Contrastive Behavioral Similarity Embeddings for Generalization in Reinforcement Learning

Elliot Hill

Duke University

March, 2024

ChatGPT4 can convert images to LaTeX

You

$$\ell_\theta(x, y; \mathcal{X}') = -\log \frac{\exp(\lambda s_\theta(x, y))}{\exp(\lambda s_\theta(x, y)) + \sum_{x' \in \mathcal{X}' \setminus \{x\}} \exp(\lambda s_\theta(x', y))}$$

Can you give me the latex code for this equation?

ChatGPT

Certainly, the LaTeX code for the equation you've provided would look like this:

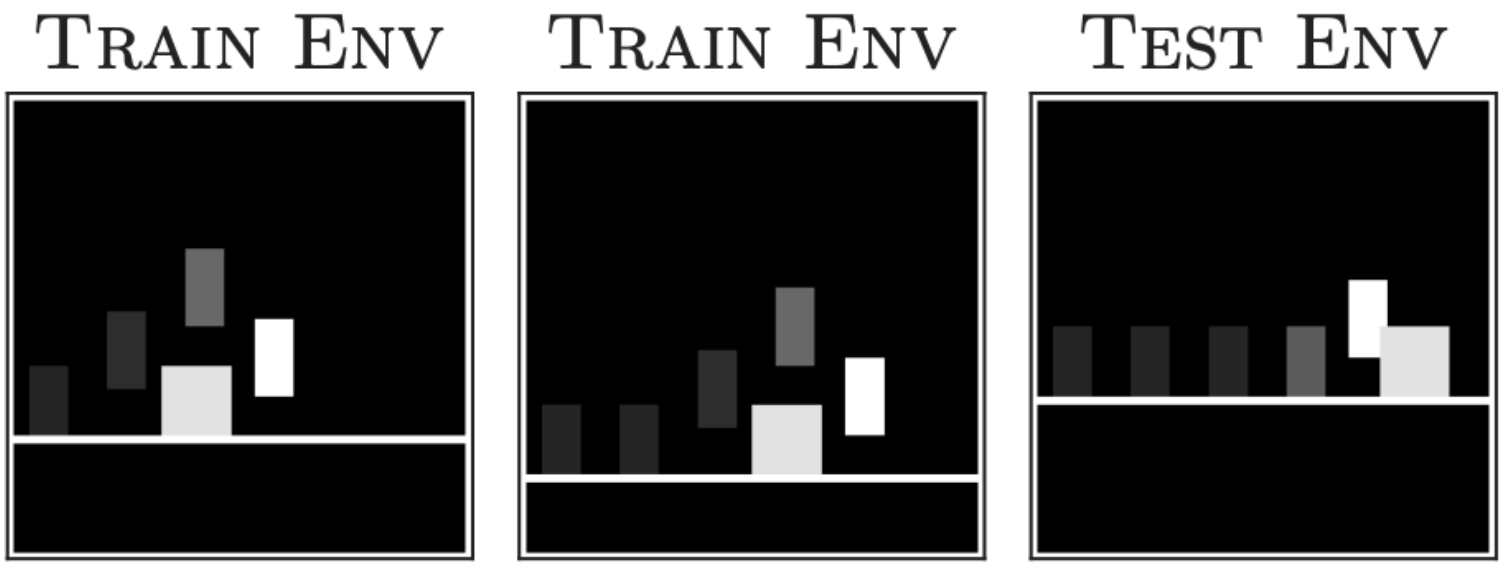
latex Copy code

```
\ell_{\theta}(x, y; x') = - \log \frac{\exp(\lambda s_{\theta}(x, y))}{\exp(\lambda s
```

Incorporating similarity into reinforcement learning methods helps them generalize to unseen environments

- Reinforcement learning methods trained on few environments rarely generalize to unseen environments
- To improve generalization, they incorporate the sequential structure of reinforcement learning into the representation learning process
- Introduce policy similarity metric (PSM) for measuring behavioral similarity between states
- Present a contrastive representation learning procedure to embed any state similarity metric to obtain policy similarity embeddings (PSEs)
- PSEs improve generalization on diverse benchmarks

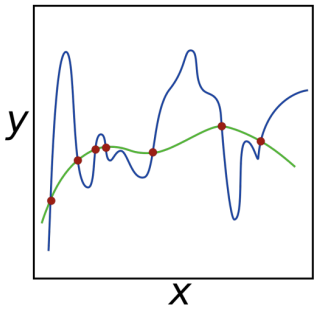
RL agents tend to have poor performance on environments different from those they are trained on



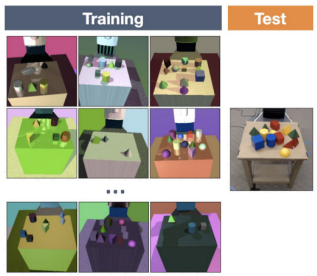
The agent fails when the blocks are moved or the height changes

Prior work on generalization ignores the sequential nature of reinforcement learning

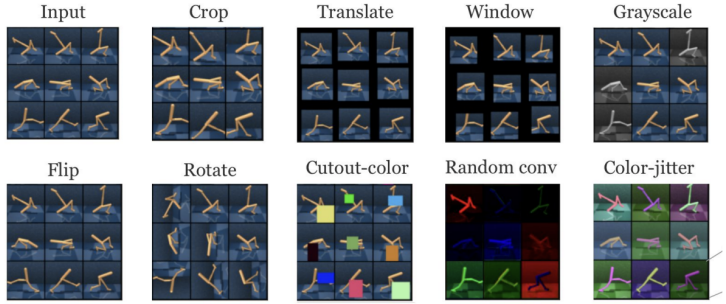
Adapted from supervised learning, *e.g.* :



Regularization
($\ell 2$-reg., Dropout,
Noise Injection)

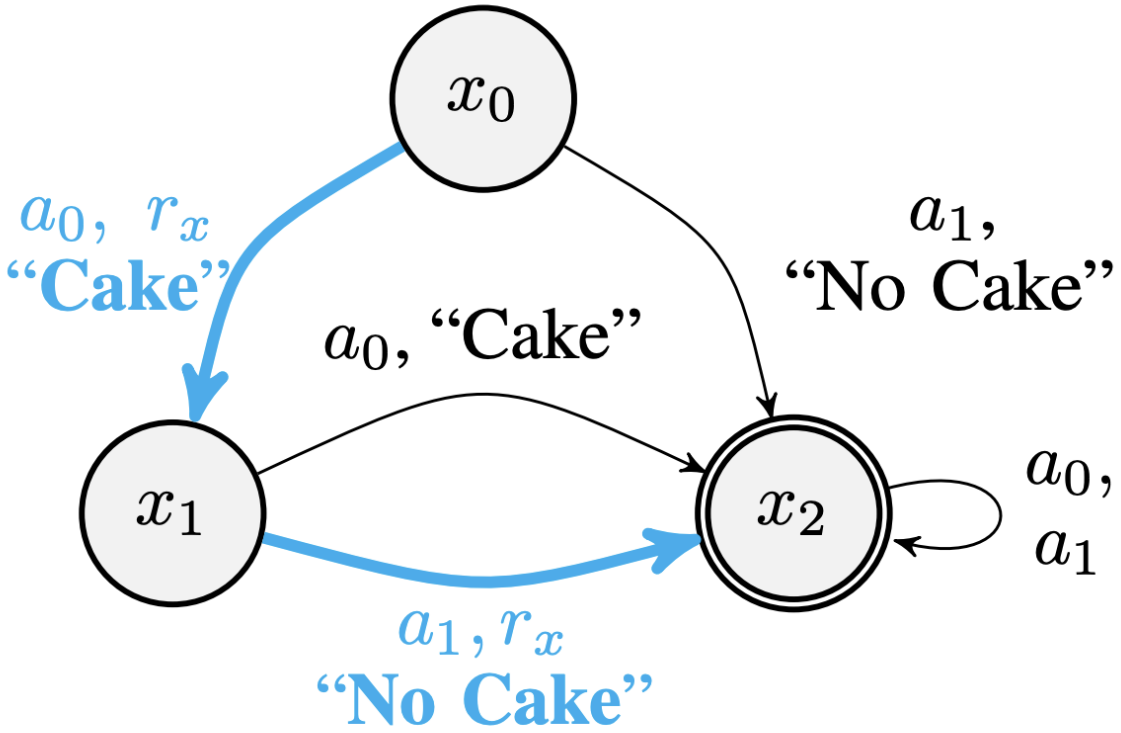


Domain
Randomization



Data Augmentation
(RandConv, RAD, DrQ ..)

A Markov decision process (MDP) is a framework for modeling stochastic sequential decision problems



Notation

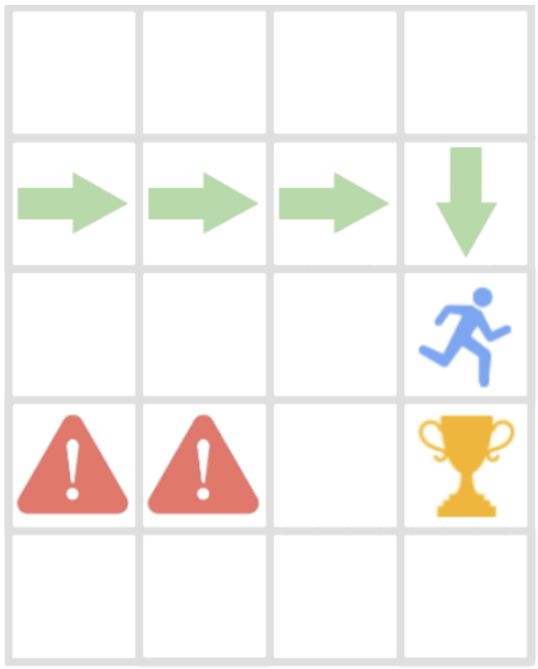
The Markov decision process (MDP) is $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$ where

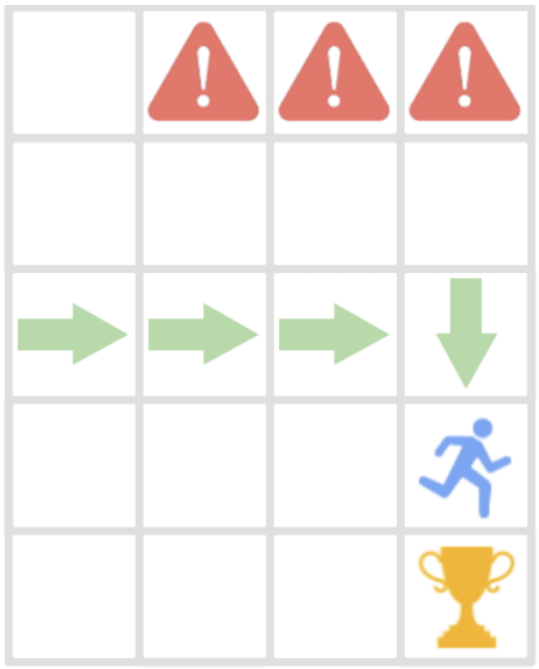
- $\mathcal{X}$ is a state space
- $\mathcal{A}$ is an action space
- R is a reward function
- P is the transition dynamics
- $\gamma \in [0, 1)$ is a discount factor

A policy $\pi(\cdot|x)$ maps states $x \in \mathcal{X}$ to distributions over actions.

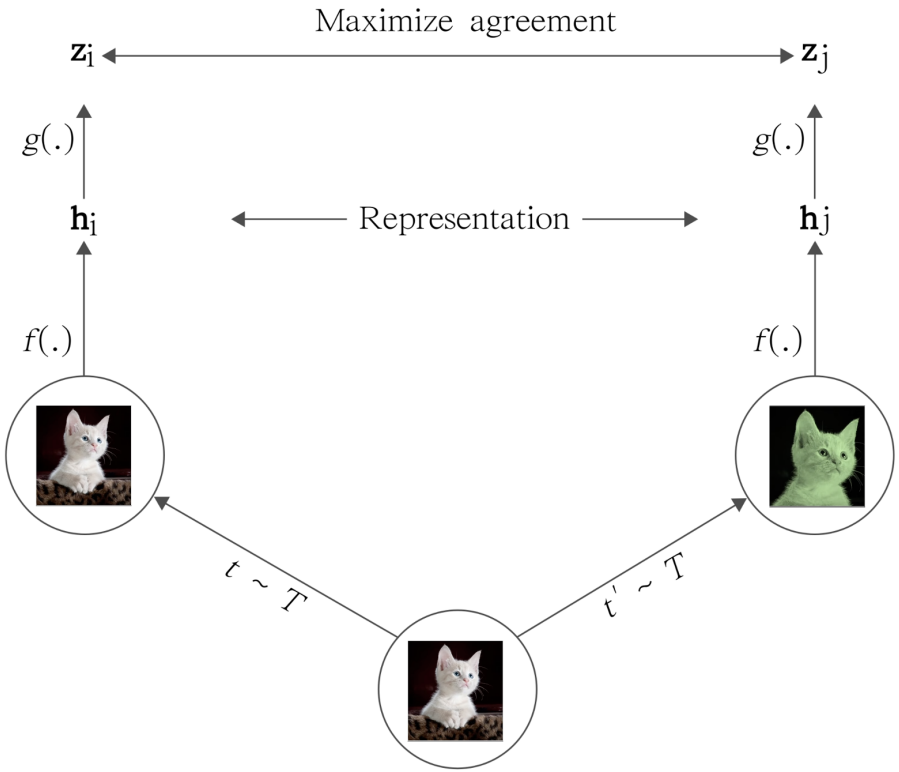
In RL, the goal is to find an optimal policy π^* that maximizes the cumulative expected return $\mathbb{E}_{a_t \sim \pi(\cdot|x_t)} \left[\sum_t \gamma^t R(x_t, a_t)\right]$

Goal: learn representations that encode "behavioral similarity" across states





Contrastive learning pulls positives together and pushes negatives apart



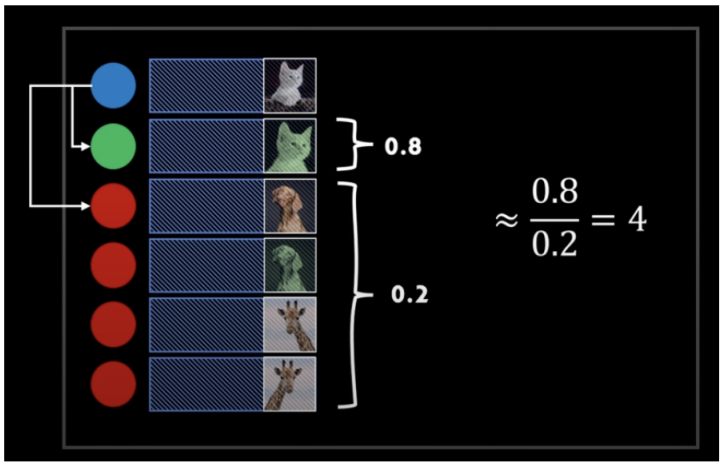
T = data augmentation, f = encoder, h = representation
g = projection head, z = projection

Contrastive learning pulls positives together and pushes negatives apart

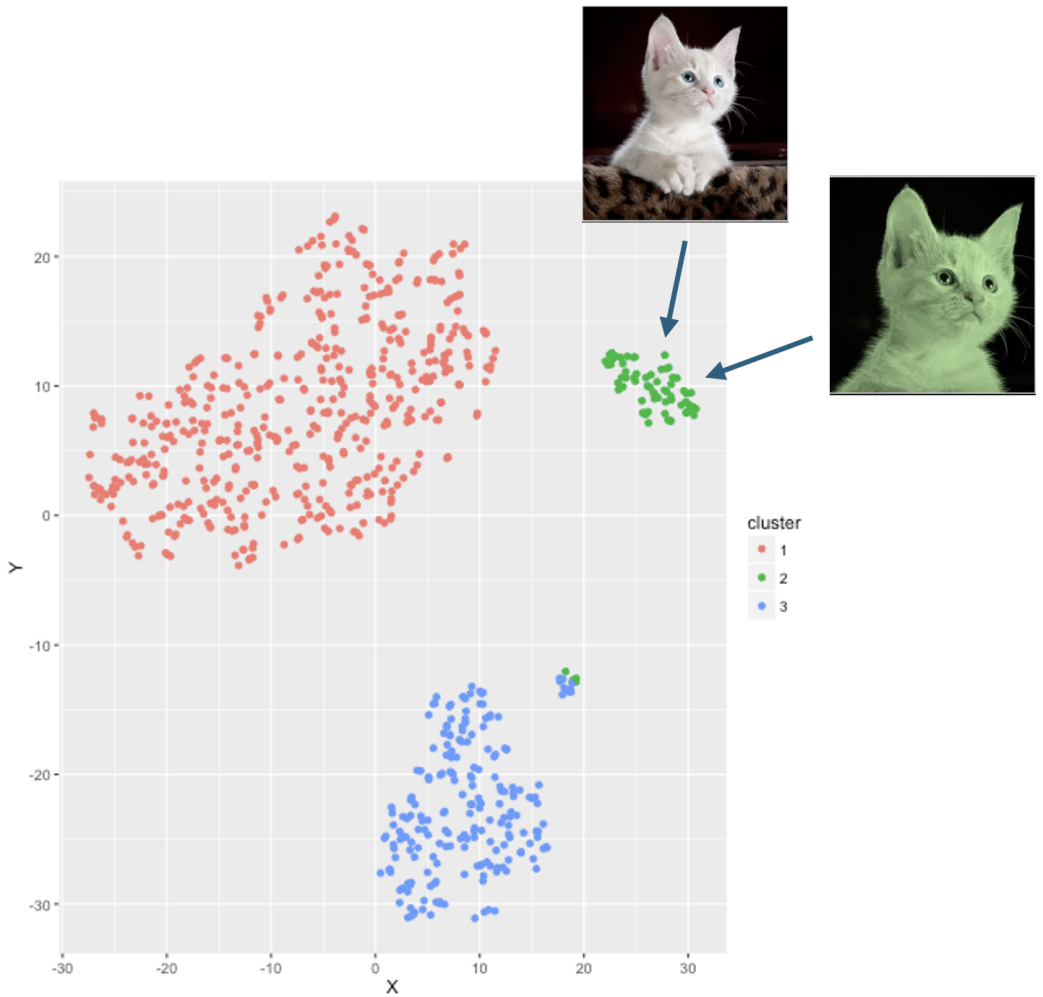
$$\ell_\theta(x, y; \mathcal{X}') = -\log \frac{\exp(\lambda s_\theta(x, y))}{\exp(\lambda s_\theta(x, y)) + \sum_{x' \in \mathcal{X}' \setminus \{x\}} \exp(\lambda s_\theta(x', y))}$$

All other images in the batch

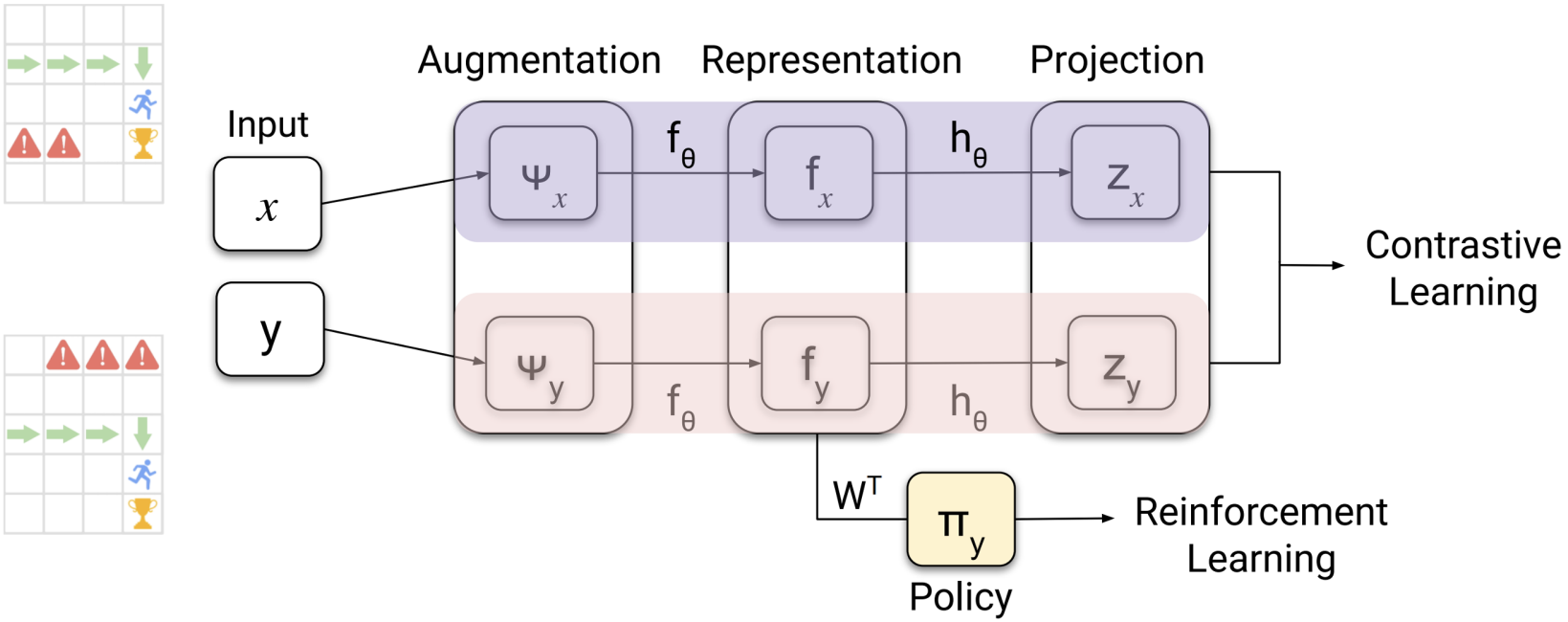
$$s_\theta(x, y) = \text{sim}(z_\theta(x), z_\theta(y)), \text{ where } \text{sim}(u, v) = \frac{u^T v}{\|u\| \|v\|}$$



Contrastive learning results in representation clusters



Contrastive metric embeddings (CME)



The policy π_θ is an affine function of the representation: $\pi_\theta(\cdot|y) = W^T f_y + b$, where W and b are learned weights and biases. The network is trained jointly using a reinforcement learning loss and the contrastive loss

They introduce a "soft" SimCLR contrastive loss to learn contrastive metric embeddings (CMEs)

Given a positive state pair $(\tilde{x}_y, y)$, the set $\mathcal{X}'$, and similarity measure Γ, the loss is given by

$$\ell_\theta(\tilde{x}_y, y; X') = -\log \frac{\Gamma(\tilde{x}_y,y)\exp(\lambda s_\theta(\tilde{x}_y,y))}{\Gamma(\tilde{x}_y,y)\exp(\lambda s_\theta(\tilde{x}_y,y))+\sum_{x'\in X'\setminus\{\tilde{x}_y\}}(1-\Gamma(x',y))\exp(\lambda s_\theta(x',y))}$$

where $\Gamma(x, y) = exp(-d(x, y)/\beta)$ is a Gaussian kernel (with scale parameter β) that convert the distance to a similarity bounded in $[0, 1]$

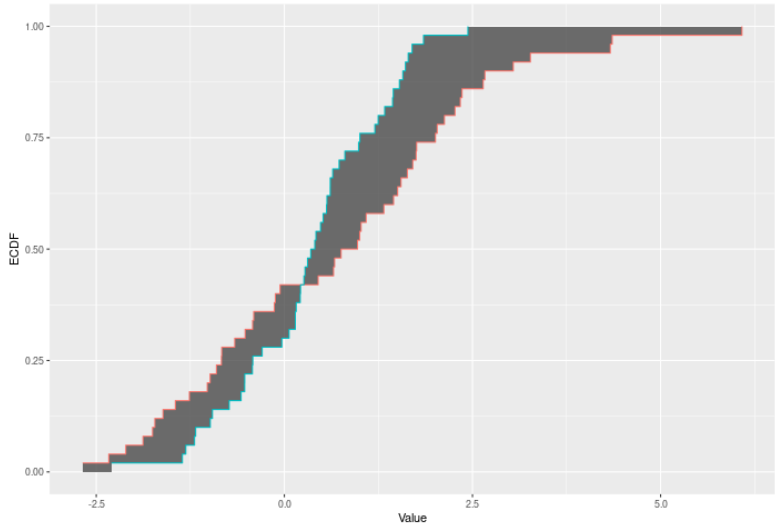
Policy similarity metric (PSM) is inspired by bisimulation

- Policy similarity metric builds on the concept of π-bisimulation [2]
- States are bisimular if they have similar expected rewards/dynamics
- Under the π-bisimulation metric, the distance between two states, x and y, is defined in terms of the difference between the expected rewards obtained when following policy π

$$d_\pi(x, y) = |R^\pi(x) - R^\pi(y)| + \gamma \mathcal{W}_1(d_\pi)(P^\pi(\cdot|x), P^\pi(\cdot|y)), \quad x, y \in \mathcal{S}.$$

where $\mathcal{W}$ is the Wasserstein distance

Wasserstein distance (earth mover distance) is the cost of converting one probability distribution into another



Policy similarity metric (PSM): states are close when the optimal policies in those states and future states are similar

π-bisimulation

$$d_\pi(x, y) = |R^\pi(x) - R^\pi(y)| + \gamma \mathcal{W}_1(d_\pi)(P^\pi(\cdot|x), P^\pi(\cdot|y)).$$

Policy similarity metric

$$d^*(x, y) = \underbrace{\text{DIST}(\pi^*(x), \pi^*(y))}_{(A)} + \gamma \underbrace{W_1(d^*)(P^{\pi^*}(\cdot|x), P^{\pi^*}(\cdot|y))}_{(B)}.$$

A) DIST captures the difference in local optimal behavior

B) $\mathcal{W}_1$ captures long-term optimal behavior difference

They use PSM to compare states across environments

$$d^*(x, y) = \text{DIST}(\pi^*_{\mathcal{X}}(x), \pi^*_{\mathcal{Y}}(y)) + \gamma W_1(d^*)(P^{\pi^*}_{\mathcal{X}}(\cdot|x), P^{\pi^*}_{\mathcal{Y}}(\cdot|y)).$$

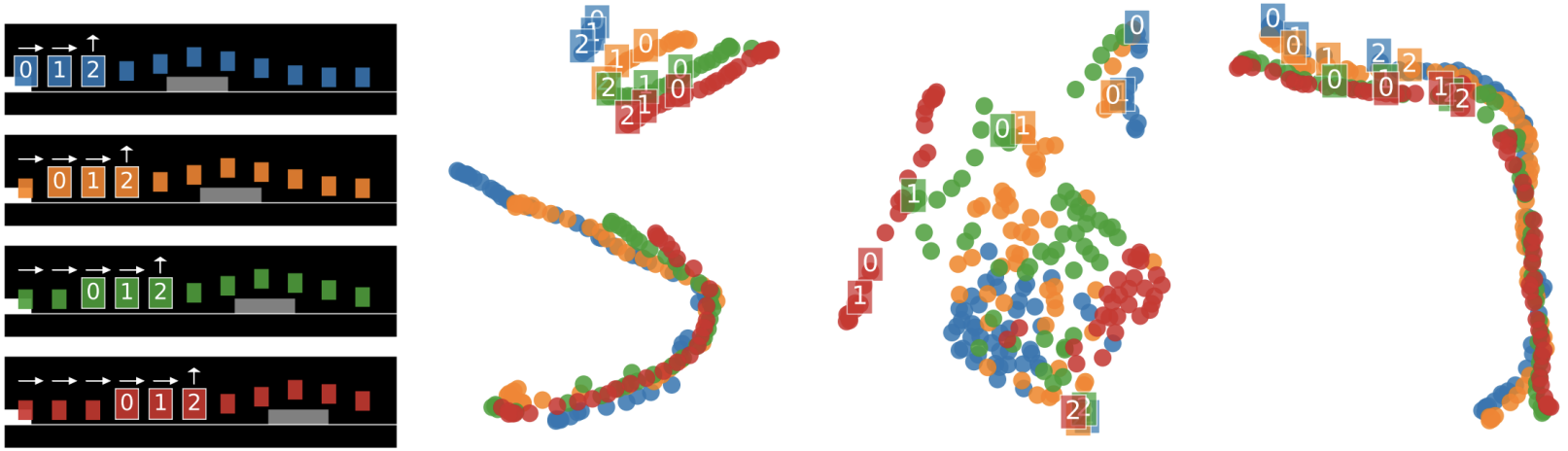
The algorithm

Algorithm 1 Contrastive Metric Embeddings (CMEs)

1: **Given**: State embedding $z_\theta(\cdot)$, Metric $d(\cdot,\cdot)$ Training environments $\{\mathcal{M}_i\}_{i=1}^N$. **Hyperparameters**: Temperature $1/\lambda$, Scale β, Total training steps K
2: **for** each step $k = 1, \ldots, K$ **do**
3: Sample a pair of training MDPs $\mathcal{M}_\mathcal{X}, \mathcal{M}_\mathcal{Y}$
4: Update θ to minimize $\mathscr{L}_{\text{CME}}$ where $\mathscr{L}_{\text{CME}} = \mathbb{E}_{\mathcal{M}_\mathcal{X}, \mathcal{M}_\mathcal{Y} \sim \rho}\left[L_\theta(\mathcal{M}_\mathcal{X}, \mathcal{M}_\mathcal{Y})\right]$
5: **end for**

1. Given a set of states $\mathcal{X}'$ from MDPs $\mathcal{M}_\mathcal{X}, \mathcal{M}_\mathcal{Y}$,
2. Select positive pairs $(\tilde{x}_y, y)$ by choosing the nearest neighbor of y in $\mathcal{X}'$ based on it's similarity given by Γ, where $\tilde{x}_y = \arg\max_{x \in \tilde{\mathcal{X}}'} \Gamma(x, y)$
3. The remaining states in $\mathcal{X}$ are negative pairs

Policy similarity embeddings (PSEs) cluster behaviorally-similar states together and dissimilar states apart

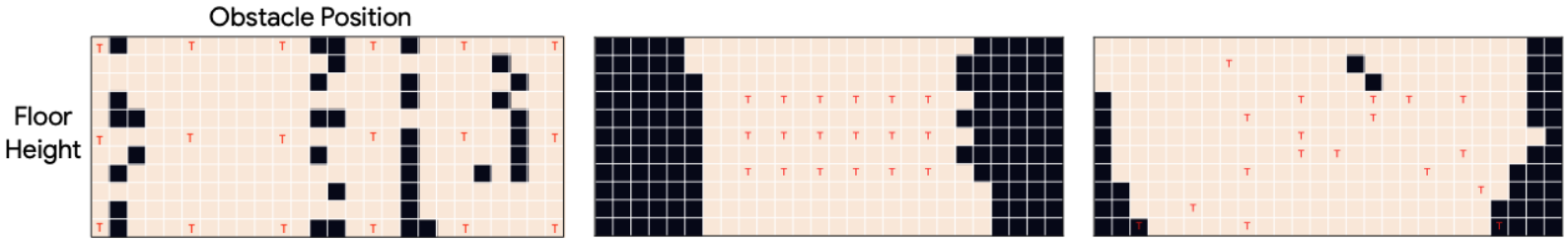


(a) Optimal Trajectories (b) PSEs (c) PSM + ℓ_2 embeddings (d) π^*-bisim. + CMEs

Unlike prior methods, PSEs partition the states into two sets: (1) all states before the jump and (2) states where actions do not affect the outcome (states after jump).

Experiments

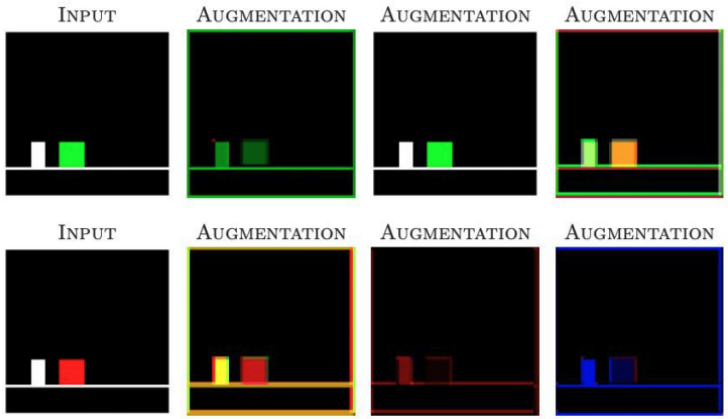
Data Augmentation	Method	Grid Configuration (%)		
		"Wide"	"Narrow"	Random
✗	Dropout and ℓ_2 reg.	17.8 (2.2)	10.2 (4.6)	9.3 (5.4)
	Bisimulation Transfer[4]	17.9 (0.0)	**17.9** (0.0)	30.9 (4.2)
	PSEs	**33.6** (10.0)	9.3 (5.3)	**37.7** (10.4)
✓	RandConv	50.7 (24.2)	33.7 (11.8)	71.3 (15.6)
	RandConv + π^*-bisimulation	41.4 (17.6)	17.4 (6.7)	33.4 (15.6)
	RandConv + PSEs	**87.0** (10.1)	**52.4** (5.8)	**83.4** (10.1)



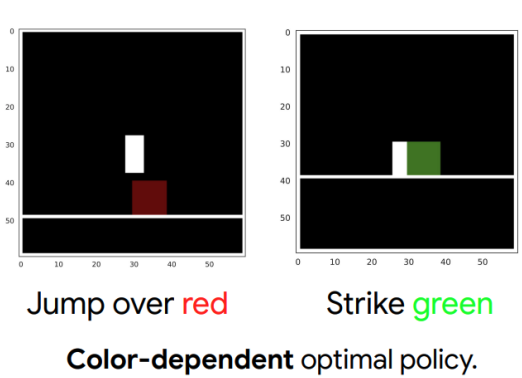
(a) "Wide" grid (b) "Narrow" grid (c) Random grid

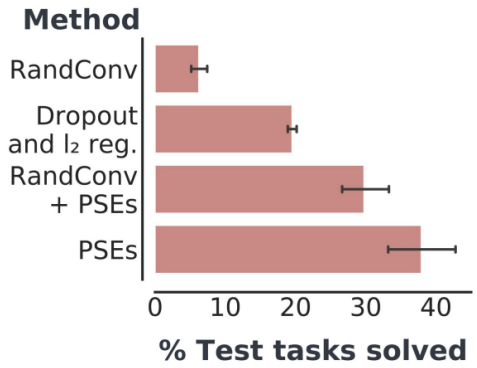
T is training task, beige is success, **black** is failure

Augmentations can hurt generalization if chosen poorly



PSEs do not require any domain knowledge but instead exploit the inherent structure of the RL tasks





In summary

- PSM defines a notion of similarity between states originated from different environments by the proximity of the long-term optimal behavior from these states
- By optimizing PSM, the agent learns an embedding in which states are close when the agent's optimal policies in current and future states are similar
- PSM is reward-agnostic, making it more robust for generalization compared to approaches that rely on reward information

Concluding thoughts

- Would I recommend reading it? Yes, it's well written
- Easy to implement? Maybe...

References

[1] Rishabh Agarwal, Marlos C Machado, Pablo Samuel Castro, and Marc G Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint arXiv:2101.05265*, 2021.

[2] Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10069–10076, 2020.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.