Learning Generalized Representations for Reinforcement Learning

DA7400 - End Sem Project Report

Nandhakishore C S (DA24M011)

Rohit Kumar (DA24S003)

Rudra Sarkar (DA24D008)

Department of Data Science and Artificial Intelligence
Indian Institute of Technology Madras

Chennai – 600036, India

Submitted as part of the course
Reinforcement Learning (DA7400), November 23, 2025

Abstract

In reinforcement learning (RL), agents operating in large, complex environments face significant challenges in learning optimal policies due to the curse of dimensionality. When the state space is exponentially large or continuous, traditional tabular methods become intractable, and even function approximation methods can struggle with sample efficiency. This project explores methods for learning generalized state representations that abstract away irrelevant information, enabling more efficient and scalable learning.

We propose a lightweight representation learning objective based on a surrogate for the on-policy bisimulation metric, which aims to group states with similar future outcomes under the current policy. The key insight is to leverage the value function as a lower bound on the bisimulation distance, thereby avoiding the need to explicitly model transition dynamics. We explore two parameterizations: a simple linear model and an energy-based embedding function.

We analyze the theoretical properties of our proposed loss functions, demonstrating their optimizational advantages like Lipschitz continuity and smoothness. These properties ensure bounded gradients and stable optimization, while also providing guarantees on convergence rates. We prove that the linear parameterization is $4\sqrt{2}l$-Lipschitz and β-smooth, and that the energy parameterization is $8l^2$-Lipschitz. These results establish that our objectives have favorable optimization landscapes.

Finally, we present comprehensive empirical results across several classic control environments like CartPole-v0, CartPole-v1, Acrobot-v1, and LunarLander-v2. We validate the effectiveness of our approach by analyzing the impact of different modeling choices including linear versus energy-based embeddings, the effect of the embedding loss coefficient λ, and logarithmic transformations of the distance metric. Our experiments show that energy embeddings consistently outperform linear ones, and that a moderate value of λ (between 0.2 and 0.8) provides the best balance between representation learning and task performance. Visualization of the learned representations using PCA confirms that our method successfully clusters states by value, creating meaningful abstractions.

Contents

List of Figures

List of Tables

List of Symbols

Symbol	Description
$\mathcal{M}$	Markov Decision Process (MDP)
$\mathcal{X}$	State space
$\mathcal{A}$	Action space
$\mathcal{P}$	State transition function
$\mathcal{R}$	Reward function
γ	Discount factor
π	Policy
π^*	Optimal policy
$V^\pi(x)$	Value function for state x under policy π
$Q^\pi(x, a)$	Action-value function under policy π
U	Set of state variables
U^*	Subset of relevant state variables
$[x^U]$	Block of states agreeing on variables U
B^π	On-policy bisimulation equivalence
$\tilde{B}^\pi$	Value-based equivalence
$d(x_1, x_2)$	Bisimulation distance between states
W_1	Wasserstein-1 distance
ϕ	Representation/embedding function
θ	Parameters of embedding function
ω	Parameters of value function
ψ	Parameters of policy
$\mathcal{L}$	Loss function
$\mathcal{L}_c$	Critic loss (Bellman error)
$\mathcal{L}_e$	Embedding loss
$\mathcal{L}_a$	Actor loss
λ	Embedding loss coefficient
c	Scaling constant, $c = 1 - \gamma$
l	Upper bound on embedding norm
m	Lower bound on embedding difference norm
N	Number of states $\lvert\mathcal{X}\rvert$
d	Embedding dimension
W	Weight matrix for embedding
$\mathcal{B}$	Replay buffer
η_c	Learning rate for critic
η_a	Learning rate for actor

1 Introduction

1.1 Motivation and Problem Statement

Reinforcement learning has emerged as a powerful paradigm for sequential decision-making, with applications ranging from game playing (Mnih et al. 2015) to robotics and autonomous systems. The standard framework for RL is the Markov Decision Process (MDP), formally defined by the tuple $\mathcal{M} = \langle \mathcal{X}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where:

- $\mathcal{X}$ is the state space, representing all possible configurations of the environment
- $\mathcal{A}$ is the action space, representing all available actions the agent can take
- $\mathcal{P} : \mathcal{X} \times \mathcal{A} \to \Delta(\mathcal{X})$ is the state transition function, defining the probabilistic dynamics
- $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is the reward function
- $\gamma \in [0, 1)$ is the discount factor

The agent's goal is to find a policy $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ that maximizes the expected discounted cumulative reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{s \sim p_0(s)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, \pi \right]$$

where p_0 is the initial state distribution.

However, when the state space $\mathcal{X}$ is very large or continuous, this optimization problem becomes extraordinarily challenging. The cardinality of the state space $|\mathcal{X}|$ can be exponentially large, and in order to guarantee true optimality, the agent would need to visit each state infinitely often—an impossible requirement in practice.

A fundamental observation motivates our work: *Should an agent at timestep t consider the full state x_t to make an optimal decision?* Intuitively, the answer is no. Conditioned on a specific goal, a human acting under an optimal policy would naturally ignore irrelevant factors in the environment as noise and focus only on the primary factors that drive the intended goal.

Consider a robot navigating through a room: the exact positions of objects far from its path may be irrelevant for determining the optimal action. Similarly, in a video game, background elements that don't affect gameplay are noise from the perspective of decision-making. For a given goal or task, many components of the state may be entirely irrelevant.

This insight motivates the central question of this work: *Can we learn compact representations of states that preserve only the information relevant for optimal decision-making, thereby enabling more efficient learning?*

1.2 Factored State Representations

To formalize this idea, we adopt a factored representation of the state space. We view the state space as essentially all the admissible configurations of a set of underlying state variables $U = \{u_1, u_2, \ldots, u_n\}$. Thus, each state $x_t \in \mathcal{X}$ is a particular configuration of U, i.e., $x_t = \{u_1, \ldots, u_n\}^t$. Therefore:

$$\mathcal{X} = \{(u_1, \ldots, u_n)^1, (u_1, \ldots, u_n)^2, \ldots\}$$

This way of representing the states in terms of the state variables is also called a factored representation (Givan, Dean, and Greig 2003).

Definition 1.1 (Factored Representation and State Blocks)**.** Let X be a set and X^U be its factored representation with respect to variables U. For any state $x \in X$ with factored form $x^U = \{u_1, \ldots, u_n\}$, the notation $[x^U]$ denotes the block that comprises of states which agree upon all the variables in U:

$$[x^U] = \{x_i \in X : x_i^U = x^U\}$$

The underlying goal of representation learning is to provide the agent with the subset $U^* \subseteq U$ such that the corresponding variables serve as the sufficient factors for the maximization objective. So instead of focusing on the entire state x_t, an agent will focus on the elements of the state that are relevant for the goal.

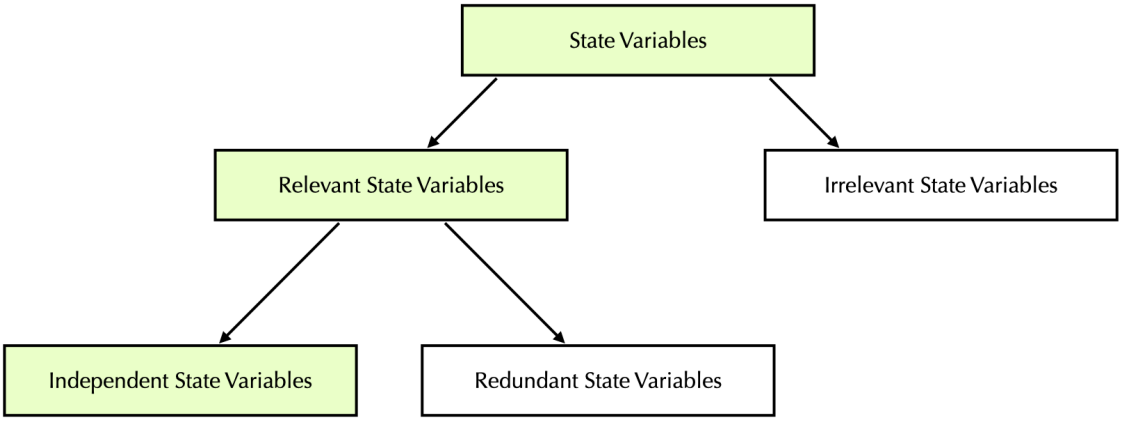


Figure 1: Illustration of the representation learning goal: identifying relevant state variables U^* that suffice for optimal decision-making.

Let $x_t^U = \{u_1, \ldots, u_n\}^t$ and let $f^* : \mathcal{X} \to [\mathcal{X}^{U^*}]$ be a function that maps states to their blocks. Therefore, if for two states x_1, x_2 we have $x_1^U \neq x_2^U$ but $x_1^{U^*} = x_2^{U^*}$, then:

$$x_1 \equiv_{f^*} x_2$$

Example 1.2. Let the factored representation of the state space be:

$$x_1 = (u_1^1, u_2^1, u_3^0)$$
$$x_2 = (u_1^0, u_2^1, u_3^1)$$
$$x_3 = (u_1^1, u_2^1, u_3^1)$$

Let $U^* = \{u_1, u_2\}$. Hence:

$$[x_1^{U^*}] = [x_3^{U^*}] = \{x_1, x_3\} \quad \text{and} \quad [x_2^{U^*}] = \{x_2\}$$

Thus:

$$[\mathcal{X}^{U^*}] = \{\{x_1, x_3\}, \{x_2\}\}$$

Hence, the goal is to find f^* that aggregates states which agree with respect to U^*. This defines a partitioning of the state space $\mathcal{X}$. This actually reduces the actual MDP to a minimized version where lifting the optimal policy from the minimized MDP to the actual MDP retains optimality.

1.3 On-Policy Bisimulation

The equivalence relation we use in our work is state bisimilarity with respect to the current policy. On-policy bisimulation renders two different states x_1, x_2 similar with respect to policy π if:

- $\mathcal{R}(x_1, \pi(x_1)) = \mathcal{R}(x_2, \pi(x_2))$
- $\mathcal{P}(x_1, \pi(x_1)) = \mathcal{P}(x_2, \pi(x_2))$

Definition 1.3 (On-Policy Bisimulation)**.** Let $B^\pi : \mathcal{X} \to \mathcal{X}_{B^\pi}$ be the function that maps states to their associated blocks in $\mathcal{X}$ partitioned by B^π. Two states x_1, x_2 are bisimilar under the current policy π if and only if:

$$x_1 \equiv_{B^\pi} x_2 \Leftrightarrow \mathcal{R}(x_1, \pi(x_1)) = \mathcal{R}(x_2, \pi(x_2)) \text{ and } \mathcal{P}(x_1, \pi(x_1)) = \mathcal{P}(x_2, \pi(x_2))$$

1.4 Concrete Problem Statement

Problem Statement:

We seek representation learning methods that:

1. Approximate the on-policy bisimulation metric and therefore render bisimilar states as equivalent.
2. Provide lightweight models that reduce computational overhead.
3. Possess favorable theoretical properties (Lipschitz continuity, smoothness) that guarantee stable optimization.

2 Background and Related Work

2.1 Bisimulation Metrics

The foundational work on bisimulation metrics for MDPs is due to Ferns, Panangaden, and Precup (2004), who introduced a formal metric to quantify the behavioral similarity between states. They formulated bisimilarity for two states x_1, x_2 as:

$$d(x_1, x_2) = \max_{a \in \mathcal{A}} \left[(1-\gamma)|r(x_1, a) - r(x_2, a)| + \gamma W_1(\mathcal{P}(\cdot|x_1, a), \mathcal{P}(\cdot|x_2, a)) \right]$$

where W_1 is the Wasserstein-1 distance between the transition distributions.

They also showed in one of their key results that:

$$(1-\gamma)|V_k(x_1) - V_k(x_2)| \leq d_k(x_1, x_2)$$

This shows that the value function difference provides a lower bound on the bisimulation distance, justifying the use of value similarity as a surrogate for bisimulation.

However, the resulting partitions were considered **pessimistic** by Castro and Precup (2010), since similarity was enforced across *all* actions. In practice, two states might behave similarly under an **optimal policy**, even if their behavior differs under suboptimal or random actions.

2.2 On-Policy Bisimulation

Castro (2020) addressed this limitation by proposing an **on-policy bisimulation metric**, weighting actions by the policy's action probabilities to relax the strict aggregation criterion. This allows for more aggressive state abstraction since states only need to behave similarly under actions the policy is likely to take.

2.3 Learning Bisimulation Representations

Ferns, Panangaden, and Precup (2004) defined a bisimulation metric using the **Wasserstein-1 distance** to capture distributional differences in transitions. Castro (2020) proposed a **differentiable bisimulation metric**, learning a neural distance function that approximates the theoretical metric.

Zhang et al. (2020) bypassed explicit metric learning by mapping states into a latent space with an L_1 norm. They modeled the transition dynamics with a **Gaussian distribution** and optimized representations such that the L_1 distance between latent embeddings approximates the bisimulation distance.

2.4 DeepMDP and Connections to Bisimulation

Gelada et al. (2019) proposed learning a latent-space model of the MDP trained with two objectives:

1. Predicting rewards
2. Predicting the distribution of next latent states

When the transition loss is measured using a **Wasserstein metric**, the DeepMDP objective aligns closely with the bisimulation metric formulation, establishing a theoretical connection between the two.

3 Proposed Work

3.1 Surrogate Objective

We bootstrap from the result of Ferns, Panangaden, and Precup (2004) and use the equivalence function $\tilde{B}^\pi : \mathcal{X} \to \mathcal{X}_{V^\pi}$. Two states x_1, x_2 are rendered equivalent if:

$$x_1 \equiv_{\tilde{B}^\pi} x_2 \Leftrightarrow V^\pi(x_1) = V^\pi(x_2)$$

This makes the computation simpler, as we don't have to explicitly model the distributions for the transition dynamics.

Thus, we aim to learn a representation function $\phi : \mathcal{X} \to \mathbb{R}^d$ such that the Euclidean norm between two states in the representative space approximates their value difference:

$$\|\phi(x_1) - \phi(x_2)\| \approx c|V^\pi(x_1) - V^\pi(x_2)|$$

where $c = (1 - \gamma)$.

We seek lightweight representative networks ϕ that can help achieve this objective:

$$\min_\phi \mathbb{E}_{x,y\sim p(x,y)} \left[(\|\phi(x) - \phi(y)\| - c|V^\pi(x) - V^\pi(y)|)^2 \right]$$

3.2 Linear Parameterization

We first model $\phi(\cdot)$ as a linear function approximator. Assuming $|\mathcal{X}| = N$ and each $x_i \in \mathcal{X}$ is a one-hot vector with 1 as the i-th entry, i.e., $x_i = e_i$, we have $W \in \mathbb{R}^{d\times N}$, and W_i is the i-th column of W. Hence:

$$\phi(x_i) = Wx_i = W_i$$

The objective becomes:

$$\min_W \mathbb{E}_{x,y\sim p(x)p(x)} \left[\|\phi(x) - \phi(y)\| - c|V^\pi(x) - V^\pi(y)| \right]^2$$

3.3 Energy Embeddings

We also model $\phi(x) = e^{-Wx}$ (element-wise exponential). The objective translates to:

$$\min_W \mathbb{E}_{x,y\sim p(x,y)} \left[\|e^{-Wx} - e^{-Wy}\| - c|V^\pi(x) - V^\pi(y)|\right]^2$$

3.4 Algorithm

The complete training procedure is presented in Algorithm 1.

Algorithm 1: Training Procedure for Representation Learning

Input: Environment $\mathcal{M}$, hyperparameters λ, η_c, η_a, embedding dimension d
Output: Learned representation ϕ_θ, policy π_ψ, value function V_ω
Initialize parameters θ (embedding), ω (critic), ψ (actor);
Initialize target parameters $\theta^- \leftarrow \theta$, $\omega^- \leftarrow \omega$;
Initialize replay buffer $\mathcal{B} \leftarrow \emptyset$;
for *episode* $= 1, 2, \ldots$ **do**
  Reset environment, observe initial state x_0;
  for *timestep* $t = 0, 1, 2, \ldots$ **do**
    Compute embedding $\phi_\theta(x_t)$;
    Execute action $a_t \sim \pi_\psi(\cdot|x_t)$, observe (x_t, a_t, r_t, x_{t+1});
    Store transition in replay buffer $\mathcal{B}$;
    Sample batch $\{(x_i, a_i, r_i, x_i')\}_{i=1}^B \sim \mathcal{B}$;
    `// Critic Loss`
    $\mathcal{L}_c = \mathbb{E}\left[(V_\omega(x) - (r + \gamma V_{\omega^-}(x')))^2\right]$;
    `// Embedding Loss`
    $\mathcal{L}_e = \mathbb{E}_{x,y}\left[(\|\phi_\theta(x) - \phi_\theta(y)\| - c|V_\omega(x) - V_\omega(y)|)^2\right]$;
    `// Update critic and embedding`
    $(\omega, \theta) \leftarrow (\omega, \theta) - \eta_c \nabla(\mathcal{L}_c + \lambda \mathcal{L}_e)$;
    `// Actor Loss`
    $\mathcal{L}_a = -\mathbb{E}_{x\sim\mathcal{B}}[V_\omega(x)]$;
    `// Update actor`
    $\psi \leftarrow \psi - \eta_a \nabla_\psi \mathcal{L}_a$;
    if *periodic target update* **then**
      $\omega^- \leftarrow \omega$, $\theta^- \leftarrow \theta$;
    end
  end
end

4 Optimizational Advantages

4.1 Linear Function Approximator

Proposition 4.1 (Lipschitz Continuity)**.** *Given $\phi(x) = Wx$ which is linear with respect to the state vectors, and assuming the feature vectors are bounded $\|\phi(x)\| \leq l$, the function*

$$\mathcal{L}(W) = \mathbb{E}_{x,y\sim p(x)p(x)} \left[(\|Wx - Wy\| - c|V^\pi(x) - V^\pi(y)|)^2\right]$$

is Lipschitz continuous with a Lipschitz constant $4\sqrt{2}l$.

Proof. We have $\mathcal{L}(W) = \mathbb{E}_{x,y\sim p(x)p(x)} \left[\|Wx - Wy\| - c|V^\pi(x) - V^\pi(y)|\right]^2$.

Thus:

$$\nabla_W \mathcal{L}(W) = 2 \sum_{x,y} \left(\|W(x-y)\| - c|V^\pi(x) - V^\pi(y)|\right) \frac{W(x-y)(x-y)^\top}{\|W(x-y)\|} p(x,y)$$

Now, we have:

$$\begin{aligned} |\|W(x-y)\| - c|V^\pi(x) - V^\pi(y)|| &\leq \|W(x-y)\| \\ &\leq \|Wx\| + \|Wy\| \leq 2l \end{aligned}$$

Thus we can upper bound the norm of the gradient:

$$\begin{aligned} \|\nabla_W \mathcal{L}(W)\| &= \left\| 2 \sum_{x,y} \left(\|W(x-y)\| - c|V^\pi(x) - V^\pi(y)|\right) \frac{W(x-y)(x-y)^\top}{\|W(x-y)\|} p(x,y) \right\| \\ &\leq 2 \sum_{x,y} \left\| \left(\|W(x-y)\| - c|V^\pi(x) - V^\pi(y)|\right) \frac{W(x-y)(x-y)^\top}{\|W(x-y)\|} p(x,y) \right\| \\ &\leq 2 \sum_{x,y} |\|W(x-y)\| - c|V^\pi(x) - V^\pi(y)|| \frac{\|W(x-y)\|\|(x-y)\|}{\|W(x-y)\|} p(x,y) \\ &\leq 4l \sum_{x,y} \|(x-y)\| p(x,y) \leq 4\sqrt{2}l \sum_{x,y} p(x,y) = 4\sqrt{2}l \end{aligned}$$

The last inequality uses the fact that x, y are one-hot vectors, so $\|(x-y)\| \leq \sqrt{2}$. □

Proposition 4.2 (Smoothness)**.** *With a linear parameterization* $\phi(x) = Wx$ *and assuming* $\|\phi(x)\| \leq l$*, the function* $\mathcal{L}(W)$ *is* β*-smooth with*

$$\beta = 4 \left(1 + |r_{\max} - r_{\min}| \left(\frac{1}{m} + \frac{l\sqrt{2N}}{m^2}\right)\right)$$

where $m = \min_W \|W(x-y)\|$ *and* $N = |\mathcal{X}|$*. That is:*

$$\|\nabla_W \mathcal{L}(W) - \nabla_{W'} \mathcal{L}(W')\| \leq \beta \|W - W'\|$$

Proof. We have:

$$\begin{aligned} &\|\nabla_W \mathcal{L}(W) - \nabla_{W'} \mathcal{L}(W')\| \\ &\leq 2 \sum_{x,y} \left\| \frac{\|W(x-y)\| - c\Delta V^\pi(x,y)}{\|W(x-y)\|} W - \frac{\|W'(x-y)\| - c\Delta V^\pi(x,y)}{\|W'(x-y)\|} W' \right\| \|x-y\|^2 p(x,y) \end{aligned}$$

where $\Delta V^\pi(x,y) = V^\pi(x) - V^\pi(y)$.

This can be upper bounded as:

$$
\begin{aligned}
&\leq 4 \sum_{x,y} \left\| \frac{\|W(x-y)\| - c\Delta V^{\pi}(x,y)}{\|W(x-y)\|} W - \frac{\|W'(x-y)\| - c\Delta V^{\pi}(x,y)}{\|W'(x-y)\|} W' \right\| p(x,y) \\
&\leq 4 \sum_{x,y} \left(\|W - W'\| + c\Delta V^{\pi}(x,y) \left\| \frac{W}{\|W(x-y)\|} - \frac{W'}{\|W'(x-y)\|} \right\| \right) p(x,y) \\
&\leq 4 \sum_{x,y} \left(\|W - W'\| + c\Delta V^{\pi}(x,y) \|W - W'\| \left(\frac{1}{m} + \frac{\sqrt{2}\|W'\|}{m^2} \right) \right) p(x,y) \\
&\leq 4 \left(1 + |r_{\max} - r_{\min}| \left(\frac{1}{m} + \frac{\sqrt{2}\|W'\|}{m^2} \right) \right) \|W - W'\|
\end{aligned}
$$

Since x, y are one-hot vectors and $\|\phi(x)\| \leq l$, we have:

$$
\|W\| \leq \left(\sum_{i=1}^{N} \|W_{[i,:]}\|^2 \right)^{1/2} \leq l\sqrt{N}
$$

Therefore:

$$
\|\nabla_W \mathcal{L}(W) - \nabla_{W'} \mathcal{L}(W')\| \leq 4 \left(1 + |r_{\max} - r_{\min}| \left(\frac{1}{m} + \frac{l\sqrt{2N}}{m^2} \right) \right) \|W - W'\|
$$

□

Proposition 4.3 (Expansiveness of Update Rule)**.** *Assuming $\mathcal{L}(W)$ is β-smooth, the update rule $U_{\lambda,\mathcal{L}}(W) = W - \lambda \nabla_W \mathcal{L}(W)$ is $(1 + \lambda\beta)$-expansive.*

This proof exactly mimics the proof from Hardt, Recht, and Singer (2016)

Proof.

$$
\begin{aligned}
\|U_{\lambda,\mathcal{L}}(W) - U_{\lambda,\mathcal{L}}(W')\| &= \|W - \lambda \nabla_W \mathcal{L}(W) - W' + \lambda \nabla_{W'} \mathcal{L}(W')\| \\
&\leq \|W - W'\| + \lambda \|\nabla_W \mathcal{L}(W) - \nabla_{W'} \mathcal{L}(W')\| \\
&\leq \|W - W'\| + \lambda\beta \|W - W'\| \\
&= (1 + \lambda\beta) \|W - W'\|
\end{aligned}
$$

□

4.2 Energy Embeddings

Proposition 4.4 (Lipschitz Continuity for Energy)**.** *With an energy embedding representation function $\phi(x) = \exp(-Wx)$ and assuming the state features are bounded $\|\phi(x)\| \leq l$, the loss function*

$$
\mathcal{L}(W) = \mathbb{E}_{x,y} \left[\|\phi(x) - \phi(y)\| - c|V^{\pi}(x) - V^{\pi}(y)| \right]^2
$$

is Lipschitz continuous with a Lipschitz constant $8l^2$.

Proof. The gradient is:

$$\begin{aligned}\|\nabla_W \mathcal{L}(W)\| &= \left\| \sum_{x,y} 2(\|\phi(x)-\phi(y)\| - \Delta V^\pi(x,y)) \frac{(\phi(x)-\phi(y))\nabla_W(\phi(x)-\phi(y))}{\|\phi(x)-\phi(y)\|} p(x,y) \right\| \\ &\leq 2\sum_{x,y} |\|\phi(x)-\phi(y)\| - c\Delta V^\pi(x,y)| \|\nabla_W\phi(x) - \nabla_W\phi(y)\| p(x,y) \\ &\leq 2\sum_{x,y} \|\phi(x)-\phi(y)\| \|-x\phi(x)+y\phi(y)\| p(x,y) \\ &\leq 2\sum_{x,y} 2l \cdot 2l \cdot p(x,y) = 8l^2\end{aligned}$$

where we used $\nabla_W \phi(x) = \nabla_W e^{-Wx} = -x \odot e^{-Wx} = -x \odot \phi(x)$ and $\|\phi(x)-\phi(y)\| \leq 2l$. □

4.3 Logarithmic Transformation

We experimented with a variant (Geist, Scherrer, and Pietquin 2019) of the modified bisimulation loss where we transformed the norm of the feature difference via log. So the objective becomes:

$$\min_W \mathcal{L}(W) = \min_W \sum_{x,y} \left(\log \|\phi(x)-\phi(y)\| - \Delta V(x,y)\right)^2 p(x,y)$$

Proposition 4.5 (Lipschitz with Log Transform)**.** *With a linear parameterization $\phi(x) = Wx$, $\mathcal{L}(W)$ is Lipschitz continuous with a Lipschitz constant $\frac{2\sqrt{2}\log(2l)}{\epsilon}$. With energy parameterization $\phi(x) = \exp(-Wx)$, $\mathcal{L}(W)$ is Lipschitz continuous with a Lipschitz constant $\frac{4l\log(2l)}{\epsilon}$. We additionally assume $\|\phi(x)\| \leq l$ and $\min \|\phi(x)-\phi(y)\| = \epsilon > 0$.*

Proof for Linear Case. We have $\|\phi(x)\| \leq l$ and $\min \|\phi(x)-\phi(y)\| = \epsilon$. Therefore:

$$\nabla_W \mathcal{L}(W) = 2\sum_{x,y} \left(\log \|W(x-y)\| - \Delta V^\pi(x,y)\right) \frac{W(x-y)(x-y)^\top}{\|W(x-y)\|^2} p(x,y)$$

Upper bounding the norm of the gradient:

$$\begin{aligned}\|\nabla_W \mathcal{L}(W)\| &\leq 2\sum_{x,y} |\log \|W(x-y)\| - \Delta V^\pi(x,y)| \frac{\|x-y\|}{\|W(x-y)\|} p(x,y) \\ &\leq 2\sum_{x,y} |\log \|W(x-y)\| - \Delta V^\pi(x,y)| \frac{\sqrt{2}}{\epsilon} p(x,y) \\ &\leq \frac{2\sqrt{2}\log(2l)}{\epsilon}\end{aligned}$$

where we used the fact that $|\log \|W(x-y)\|| \leq \log(2l)$ and $|\Delta V^\pi(x,y)| \leq 2l$. □

Proof for Energy Case. For energy parameterization:

$$\nabla_W \mathcal{L}(W) = 2\sum_{x,y} \left(\log \|\phi(x)-\phi(y)\| - \Delta V^\pi(x,y)\right) \frac{(\phi(x)-\phi(y))(-x\phi(x)+y\phi(y))}{\|\phi(x)-\phi(y)\|^2} p(x,y)$$

Upper bounding:

$$\|\nabla_W \mathcal{L}(W)\| \leq 2 \sum_{x,y} |\log \|\phi(x) - \phi(y)\| - \Delta V^\pi(x,y)| \frac{\|x\phi(x) - y\phi(y)\|}{\|\phi(x) - \phi(y)\|} p(x,y)$$
$$\leq \frac{2\log(2l) \times 2l}{\epsilon} = \frac{4l\log(2l)}{\epsilon}$$

□

4.4 Implications for Optimization

To highlight the advantages of these theoretical properties:

- If $\mathcal{L}(W)$ is Lipschitz continuous, then its gradients are bounded, preventing arbitrarily large updates during optimization.
- The Lipschitz property also ensures robustness under small perturbations in the parameter or input space.
- When $\mathcal{L}(W)$ is β-smooth, choosing a step size $\alpha \leq \frac{1}{\beta}$ guarantees descent and stable convergence, even under stochastic gradients.
- For a β-smooth loss, the gradient update operator is $(1 + \alpha\beta)$-expansive, meaning it can be locally expansive in certain directions.
- In non-stationary or non-convex settings, such local expansiveness—especially along directions of negative curvature—combined with stochasticity prevents premature convergence and encourages exploration of the parameter space.

5 Results and Observations

5.1 Experimental Setup

We evaluated our method on four classic control environments from OpenAI Gym: CartPole-v0, CartPole-v1, Acrobot-v1, and LunarLander-v2. All experiments used a batch size of 256, an embedding dimension of $d = 32$, and were trained for 600 episodes. We report average returns over the last 100 episodes, averaged over 200 independently random seeds. Refer to the main branch of the github repo https://github.com/rudra-iitm-phd/da7400-project to find the working code

5.2 Effect of Embedding Loss Coefficient

Figure 2 shows that a small positive λ (e.g., 0.2 or 0.4) generally improves performance compared to $\lambda = 0$ (no embedding loss). However, very large values can sometimes hurt performance, especially in simpler environments.

5.3 Linear vs. Energy Embeddings

Figure 3 demonstrates that energy embeddings consistently outperform linear embeddings across most environments.

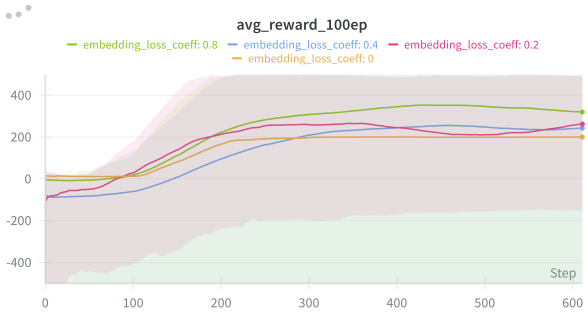


Figure 2: Average return over 100 episodes averaged over 4 games with respect to various values of the embedding loss coefficient λ. A moderate value of λ generally improves performance.

Figure 3: Average return over 100 episodes averaged over 4 games with respect to the two different types of embedding. Energy embeddings show superior performance.

Figure 4: Average return over 100 episodes averaged over 4 games comparing (a) using ℓ_2 norm of the feature difference vs. (b) using the log of the ℓ_2 norm. The effect is mixed across environments.

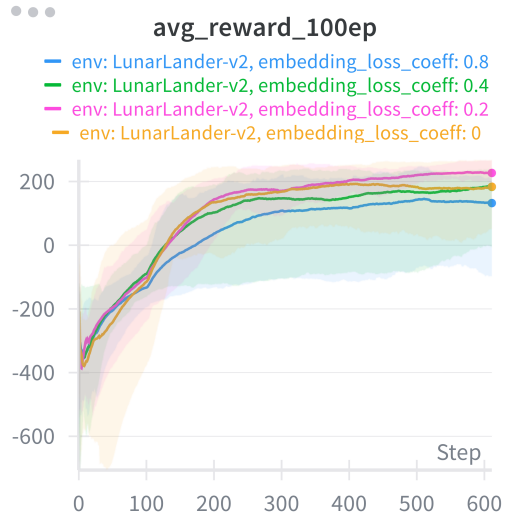
5.4 Logarithmic Transformation

Figure 4 suggests the effectiveness of transforming the feature difference with log over the linear or the vanilla setting

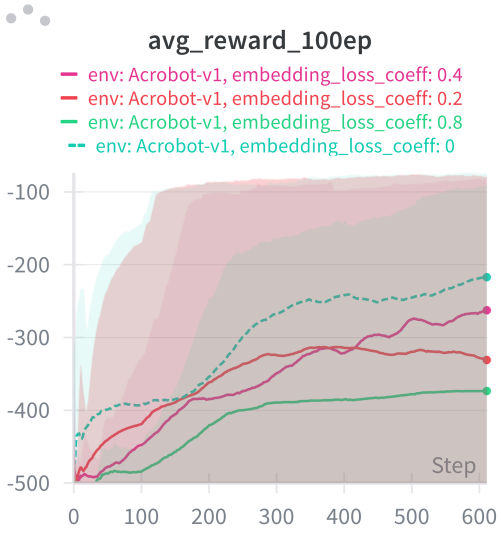
5.5 Detailed Environment-Specific Results

5.5.1 Comparative study with respect to λ

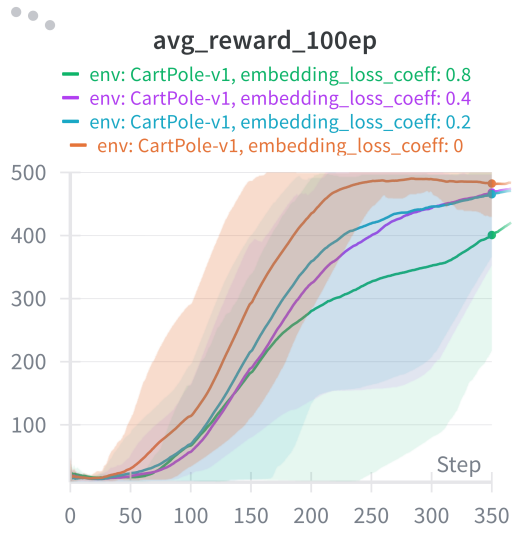
We have observed that on average $\lambda \in [0.2, 0.4]$ has a better edge for environments LunarLander-v2, CartPole-v0 and v1 while for acrobot it proves quite detrimental. The results can be referred from Figures 5a, 5b, 6a, 6b



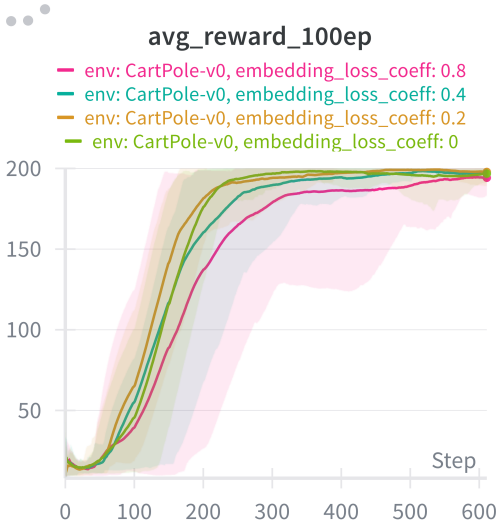
(a) LunarLander-v2: Comparative study for λ



(b) Acrobot-v1: Comparative study for λ



(a) CartPole-v1: Comparative study for λ

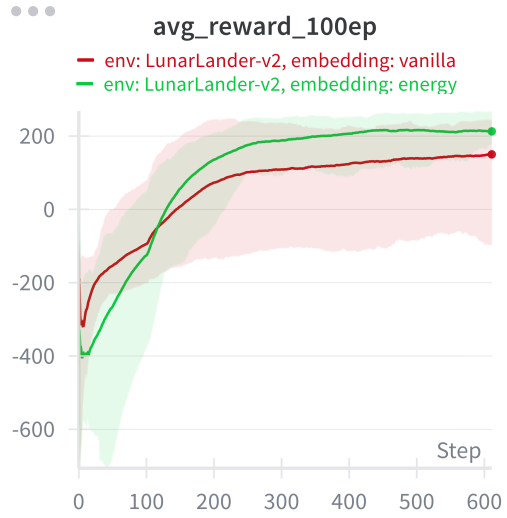


(b) CartPole-v0: Comparative study for λ

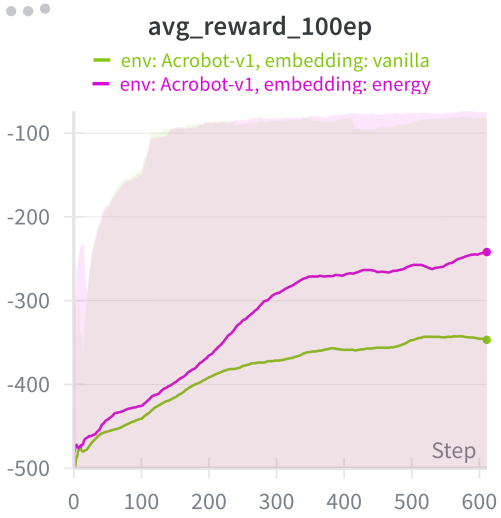
5.5.2 Comparative study with respect to Embedding strategy

On average the energy embedding strategy seemed to outperform in the environments : LunarLander-v2 and Acrobot-v1 (Figures 7a, 7b), but there is not a clear distinction between the effectiveness

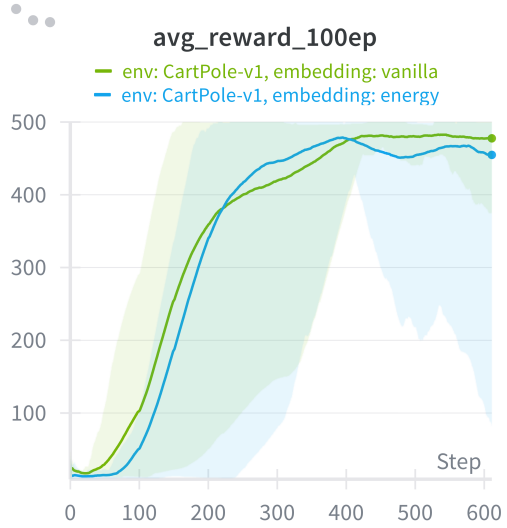
of energy embeddings from the vanilla in the Cartpole environments (Figures 8a,8b). One likely conclusion could be that for complex environments energy embeddings have a better edge while for simpler environmnets like cartpole, vanilla embedding suffices.



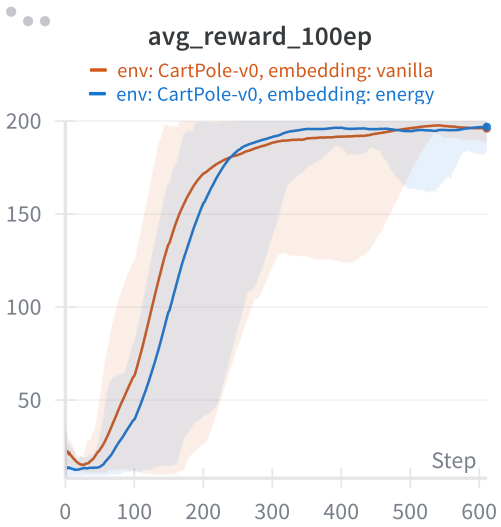
(a) LunarLander-v2: Embedding types



(b) Acrobot-v1: Embedding types



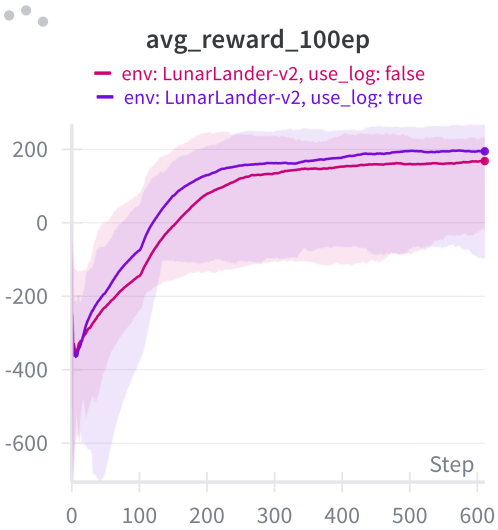
(a) CartPole-v1: Embedding types



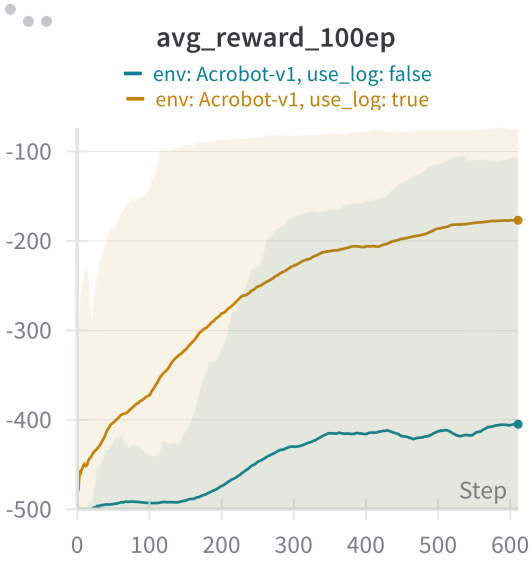
(b) CartPole-v0: Embedding types

5.5.3 Comparative study with respect to log transform

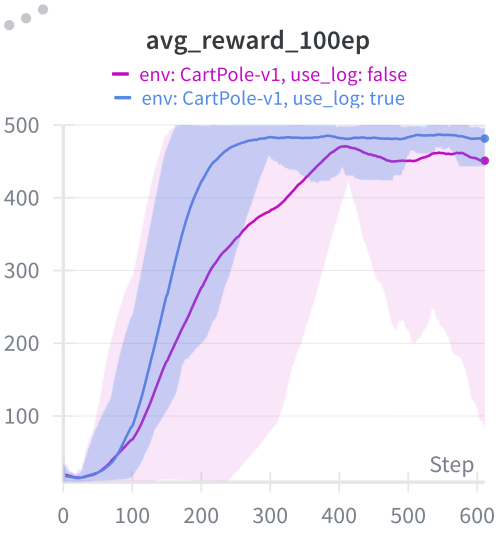
On average the transformation with log seemed to outperform the vanilla feature differences. It can be clearly seen that the transformation has proved its efficiency in all the four environments (Figures 9a,9b,10a,10b)



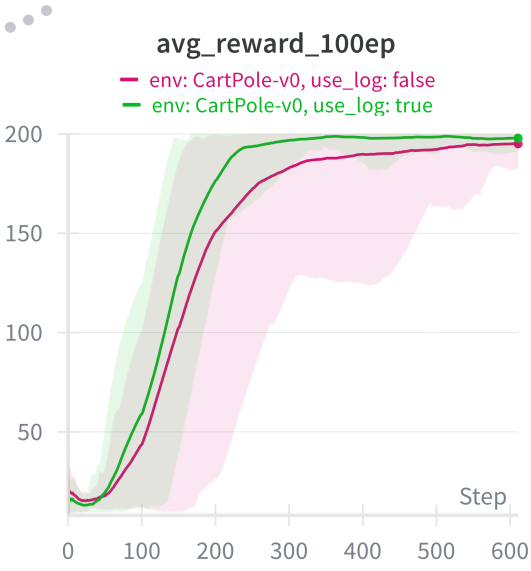
(a) LunarLander-v2: log vs. vanilla ℓ_2



(b) Acrobot-v1: log vs. vanilla ℓ_2



(a) CartPole-v1: log vs. vanilla ℓ_2



(b) CartPole-v0: log vs. vanilla ℓ_2

5.6 Best Hyperparameter Configurations

Table 1: Best-performing hyperparameter settings across environments.

Environment	Batch	Embedding	λ	Log	Episodes	Return
CartPole-v1	256	Energy	0.8	False	600	499.37
CartPole-v0	256	Vanilla	0.8	True	600	200.00
LunarLander-v2	256	Energy	0.2	True	600	268.35
Acrobot-v1	256	Energy	0.0	True	600	-74.71

5.7 Overall Comparative Study

We have grouped all the 16 (2 embedding strategies, 2 transformation strategies, and 4 values of λ resulting in $2 \times 2 \times 4 = 16$) configurations and reported their average performance in Figure 11. From all the experiments it has been inferred that on average the joint configuration of **energy** embedding, combined with a log **transformation** of the embedding difference and a $\lambda = 0.2$ has a better edge than any other configuration in the prescribed set.

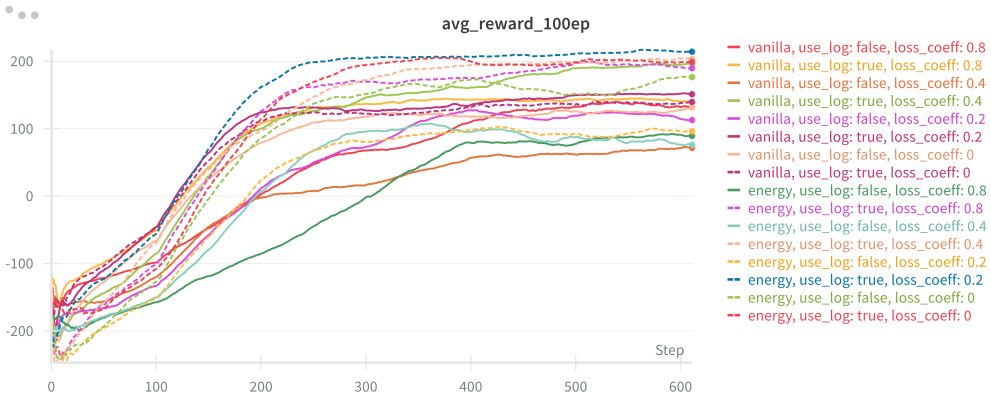


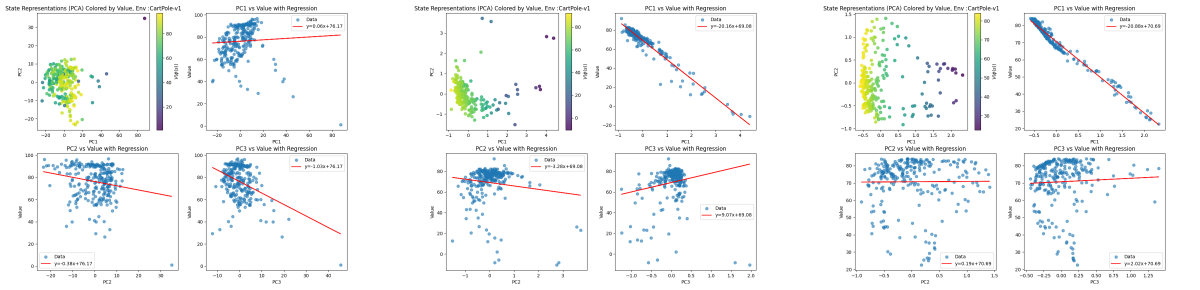
Figure 11: Comprehensive comparative study with the joint combination of embedding types, λ, and log transformation across all environments.

5.8 Visualization of Learned Representations

To conclude the efficacy of our approach, we projected the state representations onto the top 2 principal components. To present the contrast between the embeddings in presence and in absence of bisimulation, each figure block is accompanied by a) plot without bisimulation b) plot with the bisimulation and vanilla embeddings and c) plot with the bisimulation and energy embeddings. We have generated these 3 plots for each environment as can be observed in Figures 12, 13, 14, 15.

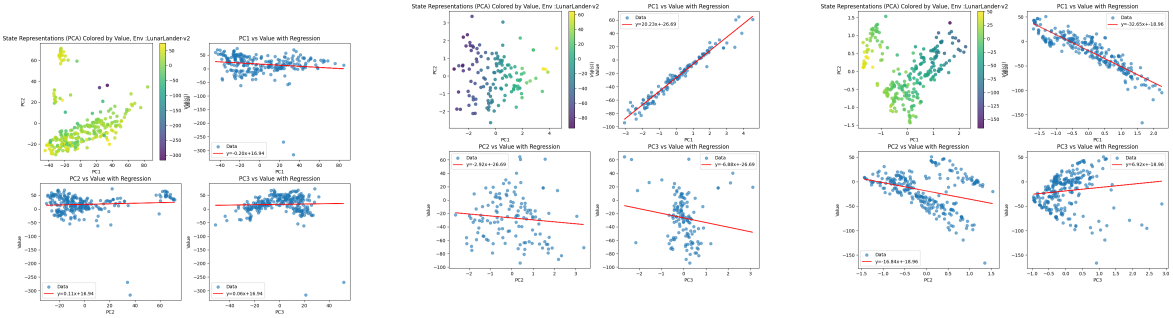
We observe a smoothness in the embedding surface when the surrogate Bisimulation loss is incorporated. This can be inferred from the smooth change in color gradients as opposed to the abrupt and presence of random colored points in the plot resulting from the absence of the surrogate loss. This also suggests that the modified metric in fact aggregates states based on value similarity

and can be further inferred from the plots that nearby embeddings in presence of the surrogate loss have similar values while in absence of the loss, the space is quite random and nothing can be inferred about the neighborhood embeddings. In addition to these, each plot is accompanied by the correlation of the state values with respect to the top 3 principal components. From the plots it can be inferred that in presence of the surrogate bisimulation loss, the value correlates highly with the top principal component of the embeddings and this is the reason such a smoothness in the color gradients have been observed in the plots. There is also a stark contrast in absence of bisimulation loss, when the state values does not correlate at all with the principal components and hence the smoothness was absent in the plots with $\lambda = 0$ (i.e plots carried out from experiments without the incorporation of the bisimulation loss). To summarize, the proposed method has successfully aggregated states based on value similarity and has leveraged this geometry of the state space to solve the environments



(a) CartPole-v1: $\lambda = 0$ (b) CartPole-v1: $\lambda = 0.8$, vanilla (c) CartPole-v1: $\lambda = 0.4$, energy

Figure 12: CartPole-v1 representation visualizations.

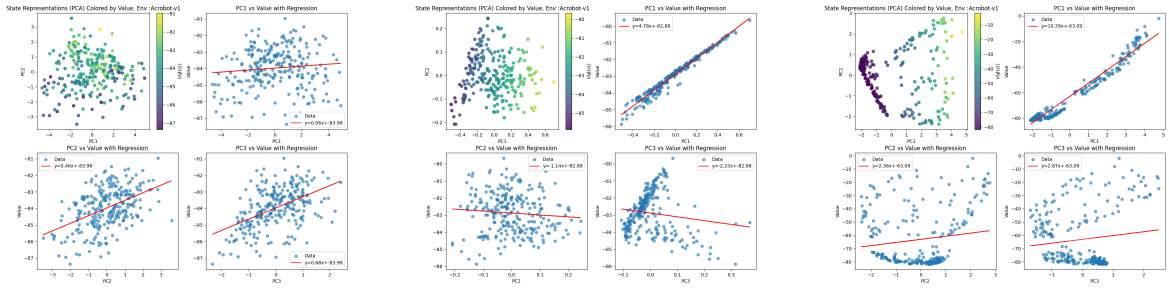


(a) LunarLander-v2: $\lambda = 0$ (b) LunarLander-v2: $\lambda = 0.4$, vanilla (c) LunarLander-v2: $\lambda = 0.4$, energy

Figure 13: LunarLander-v2 representation visualizations.

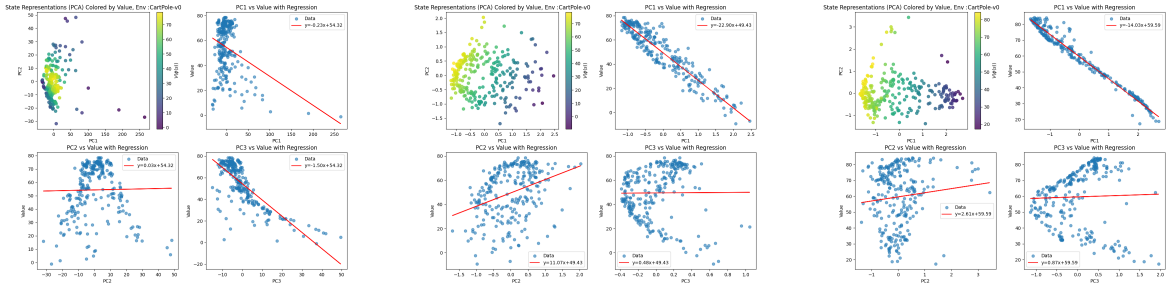
6 Conclusion

We proposed a lightweight representation learning objective for reinforcement learning based on a value-aware surrogate for the on-policy bisimulation metric. The key contributions of this work are:



(a) Acrobot-v1: $\lambda = 0$ (b) Acrobot-v1: $\lambda = 0.2$, vanilla (c) Acrobot-v1: $\lambda = 0.2$, energy

Figure 14: Acrobot-v1 representation visualizations.



(a) CartPole-v0: $\lambda = 0$ (b) CartPole-v0: $\lambda = 0.8$, vanilla (c) CartPole-v0: $\lambda = 0.8$, energy

Figure 15: CartPole-v0 representation visualizations showing clear value-based clustering with $\lambda > 0$.

1. A representation learning objective that is lightweight and uses a lower bound of bisimulation distance based on value function differences.
2. The algorithm learns good meaningful representations, validated using PCA visualizations showing clear value-based clustering.
3. We have shown several optimizational advantages associated with the objective function, including Lipschitz continuity and smoothness properties.
4. Comprehensive experimental validation across multiple classic control environments demonstrating that energy embeddings with moderate λ values and log of the feature difference provide the best performance.

7 Future Work

Several exciting directions remain for future investigation:

1. More detailed analysis on the regularization component of the objective function and its effect on generalization.
2. Extension of the algorithm to more complex continuous control environments with high-dimensional state and action spaces.

3. Investigation of transfer learning capabilities: whether representations learned on one task can accelerate learning on related tasks.

4. Theoretical analysis of the approximation quality: how close are the learned representations to true bisimulation metrics?

References

Castro, Pablo Samuel (2020). "Scalable methods for computing state similarity in deterministic Markov decision processes". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04, pp. 3484–3491.

Castro, Pablo Samuel and Doina Precup (2010). "Using bisimulation for policy transfer in MDPs". In: *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Ferns, Norm, Prakash Panangaden, and Doina Precup (2004). "Metrics for finite Markov decision processes". In: *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 162–169.

Geist, Matthieu, Bruno Scherrer, and Olivier Pietquin (2019). "A theory of regularized markov decision processes". In: *International Conference on Machine Learning*. PMLR, pp. 2190–2199.

Gelada, Carles et al. (2019). "Deepmdp: Learning continuous latent space models for representation learning". In: *International Conference on Machine Learning*. PMLR, pp. 2170–2179.

Givan, Robert, Thomas Dean, and Matthew Greig (2003). "Equivalence notions and model minimization in Markov decision processes". In: *Artificial Intelligence* 147.1-2, pp. 163–223.

Hardt, Moritz, Ben Recht, and Yoram Singer (2016). "Train faster, generalize better: Stability of stochastic gradient descent". In: *International conference on machine learning*. PMLR, pp. 1225–1234.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: *nature* 518.7540, pp. 529–533.

Zhang, Amy et al. (2020). "Learning invariant representations for reinforcement learning without reconstruction". In: *arXiv preprint arXiv:2006.10742*.