VE-JEPA: JOINT LATENT AND VALUE LEARNING FOR OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Predictive representations are typically assessed by how accurately they predict. We examine whether predictive accuracy is informative about downstream control in offline reinforcement learning. We introduce VE-JEPA, a planning-free joint-embedding predictive architecture in which a single encoder is trained through latent prediction, reward prediction, and expectile value learning, with an advantage-weighted policy operating directly on encoded observations. Across three medium-replay locomotion tasks and five seeds per condition, allowing Q and V losses to update the encoder substantially improves control over prediction-only and prediction-plus-reward encoder training. Ablations show that latent prediction improves reward prediction from frozen encoder features, while its incremental control benefit varies across tasks and seeds. Encoder learning rate also affects control performance. Reducing the encoder learning rate tenfold raises mean AntMaze medium-play success from 5.0% to 36.0%, averaged across seven training seeds. Measurements on fixed observations show reduced representation movement despite larger encoder-gradient norms and higher predictive loss. Extensions across locomotion datasets reveal a training-budget tradeoff: slower encoder updates can delay performance improvements at a fixed budget. Across nine D4RL locomotion datasets with ten seeds per method and dataset, VE-JEPA achieves an aggregate interquartile mean normalized return of 74.7 against Decision ConvFormer's 71.5 at 200,000 training steps under fixed-target evaluation, a difference of $+3.2$ with a 95% bootstrap confidence interval of $[+0.4, +5.9]$. This comparison varies across data regimes: VE-JEPA leads by 13.8 points on medium-replay and trails by 8.8 on medium-expert; the overall ordering is sensitive to target-return selection. These results highlight the importance of value-informed encoder training and task-dependent encoder adaptation when learning predictive representations for offline control.

1 INTRODUCTION

Offline reinforcement learning (RL) learns a policy from previously collected transitions. The dataset may contain successful demonstrations, exploratory behaviour, or mixtures of policies with different returns. A useful policy must therefore do more than reproduce the most common actions: it must identify which parts of the available experience support high return. Value estimation and return-conditioned sequence modelling address this problem in different ways, and their relative performance depends on the dataset (Bhargava et al., 2024).

Representation learning introduces a further design choice. An encoder can be trained before policy learning, or adapted together with the control objective. Predicting future representations is attractive because it avoids reconstructing every observation detail (Assran et al., 2023; Bardes et al., 2024). However, predictability alone does not establish that the learned features preserve the information needed for reward prediction and value estimation. This distinction is particularly relevant on state-based benchmarks, where the original observation is already a compact description of the system.

To investigate these choices, we introduce **VE-JEPA**, a joint-embedding predictive architecture trained with reward and value objectives. Its encoder and latent dynamics model are optimized together with twin critics and an expectile value function. A separate policy is fitted to dataset actions

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.

using advantage weights. The policy acts directly on the encoded observation, without planning at test time. This design combines established latent-model and implicit Q-learning components (Hansen et al., 2024; Kostrikov et al., 2022; Chitnis et al., 2023); our focus is the empirical behaviour of this combination and the choices that affect it.

Our experiments examine how this joint model performs across data regimes, which objectives should train its encoder, and how encoder learning rate affects control. At the standard budget, VE-JEPA outperforms the return-conditioned sequence baselines on medium-replay data, while the ordering reverses on medium-expert data. Longer training narrows the latter gap.

A controlled comparison within VE-JEPA shows that allowing Q and V losses to update the encoder substantially improves performance over prediction-only and prediction-plus-reward encoder training on three medium-replay tasks. Reward prediction alone does not recover the full model's performance. Conversely, removing latent prediction yields smaller, task-dependent changes in control. Evaluating frozen encoder features shows that predictive supervision improves the accessibility of reward-related information, most consistently over short horizons. On AntMaze medium-play, a tenfold reduction in encoder learning rate improves success on seven additional seeds, confirming the earlier exploratory sweep. Measurements on fixed observations show less representation movement, despite larger encoder-gradient norms and higher predictive loss. The rate intervention is task dependent: slower updates reduce umaze success and can delay progress on medium-expert data. Separate trajectory-model and pixel studies provide supporting evidence about the limitations of predictive fit, without substituting for these within-model comparisons.

Contributions.

1. We introduce and evaluate VE-JEPA, a planning-free architecture combining latent prediction, reward and value learning, and an advantage-weighted policy. A ten-seed study on nine locomotion datasets reveals opposing performance differences across replay and expert-data regimes that aggregate scores obscure.
2. We examine the contributions of value supervision and latent prediction within VE-JEPA. Joint Q and V supervision substantially improves control over restricted encoder objectives; prediction-removal experiments and frozen-feature evaluations identify additional, task-dependent benefits of predictive supervision.
3. We characterize encoder learning-rate sensitivity through task and budget extensions, additional-seed AntMaze confirmation, and representation-dynamics measurements. Reduced feature movement accompanies improved medium-play success, while predictive loss and effective rank do not consistently track control performance.

2 Related Work

Sequence modelling for offline control. Decision Transformer (DT) (Chen et al., 2021) conditions action prediction on a desired return. Trajectory Transformer (Janner et al., 2021) models trajectory distributions and uses search for control. Decision ConvFormer (DC) (Kim et al., 2024) replaces attention with local filtering in a sequence policy. Our DT and DC comparisons examine return-conditioned policies across medium, medium-replay, and medium-expert data, following the regime-dependent perspective of Bhargava et al. (2024).

Predictive representations. Joint-embedding predictive architectures learn by predicting a target representation rather than reconstructing the input (Assran et al., 2023; Bardes et al., 2024). In RL, SPR (Schwarzer et al., 2021) uses multi-step latent prediction as an auxiliary objective, while masked trajectory models (Wu et al., 2023) learn from partially observed trajectories. TD-JEPA (Bagatella et al., 2026) learns long-term, policy-conditioned predictive representations from reward-free data for zero-shot RL. That setting differs from ours: VE-JEPA has access to the training task's rewards. Our experiments do not establish whether pretraining is beneficial when rewards are unavailable or downstream tasks change.

Latent models and value learning. TD-MPC2 (Hansen et al., 2024) jointly learns latent dynamics, rewards, and values for online model predictive control. IQL-TD-MPC (Chitnis et al., 2023) combines

implicit Q-learning and hierarchical model predictive control. VE-JEPA uses a related latent-model formulation with an IQL-style value objective (Kostrikov et al., 2022), but the evaluated policy does not perform model predictive control. We claim an empirical study of this configuration, rather than novelty for its individual losses. For evaluation, we follow Agarwal et al. (2021) in reporting aggregate uncertainty and examining the sensitivity of conclusions to the reporting protocol.

3 METHOD

3.1 JOINT REPRESENTATION AND VALUE LEARNING

Let $\mathcal{D}$ contain transitions $(s_t, a_t, r_t, s_{t+1}, \delta_t)$, where δ_t indicates a true termination. VE-JEPA consists of an encoder f_θ, latent dynamics model g_ϕ, reward predictor R, twin critics Q_1, Q_2, value function V, and deterministic policy π. The name denotes the value-oriented joint-embedding predictive model used in this study; we do not establish a formal value-equivalence guarantee.

For a sampled sequence of length H, the model starts from the encoded observation and recursively predicts subsequent latents:

$$\hat{z}_0 = f_\theta(s_0), \qquad \hat{z}_{t+1} = g_\phi(\hat{z}_t, a_t), \qquad \bar{z}_{t+1} = \mathrm{sg}[f_\theta(s_{t+1})]\,. \tag{1}$$

Here sg stops gradients. The target uses the current encoder with shared weights, not an exponential-moving-average (EMA) encoder. Dataset actions drive the rollout. With validity mask m_t and rollout weight ρ^t, the predictive, reward, and critic losses are

$$\mathcal{L}_{\text{pred}} = \mathbb{E}_{\mathcal{D}}\left[\sum_{t=0}^{H-1} \rho^t m_t \|\hat{z}_{t+1} - \bar{z}_{t+1}\|_2^2 / d_z\right], \tag{2}$$

$$\mathcal{L}_R = \mathbb{E}_{\mathcal{D}}\left[\sum_{t=0}^{H-1} \rho^t m_t \big(R(\hat{z}_t, a_t) - r_t\big)^2\right], \tag{3}$$

$$\mathcal{L}_Q = \mathbb{E}_{\mathcal{D}}\left[\sum_{t=0}^{H-1} \rho^t m_t \sum_{i=1}^{2} \big(Q_i(\hat{z}_t, a_t) - y_t\big)^2\right], \tag{4}$$

$$y_t = \mathrm{sg}[r_t + \gamma(1-\delta_t)V(\bar{z}_{t+1})]\,, \tag{5}$$

where d_z is the latent dimension. The division by d_z matches the implementation's mean squared error over latent coordinates. Masked positions contribute zero; the implementation averages over the full batch.

The value function is fitted at the initial encoded state using target critics Q_1^-, Q_2^-:

$$q_0^- = \mathrm{sg}\Big[\min_i Q_i^-(f_\theta(s_0), a_0)\Big]\,, \tag{6}$$

$$\mathcal{L}_V = \mathbb{E}_{\mathcal{D}}\big[\ell_\tau\big(q_0^- - V(f_\theta(s_0))\big)\big]\,, \qquad \ell_\tau(u) = |\tau - \mathbb{I}[u<0]|u^2. \tag{7}$$

The joint objective is $\mathcal{L} = \lambda_{\text{pred}}\mathcal{L}_{\text{pred}} + \lambda_R\mathcal{L}_R + \lambda_Q\mathcal{L}_Q + \mathcal{L}_V$. Reward, critic, value, and predictive losses update the encoder. Target critics are updated by exponential averaging. The value loss uses encoded initial states, while the rollout critics also use predicted latents; bootstrap values are evaluated on encoded next states. Stopping gradients through these targets does not prevent them from changing as the shared encoder is updated.

3.2 POLICY EXTRACTION AND ENCODER UPDATES

The policy minimizes advantage-weighted regression on dataset actions:

$$\mathcal{L}_\pi = \mathbb{E}_{\mathcal{D}}\big[w(s,a)\,\|\pi(\mathrm{sg}[f_\theta(s)]) - a\|_2^2 / d_a\big]\,, \quad w(s,a) = \mathrm{sg}\Big[\min\{e^{\beta A(s,a)}, w_{\max}\}\Big]\,, \tag{8}$$

with $A(s,a) = \min_i Q_i^-(f_\theta(s), a) - V(f_\theta(s))$ and action dimension d_a. The policy has a separate optimizer. Its gradient does not reach the encoder. As in IQL, the value targets and policy weights use dataset actions; this avoids maximizing over unseen actions.

We vary the encoder step size through $\eta_{\text{enc}} = \alpha\eta$, keeping the other optimizer settings fixed. The default is $\alpha = 1$; the AntMaze study also tests $\alpha \in \{0, 0.01, 0.1\}$ and freezing after a warm-up. Learning-rate ratios are interventions on optimization, not direct measurements of the rate of representation change.

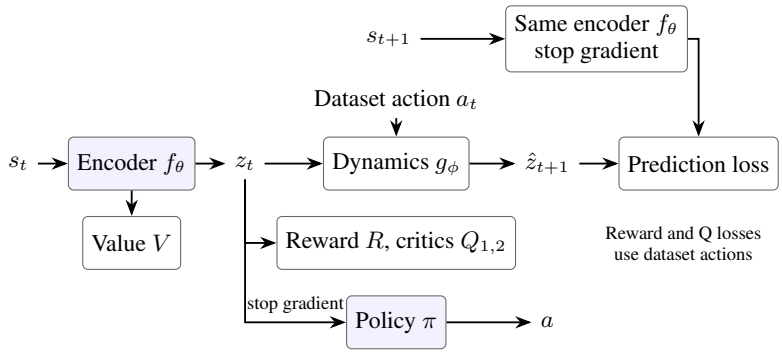


Figure 1: **VE-JEPA training structure.** Solid arrows show forward computation; one latent prediction step is shown. Prediction targets use the same encoder with stopped gradients. Reward and Q losses also train the latent rollout; V is trained on encoded states. The policy uses detached features and detached advantage weights (not shown), so its loss does not update the encoder. Only the encoder and policy are used to act at evaluation. Section 3 defines the complete multi-step objectives and value targets.

4 Experimental Setup

Tasks and baselines. We evaluate hopper, halfcheetah, and walker2d on D4RL medium, medium-replay, and medium-expert datasets (Fu et al., 2020). Medium-replay contains experience collected during policy training; medium-expert mixes intermediate and expert behaviour. We compare VE-JEPA with DT, DC, and IQL. We train DC using its original released implementation with fixed target returns in the primary comparison. Appendix E reports target-return sensitivity and an additional convolutional sequence baseline. AntMaze experiments cover umaze, medium-play, and large-play, and report success percentages. These tasks examine sparse-reward navigation in addition to locomotion.

Budgets and independent runs. The main locomotion comparison uses 200,000 gradient steps and ten seeds (0–9) per method and dataset. Each run is evaluated at 20 equally spaced checkpoints, with ten episodes per checkpoint. The exploratory AntMaze sweep uses three seeds per configuration, with 20 episodes per checkpoint for VE-JEPA and IQL and 30 for DT and DC. Supporting trajectory-model protocols appear in Appendix D. The longer-budget medium-expert experiment uses five seeds and one million steps. The encoder-rate extension uses ten seeds on each of the nine locomotion datasets at 200,000 steps. Equal gradient-step budgets do not imply equal compute or equal numbers of sampled transitions: the model architectures, batch sizes, and rollout lengths differ.

The encoder-objective comparison uses five additional seeds (10–14) per condition on each medium-replay task. The AntMaze medium-play confirmation uses seven additional seeds (3–9) at each of the two encoder rates, with the selected $0.1\times$ rate fixed before these runs. Both studies use 200,000 steps and the same episode counts as their corresponding VE-JEPA experiments. These cohorts are reported separately from the original benchmark and sweep.

The prediction-removal comparison uses the same five locomotion seeds and reuses their full-objective controls. We also freeze the final encoders and fit small networks to predict discounted reward sums from their features and recorded action sequences. Evaluation trajectories are excluded from fitting these networks, but were available during encoder training. Appendix B specifies this evaluation and its robustness checks.

Scores and uncertainty. A run's score is the mean return at its final three evaluation checkpoints, normalized using the D4RL reference returns. This summarizes late-training performance; it is not the return of a single selected deployment checkpoint. Dataset rows report the mean and sample standard deviation across seeds. Locomotion aggregates use a pooled interquartile mean (IQM), implemented by removing $\lfloor n/4 \rfloor$ observations from each tail of the sorted sample. For the original

Table 1: D4RL locomotion at 200,000 gradient steps, ten seeds per cell. Dataset rows report mean ± sample standard deviation of last-three-checkpoint scores; regime and overall rows report pooled IQM. We train DC using its original released implementation with fixed conditioning targets. Bold marks the largest mean or IQM in each row, including ties at the reported precision, not statistical significance.

Dataset	DT	DC	IQL	VE-JEPA
hopper-medium	61.1 ± 3.6	**65.3** ± 4.8	53.7 ± 9.2	56.8 ± 6.3
halfcheetah-medium	42.2 ± 0.5	42.5 ± 0.4	**46.5** ± 0.1	46.0 ± 0.2
walker2d-medium	68.7 ± 3.6	73.2 ± 2.0	80.4 ± 1.2	**81.6** ± 2.3
IQM medium	58.1	**62.1**	57.6	59.0
hopper-medium-replay	59.3 ± 12.0	72.2 ± 7.3	82.9 ± 15.7	**90.5** ± 9.0
halfcheetah-medium-replay	34.9 ± 2.0	40.2 ± 0.7	44.2 ± 0.4	**44.4** ± 0.4
walker2d-medium-replay	61.0 ± 6.1	58.9 ± 7.2	**79.1** ± 6.3	74.9 ± 6.1
IQM medium-replay	52.1	56.5	68.6	**70.2**
hopper-medium-expert	103.8 ± 4.6	**109.2** ± 2.0	41.8 ± 37.7	81.5 ± 20.6
halfcheetah-medium-expert	87.7 ± 1.0	**91.7** ± 0.7	78.9 ± 3.3	81.3 ± 4.3
walker2d-medium-expert	**107.8** ± 0.2	96.8 ± 13.5	106.3 ± 5.9	107.6 ± 1.8
IQM medium-expert	**101.6**	100.5	84.4	91.7
IQM, all nine datasets	67.6	71.5	68.3	**74.7**

benchmark and supporting comparisons, we use 20,000 bootstrap replicates, resampling seeds independently for each method within each dataset. These are stratified, unpaired intervals; they are not simultaneous confidence bounds. Matched control comparisons use 100,000 paired bootstrap replicates; representation evaluations enumerate all 5^5 resamples of the five seed pairs. Neither checkpoints nor episodes are independent training runs. An interval containing zero does not establish equivalence. AntMaze aggregates are arithmetic means across the three mazes.

Supporting studies. An observation-perturbation study appends independent or autoregressive reward-irrelevant coordinates to hopper and walker2d medium-replay states. It uses three seeds and a separate 100-episode evaluation of the final checkpoint (Appendix H). Pixel diagnostics use a separate CNN-based offline control pipeline, not a visual implementation of VE-JEPA (Appendix I).

5 Results and Analysis

5.1 Performance across data regimes

VE-JEPA attains an overall IQM of 74.7 (Table 1). Its differences from DT and IQL are $+7.1$ $[+4.1, +9.9]$ and $+6.4$ $[+2.7, +9.7]$ points, respectively. The difference from DC is $+3.2$ $[+0.4, +5.9]$ under the fixed-target, last-three protocol. Target-return selection changes this ordering at exact checkpoints (Appendix E); the aggregate advantage is protocol dependent.

The regime breakdown clarifies this result (Figure 2). On medium data, the IQMs range from 57.6 to 62.1. On medium-replay, VE-JEPA reaches 70.2, compared with 52.1 for DT, 56.5 for DC, and 68.6 for IQL. The difference from IQL is $+1.6$ with a bootstrap interval spanning zero. On medium-expert, VE-JEPA reaches 91.7 and trails DC's 100.5 by 8.8 points, although it outperforms DC on walker2d-medium-expert. IQL obtains 84.4, with substantial seed variability on hopper-medium-expert. These results are consistent with the usefulness of value estimation on replay data and the strength of return-conditioned imitation when expert trajectories are available; they do not directly measure trajectory stitching.

Training budget. The learning curves show that a fixed budget can compare methods at different stages of optimization (Appendix Figure 8). In particular, VE-JEPA is still improving on hopper-medium-expert at 200,000 steps. A separate five-seed experiment extends all three medium-expert datasets to one million steps. VE-JEPA then reaches an IQM of 105.4, compared with 98.8 for fixed-target DC; the slower-encoder configuration reaches 105.1. DC's deficit is concentrated in walker2d and largely recovers with a higher target return. These results suggest that training budget

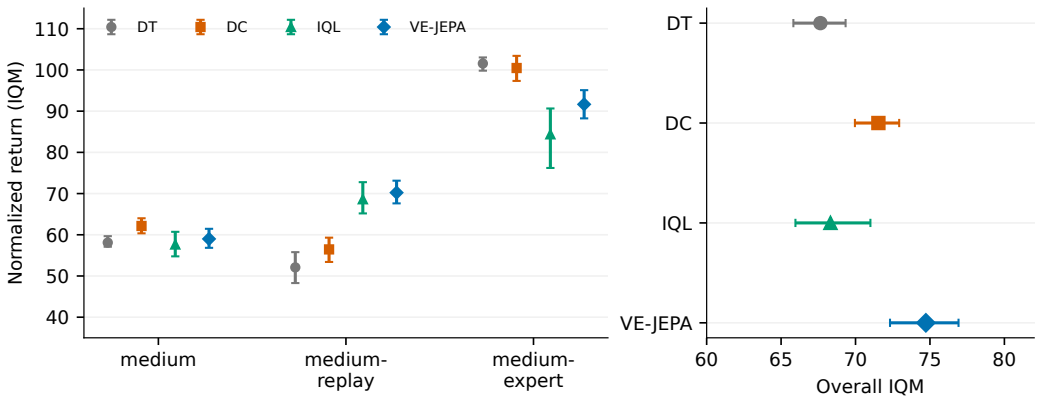


Figure 2: **Regime-specific and overall performance.** Points show pooled IQM; intervals are 95% stratified bootstrap intervals over ten seeds per dataset. Medium-expert favours the sequence baselines at this budget, while medium-replay favours VE-JEPA and IQL. Marginal intervals in this plot do not replace the bootstrap intervals for differences reported in the text.

Table 2: **Effect of encoder training objectives.** Medium-replay normalized returns at 200,000 steps, five additional seeds per cell. Entries are mean ± sample standard deviation of last-three scores. Conditions differ in the objectives supervising the encoder, not in which heads are trained.

Encoder objectives	Hopper	HalfCheetah	Walker2d
Prediction	10.8 ± 4.0	21.4 ± 15.7	7.7 ± 2.8
Prediction + reward	9.5 ± 2.5	33.0 ± 6.5	9.2 ± 3.7
Prediction + reward + Q + V	**93.9** ± 10.1	**44.3** ± 0.4	**82.5** ± 9.0

and target-return selection both affect this comparison. The five-seed extension is reported separately in Appendix E.

5.2 Effect of encoder training objectives

We vary which objectives supervise the encoder while retaining the architecture, all component losses, and the optimization procedure. The encoder receives prediction gradients, prediction and reward gradients, or the full combination including Q and V gradients. All heads continue training in each condition; the policy reads detached features. Appendix A specifies the intervention and its interaction with the common gradient-norm constraint.

Value-learning supervision improves return on all three tasks (Table 2). Compared with prediction and reward supervision, the full encoder objective increases mean return by 84.4 points on Hopper, 11.3 on HalfCheetah, and 73.3 on Walker2d. It outperforms both restricted objectives on every matched seed. These results show that reward prediction alone does not recover the performance obtained when Q and V losses also supervise the encoder. This comparison measures their joint contribution to encoder training.

5.3 Contribution of latent prediction

We set $\lambda_{\text{pred}} = 0$ while retaining the encoder, latent dynamics, reward and value objectives, and policy learning. Unlike the preceding encoder-supervision intervention, this removes predictive supervision from the joint model. Full VE-JEPA exceeds this condition by 1.82, 0.16, and 9.43 mean last-three-checkpoint points on Hopper, HalfCheetah, and Walker2d, respectively. The differences vary across seeds; the Walker2d estimate is particularly sensitive to one pair (Appendix Table 5).

Frozen-feature evaluation measures how accurately reward sums can be predicted from the learned representations. With 10,000 fitting updates, predictive supervision reduces one- and three-step reward-return error in every matched training-seed pair on both trajectory partitions. At ten steps, the

Table 3: **Reward prediction from frozen representations.** Ten-step discounted reward-sum prediction, original partition, 10K fitting updates; three regression initializations averaged within each of five encoder seeds. Bold marks higher observed R^2. Paired uncertainty and robustness: Appendix Table 6.

Task	Without prediction: R^2	Full VE-JEPA: R^2	MSE reduction
Hopper	0.99648	**0.99692**	12.6%
HalfCheetah	0.98989	**0.99018**	2.9%
Walker2d	0.97799	**0.97831**	1.4%

mean reduction is 12.6%, 2.9%, and 1.4% on the original partition (Table 3) and remains positive on the second. The original-partition Walker2d interval spans zero. These results support improved reward prediction from frozen features, with smaller and task-dependent gains at longer horizons.

5.4 Encoder learning-rate sensitivity

With the default encoder learning rate, VE-JEPA reaches 77.2% success on umaze, but only 3.9% on medium-play and 0.0% on large-play. IQL reaches 15.0%, 48.9%, and 18.3%, respectively, at the same gradient-step budget. Replacing the learned encoder with an identity map yields 40.0% on medium-play. This control also removes latent normalization from the dynamics output to match the state space, so it is not an encoder-only architectural intervention.

The learning-rate sweep provides a more direct comparison within VE-JEPA (Figure 3). Mean medium-play success is 0.0%, 28.3%, 40.0%, and 3.9% at encoder multipliers 0, 0.01, 0.1, and 1. Freezing after 20,000 or 50,000 steps yields 3.3% and 2.8%, respectively. Thus, slowing continued adaptation performs better than either of these freezing schedules. Turning off the predictive loss also fails to recover performance in the tested configuration. That result shows that removing the auxiliary loss alone is insufficient; it does not rule out interactions between that loss and encoder optimization.

Applying the $0.1\times$ multiplier to the other mazes yields 48.9%, 40.0%, and 28.9% success, an arithmetic mean of 39.3%. The default mean is 27.0%, and IQL's is 27.4%. The improvement on the longer mazes comes with lower umaze performance. This is an exploratory three-seed result: the multiplier was selected on medium-play rather than on held-out validation data.

Confirmation with additional seeds. Holding the selected multiplier fixed, we repeat the default and $0.1\times$ conditions on seven additional seeds. Mean success is $5.0\% \pm 4.6$ with the default rate and $36.0\% \pm 5.7$ with the slower encoder. The paired difference is $+31.0$ percentage points $[+28.3, +34.3]$. The improvement persists across all seven additional seed pairs (Figure 3). The original sweep remains the source of hyperparameter selection; its three-seed results are not pooled with this cohort.

The locomotion extension reinforces this qualification (Appendix Table 13). Across ten seeds per dataset at 200,000 steps, the slower encoder changes IQM from 59.0 to 60.3 on medium, 70.2 to 71.4 on medium-replay, and 91.7 to 85.4 on medium-expert. Overall IQM decreases from 74.7 to 71.5. At one million steps, the separate five-seed medium-expert comparison reaches 105.4 with the default rate and 105.1 with the slower rate. Slowing the encoder therefore has a budget-dependent cost.

Detailed encoder-rate results are reported in Appendix G.

5.5 Representation dynamics during training

We measure encoder features on 4,096 fixed observations and objective-specific encoder gradients on a fixed sequence batch every 1,000 updates. We also separate changes in value outputs into an encoder component, evaluated with the current value head, and a head component evaluated on previous features (Appendix A). These are output changes, not parameter update norms or a direct decomposition of Bellman targets.

Over steps 101,000–200,000, the slower encoder reduces mean feature drift by 77% and encoder-induced value-output change by 55%; both quantities decrease in every matched seed pair. Changes attributable to the value head decrease much less, and only in four of seven pairs. This supports a

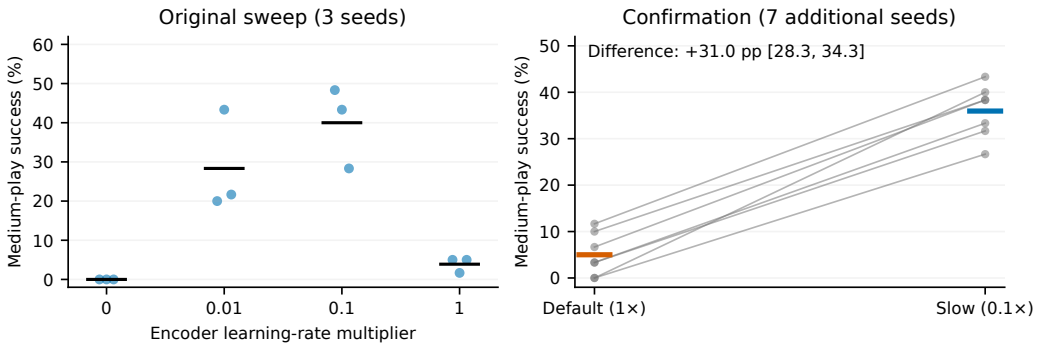


Figure 3: **Discovery and confirmation of encoder-rate sensitivity.** Left: original three-seed sweep, with individual scores and mean bars; the $0.1\times$ rate was selected here. Right: seven additional matched seed pairs at the selected and default rates. Colored bars show means; the interval is a paired-seed 95% bootstrap interval for their difference. All scores average the final three evaluations. Cohorts are kept separate.

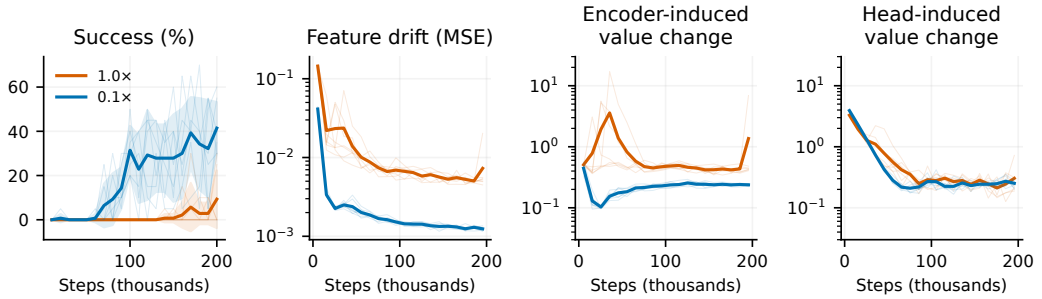


Figure 4: **Training dynamics on additional AntMaze seeds.** Curves compare default and $0.1\times$ encoder rates. Diagnostics average nonoverlapping 10,000-step windows, excluding initialization; success uses scheduled evaluations. Thick lines show seed means and faint lines show individual seeds. Success shading is one seed standard deviation. Diagnostic axes are logarithmic; the two value-change panels share the same vertical scale. All observed windows and seeds are retained.

change in representation movement, without establishing that reduced drift mediates the success improvement.

The slower encoder nevertheless has larger pre-clipping encoder-gradient norms, higher predictive and value losses, and lower effective rank in all seven late-training comparisons. Raw gradient magnitude therefore does not describe feature adaptation, and lower loss or higher rank is not a sufficient account of the performance ordering. In the locomotion objective comparison, full training also reduces feature movement, but does not uniformly reduce encoder-induced value-output changes. The diagnostic pattern is task dependent.

Supporting studies. Separate T-JEPA studies distinguish policy from critic representations (Appendix D); observation perturbations show task-dependent sensitivity to added coordinates (Appendix H). In the pixel pipeline, copying the current Breakout latent captures most predictive-loss reduction (Appendix I), motivating checks of objective informativeness.

6 Discussion and Limitations

6.1 Implications for representation learning

The encoder-supervision experiments distinguish reward prediction from value-informed representation learning. Although both provide task-related supervision, adding reward prediction to the predictive encoder objective does not recover the full model's performance on Hopper and Walker2d. Allowing Q and V losses to update the encoder produces substantial improvements across all three medium-replay tasks. This suggests that, in these settings, learning features for immediate reward prediction is insufficient to support the downstream value learner as effectively as joint value supervision.

Improved reward prediction from frozen features persists under longer regression fitting and a second trajectory partition. The control gains are smaller than those from value supervision. Walker2d has the largest mean control difference but more variable ten-step decoding effects, so these evaluations do not establish a mechanism mediating policy return.

The encoder-rate experiments examine how quickly a value-informed representation should change during training. On AntMaze medium-play, reducing the encoder learning rate improves success and decreases both feature movement and the encoder's contribution to changes in value outputs. These reductions occur despite larger pre-clipping encoder-gradient norms. Gradient magnitude therefore does not directly describe representation movement.

Together, these results connect representation quality to encoder supervision, adaptation, and training budget. Predictive loss and effective rank alone do not explain control performance: a slower encoder can improve success despite higher loss, yet require additional training on other tasks.

6.2 Evaluation protocol

Our last-three-checkpoint rule avoids maximizing over test evaluations, but summarizes several late policies rather than a selected deployment policy. Best-checkpoint reporting changes the ten-seed VE-JEPA–DT aggregate gap from 7.1 to 8.3 points (Appendix F). A deployment comparison would require an explicit selection rule and independent evaluation. Likewise, aggregate IQM does not replace dataset-level results or establish a preference for VE-JEPA on every data regime.

6.3 Limitations

The evidence covers state-based locomotion and three AntMaze datasets; pixel experiments use a separate pipeline. Additional-seed confirmation strengthens medium-play evidence but does not remove task-specific hyperparameter selection. Q and V supervision is tested jointly under shared gradient clipping, while prediction removal affects both encoder and dynamics. Reward decoding evaluates trajectories excluded from regression fitting, not encoder training, and remains conditional on its fitting budget. These evaluations do not establish causal mediation of control gains. Feature drift depends on representation scale, and the value-output decomposition is order dependent. Longer training does not establish asymptotic convergence, nor do equal step budgets establish wall-clock efficiency. Generalization beyond the reported tasks and methods remains open.

7 Conclusion

VE-JEPA combines predictive and value learning for offline control. Value supervision improves control, while latent prediction improves reward prediction from frozen features with task-dependent control gains. Slower encoder updates improve AntMaze medium-play success, with task- and budget-dependent effects. These results motivate evaluating representations through both predictive fit and downstream control.

AI use statement

We used generative AI tools primarily for manuscript editing, literature discovery, and assistance with output verification and statistical checks. We take responsibility for the final work.

REFERENCES

R. Agarwal et al. Deep reinforcement learning at the edge of the statistical precipice. *NeurIPS*, 2021.

M. Assran et al. Self-supervised learning from images with a joint-embedding predictive architecture. *CVPR*, 2023.

M. Bagatella et al. TD-JEPA: Latent-predictive representations for zero-shot reinforcement learning. *ICLR*, 2026.

A. Bardes et al. Revisiting feature prediction for learning visual representations from video. *TMLR*, 2024.

P. Bhargava et al. When should we prefer Decision Transformers for offline reinforcement learning? *ICLR*, 2024.

L. Chen et al. Decision transformer: Reinforcement learning via sequence modeling. *NeurIPS*, 2021.

R. Chitnis et al. IQL-TD-MPC: Implicit Q-learning for hierarchical model predictive control. *arXiv:2306.00867*, 2023.

J. Fu et al. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv:2004.07219*, 2020.

N. Hansen, H. Su, and X. Wang. TD-MPC2: Scalable, robust world models for continuous control. *ICLR*, 2024.

M. Janner, Q. Li, and S. Levine. Offline reinforcement learning as one big sequence modeling problem. *NeurIPS*, 2021.

J. Kim et al. Decision ConvFormer: Local filtering in MetaFormer is sufficient for decision making. *ICLR*, 2024.

I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit Q-learning. *ICLR*, 2022.

M. Schwarzer et al. Data-efficient reinforcement learning with self-predictive representations. *ICLR*, 2021.

P. Wu et al. Masked trajectory models for prediction, representation, and control. *ICML*, 2023.

APPENDIX

A ENCODER SUPERVISION AND CONFIRMATION PROTOCOL

The new comparisons use the same state encoder, latent rollout, heads, batch size, loss weights, and training budget as the corresponding default VE-JEPA configuration. The locomotion comparison uses seeds 10–14, encoder multiplier 1, and three encoder objectives: prediction; prediction plus reward; and prediction plus reward plus Q and V. AntMaze confirmation uses seeds 3–9, full encoder supervision, and multipliers 1 and 0.1. The original benchmark and exploratory sweep are separate cohorts.

All component losses remain active. We replace only the encoder's accumulated gradient with that of the selected objective before applying the common global gradient-norm constraint of 10. Consequently, this intervention can change the clipping factor applied to other joint-model parameters; identical non-encoder updates are not guaranteed. The detached-feature policy update is unchanged. The ablation estimates the effect of encoder supervision under this shared optimization procedure, not its effect independently of gradient clipping.

Runs share initialization seeds and the data-sampling procedure within each comparison. Synthetic control checks verified the intended encoder gradients, unchanged non-encoder derivatives before clipping, and that diagnostics preserve random states and a reference parameter trajectory. Each final score averages evaluations at 180K, 190K, and 200K. Confidence intervals below resample matched seed pairs 100,000 times; they are descriptive and not adjusted for multiple comparisons.

Table 4: Full encoder supervision minus the restricted encoder objective: mean normalized-return differences and paired-seed 95% bootstrap intervals.

Task	Full minus prediction	Full minus prediction + reward
hopper	$+83.1$ $[+71.8, +91.3]$	$+84.4$ $[+76.0, +90.9]$
halfcheetah	$+22.9$ $[+10.7, +35.1]$	$+11.3$ $[+6.4, +16.6]$
walker2d	$+74.8$ $[+69.4, +79.7]$	$+73.3$ $[+63.9, +82.1]$

Diagnostic measurements. We use 4,096 fixed observations for feature diagnostics and a separate fixed batch of 256 sequences for losses and gradients. Diagnostic sampling does not alter the training random state. Measurements occur in evaluation mode every 1,000 updates. For features $z_t = f_{\theta_t}(s)$, drift is the mean squared change from the preceding measurement. Effective rank is the participation ratio of the centered feature covariance. These measures depend on the learned feature geometry; low rank alone does not establish loss of control-relevant information.

For value head V_t, the signed output change decomposes as

$$V_t(z_t) - V_{t-1}(z_{t-1}) = [V_t(z_t) - V_t(z_{t-1})] + [V_t(z_{t-1}) - V_{t-1}(z_{t-1})].$$

We report the mean absolute magnitude of each component. Absolute components need not sum to absolute total change. The decomposition is order dependent and does not directly measure movement of Bellman targets. TD RMSE uses the first transition in each diagnostic sequence; the training Q loss also includes masked latent-rollout positions. Objective-specific encoder-gradient norms include the training loss weights and are measured before clipping. Excluded objectives still have diagnostic derivatives but do not supervise the encoder. Neither value-head parameter gradients nor Adam update norms were logged.

Scope of the diagnostic findings. Late-window summaries average steps 101K–200K within each seed, then across seeds. Q-loss gradients have the largest mean encoder-gradient norm in the full configurations, but norms do not establish causal importance or gradient alignment. AntMaze's slower encoder has larger raw encoder gradients but less feature movement. It also has higher predictive and value losses and lower rank in every matched late-window pair. In locomotion, full supervision instead has higher rank than the restricted conditions. These observations do not support a universal loss- or rank-based explanation of control performance.

B CONTRIBUTION OF PREDICTIVE SUPERVISION

B.1 MATCHED OBJECTIVE-REMOVAL COMPARISON

We compare the full objective with $\lambda_{\text{pred}} = 0$ on the three medium-replay tasks, using seeds 10–14, 200,000 training steps, and the default encoder learning rate. Full-objective runs are the complete five-seed cohorts from the encoder-supervision study; all are reused without selection by score. Architecture, reward scaling, data processing, and remaining hyperparameters are matched. Removing prediction disables its gradients to both encoder and dynamics. The dynamics still train through rollout reward and critic losses; this is not a no-dynamics or raw-state baseline. The common gradient-norm constraint is unchanged, so effects include any resulting changes in clipping.

Table 5: **Incremental contribution of latent prediction.** Mean $\pm$ sample standard deviation over five training seeds. Each score averages the 180K, 190K, and 200K evaluations. Differences are full minus without prediction; intervals use matched seed pairs.

Task	Full VE-JEPA	Without prediction	Difference	95% interval
Hopper	93.89 ± 10.12	92.07 ± 5.77	$+1.82$	$[-6.14, +7.85]$
HalfCheetah	44.32 ± 0.35	44.16 ± 0.54	$+0.16$	$[-0.26, +0.62]$
Walker2d	82.48 ± 8.97	73.05 ± 12.13	$+9.43$	$[+0.06, +22.62]$

Prediction yields positive mean differences on all tasks, with intervals spanning zero on Hopper and HalfCheetah. Walker2d's interval is close to zero at its lower endpoint. Its paired differences are $+10.27$, -1.29, $+4.08$, -0.61, and $+34.71$: seed 14 contributes approximately 74% of the net gain. An exploratory exclusion of this pair gives $+3.11$ with interval $[-0.95, +7.55]$; all five pairs remain in the primary result. This heterogeneity motivates retaining individual curves rather than interpreting the mean as a uniform improvement (Appendix Figure 5).

B.2 REWARD PREDICTION FROM FROZEN REPRESENTATIONS

We freeze each final encoder and train a separate nonlinear regression network on its features. Such representation evaluations are also called nonlinear probes. For horizon h, its input is $[f_\theta(s_t), a_t, \ldots, a_{t+h-1}]$ and its target is the observed discounted reward sum $\sum_{j=0}^{h-1} 0.99^j r_{t+j}$, for $h \in \{1, 3, 10\}$. The target follows dataset actions; it is not an optimal value estimate. The network has one 128-unit ReLU hidden layer and a scalar output. Encoder parameters remain fixed. A raw-observation reference uses the same regression architecture, with a different input dimension and hence parameter count.

Trajectories are partitioned 70%/15%/15% for fitting, validation, and testing. We sample at most 30,000/10,000/10,000 sequence starts, respectively, without crossing trajectory boundaries or terminal transitions. All horizons use starts with ten available future rewards, excluding near-terminal starts. Partitions and starts are shared across encoder conditions and training seeds. Features and targets are standardized using fitting data only, after applying the saved encoder normalization. Test trajectories were excluded from regression fitting, not from the original encoder training.

Regression uses Adam with learning rate 10^{-3}, batch size 256, and three matched initializations. Validation MSE is checked every 100 updates; the lowest-validation-error checkpoint within each budget is evaluated on the test partition. We retain the original 2,000-update results and extend fitting to 10,000 updates on both the original partition (A; split seed 20260920) and a second partition (B; seed 20260921). Initialization seeds are 20260920–20260922. The ten-step endpoint was selected before evaluation; one- and three-step results are secondary. The 10K extension and second partition are follow-up robustness checks, not additional independent encoder-training runs.

We average the three regression initializations within each encoder-training seed before comparing the five seed pairs. Confidence intervals in the following tables enumerate all 5^5 empirical paired-bootstrap resamples. These agree with the earlier Monte Carlo estimates to reporting precision and are not adjusted for multiple comparisons. Neither test transitions nor repeated partitions are treated as independent RL runs. Relative error reduction is $100(1 - \overline{\text{MSE}}_{\text{full}}/\overline{\text{MSE}}_{\text{without}})$; absolute R^2 and MSE values are retained to show the size of the residual errors.

Table 6: **Ten-step reward prediction: fitting-budget and partition sensitivity.** Differences favor full VE-JEPA when positive. Both the difference and interval in the last two columns are scaled by 10^3.

Split	Task	Updates	Without: R^2	Full: R^2	MSE reduction	$10^3\Delta R^2$	95% interval
A	Hopper	2K	0.99478	0.99528	+9.51%	$+0.496$	$[+0.384, +0.662]$
A	Hopper	10K	0.99648	0.99692	+12.57%	$+0.443$	$[+0.314, +0.572]$
A	HalfCheetah	2K	0.98882	0.98967	+7.61%	$+0.851$	$[+0.575, +1.177]$
A	HalfCheetah	10K	0.98989	0.99018	+2.92%	$+0.295$	$[+0.093, +0.496]$
A	Walker2d	2K	0.97583	0.97530	-2.17%	-0.524	$[-1.821, +0.584]$
A	Walker2d	10K	0.97799	0.97831	+1.43%	$+0.314$	$[-1.305, +1.569]$
B	Hopper	2K	0.99499	0.99529	+6.16%	$+0.309$	$[+0.043, +0.588]$
B	Hopper	10K	0.99675	0.99700	+7.58%	$+0.246$	$[+0.101, +0.374]$
B	HalfCheetah	2K	0.98826	0.98851	+2.12%	$+0.249$	$[+0.182, +0.323]$
B	HalfCheetah	10K	0.98826	0.98847	+1.73%	$+0.203$	$[+0.089, +0.305]$
B	Walker2d	2K	0.97576	0.97684	+4.42%	$+1.072$	$[+0.620, +1.586]$
B	Walker2d	10K	0.97761	0.97872	+4.99%	$+1.118$	$[+0.724, +1.423]$

At 10K updates, the mean ten-step benefit remains positive for every task and partition. It is consistent across all seed pairs on Hopper; HalfCheetah's benefit attenuates with additional fitting. Walker2d changes sign on partition A and is positive on B, with its A interval still spanning zero. These results support a modest, task-dependent benefit rather than a uniform effect size. At shorter horizons, every paired encoder seed favors prediction on both partitions (Appendix Table 7).

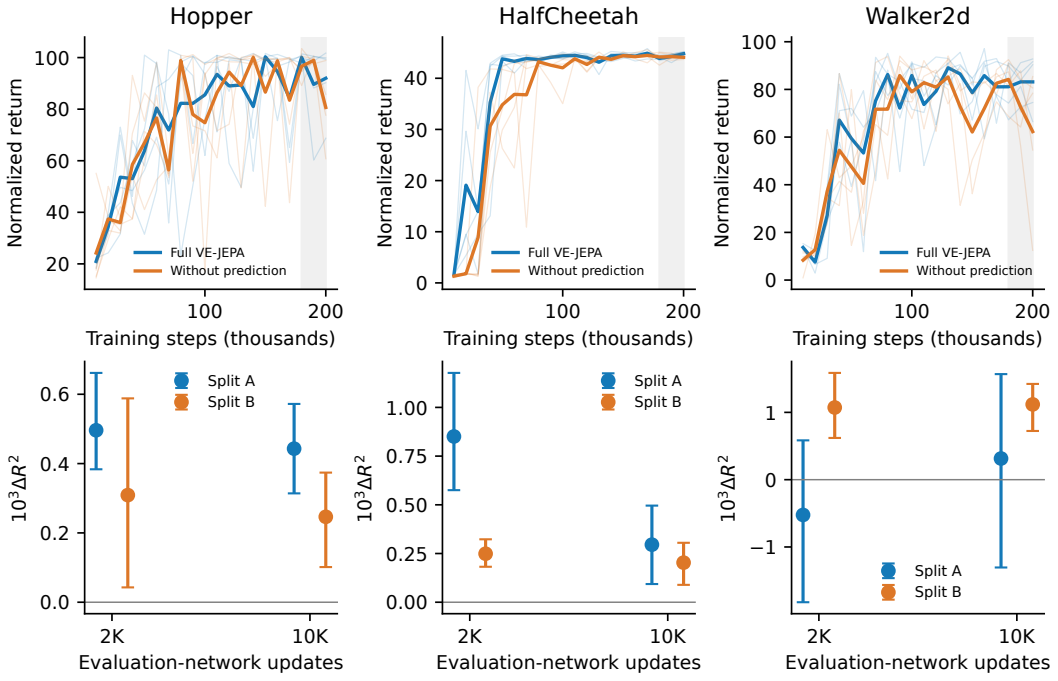


Figure 5: **Control and representation evidence for predictive supervision.** Top: individual training curves (faint) and seed means (thick); shading marks the primary 180K–200K reporting window. Bottom: paired mean ten-step R^2 differences with 95% empirical bootstrap intervals for both fitting budgets and partitions. Positive differences favor full VE-JEPA. Partitions share the same five trained encoder pairs and are not independent cohorts.

The conclusions are conditional on the regression architecture and budget. Hopper often selects checkpoints near the end of 10K fitting; this extension does not establish asymptotic convergence. On the original partition at 2K, Walker2d's ten-step comparison favors prediction if represented test trajectories are weighted equally, rather than weighting sampled starts equally. We retain the latter primary metric; the sensitivity indicates heterogeneous effects across trajectories. Both learned

Table 7: **Reward-return decoding across horizons at 10K fitting updates.** MSE uses the original reward-sum units. Reduction compares full VE-JEPA with without-prediction encoders; wins count paired RL seeds after averaging regression initializations. The raw reference repeats the same three fitted networks across RL seeds and does not constitute five independent representation samples.

Split	Task	Horizon	Raw MSE	Without MSE	Full MSE	Reduction	Wins
A	Hopper	1	0.0004	0.0017	0.0014	+16.29%	5/5
A	Hopper	3	0.0108	0.0190	0.0161	+15.13%	5/5
A	Hopper	10	0.3584	0.2931	0.2563	+12.57%	5/5
A	HalfCheetah	1	0.0256	0.0286	0.0243	+15.17%	5/5
A	HalfCheetah	3	0.3799	0.2500	0.2228	+10.89%	5/5
A	HalfCheetah	10	6.8928	3.7886	3.6779	+2.92%	4/5
A	Walker2d	1	0.0029	0.0111	0.0098	+11.54%	5/5
A	Walker2d	3	0.0988	0.1604	0.1465	+8.66%	5/5
A	Walker2d	10	3.5692	3.2720	3.2252	+1.43%	4/5
B	Hopper	1	0.0004	0.0016	0.0014	+9.52%	5/5
B	Hopper	3	0.0121	0.0191	0.0173	+9.29%	5/5
B	Hopper	10	0.3802	0.3014	0.2786	+7.58%	5/5
B	HalfCheetah	1	0.0228	0.0229	0.0194	+15.59%	5/5
B	HalfCheetah	3	0.3042	0.2205	0.1958	+11.17%	5/5
B	HalfCheetah	10	5.9896	3.9108	3.8430	+1.73%	5/5
B	Walker2d	1	0.0030	0.0115	0.0096	+16.17%	5/5
B	Walker2d	3	0.1064	0.1698	0.1516	+10.71%	5/5
B	Walker2d	10	3.8540	3.4917	3.3173	+4.99%	5/5

encoders exceed the raw reference's mean ten-step R^2 at 10K on both partitions, but the different input dimensions preclude a parameter-count-matched interpretation.

B.3 Final-policy evaluation and predictive diagnostics

We evaluate every final policy on 100 episodes, using the same reset seeds (202609200–202609299) for both conditions and all training seeds within a task. This assesses the 200K checkpoint only and supplements, rather than replaces, the last-three-checkpoint comparison. Seed 14's Walker2d final score is 39.20 without prediction versus 89.26 with prediction, with mean episode lengths 438.58 and 994.05. Its deficit persists beyond the original ten-episode measurement, although its magnitude is smaller. The five training seeds remain the replication units.

Table 8: **Final policies evaluated over 100 episodes each.** Mean $\pm$ sample standard deviation across five training seeds, with paired differences and 95% empirical bootstrap intervals. These are not the primary last-three scores in Appendix Table 5.

Task	Full VE-JEPA	Without prediction	Difference	95% interval
Hopper	90.02 ± 14.07	79.27 ± 16.51	$+10.74$	$[-2.04, +23.53]$
HalfCheetah	44.78 ± 0.43	44.19 ± 0.41	$+0.58$	$[-0.04, +1.21]$
Walker2d	85.32 ± 6.30	69.40 ± 18.29	$+15.92$	$[+0.26, +34.49]$

Within each full-model representation, learned one-step dynamics outperform copying the current latent on the test starts. Mean learned/copy-current MSEs are 0.0264/0.0489 (Hopper), 0.0520/0.6736 (HalfCheetah), and 0.0668/0.0872 (Walker2d) on partition A. These comparisons show predictive structure beyond copying in this setting. Absolute latent errors are not comparable measures of representation quality across independently trained encoders, and the without-prediction dynamics were not optimized to match encoded next states. Neither these diagnostics nor the regression results establish that the measured feature differences mediate the control gains.

C Implementation Details

The training parameters and comparison configurations are described below.

The state encoder has two hidden layers of width 256 with ELU activations, a 128-dimensional output, and output LayerNorm. The dynamics model uses the same hidden width and depth and applies LayerNorm to its predicted latent. Reward, value, and critic heads are MLPs; the policy has a final tanh activation. For the identity-encoder control, dynamics-output LayerNorm is removed to preserve the standardized state representation.

Parameter	Default value
Batch size	256
Latent rollout length H	3
Rollout loss weight ρ	0.5
Discount γ	0.99
Adam learning rate η	3×10^{-4}
Encoder multiplier α	1
Expectile τ	0.7
Advantage temperature β	3
Maximum advantage weight $w_{\max}$	100
Target-critic EMA coefficient	0.005
$(\lambda_{\text{pred}}, \lambda_R, \lambda_Q)$	$(10, 1, 1)$
Joint-model gradient clipping norm	10
Locomotion evaluation episodes per checkpoint	10
AntMaze episodes per checkpoint (VE-JEPA, IQL)	20
AntMaze episodes per checkpoint (DT, DC)	30
AntMaze expectile τ and temperature β	0.9 and 10
Distractor final-checkpoint evaluation episodes	100

The joint optimizer contains the encoder, dynamics, reward predictor, critics, and value function. The policy uses a separate Adam optimizer with the same default learning rate. The encoder learning-rate study changes only the encoder parameter group's multiplier. The reported VE-JEPA configuration uses no planning, bisimulation penalty, or added distractor dimensions in the main benchmark. Appendix H describes the separate observation-perturbation study and an auxiliary penalty configuration. Observations are standardized using dataset statistics. The dataset files use the D4RL v2 naming convention; locomotion evaluation uses the corresponding Gym v3 environments and AntMaze evaluation uses v2. Locomotion rewards are multiplied by $1000/(R_{\max} - R_{\min})$, where $R_{\max}$ and $R_{\min}$ are the maximum and minimum trajectory returns in the dataset. AntMaze training subtracts one from each reward. State standard deviations include an additive 10^{-6} for numerical stability.

DT uses a context length of 20, three layers, embedding dimension 128, a batch size of 64, and a learning rate of 10^{-4}. DC uses context length 8, three layers, GELU, a convolution window of 6, and batch size 64. Its embedding dimension is 256 on hopper-medium and hopper-medium-replay, and 128 elsewhere. Its learning rate is 10^{-4} on these two Hopper tasks, walker2d-medium and AntMaze, and 10^{-3} on the remaining locomotion tasks. DC uses AdamW, weight decay 10^{-4}, gradient clipping at 0.25, and 10,000 warmup steps. IQL uses two hidden layers of width 256, batch size 256, and learning rate 3×10^{-4}. The T-JEPA target-space experiments use a context length of 20 and a three-layer, 128-dimensional trajectory encoder, followed by 100,000 fine-tuning steps. Their pretraining cost is additional to this downstream budget. The saved run configurations identify the corresponding pretraining checkpoints and auxiliary-loss settings.

D SCOPE OF THE REPRESENTATION COMPARISONS

Comparison	Interpretation	Seeds/task
VE-JEPA vs frozen latent critic	Different encoder architectures and training pipelines; not an isolated joint-training intervention	10 vs 3
Latent vs raw target	Target-space comparison within the fine-tuned trajectory pipeline	5 vs 5
Pretrained vs scratch	Trainable policy encoders and raw-state critics; different policy learning rates and additional pretraining compute	3 vs 3
Encoder learning-rate sweep	Parameter-group learning-rate intervention within VE-JEPA on AntMaze medium-play	3 per arm

The bootstrap resamples methods independently, even when runs share a seed index. The frozen/joint comparison has unequal seed counts; its interval reflects sampling variability of these configurations and does not account for architectural confounding. The rank and copying diagnostics do not certify control-relevant information, and should be interpreted together with policy performance.

Trajectory pretraining and downstream gradient flow. The target-space study uses the trajectory Transformer encoder, a context of 20 steps split into 12 context and eight target steps, an EMA teacher, and a cross-attention predictor. Pretraining combines future-feature prediction with an action auxiliary weighted by 0.1. The raw-target arm adds a linear output head for future states and actions. Both arms receive 100,000 pretraining steps followed by 100,000 downstream steps. Appendix Table 9 reports frozen and fine-tuned policy variants; the possible frozen-feature advantage is not resolved by its interval.

Table 9: T-JEPA target-space comparisons on medium-replay, five seeds per environment. Scores are last-three IQMs; differences are latent minus raw.

Policy training	Latent target	Raw target	Difference (95% CI)
Frozen encoder	59.0	52.9	$+6.1$ $[-1.0, +14.3]$
Fine-tuned encoder	60.4	61.4	-1.0 $[-6.2, +4.0]$

In the advantage-weighted policy study, return tokens are zeroed and raw-state critics supply weights to the policy. The critics do not backpropagate through the trajectory encoder; the policy loss updates it in the fine-tuned and from-scratch arms. Their IQMs are 70.2 and 70.5, respectively; retaining frozen pretrained policy features yields 70.6. All three use 200,000 downstream steps, and the pretrained arms incur an additional pretraining stage. Their policy optimizer uses 10^{-3} in downstream mode, versus 10^{-4} in the from-scratch mode; all raw-state critics use 3×10^{-4}. Thus, even with both encoders trainable, this comparison does not isolate initialization. The frozen latent-critic configuration instead feeds trajectory features to Q and V and fits separate one-step latent dynamics and reward heads. It does not reuse the pretrained cross-attention predictor as the recursive dynamics model. Its comparison with VE-JEPA changes several components and is not a matched test of encoder-gradient routing.

D.1 POLICY AND CRITIC REPRESENTATIONS

The frozen T-JEPA latent-critic configuration reaches a medium-replay IQM of 46.9, compared with 70.2 for VE-JEPA. The difference is $+23.3$ $[+19.6, +26.6]$. The frozen configuration uses three seeds per environment, while VE-JEPA uses ten. A frozen pretrained-policy configuration with critics reading raw states reaches 70.6. Frozen features are therefore not uniformly ineffective: the downstream configuration matters.

These comparisons indicate that the frozen trajectory features can be a poor input to the downstream value learner. They do not identify training placement as the sole cause. The frozen configuration uses a trajectory Transformer, whereas VE-JEPA uses a state MLP, a recursive latent rollout, and a different optimization procedure. Their predictive targets and losses also differ. A controlled attribution to joint training would require holding these choices fixed and varying only encoder

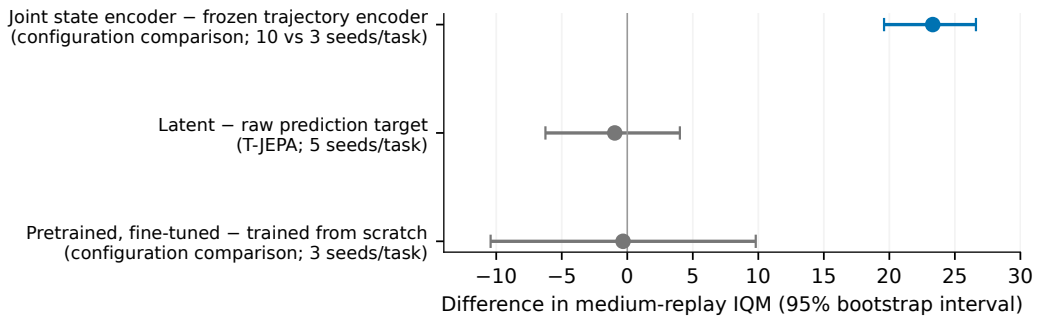


Figure 6: **Three representation comparisons on medium-replay.** Points and intervals show IQM differences and 95% stratified bootstrap intervals. The first row changes the overall configuration. The other rows examine target space and pretrained-policy configurations in separate trajectory-model experiments. The third row also changes the policy-training learning rate. Their seed counts and scopes are shown explicitly; the rows do not form a single factorial ablation of VE-JEPA.

updates. We therefore report the result as a comparison of configurations, not as the effect of moving an identical loss.

D.2 Prediction targets and pretrained policies

In the fine-tuned trajectory-model comparison, latent prediction minus raw-signal prediction gives -1.0 $[-6.2, +4.0]$. In the pretrained-policy comparison, both encoders remain trainable during advantage-weighted policy learning and the critics read raw states. Pretrained and fine-tuned features yield an IQM of 70.2, compared with 70.5 from random initialization: a difference of -0.3 $[-10.4, +9.8]$. Both arms receive 200,000 downstream steps; pretraining is additional compute. The policy optimizer uses a base learning rate of 10^{-3} for fine-tuning and 10^{-4} from scratch, while both critic optimizers use 3×10^{-4}. The result therefore compares training configurations rather than isolating initialization. The frozen-policy result of 70.6 is a separate control.

Neither experiment resolves an advantage for its pretrained or latent-target configuration. The intervals nevertheless allow effects of several normalized points. These results do not demonstrate equivalence, do not cover reward-free transfer, and do not show that VE-JEPA's own predictive loss is unnecessary. They instead limit the evidence for those specific pretraining choices on the medium-replay tasks studied here.

E Decision ConvFormer Implementation and Evaluation

DC implementation and numerical consistency. We train and evaluate DC using the model, sampler, training step and episode evaluator from the original DC repository at commit `3db9b9ec`. We compared dataset normalization and model, optimizer and scheduler construction against the upstream training entry point on hopper-medium, hopper-medium-replay and AntMaze umaze, using two seeds per task. Under identical data and random states, sampled batches, losses, optimizer states and model parameters matched exactly across 20 updates. Evaluation of saved 100K checkpoints also matched exactly at two conditioning targets per case. For AntMaze, the numerical check explicitly seeded the underlying simulator and matched NumPy's global random state. The installed D4RL wrapper does not forward the outer environment seed to the simulator. Our saved AntMaze benchmark scores therefore use stochastic episode starts rather than paired episode starts across target settings; this check does not retroactively change those evaluations. These short checks establish numerical consistency for the tested settings, rather than recovery of every published benchmark score.

Data and reporting choices. All methods use the same stored D4RL v2 trajectory datasets. For medium-expert, we use the supplied mixture rather than the upstream entry point's concatenation and shuffling of separately loaded medium and expert datasets. This data-loading adaptation is outside the numerical check above. The primary DC targets are 3600 for Hopper, 12000 for HalfCheetah, 5000

Table 10: DC target-return sensitivity. Our exact-checkpoint scores use ten seeds and 100 episodes per seed and target; published scores use the original paper's protocol. Selected scores choose one target per task from the tested grid. Values are task means, not pooled IQMs.

Dataset	Published	100K selected	200K default	200K selected
hopper-medium	92.5	72.3	65.3	68.7
halfcheetah-medium	43.0	42.5	42.3	42.3
walker2d-medium	79.2	74.5	72.4	74.1
hopper-medium-replay	94.2	75.1	74.1	74.1
halfcheetah-medium-replay	41.3	40.2	40.3	40.4
walker2d-medium-replay	76.6	61.1	58.5	63.4
hopper-medium-expert	110.4	107.1	108.9	108.9
halfcheetah-medium-expert	93.0	92.1	91.9	92.7
walker2d-medium-expert	109.6	107.7	95.9	108.2

for Walker2d and 1 for AntMaze. Evaluation preserves training random states. The ten-seed primary comparison uses the final three checkpoints at 180K, 190K and 200K updates; the separate five-seed extension uses 900K, 950K and 1M. These are gradient-update budgets, not matched computational costs.

The published DC evaluation searches six initial return targets and reports the highest scores (Kim et al., 2024). Our sensitivity analysis uses the public repository's target multipliers: $\{1, 2, 10, 20\}$ for locomotion and $\{1, 10, 100, 1000, 100000\}$ for AntMaze. Each exact checkpoint is evaluated over 100 episodes per seed and target. In Appendix Table 10, the selected target maximizes the task's mean across seeds, rather than selecting a separate target for every seed. Selection and reporting use the same episodes, so these values are optimistic sensitivities. The published column is a reference under the original protocol, not an additional arm of our experiment.

Target selection materially affects walker2d-medium-expert: its exact 200K mean rises from 95.9 to 108.2, and its exact 1M mean from 93.0 to 107.9. Across all nine exact 200K checkpoints, default-target DC has IQM 71.8, task-selected DC 74.7, and VE-JEPA 74.2. Across the exact 1M medium-expert checkpoints, task-selected DC and VE-JEPA reach 105.0 and 104.9. VE-JEPA's saved exact-checkpoint evaluations use ten episodes per seed, compared with 100 for the DC sensitivity analysis. These comparisons supplement the primary late-training summary; they do not establish a target-independent ordering. On medium-replay, the exact 200K IQMs remain higher for VE-JEPA (64.0) than for task-selected DC (60.1).

The target grid does not recover all published scores, particularly on Hopper and AntMaze umaze. We therefore report our DC runs under the stated protocol, with the remaining reproduction differences explicit. Checkpoint selection and target-return selection are distinct; the published target search does not by itself establish selection of the best training checkpoint.

Convolutional sequence baseline. We additionally evaluate a convolutional sequence baseline (CSB) with compound tokens, context length 20, three layers of width 128, ReLU, kernel size 5 and learning rate 10^{-4}. It differs from DC's interleaved return, state and action tokens and modality-specific depthwise filters. CSB is an additional baseline; these jointly differing architectures and optimization settings do not isolate the effect of an individual DC component. At 1M, pooled IQMs are 105.6 for CSB, 98.8 for fixed-target DC and 105.4 for VE-JEPA. The VE-JEPA–DC difference is $+6.6$ with a 95% bootstrap interval of $[+4.3, +9.6]$; the target-return analysis above qualifies its interpretation. On AntMaze, fixed-target DC reaches mean success rates of 56.7%, 0% and 0% on umaze, medium-play and large-play. Target search yields at most 1.3% mean medium-play success at the tested 100K/200K checkpoints and zero on large-play. These three-seed DC results remain separate from the seven-seed encoder-rate confirmation.

Table 11: Convolutional sequence baseline (CSB), DC and VE-JEPA at 200K updates. Task rows report final-three means and sample standard deviations across ten seeds; the final row reports pooled IQM.

Dataset	CSB	DC	VE-JEPA
hopper-medium	61.9 ± 7.1	65.3 ± 4.8	56.8 ± 6.3
halfcheetah-medium	42.6 ± 0.2	42.5 ± 0.4	46.0 ± 0.2
walker2d-medium	72.0 ± 3.4	73.2 ± 2.0	81.6 ± 2.3
hopper-medium-replay	71.4 ± 18.0	72.2 ± 7.3	90.5 ± 9.0
halfcheetah-medium-replay	35.9 ± 1.1	40.2 ± 0.7	44.4 ± 0.4
walker2d-medium-replay	63.3 ± 7.0	58.9 ± 7.2	74.9 ± 6.1
hopper-medium-expert	110.8 ± 0.5	109.2 ± 2.0	81.5 ± 20.6
halfcheetah-medium-expert	92.7 ± 0.4	91.7 ± 0.7	81.3 ± 4.3
walker2d-medium-expert	108.5 ± 0.2	96.8 ± 13.5	107.6 ± 1.8
Overall IQM	72.7	71.5	74.7

Table 12: Medium-expert comparison at 1M updates, five seeds per task. Entries average evaluations at 900K, 950K and 1M; DC uses fixed target returns.

Medium-expert task	CSB	DC	VE-JEPA
hopper	111.0 ± 0.2	110.7 ± 0.5	110.7 ± 1.8
halfcheetah	93.0 ± 0.3	91.7 ± 0.6	90.9 ± 1.3
walker2d	108.4 ± 0.1	93.4 ± 10.2	109.3 ± 0.2

F Reporting Protocol Comparison

Method	Best-checkpoint IQM	Last-three IQM	Mean of dataset means
DT	80.6	67.6	69.6
DC	83.9	71.5	72.2
IQL	79.9	68.3	68.2
VE-JEPA	88.8	74.7	73.8

All entries use the same ten-seed locomotion runs. Best-checkpoint IQM selects the maximum evaluation score separately for each run before pooling. Last-three IQM averages the final three evaluations per run before pooling. The final column averages the nine dataset means under the last-three rule. These are alternative summaries of the same experiments, not additional trials. The performance margin of VE-JEPA over DC is 3.2 points under last-three IQM, 4.9 under best-checkpoint IQM, and 1.6 under the mean of dataset means. The last comparison has a 95% bootstrap interval of $[-0.4, +3.6]$. Best-checkpoint selection uses the evaluation scores themselves; it is an optimistic summary rather than independent policy selection.

Terminal-transition handling. During development, windowed sampling dropped terminal transitions in earlier AntMaze runs, removing the sparse reward signal. The AntMaze VE-JEPA results reported here use the corrected runs. This failure motivates checking terminal counts and reward mass after sequence construction in addition to monitoring training losses.

G Encoder-Rate Extension and Learning Curves

Appendix Table 13 uses ten seeds per locomotion dataset with $\alpha = 0.1$, retaining all other default VE-JEPA settings. The default-rate runs are the same runs as Table 1. This extension evaluates the chosen multiplier on additional tasks; it is not a new hyperparameter search or a held-out validation of its original selection on AntMaze medium-play. The one-million-step comparison is a separate five-seed experiment on the three medium-expert datasets. Appendix Figure 8 shows the full learning curves for the main ten-seed comparison.

In the AntMaze zero-rate arm, the encoder learning rate is zero from initialization. Scheduled freezing additionally disables encoder gradients after 20,000 or 50,000 steps. This distinction affects which

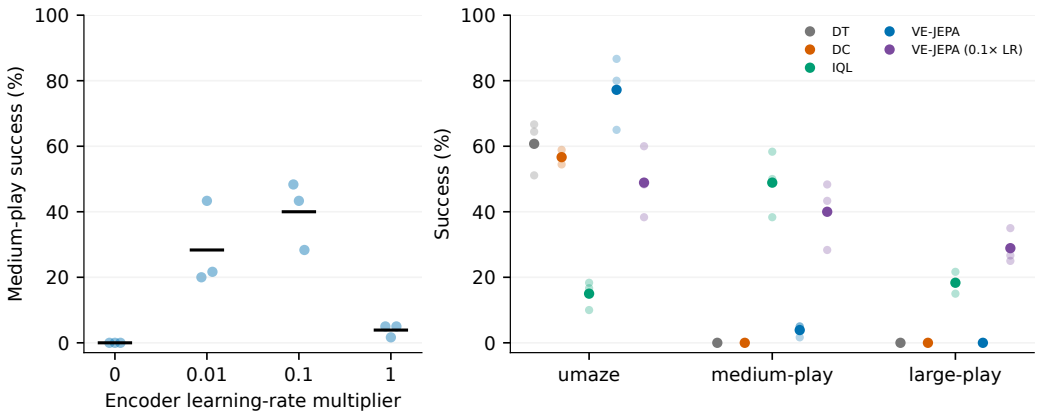


Figure 7: **Encoder learning-rate sensitivity.** Left: medium-play success for four encoder learning-rate multipliers, with other settings fixed. Small points show all three seed scores and black horizontal marks show their mean; the horizontal positions are categorical, including the zero-learning-rate arm. Right: mean success by maze, with individual seed scores shown faintly. The $0.1\times$ setting was selected using the medium-play sweep.

Table 13: Encoder-rate extension on locomotion: ten seeds per dataset at 200,000 steps. Entries use the same last-three IQM protocol as Table 1. Differences are slow minus default, with 95% stratified bootstrap intervals. Bold indicates the higher observed IQM in each row.

Data regime	$\alpha = 1$	$\alpha = 0.1$	Difference (95% CI)
medium	59.0	**60.3**	$+1.3$ $[-2.5, +4.8]$
medium-replay	70.2	**71.4**	$+1.2$ $[-3.9, +6.3]$
medium-expert	**91.7**	85.4	-6.3 $[-13.8, +0.5]$
All nine datasets	**74.7**	71.5	-3.2 $[-6.8, +0.5]$

gradients enter the joint clipping operation; the freezing controls should therefore be interpreted as the implemented update schedules, not as a complete isolation of every consequence of representation stationarity.

H Reward-Irrelevant Observation Coordinates

We append $k \in \{4, 16, 64\}$ coordinates to hopper and walker2d medium-replay observations, with a shared $k = 0$ control. Independent coordinates follow $\mathcal{N}(0, 1)$. Autoregressive coordinates follow $x_{t+1} = 0.95x_t + 0.3122\epsilon_t$, with $x_0 = 0$ and independent standard Gaussian innovations. Training and evaluation use the same process family. The original state is retained, so this construction adds irrelevant information without removing the sufficient state observation. Coordinates are standardized using training-data statistics.

The primary grid compares IQL, VE-JEPA, and VE-JEPA with an additional bisimulation-inspired penalty of coefficient one. The latter penalizes the squared difference between summed latent L1 distance for shuffled transition pairs and a stopped-gradient target combining absolute reward difference with discounted Gaussian Wasserstein distance from the dynamics mean and scale heads. It is an auxiliary configuration of this implementation, not a reproduction of a separate bisimulation algorithm. Each configuration uses three seeds and 200,000 steps, giving 126 runs. Scores in Appendix Table 14 are means and sample standard deviations from 100-episode evaluations of the final checkpoint. This protocol is separate from the main last-three-checkpoint comparison.

The predictive configuration is more robust than IQL in some settings and less robust in others. In particular, the ordering under 64 autoregressive coordinates differs between hopper and walker2d. The added penalty also underperforms without distractors, limiting any attribution of its losses to

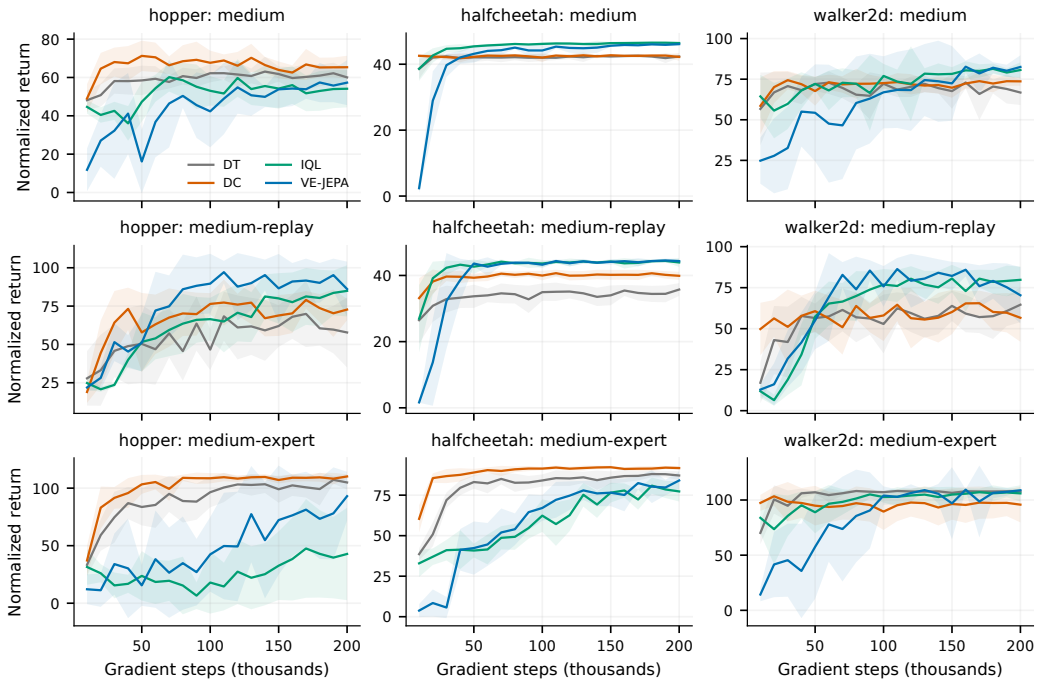


Figure 8: **Learning curves for the main comparison.** Lines and shading show the mean and one sample standard deviation across ten seeds. Evaluation occurs every 10,000 gradient steps. Some sequence-policy curves decline after an early peak, while VE-JEPA continues improving on hopper-medium-expert. The shading describes run variability, not a confidence interval for the mean.

Table 14: Observation-perturbation results on medium-replay. Bold marks the highest observed mean, not statistical significance.

Environment	Distractors	IQL	VE-JEPA	VE-JEPA + penalty
hopper	None	89.6 ± 10.3	**89.7** ± 9.0	59.2 ± 27.9
hopper	IID, $k = 4$	**98.0** ± 1.9	76.6 ± 21.3	28.3 ± 16.9
hopper	IID, $k = 16$	**94.5** ± 3.0	87.4 ± 23.1	43.3 ± 3.7
hopper	IID, $k = 64$	67.1 ± 6.7	**88.7** ± 6.7	32.3 ± 8.3
hopper	AR, $k = 4$	**87.9** ± 6.2	83.8 ± 20.9	72.8 ± 19.0
hopper	AR, $k = 16$	74.3 ± 5.5	**77.2** ± 17.7	43.3 ± 30.3
hopper	AR, $k = 64$	44.9 ± 3.5	**55.3** ± 13.1	18.7 ± 3.5
walker2d	None	**84.5** ± 4.9	67.5 ± 11.7	40.2 ± 15.7
walker2d	IID, $k = 4$	**80.0** ± 4.6	79.6 ± 11.9	28.7 ± 28.3
walker2d	IID, $k = 16$	72.7 ± 5.2	**77.5** ± 7.2	50.0 ± 22.8
walker2d	IID, $k = 64$	64.2 ± 2.3	**74.2** ± 4.0	28.1 ± 3.0
walker2d	AR, $k = 4$	**77.2** ± 1.3	57.2 ± 25.1	40.0 ± 15.8
walker2d	AR, $k = 16$	**65.2** ± 1.6	40.6 ± 15.6	39.3 ± 4.7
walker2d	AR, $k = 64$	**35.2** ± 2.2	24.5 ± 5.5	5.9 ± 1.8

irrelevant predictability. These comparisons do not isolate which encoder loss should suppress distractors; that would require matched gradient-routing interventions.

I Pixel Objective Diagnostics

These diagnostics use a separate Nature-DQN-style CNN and conservative Q-learning pipeline. They do not constitute a visual VE-JEPA benchmark. The initial MSE-to-EMA pilot and the later cosine-objective diagnostic are different configurations: avoiding collapse does not ensure that the

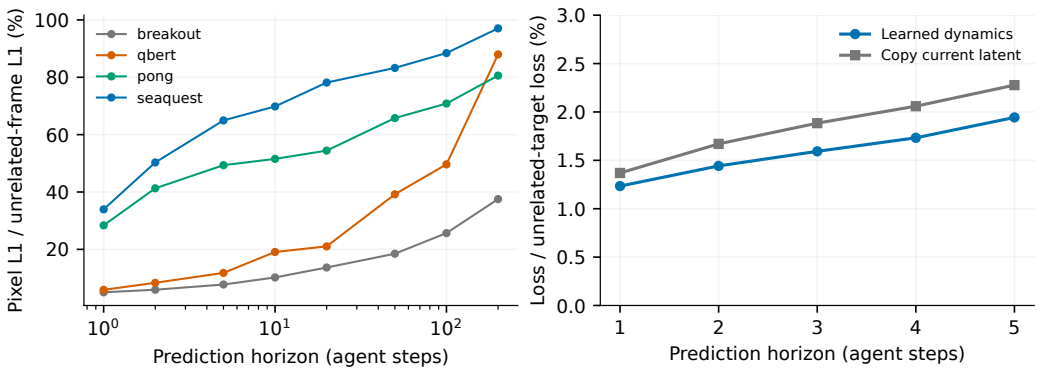


Figure 9: **Temporal redundancy and a copying baseline.** Left: raw pixel L1 change relative to unrelated observations, measured separately in four Atari games. Right: predictive loss relative to unrelated-target loss in Breakout after 20,000 updates, measured in training mode. Copying the current latent has a small residual loss before fitting any dynamics. These are diagnostic measurements, not multi-seed control-performance estimates.

resulting prediction task requires informative dynamics. The following measurements therefore assess the objective, rather than establish a general negative result about predictive representations in visual control. Low predictive loss can arise without learning informative dynamics. We examine two diagnostics in the separate pixel-based experiments.

Representation rank. For covariance eigenvalues λ_i, we track the participation ratio $\left(\sum_i \lambda_i\right)^2 / \sum_i \lambda_i^2$. A pilot pixel experiment showed decreasing rank under an MSE-to-EMA objective. Adding SPR-style projection and asymmetric prediction heads with a cosine objective improved the observed rank trajectory. This is a diagnostic of that pixel implementation; VE-JEPA's state-based loss remains the MSE objective specified in Section 3. Rank alone cannot determine which features are useful for control.

Copy-current-latent baseline. In the Breakout diagnostic, copying the current latent gives losses of 0.02322 at horizon one and 0.03879 at horizon five. Relative to the respective unrelated-target losses, these are approximately 1.4% and 2.3%. The learned predictor gives 0.02092 and 0.03310. The copy baseline therefore accounts for most of the loss reduction relative to unrelated targets, with a smaller additional reduction from learned dynamics.

The raw-observation diagnostic also varies across games. At horizon five, pixel change is approximately 8% of the unrelated-frame distance in Breakout and 12% in Qbert, but 49% in Pong and 65% in Seaquest. Frame stacking and visually persistent scenes can make short-horizon prediction easy, but the degree of redundancy is empirical rather than a universal property of four-frame stacks. These measurements motivate stronger temporal baselines before interpreting pixel auxiliary-loss results as evidence for or against predictive representation learning.

J Additional Training Diagnostics

Lines below show condition means and faint individual seeds. Diagnostics are averaged in nonoverlapping 10K windows, excluding initialization. Gradient norms are pre-clipping derivatives with respect to the encoder. The different loss and rank orderings across conditions qualify simple explanations of control performance based on these quantities alone.

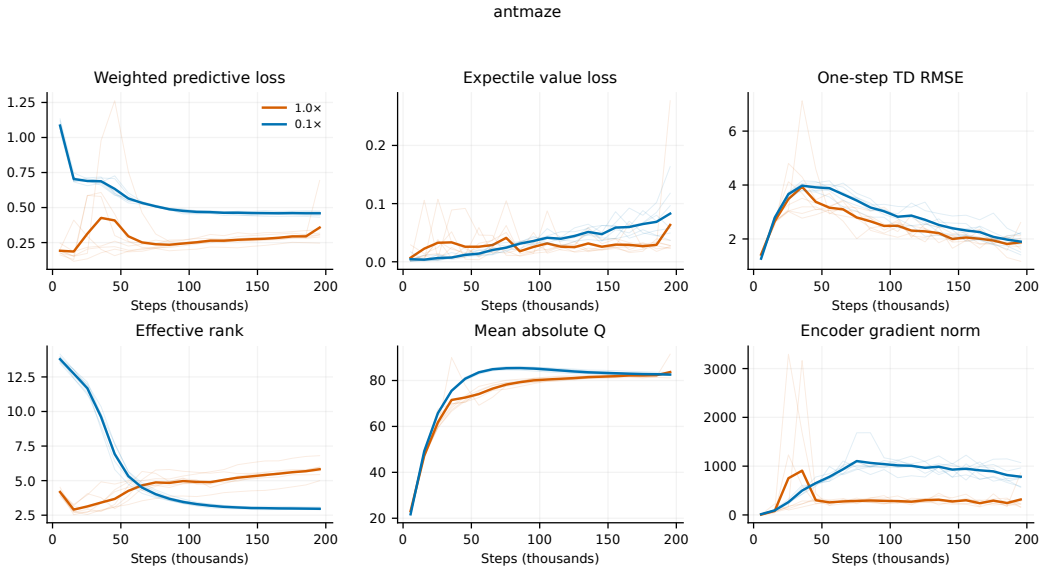


Figure 10: AntMaze medium-play diagnostics, seven additional seeds per encoder rate.

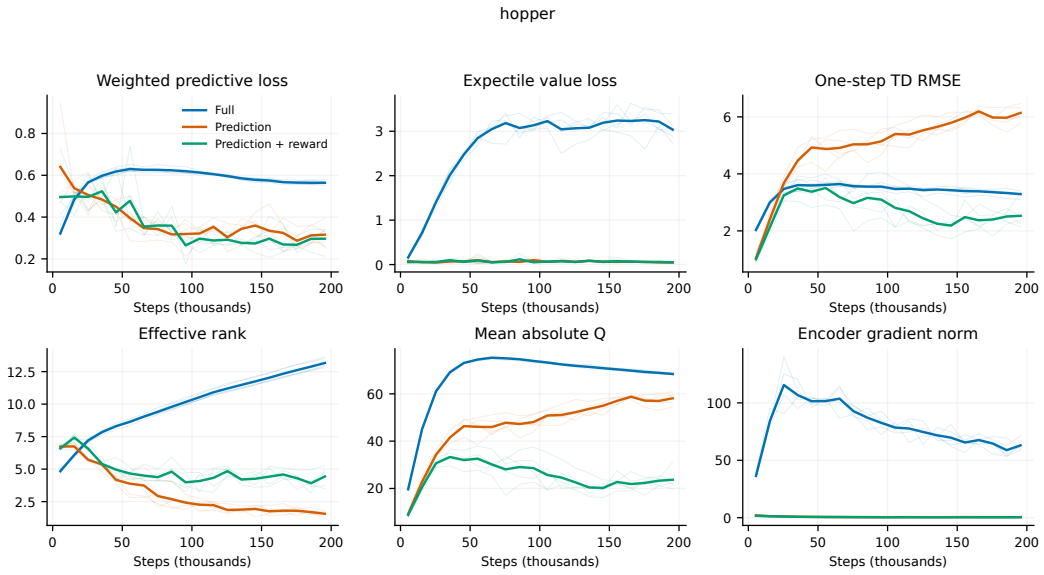


Figure 11: Hopper medium-replay diagnostics, five seeds per encoder objective.

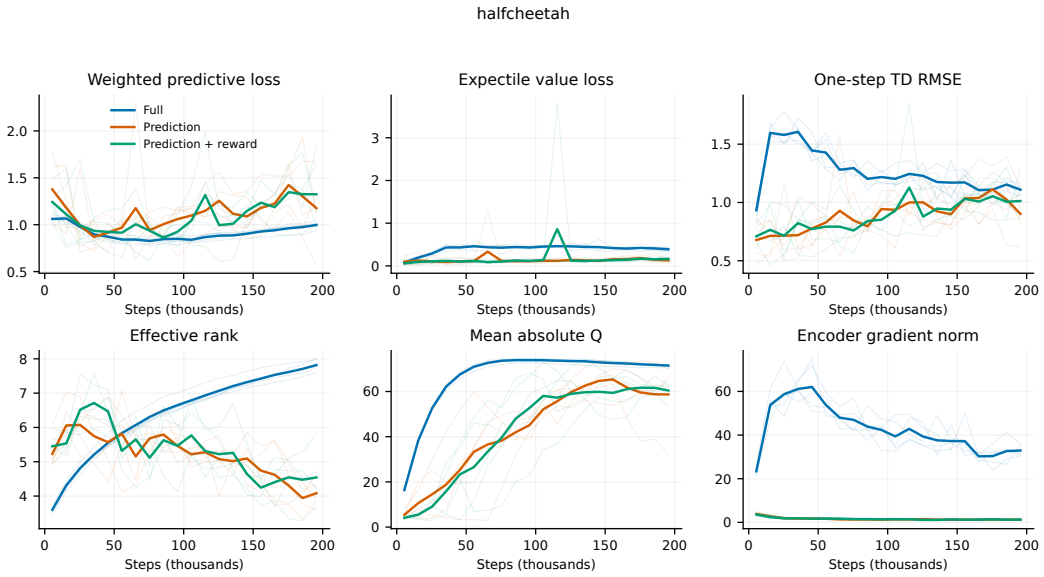


Figure 12: HalfCheetah medium-replay diagnostics, five seeds per encoder objective.

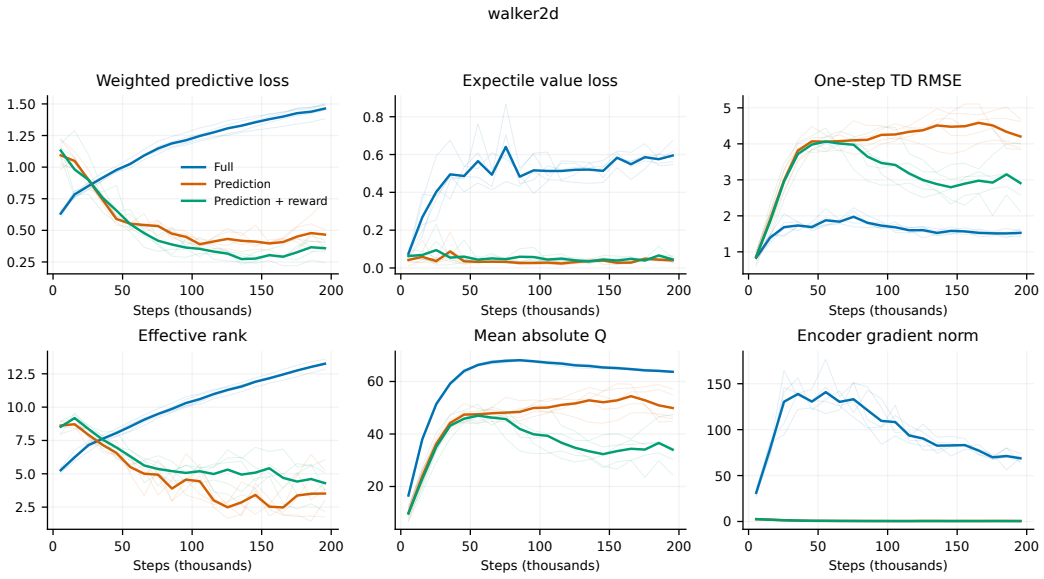


Figure 13: Walker2d medium-replay diagnostics, five seeds per encoder objective.

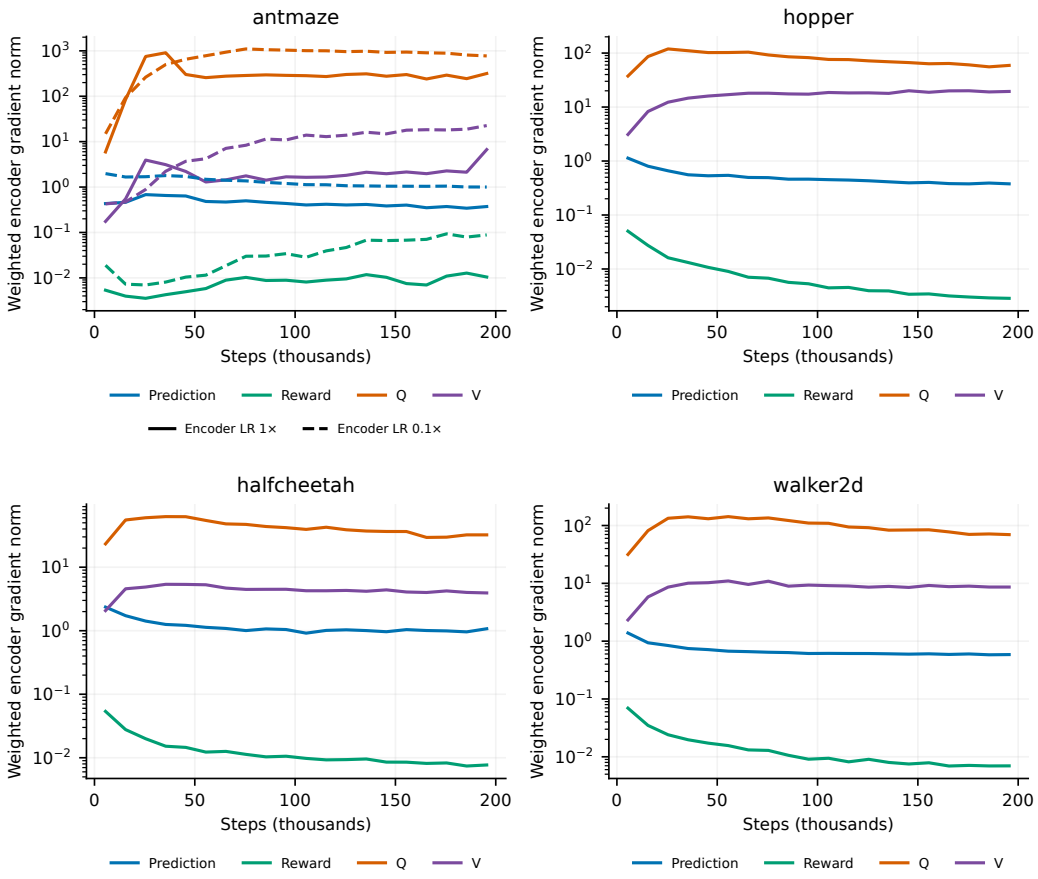


Figure 14: Weighted loss-specific encoder-gradient norms before clipping. Curves show seed means of nonoverlapping 10K-window averages. Locomotion panels use full encoder supervision. In AntMaze, solid lines use the default encoder rate and dashed lines use $0.1\times$. Q-loss gradients have the largest late-window means, but norms do not measure optimizer update sizes or isolate the causal contribution of an individual objective.