SPARSE-TD-JEPA: IMPROVING ZERO-SHOT REINFORCEMENT LEARNING WITH SPARSE LATENT TASK REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Successor-feature-based zero-shot reinforcement learning methods pre-train a task encoder and successor-feature predictor on offline, reward-free data, then recover a policy at test time by solving a closed-form linear regression over a small labeled dataset. We identify an underexplored limitation of this approach: when the task encoder produces dense representations, the inferred reward signal becomes diffuse across the state space, reducing the contrast between goal-relevant states and the background and providing a weaker signal for the policy to follow. We address this limitation by introducing an ℓ_1 sparsity penalty that concentrates reward-predictive structure into a smaller effective subspace of the task encoder's latent space. The penalty is applied only during training and requires no architectural change or modification to the closed-form inference procedure. We instantiate this approach as *Sparse-TD-JEPA*, a single-line of code addition based on Temporal-Difference Joint-Embedding Predictive Architecture (TD-JEPA) that significantly improved its performance in zero-shot reinforcement learning. We evaluate Sparse-TD-JEPA on 65 tasks across the DeepMind Control (DMC) Suite and OGBench, covering locomotion, navigation, trajectory stitching, puzzle solving, and robotic manipulation. Sparse-TD-JEPA improves over TD-JEPA in all 13 domain-level comparisons. On DMC, it increases the average return from 661 ± 6 to 714 ± 3. On OGBench, it increases the average success rate from 38.0 ± 0.8 to 48.4 ± 2.3, with improvements of 13.4–18.0 percentage points on the evaluated antmaze navigation and stitching domains. Overall, Sparse-TD-JEPA achieves the highest mean performance among established zero-shot baselines in nine of the 13 domains. Analysis further shows that sparsity produces a sharper and more task-informative inferred reward landscape, offering a mechanistic explanation for the observed performance gains.

1 INTRODUCTION

Zero-shot reinforcement learning studies the problem of pre-training a single policy on offline, reward-free data such that it can solve any downstream task without further interaction with the environment. It is important because it scales the achievable tasks of a single model without requiring an equal amount of scaling to the training data or online experience, making RL more practical across the diverse and open-ended settings encountered in the real world.

A prominent line of work addresses this problem through successor features and task encoders, where a latent representation of states is learned such that any reward function can be expressed as a linear function of that representation Barreto et al. (2017); Touati & Ollivier (2021). At test time, a small set of labeled transitions suffices to recover a task vector via linear regression, which is then used to retrieve the corresponding pre-trained policy. Recent zero-shot RL methods have explored a diverse range of objectives for constructing task representations. Laplacian representations capture the geometry of the environment through eigenfunctions of its transition graph Wu et al. (2018), while ICVF learns representations through intention-conditioned value prediction Ghosh et al. (2023). HILP instead organizes the latent space around temporal distances for goal reaching Park et al. (2024). Self-predictive approaches such as BYOL and BYOL-γ have also been adapted to the successor-feature setting by predicting future latent representations over one or multiple time

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

scales Grill et al. (2020); Lawson et al. (2025). More recent methods, including RLDP and TD-JEPA, improve multi-step latent-dynamics prediction through chained or temporal-difference objectives Jajoo et al. (2026); Bagatella et al. (2025). Although these approaches differ in how they learn task-relevant structure, they all ultimately rely on the learned task representation to support accurate linear reward inference at test time.

Despite this progress, a critical bottleneck has received comparatively little attention. When the task encoder spreads reward-predictive information diffusely across all latent dimensions, the predicted reward landscape is less concentrated near the goal. The resulting reward signal has a weaker gradient to follow, leading to degraded downstream performance. We find that this difference in landscape sharpness persists across a wide range of labeled-set sizes, suggesting it reflects a structural property of the encoder geometry.

We address these challenges through a sparsity penalty for the task encoding representation to concentrate its reward-predictive structure into a smaller effective subspace of its latent space. We instantiate this idea as *Sparse-TD-JEPA*, a single-line code addition based on TD-JEPA framework that significantly improves performance on various navigation, locomotion, and puzzle tasks. Empirically, this simple modification produces consistent gains across all 13 evaluated domain-level comparisons. On *DeepMind-Control (DMC)*, Sparse TD-JEPA improves the average return from 661 ± 6 to 714 ± 3, a gain of 53 points. On *OGBench*, it raises the average success rate from 38.0 ± 0.8 to 48.4 ± 2.3, an absolute improvement of 10.4 percentage points. Moreover, analysis experiment shows that the resulting encoder produces a sharper, more peaked predicted reward landscape, giving the downstream policy a clearer signal to follow.

Our contributions are as follows:

- We demonstrate theoretically and empirically that ℓ_1 sparsity on the task encoder improves zero-shot reinforcement learning performance.
- Through controlled analysis experiments, we show that the sparsity-penalized task encoder produces a smoother and more informative reward landscape.
- We evaluate our method across standard zero-shot reinforcement learning benchmarks covering robotic locomotion, navigation, and manipulation tasks. We demonstrate consistent gains over the base method and other established zero-shot baselines.

2 Related Work

Latent-predictive representation learning. The Joint-Embedding Predictive Architecture (JEPA) LeCun et al. (2022) is a representations learning architecture that splits an input into a context view and a target view, then trains a predictor to map the context encoding to the target encoding in a shared latent space. This principle has been instantiated across modalities. Image-based Joint-Embedding Predictive Architecture (I-JEPA) Assran et al. (2023) applies it to static images. It uses an asymmetric context/target-encoder design to predict the representations of masked image regions from visible patches. Video-based Joint-Embedding Predictive Architecture (V-JEPA) Bardes et al. (2024); Assran et al. (2025) extends this to video. It predicts masked spatio-temporal tokens from unmasked ones and supports downstream robotic planning without task-specific reward. Temporal-Difference Joint-Embedding Predictive Architecture (TD-JEPA) Bagatella et al. (2025) instantiates the same context-target prediction problem in reinforcement learning. A state encoder and a task encoder are trained via a temporal-difference latent-predictive loss over offline, off-policy transitions. The predictor is conditioned on the action and a policy latent.

Zero-shot reinforcement learning. Zero-shot Reinforcement Learning (RL) methods are methods that pre-train an agent once on offline, reward-free data, then at test time solve arbitrary downstream tasks without further interaction. Most zero-shot RL methods learn a *task encoder* alongside *successor features*. The task encoder maps states to a space in which any reward is linear. The *successor features* predict how much of each task-encoder dimension a policy accumulates over time. Forward-Backward (FB) representations Touati & Ollivier (2021), a zero-shot RL method, learns both jointly via a bilinear decomposition of successor measures. Other zero-shot reinforcement learning methods construct the task encoder through different objectives. Bootstrap Your Own Latent (BYOL) Grill et al. (2020) predicts an encoder's own future latents. The Laplacian repre-

sentation Wu et al. (2018) uses eigenfunctions of the environment's transition graph. The Intention-Conditioned Value Function (ICVF) Ghosh et al. (2023) method uses an implicit value-based objective. HILbert foundation Policy (HILP) Park et al. (2024) enforces distance-preservation for goal-reaching. Regularized Latent Dynamics Prediction (RLDP) Jajoo et al. (2026) uses chained multi-step latent prediction. BYOL-γ Lawson et al. (2025) extends BYOL to multi-step targets. TD-JEPA Bagatella et al. (2025) uses a temporal-difference latent-predictive objective as the core training signal for both encoders. It distills zero-shot policies end to end.

Sparse and structured representation learning. Sparsity-inducing penalties encourage weights or activations toward zero. We refer to Hoefler et al. (2021) for a survey. A useful distinction is between *model sparsity*, which permanently removes structural elements, and *ephemeral sparsity*, which sparsifies activations per-example without altering the architecture. Dropout Srivastava et al. (2014) applies ephemeral sparsity to hidden units for regularization. k-sparse autoencoders Makhzani & Frey (2013) retain only the top-k activations per example for unsupervised feature learning. Closer to our motivation, Liu et al. (2019) show that ephemeral sparsity in the last hidden layer of a control network improves value-function learning on classic control tasks. It reduces activation overlap between unrelated states. This mitigates interference during bootstrapping. It is evidence that sparsity can directly aid the kind of TD-based learning our method builds on.

3 Background

3.1 Problem Setup

We consider a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s' \mid s, a)$ is the transition kernel, giving the probability measure over the next state s' when action a is taken in state s, $R(s, a)$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor, giving down-weights rewards received further in the future (Sutton et al., 1998; Touati & Ollivier, 2021). A *policy* $\pi : \mathcal{S} \to \mathrm{Prob}(\mathcal{A})$ maps each state to a probability distribution over actions (Bagatella et al., 2025; Touati & Ollivier, 2021).

In *zero-shot offline RL* (Touati & Ollivier, 2021; Park et al., 2025), a single pre-trained policy must generalize to downstream reward without further environment interaction. The agent pre-trains on a static, reward-free dataset $\mathcal{D} = \{(s, a, s')\}$ of transitions, where s is the current state, a is the action taken, and s' is the resulting next state (Bagatella et al., 2025). At evaluation time it receives a small set of labeled transitions $\mathcal{D}_{\mathrm{rwd}} = \{(s, r)\}$, where $r \in \mathbf{R}$ is the scalar reward observed at state s (Bagatella et al., 2025), and must produce a near-optimal policy with no further updates to the pre-trained representation.

3.2 Temporal Difference Joint-Embedding Predictive Architecture

TD-JEPA (Bagatella et al., 2025) trains a *state encoder* ϕ and a *task encoder* ψ, together with two predictors T_ϕ and T_ψ, via temporal-difference latent-predictive losses on offline transitions $(s, a, s') \in \mathcal{D}$. The *state encoder* $\phi : \mathcal{S} \to \mathbf{R}^{d_\phi}$ compresses each state to a d_ϕ-dimensional latent vector. The *task encoder* $\psi : \mathcal{S} \to \mathbf{R}^{d_\psi}$ maps each state to a d_ψ-dimensional embedding used to define the space of linear rewards. The predictor $T_\phi : \mathbf{R}^{d_\phi} \times \mathcal{A} \times Z \to \mathbf{R}^{d_\psi}$ takes a state latent, an action, and a task vector and predicts in the d_ψ-dimensional task-encoder space. The predictor $T_\psi : \mathbf{R}^{d_\psi} \times \mathcal{A} \times Z \to \mathbf{R}^{d_\phi}$ takes a task latent, an action, and a task vector and predicts in the d_ϕ-dimensional state-encoder space.

Rewards are assumed linear in the task encoding:

$$r(s) \approx \psi(s)^\top z_r, \quad z_r \in \mathbf{R}^{d_\psi}, \tag{1}$$

where z_r is a *task vector* identifying a specific reward function (Barreto et al., 2017; Bagatella et al., 2025). Task vectors are constrained to a sphere of radius $\sqrt{d_\psi}$ via

$$\mathrm{project_z}(z) = \sqrt{d_\psi}\, \frac{z}{\|z\|_2}. \tag{2}$$

During pre-training, task vectors z are sampled from a mixture of the uniform distribution on this sphere and projected task encodings of dataset states, and the actor is conditioned on z; each z thus

Algorithm 1 Update (ϕ: state encoder, ψ: task encoder, T_ϕ: state predictor, T_ψ: task predictor) in SPARSE TD-JEPA

$\{(s_i, a_i, s_i')\}_{i=1}^{B} \leftarrow \text{SAMPLE}(\mathcal{D})$; $z_i \leftarrow \text{SAMPLE_Z}()$

$\mathcal{L} \leftarrow \mathcal{L}_\phi + \mathcal{L}_\psi + \lambda_\phi \mathcal{L}_{\text{orth}}(\phi) + \lambda_\psi \mathcal{L}_{\text{orth}}(\psi)$

$\mathcal{L} \leftarrow \mathcal{L} + \lambda_{\text{sp}} \cdot \frac{1}{B} \sum_i \|\text{project_z}(\psi(s_i))\|_1$ ▷ **only one** line needs to be added.

$\phi, \psi, T_\phi, T_\psi \leftarrow \text{ADAM_STEP}(\phi, \psi, T_\phi, T_\psi, \nabla\mathcal{L})$

indexes a policy $\pi_z(\cdot \mid s)$ that is distilled end-to-end. Throughout our analysis we write $V(s) := \psi(s)^\top z_r$ for the *inferred reward estimate* induced by a recovered task vector z_r; it is a point estimate of the immediate reward $r(s)$ in Eq. equation 1.

Zero-shot inference. Given $\mathcal{D}_{\text{rwd}} = \{(s, r)\}$, the task vector is recovered by (Bagatella et al., 2025)

$$z_r = \text{project_z}\big((\Psi^\top \Psi)^{-1} \Psi^\top r\big) \tag{3}$$

$$\Psi = [\psi(s_1) \ \cdots \ \psi(s_N)]^\top. \tag{4}$$

Here $N = |\mathcal{D}_{\text{rwd}}|$ is the number of state-reward pairs inside buffer $\mathcal{D}_{\text{rwd}}$, $\Psi \in \mathbf{R}^{N \times d_\psi}$ is the feature matrix stacking the task encodings of all N labeled states, and $r \in \mathbf{R}^N$ collects their observed rewards. The inner expression $(\Psi^\top \Psi)^{-1} \Psi^\top r$ is the ordinary least-squares fit to $\Psi z \approx r$, minimising $\sum_{(s,r) \in \mathcal{D}_{\text{rwd}}} (r - \psi(s)^\top z)^2$ over $z \in \mathbf{R}^{d_\psi}$. The result is then projected to the sphere via project_z to yield z_r. No further updates to ϕ or ψ are required at test time.

Reward Encoding Is Difficult In Zero-Shot RL. Zero-shot inference (Eq. equation 4) solves $\Psi z \approx r$ by least squares. In sparse-reward environments, the reward signal is nonzero for fewer than 1% of transitions, making r almost entirely zero.

With a dense ψ, only a handful of its d_ψ output coordinates may actually track reward-relevant structure; the remaining coordinates vary across states for reasons unrelated to reward. This is expected of ψ as trained: the TD-JEPA objective optimises ψ to serve every policy π_z, $z \in Z$, and to supply targets for the multi-step predictor T_ψ, not to discriminate the single reward r given at inference time. Individual coordinates may therefore encode nuisance state variables, structure relevant to other tasks in Z, or transient dynamics needed for multi-step prediction – all useful for ψ's training objective, but uninformative about r.

Because the least-squares fit in Eq. equation 4 assigns a weight to every coordinate of ψ regardless of relevance, the reward-irrelevant coordinates pick up spurious correlations with the few nonzero entries of r over the finite sample $\mathcal{D}_{\text{rwd}}$. The resulting z_r is nonzero across all d_ψ coordinates, but mass on reward-irrelevant coordinates adds noise to every downstream prediction $\psi(s)^\top z_r$ (Eq. equation 1), reducing the contrast between goal-region states and the background and flattening the predicted reward landscape relative to what a more concentrated z_r would produce.

4 METHOD: SPARSE TD-JEPA

We propose to induce *per-example sparsity* in the projected task embeddings $\text{project_z}(\psi(s))$ during training, encouraging each embedding to concentrate its mass on fewer strongly active dimensions. The intended effect is that reward-predictive information occupies a smaller effective subspace of ψ's output, so that least-squares inference recovers z_r as a direction aligned with that subspace, producing a sharper, more concentrated inferred reward landscape with a stronger signal for the downstream policy to follow. We note that Eq. equation 5 constrains each embedding individually and does not by itself force different states to share a support; whether a shared low-dimensional structure emerges is an empirical question, which we probe directly in the experiments (via the inferred reward landscape and its correlation with task progress) rather than assume. Because the encoder output is constrained to the sphere of radius $\sqrt{d_\psi}$, the ℓ_1 penalty cannot shrink the magnitude of the representation; on the sphere, $\|x\|_1$ is minimized exactly by coordinate-sparse directions,

so the penalty purely reorients embeddings toward sparser directions rather than acting as a generic shrinkage regularizer.

4.1 Sparsity Penalty

We augment the TD-JEPA objective with an ℓ_1 penalty on the *projected* task encoder output:

$$\mathcal{L}_{\text{sp}} = \frac{1}{B}\sum_{i=1}^{B} \big\| \operatorname{project_z}(\psi(s_i)) \big\|_1. \tag{5}$$

Here $\|\cdot\|_1 = \sum_k |(\cdot)_k|$ denotes the sum of absolute values, and the mean is taken over the B states in the minibatch. The full training objective is

$$\mathcal{L} = \mathcal{L}_\phi + \mathcal{L}_\psi + \lambda_\phi \mathcal{L}_{\text{orth}}(\phi) + \lambda_\psi \mathcal{L}_{\text{orth}}(\psi) + \lambda_{\text{sp}} \mathcal{L}_{\text{sp}}, \tag{6}$$

where $\lambda_\phi, \lambda_\psi \geq 0$ are the orthonormality loss coefficients fixed from the base TD-JEPA configuration and $\lambda_{\text{sp}} \geq 0$ is the sparsity-controlling coefficient. Dividing by B mirrors the normalisation of $\mathcal{L}_\phi$ and $\mathcal{L}_\psi$ (both use `.pow(2).sum(-1).mean()`), making λ_{sp} a stable fraction of the main loss across values of B and d_ψ. The implementation of Sparse TD-JEPA is extremely simple, requiring only one additional line of code added to the base update step, as shown in Algorithm 1.

4.2 Theoretical Analysis

Appendix A formally characterizes what the penalty in Eq. equation 5 optimizes and how this propagates to zero-shot inference. The central quantity of the analysis is the ***effective dimension*** (numerical sparsity) of an embedding $x = \operatorname{project_z}(\psi(s))$,

$$d_{\text{eff}}(x) \;:=\; \frac{\|x\|_1^2}{\|x\|_2^2} \;\in\; \big[1,\, \|x\|_0\big], \qquad \|x\|_1 \;=\; \sqrt{d_\psi\, d_{\text{eff}}(x)}, \tag{7}$$

where the identity on the right holds for every x on the sphere of radius $\sqrt{d_\psi}$. The ℓ_1 penalty is therefore an exact monotone measure of effective dimension and cannot shrink magnitude (Proposition 1). Rotation-invariant regularizers such as ℓ_2 penalties are constant on the sphere and have zero gradient (Theorem 1). Among ℓ_p penalties, $p = 1$ is the convex choice with the largest sparse-to-dense range (Proposition 2). Finally, the inference error splits into an N-independent approximation term and a variance term $\sigma^2 \operatorname{rank}(\Psi)/N$ (Proposition 5). The flat sparse-vs-dense gap across N (Figure 5) therefore reflects a change in feature geometry.

5 Experiments

We design our experiments to answer the following research questions:

- RQ1: Does inducing sparsity in the task encoder improve zero-shot performance over the dense TD-JEPA baseline and established zero-shot RL methods, across locomotion, navigation, object manipulation, and puzzle-solving benchmarks?
- RQ2: Does sparsity produce a sharper, more spatially concentrated predicted reward signal near the goal, and does this explain the stuck-policy failure mode observed with the dense encoder?
- RQ3: How does the sharpness of the inferred reward signal depend on the size of the labeled-set N used for zero-shot inference, and does our method retain its advantage across different inference budgets?

5.1 Environments and Datasets

We evaluate our method on two comprehensive offline reinforcement learning benchmark suites: DeepMind Control Suite (DMC) Tassa et al. (2018); Tunyasuvunakool et al. (2020) and Offline Goal-Conditioned RL Benchmark (OGBench) Park et al. (2025), both are popular in recent zero-shot RL literature Lawson et al. (2025); Park et al. (2024); Jajoo et al. (2026). All datasets are publicly available and are used as-is in our experiments.

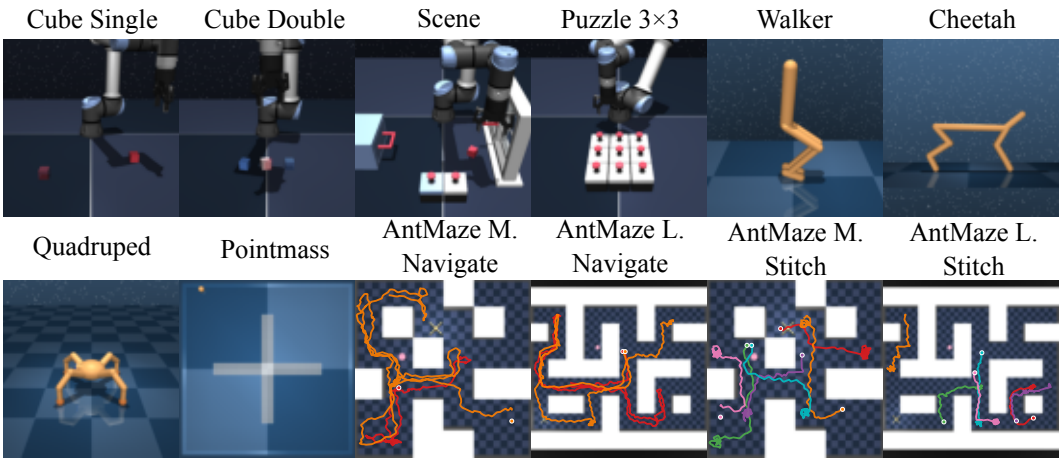

Figure 1: We test our method for zero-shot performance on various control, locomotive, and exploration tasks.

DMC is a MuJoCo environment with locomotion tasks spanning *walker*, *cheetah*, *quadruped*, and *pointmass* domains. We evaluate our method on all four domains across 20 tasks, each trained on 3 million environment steps of interaction. For training, we use the offline data set constructed with the EXploratory data for Offline Reinforcement Learning (ExoRL) protocol Yarats et al. (2022). For zero-shot evaluation, we execute 10 independent rollouts per task and report the average return.

OGBench is a MuJoCo RL benchmark with various locomotion, navigation, object manipulation, robotic arm manipulation, and puzzle-solving tasks. We evaluate our method on nine OGBench domains, each with 5 tasks, for a total of 45 different tasks. We use the *Antmaze* domain for navigation, goal-stitching, and exploration Park et al. (2025), the *Scene* and *Puzzle* domains for puzzle-solving locomotion, and the *Cube Single* and *Cube Double* domains for object manipulation tasks using robotic arm. For each domain, our model is trained on 1 million environment steps of offline interaction. For zero-shot evaluation, we execute 10 independent rollouts per task and report the average return. Performance is measured as the success rate of the binary outcome in each rollout Park et al. (2025).

5.2 Baselines

We compare our method against eight zero-shot RL baselines: Laplacian Wu et al. (2018), ICVF Ghosh et al. (2023), HILP Park et al. (2024), FB Touati & Ollivier (2021), RLDP Jajoo et al. (2026), BYOL Grill et al. (2020), BYOL-γ Lawson et al. (2025), and TD-JEPA Bagatella et al. (2025). Each baseline is tuned over its own per-domain grid of learning rates and orthonormality (or goal-sampling) coefficients to ensure a fair comparison. ICVF, BYOL, BYOL-γ are representation-learning objectives not originally proposed for zero-shot RL. We use the public codebase of Bagatella et al. (2025), which contains the adaptation of these methods for the tasks in our experiments.

5.3 Computing And Environment Infrastructure

OGBench experiments were run on a single NVIDIA H200 NVL GPU (141GB HBM3e), with an Intel Xeon 6960P (Granite Rapids) CPU, 754GB system RAM, and Linux (kernel 5.14, RHEL 9-family). DMC experiments were run on a single NVIDIA H100 NVL GPU (94GB HBM3), with an AMD EPYC 9454 (48-Core) CPU, 594GB system RAM, and Linux (kernel 5.15, Ubuntu 22.04-family). All experiments used Python 3.11 and PyTorch 2.9 (CUDA 12.8), with gymnasium 1.2, ogbench 1.2, MuJoCo 3.3, NumPy 2.3, pandas 2.3, and scikit-learn 1.9.

	Laplacian	ICVF	HILP	FB	RLDP	BYOL	BYOL-γ	TD-JEPA	**Ours**
DMC (avg)	591 ± 11	619 ± 10	620 ± 8	648 ± 4	610 ± 14	619 ± 11	645 ± 11	661 ± 6	$\mathbf{714 \pm 3}$
walker	770 ± 5	727 ± 16	796 ± 8	$\mathbf{812 \pm 6}$	724 ± 18	747 ± 11	786 ± 10	785 ± 7	$\mathbf{814 \pm 8}$
cheetah	615 ± 19	606 ± 17	618 ± 6	673 ± 5	576 ± 45	623 ± 24	647 ± 9	689 ± 7	$\mathbf{720 \pm 6}$
quadruped	635 ± 39	709 ± 14	695 ± 11	596 ± 9	665 ± 14	612 ± 28	683 ± 26	691 ± 5	$\mathbf{801 \pm 10}$
pointmass	345 ± 22	436 ± 11	371 ± 37	$\mathbf{513 \pm 20}$	$\mathbf{476 \pm 39}$	$\mathbf{493 \pm 41}$	465 ± 18	$\mathbf{479 \pm 24}$	$\mathbf{523 \pm 40}$
OGBench (avg)	14.8 ± 1.3	30.9 ± 0.6	38.0 ± 1.1	39.0 ± 0.7	27.1 ± 0.8	26.4 ± 0.8	30.4 ± 0.9	38.0 ± 0.8	$\mathbf{48.4 \pm 2.3}$
antmaze-mn	50.0 ± 4.9	$\mathbf{79.8 \pm 2.6}$	$\mathbf{83.6 \pm 2.6}$	73.0 ± 2.7	74.6 ± 4.2	58.4 ± 2.0	51.4 ± 1.6	70.4 ± 3.7	$\mathbf{85.4 \pm 4.4}$
antmaze-ln	21.6 ± 3.9	58.4 ± 1.9	52.6 ± 3.9	36.8 ± 4.3	36.4 ± 4.7	26.6 ± 3.0	21.8 ± 3.6	57.2 ± 4.3	$\mathbf{70.6 \pm 8.4}$
antmaze-ms	21.4 ± 4.3	39.0 ± 3.3	50.6 ± 2.5	$\mathbf{70.4 \pm 4.0}$	58.4 ± 3.3	60.6 ± 5.1	45.6 ± 2.8	61.6 ± 4.5	$\mathbf{79.6 \pm 8.3}$
antmaze-ls	11.8 ± 1.5	13.2 ± 1.6	12.2 ± 1.8	$\mathbf{49.8 \pm 5.6}$	19.6 ± 2.7	25.8 ± 4.3	20.2 ± 1.8	40.6 ± 2.5	$\mathbf{55.6 \pm 5.5}$
antmaze-me	0.8 ± 0.6	0.0 ± 0.0	2.0 ± 0.8	$\mathbf{51.6 \pm 2.7}$	4.8 ± 2.4	11.4 ± 2.3	19.6 ± 2.5	20.2 ± 2.4	26.6 ± 3.9
scene	7.8 ± 1.3	$\mathbf{45.4 \pm 2.3}$	$\mathbf{43.8 \pm 1.9}$	12.8 ± 1.6	11.6 ± 1.6	15.4 ± 1.4	14.4 ± 2.3	38.4 ± 1.4	$\mathbf{43.4 \pm 2.7}$
puzzle-3x3	2.8 ± 0.7	16.6 ± 0.7	2.8 ± 0.7	4.8 ± 0.7	14.6 ± 0.9	13.2 ± 1.9	18.8 ± 0.4	15.6 ± 1.1	$\mathbf{22.0 \pm 1.8}$
cube-single	15.1 ± 1.5	20.4 ± 1.9	$\mathbf{74.2 \pm 3.5}$	49.6 ± 3.8	19.8 ± 2.4	22.0 ± 3.2	$\mathbf{79.4 \pm 2.8}$	34.2 ± 2.9	46.0 ± 1.4
cube-double	2.0 ± 0.4	5.0 ± 0.8	$\mathbf{20.0 \pm 2.7}$	2.6 ± 0.4	3.8 ± 0.8	4.4 ± 0.7	2.6 ± 0.7	3.6 ± 0.8	6.4 ± 1.6

Table 1: Performance of zero-shot algorithms for DMC (reward) and OGBench (success rate) with proprioception inputs. We report means and standard errors across seeds; DMC values are rounded to the nearest integer and OGBench values to one decimal place. Numbers are bold for top algorithms if confidence intervals overlap.

6 Results

6.1 Zero-shot Performance

Table 1 reports zero-shot performance across all benchmarks. Sparse TD-JEPA has a higher mean score than TD-JEPA in every domain. On antmaze-medium-navigate (85.4 ± 4.4), it surpasses the previous best, HILP (83.6 ± 2.6), and improves over the TD-JEPA baseline (70.4 ± 3.7) by 15 points. On antmaze-large-navigate (70.6 ± 8.4), it outperforms the next best method, ICVF (58.4 ± 1.9), and improves over TD-JEPA (57.2 ± 4.3) by 13.4 points. Gains are similarly consistent on the more challenging stitch tasks: on antmaze-medium-stitch (79.6 ± 8.3 vs. 61.6 ± 4.5 for TD-JEPA) and antmaze-large-stitch (55.6 ± 5.5 vs. 40.6 ± 2.5 for TD-JEPA), our method achieves the highest success rates, including over the strongest stitch baseline, FB. On the DMC locomotion benchmark, Sparse TD-JEPA achieves the best performance on all four tasks.

In the next experiment, we investigate why our method produces higher task performance than the dense baseline.

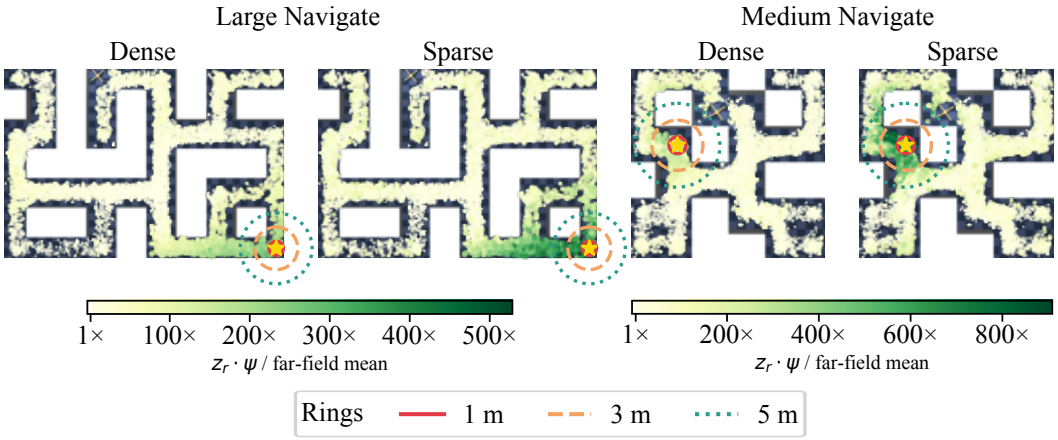

Figure 2: Spatial reward landscape for task 3 on antmaze-large-navigate (*left pair*) and antmaze-medium-navigate (*right pair*). Colour encodes $z_r^\top \psi(s)$ normalized by the model's own mean predicted reward for far-field states (>10 m from goal). Within each domain, Dense TD-JEPA (*left*) and Sparse TD-JEPA (*right*) share the same colour scale. The star marks the goal; rings indicate 1 m, 3 m, and 5 m distance.

Analysis Experiment 1: Distance–Reward Correlation. Figure 2 shows the spatial concentration of the predicted reward, $z_r^\top \psi(s)$ normalized by each model's own mean predicted reward for far-field states (>10 m), for both encoders on the same task. For the dense (baseline) encoder, the normalized

reward stays elevated across nearly the whole goal room, giving little indication of the exact goal location once the agent is inside it. For the sparse encoder, the same signal is more informative: it peaks sharply at the goal itself and falls off visibly toward the room's edges.

To verify this observation quantitatively, we measure the correlation betweem the Dense and Sparse model's learned value estimate $V(s) = \psi(s)^\top z_r$ against a distance-based task progress signal x: for the goal-conditioned OGBench domains, x is the Euclidean distance from s to the goal position; for the DMC domains, which have no spatial goal, x is the environment's own instantaneous reward. An example of such correlation is shown in Figure 4. For each of 20 independently resampled observation batches (shared between sparse and baseline), we fit $V(s) \sim x$ via linear regression and report the 5-fold cross-validated R^2. The statistical significance is determined through a Welch's t-test.

The results are shown in Figure 3. Sparse encoder produced a statistically significant higher correlation between predicted reward and task completion distance than Dense encoder in all domains except for Large Navigate. Testing whether the sparse encoder better represents other task dimensions besides distance-to-goal is left to future work.

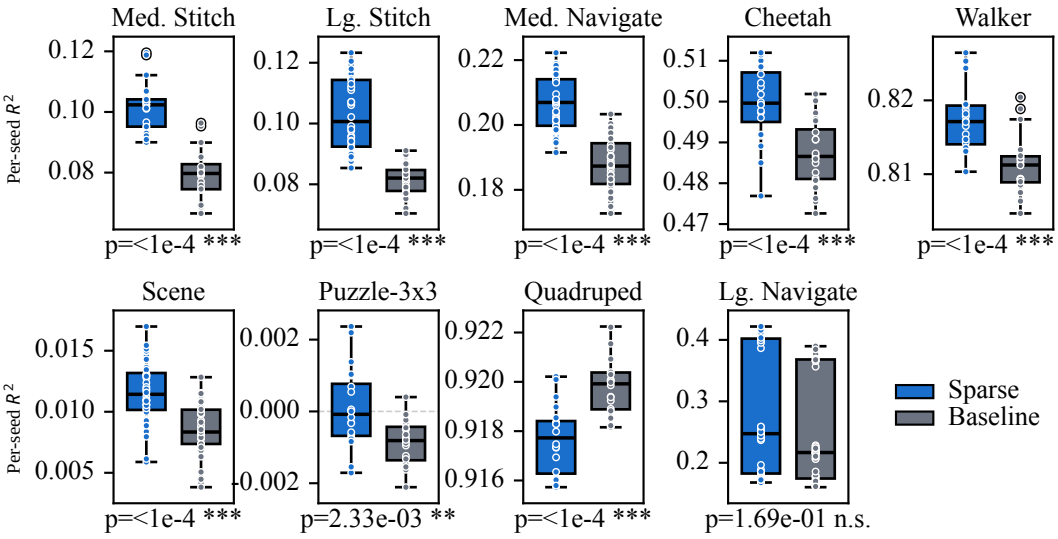

Figure 3: Correlation between each model's learned value estimate $V(s) = \psi(s) \cdot z_r$ and distance signal.

Analysis Experiment 2: Effect of Labeled-Set Size N on Reward Signal Quality. We characterise how well the inferred value estimate $V(s) = \psi(s)^\top z_r$ tracks the ground-truth signal (distance-to-goal for OGBench, reward for DMC), measured by R^2, as a function of the number of labeled transitions N used to estimate z_r. N is the number of labeled (state, reward) transitions used to solve for the task vector z_r via least-squares ($z_r = \mathrm{lstsq}(\psi(s), r)$, then projected onto the ψ-sphere).

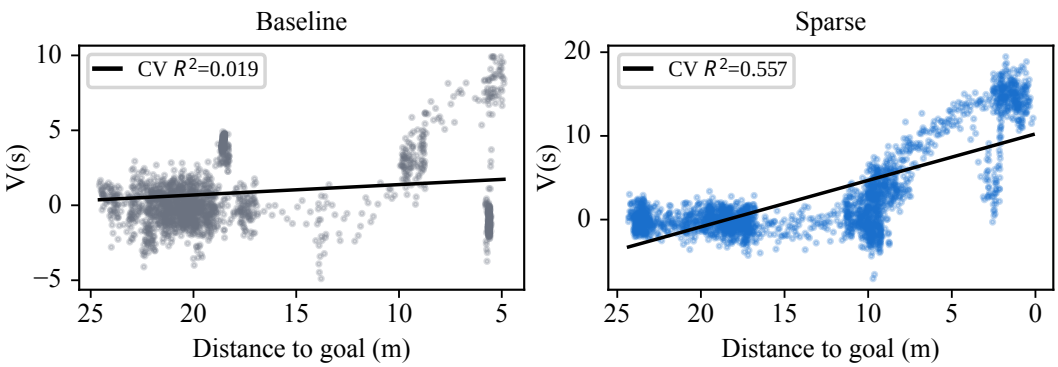

Figure 4: Example of linear regression correlation of Dense (Baseline) and Sparse task encoders.

z_r has ψ-dim $= 50$ free parameters, fit by plain unregularized least squares. Below $N \approx 50$ the system is underdetermined, and for N in the low hundreds (just above 50) unregularized OLS sits at the interpolation threshold, where the fit is barely determined and hence maximally sensitive to noise in those specific points. To stay clear of this unstable regime, we sweep N over log-decades starting at $N = 10^4$ (well above ψ-dim) up to each domain's full training-dataset size; since that full-dataset size differs by domain, the swept range is not the same across domains.

For each N, we refit z_r from 20 independently resampled batches of N labeled transitions and, for each, compute the 5-fold cross-validated R^2 (degree 1) between $V(s)$ and the ground-truth signal, evaluated on a shared, fixed set of states so that sparse and baseline are compared on identical inputs at every N. This isolates the effect of labeled-set size from any confound in which states each policy visits. We report the mean R^2 across each 20 independently resampled batches with their standard error (SEM), and flag any N where the sparse-vs-baseline difference is statistically significant via Welch's t-test ($p < 0.05$).

Results are shown in Figure 5. The sparse encoder attains a significantly higher R^2 than the baseline across nearly the full swept range of N in six of the nine domains (antmaze-ls, antmaze-mn, antmaze-ms, scene, cheetah, walker); antmaze-ln shows the same direction without reaching significance at any N; puzzle-3x3 shows no reliable signal for either model ($R^2 \approx 0$). Within each domain the sparse-vs-baseline gap in R^2 stays roughly flat across N, indicating that the sparse encoder's advantage in how well $V(s)$ tracks the ground-truth signal is not merely a low-data artifact tied to the $N = 10^4$ inference budget used in the main experiments (see the Experiments section), but persists as more labeled data becomes available.

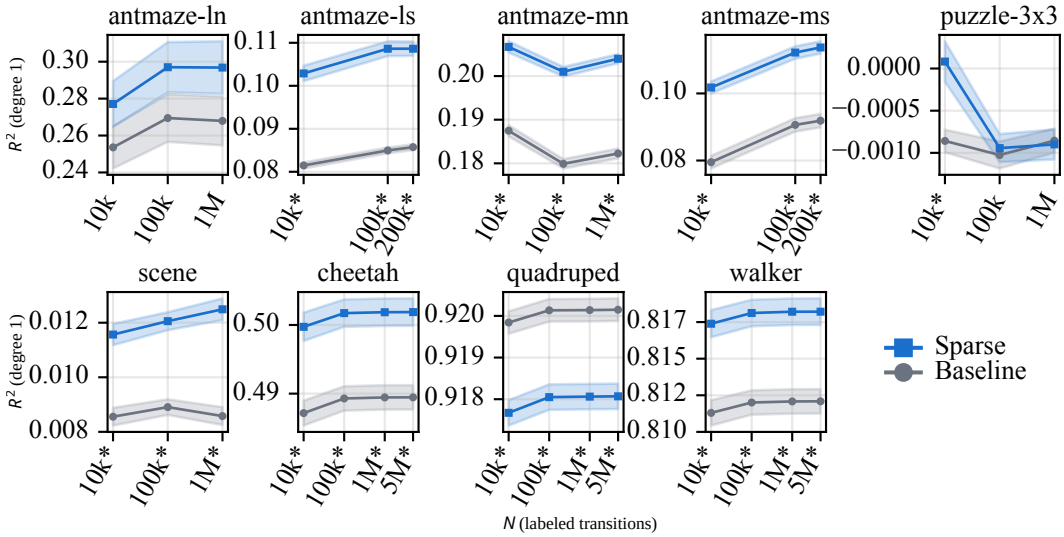

Figure 5: Cross-validated R^2 between learned value estimate and the distance signal.

7 Conclusion

We identified an underexplored failure mode of successor-feature-based zero-shot RL, dense task encoders yield a diffuse inferred reward signal with little contrast between goal-relevant states and the background, and addressed it with a simple one-line ℓ_1 penalty on TD-JEPA's projected task embeddings, which provably controls only the effective dimensionality of the representation and leaves the closed-form inference procedure untouched. Sparse TD-JEPA improves over the dense baseline in almost every evaluated domain, and our analyses tie these gains to a sharper, more goal-concentrated inferred reward that persists across labeled-set sizes. Future work includes applying the sparsity penalty to other successor-feature-based zero-shot RL methods beyond TD-JEPA, and leveraging the resulting sparse, low-effective-dimension task embeddings for mechanistic interpretability of what individual latent dimensions encode.

AI USE STATEMENT

In this work, we used AI tools (LLM-based assistants) to assist with polishing the manuscript text. We have not used generative AI tools for research ideation or method design. The identification of the failure mode, the proposed sparsity penalty, the theoretical analysis, and all experimental conclusions are the authors' own. We have reviewed all AI-assisted work: AI-assisted text was checked against the underlying experimental results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken several steps to make our results reproducible. All theoretical claims summarized in Section 4.2 are proved in Appendix A, which states every assumption and includes complete proofs. Appendix C documents the seeds and the result-aggregation protocol. Appendix D specifies all baselines and their hyperparameter grids, and Appendix E gives full details of the datasets, environments, and evaluation protocol. Compute and infrastructure details are provided in Section 5.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15619–15629, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Marco Bagatella, Matteo Pirotta, Ahmed Touati, Alessandro Lazaric, and Andrea Tirinzoni. Td-jepa: Latent-predictive representations for zero-shot reinforcement learning. *arXiv preprint arXiv:2510.00739*, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P Van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

David L Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Dibya Ghosh, Chethan Anand Bhateja, and Sergey Levine. Reinforcement learning from passive data via latent intentions. In *International Conference on Machine Learning*, pp. 11321–11339. PMLR, 2023.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124, 2021.

Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

Niall Hurley and Scott Rickard. Comparing measures of sparsity. *IEEE Transactions on Information Theory*, 55(10):4723–4741, 2009.

Pranaya Jajoo, Harshit Sikchi, Siddhant Agarwal, Amy Zhang, Scott Niekum, and Martha White. Regularized latent dynamics prediction is a strong baseline for behavioral foundation models. *arXiv preprint arXiv:2603.15857*, 2026.

Daniel Lawson, Adriana Hugessen, Charlotte Cloutier, Glen Berseth, and Khimya Khetarpal. Self-predictive representations for combinatorial generalization in behavioral cloning. *arXiv preprint arXiv:2506.10137*, 2025.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Vincent Liu, Raksha Kumaraswamy, Lei Le, and Martha White. The utility of sparse representations for control in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4384–4391, 2019.

Alireza Makhzani and Brendan Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with hilbert representations. *arXiv preprint arXiv:2402.15567*, 2024.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. *Advances in Neural Information Processing Systems*, 34:13–23, 2021.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: https://doi.org/10.1016/j.simpa.2020.100022. URL `https://www.sciencedirect.com/science/article/pii/S2665963820300099`.

Twan van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

Stefan Wager, Sida Wang, and Percy S Liang. Dropout training as adaptive regularization. *Advances in Neural Information Processing Systems*, 26:351–359, 2013.

Yifan Wu, George Tucker, and Ofir Nachum. The laplacian in rl: Learning representations with efficient approximations. *arXiv preprint arXiv:1810.04586*, 2018.

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. *arXiv preprint arXiv:2201.13425*, 2022.

A Theoretical Analysis of the Sparsity Penalty

This appendix develops the formal analysis summarized in Section 4.2. Throughout, we characterize (a) the quantity that the penalty $\mathcal{L}_{\text{sp}}$ of Eq. equation 5 actually optimizes under TD-JEPA's sphere constraint, (b) the behavior of the broader family of norm-based regularizers under the same constraint, (c) the optimization dynamics that the penalty induces on the sphere, and (d) how the geometry of the task encoder propagates to the least-squares inference step of Eq. equation 4. The section closes with a systematic comparison of alternative regularizers in light of these results.

A.1 Setup and Notation

Throughout this appendix we abbreviate $d := d_\psi$ and write

$$\mathcal{S}_d \;:=\; \left\{\, x \in \mathbf{R}^d \;:\; \|x\|_2 = \sqrt{d}\,\right\} \tag{8}$$

for the sphere onto which task embeddings and task vectors are projected, and

$$\Pi(y) \;:=\; \text{project_z}(y) \;=\; \sqrt{d}\,\frac{y}{\|y\|_2}, \qquad y \neq 0, \tag{9}$$

for the projection map of Eq. equation 2. The penalized quantity in Eq. equation 5 is $x = \Pi(\psi(s)) \in \mathcal{S}_d$, so every statement below about $x \in \mathcal{S}_d$ applies verbatim to the embeddings seen by the penalty. For $x \in \mathbf{R}^d$ we write $\|x\|_p = (\sum_k |x_k|^p)^{1/p}$ for $p \geq 1$, $\text{supp}(x) = \{k : x_k \neq 0\}$, and $\|x\|_0 = |\text{supp}(x)|$. We call x 1-*sparse* if $\|x\|_0 = 1$ and *flat on its support* if $|x_k|$ is constant over $k \in \text{supp}(x)$; a flat vector with full support ($\|x\|_0 = d$) is called *flat*. An *orthogonal transformation* is a matrix $U \in O(d)$, i.e., $U^\top U = I$.

Our analysis is phrased in terms of the following continuous notion of sparsity, standard in the sparse-recovery literature (Hoyer, 2004; Hurley & Rickard, 2009).

Definition 1 (Effective dimension / numerical sparsity)**.** For $x \in \mathbf{R}^d \setminus \{0\}$, the *effective dimension* of x is

$$d_{\text{eff}}(x) \;:=\; \frac{\|x\|_1^2}{\|x\|_2^2}. \tag{10}$$

The functional d_{eff} is scale-invariant ($d_{\text{eff}}(cx) = d_{\text{eff}}(x)$ for $c \neq 0$) and continuous on $\mathbf{R}^d \setminus \{0\}$; Proposition 1 shows $1 \leq d_{\text{eff}}(x) \leq \|x\|_0$, so d_{eff} is a continuous surrogate for the (discontinuous) support size $\|x\|_0$, and $d_{\text{eff}}(x) = k$ exactly on vectors that are flat on a support of size k. Intuitively, $d_{\text{eff}}(x)$ counts the number of coordinates of x that carry non-negligible mass.

A.2 The Penalty Is an Exact Measure of Effective Dimensionality

The first result makes precise, and strengthens, the informal claim of Section 4: under the sphere constraint the ℓ_1 penalty coincides, up to a fixed strictly monotone transformation, with the effective dimension of the embedding. In particular the penalty carries no information about, and exerts no pressure on, the magnitude of the representation, because that magnitude is fixed by the constraint.

Proposition 1 (Exact identity and extreme points)**.** *Let $x \in \mathcal{S}_d$. Then:*

- *(i) $\|x\|_1 = \sqrt{d\, d_{\text{eff}}(x)}$, and consequently the per-example penalty is a strictly increasing function of the effective dimension $d_{\text{eff}}(x)$;*
- *(ii) $1 \leq d_{\text{eff}}(x) \leq \|x\|_0 \leq d$, with $d_{\text{eff}}(x) = \|x\|_0$ if and only if x is flat on its support;*
- *(iii) $\sqrt{d} \leq \|x\|_1 \leq d$; the lower bound is attained exactly at the $2d$ signed axis vectors $x = \pm\sqrt{d}\, e_k$ (the 1-sparse elements of $\mathcal{S}_d$), and the upper bound exactly at the 2^d flat vectors with $|x_k| = 1$ for all k (the maximally dense elements).*

Proof. (i) is immediate from Definition 1 and $\|x\|_2^2 = d$: $d_{\text{eff}}(x) = \|x\|_1^2/d$, hence $\|x\|_1 = \sqrt{d\, d_{\text{eff}}(x)}$, and $t \mapsto \sqrt{d\, t}$ is strictly increasing.

(ii) Lower bound: $\|x\|_1^2 = \sum_k x_k^2 + \sum_{j\neq k} |x_j||x_k| \geq \|x\|_2^2$, so $d_{\text{eff}}(x) \geq 1$, with equality if and only if all cross terms vanish, i.e., x has at most one nonzero coordinate. Upper bound: by the Cauchy–Schwarz inequality applied over $\text{supp}(x)$,

$$\|x\|_1 = \sum_{k\in\text{supp}(x)} 1 \cdot |x_k| \;\leq\; \sqrt{\|x\|_0}\left(\sum_k x_k^2\right)^{1/2} = \sqrt{\|x\|_0}\,\|x\|_2, \tag{11}$$

so $d_{\text{eff}}(x) \leq \|x\|_0$, with equality if and only if $(|x_k|)_{k\in\text{supp}(x)}$ is proportional to the all-ones vector, i.e., x is flat on its support.

(iii) follows by combining (i) with (ii): $\|x\|_1 = \sqrt{d\, d_{\text{eff}}(x)} \in [\sqrt{d}, \sqrt{d\cdot d}] = [\sqrt{d}, d]$, and the equality characterizations transfer. For the lower extreme, $d_{\text{eff}}(x) = 1$ forces $\|x\|_0 = 1$, and membership in $\mathcal{S}_d$ then forces $x = \pm\sqrt{d}\, e_k$. For the upper extreme, $d_{\text{eff}}(x) = d$ forces $\|x\|_0 = d$ and flatness, and membership in $\mathcal{S}_d$ forces $|x_k| = 1$ for all k. □

Corollary 1 (No shrinkage; the penalty acts on direction only)**.** *For every state s, the penalized embedding satisfies $\|\Pi(\psi(s))\|_2 = \sqrt{d}$ identically, so the penalty cannot decrease (or increase) the norm of any representation. By Proposition 1(i), minimizing $\mathcal{L}_{\text{sp}} = \frac{1}{B}\sum_i \|\Pi(\psi(s_i))\|_1 = \frac{\sqrt{d}}{B}\sum_i \sqrt{d_{\text{eff}}\big(\Pi(\psi(s_i))\big)}$ is equivalent to minimizing the batch average of $\sqrt{d_{\text{eff}}(\cdot)}$; the sole quantity the penalty can move is the per-example effective dimension.*

Remark 1. This is the formal counterpart of the ridge-versus-lasso distinction (Tibshirani, 1996) in a constrained setting: in unconstrained regression, an ℓ_1 penalty both shrinks and sparsifies, and its effects are entangled; on the sphere the shrinkage component is annihilated by the constraint, leaving sparsification as the only degree of freedom. The training signal added by Algorithm 1 is therefore a pure effective-dimensionality pressure.

A.3 Behavior of Norm-Based Regularizers Under the Sphere Constraint

We next characterize how the broader design space of regularizers interacts with the sphere constraint. The organizing concept is rotation invariance.

Definition 2 (Rotation-invariant regularizer)**.** A function $\Omega : \mathbf{R}^d \to \mathbf{R}$ is *rotation-invariant* if $\Omega(Ux) = \Omega(x)$ for every $x \in \mathbf{R}^d$ and every $U \in O(d)$.

Examples include $\Omega(x) = \|x\|_2$, $\Omega(x) = \|x\|_2^2$, any composition $\Omega(x) = h(\|x\|_2)$ with $h : \mathbf{R}_{\geq 0} \to \mathbf{R}$, and smoothed variants such as $\Omega(x) = \mathbb{E}_{\xi\sim\mathcal{N}(0,\tau^2 I)}\, h(\|x+\xi\|_2)$, since an isotropic Gaussian is itself rotation-invariant.

Theorem 1 (Inertness of rotation-invariant regularizers)**.** *Let $\Omega : \mathbf{R}^d \to \mathbf{R}$ be rotation-invariant. Then:*

- *(i) Ω is constant on $\mathcal{S}_d$; call this constant c_Ω;*
- *(ii) the composition $\Omega\circ\Pi$ is constant on $\mathbf{R}^d\setminus\{0\}$, and its gradient is identically zero wherever it is differentiable;*
- *(iii) for every coefficient $\lambda \geq 0$, the objective $\mathcal{L} + \lambda\,\frac{1}{B}\sum_i \Omega\big(\Pi(\psi(s_i))\big)$ has the same gradients with respect to all parameters as $\mathcal{L}$, at every parameter vector; hence it has the same stationary points, and gradient-based training (including SGD and Adam, for any step-size schedule and any realization of minibatch sampling) produces identical parameter trajectories.*

Proof. (i) The orthogonal group acts transitively on each centered sphere: for any x, y with $\|x\|_2 = \|y\|_2$ there exists $U \in O(d)$ with $Ux = y$ (complete $x/\|x\|_2$ and $y/\|y\|_2$ to orthonormal bases and map one to the other). Hence $\Omega(y) = \Omega(Ux) = \Omega(x)$ for all $x, y \in \mathcal{S}_d$.

(ii) For every $y \neq 0$, $\Pi(y) \in \mathcal{S}_d$, so $\Omega(\Pi(y)) = c_\Omega$ by (i). A constant function has zero gradient wherever differentiable.

(iii) By (ii) the added term equals the constant λc_Ω as a function of the parameters (through any parameterization of ψ), so its contribution to every parameter gradient is zero; the chain rule through

Π and ψ transmits a zero upstream gradient. Identical gradients at every point imply identical update sequences for any first-order method fed the same minibatch sequence, hence identical trajectories and identical sets of stationary points. □

Corollary 2 (Instances)**.** *Under the sphere constraint of TD-JEPA, an ℓ_2 penalty $\|\Pi(\psi(s))\|_2$, a squared penalty $\|\Pi(\psi(s))\|_2^2$, any penalty of the form $h(\|\Pi(\psi(s))\|_2)$, and any isotropically smoothed norm penalty are all inert in the sense of Theorem 1: adding them changes neither the training trajectory nor the final encoder, for any coefficient value.*

Two widely used regularizers act on parameters or activations rather than directly on the projected embedding; the following remarks characterize their behavior in this architecture.

Remark 2 (Weight decay). The projection is scale-invariant: $\Pi(cy) = \Pi(y)$ for all $c > 0$. Consequently, if the final linear layer of ψ is scaled by $c > 0$, every projected embedding $\Pi(\psi(s))$, and hence every quantity entering the reward model of Eq. equation 1 and the inference step of Eq. equation 4, is unchanged. Along such scaling rays the data-fitting terms of the objective are constant while the weight-decay term strictly decreases, so weight decay on the output layer moves parameters along function-preserving rays: it controls the parameter norm (equivalently, the effective learning rate of normalized layers; cf. van Laarhoven, 2017) without expressing any preference among the distinct projected representations that training could select. In particular it exerts no pressure on $d_{\text{eff}}(\Pi(\psi(s)))$.

Remark 3 (Dropout). Dropout applied to the task-encoder output (Srivastava et al., 2014) multiplies coordinates by independent Bernoulli variables. In expectation, dropout on a representation feeding a (locally) linear readout is equivalent, to second order, to a coordinate-wise *quadratic* penalty weighted by the per-coordinate second moments $\mathbb{E}[x_k^2]$ (Wager et al., 2013). Two properties of this induced penalty are relevant here. First, it is quadratic, and quadratic penalties lack the polyhedral vertex geometry through which ℓ_1 produces (approximately) sparse solutions (Tibshirani, 1996). Second, TD-JEPA's orthonormality loss drives the second moments toward isotropy, $\mathbb{E}[\psi(s)\psi(s)^\top] \approx I$; in that regime the induced penalty degenerates toward a function of the ℓ_2 norm alone, which is inert on the sphere by Corollary 2. Dropout therefore injects noise, but does not supply a per-example effective-dimensionality pressure.

A.4 The ℓ_p Family: Which Exponents Favor Sparse Directions

Theorem 1 shows that a regularizer must be coordinate-anisotropic to have any effect at all under the sphere constraint. The canonical anisotropic family is the ℓ_p family. The next proposition determines, for every exponent, which directions on the sphere the penalty favors, and locates ℓ_1 within the family.

Proposition 2 (ℓ_p penalties on the sphere)**.** *Let $x \in \mathcal{S}_d$.*

- *(i) For $1 \le p < 2$: $d^{1/2} \le \|x\|_p \le d^{1/p}$, with the minimum attained exactly at 1-sparse vectors and the maximum exactly at flat vectors. Minimizing $\|x\|_p$ over $\mathcal{S}_d$ therefore favors sparse directions.*
- *(ii) For $p = 2$: $\|x\|_2 \equiv \sqrt{d}$ on $\mathcal{S}_d$; the penalty is constant (Theorem 1).*
- *(iii) For $2 < p \le \infty$: $d^{1/p} \le \|x\|_p \le d^{1/2}$, with the minimum attained exactly at flat vectors and the maximum exactly at 1-sparse vectors. Minimizing $\|x\|_p$ over $\mathcal{S}_d$ therefore favors* dense *directions.*
- *(iv) Within the convex range $p \ge 1$, the ratio between the penalty's value at its densest and sparsest extremes on $\mathcal{S}_d$ equals $d^{|1/p-1/2|}$, which is maximized at $p = 1$, where it equals $\sqrt{d}$. In this sense ℓ_1 is the convex member of the family with the largest dynamic range between sparse and dense embeddings.*

Proof. We use two standard inequalities for $0 < a \le b \le \infty$ and $x \in \mathbf{R}^d$:

$$\text{(N)}\ \ \|x\|_b \le \|x\|_a, \qquad \text{(H)}\ \ \|x\|_a \le d^{\,1/a-1/b}\,\|x\|_b. \tag{12}$$

(N) follows by normalizing $\|x\|_a = 1$, so that $|x_k| \le 1$ and $|x_k|^b \le |x_k|^a$ for each k; summing gives $\|x\|_b^b \le 1$. Equality in (N) holds if and only if $\|x\|_0 \le 1$: with two or more nonzero entries, each

normalized entry is strictly below 1, so $|x_k|^b < |x_k|^a$ for those entries. (H) is Hölder's inequality applied to $\sum_k |x_k|^a \cdot 1$ with exponent pair $(b/a,\ (1-a/b)^{-1})$:

$$\|x\|_a^a = \sum_k |x_k|^a \cdot 1 \le \Big(\sum_k |x_k|^b\Big)^{a/b} d^{\,1-a/b} = \|x\|_b^a \, d^{\,1-a/b}, \tag{13}$$

with equality if and only if $(|x_k|^a)_k$ is proportional to the all-ones vector, i.e., $|x|$ is flat.

(i) Apply (N) with $(a,b)=(p,2)$: $\|x\|_p \ge \|x\|_2 = d^{1/2}$, equality iff x is 1-sparse. Apply (H) with $(a,b)=(p,2)$: $\|x\|_p \le d^{1/p-1/2}\|x\|_2 = d^{1/p}$, equality iff x is flat.

(iii) Apply (N) with $(a,b)=(2,p)$: $\|x\|_p \le \|x\|_2 = d^{1/2}$, equality iff x is 1-sparse. Apply (H) with $(a,b)=(2,p)$: $d^{1/2} = \|x\|_2 \le d^{1/2-1/p}\|x\|_p$, i.e., $\|x\|_p \ge d^{1/p}$, equality iff x is flat. (For $p=\infty$ the same conclusions follow by taking limits, with $\|x\|_\infty \in [1, \sqrt{d}\,]$.)

(ii) is immediate, and (iv) follows by comparing the extreme values from (i) and (iii): the ratio is $d^{1/p}/d^{1/2} = d^{1/p-1/2}$ for $p<2$ and $d^{1/2}/d^{1/p} = d^{1/2-1/p}$ for $p>2$; over $p \ge 1$ the exponent $|1/p - 1/2| \le 1/2$ is maximized at $p=1$. □

Remark 4 (Boundary members of the family). For $0<p<1$ the map $x \mapsto \sum_k |x_k|^p$ favors sparse directions even more strongly, but is non-convex with unbounded gradients at the coordinate hyperplanes, which is problematic for stochastic first-order training; the limit $p \to 0$ recovers the support size $\|x\|_0$, which is piecewise constant and provides no gradient signal at all. The exponent $p=1$ is the classical convex surrogate for $\|x\|_0$ used throughout sparse recovery (Tibshirani, 1996; Donoho, 2006), and by Proposition 2(iv) it is also the convex member with the largest sparse-to-dense dynamic range on the sphere. Proposition 2(iii) additionally yields a falsifiable control: a penalty with $p>2$ (or, equivalently, rewarding large $\|x\|_p$ for $p<2$) is predicted to push embeddings toward the flat, maximally dense extreme, the opposite intervention to Eq. equation 5.

A.5 Optimization Geometry of the Penalty on the Sphere

We now describe the dynamics that the penalty induces. Since embeddings live on $\mathcal{S}_d$, the relevant object is the Riemannian gradient of $\|\cdot\|_1$ on the sphere, i.e., the tangential component of the Euclidean subgradient.

Proposition 3 (Gradient flow of the penalty: mass transport with a participation threshold)**.** *Let $x \in \mathcal{S}_d$ have no zero coordinate, and let $\sigma = \operatorname{sign}(x) \in \{\pm 1\}^d$. The Riemannian gradient of $f(x) = \|x\|_1$ on $\mathcal{S}_d$ at x is*

$$\operatorname{grad} f(x) \;=\; \sigma \;-\; \frac{\|x\|_1}{d}\, x, \tag{14}$$

and the penalty-only descent flow $\dot{x} = -\operatorname{grad} f(x)$ preserves $\mathcal{S}_d$ and satisfies, for every coordinate k,

$$\frac{\mathrm{d}}{\mathrm{d}t}\,|x_k| \;=\; \frac{\|x\|_1}{d}\Big(|x_k| - \tau(x)\Big), \qquad \tau(x) := \frac{d}{\|x\|_1} \in [1, \sqrt{d}\,]. \tag{15}$$

That is, at fixed total ℓ_2 mass, the penalty continuously transports magnitude from coordinates below the state-dependent participation threshold $\tau(x)$ to coordinates above it: strongly active coordinates grow and weakly active coordinates decay.

Proof. On the open set where no coordinate vanishes, $f(x) = \sigma^\top x$ is smooth with Euclidean gradient σ. The tangent space of $\mathcal{S}_d$ at x is $\{v : x^\top v = 0\}$, with orthogonal projector $P_x = I - xx^\top/\|x\|_2^2 = I - xx^\top/d$. Hence

$$\operatorname{grad} f(x) = P_x\sigma = \sigma - \frac{\sigma^\top x}{d}\, x = \sigma - \frac{\|x\|_1}{d}\, x, \tag{16}$$

using $\sigma^\top x = \|x\|_1$. The flow preserves the sphere because $\frac{\mathrm{d}}{\mathrm{d}t}\|x\|_2^2 = 2x^\top\dot{x} = -2\big(x^\top\sigma - \frac{\|x\|_1}{d}\|x\|_2^2\big) = -2(\|x\|_1 - \|x\|_1) = 0$. For each coordinate, $\frac{\mathrm{d}}{\mathrm{d}t}|x_k| = \sigma_k\dot{x}_k = -\sigma_k\big(\sigma_k - \frac{\|x\|_1}{d}x_k\big) = \frac{\|x\|_1}{d}|x_k| - 1$, which equals the right-hand side of Eq. equation 15. The range of τ follows from Proposition 1(iii): $\|x\|_1 \in [\sqrt{d}, d]$ implies $\tau(x) = d/\|x\|_1 \in [1, \sqrt{d}\,]$. □

Proposition 4 (Critical points form a discrete effective-dimension ladder). *Consider $f(x) = \|x\|_1$ restricted to $\mathcal{S}_d$, in the sense of Clarke-stationarity (there exist $g \in \partial\|\cdot\|_1(x)$ and $\mu \in \mathbf{R}$ with $g = \mu x$). Then:*

(i) the stationary points are exactly the vectors that are flat on their support: for each support size $k \in \{1, \dots, d\}$ these are the points with $|x_j| = \sqrt{d/k}$ for j in the support, and the corresponding penalty value is $f(x) = \sqrt{kd}$, strictly increasing in k;

(ii) for every $k \geq 2$, no stationary point with support size k is a local minimizer: within the great circle obtained by rotating any two supported coordinates, the flat point is a strict local maximizer of f;

(iii) the global minimizers are exactly the $2d$ signed axis vectors $\pm\sqrt{d}\, e_k$, with value $\sqrt{d}$.

Proof. (i) The subdifferential of $\|\cdot\|_1$ at x is $\partial\|\cdot\|_1(x) = \{g : g_j = \operatorname{sign}(x_j) \text{ if } x_j \neq 0,\ g_j \in [-1, 1] \text{ otherwise}\}$. Stationarity requires $g = \mu x$ for some such g. On the support this reads $\operatorname{sign}(x_j) = \mu x_j$, so $\mu > 0$ and $|x_j| = 1/\mu$ is constant across the support; off the support the condition $g_j = \mu \cdot 0 = 0 \in [-1, 1]$ is always satisfiable. Thus stationary points are precisely the flat-on-support vectors. If the support has size k, the constraint $\|x\|_2^2 = k/\mu^2 = d$ gives $1/\mu = \sqrt{d/k}$ and $f(x) = k\sqrt{d/k} = \sqrt{kd}$.

(ii) Let x be stationary with support size $k \geq 2$ and pick supported coordinates $i \neq j$, so $|x_i| = |x_j| = c := \sqrt{d/k}$. Consider the curve on $\mathcal{S}_d$ that rotates the pair while fixing all other coordinates:

$$x_i(\theta) = \sqrt{2}\, c\, \sigma_i \cos\theta, \qquad x_j(\theta) = \sqrt{2}\, c\, \sigma_j \sin\theta, \qquad \theta \in (0, \tfrac{\pi}{2}), \tag{17}$$

which preserves $x_i^2 + x_j^2 = 2c^2$ and hence membership in $\mathcal{S}_d$, and passes through x at $\theta = \pi/4$. Along this curve, $f(\theta) = C + \sqrt{2}\, c\, (\cos\theta + \sin\theta) = C + 2c \sin(\theta + \pi/4)$ with C collecting the fixed coordinates; this is strictly maximized at $\theta = \pi/4$. Hence f strictly decreases in both directions along the curve, and x is not a local minimizer.

(iii) By Proposition 1(iii), $f \geq \sqrt{d}$ on $\mathcal{S}_d$ with equality exactly at the signed axis vectors, which are the $k = 1$ rung of the ladder in (i). □

Remark 5 (Interpretation). Propositions 3 and 4 together describe the penalty's contribution to training as a descent process on a ladder of configurations indexed by effective dimension: penalty values $\sqrt{kd}$ at the flat-on-support configurations interpolate exactly the effective dimensions $d_{\text{eff}} = k$ of Proposition 1, higher rungs are unstable to the pairwise rotations of Proposition 4(ii), and the flow of Proposition 3 implements the descent by rich-get-richer mass transport among coordinates. In Sparse TD-JEPA this pressure acts jointly with $\mathcal{L}_\phi$, $\mathcal{L}_\psi$, and the orthonormality losses of Eq. equation 6, so training equilibrates at an intermediate effective dimension determined by the trade-off between predictive fit and the sparsity coefficient λ_{sp}; the coefficient functions as a continuous dial on per-example effective dimensionality, rather than a hard architectural choice such as a top-k constraint (Makhzani & Frey, 2013).

A.6 Propagation to Zero-Shot Inference

We now connect encoder geometry to the least-squares inference of Eq. equation 4. Two preliminary observations let us analyze the unprojected least-squares solution without loss of generality.

Lemma 1 (The final projection preserves all scale-invariant readouts). *Let $\hat{z} \in \mathbf{R}^d \setminus \{0\}$ and $z_r = \Pi(\hat{z})$. Then $z_r = c\,\hat{z}$ with $c = \sqrt{d}/\|\hat{z}\|_2 > 0$, so $V(s) = \psi(s)^\top z_r = c\,\psi(s)^\top \hat{z}$ for every state. Consequently every scale-invariant functional of the landscape, the normalized maps of Figures 2 and 6, 8, correlation coefficients and cross-validated R^2 against any reference signal (Figures 3–5), the ranking of states by predicted reward, and the location of the maximizer, is identical for z_r and $\hat{z}$.*

Proof. Immediate from the definition of Π and positive homogeneity: multiplying V by a positive constant preserves ratios, orderings, argmaxima, and correlation-type statistics, and R^2 of a linear fit is invariant under positive rescaling of either variable. □

Proposition 5 (Exact bias–variance decomposition of the inferred landscape)**.** *Fix the encoder ψ and labeled states $s_1, \dots, s_N$, and let $\Psi \in \mathbf{R}^{N\times d}$ be the feature matrix of Eq. equation 4 with $m := \operatorname{rank}(\Psi)$. Suppose the observed labels are $y_i = r(s_i) + \varepsilon_i$ where $r \in \mathbf{R}^N$ collects the ground-truth rewards and the ε_i are independent with mean zero and variance σ^2. Let $\hat{z} = \Psi^\dagger y$ be the (minimum-norm) least-squares solution and $\hat{V} = \Psi\hat{z}$ the inferred rewards at the labeled states. Then*

$$\mathbb{E}\left[\frac{1}{N}\left\|\hat{V} - r\right\|_2^2\right] = \underbrace{\frac{1}{N}\left\|(I - P_\Psi)\, r\right\|_2^2}_{\text{approximation error of the feature span}} + \underbrace{\frac{\sigma^2 m}{N}}_{\text{estimation variance}}, \tag{18}$$

where $P_\Psi = \Psi\Psi^\dagger$ is the orthogonal projector onto the column span of Ψ. The first term depends only on how well the span of the learned features $\{\psi(\cdot)^\top w : w \in \mathbf{R}^d\}$, evaluated on the labeled states, can represent the reward, a property of the encoder's geometry that is independent of N once the span is fixed, while the second term scales as the number of effective feature directions m divided by N.

Proof. P_Ψ is the orthogonal projector onto $\operatorname{col}(\Psi)$, with $\operatorname{rank}(P_\Psi) = m$, and $\hat{V} = P_\Psi y = P_\Psi r + P_\Psi\varepsilon$. Therefore

$$\hat{V} - r = -(I - P_\Psi)r + P_\Psi\varepsilon, \tag{19}$$

an orthogonal decomposition ($(I - P_\Psi)r \perp P_\Psi\varepsilon$ deterministically), so

$$\|\hat{V} - r\|_2^2 = \|(I - P_\Psi)r\|_2^2 + \|P_\Psi\varepsilon\|_2^2. \tag{20}$$

Taking expectations, $\mathbb{E}\|P_\Psi\varepsilon\|_2^2 = \mathbb{E}\,\operatorname{Tr}\big(P_\Psi\varepsilon\varepsilon^\top\big) = \sigma^2\operatorname{Tr}(P_\Psi) = \sigma^2 m$, since P_Ψ is idempotent and symmetric. Dividing by N gives Eq. equation 18. □

Corollary 3 (Reading the labeled-set-size sweep through Eq. equation 18)**.** *Consider two encoders evaluated at a common sequence of labeled-set sizes N, as in Analysis Experiment 2. By Eq. equation 18, the difference between their expected landscape errors is*

$$\Delta(N) = \Delta_{\text{approx}} + \frac{\sigma^2\,\Delta m}{N}, \tag{21}$$

the sum of an N-independent difference in approximation error and a difference in estimation variance that decays as $1/N$. The two components are therefore identifiable from the shape of the empirical gap as a function of N: a gap arising from estimation variance shrinks by an order of magnitude per decade of N, while a gap arising from the feature span's approximation error is constant in N. In Figure 5, the sparse-vs-dense gap in cross-validated R^2 is approximately constant across the swept decades of N, which corresponds to the approximation term: the penalty's effect manifests as a change in the geometry of the learned feature span itself, in the exact sense of the first term of Eq. equation 18, rather than as a change in estimation variance at the inference stage.

Remark 6 (Relation to the sparse-reward mechanism of Section 3). Proposition 5 also formalizes the informal account given in Section 3 of how a diffuse encoder degrades the landscape. Coordinates of ψ that vary for reasons unrelated to reward enlarge the effective span dimension m entering the variance term, and, more importantly at the large N used in practice, populate the feature span with directions whose projection $P_\Psi r$ misallocates reward mass across the state space; both effects lower the contrast of $V(s) = \psi(s)^\top z_r$ between goal-region states and the background. By Lemma 1 this contrast loss survives the final projection unchanged. Conversely, an encoder whose per-example activity is concentrated (Proposition 1) can only improve the landscape through these same two terms, which is precisely the signature observed in Figures 2, 3, and 5.

A.7 Gauge Symmetry

The results so far characterize the penalty and the inference step separately. This subsection makes precise how the two connect, by identifying an exact symmetry of the base method and locating the penalty's action relative to it.

Proposition 6 (Rotational gauge symmetry of the base objective)**.** *Let $R \in O(d)$. Consider the reparameterization*

$$\psi \mapsto R\psi, \qquad T_\phi \mapsto R\,T_\phi(\cdot,\cdot,R^\top\cdot), \qquad T_\psi \mapsto T_\psi(R^\top\cdot,\cdot,R^\top\cdot), \qquad z \mapsto Rz, \tag{22}$$

applied consistently to the task encoder, both predictors' ψ-space inputs and outputs, the actor's task-vector conditioning, and the sampled task vectors. Then every term of the base TD-JEPA objective in Eq. equation 6 with $\lambda_{\mathrm{sp}} = 0$ is unchanged: the latent-prediction losses $\mathcal{L}_\phi, \mathcal{L}_\psi$ compare ψ-space vectors through Euclidean distances, which are preserved by R; the orthonormality loss depends on ψ through $\mathbb{E}[\psi\psi^\top]$ compared against I in a rotation-invariant matrix norm, and $\|R\Sigma R^\top - I\|_F = \|\Sigma - I\|_F$; the projection is equivariant, $\Pi(Ry) = R\,\Pi(y)$; and the uniform distribution on $\mathcal{S}_d$ used for task-vector sampling is rotation-invariant. Moreover, zero-shot inference is equivariant: replacing Ψ by $\Psi R^\top$ maps $\hat{z} \mapsto R\hat{z}$ and leaves every inferred value $V(s) = \psi(s)^\top \hat{z}$, hence every landscape, and every downstream policy retrieval, identical.

Proof sketch. Each claim is a direct computation. For inference: with $\Psi' = \Psi R^\top$, $\hat{z}' = (\Psi')^\dagger y = R\,\Psi^\dagger y = R\hat{z}$ (using $(AR^\top)^\dagger = RA^\dagger$ for orthogonal R), so $\psi'(s)^\top \hat{z}' = \psi(s)^\top R^\top R\hat{z} = \psi(s)^\top \hat{z}$. The loss terms follow from isometry of R and invariance of the Frobenius norm and of the uniform sphere measure. □

Proposition 6 has two consequences that sharpen the interpretation of the penalty.

Lemma 2 (Second-moment regularizers do not determine per-example effective dimension)**.** *There exist encoders with identical second moments $\mathbb{E}[\psi\psi^\top] = I$, hence identical orthonormality loss for every coefficient, whose per-example effective dimensions differ by a factor of order d. Concretely, let $k(s)$ be uniform on $\{1, \dots, d\}$ and $\xi(s) \in \{\pm 1\}$ unbiased, and define the axis encoder $\psi^{(1)}(s) = \sqrt{d}\,\xi(s)\,e_{k(s)}$ and, for any fixed $R \in O(d)$, the rotated encoder $\psi^{(2)} = R\,\psi^{(1)}$. Then $\mathbb{E}[\psi^{(1)}\psi^{(1)\top}] = \mathbb{E}[\psi^{(2)}\psi^{(2)\top}] = I$, while $d_{\mathrm{eff}}(\psi^{(1)}(s)) = 1$ identically and $d_{\mathrm{eff}}(\psi^{(2)}(s)) = \|Re_{k(s)}\|_1^2$, which for a uniformly random rotation concentrates at $(2/\pi)\,d\,(1 + o(1))$.*

Proof. $\mathbb{E}[\psi^{(1)}\psi^{(1)\top}] = d \cdot \frac{1}{d}\sum_k e_k e_k^\top = I$, and second moments are preserved by any fixed rotation: $\mathbb{E}[\psi^{(2)}\psi^{(2)\top}] = RIR^\top = I$. The effective dimensions follow from Definition 1 with $\|\psi^{(2)}(s)\|_2 = \sqrt{d}$ and $\|\psi^{(2)}(s)\|_1 = \sqrt{d}\,\|Re_{k(s)}\|_1$. For Haar-random R, the column Re_k is uniform on the unit sphere, whose coordinate marginals satisfy $\mathbb{E}|u_1| = \sqrt{2/(\pi d)}\,(1 + O(1/d))$; hence $\mathbb{E}\|Re_k\|_1 = d\,\mathbb{E}|u_1| = \sqrt{2d/\pi}\,(1 + O(1/d))$, and $\|Re_k\|_1$ concentrates around this value by standard concentration of Lipschitz functions on the sphere, giving $d_{\mathrm{eff}} = \|Re_k\|_1^2 = (2/\pi)d\,(1 + o(1))$ with high probability. □

Remark 7 (Locating the penalty's action). Combining the pieces: (a) by Proposition 6, the base objective is exactly invariant under the rotational gauge, so its optima come in $O(d)$-orbits, and every rotation-invariant regularizer is constant on each orbit (Theorem 1); (b) by Lemma 2, per-example effective dimension varies from 1 to $\Theta(d)$ *within* a single orbit, so it is a quantity that neither the base losses nor any rotation-invariant addition determines; (c) by Proposition 1, the ℓ_1 penalty is an exact monotone function of precisely this quantity. The penalty is thus the minimal coordinate-anisotropic addition that breaks the rotational degeneracy in the direction of concentrated embeddings. At the same time, the inference-equivariance statement in Proposition 6 shows that re-expressing a *fixed* set of learned features in a sparser basis leaves every inferred landscape unchanged; a change in downstream landscapes therefore reflects a change in the learned feature functions themselves, the span entering the approximation term of Proposition 5, induced by the altered training dynamics of Propositions 3 and 4. This is consistent with, and predicted by, the N-independent structure of the empirical gap in Corollary 3. Finally, as stated in Section 4, per-example concentration does not by itself force different states to share a support; whether the concentrated coordinates align into a shared low-dimensional structure is an empirical property of the trained encoder, which the landscape and correlation analyses of the Experiments section probe directly. We note that the combination of pressures at work here, per-example ℓ_1 concentration under a global second-moment isotropy constraint, is the classical recipe of sparse dictionary learning (Olshausen & Field, 1996), which likewise uses it to align a rotationally degenerate linear representation with structure-bearing coordinates.

A.8 Alternative Regularizers Under the Sphere Constraint

Table 2 collects the conclusions of this appendix for the design space of embedding-level regularizers in this architecture. The behaviors fall into four classes: penalties that are provably inert under the

sphere constraint; penalties that act only on parameter scale; penalties that favor the dense extreme; and sparsity-promoting penalties, within which ℓ_1 is the convex member with maximal dynamic range that leaves both the architecture and the closed-form inference procedure of Eq. equation 4 untouched. Hard sparsification schemes such as top-k selection (Makhzani & Frey, 2013) do enforce concentration, but do so by changing the forward computation at training and test time, replacing the continuous coefficient λ_{sp} with a discrete architectural constant and altering the feature map seen by least-squares inference; the ℓ_1 penalty achieves graded concentration with no inference-time change, which is the property exploited by the single-line implementation of Algorithm 1. The utility of activation sparsity for TD-based value learning observed by Liu et al. (2019) is complementary evidence from the online control setting.

Regularizer	Behavior on $\mathcal{S}_d$	Effect on training	Shown in
ℓ_2, $\|x\|_2^2$, any $h(\|x\|_2)$	constant	none (zero gradient)	Thm. 1, Cor. 2
isotropically smoothed norms	constant	none (zero gradient)	Cor. 2
weight decay (output layer)	function-preserving	parameter scale only	Rem. 2
dropout on ψ output	$\approx$ quadratic, $\to$ isotropic	noise; no sparsity pressure	Rem. 3
orthonormality (second moments)	orbit-invariant	blind to per-example $d_{\text{eff}}(x)$	Lem. 2
ℓ_p, $2 < p \leq \infty$	minimized at flat x	favors dense embeddings	Prop. 2(iii)
ℓ_p, $1 \leq p < 2$	minimized at sparse x	favors sparse embeddings	Prop. 2(i)
ℓ_p, $p < 1$; ℓ_0	minimized at sparse x	non-convex / no gradient	Rem. 4
top-k (hard)	$d_{\text{eff}}(x) \leq k$ enforced	alters architecture & inference	§A.8
ℓ_1 (Eq. equation 5)	$= \sqrt{d\, d_{\text{eff}}(x)}$ exactly	continuous dial on $d_{\text{eff}}(x)$	Prop. 1

Table 2: Behavior of embedding-level regularizers under TD-JEPA's sphere constraint. $d_{\text{eff}}(x)$ denotes the effective dimension of Definition 1.

B Qualitative Rollout Analysis of the Stuck-Policy Failure Mode

RQ2 in the main paper's Experiments section asks whether the sharper, sparsity-induced reward landscape explains the stuck-policy failure mode we observe with the dense TD-JEPA baseline. Figures 6–8 give concrete, single-episode illustrations of what this failure mode looks like in practice. The systematic evidence for the hypothesis itself remains the aggregate correlation results reported in the main experiment, across all seeds and domains.

Each figure shows one evaluation episode for a fully trained model, comparing Sparse TD-JEPA (*Ours*, top row) against the dense TD-JEPA baseline (*bottom row*) on the same task, seeded with the same evaluation seed. *Reward Landscape* shows the spatial reward-landscape: colour encodes $z_r^\top \psi(s)$ normalized by each model's own mean predicted reward for far-field states (> 10 m from goal), with the goal marked by a star and dotted rings at 1 m, 3 m, and 5 m. *Rollout* overlays the agent's actual trajectory on the maze, coloured by its own learned value estimate $V(s) = \psi(s)^\top z_r$ along the path (red = low, green = high). *Value Convergence vs. Distance to Goal* plots $V(s)$ and the Euclidean distance-to-goal against timestep, together with their ordinary-least-squares trend lines; *Task Success* marks with a check or cross whether the episode reached the goal.

We highlight two recurring, related instances of this failure mode below.

B.1 Failure Mode 1: Flat Value Signal

One failure mode we observed empirically is that the dense baseline often never acquires enough signal to leave the vicinity of its start state at all. As an example, Figures 6 and 7 show one full episode each on antmaze-large-stitch and antmaze-medium-navigate. In both cases the sparse agent's distance-to-goal decreases steadily over the episode while $V(s)$ rises correspondingly, reaching the goal well before the episode times out. The dense baseline, by contrast, remains confined to a small region near its starting position for the entire episode: its distance-to-goal trend is essentially flat, and its $V(s)$ trace is noisy with no discernible trend.

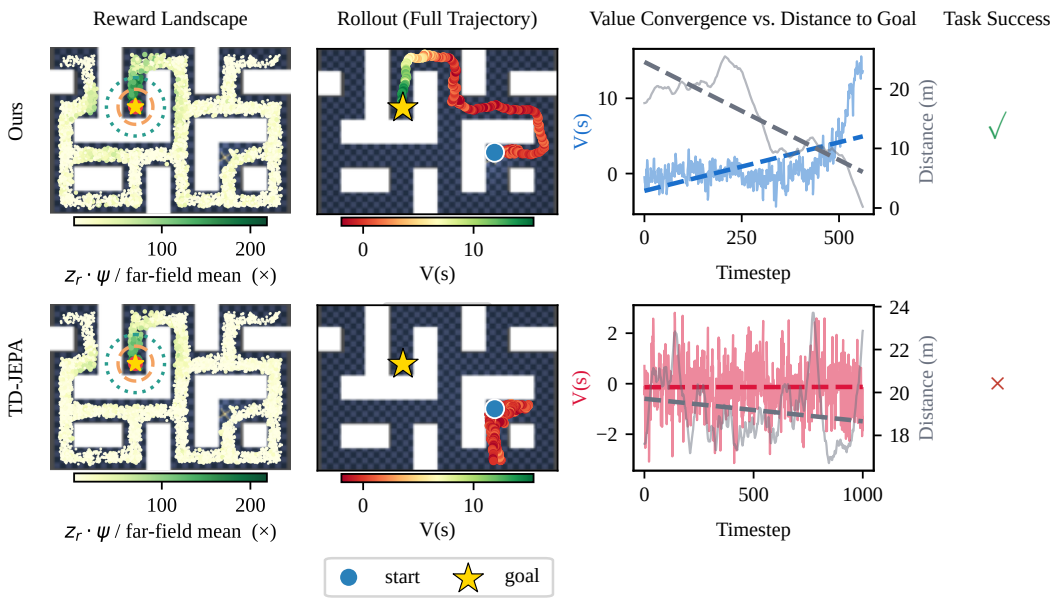

Figure 6: Full antmaze-large-stitch episode (task 4). Sparse TD-JEPA navigates from its start position to the goal, with $V(s)$ rising and distance-to-goal falling together over the episode. The dense TD-JEPA baseline instead loops within a small region far from the goal for the entire episode, never making net progress, and fails.

B.2 Failure Mode 2: Diffuse Reward Signal Near the Goal

A related instance of the same underlying issue occurs even when the dense baseline does make it near the goal. The predicted reward stays elevated across most of the goal room rather than peaking sharply at the goal itself, giving the policy little local gradient to follow. As an example, Figure 8 shows the last five metres of one such episode on the antmaze-large-stitch task. The sparse agent's $V(s)$ rises steadily as distance-to-goal falls, and it reaches the goal. The dense baseline's $V(s)$ instead fluctuates without a clear upward trend even as it moves in and out of the goal room, and it visibly doubles back on its own path near the goal rather than committing to it; the episode times out without success. This is consistent with the diffuse reward landscape reported in the main paper's Experiments section: near the goal, small differences in position are not reflected in $V(s)$ for the dense encoder, so the policy has no local signal telling it exactly where to stop.

C Seeds and Result Aggregation

Every method reported in the main paper's results table is run with the same fixed seeds. Each seed controls network initialization and training data sampling.

3917	3502	8948	9460	4729
2226	1744	7742	4501	6341

All 10 of these seeds are used for OGBench domains. The first 5 of these seeds are used for DMC domains. For each seed, success rate is measured over 10 evaluation rollouts per task at the final training checkpoint (1M gradient steps for OGBench, 3M for DMC). The evaluation environment is seeded independently of the training seed, across all methods, and is set to 0. This ensures that every model is evaluated against the same sequence of environment resets per task. All runs use deterministic execution, which enables PyTorch's deterministic algorithms and configures CUDA Deep Neural Network library (cuDNN) to operate in deterministic mode.

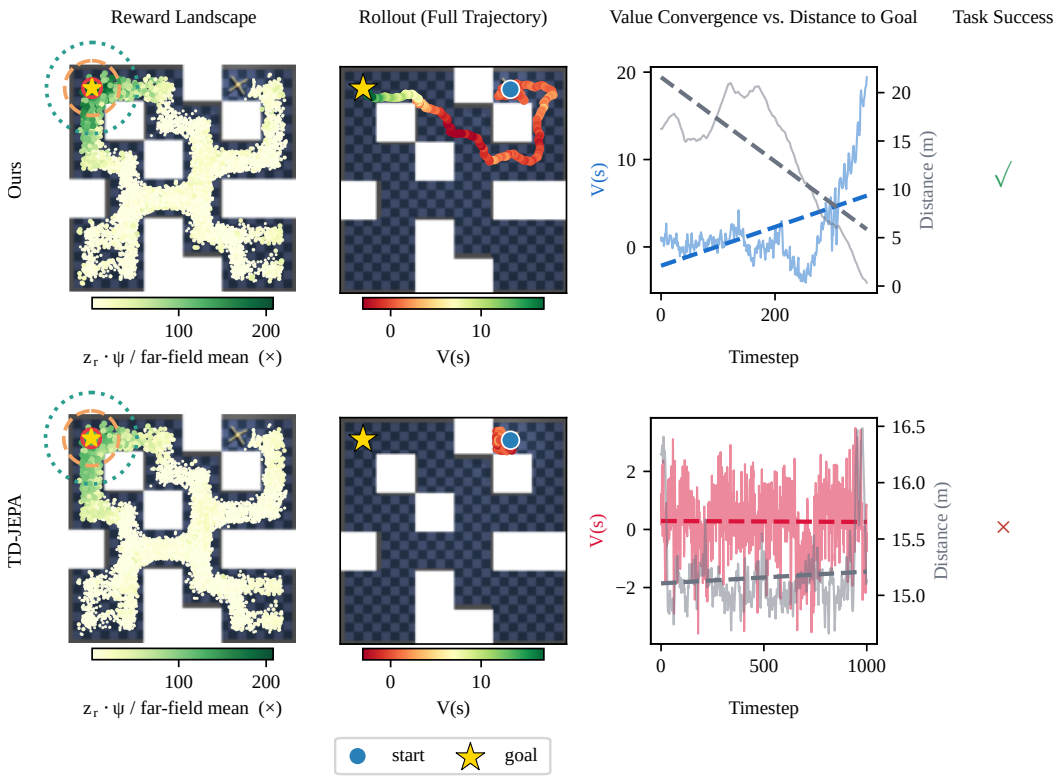

Figure 7: Full antmaze-medium-navigate episode (task 4). Sparse TD-JEPA reaches the goal directly from its start position. The dense TD-JEPA baseline remains tangled in a small loop immediately around its start position for the full episode and fails to make progress toward the goal.

D BASELINES

This section gives an overview of the tested baselines' architecture and learning hyperparameters. Every method below is swept over the same 10 (OGBench) / 5 (DMC) seeds listed in the *Seeds and Result Aggregation* section. For each seed we keep the grid point with the highest final-checkpoint success rate. OGBench grids are $2 \times 2 = 4$ combinations per domain. DMC grids are $2 \times 3 = 6$ combinations per domain.

D.1 LAPLACIAN

Wu et al. (2018)

Domain group	$\text{lr}_{\text{features}}$	ortho. coef.
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{0,\ 1\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{0,\ 1\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0,\ 0.1,\ 1\}$

Table 3: Laplacian hyperparameter grid.

D.2 HILBERT FOUNDATION POLICY (HILP)

Park et al. (2024)

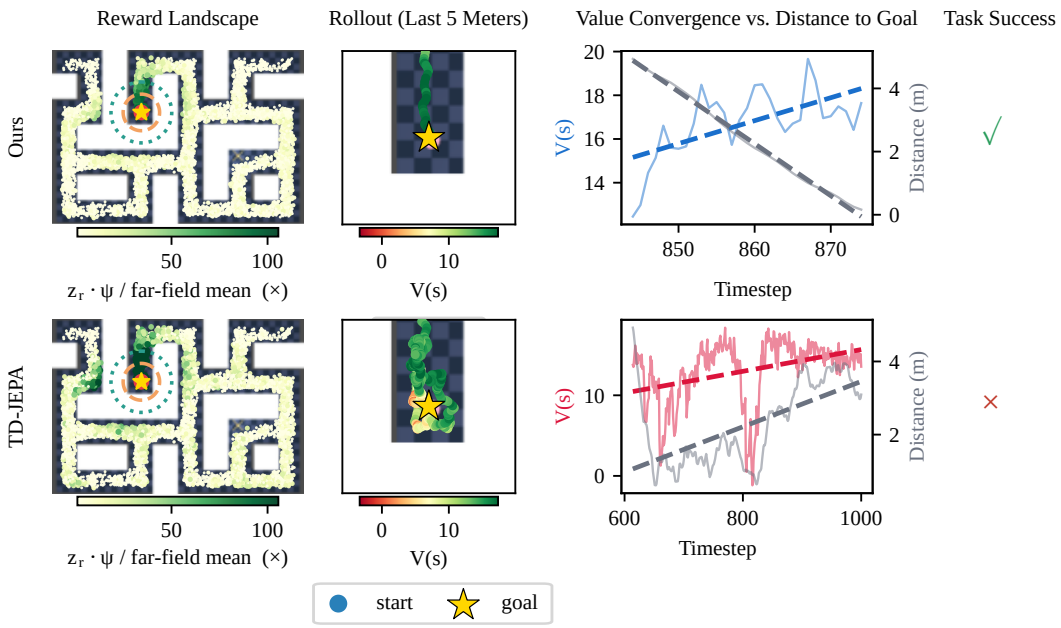

Figure 8: Last five metres of an antmaze-large-stitch episode (task 4). *Rollout* zoomed to the 10 m×10 m region centred on the goal. Sparse TD-JEPA's value estimate $V(s)$ rises monotonically as it closes the remaining distance and succeeds; the dense TD-JEPA baseline's $V(s)$ is noisy and directionless over the same last-mile window, even dropping sharply to near zero at one point, and the agent wanders near the goal room without ever committing to the goal, ultimately failing the episode.

Domain group	$\text{lr}_{\text{features}}$	goal-sampling prob.
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{0.375,\ 0.5\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{0.375,\ 0.5\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0.375,\ 0.2,\ 0.5\}$

Table 4: HILP hyperparameter grid (also used, unmodified, for ICVF).

D.3 Intention-Conditioned Value Function (ICVF)

Ghosh et al. (2023) ICVF reuses the same numeric grid as HILP (below); it differs only in fixed (non-swept) settings, namely expectile $= 0.9$ (vs. 0.5 for HILP) and a normalised feature encoder.

D.4 Forward-Backward (FB)

Touati & Ollivier (2021)

Domain group	lr_b	ortho. coefficient
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{100,\ 1000\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{100,\ 1000\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0.1,\ 1,\ 10\}$

Table 5: FB hyperparameter grid.

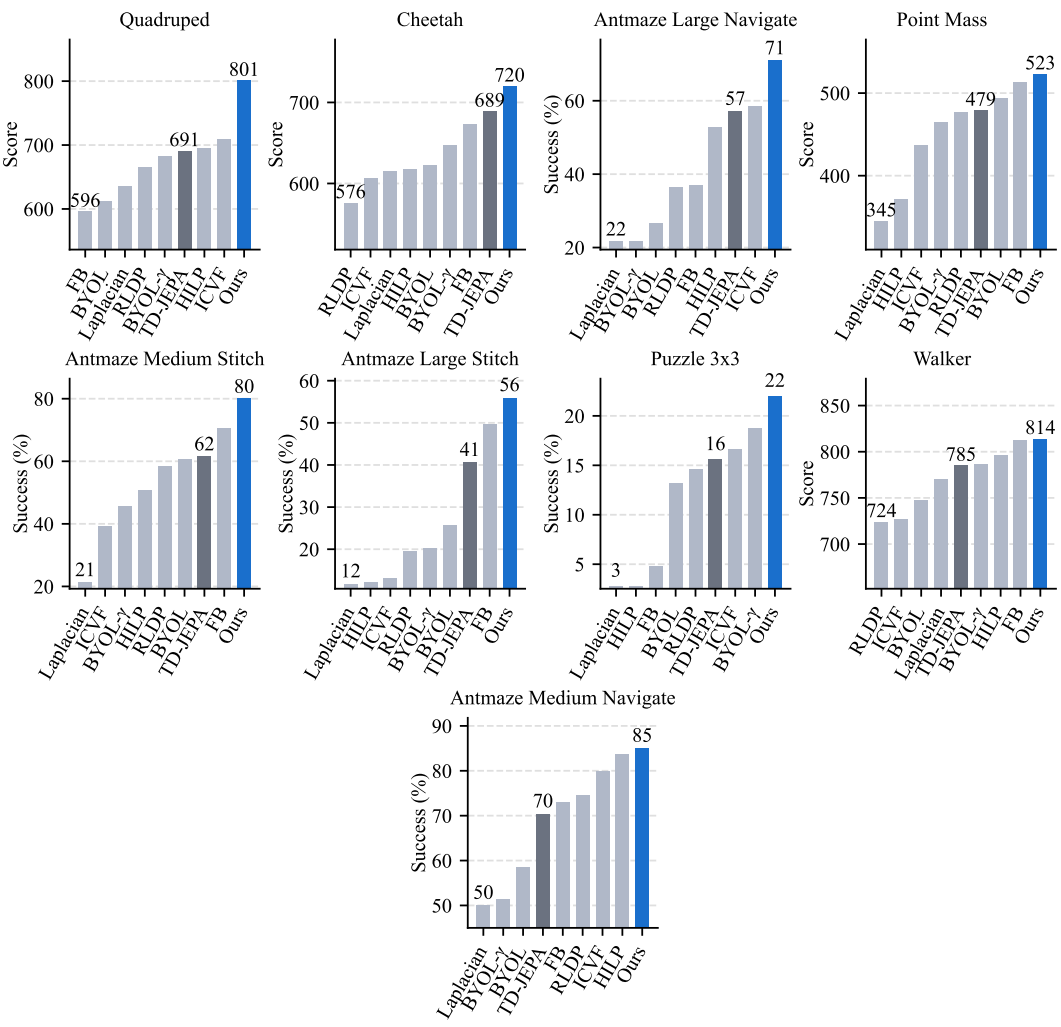

Figure 9: Sparse-TD-JEPA achieves better zero-shot performance than the baseline framework in all tested domains, and outperforms all other established zero-shot baselines in most domains.

D.5 Regularized Latent Dynamics Prediction (RLDP)

Jajoo et al. (2026)

Domain group	lr_b	ortho. coefficient
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{0.001, 0.01\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{0.01, 0.1\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0.001, 0.01, 0.1\}$

Table 6: RLDP hyperparameter grid.

D.6 Bootstrap Your Own Latent (BYOL)

Grill et al. (2020)

Domain group	$\text{lr}_{\text{features}}$	ortho. coefficient
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{0.001, 0.01\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{0.01, 0.1\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0.001, 0.01, 0.1\}$

Table 7: BYOL hyperparameter grid (also used, unmodified, for BYOL-γ).

D.7 BYOL-γ

Lawson et al. (2025) BYOL-γ reuses the same numeric grid as BYOL (above); it differs only in fixed (non-swept) settings, namely multi-step targets enabled (`multi_step=True`) and a discounted future-goal sampling horizon (`data.future`$= 0.99$ OGBench / 0.98 DMC).

D.8 Temporal Difference Joint-Embedding Predictive Architecture (TD-JEPA)

Bagatella et al. (2025)

Domain group	lr_{ψ}	ψ_{ortho}-coefficient
OGBench Antmaze	$\{10^{-4}, 10^{-5}\}$	$\{0.1, 1\}$
OGBench Manipulation	$\{10^{-4}, 10^{-5}\}$	$\{1, 10\}$
DMC	$\{10^{-4}, 10^{-5}\}$	$\{0.01, 0.1, 1\}$

Table 8: TD-JEPA hyperparameter grid.

E Dataset and Evaluation Environment Details

We evaluate our method on 65 tasks across the standard zero-shot reinforcement learning benchmarking frameworks, the *DeepMind Control Suite* (DMC) Tassa et al. (2018); Tunyasuvunakool et al. (2020) and the *Offline Goal-Conditioned RL Benchmark* (OGBench) Park et al. (2025). Task descriptions and dimensions for DMC are listed in detail in Table 9. Task descriptions and dimensions for OGBench are listed in detail in Table 10.

Table 9: DMC dataset domains, dimensions, and tasks.

Domain	State dim.	Action dim.	Tasks	Task Description
Walker	24	6	Stand	Upright torso, minimal height
			Walk	Stand reward plus forward velocity
			Run	Stand reward plus higher velocity
			Spin	Continuous rotation while balanced
Cheetah	17	6	Walk	Reward linear in forward velocity, max 10 m/s
			Walk Backward	Reward linear in backward velocity
			Run	Reward linear in forward velocity, max 10 m/s
			Run Backward	Backward velocity, higher target speed
Pointmass	4	2	Reach Top Left	Navigate to top-left target
			Reach Top Right	Navigate to top-right target
			Reach Bottom Left	Navigate to bottom-left target
			Reach Bottom Right	Navigate to bottom-right target
			Reach Bottom Left Long	Navigate to distant bottom-left target
			Loop	Trace a looped trajectory
			Square	Trace a square trajectory
			Fast Slow	Alternate fast and slow movement segments
Quadruped	78	12	Stand	Maintain balanced standing posture
			Walk	Locomote at moderate speed
			Run	Locomote at higher speed
			Jump	Jump in place

Table 10: OGBench dataset domains, dimensions, and tasks.

Domain	State dim	Action dim	Task	Task Description
Antmaze Medium Navigate	29	8	Task1–5	Navigate to five goals in the medium maze
Antmaze Large Navigate	29	8	Task1–5	Navigate to five goals in the large maze
Antmaze Medium Stitch	29	8	Task1–5	Reach five goals by stitching sub-trajectories
Antmaze Large Stitch	29	8	Task1–5	Reach five goals by stitching sub-trajectories
Antmaze Medium Explore	29	8	Task1–5	Reach five goals from low-quality, high-coverage exploratory data
Cube Single	28	5	Horizontal Vertical1 Vertical2 Diagonal1 Diagonal2	Pick-and-place, horizontal offset Pick-and-place, vertical offset Pick-and-place, vertical offset Pick-and-place, diagonal offset Pick-and-place, diagonal offset
Cube Double	37	5	Single Pick-and-Place Double Pick-and-Place 1 Double Pick-and-Place 2 Swap Stack	Move one cube Move both cubes Move both cubes Exchange cube positions Stack one cube on the other
Scene	40	5	Open Unlock-and-Lock Rearrange Medium Put in Drawer Rearrange Hard	Open drawer or window Toggle button locks Reconfigure objects Unlock, open, place, close Reconfigure, up to 8 steps
Puzzle 3x3	55	5	Task1–5	Press buttons to reach five color targets