Representations and Exploration for Deep Reinforcement Learning using Singular Value Decomposition

Alex-Răzvan Ispas
Diego Andrés Torres Guarín
Thanh Gia Hieu Khuong

January 2024

Motivation

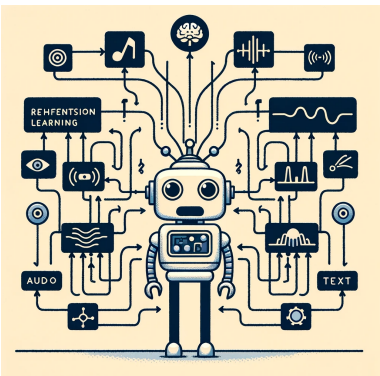
Current RL systems face challenges in environments with one or more of the following:

- Complex Observations (often multimodal)
- Partial Observability
- Sparse Reward Signals

Motivation

Current RL systems face challenges in environments with one or more of the following:

- Complex Observations (often multimodal)
- Partial Observability
- Sparse Reward Signals



Motivation

- **Learning good representations** allows the model to process the input more efficiently, as well as making it more robust to missing information.
- Good **exploration bonuses** allow the model to discover potentially rewarding regions.

Motivation

- **Learning good representations** allows the model to process the input more efficiently, as well as making it more robust to missing information.
- Good **exploration bonuses** allow the model to discover potentially rewarding regions.

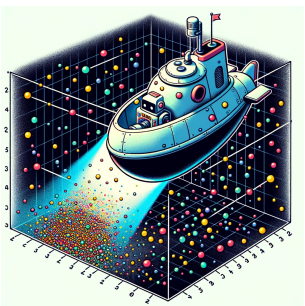


Figure: RL agent exploring a good representation space

Previous Approaches

Representation Learning:

Similar to an AutoEncoder, they try to derive useful representations by reconstructing observations.

Issues

- Can become impractical in complex observation scenarios.
- Prone to representation collapse when done in the latent space.

Previous Approaches

Representation Learning:

Similar to an AutoEncoder, they try to derive useful representations by reconstructing observations.

Issues

- Can become impractical in complex observation scenarios.
- Prone to representation collapse when done in the latent space.

Exploration:

Count-based exploration bonuses can lead an agent to discover useful regions

Issues

- Exact counts can be impractical in rich-observation settings.
- Decoupled from the agent's state representation, leading to suboptimal exploration.

Proposed solution

Auxiliary loss to drive **representation learning**.

$$\mathcal{L}_{\text{diag}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k}\sum_{i=1}^{k}\mathbf{\Sigma}(\theta)_{[i,i]},$$

$$\mathcal{L}_{\text{off}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbf{\Sigma}(\theta)^2_{[i,j]}.$$

Figure: Proposed losses (to be explained)

Proposed solution

Auxiliary loss to drive **representation learning**.

1. Preserves transition structure

$$\mathcal{L}_{\text{diag}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k}\sum_{i=1}^{k}\mathbf{\Sigma}(\theta)_{[i,i]},$$

$$\mathcal{L}_{\text{off}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbf{\Sigma}(\theta)^2_{[i,j]}.$$

Figure: Proposed losses (to be explained)

Proposed solution

Auxiliary loss to drive **representation learning**.

1. Preserves transition structure
2. Provides pseudo-counts for **exploration bonus**

$$\mathcal{L}_{\text{diag}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k}\sum_{i=1}^{k}\mathbf{\Sigma}(\theta)_{[i,i]},$$

$$\mathcal{L}_{\text{off}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbf{\Sigma}(\theta)^2_{[i,j]}.$$

Figure: Proposed losses (to be explained)

Proposed solution

Auxiliary loss to drive **representation learning**.

1. Preserves transition structure
2. Provides pseudo-counts for **exploration bonus**
3. Suitable for large-scale problems

$$\mathcal{L}_{\text{diag}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k}\sum_{i=1}^{k}\mathbf{\Sigma}(\theta)_{[i,i]},$$

$$\mathcal{L}_{\text{off}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbf{\Sigma}(\theta)^2_{[i,j]}.$$

Figure: Proposed losses (to be explained)

Proposed solution

Auxiliary loss to drive **representation learning**.

1. Preserves transition structure
2. Provides pseudo-counts for **exploration bonus**
3. Suitable for large-scale problems
4. Effective in tabular and function approximation

$$\mathcal{L}_{\text{diag}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k}\sum_{i=1}^{k}\mathbf{\Sigma}(\theta)_{[i,i]},$$

$$\mathcal{L}_{\text{off}}\left(\mathbf{\Sigma}(\theta)\right) \coloneqq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbf{\Sigma}(\theta)^2_{[i,j]}.$$

Figure: Proposed losses (to be explained)

Markov Decision Process Set Up

The MDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, p, r, d_0)$ such that:

- $\mathcal{S}$ - the set of finite stets
- $\mathcal{A}$ - set of finite actions
- p - the transition function
- r - the reward function
- d_0 - the starting state

Other notations

- $P_\pi \in \mathbb{R}^{|S|\times|S|}$ - the transition matrix induced by π
- $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t\right]$ - performance of policy π
- $\gamma \in [0, 1)$ - the discount factor
- R_t - reward observed at time t

Methodology

Why SVD decomposition?

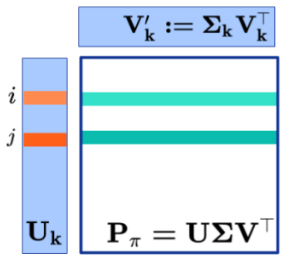


Figure: top-k SVD

Why SVD decomposition?

- Similar states will correspond to rows of U that are similar as well

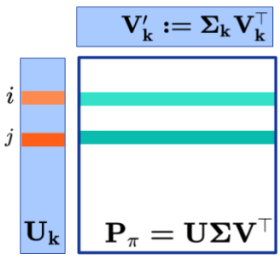


Figure: top-k SVD

Why SVD decomposition?

- Similar states will correspond to rows of U that are similar as well

  $\Rightarrow$ U is suitable as state representations

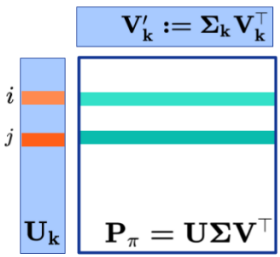


Figure: top-k SVD

Why SVD decomposition?

- Similar states will correspond to rows of U that are similar as well

  $\Rightarrow$ U is suitable as state representations

+ It generalizes the eigen decomposition of P_π

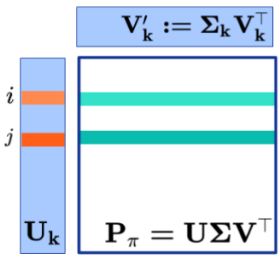


Figure: top-k SVD

Why SVD decomposition?

- Similar states will correspond to rows of U that are similar as well

  $\Rightarrow$ U is suitable as state representations

+ It generalizes the eigen decomposition of P_π
+ It always exists

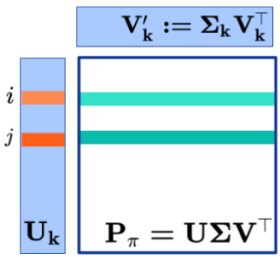


Figure: top-k SVD

Tabular Setting

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^{n}$ - collection of n transition tuples while using π

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^{n}$ - collection of n transition tuples while using π
- $d \in \Delta(s)$ - the "a priori" distribution of states s

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^{n}$ - collection of n transition tuples while using π
- $d \in \Delta(s)$ - the "a priori" distribution of states s
- $D = \text{diag}(d) \in \mathbb{R}^{|S| \times |S|}$

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^{n}$ - collection of n transition tuples while using π
- $d \in \Delta(s)$ - the "a priori" distribution of states s
- $D = \text{diag}(d) \in \mathbb{R}^{|S| \times |S|}$

Having a good estimate of P_π can become prohibitively expensive, as it is necessary to visit every state many times

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^{n}$ - collection of n transition tuples while using π
- $d \in \Delta(s)$ - the "a priori" distribution of states s
- $D = \text{diag}(d) \in \mathbb{R}^{|S| \times |S|}$

Having a good estimate of P_π can become prohibitively expensive, as it is necessary to visit every state many times

$\Rightarrow$ a weighted version of P_π is considered:

Tabular Setting

- $\mathcal{D} = \{(s, a, r, s')_i\}_{i=1}^n$ - collection of n transition tuples while using π
- $d \in \Delta(s)$ - the "a priori" distribution of states s
- $D = \text{diag}(d) \in \mathbb{R}^{|S| \times |S|}$

Having a good estimate of P_π can become prohibitively expensive, as it is necessary to visit every state many times

$\Rightarrow$ a weighted version of P_π is considered:

We will decompose DP_π ($DP_\pi \propto P_\pi$ when d is uniform).

Why do we estimate the scaled matrix DP_π instead of P_π?

Why do we estimate the scaled matrix DP_π instead of P_π?

- In the practical case, it is impossible to visit every single state

Why do we estimate the scaled matrix DP_π instead of P_π?

- In the practical case, it is impossible to visit every single state
- For the rare and hence unvisited states, we have no way of estimating their row in P_π

Why do we estimate the scaled matrix DP_π instead of P_π?

- In the practical case, it is impossible to visit every single state
- For the rare and hence unvisited states, we have no way of estimating their row in P_π
- ↑ When estimating DP_π, filling the corresponding row with 0 is a good estimation

Representation Learning

Representation Learning

- $x_s \in \mathbb{R}^{|\mathcal{S}|}$ - one-hot encoding corresponding to state s

Representation Learning

- $x_s \in \mathbb{R}^{|\mathcal{S}|}$ - one-hot encoding corresponding to state s
- $y_s \in \mathbb{R}^{n \times |\mathcal{S}|}$ - one-hot encoding corresponding to s'

Representation Learning

- $x_s \in \mathbb{R}^{|\mathcal{S}|}$ - one-hot encoding corresponding to state s
- $y_s \in \mathbb{R}^{n \times |\mathcal{S}|}$ - one-hot encoding corresponding to s'
- $X \in \mathbb{R}^{n \times |\mathcal{S}|}$ - the stack of x_s for states s in dataset $\mathcal{D}$

Representation Learning

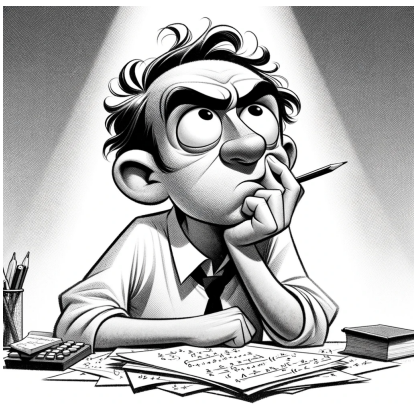
- $x_s \in \mathbb{R}^{|\mathcal{S}|}$ - one-hot encoding corresponding to state s
- $y_s \in \mathbb{R}^{n \times |\mathcal{S}|}$ - one-hot encoding corresponding to s'
- $X \in \mathbb{R}^{n \times |\mathcal{S}|}$ - the stack of x_s for states s in dataset $\mathcal{D}$
- $Y \in \mathbb{R}^{n \times |\mathcal{S}|}$ - the stack of s' in dataset $\mathcal{D}$

Representation Learning

- $x_s \in \mathbb{R}^{|\mathcal{S}|}$ - one-hot encoding corresponding to state s
- $y_s \in \mathbb{R}^{n \times |\mathcal{S}|}$ - one-hot encoding corresponding to s'
- $X \in \mathbb{R}^{n \times |\mathcal{S}|}$ - the stack of x_s for states s in dataset $\mathcal{D}$
- $Y \in \mathbb{R}^{n \times |\mathcal{S}|}$ - the stack of s' in dataset $\mathcal{D}$

We provide a sample estimation of DP_π that we would like to decompose. However, we will **never** actually construct this.

Then how are we going to estimate DP_{π}?



Representation Learning: Estimator of DP_π

Representation Learning: Estimator of DP_π

Let us define:

$$A_n := \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

Representation Learning: Estimator of DP_π

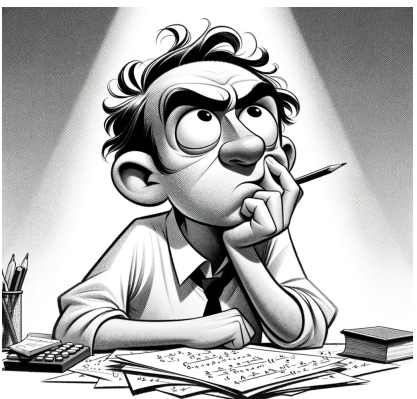
Let us define:

$$A_n := \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

Theorem

A_n is an unbiased and a consistent estimator of DP_π, i.e.,
$\forall j, \mathbb{E}[A_{n,[i,j]}] = DP_{\pi_{[i,j]}}, \quad A_{n,[i,j]} \xrightarrow{a.s.} DP_{\pi_{[i,j]}}$.

Can the estimator and the SVD decomposition lead me to a loss function?



Representation Learning: Loss function

Let us go deeper:

$$A_n = \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

Representation Learning: Loss function

Let us go deeper:

$$A_n = \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

$$A_n = U \Sigma V^T$$

Representation Learning: Loss function

Let us go deeper:

$$A_n = \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

$$A_n = U \Sigma V^T$$

$$U^T A_n V = \Sigma$$

Representation Learning: Loss function

Let us go deeper:

$$A_n = \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

$$A_n = U \Sigma V^T$$

$$U^T A_n V = \Sigma$$

$$\frac{1}{n} \sum_{i=1}^{n} U^T x_i y_i^T V = \Sigma$$

Representation Learning: Loss function

Let us go deeper:

$$A_n = \frac{1}{n} X^T Y = \frac{1}{n} \sum_{i=1}^{n} x_i y_i^T$$

$$A_n = U \Sigma V^T$$

$$U^T A_n V = \Sigma$$

$$\frac{1}{n} \sum_{i=1}^{n} U^T x_i y_i^T V = \Sigma$$

Now, let $f_U : S \to \mathbb{R}^{|S|}$ and $g_V : S \to \mathbb{R}^{|S|}$:

$$\frac{1}{n} \sum_{i=1}^{n} f_U(x_i) g_V(y_i)^T = \Sigma.$$

Representation Learning: Loss function

If we want the top-k decomposition, let $\hat{U} \in \mathbb{R}^{|S| \times k}$ and $\hat{V} \in \mathbb{R}^{|S| \times k}$ be the estimates for top-k left and right singular vectors ($\Sigma(\hat{U}, \hat{V}) \in \mathbb{R}^{k \times k}$):

Representation Learning: Loss function

If we want the top-k decomposition, let $\hat{U} \in \mathbb{R}^{|S| \times k}$ and $\hat{V} \in \mathbb{R}^{|S| \times k}$ be the estimates for top-k left and right singular vectors ($\Sigma(\hat{U}, \hat{V}) \in \mathbb{R}^{k \times k}$):

$$\Sigma(\hat{U}, \hat{V}) := \frac{1}{n} \sum_{i=1}^{n} f_{\hat{U}}(x_i) g_{\hat{V}}(y_i)^T,$$

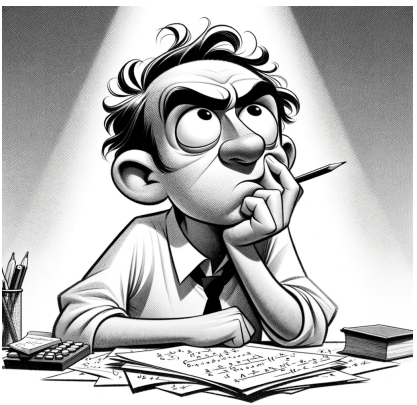
Representation Learning: Loss function

If we want the top-k decomposition, let $\hat{U} \in \mathbb{R}^{|S| \times k}$ and $\hat{V} \in \mathbb{R}^{|S| \times k}$ be the estimates for top-k left and right singular vectors ($\Sigma(\hat{U}, \hat{V}) \in \mathbb{R}^{k \times k}$):

$$\Sigma(\hat{U}, \hat{V}) := \frac{1}{n} \sum_{i=1}^{n} f_{\hat{U}}(x_i) g_{\hat{V}}(y_i)^T,$$

$$\hat{U}^*, \hat{V}^* := \arg\min_{\hat{U}, \hat{V}} - \sum_{i=1}^{k} \Sigma(\hat{U}, \hat{V})_{[i,i]}$$
$$\text{s.t. } \Sigma(\hat{U}, \hat{V})_{[i,j]} = 0 \text{ for all } i \neq j$$

Can we extend it for Function approximation?



Representation Learning: Constrained Optimization

$$\hat{U}^*, \hat{V}^* := \arg\min_{\hat{U},\hat{V}} -\sum_{i=1}^{k} \Sigma(\hat{U}, \hat{V})_{[i,i]}$$
$$\text{s.t. } \Sigma(\hat{U}, \hat{V})_{[i,j]} = 0 \text{ for all } i \neq j$$

Representation Learning: Constrained Optimization

$$\hat{U}^*, \hat{V}^* := \arg\min_{\hat{U},\hat{V}} - \sum_{i=1}^{k} \Sigma(\hat{U}, \hat{V})_{[i,i]}$$
$$\text{s.t. } \Sigma(\hat{U}, \hat{V})_{[i,j]} = 0 \text{ for all } i \neq j$$

We need:

- Soft constraints
- Batch normalization to constrain the weights of U and V to remain orthogonal matrices

Representation Learning: Loss with soft constraints

Let functions $f_{\theta_1} : \mathbb{R}^d \to \mathbb{R}^k$ and $g_{\theta_2} : \mathbb{R}^d \to \mathbb{R}^k$ that are parameterized by weights $\theta := [\theta_1, \theta_2]$ and take states as inputs:

$$\Sigma(\theta) := \frac{1}{n} \sum_{i=1}^{n} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^\top \in \mathbb{R}^{k \times k}. \tag{1}$$

Representation Learning: Loss with soft constraints

Let functions $f_{\theta_1} : \mathbb{R}^d \to \mathbb{R}^k$ and $g_{\theta_2} : \mathbb{R}^d \to \mathbb{R}^k$ that are parameterized by weights $\theta := [\theta_1, \theta_2]$ and take states as inputs:

$$\Sigma(\theta) := \frac{1}{n} \sum_{i=1}^{n} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^\top \in \mathbb{R}^{k \times k}. \tag{1}$$

Specifically, let

$$\mathcal{L}_{diag}(\Sigma(\theta)) := -\frac{1}{k} \sum_{i=1}^{k} \Sigma(\theta)_{i,i}, \tag{2}$$

$$\mathcal{L}_{off}(\Sigma(\theta)) := \frac{1}{k^2 - k} \sum_{i,j=1, i \neq j}^{k} \Sigma(\theta)_{i,j}^2. \tag{3}$$

Representation Learning: Loss with soft constraints

Let functions $f_{\theta_1} : \mathbb{R}^d \to \mathbb{R}^k$ and $g_{\theta_2} : \mathbb{R}^d \to \mathbb{R}^k$ that are parameterized by weights $\theta := [\theta_1, \theta_2]$ and take states as inputs:

$$\Sigma(\theta) := \frac{1}{n} \sum_{i=1}^{n} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^\top \in \mathbb{R}^{k \times k}. \tag{1}$$

Specifically, let

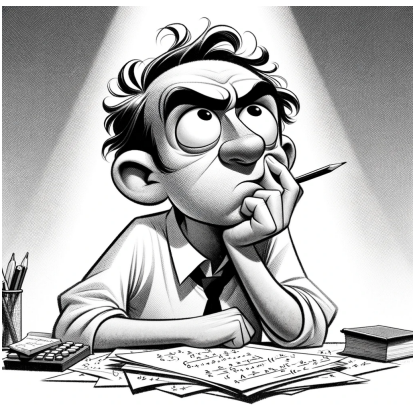
$$\mathcal{L}_{diag}(\Sigma(\theta)) := -\frac{1}{k} \sum_{i=1}^{k} \Sigma(\theta)_{i,i}, \tag{2}$$

$$\mathcal{L}_{off}(\Sigma(\theta)) := \frac{1}{k^2 - k} \sum_{i,j=1, i \neq j}^{k} \Sigma(\theta)_{i,j}^2. \tag{3}$$

We define a loss with soft constraints by combining (2) and (3):

$$\mathcal{L}(\Sigma(\theta)) := -L_{diag}(\Sigma(\theta)) + \lambda_r L_{off}(\Sigma(\theta)),$$

How do we make it work with a mini-batch?



Representation Learning: Mini-batch Optimisation

Let us define:

$$\hat{\Sigma}(\theta) = \frac{1}{b} \sum_{i=1}^{b} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^T,$$

where $b << n$ is the minibatch

Representation Learning: Mini-batch Optimisation

Let us define:

$$\hat{\Sigma}(\theta) = \frac{1}{b}\sum_{i=1}^{b} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^T,$$

where $b << n$ is the minibatch

Theorem

Gradient of $L_{diag}(\hat{\Sigma}(\theta))$ is an unbiased estimator of $L_{diag}(\Sigma(\theta))$, and gradient of $L_{off}(\hat{\Sigma}(\theta))$ is in general a biased estimate of $L_{off}(\Sigma(\theta))$, i.e.,

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\mathcal{L}_{diag}(\hat{\Sigma}(\theta))\right] = \frac{\partial}{\partial\theta}\mathcal{L}_{diag}(\Sigma(\theta)),$$

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\mathcal{L}_{off}(\hat{\Sigma}(\theta))\right] \neq \frac{\partial}{\partial\theta}\mathcal{L}_{off}(\Sigma(\theta)),$$

Proof.

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\,\mathcal{L}_{\text{off}}\left(\frac{1}{b}\Sigma(\theta)\right)\right] = \mathbb{E}\left[\frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\left(\frac{1}{b}\Sigma(\theta)^2_{[i,j]}\right)\right],$$

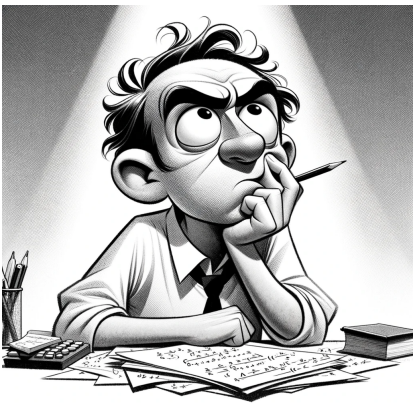
$$= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(\frac{1}{b}\Sigma(\theta)^2_{[i,j]}\right)\right],$$

$$= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(f_{\theta_1}(S)g_{\theta_2}(S')^\top_{[i,j]}\right)^2\right],$$

$$\neq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(f_{\theta_1}(S)g_{\theta_2}(S')^\top_{[i,j]}\right)\right]^2,$$

$$= \frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\Sigma(\theta))$$

Is there a way in which we can avoid this issue?



Representation Learning: Double-sampling trick

We estimate a separate batch of data:

$$\hat{\hat{\Sigma}}(\theta) = \frac{1}{b_2} \sum_{i=1}^{b_2} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^T,$$

where $b_2 << n$ is the minibatch

Representation Learning: Double-sampling trick

We estimate a separate batch of data:

$$\hat{\hat{\Sigma}}(\theta) = \frac{1}{b_2} \sum_{i=1}^{b_2} f_{\theta_1}(s_i) g_{\theta_2}(s_i)^T,$$

where $b_2 << n$ is the minibatch

We define $\hat{\mathcal{L}}(\theta)$ as:

$$\hat{\mathcal{L}}_{\text{off}}(\theta) := \frac{1}{(k^2 - k)} \sum_{i,j=1, i \neq j}^{k} \hat{\Sigma}(\theta)_{[i,j]} \cdot sg\left(\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right)$$
$$+ \frac{1}{(k^2 - k)} \sum_{i,j=1, i \neq j}^{k} \hat{\hat{\Sigma}}(\theta)_{[i,j]} \cdot sg\left(\hat{\Sigma}(\theta)_{[i,j]}\right),$$

where sg is the stop gradient sign

Representation Learning: Double-sampling trick

$$\hat{\mathcal{L}}_{\text{off}}(\theta) := \frac{1}{(k^2-k)} \sum_{i,j=1, i\neq j}^{k} \hat{\Sigma}(\theta)_{[i,j]} \cdot sg\left(\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right)$$

$$+ \frac{1}{(k^2-k)} \sum_{i,j=1, i\neq j}^{k} \hat{\hat{\Sigma}}(\theta)_{[i,j]} \cdot sg\left(\hat{\Sigma}(\theta)_{[i,j]}\right),$$

where sg is the stop gradient sign

Representation Learning: Double-sampling trick

$$\hat{\mathcal{L}}_{\text{off}}(\theta) := \frac{1}{(k^2-k)} \sum_{i,j=1, i\neq j}^{k} \hat{\Sigma}(\theta)_{[i,j]} \cdot sg\left(\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right)$$

$$+ \frac{1}{(k^2-k)} \sum_{i,j=1, i\neq j}^{k} \hat{\hat{\Sigma}}(\theta)_{[i,j]} \cdot sg\left(\hat{\Sigma}(\theta)_{[i,j]}\right),$$

where sg is the stop gradient sign

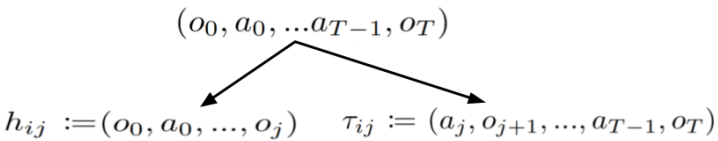
Theorem

Gradient $\frac{\partial}{\partial\theta}\hat{\mathcal{L}}_{off}(\theta)$ *is an unbiased estimator of* $\frac{\partial}{\partial\theta}\mathcal{L}_{off}(\Sigma(\theta))$*, i.e.,*

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\hat{\mathcal{L}}_{off}(\theta)\right] = \frac{\partial}{\partial\theta}\mathcal{L}_{off}(\Sigma(\theta))$$

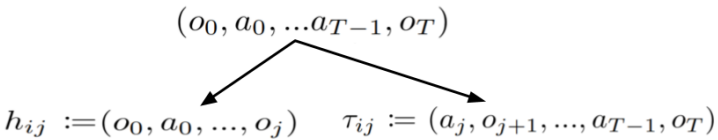
Representation Learning: Partial Observability

Use trajectories of consecutive states as input instead of individual states, it should compensate the lack of information.



Representation Learning: Partial Observability

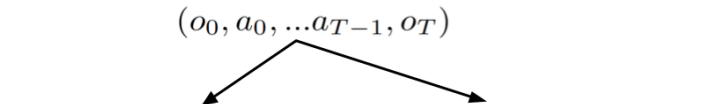
Use trajectories of consecutive states as input instead of individual states, it should compensate the lack of information.



$$\Sigma(\theta) := \frac{1}{nT} \sum_{i=1}^{n} \sum_{j=0}^{T-1} f_{\theta_1}(h_{ij}) g_{\theta_2}(\tau_{ij})^\top$$

Representation Learning: Partial Observability

Use trajectories of consecutive states as input instead of individual states, it should compensate the lack of information.

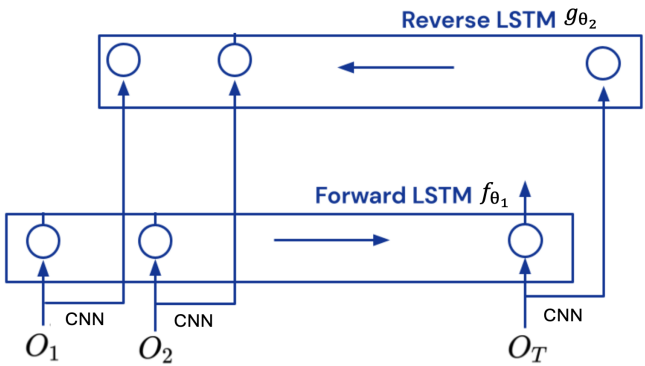


$$\Sigma(\theta) := \frac{1}{nT} \sum_{i=1}^{n} \sum_{j=0}^{T-1} f_{\theta_1}(h_{ij}) g_{\theta_2}(\tau_{ij})^\top$$

- After a state-processing network, an LSTM is used to produce the representation of a trajectory.

Representation Learning: Partial Observability

$$\Sigma(\theta) := \frac{1}{nT} \sum_{i=1}^{n} \sum_{j=0}^{T-1} f_{\theta_1}(h_{ij}) g_{\theta_2}(\tau_{ij})^\top$$



Exploration Bonus: Estimation of state visitation frequency

Let the decomposition of $DP_\pi = U\Sigma V^T$ and f_U the function parameterized with the left singular values U. Let $\Lambda = \Sigma^2$.

Theorem

If $P^T P_\pi$ and D are invertible, then for $\alpha_s := \left(P^T P_\pi\right)^{-1}_{[s,s]}$,

$$\|f_U(x_s)\|^2_{\Lambda^{-1}} = \frac{\alpha_s}{d(s)^2}.$$

- *The norms of the representations can be used to indicate pseudo-counts*
- *Estimating the bonus only requires computing the norm*

Exploration Bonus: Visualization

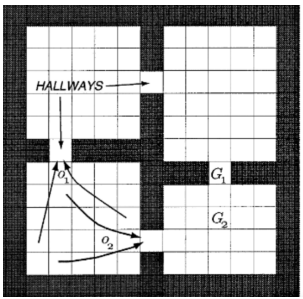


Figure: 4-room environment

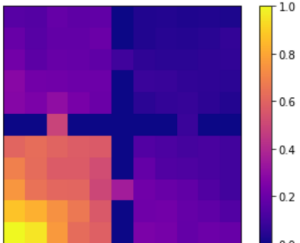


Figure: Visit frequency with a random policy

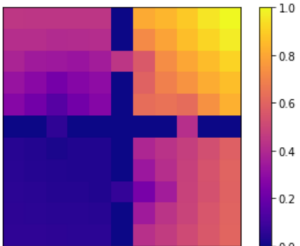


Figure: Bonus constructed using the learned representation

Criticism: Mathematics

Criticism: Mathematics

The authors tend to:

- remove important constraints
- mislead readers into believe their contributions are sound

.

Criticism: Flaw in Exploration Bonuses

Criticism: Flaw in Exploration Bonuses

Reminder of Pseudo-count Theorem:

Theorem

If $P^T P_\pi$ and D are invertible, then for $\alpha_s := \left(P^T P_\pi\right)^{-1}_{[s,s]}$,

$$\|f_U(x_s)\|^2_{\Lambda^{-1}} = \frac{\alpha_s}{d(s)^2}.$$

Criticism: Flaw in Exploration Bonuses

Reminder of Pseudo-count Theorem:

Theorem

If $P^T P_\pi$ and D are invertible, then for $\alpha_s := \left(P^T P_\pi\right)^{-1}_{[s,s]}$,

$$\|f_U(x_s)\|^2_{\Lambda^{-1}} = \frac{\alpha_s}{d(s)^2}.$$

Useless if α_s varies a lot

1. Why call it "state-dependant constant"?
2. No mathematical analyses to check the behavior of α_s

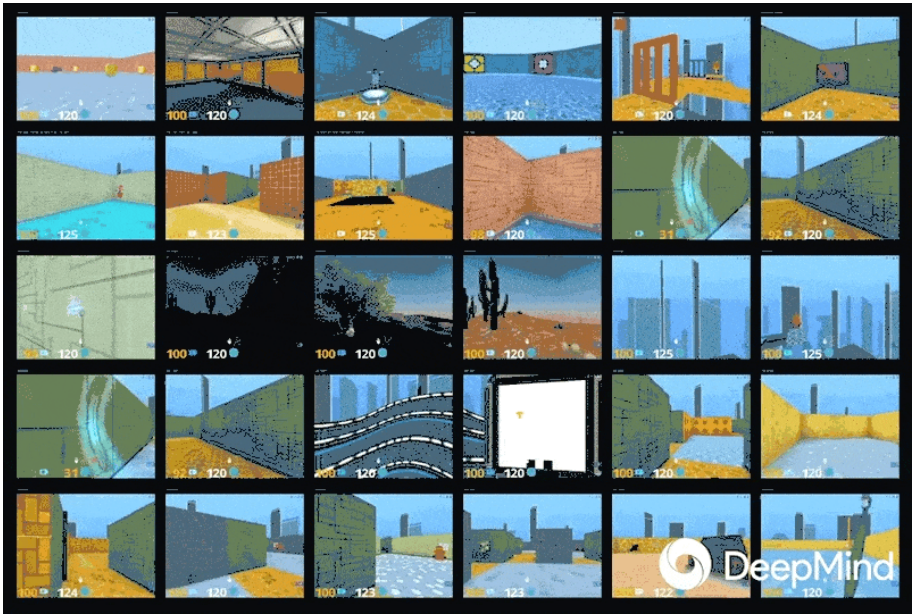
Criticism: Flaw in Function approximation setting

Using **batch-normalization is a bad way** to replace the constraint $\hat{U}^\top \hat{U} = \hat{V}^\top \hat{V} = I$ in function approximation:

1. This normalization is not w.r.t the Euclidean norm.
2. It disregards the orthogonality condition on the columns of $\hat{U}$ and $\hat{V}$.

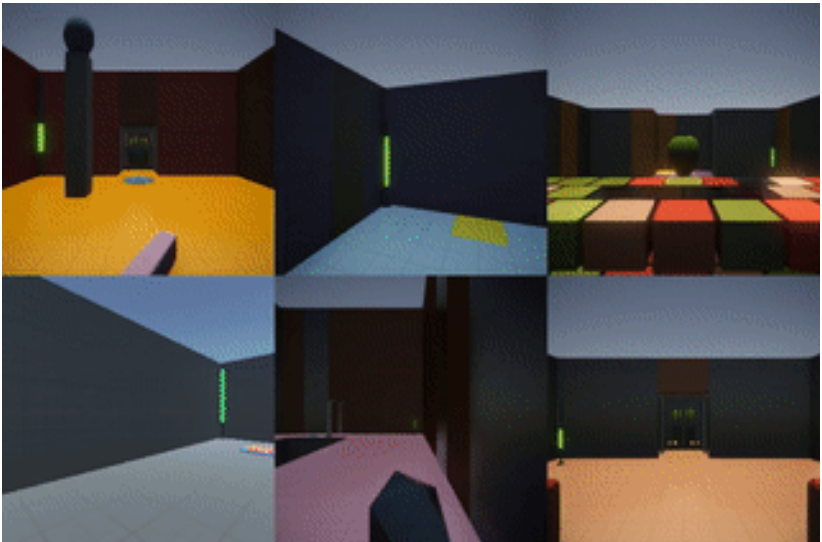
Experiments

Experiments: Environments (DM-Lab-30)



- Partial observability: Challenge for model-free approach

Experiments: Environments (DM-Hard-8)



- Partial observability
- Sparse reward: serves as exploration test bed
- Highly variable initial conditions every episode

Experiments: Baselines

Experiments: Baselines

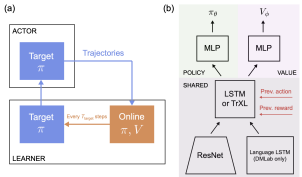


Figure: **V-MPO**: On-Policy MAP Policy Optimization (DeepMind (2019)):

- Approximate policy iteration algo.
- No exploration bonus
- No auxiliary loss (in this experiment)

Experiments: Baselines

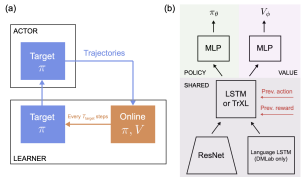


Figure: **V-MPO**: On-Policy MAP Policy Optimization (DeepMind (2019)):

- Approximate policy iteration algo.
- No exploration bonus
- No auxiliary loss (in this experiment)

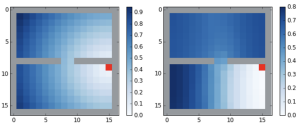


Figure: **The Laplacian in RL** (Yifan et al. (2018)). Visualization of the shaped reward defined by the L2 distance (left) and Laplacian representation (right):

- Combined with V-MPO
- No exploration bonus
- Auxiliary loss

Experiments: Baselines

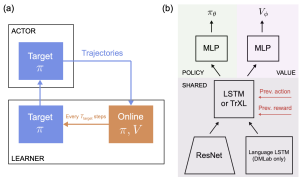


Figure: **V-MPO**: On-Policy MAP Policy Optimization (DeepMind (2019)):

- Approximate policy iteration algo.
- No exploration bonus
- No auxiliary loss (in this experiment)

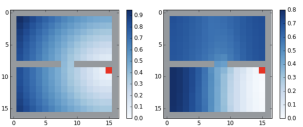


Figure: **The Laplacian in RL** (Yifan et al. (2018)). Visualization of the shaped reward defined by the L2 distance (left) and Laplacian representation (right):

- Combined with V-MPO
- No exploration bonus
- Auxiliary loss

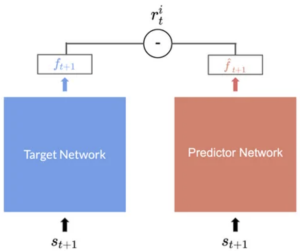


Figure: Exploration by **Random Network Distillation** (OpenAI (2018)):

- Combined with V-MPO
- Exploration bonus
- No auxiliary loss

Experiments: Results

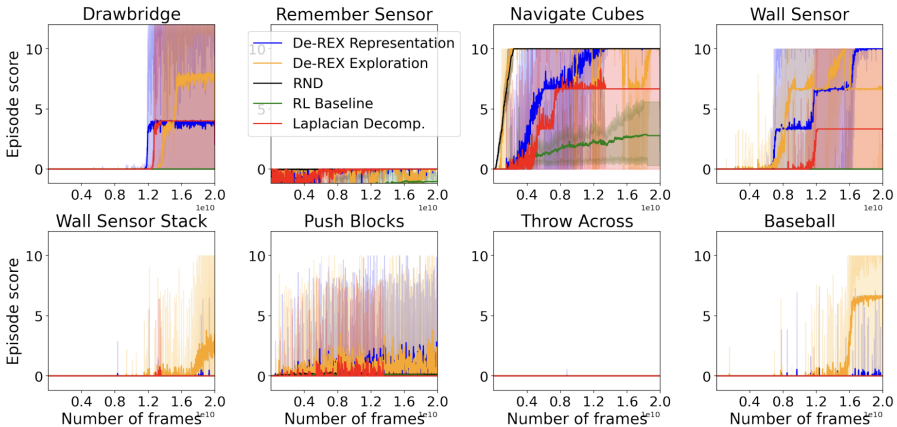


Figure: Learning curves for each task on DM-Hard-8. The shaded area corresponds to the minimum and maximum across three seeds.

Criticism: Experiments

Criticism: Experiments

Again, in order to mislead readers, the authors deliberately:

Criticism: Experiments

Again, in order to mislead readers, the authors deliberately:

- Compare against a worse modified version of the V-MPO baseline

Criticism: Experiments

Again, in order to mislead readers, the authors deliberately:

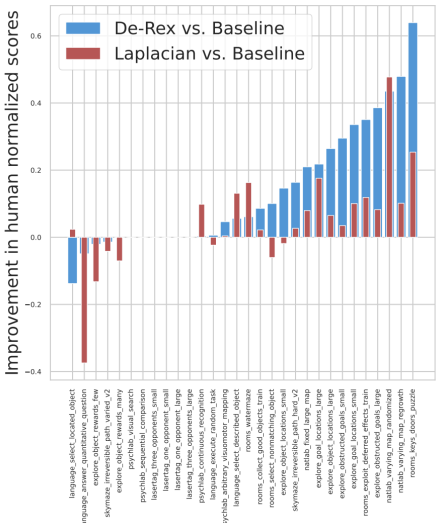
- Compare against a worse modified version of the V-MPO baseline
- Omit mentioning relevant work, even from the same co-author.

Criticism: Experiments with worse V-MPO baseline

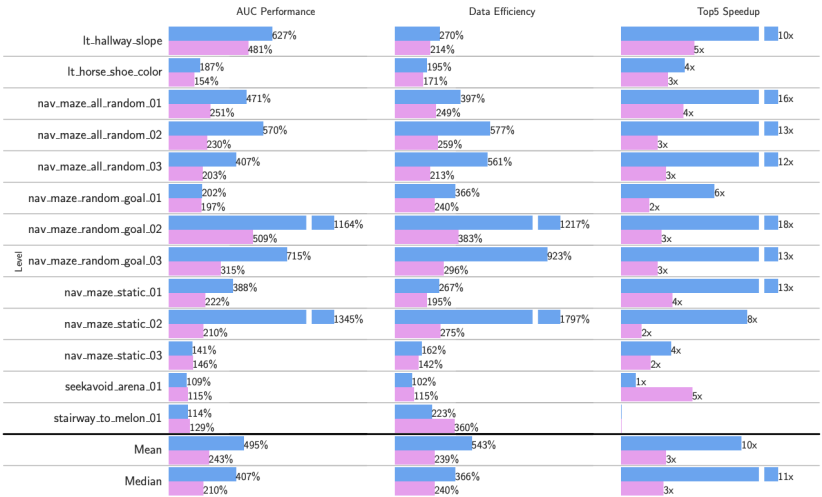
- The authors avoid mentioning the representation loss of the original VMPO.

Criticism: Experiments with worse V-MPO baseline

- The authors avoid mentioning the representation loss of the original VMPO.
- It is not until the appendix that it is mentioned, saying they did not use **pixel control** to "avoid confounding effects".



Appendix: Pixel control - performance boost



Criticism: Experiments

Criticism: Experiments

- They omit mentioning *"BYOL-Explore: Exploration by Bootstrapped Prediction"*.

Criticism: Experiments

- They omit mentioning *"BYOL-Explore: Exploration by Bootstrapped Prediction"*.
- Although, Zhaohan Daniel Guo is the **author/co-author of both paper**.

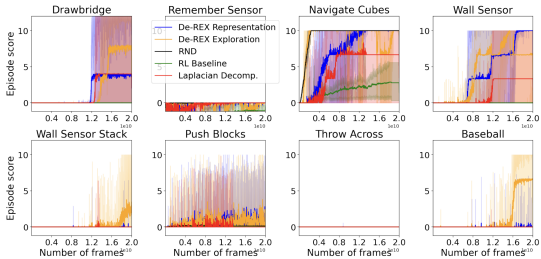


Figure: De-Rex (this paper) on DM-Hard-8

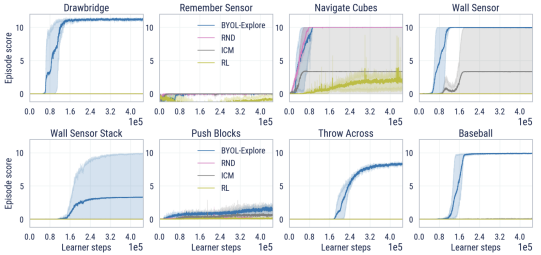


Figure: BYOL-Explore on DM-Hard-8

Discussion and Further work

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

- **Simplicity:** model-free, "pseudo-exploration bonus"

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

- **Simplicity:** model-free, "pseudo-exploration bonus"
- **Theory results:** theoretically sound in the Tabular Setting

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

- **Simplicity:** model-free, "pseudo-exploration bonus"
- **Theory results:** theoretically sound in the Tabular Setting
- **Extensibility:** compatibility with any RL algorithm

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

- **Simplicity:** model-free, "pseudo-exploration bonus"
- **Theory results:** theoretically sound in the Tabular Setting
- **Extensibility:** compatibility with any RL algorithm

Further work:

- Theoretical: Analyse the Function approximation setting

Discussion and Further work

Despite some flaws, the paper proposed an SVD-based representation and exploration procedure with:

- **Simplicity:** model-free, "pseudo-exploration bonus"
- **Theory results:** theoretically sound in the Tabular Setting
- **Extensibility:** compatibility with any RL algorithm

Further work:

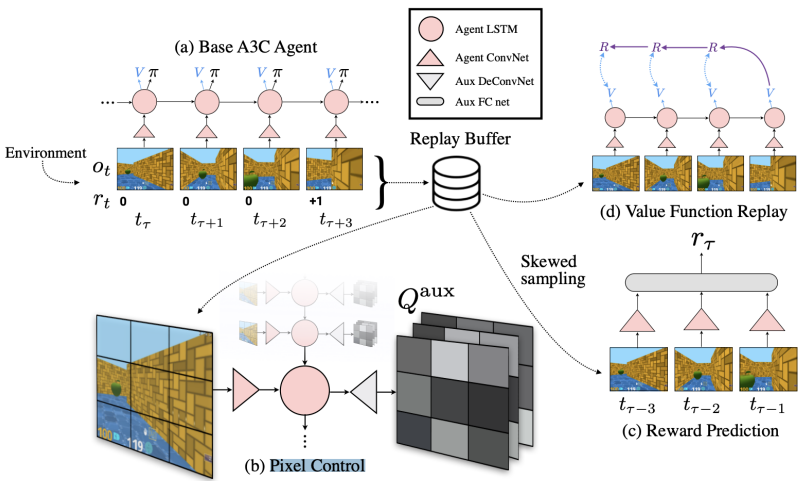
- Theoretical: Analyse the Function approximation setting
- Empirical: Minimize variance and conduct comprehensive comparisons with SoTA methods to ensure a balanced evaluation

Thank you for your attention

Appendix

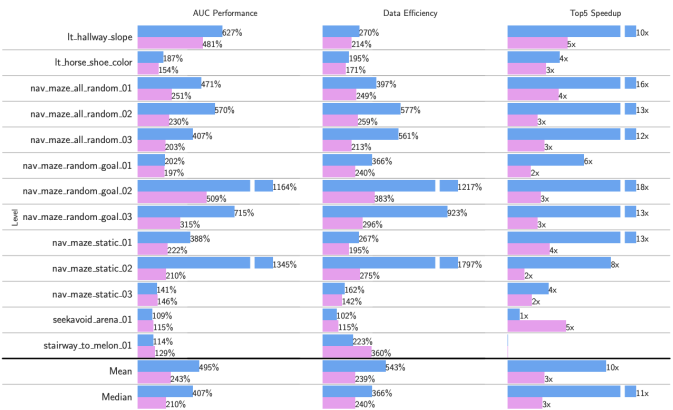
Pixel control (representation learning auxiliary loss)

Pixel Control - auxiliary policies Q^{aux} are trained to maximise change in pixel intensity of different regions of the input. The agent CNN and LSTM are used for this task along with an auxiliary deconvolution network. This auxiliary control task requires the agent to learn how to control the environment.



Pixel control - performance boost

A breakdown of the improvement over A3C for each level on Labyrinth. The values for $\mathrm{A3C} + \mathrm{RP} + \mathrm{VR}$ (reward prediction and value function replay) and UNREAL (reward prediction, value function replay and pixel control) are normalised by the A3C value. AUC Performance gives the robustness to hyperparameters (area under the robustness curve).



Proof of Theorem 1

Proof.

Let $v \in \mathbb{R}^{|S|\times 1}$ be an eigenvector of P_π and λ be its associated eigenvalue

$$P_\pi v = \lambda v.$$

By applying Neuman series, it can be seen that the eigenvectors for P_π and $(I - \gamma P_\pi)^{-1}$ are the same:

$$(I - A)^{-1} = \sum_{n=0}^{\infty} A^n$$

$$(I - \gamma P_\pi)^{-1} v = \left(\sum_{t=0}^{\infty} \gamma^t P_\pi^t \right) v = \sum_{t=0}^{\infty} (\gamma\lambda)^t v.$$

□

Proof of Theorem 1

Proof.

Further, as $0 \leq \gamma < 1$ and $|\lambda| \leq 1$,

$$(I - \gamma P_\pi)^{-1} v = \frac{1}{1 - \gamma\lambda} v$$

$$\lambda' = \frac{1}{1 - \gamma\lambda}$$

$$(I - \gamma P_\pi)^{-1} v = \lambda' v$$

Proof of Theorem 2

Proof.

$$\mathbb{E}[A_n] = \mathbb{E}\left[\frac{1}{n}X^T Y\right] = \mathbb{E}\left[\frac{1}{n}X^T E[Y|X]\right].$$

Proof of Theorem 2

Proof.

$$\mathbb{E}[A_n] = \mathbb{E}\left[\frac{1}{n}X^T Y\right] = \mathbb{E}\left[\frac{1}{n}X^T E[Y|X]\right].$$

$$E[Y|X] = XP_\pi$$

Proof of Theorem 2

Proof.

$$\mathbb{E}[A_n] = \mathbb{E}\left[\frac{1}{n}X^T Y\right] = \mathbb{E}\left[\frac{1}{n}X^T E[Y|X]\right].$$

$$E[Y|X] = XP_\pi$$

$$\mathbb{E}[A_n] \overset{(a)}{=} \mathbb{E}\left[\frac{1}{n}X^T X P_\pi\right] = \mathbb{E}\left[\frac{1}{n}X^T X\right] P_\pi.$$

Proof of Theorem 2

Proof.

$$\mathbb{E}[A_n] = \mathbb{E}\left[\frac{1}{n}X^T Y\right] = \mathbb{E}\left[\frac{1}{n}X^T E[Y|X]\right].$$

$$E[Y|X] = XP_\pi$$

$$\mathbb{E}[A_n] \overset{(a)}{=} \mathbb{E}\left[\frac{1}{n}X^T XP_\pi\right] = \mathbb{E}\left[\frac{1}{n}X^T X\right]P_\pi.$$

Because of one-hot encoding, $\frac{1}{n}X^T X$ is a diagonal matrix. Therefore:

$$\mathbb{E}\left[\frac{1}{n}X^T X\right]_{[i,j]} = \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n} x_i x_j^T\right]_{[i,j]}$$

Proof of Theorem 2

Proof.

$$\mathbb{E}[A_n] = \mathbb{E}\left[\frac{1}{n}X^T Y\right] = \mathbb{E}\left[\frac{1}{n}X^T E[Y|X]\right].$$

$$E[Y|X] = XP_\pi$$

$$\mathbb{E}[A_n] \overset{(a)}{=} \mathbb{E}\left[\frac{1}{n}X^T X P_\pi\right] = \mathbb{E}\left[\frac{1}{n}X^T X\right] P_\pi.$$

Because of one-hot encoding, $\frac{1}{n}X^T X$ is a diagonal matrix. Therefore:

$$\mathbb{E}\left[\frac{1}{n}X^T X\right]_{[i,j]} = \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n} x_i x_j^T\right]_{[i,j]}$$

$$= \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n}\{s_j = s_i \wedge s_j = s_j\}\right]$$

□

Proof of Theorem 2

Proof.

$$= \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n}\{s_j = s_i \wedge s_j = s_j\}\right]$$

$$= \mathbb{E}[\{s_j = s_i\}]$$

$$= d(s_i).$$

Therefore, $\mathbb{E}\left[\frac{1}{n}X^T X\right] = D$, and:

$$\mathbb{E}[A_n] = DP_\pi.$$

□

Proof of Theorem 4

Proof.

We rewrite $\Sigma(\theta) := \frac{1}{n}\sum_{i=1}^{n} f_{\theta_1}(s_i) g_{\theta_2}(s_i')^\top \in \mathbb{R}^{k\times k}$ as:

$$\Sigma(\theta) := \mathbb{E}\left[f_{\theta_1}(S) g_{\theta_2}(S')^\top\right]$$

We will also use:

$$\mathcal{L}_{\text{off}}(\Sigma(\theta)) := \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k} \Sigma(\theta)_{[i,j]}^2 \tag{4}$$

$$\Sigma(\theta) := \frac{1}{b}\sum_{i=1}^{b} f_{\theta_1}(s_i) g_{\theta_2}(s_i')^\top \tag{5}$$

□

Proof.

By applying (4) and (5):

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial}{\partial\theta} L_{\text{diag}}\left(\Sigma(\theta)\right)\right] &= \mathbb{E}\left[\frac{1}{k}\sum_{i=1}^{k}\frac{\partial}{\partial\theta}\Sigma(\theta)_{i,i}\right] \\
&\overset{(a)}{=} \frac{1}{k}\sum_{i=1}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\Sigma(\theta)_{i,i}\right] \\
&= \frac{1}{k}\sum_{i=1}^{k}\frac{\partial}{\partial\theta}\Sigma(\theta)_{i,i} \\
&= \frac{\partial}{\partial\theta} L_{\text{diag}}\left(\Sigma(\theta)\right),
\end{aligned}$$

(a) Is the case where the sampling distribution is fixed

□

Proof of Theorem 4

Proof.

$$
\begin{aligned}
\mathbb{E}\left[\frac{\partial}{\partial\theta}\,\mathcal{L}_{\text{off}}\left(\frac{1}{b}\Sigma(\theta)\right)\right] &= \mathbb{E}\left[\frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\left(\frac{1}{b}\Sigma(\theta)^2_{[i,j]}\right)\right], \\
&= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(\frac{1}{b}\Sigma(\theta)^2_{[i,j]}\right)\right], \\
&= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(f_{\theta_1}(S)g_{\theta_2}(S')^\top_{[i,j]}\right)^2\right], \\
&\neq \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\left(f_{\theta_1}(S)g_{\theta_2}(S')^\top_{[i,j]}\right)\right]^2, \\
&= \frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\Sigma(\theta))
\end{aligned}
$$

Proof of Theorem 5

Proof.

$$\hat{\mathcal{L}}_{\text{off}}(\theta) = \frac{1}{k^2 - k} \sum_{\substack{i,j=1 \\ i \neq j}}^{k} \hat{\Sigma}(\theta)_{[i,j]} sg\left(\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right) + \sum_{\substack{i,j=1 \\ i \neq j}}^{k} \hat{\hat{\Sigma}}(\theta)_{[i,j]} sg\left(\hat{\Sigma}(\theta)_{[i,j]}\right)$$

We note the first and the second element of the sum with *I* and *II* respectively:

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\theta)\right] = \mathbb{E}\left[\frac{\partial}{\partial\theta}(\text{I})\right] + \mathbb{E}\left[\frac{\partial}{\partial\theta}(\text{II})\right]$$

□

Proof of Theorem 5

$$
\begin{aligned}
\mathbb{E}\left[\frac{\partial}{\partial\theta}(\mathsf{I})\right] &= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbb{E}\left[\frac{\partial}{\partial\theta}\hat{\Sigma}(\theta)_{[i,j]}sg\left(\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right)\right], \\
&= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\mathbb{E}\left[\frac{\partial}{\partial\theta}\hat{\Sigma}(\theta)_{[i,j]}\right]\mathbb{E}\left[\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right], \\
&= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\mathbb{E}\left[\hat{\Sigma}(\theta)_{[i,j]}\right]\mathbb{E}\left[\hat{\hat{\Sigma}}(\theta)_{[i,j]}\right], \\
&= \frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\frac{\partial}{\partial\theta}\Sigma(\theta)_{[i,j]}\Sigma(\theta)_{[i,j]}, \\
&= \frac{1}{2}\frac{\partial}{\partial\theta}\left(\frac{1}{k^2-k}\sum_{\substack{i,j=1\\ i\neq j}}^{k}\Sigma(\theta)^2_{[i,j]}\right), \\
&= \frac{1}{2}\frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\Sigma(\theta))
\end{aligned}
$$

Proof of Theorem 5

Proof.

It can be observed that $\mathbb{E}\left[\frac{\partial}{\partial\theta} I)\right] = \mathbb{E}\left[\frac{\partial}{\partial\theta} II\right]$. Therefore:

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\theta)\right] &= \mathbb{E}\left[\frac{\partial}{\partial\theta}(\text{I})\right] + \mathbb{E}\left[\frac{\partial}{\partial\theta}(\text{II})\right], \\
&= 2 \cdot \mathbb{E}\left[\frac{\partial}{\partial\theta}(\text{I})\right] \\
&= \frac{\partial}{\partial\theta}\mathcal{L}_{\text{off}}(\Sigma(\theta))
\end{aligned}$$

□