The Utility of Sparse Representations for Control in RL

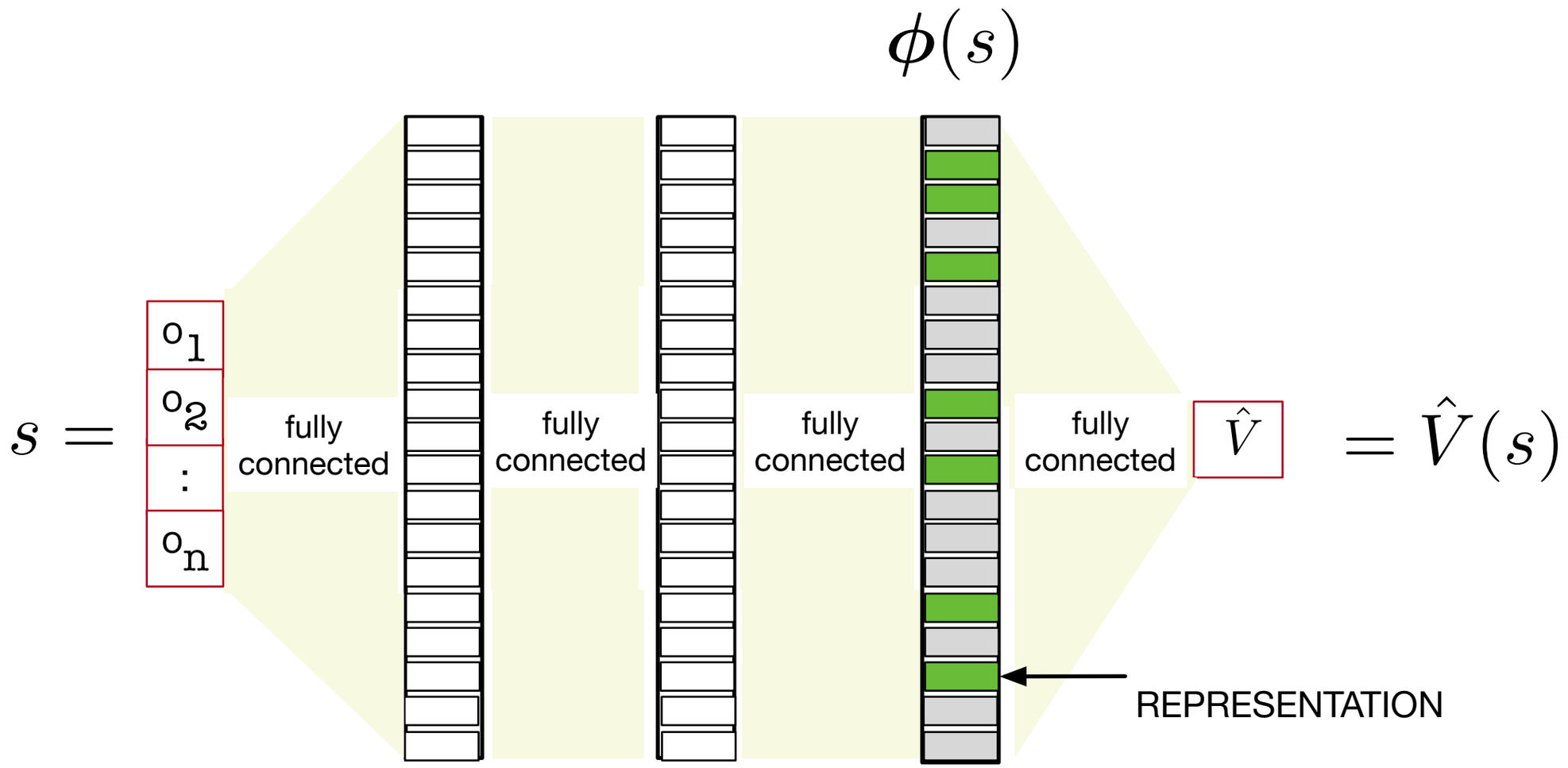
Martha White
Assistant Professor
University of Alberta
Thanks to collaborators Raksha Kumaraswamy, Vincent Liu, Lei Le

Goals for the talk

- Show some evidence that **sparse representations are useful for incrementally** learning values/policies

- Discuss a hypothesis that **interference is a problem due to bootstrapping**, not just from interference in the network

Learning representations

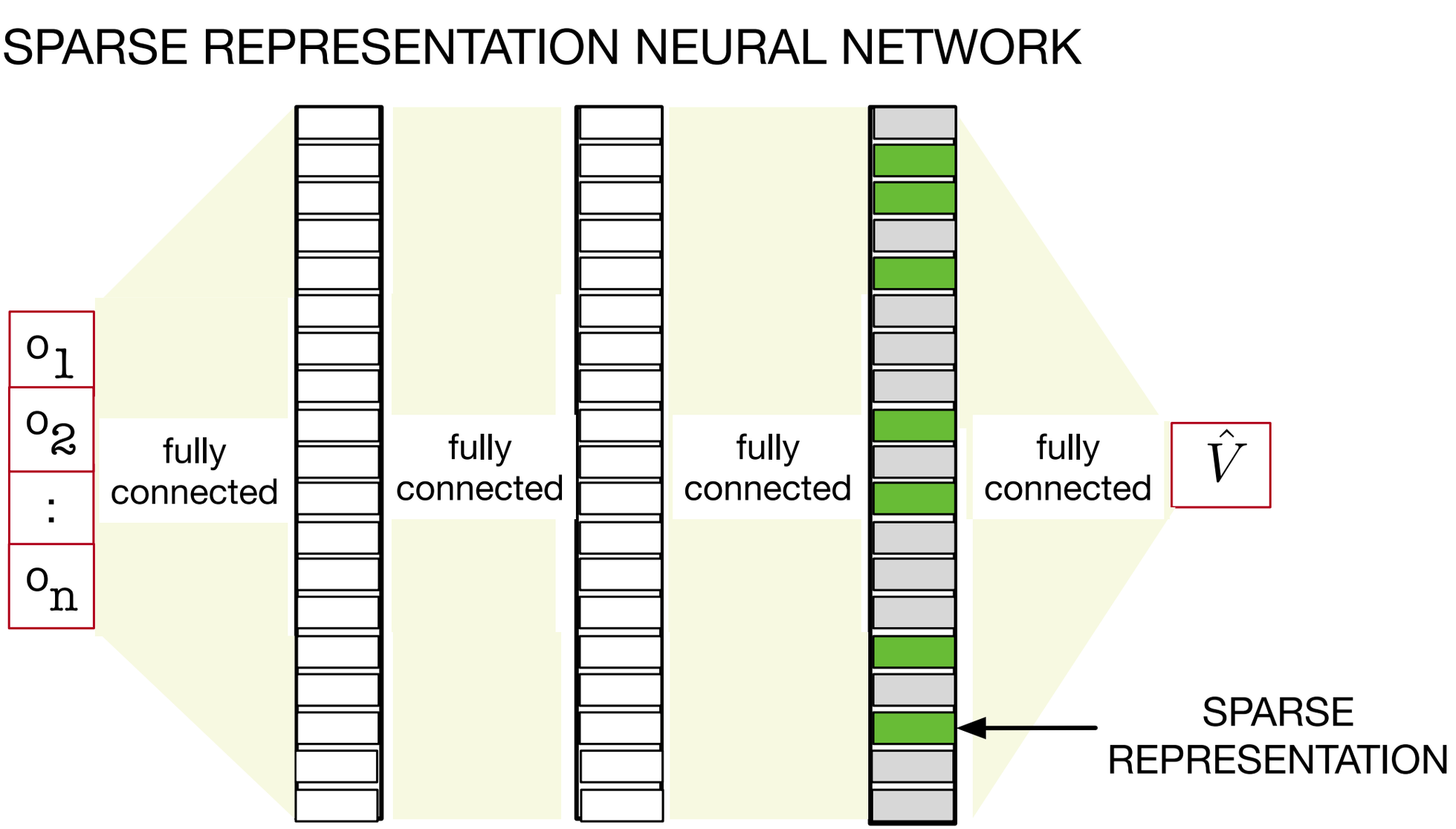


$$\hat{V}(s) = \boldsymbol{\phi}(s)^\top \mathbf{w}$$

What are desirable properties for this representation?

- Depends significantly on the problem setting
- In **RL**, using **incremental** learning methods that **bootstrap** (like Q-learning), for **control** where the agent uses Q-values to take actions (e.g., epsilon-greedy)
- One hypothesized desirable property: **Sparsity**
  - which will be motivated soon

Sparse Representations with NNs



Recognizing the utility of sparsity is not new

- Sutton, 1996 “Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding”

- French advocated for sparsity in work on catastrophic interference in 1990

- My goal:

  - re-emphasize the importance of sparsity, when learning (deep) neural networks

  - highlight the connection to interference in RL

Let’s start with a motivating experiment

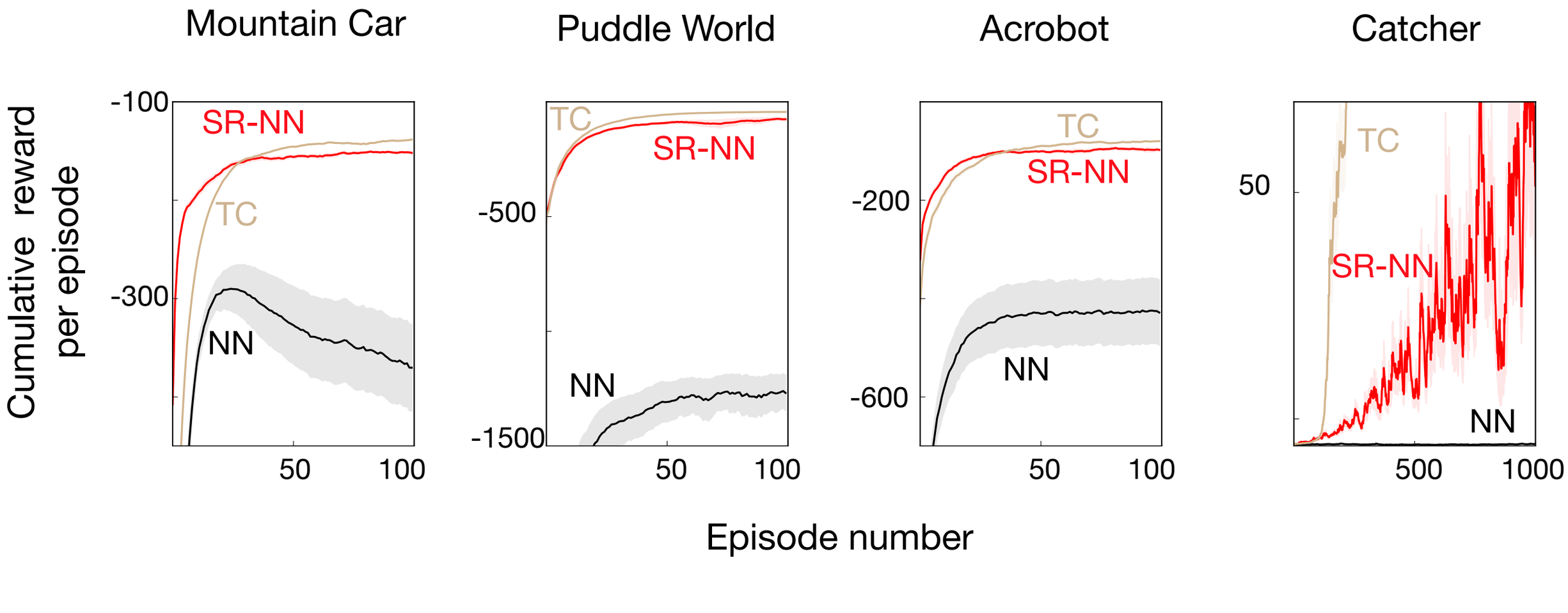
- Test two sparse representations
  - Tile Coding
  - Sparse Representation NNs (SR-NNs)
- Test one standard (dense) NN
- … in an online learning setting, with Sarsa and epsilon-greedy

Experiments

- Four simple domains, where
  - we can do a thorough empirical study
  - we expect RL + NNs *should* easily learn the optimal policy
- Learn representations ahead of time from a batch of samples from a suboptimal policy
- Sarsa + fixed basis with epsilon = 0.1
- No experience replay or target networks
- Architecture: 2 or 4 inputs —> 32 —> 256, ReLu

Sarsa + Sparse representations finds optimal policies whereas Sarsa + Dense representations usually fails

Sparsity seems useful in all four domains



Swept stepsize for all methods; sparse reps not that sensitive to stepsize
30 runs with shading corresponding to 95% confidence interval

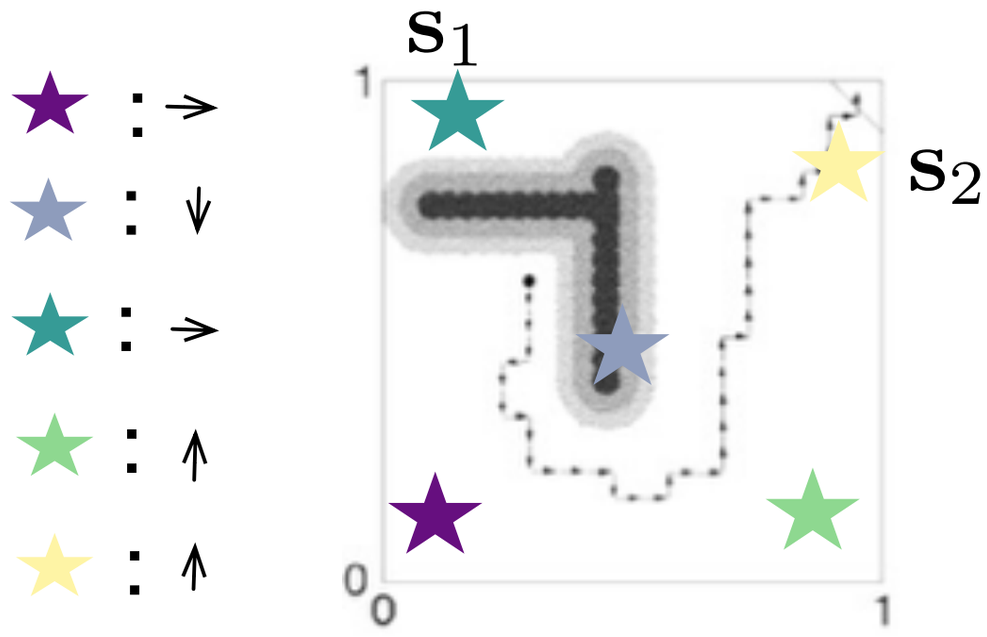
What is happening here?

Is sparsity helping, and if so, why?

One possibility: Interference

- Interference = incorrect generalization that overwrites or interferes with previous learning

- Learning incrementally: changing value estimate in one state (s1) interferes with another state (s2)



Interference problematic for control

- In a passive setting, bad estimates during learning may not be problematic, as long as learning converges at the end

- **But**, we use intermediate value estimates to **take actions**

- If we have bad intermediate value estimates, then this influences data gathering (in unpredictable ways)

Interference under bootstrapping

- Could be particularly problematic when bootstrapping, because interference incorrectly changes targets!

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha\big(\underbrace{R_{t+1} + \gamma \boldsymbol{\phi}(S_{t+1}, A_{t+1})^\top \mathbf{w}_t}_{\hat{G}_t} - \boldsymbol{\phi}(S_t, A_t)^\top \mathbf{w}_t\big)\boldsymbol{\phi}(S_{t+1}, A_{t+1})$$

$$\boldsymbol{\phi}(S_t, A_t = 2) = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\phi}(S_t) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

Changing weights while in St could change values for states with shared features

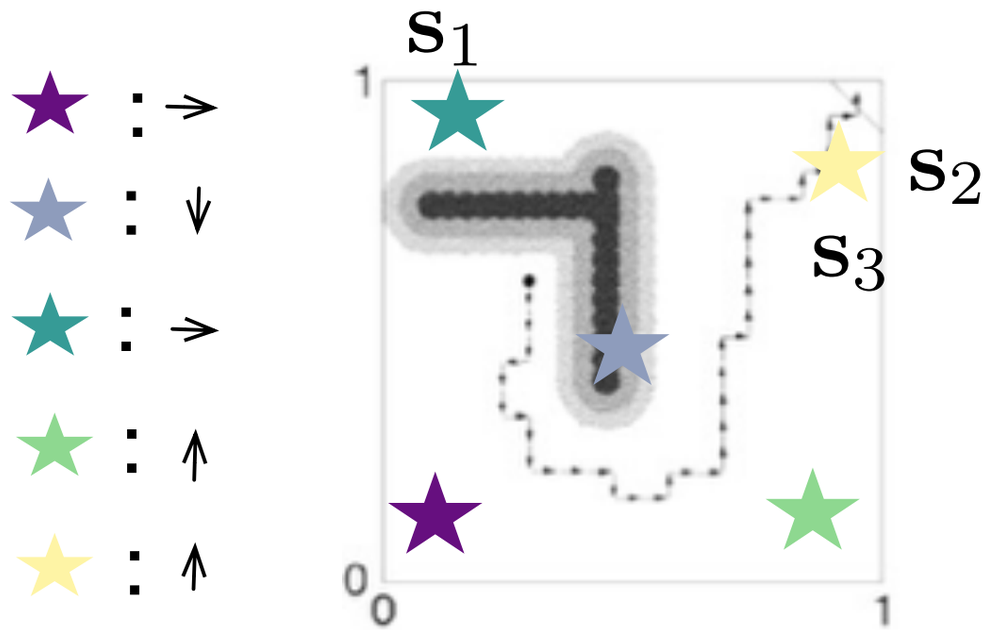
Example of problem of interference under bootstrapping

Update $\hat{Q}(\mathbf{s}_1, a)$

Imagine decreases value in $\hat{Q}(\mathbf{s}_2, a)$

Bootstrapping could decrease value of action up in $\mathbf{s}_3$

Agent decides down is better from state $\mathbf{s}_3$

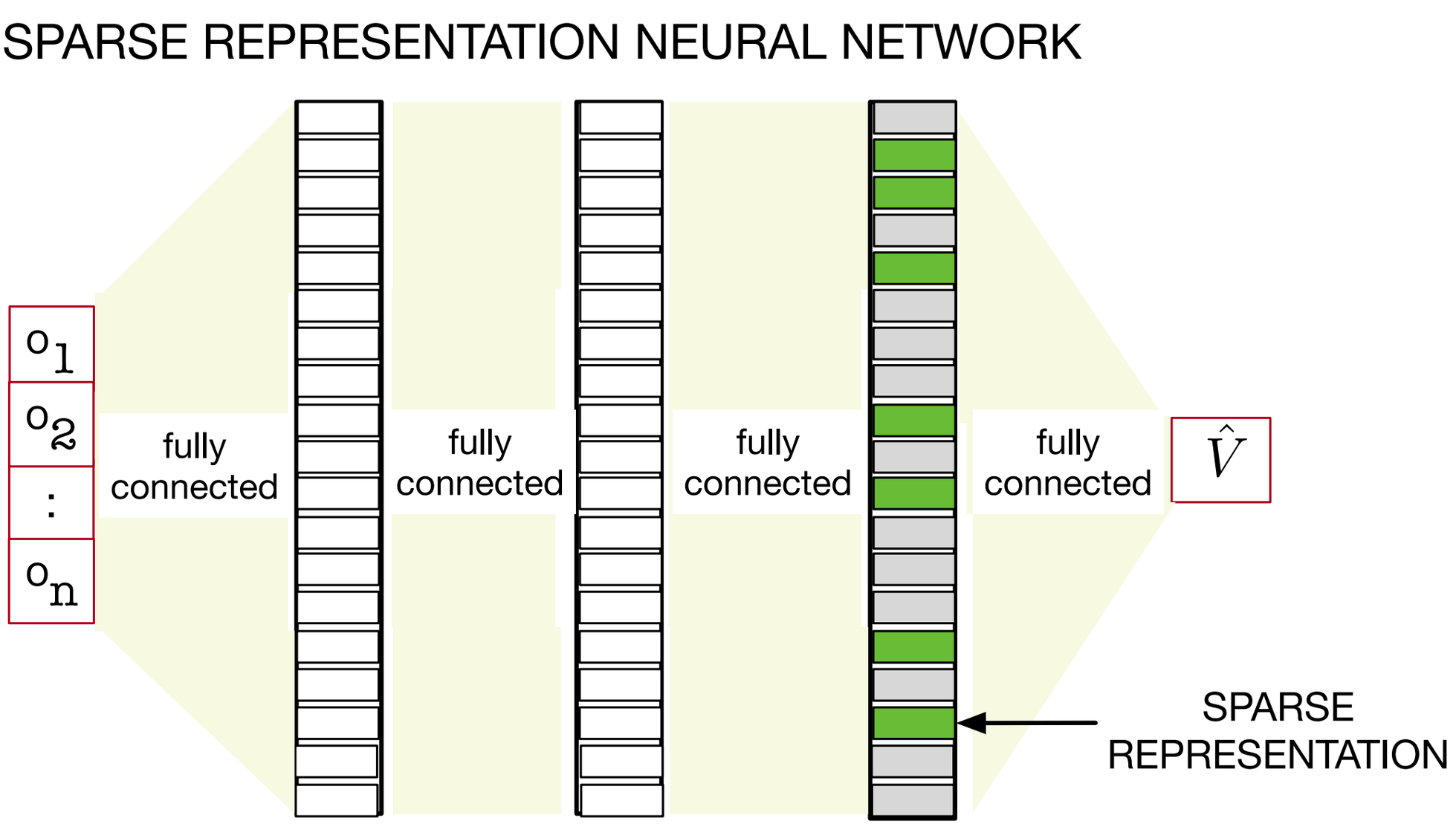


$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha\big(\underbrace{R_{t+1} + \gamma\boldsymbol{\phi}(S_{t+1}, A_{t+1})^\top \mathbf{w}_t}_{\hat{G}_t} - \boldsymbol{\phi}(S_t, A_t)^\top \mathbf{w}_t\big)\boldsymbol{\phi}(S_{t+1}, A_{t+1}$$

Interference can be highly catastrophic

- For our setting: using **incremental** learning methods that **bootstrap** (like Q-learning), for **control** where the agent uses Q-values to take actions (e.g., epsilon-greedy)
  - Desirable to have a representation that maintains **locality**

Sparsity provides some locality and some generalization



A sparse set of attributes describing an input (observation vector)

Overlapping attributes (not like state aggregation)

Sparsity could help reduce interference

- Each input only activates a small number of attributes

- If values are a linear function of attributes

  - each update only changes a small number of weights

  - each update only changes the values for a smaller set of inputs that share those attributes

  - attributes should represent similarity—other inputs with the same attributes should be similar or local to the given input—and so generalization should be reasonable

Different than interference inside the network

- Interference is usually discussed as a problem of overwriting/ interfering with values in the network
  - particularly when seeing tasks sequentially
- **Claim**: Even for a fixed basis for one problem, interference can occur
- The main point is that value updates in one state interfere with values in another state
  - bad because we bootstrap
  - bad because we use our estimates to take action

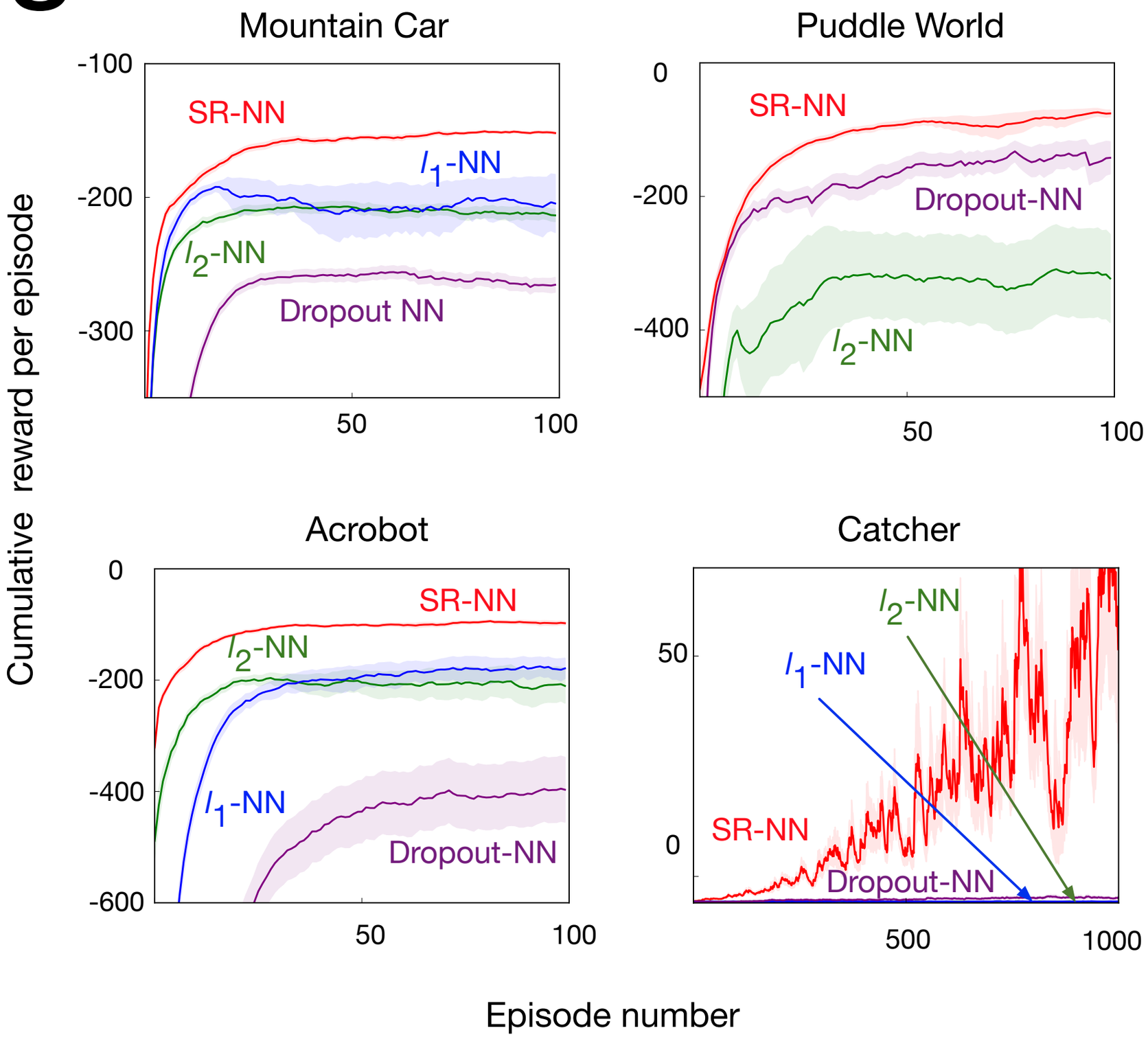
Other strategies for mitigating interference

- Target networks — fix the targets for value functions, so interference cannot impact targets

- And many others for interference in NNs
  - Localized representations using node sharpening
  - Replay (including older work on pseudo-patterns)
  - Fixing parts of the network
  - Elastic weight consolidation

Sarsa + Sparse representations finds optimal policies whereas Sarsa + Dense representations usually fails

Performance with other regularizations on the NN



Is sparsity influencing this performance?

- Dropout-NN did ok in Puddle-World, and is the one domain where it seemed to learn a sparse representation!

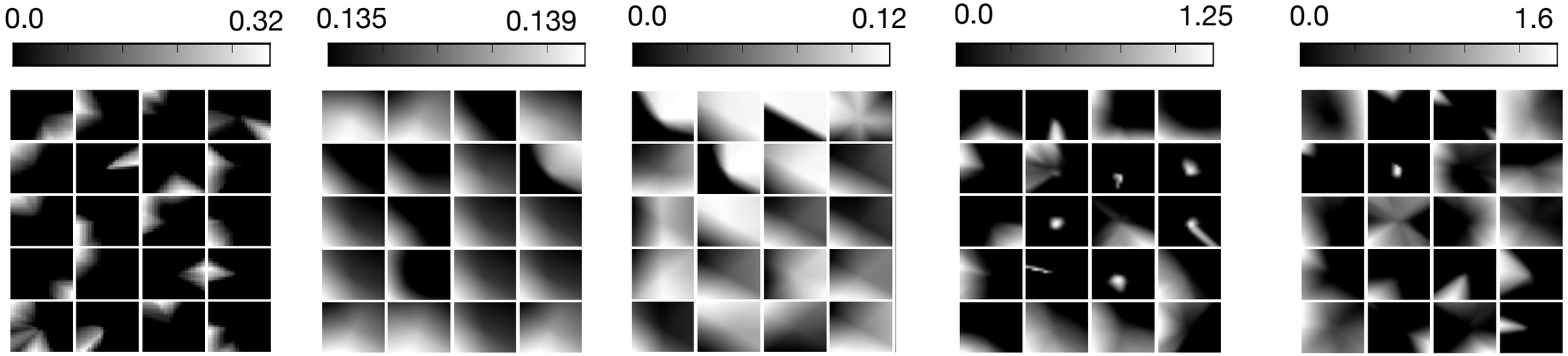
- SR-NN was the least sparse in Catcher, and the most noisy in that domain

- We can look at

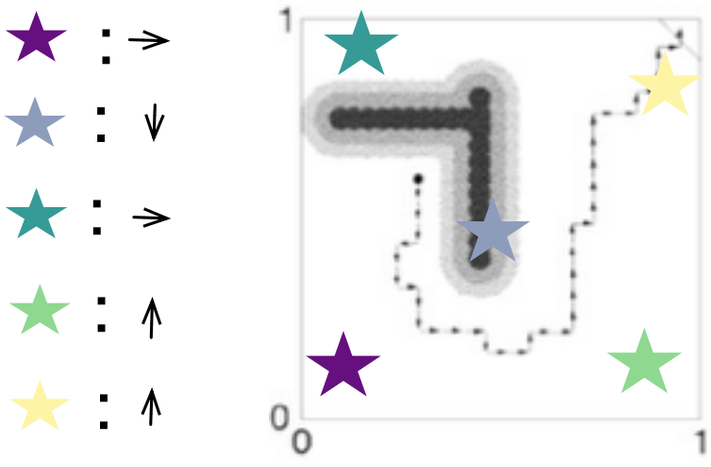
- (a) which representations are sparse, and

- (b) when interference occurs

Representation sparsity in Puddle World



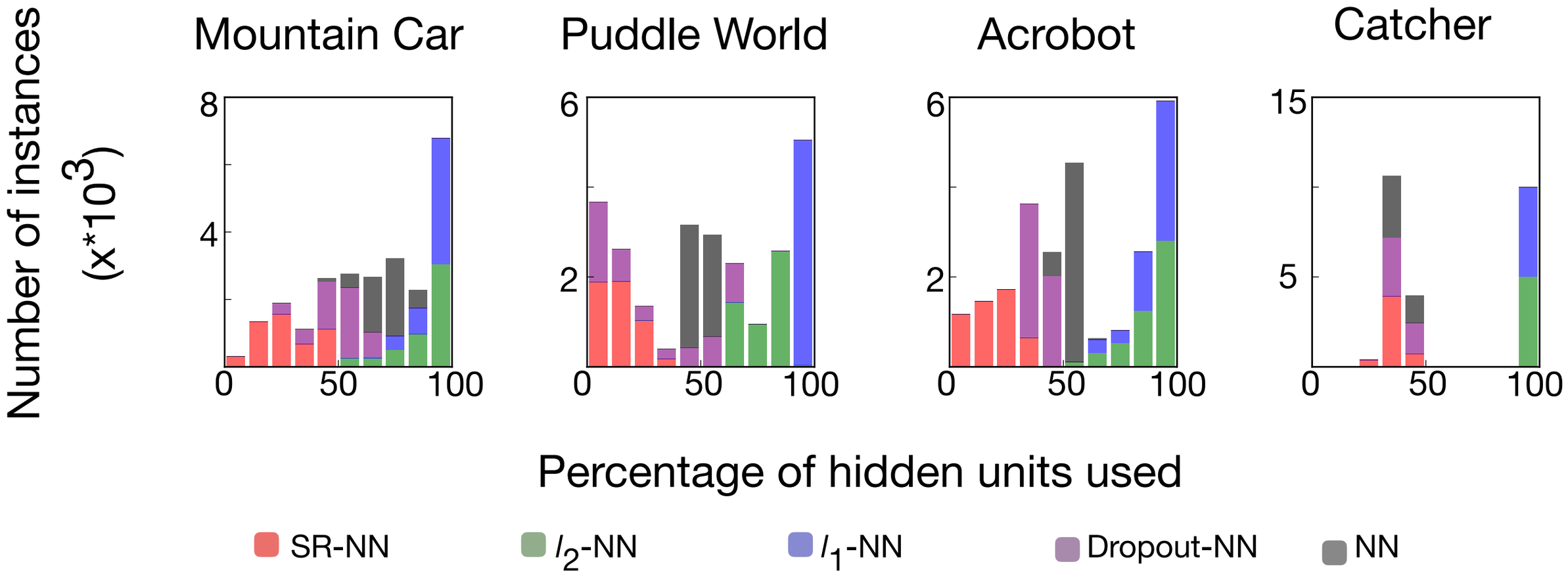
SR-NN l_2-NN l_1-NN Dropout-NN NN



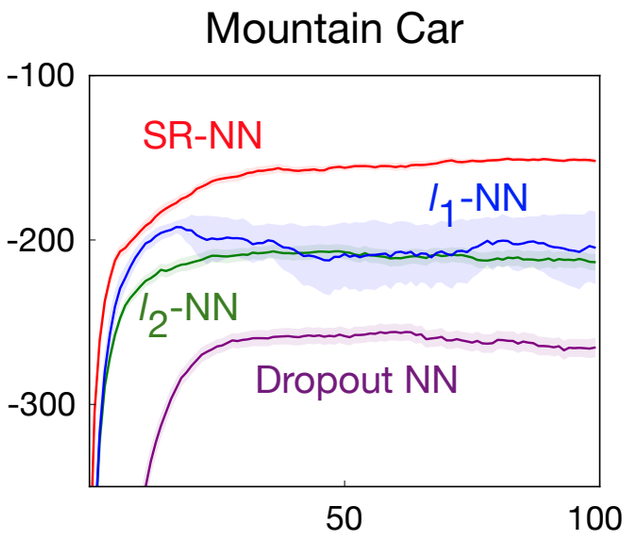
Representation overlap (number of shared active features)

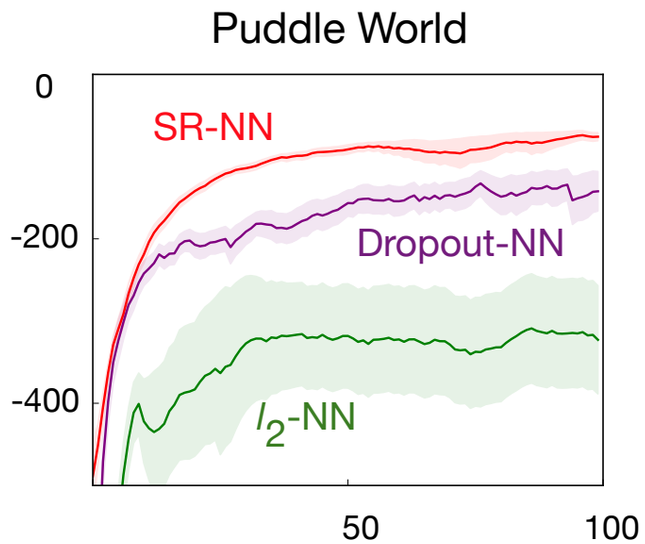
SR-NN	ℓ_2-NN	ℓ_1-NN	Dropout-NN	NN
8.8	111.5	142.5	31.2	54.0

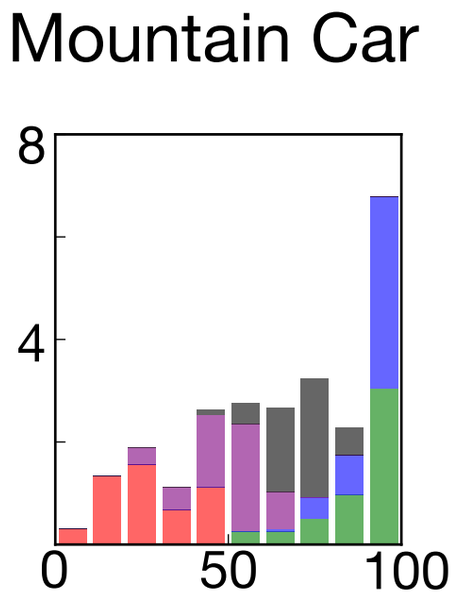
Rep sparsity (cont.)

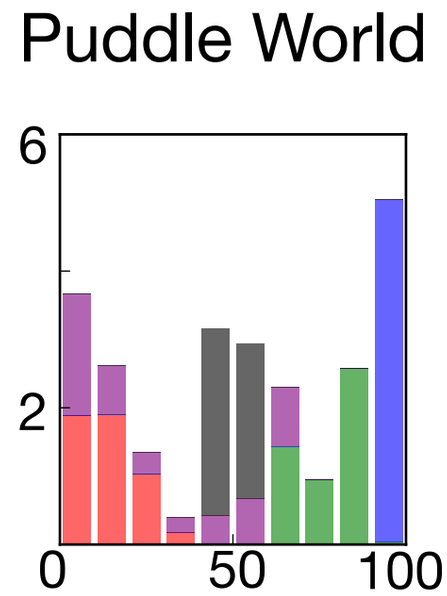


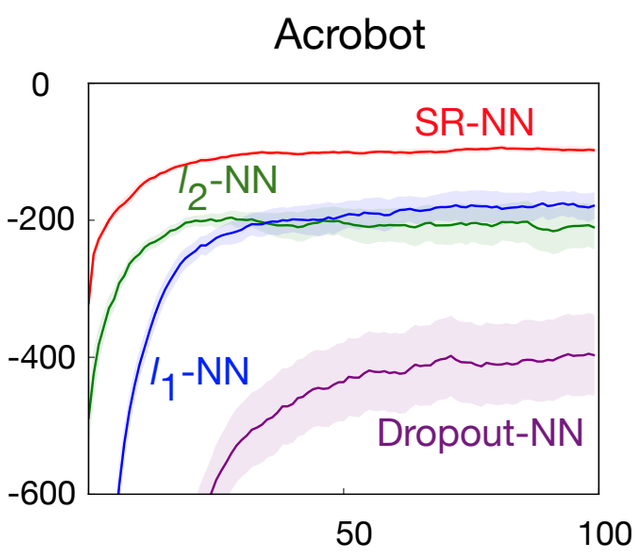
Sparsity and performance

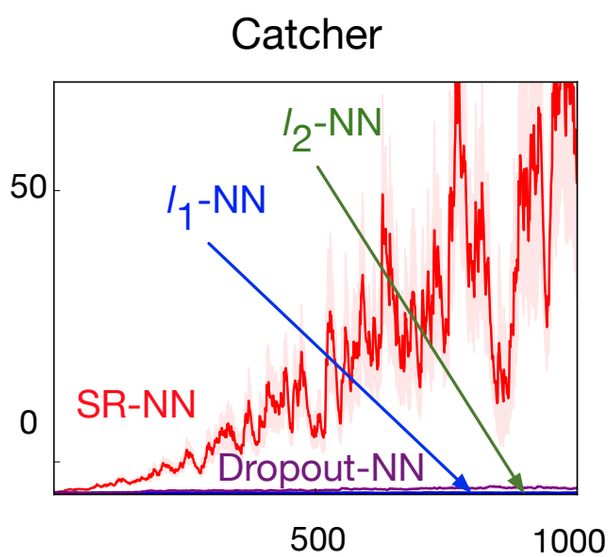


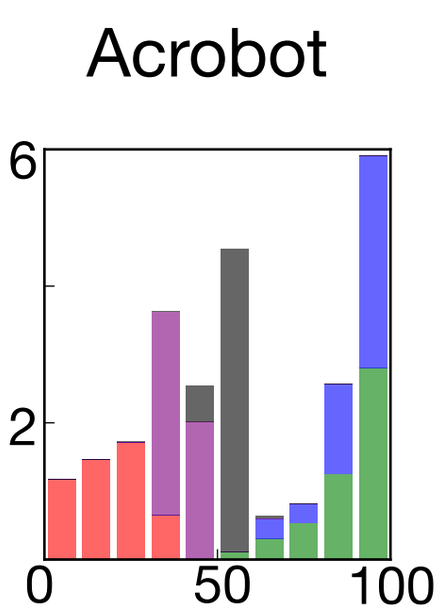


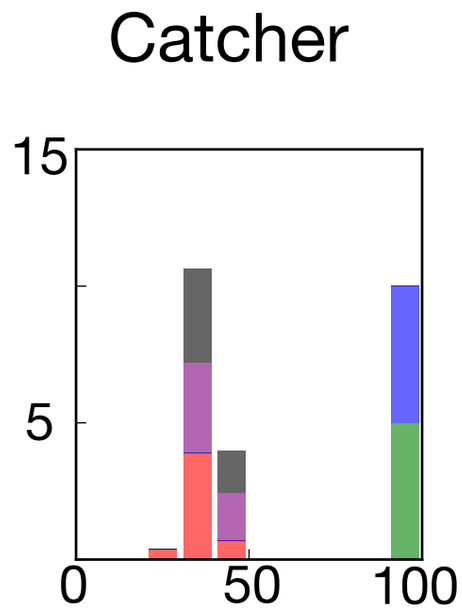






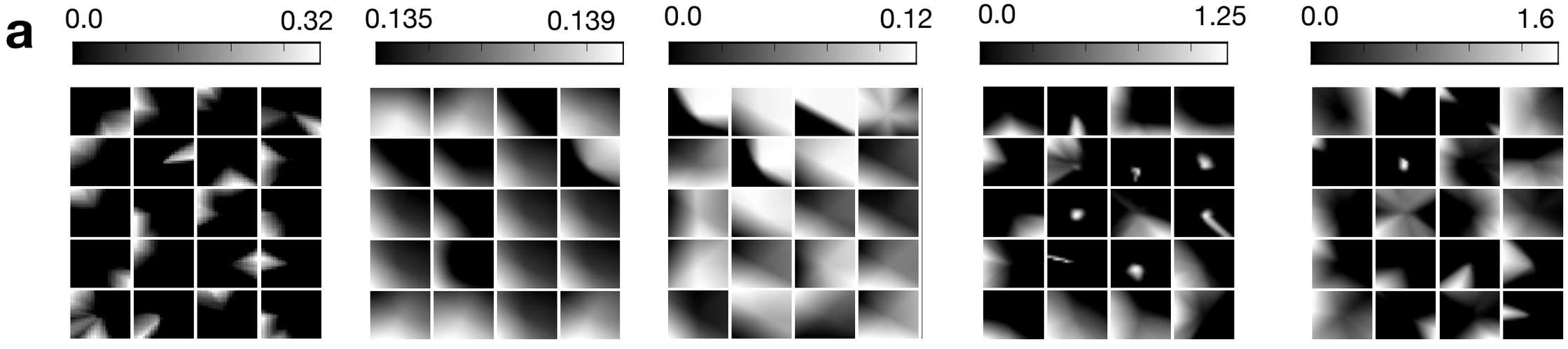






Clear interference in values
(which will be used in bootstrap targets)

Value interference



Hypotheses

- Sparse Representation NNs are promising for control in RL
  - without yet exactly knowing why they help
- Interference is a real problem in RL, even under function approximation with a fixed basis (linear fcn approx)
  - usually discussed for hidden layers in NNs
- Target networks might be playing a role in mitigating interference in bootstrap values in targets
  - rather than just stabilizing training of NNs