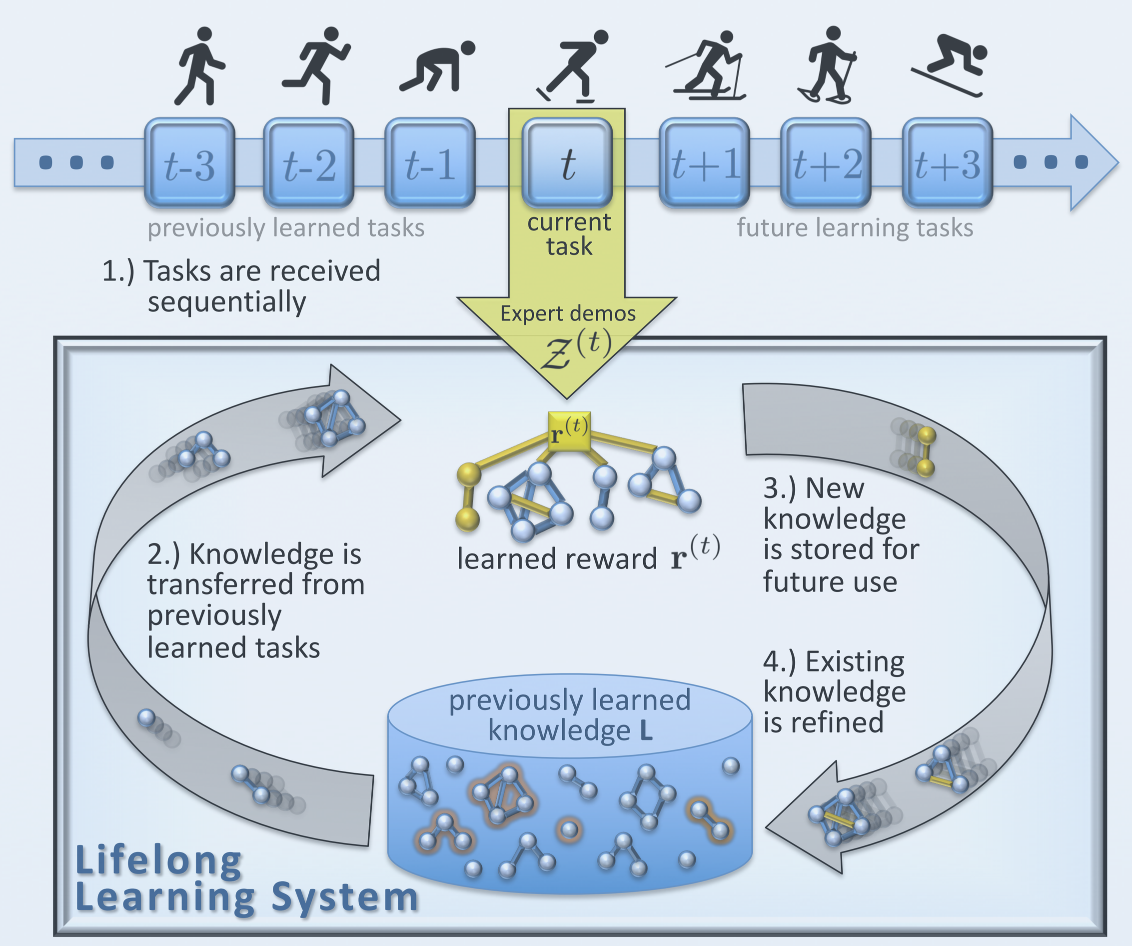
Goal: Estimate the set of all reward functions $\mathcal{R} = \left\{ \mathbf{r}\left(\boldsymbol{\theta}^{(1)}\right), \ldots, \mathbf{r}\left(\boldsymbol{\theta}^{(N_{max})}\right) \right\}$

How? Upon observing the N-th task, solve:

$$\max_{\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(N)}} P(\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(N)}) \prod_{t=1}^{N} \left(\prod_{i=1}^{n_t} P(\zeta_j \mid \mathbf{r}^{(t)}) \right)^{\frac{1}{n_t}}$$

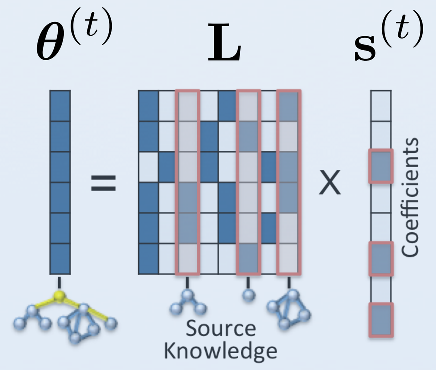
Reward prior $P\left(\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(N)}\right)$ encourages tasks to share structure.

Efficient Lifelong Inverse Reinforcement Learning



The task parameters are factorized into a shared basis and task-specific coefficients

$$\underset{d \times 1}{\boldsymbol{\theta}^{(t)}} = \underset{d \times k \;\; k \times 1}{\mathbf{L}\mathbf{s}^{(t)}} \ .$$



A prior is chosen to encourage knowledge transfer

$$P\left(\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(N)}\right) \propto \exp\left(-N\lambda \|\mathbf{L}\|_{\mathsf{F}}^2\right) \prod_{t=1}^{N} \exp\left(-\mu \|\mathbf{s}^{(t)}\|_1\right) \ .$$

The multi-task objective encourages reward models to share structure

$$\min_{\mathbf{L}} \underbrace{\frac{1}{N}}_{\text{\#tasks seen so far}} \sum_{t=1}^{N} \min_{\mathbf{s}^{(t)}} \left\{ \underbrace{-\frac{1}{n_t} \sum_{\zeta_j^{(t)} \in \mathcal{Z}^{(t)}} \log P\left(\zeta_j^{(t)} \mid \mathbf{L}\mathbf{s}^{(t)}, T^{(t)}\right)}_{\text{IRL log-likelihood}} + \underbrace{\mu \|\mathbf{s}^{(t)}\|_1}_{\text{sparsity}} \right\} + \underbrace{\lambda \|\mathbf{L}\|_{\mathsf{F}}^2}_{\text{complexity}} \ .$$

We can approximate this objective as a sparse coding problem, which we can solve efficiently online [Ruvolo and Eaton, ICML '13] as

$$\min_{\mathbf{L}} \frac{1}{N} \sum_{t=1}^{N} \min_{\mathbf{s}^{(t)}} \left\{ \left(\boldsymbol{\alpha}^{(t)} - \mathbf{L}\mathbf{s}^{(t)}\right)^\top \mathbf{H}^{(t)} \left(\boldsymbol{\alpha}^{(t)} - \mathbf{L}\mathbf{s}^{(t)}\right) + \mu \|\mathbf{s}^{(t)}\|_1 \right\} + \lambda \|\mathbf{L}\|_{\mathsf{F}}^2 \ ,$$

where $\boldsymbol{\alpha}^{(t)} = \arg\min_{\boldsymbol{\alpha}} - \sum_{\zeta_j^{(t)} \in \mathcal{Z}^{(t)}} \log P\left(\zeta_j^{(t)} \mid \boldsymbol{\alpha}, T^{(t)}\right)$,

$$\mathbf{H}^{(t)} = \frac{1}{n_t} \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \mathcal{L}\left(\mathbf{r}\left(\mathbf{L}\mathbf{s}^{(t)}\right), \mathcal{Z}^{(t)}\right) = \left(-\sum_{\tilde{\zeta} \in \mathcal{Z}_{MDP}} \mathbf{x}_{\tilde{\zeta}} P(\tilde{\zeta} \mid \boldsymbol{\theta})\right) \left(\sum_{\tilde{\zeta} \in \mathcal{Z}_{MDP}} \mathbf{x}_{\tilde{\zeta}}^\top P(\tilde{\zeta} \mid \boldsymbol{\theta})\right) + \sum_{\tilde{\zeta} \in \mathcal{Z}_{MDP}} \mathbf{x}_{\tilde{\zeta}} \mathbf{x}_{\tilde{\zeta}}^\top P(\tilde{\zeta} \mid \boldsymbol{\theta}) \ .$$

Upon observing task *t*, solve

$$\mathbf{s}^{(t)} \leftarrow \arg\min_{\mathbf{s}} \ell\left(\mathbf{L}_N, \mathbf{s}, \boldsymbol{\alpha}^{(t)}, \mathbf{H}^{(t)}\right) \qquad \mathbf{L}_{N+1} \leftarrow \arg\min_{\mathbf{L}} \lambda \|\mathbf{L}\|_{\mathsf{F}}^2 + \frac{1}{N} \sum_{t=1}^{N} \ell\left(\mathbf{L}, \mathbf{s}^{(t)}, \boldsymbol{\alpha}^{(t)}, \mathbf{H}^{(t)}\right) \ ,$$

where $\ell(\mathbf{L}, \mathbf{s}, \boldsymbol{\alpha}, \mathbf{H}) = \mu \|\mathbf{s}\|_1 + (\boldsymbol{\alpha} - \mathbf{L}\mathbf{s})^\top \mathbf{H} (\boldsymbol{\alpha} - \mathbf{L}\mathbf{s})$.

ELIRL Algorithm – Training

Given a new task t,

1. Observe demonstrated trajectories $\mathcal{Z}^{(t)}$
2. Use single-task MaxEnt IRL to find $\boldsymbol{\alpha}^{(t)}$ and $\mathbf{H}^{(t)}$
3. $\mathbf{s}^{(t)} \leftarrow \arg\min_{\mathbf{s}} (\boldsymbol{\alpha}^{(t)} - \mathbf{L}\mathbf{s})^\top \mathbf{H}^{(t)} (\boldsymbol{\alpha}^{(t)} - \mathbf{L}\mathbf{s}) + \mu \|\mathbf{s}\|_1$
4. $\mathbf{L} \leftarrow \mathrm{update}L(\mathbf{L}, \mathbf{s}^{(t)}, \boldsymbol{\alpha}^{(t)}, \mathbf{H}^{(t)}, \lambda)$

Per-task Computational Complexity:

$$\mathcal{O}(\underbrace{i\xi(d, |\mathcal{A}|, |\mathcal{S}|)}_{\text{MaxEnt}} + \underbrace{MH + Md^2}_{\text{Hessian}} + \underbrace{k^2 d^3}_{\text{ELIRL overhead}})$$

ELIRL Algorithm – Testing

Given a previously encountered task t,

1. [Optional*] Re-optimize the task-specific coefficients
   $\mathbf{s}^{(t)} \leftarrow \arg\min_{\mathbf{s}} (\boldsymbol{\alpha}^{(t)} - \mathbf{L}_{new}\mathbf{s})^\top \mathbf{H}^{(t)} (\boldsymbol{\alpha}^{(t)} - \mathbf{L}_{new}\mathbf{s}) + \mu \|\mathbf{s}\|_1$
2. Approximate the reward parameters $\boldsymbol{\theta}^{(t)} \leftarrow \mathbf{L}_{new}\mathbf{s}^{(t)}$
3. Use $\boldsymbol{\theta}^{(t)}$ as the reward function of $\mathcal{T}^{(t)}$ in standard reinforcement learning

*This optional step allows earlier tasks to benefit from the newest knowledge.

Computational Complexity of the Re-optimization:

Constructing and solving an instance of LASSO [Ruvolo and Eaton, ICML '13]

$$\mathcal{O}(d^3 + kd^2 + dk^2)$$

Theoretical Convergence Guarantees of ELIRL

ELIRL inherits guaranteed convergence from ELLA [Ruvolo and Eaton, ICML '13]

- $\mathbf{L}$ converges to a local optimum of the approximate cost function
- The approximation is good if $\boldsymbol{\theta}^{(t)} = \mathbf{L}\mathbf{s}^{(t)}$ is close to $\boldsymbol{\alpha}^{(t)}$
- This holds if the factored representation sufficiently captures task relatedness

Extension of ELIRL for Cross-Domain Transfer

ELIRL supports tasks with different feature spaces.

- We can assume tasks come from a set of groups $\left\{\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(G_{max})}\right\}$
- Tasks from within a group $\mathcal{G}^{(g)}$ share the same feature space
- ELIRL learns projection matrices $\boldsymbol{\Psi}^{(g)}$ that map the knowledge in $\mathbf{L}$ to $\mathcal{G}^{(g)}$
- The parameters are factored as $\boldsymbol{\theta}^{(t)} = \boldsymbol{\Psi}^{(g)}\mathbf{L}\mathbf{s}^{(t)}$
- A complexity term $\mu_2 \|\boldsymbol{\Psi}^{(g)}\|_{\mathsf{F}}^2$ is added to the cost function for each group
- Optimization follows the same online procedure

Extension of ELIRL for Continuous State-Action Spaces

ELIRL can easily be adapted to handle base learners other than MaxEnt IRL.

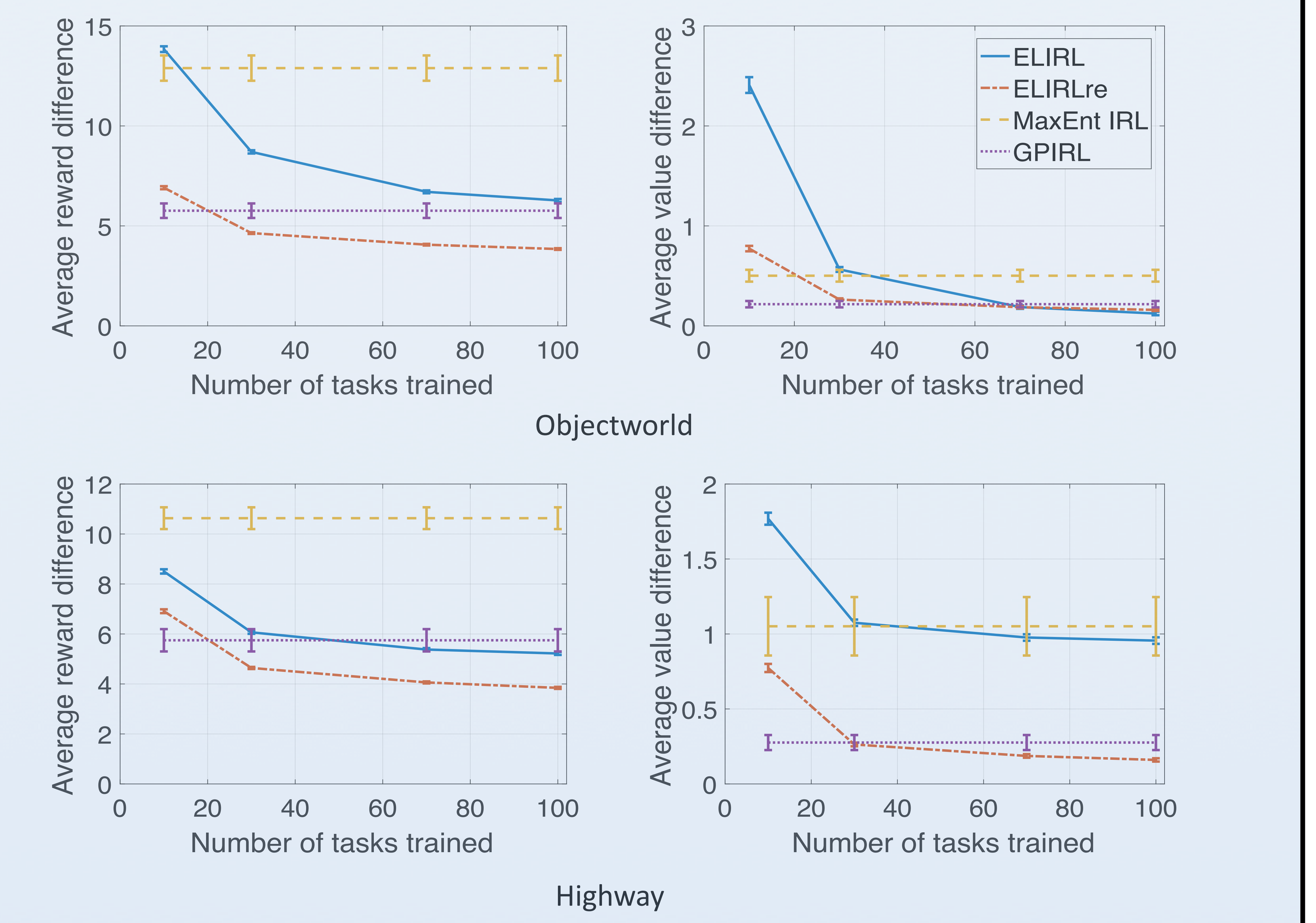
To handle continuous environments, we use AME IRL [Levine and Koltun, ICML '12]

- AME IRL approximates the MaxEnt log-likelihood in infinite state-action spaces
- Using it as the base learners requires only computing the Hessian
- ELIRL then wraps around AME IRL to enable it to operate in a lifelong setting

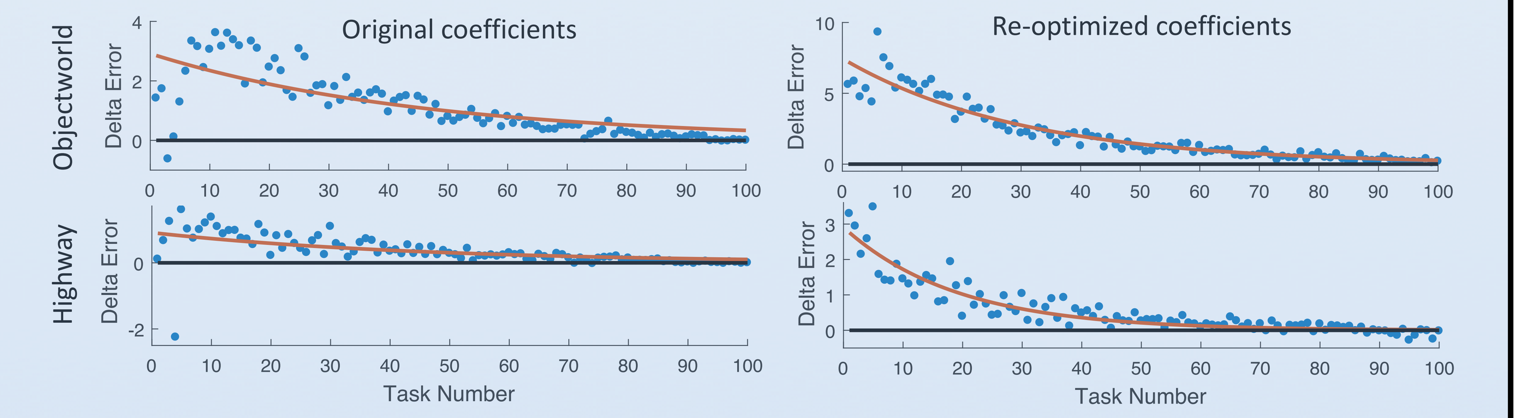
Experimental Results

Objectworld: A 32x32 grid with colored objects. Each color has an associated reward on its surrounding 5x5 grid that varies with each task.
Highway: A 3-lane highway with 4 possible speeds. Each driver prefers a particular speed and lane, with different associated weights for each driver.

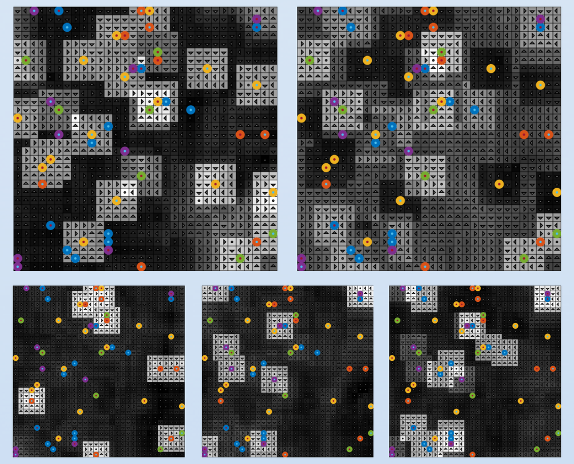
ELIRL improves reward as it learns more tasks → Improved policy performance!



ELIRL transfers knowledge from new tasks to all previous tasks without retraining

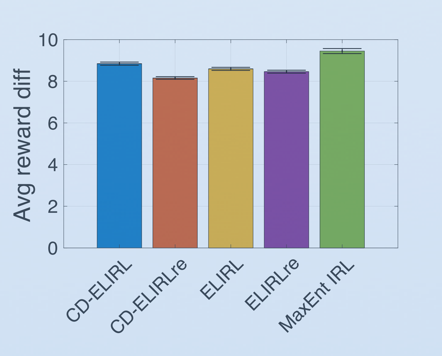


ELIRL learns to focus on specific colors for Objectworld

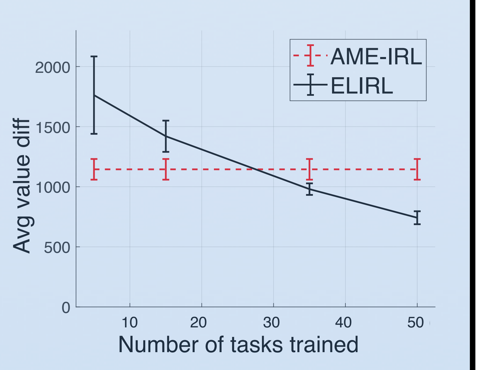


Each figure visualizes the Objectworld reward function given by one column of the learned $\mathbf{L}$ matrix

Cross-domain transfer



Continuous spaces



Acknowledgement: This research was partly supported by AFRL grant #FA8750-16-1-0109 and DARPA agreement #FA8750-18-2-0117.