CAN IN-SAMPLE OFFLINE RL LEARN CRITICS FIRST AND EXTRACT POLICIES LATER?

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent progress in robotics and foundation models is driving offline reinforcement learning toward increasingly expressive but computationally expensive policy classes, including diffusion-based, flow-based, and large VLA policies. In-sample value-learning methods such as IQL and XQL are appealing in this setting because they estimate value functions without sampling actions from the current policy. This suggests a simple two-stage strategy: first perform value learning, then freeze the learned critic and extract a policy afterward, which we refer to as the *Freeze-then-Extract* protocol. However, existing in-sample value-learning methods typically still rely on alternating policy and critic updates in practice. Motivated by this gap, we systematically evaluate Freeze-then-Extract across offline RL methods and benchmarks and find that, despite its appeal, it often underperforms standard alternating updates. To understand this degradation, we investigate how the learned critic evolves throughout training. We find that advantage estimates vary substantially across training steps, even toward the end of critic training. When policy extraction relies on a single frozen critic, these fluctuations can lead to transient high-advantage peaks and a reduction in the effective number of training samples. We therefore investigate whether aggregating signals from multiple frozen critics, obtained from different checkpoints or independent initializations, can stabilize policy extraction while preserving policy-sampling-free value estimation. We find that such ensembles improve average extraction performance. In VLA experiments, critic-first training reaches strong performance with fewer expensive policy updates. Our results identify temporal variability in advantage estimates as a key obstacle to reusable frozen critics in scalable offline RL.

1 INTRODUCTION

Recent progress in robotic foundation models has made increasingly expressive policy classes, such as diffusion-based (Chi et al., 2025; Liu et al., 2024; Bjorck et al., 2025), flow-based (Black et al., 2024; Physical Intelligence et al., 2025), and Vision-Language-Action (VLA) policies (Open X-Embodiment Collaboration, 2024; Kim et al., 2024; Black et al., 2024; Bjorck et al., 2025), an attractive direction for offline reinforcement learning (Wang et al., 2023b; Hansen-Estruch et al., 2023; Park et al., 2025c; Physical Intelligence et al., 2025). These policies can represent complex and multi-modal action distributions, but they often require multiple network evaluations per action or expensive large-model inference. Consequently, methods that repeatedly query the current policy, whether for actor updates or for critic learning, can become increasingly costly as policy classes scale. This makes it important to minimize unnecessary policy queries throughout offline RL training.

From this perspective, in-sample offline RL methods, including IQL (Kostrikov et al., 2022) and XQL (Garg et al., 2023), are especially appealing for expensive policy classes. Their critic updates are in-sample: they use only actions observed in the offline dataset, rather than actions sampled from the current policy. Consequently, the critic can be trained without repeatedly querying the policy, so value learning is effectively decoupled from policy learning. Building on the terminology of decoupled value learning and policy extraction in prior work (Park et al., 2024), we refer to offline RL methods with this property as *decoupled offline RL* methods.

However, there is an important gap between this conceptual decoupling and the actual implementation of these methods. Although the value function can in principle be learned without policy sampling,

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.

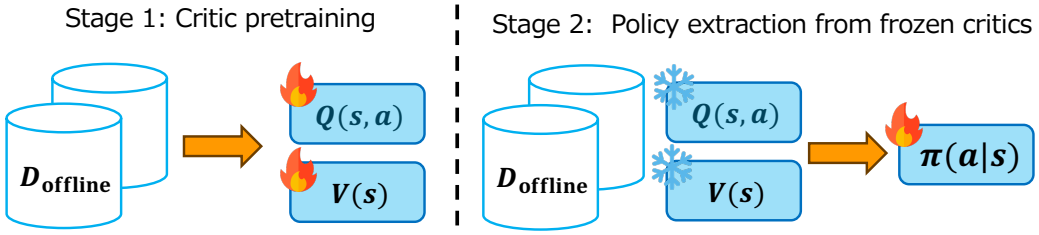

Figure 1: Freeze-then-Extract first trains value functions without policy updates, then freezes them for policy extraction.

existing methods still typically update the value function and the policy alternately, much like standard actor-critic reinforcement learning (Sutton & Barto, 2018; Haarnoja et al., 2018; Fujimoto et al., 2018). This convention is acceptable when policy updates are cheap, but can become wasteful when early policy updates are performed using immature value estimates and the policy itself is expensive to query. This raises the central question of this work: *Can decoupled offline RL learn values first, freeze them, and extract policies later?*

The most direct way to operationalize this idea is the *Freeze-then-Extract* protocol, shown in Fig. 1. In the first stage, we train the value functions using the original value-learning objective of each base method, without policy updates or policy sampling. In the second stage, we freeze the learned value functions and train only the policy using the corresponding extraction objective. This strategy is attractive because it can reduce expensive policy updates and, in goal-conditioned or multi-task settings, allows a general value function to be pretrained from broad offline data and reused to extract task-specific policies for downstream tasks. Yet, it remains unclear when a single frozen value function provides a reliable signal for policy extraction.

In this work, we evaluate Freeze-then-Extract across representative in-sample offline RL methods, including IQL (Kostrikov et al., 2022), XQL (Garg et al., 2023), and SQL (Xu et al., 2023), on D4RL (Fu et al., 2020) and OGBench (Park et al., 2025a), and compare it against standard alternating updates. We find that, despite its appeal, Freeze-then-Extract shows substantial performance degradation relative to standard alternating updates, underperforming on average across all three algorithms and both benchmark suites (Fig. 2). This counterintuitive performance drop suggests that the two-stage protocol introduces a difficulty that is not apparent from the value-learning objective alone. Detailed experimental settings and per-task results are provided in Appendices G and B.

To understand this degradation, we investigate how the learned critic evolves throughout training. We find that advantage estimates vary substantially across training steps, even toward the end of critic training. When a single checkpoint is frozen, policy learning can become locked onto transient, potentially spurious high-advantage peaks. In addition, advantage weighting can concentrate policy learning on a small subset of samples. We therefore investigate whether aggregating signals from multiple frozen critics, obtained from different checkpoints or independent initializations, can stabilize policy extraction while preserving policy-sampling-free value learning. We find that such ensembles improve average extraction performance. In VLA experiments, critic-first training reaches strong performance with fewer expensive policy updates. We also examine the reuse of frozen critics for downstream goals without additional value learning.

This paper makes the following contributions:

1. **Systematic evaluation.** We systematically evaluate *Freeze-then-Extract* with IQL, XQL, and SQL on D4RL and OGBench, and show that single-critic extraction underperforms alternating updates on average.
2. **Analysis of frozen-critic degradation.** We analyze late-training variability in advantage estimates and its interaction with potentially spurious peaks and concentrated extraction weights, motivating the hypothesis that alternating updates provide implicit temporal ensembling.
3. **Multiple frozen critics and practical implications.** We investigate policy extraction from multiple frozen critics and demonstrate its usefulness for mitigating single-critic degradation, while examining the practical benefits of critic-first training with expensive VLA policies and reusable value functions.

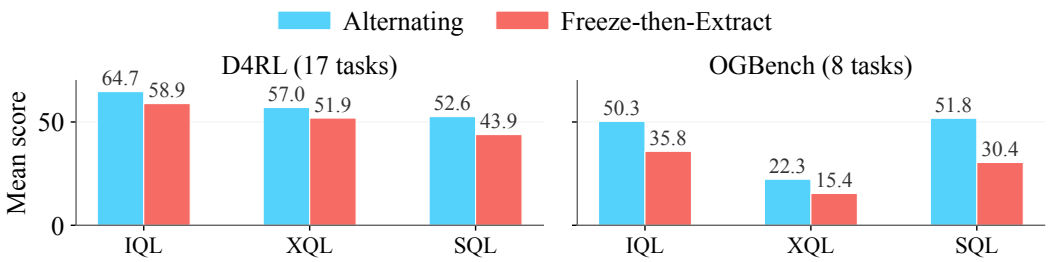

Figure 2: Average normalized performance of standard alternating updates and single-critic Freeze-then-Extract on D4RL and OGBench.

2 Related Work

Offline reinforcement learning. Offline reinforcement learning learns a policy from a fixed dataset without additional environment interaction (Levine et al., 2020). A central difficulty in this setting is distributional shift: when the learned policy selects actions that are not well covered by the dataset, value estimates can become unreliable and policy improvement can be unstable. To address this issue, prior work has proposed behavior-constrained methods (Fujimoto & Gu, 2021; Tarasov et al., 2023; Wu et al., 2019), conservative value-learning objectives (Kumar et al., 2020), and in-sample value-learning methods, which we discuss in more detail below (Kostrikov et al., 2022; Garg et al., 2023; Xu et al., 2023; Sikchi et al., 2024; Kim et al., 2026; Li et al., 2026). More recently, offline RL has moved toward expressive policy classes, including diffusion policies, flow-based policies, and large vision-language-action policies (Wang et al., 2023b; Kang et al., 2023; Park et al., 2025c; Physical Intelligence et al., 2025; GigaBrain Team et al., 2026). While these policies can model complex and multi-modal behaviors, they also make policy optimization expensive due to iterative sampling, multiple network evaluations, or large-model inference. This motivates us to investigate whether value learning and policy extraction can be separated in training, reducing expensive policy updates and queries while still yielding strong policies.

Decoupled value learning and policy extraction. Decoupled offline RL methods learn value functions using only actions from the offline dataset and then extract a policy from the learned values. IQL learns a state value function with expectile regression and extracts a policy through advantage-weighted regression without sampling actions from the current policy during critic learning (Kostrikov et al., 2022). XQL extends this idea with an extreme-value formulation (Garg et al., 2023), and related methods such as SQL, IVR, and dual RL also study in-sample or policy-sampling-free value learning objectives (Kostrikov et al., 2022; Xu et al., 2023; Garg et al., 2023; Sikchi et al., 2024; Kim et al., 2026; Li et al., 2026). These methods are attractive for expensive policy classes because critic learning does not require repeated policy queries. However, despite this conceptual separation, most implementations still alternate critic and actor updates as in standard actor-critic training (Kostrikov et al., 2022; Xu et al., 2023; Garg et al., 2023; Sikchi et al., 2024). Our work examines whether this convention is necessary. While Park et al. (2024) focus on policy extraction objectives for decoupled offline RL, we focus on the training protocol itself, especially the difference between alternating updates and Freeze-then-Extract.

Goal-conditioned and multi-task offline RL. Goal-conditioned and multi-task RL aim to learn policies and value functions that solve multiple tasks or reach different goals using shared data and representations. In goal-conditioned RL, policies and values are conditioned on a goal variable so that one model can generalize across desired outcomes (Andrychowicz et al., 2017; Schaul et al., 2015; Eysenbach et al., 2022; Wang et al., 2023a; Park et al., 2023; 2025a; Myers et al., 2025). In multi-task offline RL, a dataset may contain experience from many tasks, and the goal is to learn reusable representations, policies, or value functions that transfer across tasks (Lee et al., 2022; Kumar et al., 2023; He et al., 2023; Hu et al., 2024). As offline RL moves from single-task control toward large-scale datasets with many tasks, goals, and embodiments, value-function reuse becomes increasingly appealing (Chebotar et al., 2023; Springenberg et al., 2024; Park et al., 2025b; Abe et al., 2026). A critic trained on goal-conditioned or multi-task data could serve as a reusable evaluation signal for many downstream policies or architectures. This possibility is especially natural for decoupled offline RL methods, but remains largely unexplored. We therefore investigate critic reuse under Freeze-then-Extract in goal-conditioned offline RL.

3 PRELIMINARIES

We consider a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, \mu, p, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\mu \in \Delta(\mathcal{S})$ is the initial state distribution, $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics, and $\gamma \in [0, 1)$ is the discount factor (Sutton & Barto, 2018). A policy $\pi(a \mid s)$ induces a trajectory distribution $p_\pi(\tau)$ on $\tau = (s_0, a_0, \ldots, s_T, a_T)$, where T is the horizon. Its discounted return is $J(\pi) = \mathbb{E}_{\tau \sim p_\pi(\tau)}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right]$. In offline RL, the goal is to learn a policy that maximizes $J(\pi)$ using only a static dataset $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$, without additional interaction with the environment (Levine et al., 2020). We focus on decoupled offline RL methods, in which critic learning can be carried out using only actions contained in the offline dataset, without sampling actions from the current policy during training. Such methods typically learn a state-action value function $Q(s, a)$ and, when needed, a state value function $V(s)$, and perform policy extraction based on the resulting advantage $A(s, a) = Q(s, a) - V(s)$. In some experiments, we also consider multi-task or goal-conditioned offline RL, where the policy and critic are additionally conditioned on a task or goal variable c, yielding $\pi(a \mid s, c)$, $Q(s, a, c)$, and $V(s, c)$; in the goal-conditioned case, $c = g$ corresponds to a goal state and rewards are sparse success indicators based on goal achievement.

4 WHY DOES FREEZE-THEN-EXTRACT FAIL WITH A SINGLE FROZEN CRITIC?

As shown in Fig. 2, single-critic Freeze-then-Extract often underperforms alternating updates. We next investigate what makes policy extraction sensitive to freezing the critic.

4.1 TEMPORAL VARIABILITY OF ADVANTAGE ESTIMATES

For Freeze-then-Extract to work well, the learned value function should provide a sufficiently stable signal for policy extraction. However, our analysis reveals that this intuitive expectation does not necessarily hold in practice. Perhaps surprisingly, even in the later stages of training, these quantities can still vary non-negligibly across nearby critic checkpoints.

To examine this phenomenon, we analyze how the values assigned to the same state-action samples change across multiple checkpoints saved during critic training. In this experiment, we train the critic with IQL for 1M steps, which is commonly regarded as sufficient in standard offline RL settings (Kostrikov et al., 2022; Kumar et al., 2020). In Fig. 3, we visualize the advantage heatmap for a fixed state in OGBench's PointMaze using checkpoints saved every 100 update steps near the end of training.

We find that even across a relatively small number of update steps in the late stage of training, the absolute values and overall structure of the advantage landscape can still vary across checkpoints. Furthermore, in Fig. 4, we examine the relative ranking of multiple candidate actions for a given state and observe that the ordering of actions can also change across nearby checkpoints. Taken together, these results highlight a perhaps underappreciated fact: even near the end of critic training, both the advantage landscape and the induced action ranking can remain noticeably unstable. Additional state-level visualizations in Appendix A.1 show that these checkpoint-level changes in advantage landscapes and action rankings are not specific to the examples shown here. We further examine whether temporal critic variability is associated with frozen-extraction degradation. Across three tasks, larger checkpoint-to-checkpoint changes in action rankings and relative advantages tend to coincide with larger performance drops, as shown in Appendix A.2. Appendix A.3 further shows that changing only the critic checkpoint can substantially affect performance even with flow-based sampling-based extraction.

Importantly, this does not imply that the entire value function is completely unstable. In many states, the overall trends and coarse structure remain broadly consistent, and the broad pattern of which actions are better is often preserved. However, when such fluctuations are frozen into a pretrained critic, they can become non-negligible for the small number of high-advantage or borderline samples that matter most during policy extraction. As we discuss next, this becomes particularly problematic for weighted-regression-based policy extraction.

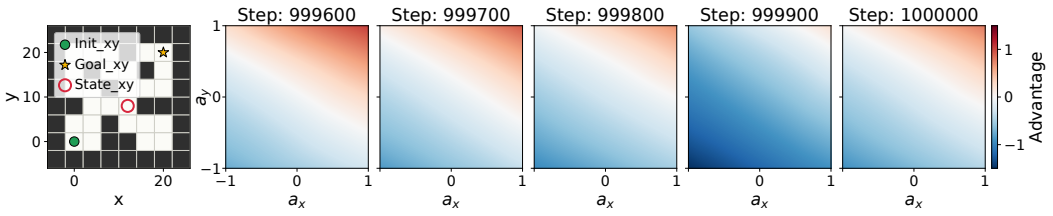

Figure 3: Advantage heatmaps for a fixed state in PointMaze, visualized using critic checkpoints saved every 100 update steps near the end of training.

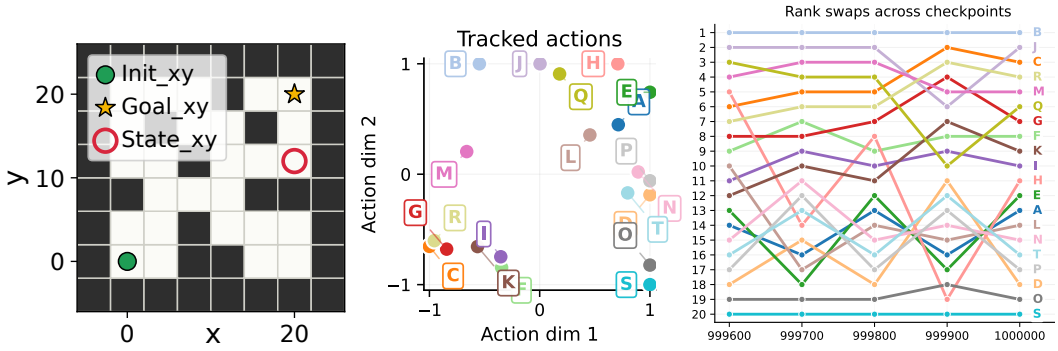

Figure 4: Relative ranking of candidate actions for a fixed state across critic checkpoints saved every 100 update steps near the end of training.

4.2 Spurious Peaks and Sample Over-Concentration

When policy extraction is performed from a single frozen checkpoint, incidental biases contained in the value function at that point are also frozen. We highlight two related effects that can make policy extraction from a single frozen critic particularly fragile.

First, if a particular state-action pair is assigned a higher advantage than it should have, policy learning may incorrectly place too much emphasis on that sample. We refer to this phenomenon as **lock-in to spurious peaks**. The action-ranking analysis in Fig. 4 provides a concrete example of this instability: even for the same state and candidate actions, the relative ordering induced by the critic can change substantially across nearby checkpoints. Once one such checkpoint is frozen, a transiently high-ranked action can become a persistent extraction target. Appendices E.1 and E.2 show that frozen-critic degradation also persists with DDPG+BC extraction and a flow-based FAWAC actor, indicating that it is not specific to AWR-style extraction or the Gaussian policy class.

Second, **AWR-style weighting can amplify the sensitivity to a single frozen critic by concentrating policy learning on a small subset of samples**. This effect is visualized in Fig. 5, where dataset samples are ranked by AWR weight and their cumulative weight mass is plotted. Although the dataset contains one million samples, the top-ranked samples account for a disproportionately large fraction of the total extraction weight: the top 5% already explain 70.7% of the cumulative AWR weight mass, and the top 10% explain 87.0%. Thus, policy learning can be dominated by a much smaller subset of high-weight samples. Prior work notes that such concentration can cause AWR overfitting (Park et al., 2024). Under Freeze-then-Extract, this concern is heightened because the same high-weight samples are repeatedly emphasized. If some of these large weights reflect transient estimation errors, this persistent emphasis can reinforce their influence on the learned policy. Consistent with this interpretation, Appendix F.2 shows that reducing the AWR inverse temperature partially alleviates frozen-extraction degradation.

4.3 Alternating Updates as Implicit Temporal Ensembling

These observations naturally raise the following question: why does standard alternating training not exhibit such severe degradation, even though it relies on the same underlying value-learning process? Our hypothesis is that **alternating updates function as a form of implicit temporal ensembling**. Rather than relying on a single fixed critic throughout policy learning, alternating updates expose the policy to a sequence of evolving critic signals.

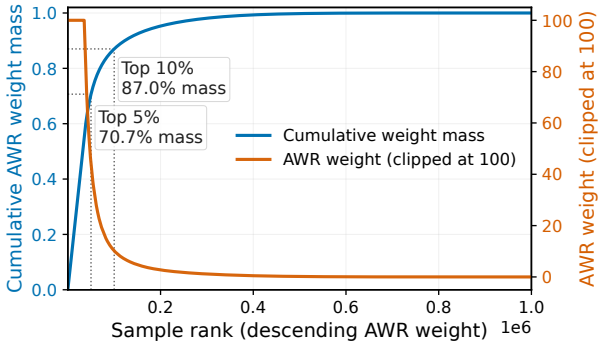

Figure 5: Dataset samples ranked by AWR weight for a single critic checkpoint, with clipped weights overlaid with cumulative weight mass to show concentration among top-ranked samples.

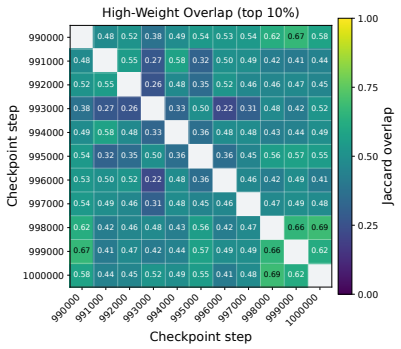

Figure 6: Pairwise Jaccard overlap $|A \cap B|/|A \cup B|$ of top-weighted sample sets across critic checkpoints.

First, **the influence of transient spurious peaks can be mitigated over time**. Under alternating updates, the policy is not trained all at once from a single fixed critic. Instead, it is updated gradually against a sequence of critics that evolve throughout training. Therefore, even if the critic at a particular moment produces a spurious peak, subsequent updates introduce learning signals from later checkpoints, reducing the influence of that particular bias.

Second, **persistent concentration on a fixed subset can be reduced because the high-weight sample set changes over time**. AWR weights can still be highly concentrated at each individual training step under alternating updates. However, as the critic evolves, the samples receiving the largest weights can also change. To examine this directly, we measure the Jaccard overlap $|A \cap B|/|A \cup B|$ between top-weighted sample sets across critic checkpoints. Fig. 6 shows that this overlap is often far from complete, even between nearby checkpoints, indicating that different samples receive high weight at different stages of training. This changing emphasis may prevent the policy from repeatedly fitting to a single critic-specific high-weight subset.

Overall, the key distinction is that alternating updates expose the policy to changing critic-induced weights, whereas Freeze-then-Extract repeatedly uses a fixed weighting signal from a single critic. Appendix F.5 further shows that simply smoothing a single critic is insufficient: extraction from the slowly updated target critic remains close to single-critic extraction.

5 Freeze-then-Extract with Multiple Critics

The analysis in Section 4 suggests that the degradation of Freeze-then-Extract may be driven in part by persistent dependence on a single critic-specific signal. We therefore consider two complementary multi-critic variants that expose policy extraction to multiple critic signals while preserving policy-sampling-free value learning: one uses critics from different checkpoints along a single value-learning trajectory, while the other uses critics from separately initialized value-learning runs.

Ensembles of value functions have been widely studied in reinforcement learning for purposes such as exploration, variance reduction, and uncertainty estimation (Osband et al., 2016; Anschel et al., 2017; An et al., 2021). Our focus here is specifically on the Freeze-then-Extract setting: motivated by the checkpoint sensitivity identified in our analysis, we examine whether diversity across frozen critics can mitigate the fragility of downstream policy extraction.

Checkpoint-based variant. This checkpoint-based variant directly operationalizes our interpretation of alternating updates as implicit temporal ensembling. We refer to it as *Checkpoint Ensemble*. We first train a single critic and save its parameters at multiple training steps, $\mathcal{T} = \{t_1, \ldots, t_M\}$. After value learning, we freeze the resulting collection

$$\mathcal{E}_{\text{ckpt}} = \{(Q^{(t_m)}, V^{(t_m)})\}_{m=1}^{M}, \tag{1}$$

where each Q-function is paired with the V-function from the same training step. This construction exposes policy extraction to temporal variation along a single critic-training trajectory. The

choice of checkpoints determines how much temporal variation is represented in the ensemble: late, closely spaced checkpoints capture local fluctuations near convergence, whereas more widely spaced checkpoints provide broader temporal diversity but may include less mature critics. The checkpoint schedules used in our experiments are described in Appendix C.

Independent-run variant. This independent-run variant provides a complementary source of critic diversity through random initialization. We refer to it as *Independent Ensemble*. We train M critics from different random initializations under otherwise identical training conditions. After training, we retain the final pair from each run:

$$\mathcal{E}_{\text{ind}} = \{(Q_m, V_m)\}_{m=1}^{M}. \tag{2}$$

Random initialization induces variation in sample-wise advantage estimates, yielding diverse signals for policy extraction. Compared with Checkpoint Ensemble, Independent Ensemble obtains diversity from different random initializations rather than from different stages of a single training trajectory. We therefore simply retain the final checkpoint from each run, avoiding an additional choice over training stage. Appendix A.4 compares diversity induced by checkpoints and random initializations.

Policy extraction from frozen ensembles. Both constructions retain the policy-extraction objective of the base algorithm. At each actor update, we sample $m \sim \text{Unif}(\{1, \dots, M\})$ and use the corresponding advantage $A_m(s,a) = Q_m(s,a) - V_m(s)$ for the entire minibatch. We optimize

$$\mathcal{L}_\pi(\phi) = -\mathbb{E}_{\substack{m\sim\text{Unif}(\{1,\dots,M\}) \\ (s,a)\sim\mathcal{D}}} \left[w_m(s,a) \log \pi_\phi(a \mid s) \right], \tag{3}$$

where the member-specific weights follow the base algorithm:

$$w_m(s,a) = \begin{cases} \min\{\exp(\beta A_m(s,a)), w_{\max}\}, & \text{IQL and XQL,} \\ \min\{\max(A_m(s,a), 0), w_{\max}\}, & \text{SQL.} \end{cases}$$

Thus, the method aggregates member-specific extraction weights rather than averaging advantages before applying the weighting. When different members assign high weight to different samples, their mixture reduces repeated dependence on a single critic-specific high-weight subset. This directly targets the spurious-peak and sample-concentration failure modes identified in Section 4.

6 Experiments

We evaluate frozen-critic ensembling from three complementary perspectives: its ability to stabilize policy extraction on standard offline RL benchmarks, its practical value when policy updates are expensive, and its usefulness for reusing pretrained value functions. We first compare Checkpoint Ensemble and Independent Ensemble on D4RL and OGBench. We then study VLA policies, where reducing actor updates and reusing critic computation can have substantial practical value, and finally evaluate goal-conditioned critic reuse. Detailed settings are provided in Appendix G.

6.1 Offline RL Benchmarks

We compare alternating updates (Alt.), single-critic Freeze-then-Extract (Frozen), Checkpoint Ensemble, and Independent Ensemble using IQL (Kostrikov et al., 2022), XQL (Garg et al., 2023), and SQL (Xu et al., 2023) on D4RL (Fu et al., 2020) and OGBench (Park et al., 2025a). Both ensembles use $M = 10$ members. Checkpoint Ensemble uses the Tail-10k schedule across tasks, while Independent Ensemble uses final critics from separately initialized value-learning runs. We additionally report *Checkpoint (Tuned)*, which selects the best-performing schedule for each algorithm–task pair among Tail-1k, Tail-10k, Tail-50k, and Full-100k based on mean score across seeds.

Both Checkpoint Ensemble and Independent Ensemble recover much of the degradation caused by single-critic Freeze-then-Extract and narrow the gap to standard alternating updates. Independent Ensemble tends to perform strongly without requiring temporal checkpoint-schedule tuning, which is particularly attractive when each policy-training run is expensive, as in VLA settings. Moreover, the ensemble-size ablation in Appendix F.1 shows that performance improves as the number of critics increases in the tested setting, suggesting that Independent Ensemble provides a relatively simple

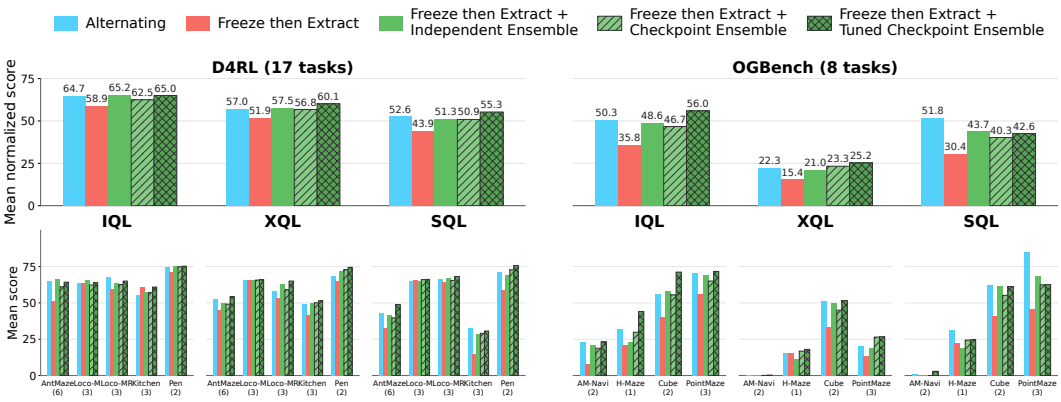

Figure 7: Alternating training vs. Freeze-then-Extract across IQL, XQL, and SQL, including multi-critic variants. Top: D4RL and OGBench averages. Bottom: task-group breakdown.

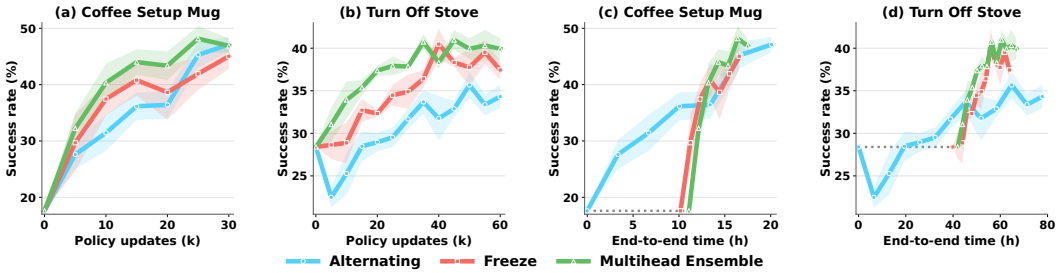

Figure 8: Practical efficiency with VLA policies on `CoffeeSetupMug` and `TurnOffStove`. (a,b) Success rate versus actor updates. (c,d) Success rate versus projected end-to-end training time.

way to improve extraction robustness. As an additional control, Appendix F.4 shows that ensembling also improves alternating training, although the average gain is larger under frozen extraction and frozen ensembling remains below alternating ensembling on average.

Checkpoint (Tuned) further exceeds alternating updates in five of the six algorithm–benchmark averages, showing that an appropriate set of temporal checkpoints can sometimes outperform standard alternating training. One possible explanation is that critic-first training allows a larger fraction of the actor-update budget to use already informative value estimates, instead of spending early actor updates while the critic is still immature. Detailed per-task results are provided in Appendix B, and checkpoint-schedule sensitivity is analyzed in Appendix C.

6.2 Practical Efficiency with VLA Policies

We next evaluate critic-first training with computationally expensive VLA policies. Motivated by the strong performance of Independent Ensemble without checkpoint-schedule tuning, we use a lightweight Multi-head Ensemble in the VLA experiments. Instead of training fully independent critics, this variant shares a common backbone across five critic heads, providing multiple extraction signals without replicating the full critic network. We compare alternating updates (Alt.), single-critic Freeze-then-Extract (Frozen), and Freeze-then-Extract with this multi-head ensemble. We collect our own offline datasets in RoboCasa-based simulation environments (Nasiriany et al., 2024) and use DEAS (Kim et al., 2026), which extends IQL to action sequences with distributional critics. We consider two settings: single-task `CoffeeSetupMug` and language-conditioned multitask `TurnOffStove`, where instructions specify the target burner. Both datasets contain successful and failed trajectories. Additional dataset, model, and training details are provided in Appendix G.4.

Fig. 8(a,b) shows clear gains in actor-update efficiency. Using the final mean success rate of Alt. as the reference, Multi-head reaches this level with 25k versus 30k updates on Coffee and 15k versus 50k on Stove, corresponding to 16.7% and 70% fewer actor updates, respectively. The benefit of critic-first training is particularly clear on the language-conditioned multitask Stove setting, where critic learning is more computationally demanding: Multi-head achieves nearly twice the improvement over Base

as Alt. Moreover, Multi-head improves early learning and outperforms single-critic extraction at nearly all evaluated checkpoints in both settings, supporting the benefit of multiple critic signals. Fig. 8(c,d) further shows that these gains remain meaningful in projected end-to-end training time after accounting for critic pretraining and advantage-cache construction: in our implementation, Multi-head reaches the final mean performance of Alt. in less projected training time on both tasks. Together, these results show practical gains beyond actor-update efficiency.

Freezing the critics also enables reuse across policy-side hyperparameter settings: cached raw advantages allow changing the extraction temperature β without retraining the critic or recomputing advantages. For a sweep over five values of β, our runtime estimates indicate reductions in total training time of 57.1% on Coffee and 57.3% on Stove compared with independently training five alternating runs. This makes critic and advantage reuse particularly useful for policy-side hyperparameter tuning with large VLA policies. Detailed learning curves, implementation-specific runtime analysis, timing protocols, and Multi-head overhead are provided in Appendix D.

6.3 Multi-task Critic Reuse

An important benefit of freezing critics is their potential reusability. A critic pretrained across diverse tasks could serve as a generic value function from which multiple downstream policies are extracted, including tasks not explicitly used during critic pretraining. As a controlled setting for studying this possibility, we consider goal-conditioned critic reuse. We pretrain goal-conditioned GCIQL (Park et al., 2023; 2025a) critics on multi-goal offline data and then freeze them. For each downstream goal, we train a separately initialized actor without further critic updates. We compare single-critic Freeze with Checkpoint Ensemble and Independent Ensemble. Full details of the experimental setup are provided in Appendix G.5.

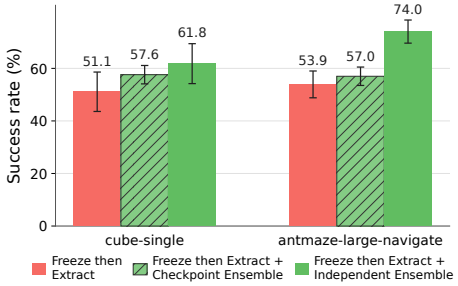

Figure 9: Goal-specific policy extraction from pretrained GCIQL critics.

Fig. 9 shows that ensembling remains effective in this critic-reuse setting. On `cube-single`, the success rate improves from $51.1 \pm 7.5\%$ with Freeze to $57.6 \pm 3.5\%$ with Checkpoint Ensemble and $61.8 \pm 7.6\%$ with Independent Ensemble. On `antmaze-large-navigate`, the success rate improves from $53.9 \pm 5.1\%$ to $57.0 \pm 3.5\%$ and $74.0 \pm 4.4\%$, respectively. These results suggest that pretrained critics can be reused for downstream policy extraction without retraining the value function, and that ensembling can improve extraction in this setting.

7 Conclusion and Future Directions

We investigated whether decoupled offline RL can learn value functions first and extract policies from frozen critics afterward. Our benchmark results show that single-critic Freeze-then-Extract often underperforms standard alternating updates. Our analysis points to temporal variability in advantage estimates as a possible explanation for this degradation: checkpoint-specific fluctuations can create spurious high-advantage peaks and over-concentrate policy learning on a small subset of samples. We hypothesize that alternating updates mitigate this dependence through implicit temporal ensembling. Motivated by this interpretation, we examined Checkpoint Ensemble and Independent Ensemble and found that both improve frozen-critic extraction on average while preserving policy-sampling-free value learning. These findings also extend to settings where policy learning is expensive or critics are reused. In VLA experiments, critic-first training reaches strong performance with fewer expensive actor updates, while a shared-backbone Multi-head Ensemble improves extraction over a single frozen critic. In goal-conditioned experiments, pretrained critics can be reused to extract policies for held-out downstream goals without additional value learning, with ensembling improving extraction.

At the same time, ensembling critic signals is not a complete solution to reliable and efficient critic-first offline RL. A more fundamental direction is to incorporate stability directly into offline RL critic learning. Another important direction is to test critic-first training at larger scale, where the ability to reuse pretrained critics could become increasingly valuable. Overall, we view this work as an initial step toward reusable and efficient value learning for offline RL.

AI USE STATEMENT

We used generative AI tools to assist with experimental design, code generation and modification, literature search, and manuscript preparation. All AI-assisted code was reviewed and tested by the authors, and literature references suggested by AI tools were checked against the original sources. All final decisions regarding experimental design, result interpretation, and manuscript content were made and reviewed by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Experimental protocols, update budgets, hyperparameters, and compute resources are provided in Appendix G, including the VLA and goal-conditioned reuse settings. Full benchmark results and checkpoint schedules are reported in Appendices B and C, with additional VLA results and runtime-estimation details in Appendix D. Code is not publicly available at submission; we plan to release the training code and configurations upon publication.

REFERENCES

Haruki Abe, Takayuki Osa, Yusuke Mukuta, and Tatsuya Harada. Cross-Embodiment Offline Reinforcement Learning for Heterogeneous Robot Datasets. In *International Conference on Learning Representations (ICLR)*, 2026.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. *Advances in Neural Information Processing Systems*, 34:7436–7447, 2021.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 176–185. PMLR, 2017.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024.

Yevgen Chebotar, Quan Vuong, Karol Hausman, Fei Xia, Yao Lu, Alex Irpan, Aviral Kumar, Tianhe Yu, Alexander Herzog, Karl Pertsch, Keerthana Gopalakrishnan, Julian Ibarz, Ofir Nachum, Sumedh Anand Sontakke, Grecia Salazar, Huong T. Tran, Jodilyn Peralta, Clayton Tan, Deeksha Manjunath, Jaspiar Singh, Brianna Zitkovich, Tomas Jackson, Kanishka Rao, Chelsea Finn, and Sergey Levine. Q-transformer: Scalable offline reinforcement learning via autoregressive q-functions. In *Proceedings of The 7th Conference on Robot Learning*, 2023.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *The International Journal of Robotics Research (IJRR)*, pp. 1684–1704, 2025.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive Learning as Goal-Conditioned Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A Minimalist Approach to Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. In *International Conference on Machine Learning (ICML)*, 2018.

Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme Q-Learning: MaxEnt RL without Entropy. In *International Conference on Learning Representations (ICLR)*, 2023.

GigaBrain Team, Boyuan Wang, Bohan Li, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. GigaBrain-0.5M*: a VLA That Learns From World Model-Based Reinforcement Learning. *arXiv preprint arXiv:2602.12099*, 2026.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning (ICML)*, 2018.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. IDQL: Implicit Q-Learning as an Actor-Critic Method with Diffusion Policies. *arXiv preprint arXiv:2304.10573*, 2023.

Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion Model is an Effective Planner and Data Synthesizer for Multi-Task Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shengchao Hu, Ziqing Fan, Li Shen, Ya Zhang, Yanfeng Wang, and Dacheng Tao. HarmoDT: Harmony Multi-Task Decision Transformer for Offline Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2024.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient Diffusion Policies for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Changyeon Kim, Haeone Lee, Younggyo Seo, Kimin Lee, and Yuke Zhu. DEAS: DEtached value learning with Action Sequence for Scalable Offline RL. In *International Conference on Learning Representations (ICLR)*, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of The 8th Conference on Robot Learning*, 2024.

Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations (ICLR)*, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-Learning for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Aviral Kumar, Rishabh Agarwal, Xinyang Geng, George Tucker, and Sergey Levine. Offline Q-learning on Diverse Multi-Task Data Both Scales And Generalizes. In *International Conference on Learning Representations (ICLR)*, 2023.

Kuang-Huei Lee, Ofir Nachum, Mengjiao Sherry Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, et al. Multi-game decision transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *arXiv preprint arXiv:2005.01643*, 2020.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled Q-chunking. In *International Conference on Learning Representations (ICLR)*, 2026.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: a Diffusion Foundation Model for Bimanual Manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

Vivek Myers, Bill Chunyuan Zheng, Benjamin Eysenbach, and Sergey Levine. Offline Goal-conditioned Reinforcement Learning with Quasimetric Representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic Learning Datasets and RT-X Models: Open X-Embodiment Collaboration. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped DQN. *Advances in Neural Information Processing Systems*, 29, 2016.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline Goal-Conditioned RL with Latent States as Actions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is Value Learning Really the Main Bottleneck in Offline RL? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking Offline Goal-Conditioned RL. In *International Conference on Learning Representations (ICLR)*, 2025a.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon Reduction Makes RL Scalable. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-Learning. In *International Conference on Machine Learning (ICML)*, 2025c.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a VLA That Learns From Experience. *arXiv preprint arXiv:2511.14759*, 2025.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal Value Function Approximators. In *International Conference on Machine Learning (ICML)*, 2015.

Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual RL: Unification and New Methods for Reinforcement and Imitation Learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Jost Tobias Springenberg, Abbas Abdolmaleki, Jingwei Zhang, Oliver Groth, Michael Bloesch, Thomas Lampe, Philemon Brakel, Sarah Bechtle, Steven Kapturowski, Roland Hafner, et al. Offline Actor-Critic Reinforcement Learning Scales to Large Models. In *International Conference on Machine Learning (ICML)*, 2024.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal Goal-Reaching Reinforcement Learning via Quasimetric Learning. In *International Conference on Machine Learning (ICML)*, 2023a.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion Policies as an Expressive Policy Class for Offline Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2023b.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior Regularized Offline Reinforcement Learning. *arXiv preprint arXiv:1911.11361*, 2019.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline RL with No OOD Actions: In-Sample Learning via Implicit Value Regularization. In *International Conference on Learning Representations (ICLR)*, 2023.

A Critic Variability and Ensemble Diversity

This appendix extends our analysis of critic variability through additional state-level visualizations, a cross-task comparison with frozen-extraction degradation, a diagnostic of checkpoint sensitivity under sampling-based extraction, and an analysis of high-weight sample diversity across checkpoints and random initializations.

A.1 Additional Critic Variability Visualizations

In Section 4.1, we visualized critic variability for a single state in PointMaze. Here, we provide additional visualizations for different state locations. These plots include both action-space advantage heatmaps and action-ranking trajectories across critic checkpoints near the end of training.

A.1.1 Additional Advantage Heatmaps

Fig. 10 visualizes the action-space advantage landscape across critic checkpoints for different state locations. Each row shows the queried state on the maze and the corresponding advantage heatmaps over candidate actions computed from critic checkpoints near the end of training. Consistent with Fig. 3, the advantage landscape can change substantially across checkpoints. This variation is not limited to a single queried state: across multiple state locations, we observe changes in both the scale and spatial pattern of the advantage landscape.

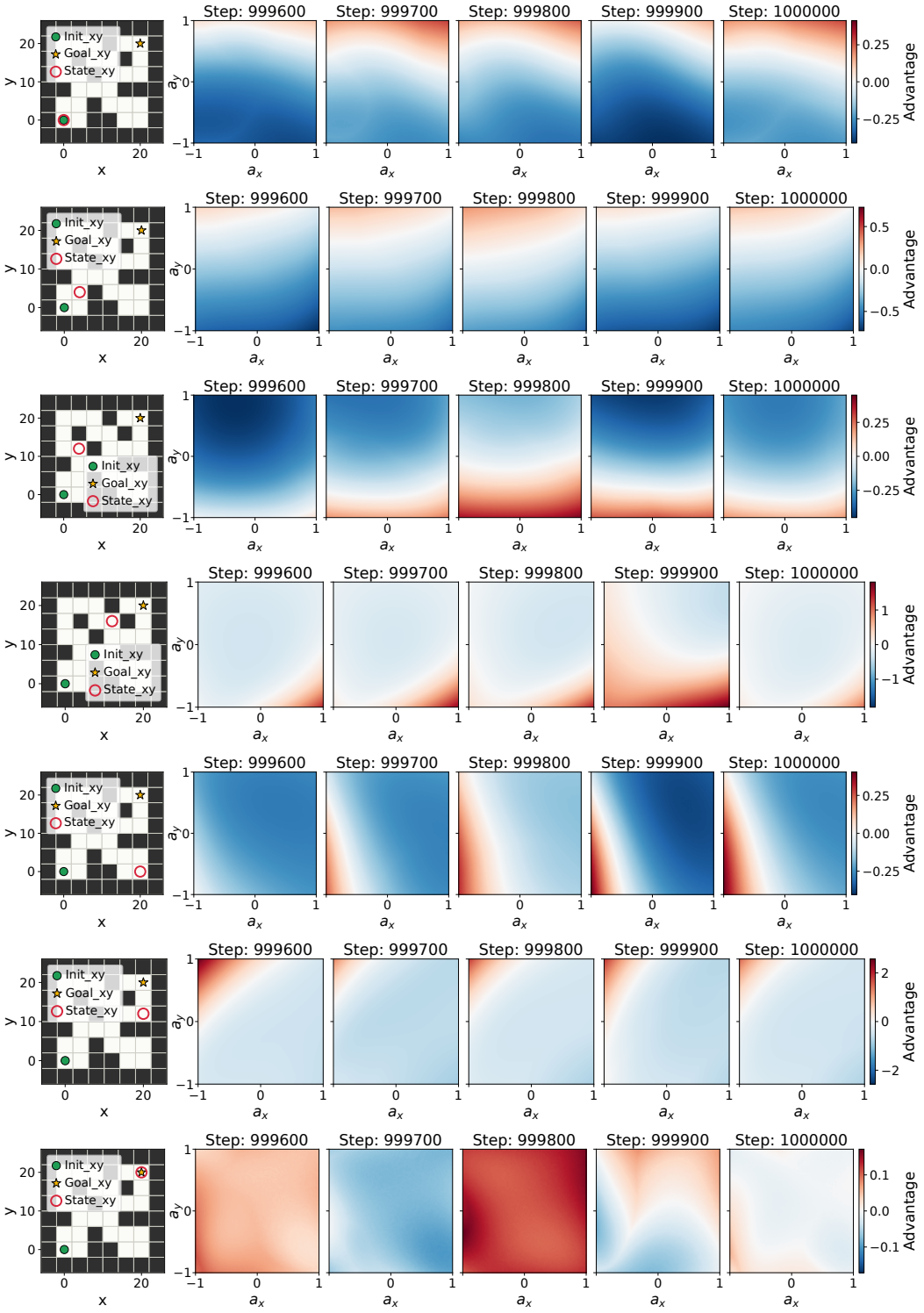

Figure 10: Additional advantage heatmaps for different PointMaze state locations. Each row shows the queried state on the maze and the corresponding action-space advantage landscapes across critic checkpoints near the end of training. These visualizations complement Fig. 3 and show that checkpoint-level variation appears across multiple state locations.

A.1.2 Additional Action-Ranking Visualizations

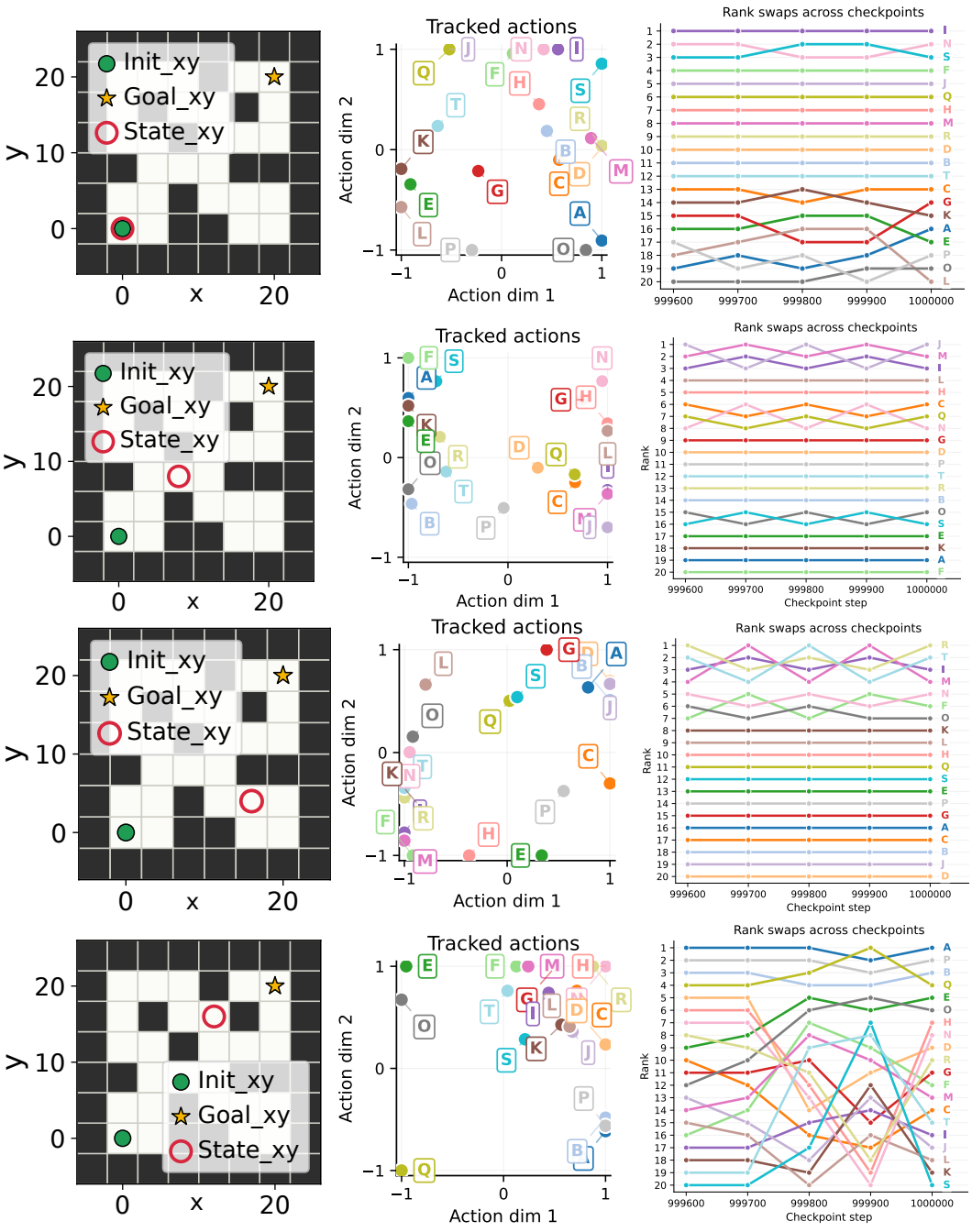

Figure 11: Additional action-ranking visualizations for different PointMaze state locations. Each row shows the queried state, the tracked candidate actions, and the relative ranking of those actions across critic checkpoints near the end of training. These examples complement Fig. 4 and show that checkpoint-level changes in action ranking are not specific to the single state shown in the main text.

Fig. 11 visualizes how the relative ranking of candidate actions changes across critic checkpoints for different state locations. Each row shows the queried state on the maze, the tracked candidate actions, and the corresponding rank trajectories across checkpoints. The degree of instability depends on the state: for some states, rankings change substantially across checkpoints, while for others the

top-ranked actions are comparatively stable. Nevertheless, rank swaps occur in all shown states. When actions move between high and low ranks across checkpoints, freezing a single critic may turn a transiently high-ranked action into a persistent extraction target, consistent with the spurious-peak mechanism discussed in the main text.

A.2 Cross-Task Relationship Between Critic Variability and Frozen-Extraction Degradation

We next examine whether greater late-stage critic variability is associated with larger performance degradation under single-critic Freeze-then-Extract. We compare `pointmaze-medium-navigate` with two D4RL medium locomotion tasks, `halfcheetah-medium` and `hopper-medium`, where frozen extraction causes little or no performance degradation.

For each task, we select 10 query states. Around each query state s, we find 20 nearby states in the offline dataset and collect their corresponding actions. We then fix the query state s and evaluate these candidate actions at s using five late-stage critic checkpoints. This allows us to measure how the relative evaluation of the same candidate actions changes across checkpoints. Because the candidate actions are collected from nearby rather than identical states, we also report the state radius of the selected neighborhood. A larger radius means that the candidate actions are taken from states farther from the query state; evaluating such actions at the fixed query state may create state-action pairs that are less common in the offline dataset, making the variability estimate less reliable. The action spread of the candidate actions is also reported and is closely matched across tasks.

We measure normalized action-rank changes and relative-advantage changes between adjacent checkpoints and compare them with the IQL performance gap between alternating and frozen extraction.

Table 1: Cross-task comparison of late-stage critic variability and frozen-extraction degradation. Fluctuation values are mean $\pm$ standard deviation across 10 query states. State radius summarizes how far the neighboring dataset states used to obtain candidate actions are from the query state. The final column reports Alt. $-$ Frozen IQL performance, so larger positive values indicate greater degradation under freezing.

Dataset	State radius	Action spread	Rank fluctuation	Relative-adv. fluctuation	Alt.–Frozen
PointMaze-medium	0.00100	0.37549	$\mathbf{0.0964 \pm 0.0893}$	$\mathbf{0.4208 \pm 0.1773}$	**28.0**
HalfCheetah-medium	0.10959	0.37537	0.0508 ± 0.0224	0.2143 ± 0.0880	-0.2
Hopper-medium	0.00832	0.37560	0.0476 ± 0.0438	0.1693 ± 0.1386	-1.6

Across these three tasks, greater critic variability tends to coincide with larger performance degradation under frozen extraction. PointMaze shows substantially larger rank and relative-advantage changes across checkpoints and also the largest performance drop. In contrast, HalfCheetah and Hopper show smaller checkpoint-to-checkpoint changes and little or no degradation from freezing. This pattern is consistent with temporal critic variability contributing to the fragility of single-critic policy extraction. However, the comparison includes only three tasks with different state and action spaces, dynamics, and dataset structure, so the result should be viewed as suggestive rather than conclusive.

A.3 Checkpoint Sensitivity under Sampling-Based Extraction

To examine whether checkpoint-specific critic variation can affect policy extraction even without fitting an AWR actor, we evaluate IFQL (Park et al., 2025c), a flow-based variant of IDQL-style sampling-based extraction (Hansen-Estruch et al., 2023). We first pretrain a flow proposal policy with behavior cloning and freeze it. We then keep the proposal policy and all extraction settings fixed while varying only the IQL critic checkpoint, evaluating checkpoints every $50{,}000$ updates from 0 to 10^6. This experiment is therefore a diagnostic of sensitivity to critic checkpoint selection rather than a comparison between alternating updates and Freeze-then-Extract. Table 2 reports selected

late-stage checkpoints from a single critic-training trajectory per task, together with the score range over all evaluated checkpoints from $500{,}000$ to 10^6 updates.

Table 2: IFQL checkpoint sensitivity with a fixed flow proposal. Checkpoint steps are in thousands. Range denotes the minimum and maximum scores across all evaluated checkpoints from 500k to 1000k, including those not displayed.

Task	500k	650k	750k	1000k	Range
pointmaze-large-navigate	0.0	100.0	1.0	95.0	0.0–100.0
cube-double-play	82.0	53.0	59.0	48.0	35.0–82.0
antmaze-large-play-v2	35.0	33.0	52.0	26.0	20.0–52.0
antmaze-large-diverse-v2	39.0	31.0	27.0	32.0	18.0–39.0
walker2d-medium-replay-v2	72.2	73.6	70.4	79.4	66.6–83.9

Performance remains checkpoint-dependent and non-monotonic on several tasks even late in critic training. Because the proposal policy and extraction procedure are fixed, these differences reflect sensitivity to the selected critic checkpoint. Thus, late-stage critic variability can translate into meaningful performance differences even under an expressive sampling-based extractor, showing that dependence on a particular critic checkpoint is not specific to Gaussian AWR fitting. This experiment does not measure a Freeze-then-Extract performance gap or compare against alternating training. Critic hyperparameters are inherited from the AWR-based IQL experiments, and IFQL-specific tuning may reduce the observed checkpoint sensitivity.

A.4 Checkpoint-wise and Initialization-wise Diversity of High-Weight Samples

In the main text, we interpreted alternating updates as implicit temporal ensembling, since high-weight samples change across critic checkpoints. Here, we examine whether similar diversity arises across critics trained with different random initializations. We compare the two sources of variation underlying Checkpoint Ensemble and Independent Ensemble: checkpoints from a single critic-training run and critics trained with different random initializations. In both cases, we compute AWR weights over the offline dataset, select the top 10% highest-weight samples, and measure pairwise Jaccard overlap between the resulting sample sets.

Fig. 12 compares checkpoint-wise and initialization-wise overlaps on `antmaze-large-diverse` and `pointmaze-medium-navigate`. The initialization-wise overlaps exhibit non-trivial diversity, similar to the checkpoint-wise overlaps, showing that critics with different initial parameters induce different high-weight sample sets even under the same training procedure. Across these two tasks, checkpoint-wise and initialization-wise overlaps also exhibit a qualitatively similar task-dependent pattern: lower overlap across checkpoints is accompanied by lower overlap across initializations, and vice versa. These results support the motivation of Independent Ensemble. Critics trained with different random initializations provide diverse but related sample-weighting signals, allowing policy extraction to aggregate multiple critic judgments.

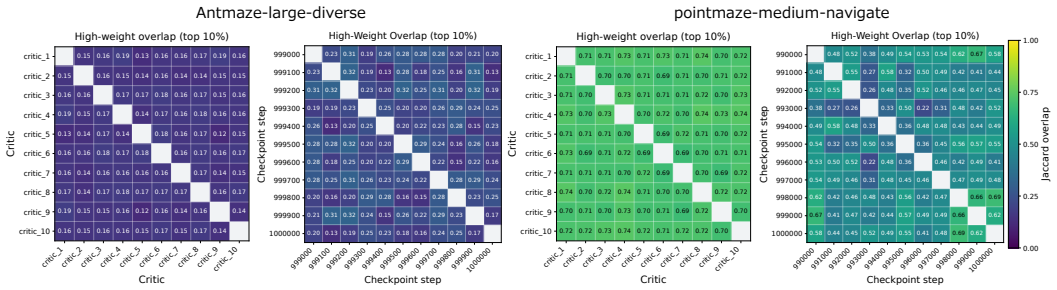

Figure 12: Pairwise Jaccard overlap of top-10% AWR-weighted sample sets for checkpoint-wise and initialization-wise critic comparisons on `antmaze-large-diverse` and `pointmaze-medium-navigate`.

B Additional Offline RL Results

This appendix reports complete per-task results for the main offline RL benchmark comparison. We compare standard alternating updates (Alt.), single-critic Freeze-then-Extract (Frozen), Independent Ensemble (Indep.), Checkpoint Ensemble with the fixed Tail-10k schedule (Ckpt.), and a tuned checkpoint result. Details of the checkpoint schedules and the complete schedule-wise results are provided in Appendix C.

B.1 Complete Per-Task Score Tables

Table 3 summarizes the simple mean over all 25 task-level means. Tables 4–9 provide the corresponding task-level results. Checkpoint (Tuned) reports the best-performing result among Tail-1k, Tail-10k, Tail-50k, and Full-100k for each algorithm–task pair.

Table 3: Mean normalized score across all 25 tasks. "Indep." denotes Independent Ensemble, "Ckpt." denotes Checkpoint Ensemble with the fixed Tail-10k schedule, and "Ckpt. (Tuned)" reports the best-performing checkpoint schedule for each algorithm–task pair.

Algorithm	Alt.	Frozen	Indep.	Ckpt.	Ckpt. (Tuned)
IQL	60.11	51.51	59.90	57.44	62.12
XQL	45.91	40.21	45.84	46.06	48.95
SQL	52.31	39.56	48.87	47.48	51.20

Table 4: D4RL per-task results for IQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-umaze-v2	87.4 ± 1.3	84.0 ± 4.2	84.6 ± 0.6	93.60 ± 0.98	93.60 ± 0.98
antmaze-umaze-diverse-v2	68.0 ± 3.1	65.4 ± 3.7	73.6 ± 2.6	62.40 ± 1.86	73.60 ± 2.54
antmaze-medium-play-v2	71.6 ± 1.0	61.6 ± 13.6	74.4 ± 1.4	69.20 ± 2.11	70.60 ± 4.57
antmaze-medium-diverse-v2	75.2 ± 2.2	54.6 ± 4.7	70.4 ± 2.7	64.80 ± 2.42	69.40 ± 1.86
antmaze-large-play-v2	41.2 ± 2.1	23.6 ± 5.4	49.8 ± 4.7	36.40 ± 1.69	36.40 ± 1.69
antmaze-large-diverse-v2	44.8 ± 2.3	17.0 ± 5.4	44.6 ± 5.8	39.80 ± 2.22	42.00 ± 3.27
halfcheetah-medium-v2	47.4 ± 0.1	47.6 ± 0.1	47.9 ± 0.1	47.56 ± 0.06	47.61 ± 0.08
hopper-medium-v2	62.9 ± 3.4	64.5 ± 3.1	69.0 ± 5.1	61.63 ± 2.93	64.53 ± 4.97
walker2d-medium-v2	80.8 ± 0.6	78.7 ± 2.8	79.8 ± 1.1	77.73 ± 2.79	79.32 ± 2.08
halfcheetah-medium-replay-v2	43.9 ± 0.7	40.9 ± 0.8	44.6 ± 0.3	43.18 ± 0.47	43.87 ± 0.38
hopper-medium-replay-v2	93.6 ± 4.8	75.7 ± 7.2	78.7 ± 6.5	77.95 ± 9.87	82.52 ± 6.16
walker2d-medium-replay-v2	66.9 ± 4.3	62.6 ± 4.1	68.5 ± 6.4	66.82 ± 3.74	68.66 ± 5.97
kitchen-complete-v0	63.8 ± 1.1	67.8 ± 0.9	60.3 ± 0.6	63.25 ± 1.99	67.55 ± 2.11
kitchen-partial-v0	54.6 ± 4.0	64.6 ± 5.3	63.9 ± 6.4	58.30 ± 6.66	63.65 ± 5.19
kitchen-mixed-v0	48.5 ± 2.1	50.1 ± 1.8	48.1 ± 2.6	50.05 ± 2.21	51.50 ± 2.57
pen-human-v1	76.9 ± 2.4	69.8 ± 2.3	74.4 ± 2.3	75.58 ± 0.90	76.33 ± 3.24
pen-cloned-v1	72.6 ± 4.4	73.2 ± 2.0	76.1 ± 1.8	74.18 ± 1.86	74.18 ± 1.86

Table 5: OGBench single-task results for IQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-large-navigate-singletask-task1-v0	46.6 ± 6.5	15.8 ± 3.0	42.6 ± 5.0	37.60 ± 3.88	46.20 ± 7.27
antmaze-giant-navigate-singletask-task1-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	31.8 ± 7.3	21.0 ± 6.7	23.4 ± 7.3	29.80 ± 7.76	44.00 ± 1.70
cube-single-play-singletask-task1-v0	78.4 ± 4.7	58.0 ± 6.0	85.6 ± 2.4	77.60 ± 3.37	92.00 ± 1.22
cube-double-play-singletask-task1-v0	33.8 ± 6.4	22.0 ± 4.0	30.2 ± 4.6	33.40 ± 3.54	50.40 ± 2.77
pointmaze-medium-navigate-singletask-task1-v0	88.0 ± 5.5	60.0 ± 24.5	69.0 ± 18.5	100.00 ± 0.00	100.00 ± 0.00
pointmaze-medium-stitch-singletask-task1-v0	58.6 ± 22.0	79.6 ± 19.9	97.8 ± 2.2	40.00 ± 24.49	59.80 ± 24.41
pointmaze-large-navigate-singletask-task1-v0	65.4 ± 16.5	29.6 ± 18.3	40.2 ± 17.8	55.20 ± 16.68	55.20 ± 16.68

Table 6: D4RL per-task results for XQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-umaze-v2	85.0 ± 2.4	80.2 ± 1.9	87.8 ± 2.1	89.40 ± 1.63	89.40 ± 1.63
antmaze-umaze-diverse-v2	46.0 ± 1.6	49.0 ± 3.5	50.2 ± 3.3	50.80 ± 3.31	52.80 ± 2.97
antmaze-medium-play-v2	52.0 ± 2.1	58.0 ± 1.4	62.0 ± 4.3	54.00 ± 3.94	56.20 ± 3.89
antmaze-medium-diverse-v2	57.6 ± 5.7	47.4 ± 12.1	45.4 ± 4.7	51.80 ± 12.75	64.40 ± 2.66
antmaze-large-play-v2	36.0 ± 3.8	21.2 ± 4.5	29.4 ± 1.6	27.40 ± 3.66	30.80 ± 3.57
antmaze-large-diverse-v2	37.6 ± 2.4	14.8 ± 5.9	24.4 ± 5.3	21.60 ± 3.61	32.20 ± 4.10
halfcheetah-medium-v2	47.2 ± 0.1	47.4 ± 0.1	47.3 ± 0.1	47.53 ± 0.13	47.53 ± 0.13
hopper-medium-v2	67.0 ± 2.6	66.2 ± 3.1	66.5 ± 1.4	67.68 ± 2.00	68.60 ± 1.72
walker2d-medium-v2	82.2 ± 0.4	83.6 ± 0.5	83.3 ± 1.1	81.91 ± 1.77	81.91 ± 1.77
halfcheetah-medium-replay-v2	42.9 ± 0.5	42.4 ± 0.4	42.7 ± 0.7	43.40 ± 0.27	43.54 ± 0.12
hopper-medium-replay-v2	63.8 ± 13.0	55.0 ± 12.7	85.5 ± 7.8	74.54 ± 4.82	83.52 ± 7.41
walker2d-medium-replay-v2	67.7 ± 2.8	63.1 ± 5.7	61.1 ± 4.2	59.07 ± 4.37	68.07 ± 7.39
kitchen-complete-v0	37.2 ± 7.8	44.1 ± 3.1	32.5 ± 10.8	38.35 ± 7.77	38.35 ± 7.77
kitchen-partial-v0	67.9 ± 2.9	44.5 ± 3.1	67.5 ± 2.1	64.50 ± 3.17	64.50 ± 3.17
kitchen-mixed-v0	42.2 ± 5.9	36.0 ± 4.5	49.1 ± 2.5	47.25 ± 3.59	51.30 ± 1.39
pen-human-v1	70.7 ± 1.7	70.0 ± 2.1	74.1 ± 1.7	77.99 ± 2.86	78.88 ± 4.23
pen-cloned-v1	66.5 ± 3.0	59.5 ± 6.0	69.3 ± 2.6	67.89 ± 3.87	70.06 ± 4.03

Table 7: OGBench single-task results for XQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-large-navigate-singletask-task1-v0	0.0 ± 0.0	0.2 ± 0.2	0.0 ± 0.0	0.20 ± 0.20	0.60 ± 0.24
antmaze-giant-navigate-singletask-task1-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	15.8 ± 3.2	15.8 ± 3.3	11.4 ± 1.1	16.80 ± 3.89	18.00 ± 5.85
cube-single-play-singletask-task1-v0	93.8 ± 1.3	62.4 ± 5.7	90.0 ± 1.9	83.40 ± 5.03	95.60 ± 1.72
cube-double-play-singletask-task1-v0	8.6 ± 3.6	4.4 ± 1.7	9.8 ± 2.6	6.60 ± 2.18	7.60 ± 2.66
pointmaze-medium-navigate-singletask-task1-v0	60.0 ± 24.5	40.0 ± 24.5	56.8 ± 23.4	79.40 ± 19.86	80.00 ± 20.00
pointmaze-medium-stitch-singletask-task1-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00
pointmaze-large-navigate-singletask-task1-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00

Table 8: D4RL per-task results for SQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-umaze-v2	84.6 ± 1.6	83.6 ± 0.8	86.0 ± 1.0	86.20 ± 1.43	87.40 ± 1.47
antmaze-umaze-diverse-v2	41.8 ± 3.3	37.8 ± 8.5	39.2 ± 1.1	40.20 ± 2.78	47.20 ± 3.68
antmaze-medium-play-v2	47.4 ± 13.9	33.0 ± 12.2	43.0 ± 8.8	47.40 ± 11.45	61.60 ± 7.59
antmaze-medium-diverse-v2	16.0 ± 2.8	8.4 ± 4.1	12.2 ± 1.7	7.00 ± 1.67	16.40 ± 4.41
antmaze-large-play-v2	35.8 ± 2.0	9.6 ± 4.7	37.2 ± 3.4	23.80 ± 6.30	40.80 ± 2.94
antmaze-large-diverse-v2	33.4 ± 2.6	21.8 ± 5.1	31.4 ± 3.7	33.20 ± 4.27	40.00 ± 4.06
halfcheetah-medium-v2	47.6 ± 0.1	47.9 ± 0.1	47.6 ± 0.1	47.47 ± 0.11	47.48 ± 0.15
hopper-medium-v2	65.3 ± 1.3	63.9 ± 4.1	65.8 ± 1.4	67.96 ± 2.03	68.13 ± 0.66
walker2d-medium-v2	81.9 ± 0.8	84.6 ± 0.2	82.6 ± 1.6	82.59 ± 1.02	82.89 ± 2.07
halfcheetah-medium-replay-v2	42.7 ± 0.4	42.1 ± 0.7	43.5 ± 0.0	42.58 ± 0.37	43.14 ± 0.30
hopper-medium-replay-v2	85.1 ± 5.5	88.1 ± 6.6	85.6 ± 9.6	78.14 ± 9.04	82.98 ± 7.07
walker2d-medium-replay-v2	71.3 ± 4.7	62.4 ± 5.7	73.1 ± 4.5	75.71 ± 5.09	78.44 ± 2.86
kitchen-complete-v0	63.8 ± 3.4	37.1 ± 9.3	50.3 ± 3.8	56.10 ± 3.41	56.95 ± 4.52
kitchen-partial-v0	24.1 ± 4.9	3.5 ± 3.5	15.3 ± 5.0	22.55 ± 5.72	22.55 ± 5.72
kitchen-mixed-v0	10.2 ± 4.5	4.8 ± 2.7	20.9 ± 2.1	8.55 ± 5.07	12.45 ± 4.84
pen-human-v1	69.5 ± 2.4	50.0 ± 2.8	71.6 ± 3.1	72.75 ± 1.16	78.30 ± 3.10
pen-cloned-v1	73.3 ± 1.6	67.4 ± 5.6	67.1 ± 3.5	72.74 ± 3.02	72.96 ± 3.26

Table 9: OGBench single-task results for SQL. Values are mean ± SEM.

Task	Alt.	Frozen	Indep.	Ckpt. (Tail-10k)	Ckpt. (Tuned)
antmaze-large-navigate-singletask-task1-v0	2.0 ± 1.8	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	5.80 ± 4.81
antmaze-giant-navigate-singletask-task1-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	31.2 ± 5.0	22.6 ± 7.8	19.2 ± 3.2	24.40 ± 5.92	24.60 ± 6.20
cube-single-play-singletask-task1-v0	89.0 ± 2.0	68.6 ± 17.0	88.6 ± 1.5	85.80 ± 2.44	88.20 ± 2.44
cube-double-play-singletask-task1-v0	35.8 ± 1.5	13.6 ± 2.7	35.2 ± 6.2	24.60 ± 4.02	34.20 ± 6.18
pointmaze-medium-navigate-singletask-task1-v0	100.0 ± 0.0	60.0 ± 24.5	100.0 ± 0.0	100.00 ± 0.00	100.00 ± 0.00
pointmaze-medium-stitch-singletask-task1-v0	99.8 ± 0.2	58.2 ± 23.8	60.0 ± 24.5	19.40 ± 19.40	19.80 ± 19.80
pointmaze-large-navigate-singletask-task1-v0	56.2 ± 7.6	20.0 ± 20.0	46.4 ± 19.8	67.80 ± 1.11	67.80 ± 1.11

C Checkpoint Ensemble Schedule Sensitivity

We further examine how the temporal coverage of the checkpoint set affects policy extraction. All variants use $M = 10$ checkpoints from a single value-learning trajectory and differ only in the selected checkpoint steps. Tail-1k, Tail-10k, and Tail-50k progressively expand the temporal span while remaining concentrated toward the end of value learning, whereas Full-100k spans nearly the entire value-learning trajectory. Tail-10k is used as the fixed Checkpoint Ensemble configuration in the main comparison.

Table 10: Checkpoint schedules used for Checkpoint Ensemble. Each schedule contains 10 checkpoints.

Schedule	Checkpoint steps
Tail-1k	$991{,}000, 992{,}000, \ldots, 1{,}000{,}000$
Tail-10k	$910{,}000, 920{,}000, \ldots, 1{,}000{,}000$
Tail-50k	$550{,}000, 600{,}000, \ldots, 1{,}000{,}000$
Full-100k	$100{,}000, 200{,}000, \ldots, 1{,}000{,}000$

Each value is averaged over five seeds and reported as mean ± SEM. Table 11 summarizes the mean across the 25 tasks. No single schedule performs best for all algorithms: Tail-10k is best for IQL, Tail-1k for XQL, and Tail-50k for SQL. We therefore use Tail-10k as a single fixed configuration in the main comparison and report the best per-task schedule as Checkpoint (Tuned).

Table 11: Sensitivity to checkpoint schedule. Values are the mean across the 25 tasks.

Algorithm	Tail-1k	Tail-10k	Tail-50k	Full-100k
IQL	56.54	**57.44**	56.25	57.18
XQL	**46.35**	46.06	45.46	45.10
SQL	46.53	47.48	**48.42**	47.11

C.1 IQL

Table 12: D4RL checkpoint-schedule results for IQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-umaze-v2	89.20 ± 1.07	93.60 ± 0.98	91.60 ± 0.87	87.60 ± 4.12
antmaze-umaze-diverse-v2	66.80 ± 2.96	62.40 ± 1.86	65.20 ± 4.54	73.60 ± 2.54
antmaze-medium-play-v2	69.60 ± 3.91	69.20 ± 2.11	65.20 ± 3.61	70.60 ± 4.57
antmaze-medium-diverse-v2	62.80 ± 1.98	64.80 ± 2.42	64.00 ± 3.05	69.40 ± 1.86
antmaze-large-play-v2	35.40 ± 5.85	36.40 ± 1.69	33.40 ± 2.01	21.40 ± 5.27
antmaze-large-diverse-v2	38.80 ± 4.15	39.80 ± 2.22	42.00 ± 3.27	35.80 ± 6.26
halfcheetah-medium-v2	47.52 ± 0.07	47.56 ± 0.06	47.61 ± 0.08	46.92 ± 0.08
hopper-medium-v2	64.53 ± 4.97	61.63 ± 2.93	63.51 ± 3.40	59.57 ± 2.46
walker2d-medium-v2	77.51 ± 1.87	77.73 ± 2.79	79.32 ± 2.08	76.98 ± 1.93
halfcheetah-medium-replay-v2	43.31 ± 0.58	43.18 ± 0.47	43.87 ± 0.38	42.79 ± 0.35
hopper-medium-replay-v2	82.52 ± 6.16	77.95 ± 9.87	80.62 ± 7.24	80.65 ± 6.77
walker2d-medium-replay-v2	64.03 ± 3.35	66.82 ± 3.74	62.87 ± 3.40	68.66 ± 5.97
kitchen-complete-v0	64.20 ± 1.55	63.25 ± 1.99	67.55 ± 2.11	62.85 ± 2.22
kitchen-partial-v0	63.65 ± 5.19	58.30 ± 6.66	52.10 ± 8.25	47.10 ± 7.42
kitchen-mixed-v0	49.05 ± 2.65	50.05 ± 2.21	45.60 ± 3.22	51.50 ± 2.57
pen-human-v1	76.33 ± 3.24	75.58 ± 0.90	73.22 ± 0.81	75.41 ± 2.00
pen-cloned-v1	74.15 ± 2.86	74.18 ± 1.86	72.74 ± 3.14	72.80 ± 2.91

Table 13: OGBench checkpoint-schedule results for IQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-large-navigate-singletask-task1-v0	35.60 ± 6.66	37.60 ± 3.88	37.40 ± 2.84	46.20 ± 7.27
antmaze-giant-navigate-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	16.60 ± 6.32	29.80 ± 7.76	44.00 ± 1.70	40.40 ± 3.33
cube-single-play-singletask-task1-v0	76.20 ± 3.34	77.60 ± 3.37	90.40 ± 1.81	92.00 ± 1.22
cube-double-play-singletask-task1-v0	25.80 ± 6.18	33.40 ± 3.54	30.60 ± 4.70	50.40 ± 2.77
pointmaze-medium-navigate-singletask-task1-v0	81.00 ± 7.56	100.00 ± 0.00	75.00 ± 18.96	93.60 ± 3.06
pointmaze-medium-stitch-singletask-task1-v0	59.80 ± 24.41	40.00 ± 24.49	49.40 ± 21.64	20.00 ± 20.00
pointmaze-large-navigate-singletask-task1-v0	49.20 ± 20.12	55.20 ± 16.68	29.00 ± 18.26	43.20 ± 17.76

C.2 XQL

Table 14: D4RL checkpoint-schedule results for XQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-umaze-v2	87.20 ± 2.13	89.40 ± 1.63	89.40 ± 1.60	88.40 ± 1.72
antmaze-umaze-diverse-v2	52.80 ± 2.97	50.80 ± 3.31	45.40 ± 3.26	50.20 ± 4.73
antmaze-medium-play-v2	56.20 ± 3.89	54.00 ± 3.94	54.20 ± 2.13	49.60 ± 2.46
antmaze-medium-diverse-v2	55.20 ± 11.43	51.80 ± 12.75	53.60 ± 3.08	64.40 ± 2.66
antmaze-large-play-v2	25.60 ± 3.91	27.40 ± 3.66	29.00 ± 3.10	30.80 ± 3.57
antmaze-large-diverse-v2	29.20 ± 3.51	21.60 ± 3.61	28.40 ± 2.14	32.20 ± 4.10
halfcheetah-medium-v2	47.29 ± 0.14	47.53 ± 0.13	47.30 ± 0.18	47.06 ± 0.17
hopper-medium-v2	68.60 ± 1.72	67.68 ± 2.00	62.99 ± 1.81	68.58 ± 2.41
walker2d-medium-v2	81.19 ± 2.82	81.91 ± 1.77	81.73 ± 0.76	81.06 ± 2.12
halfcheetah-medium-replay-v2	43.54 ± 0.12	43.40 ± 0.27	43.11 ± 0.27	42.88 ± 0.61
hopper-medium-replay-v2	72.56 ± 12.52	74.54 ± 4.82	83.52 ± 7.41	82.34 ± 6.16
walker2d-medium-replay-v2	66.95 ± 4.14	59.07 ± 4.37	68.07 ± 7.39	63.71 ± 5.10
kitchen-complete-v0	29.80 ± 9.72	38.35 ± 7.77	19.65 ± 5.06	31.40 ± 9.50
kitchen-partial-v0	63.20 ± 3.77	64.50 ± 3.17	59.30 ± 10.33	59.00 ± 6.67
kitchen-mixed-v0	42.95 ± 4.57	47.25 ± 3.59	40.40 ± 4.57	51.30 ± 1.39
pen-human-v1	76.31 ± 3.16	77.99 ± 2.86	78.88 ± 4.23	72.16 ± 3.07
pen-cloned-v1	69.26 ± 2.61	67.89 ± 3.87	67.19 ± 2.43	70.06 ± 4.03

Table 15: OGBench checkpoint-schedule results for XQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-large-navigate-singletask-task1-v0	0.60 ± 0.24	0.20 ± 0.20	0.20 ± 0.20	0.20 ± 0.20
antmaze-giant-navigate-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	18.00 ± 5.85	16.80 ± 3.89	11.60 ± 2.52	12.00 ± 1.52
cube-single-play-singletask-task1-v0	85.40 ± 3.37	83.40 ± 5.03	86.20 ± 1.56	95.60 ± 1.72
cube-double-play-singletask-task1-v0	7.00 ± 0.55	6.60 ± 2.18	7.60 ± 2.66	5.80 ± 1.83
pointmaze-medium-navigate-singletask-task1-v0	80.00 ± 20.00	79.40 ± 19.86	78.80 ± 19.73	28.80 ± 17.71
pointmaze-medium-stitch-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
pointmaze-large-navigate-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

C.3 SQL

Table 16: D4RL checkpoint-schedule results for SQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-umaze-v2	87.40 ± 1.47	86.20 ± 1.43	86.20 ± 1.85	84.00 ± 1.48
antmaze-umaze-diverse-v2	47.20 ± 3.68	40.20 ± 2.78	42.60 ± 4.31	43.40 ± 1.89
antmaze-medium-play-v2	38.00 ± 13.35	47.40 ± 11.45	61.40 ± 7.08	61.60 ± 7.59
antmaze-medium-diverse-v2	13.20 ± 4.03	7.00 ± 1.67	16.40 ± 4.41	13.20 ± 6.73
antmaze-large-play-v2	20.80 ± 6.73	23.80 ± 6.30	34.40 ± 4.30	40.80 ± 2.94
antmaze-large-diverse-v2	32.80 ± 3.43	33.20 ± 4.27	33.60 ± 2.18	40.00 ± 4.06
halfcheetah-medium-v2	47.48 ± 0.15	47.47 ± 0.11	47.29 ± 0.10	47.08 ± 0.17
hopper-medium-v2	68.13 ± 0.66	67.96 ± 2.03	62.70 ± 1.26	63.02 ± 3.54
walker2d-medium-v2	82.89 ± 2.07	82.59 ± 1.02	82.60 ± 1.00	82.08 ± 1.39
halfcheetah-medium-replay-v2	42.68 ± 0.19	42.58 ± 0.37	43.14 ± 0.30	43.10 ± 0.44
hopper-medium-replay-v2	82.98 ± 7.07	78.14 ± 9.04	68.76 ± 8.38	82.54 ± 8.58
walker2d-medium-replay-v2	72.82 ± 5.73	75.71 ± 5.09	78.44 ± 2.86	69.10 ± 4.53
kitchen-complete-v0	56.95 ± 4.52	56.10 ± 3.41	56.90 ± 3.90	55.60 ± 2.71
kitchen-partial-v0	20.00 ± 7.57	22.55 ± 5.72	16.55 ± 4.86	19.70 ± 6.39
kitchen-mixed-v0	12.45 ± 4.84	8.55 ± 5.07	11.90 ± 5.35	7.10 ± 5.18
pen-human-v1	77.34 ± 2.37	72.75 ± 1.16	75.19 ± 0.99	78.30 ± 3.10
pen-cloned-v1	72.96 ± 3.26	72.74 ± 3.02	69.69 ± 5.69	71.52 ± 2.93

Table 17: OGBench checkpoint-schedule results for SQL. Values are mean ± SEM over five seeds.

Task	Tail-1k	Tail-10k	Tail-50k	Full-100k
antmaze-large-navigate-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.20 ± 0.20	5.80 ± 4.81
antmaze-giant-navigate-singletask-task1-v0	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
humanoidmaze-medium-navigate-singletask-task1-v0	18.00 ± 5.73	24.40 ± 5.92	24.60 ± 6.20	19.00 ± 5.29
cube-single-play-singletask-task1-v0	75.20 ± 16.34	85.80 ± 2.44	88.20 ± 2.44	86.60 ± 3.94
cube-double-play-singletask-task1-v0	23.00 ± 3.18	24.60 ± 4.02	25.60 ± 3.12	34.20 ± 6.18
pointmaze-medium-navigate-singletask-task1-v0	99.80 ± 0.20	100.00 ± 0.00	100.00 ± 0.00	60.00 ± 24.49
pointmaze-medium-stitch-singletask-task1-v0	19.80 ± 19.80	19.40 ± 19.40	19.80 ± 19.80	19.40 ± 19.40
pointmaze-large-navigate-singletask-task1-v0	51.40 ± 12.90	67.80 ± 1.11	64.40 ± 1.99	50.60 ± 12.80

Overall, the effect of temporal coverage is strongly task dependent. Wider checkpoint spacing can provide additional diversity and substantially improve some tasks, but it can also include earlier critics that are less suitable for extraction. The aggregate results therefore do not identify a universally optimal checkpoint schedule.

D Additional VLA Results and Computational Efficiency

This appendix complements Section 6.2 with detailed policy-learning results and analyses of projected runtime, advantage caching, reuse across policy configurations, and Multi-head Ensemble overhead. Dataset, architecture, and training settings are provided in Appendix G.4.

D.1 Detailed Policy-Learning Results

Fig. 8(a,b) shows the policy-learning curves summarized in the main text. Tables 18 and 19 report success rates at every evaluated checkpoint, from initialization to 30k actor updates for CoffeeSetupMug and 60k for TurnOffStove. Within each task, methods are compared at the same actor-update budget; critic pretraining and cache construction are excluded from this axis and considered separately in the runtime analysis. Results are reported as the mean ± standard error of the mean (SEM) across five policy-training runs. Each trained checkpoint is evaluated over 250 episodes, comprising 50 episodes for each of five evaluation seeds.

Table 18: Success rates (%) on `CoffeeSetupMug` at each actor-update checkpoint, reported as mean ± SEM across five policy-training runs. Base is the shared initialization policy.

Actor updates	Alt.	Frozen	Multi-head Ensemble
0 (Base)	17.60	17.60	17.60
5k	27.60 ± 2.61	29.68 ± 5.07	$\mathbf{32.24 \pm 2.69}$
10k	31.44 ± 3.21	37.44 ± 2.86	$\mathbf{40.32 \pm 3.43}$
15k	36.16 ± 2.46	40.80 ± 2.76	$\mathbf{44.00 \pm 2.26}$
20k	36.40 ± 2.47	38.64 ± 4.89	$\mathbf{43.36 \pm 2.55}$
25k	45.28 ± 2.25	41.92 ± 2.87	$\mathbf{48.16 \pm 2.24}$
30k	$\mathbf{47.12 \pm 1.37}$	45.04 ± 2.25	46.88 ± 1.31

Table 19: Success rates (%) on `TurnOffStove` at each actor-update checkpoint, reported as mean ± SEM across five policy-training runs. Base is the shared initialization policy.

Actor updates	Alt.	Frozen	Multi-head Ensemble
0 (Base)	28.40	28.40	28.40
5k	22.48 ± 1.24	28.64 ± 1.65	$\mathbf{31.04 \pm 2.03}$
10k	25.28 ± 2.47	28.88 ± 2.40	$\mathbf{33.92 \pm 2.19}$
15k	28.48 ± 1.73	32.72 ± 1.37	$\mathbf{35.36 \pm 0.88}$
20k	28.96 ± 1.10	32.32 ± 0.39	$\mathbf{37.36 \pm 0.57}$
25k	29.52 ± 1.15	34.48 ± 1.24	$\mathbf{37.92 \pm 1.10}$
30k	31.68 ± 1.21	34.88 ± 2.01	$\mathbf{37.84 \pm 0.43}$
35k	33.68 ± 1.40	36.40 ± 1.02	$\mathbf{40.72 \pm 0.92}$
40k	31.76 ± 2.62	$\mathbf{40.48 \pm 1.86}$	38.32 ± 0.70
45k	32.88 ± 0.97	38.32 ± 0.53	$\mathbf{40.96 \pm 1.12}$
50k	35.68 ± 1.55	37.76 ± 2.21	$\mathbf{39.92 \pm 1.25}$
55k	33.36 ± 1.26	39.52 ± 1.38	$\mathbf{40.32 \pm 1.82}$
60k	34.32 ± 1.38	37.44 ± 1.09	$\mathbf{39.92 \pm 1.25}$

On CoffeeSetupMug, Multi-head Ensemble achieves the **highest mean success rate at every evaluated checkpoint from 5k through 25k actor updates**. At 15k updates, it reaches 44.00%, compared with 40.80% for Frozen and 36.16% for Alt. By 30k updates, Alt. attains a slightly higher mean than Multi-head Ensemble (47.12% versus 46.88%), indicating that the early-update advantage does not imply a higher final mean.

On TurnOffStove, both frozen-critic methods outperform Alt. in mean success rate at every evaluated nonzero checkpoint. Multi-head Ensemble has the **highest mean at all checkpoints except 40k**, and finishes at 39.92%, compared with 37.44% for Frozen and 34.32% for Alt. These results support improved actor-update efficiency, particularly for Multi-head Ensemble extraction, while also showing non-monotonic learning curves and variation across training runs.

D.2 Advantage Caching and Projected Runtime

Frozen extraction caches raw action-chunk advantages for every critic head. Actor updates then apply the corresponding clipped weights to the per-sample flow-matching loss without repeated critic

inference. In contrast, alternating training performs critic updates and computes advantages online throughout actor training.

Timing protocol. The projected runtimes in Fig. 8(c,d) are obtained from completed 2,000-update benchmarks on a single NVIDIA GB200 GPU. We compute the average step time over updates 201–2000 to reduce startup and warm-up effects, then extrapolate it to the full 30k or 60k training budget while retaining the measured one-time startup overhead. Cache construction is measured on a subset of the dataset and scaled to the full dataset. Evaluation and checkpoint saving are excluded. Thus, the reported runtimes are projections from short benchmarks rather than direct measurements of the complete training runs.

Table 20: Projected training-pipeline time in elapsed hours at 30k critic/actor updates for Coffee and 60k for Stove. Values are extrapolated from short benchmarks rather than directly measured full-budget runtimes. For Alt., the joint-training column includes both critic and actor updates.

Task	Method	Critic	Cache	Actor / joint training	Total
Coffee	Alt.	–	–	20.10	20.10
	Frozen	9.24	0.93	6.40	16.57
	Multi-head	10.13	0.96	6.40	17.49
Stove	Alt.	–	–	77.77	77.77
	Frozen	35.77	4.19	23.93	63.89
	Multi-head	37.66	4.28	24.80	66.75

At the full actor-update budget, in our evaluated implementation, Frozen reduces projected end-to-end training time relative to Alt. by 17.6% on Coffee and 17.9% on Stove, while Multi-head Ensemble reduces it by 13.0% and 14.2%, respectively. Importantly, the end-to-end learning curves in Fig. 8(c,d) show that the actor-update gains remain meaningful even after accounting for the upfront cost of critic training and cache construction.

On Coffee, Multi-head Ensemble reaches 48.16% success after 16.42 hours, already exceeding the final Alt. mean of 47.12% obtained after 20.10 hours. This corresponds to an 18.3% reduction in projected time to reach the final alternating performance. On Stove, using the final Alt. mean of 34.32% as a common target, Alt. first exceeds this level after 64.81 hours, whereas Frozen and Multi-head Ensemble reach it after 49.93 and 48.14 hours, corresponding to reductions of 23.0% and 25.7%, respectively. Moreover, both frozen-critic methods reach above 40% success at approximately 56 hours, whereas Alt. does not reach 40% within the evaluated 77.77-hour budget.

These results show that, in our implementation, critic-first training does not merely shift computation from actor training to an expensive pretraining stage. Despite its upfront critic and caching cost, faster subsequent policy learning can recover this cost and reduce the projected end-to-end time required to reach strong policy performance, particularly in the larger language-conditioned Stove setting.

D.3 Reuse across Policy Configurations

Because we cache raw advantages rather than the final AWR weights, the same pretrained critic and advantage cache can be reused when changing the extraction temperature β. For each β, we recompute the AWR weights from the cached advantages and train a separate actor, while critic training and advantage computation are performed only once. For a sweep over five values of β, the projected cost decreases from 100.48 to 43.09 hours on Coffee and from 388.87 to 165.96 hours on Stove, corresponding to reductions of 57.1% and 57.3%, respectively. Thus, reusing the critic and advantage cache reduces the cost of policy-side hyperparameter sweeps by more than half in both settings, providing a substantial practical benefit when training large VLA policies.

D.4 Shared-Backbone Ensemble Overhead

The Multi-head critic shares the visual-language feature encoder across all five heads, while each head retains separate value, double-Q, and target double-Q networks. Shared features are computed

only once per minibatch, and the head-specific losses are averaged during critic training. During actor extraction, cached head-specific advantages are used directly, so selecting different heads does not require additional critic inference.

Increasing the number of critic heads from one to five increases the total parameter count from 1.621B to 1.643B, a 1.37% increase, while the number of trainable parameters increases from 72.51M to 85.30M, a 17.65% increase. Despite this increase in trainable critic parameters, the additional heads introduce only a modest increase in critic-training and end-to-end runtime in our implementation. Thus, the shared-backbone design provides multiple extraction signals without replicating the full VLA backbone and with limited additional computational overhead.

E Frozen-Critic Extraction Beyond Gaussian AWR Policies

We examine whether frozen-critic degradation is specific to the Gaussian AWR setting used in our main experiments. We consider two extensions while retaining IQL value learning: DDPG+BC policy extraction and a more expressive flow-based policy with AWR extraction. In the tables below, Alt. denotes alternating updates, Frozen denotes single-critic Freeze-then-Extract, IE denotes Independent Ensemble, and Ckpt. denotes Checkpoint Ensemble.

E.1 DDPG+BC Policy Extraction

We retain IQL value learning (Kostrikov et al., 2022) and replace AWR with DDPG+BC extraction (Fujimoto & Gu, 2021; Park et al., 2024) to examine whether freezing-induced degradation persists without AWR-specific sample weighting. Table 21 compares alternating updates, single-critic Freeze-then-Extract, Independent Ensemble, and Checkpoint Ensemble with the Tail-1k schedule. Benchmark means assign equal weight to each task.

Table 21: Mean normalized scores with IQL critics and DDPG+BC policy extraction. "IE" denotes Independent Ensemble, and "Ckpt." denotes Checkpoint Ensemble with the Tail-1k schedule.

Benchmark	Tasks	Alt.	Frozen	IE	Ckpt.
D4RL	17	62.35	61.30	62.51	62.04
OGBench	8	57.38	46.98	49.65	50.65
Overall	25	60.76	56.72	58.39	58.40

With DDPG+BC, frozen extraction degrades performance on both D4RL and OGBench, although the drop is generally milder than with AWR. This is consistent with AWR-specific sample overconcentration amplifying the sensitivity to a single frozen critic. Ensembling also improves DDPG+BC extraction: on D4RL, both Independent Ensemble and Checkpoint Ensemble recover the small degradation from Frozen, while on OGBench they partially recover the larger performance drop. These results suggest that frozen-critic degradation is not specific to AWR-style extraction, while AWR-specific weighting may increase its severity.

E.2 Flow-Based Policies with AWR Extraction

To examine whether frozen-critic degradation persists beyond a Gaussian MLP actor, we use FAWAC (Park et al., 2025c), a more expressive flow-based policy, while retaining IQL critics and AWR extraction. Table 22 compares alternating updates, single-critic Freeze-then-Extract, and frozen extraction with Independent Ensemble.

Frozen underperforms alternating updates on three of the four tasks, while Independent Ensemble improves over Frozen on all four. The ensemble mean of 49.9 nearly matches the alternating mean of 50.2. Thus, both the degradation and the benefit of ensembling persist with a more expressive actor. Since AWR remains the extraction objective, this comparison does not isolate AWR-specific effects, but complements the alternative-extraction experiments above.

Table 22: Flow-based policy extraction using IQL critics and AWR. The mean row averages the four task means.

Task	Alt.	Frozen	Frozen + IE
antmaze-large-diverse-v2	41.2 ± 2.5	23.6 ± 6.9	36.8 ± 3.3
antmaze-large-play-v2	47.2 ± 1.5	32.8 ± 8.6	38.6 ± 2.2
cube-single-play	88.6 ± 6.1	61.8 ± 11.2	75.6 ± 6.7
kitchen-partial-v0	23.7 ± 6.7	34.4 ± 12.6	48.5 ± 3.0
Mean	50.2	38.2	49.9

F Ablations and Ensemble Controls

We examine the effects of ensemble size and AWR inverse temperature, followed by additional comparisons with ensembling under alternating updates and extraction from a target critic. All experiments in this section use IQL. In the tables, Alt. denotes alternating updates, Frozen denotes single-critic Freeze-then-Extract, and IE denotes Independent Ensemble. For readability, OGBench task names omit the `-singletask-task1-v0` suffix.

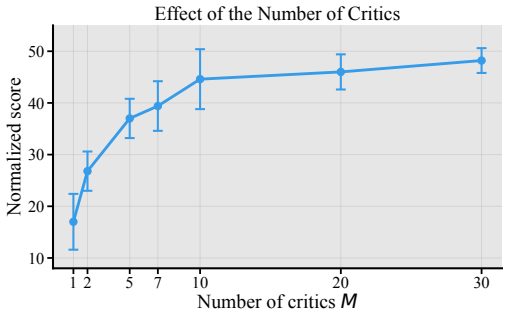

Figure 13: Effect of the number of critics in Independent Ensemble on IQL `antmaze-large-diverse-v2`. Values are mean ± SEM.

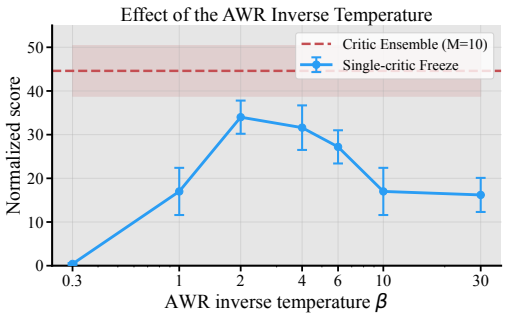

Figure 14: Effect of the AWR inverse temperature β for single-critic Freeze-then-Extract on IQL `antmaze-large-diverse-v2`. The dashed line shows Independent Ensemble with $M = 10$.

F.1 Effect of the Number of Critics in Independent Ensemble

We study how the number of frozen critics affects the performance of Independent Ensemble. In the main offline RL benchmark comparison, we use $M = 10$ critics. For this ablation, we use IQL on `antmaze-large-diverse-v2` and vary the ensemble size as $M \in \{1, 2, 5, 7, 10, 20, 30\}$. All other settings, including the dataset, critic-learning objective, and policy-extraction procedure, are kept fixed.

Fig. 13 shows that increasing the number of critics improves performance in this setting. Even a small ensemble with $M = 2$ improves over single-critic Freeze-then-Extract, increasing the score from 17.0 ± 5.4 to 26.8 ± 3.8. Performance continues to improve as the ensemble grows, reaching 44.6 ± 5.8 at $M = 10$, 46.0 ± 3.4 at $M = 20$, and 48.2 ± 2.4 at $M = 30$. The standard errors at $M = 20$ and $M = 30$ are also smaller than at $M = 10$. However, the largest gains occur when moving from a single critic to a moderate-size ensemble, while improvements become more gradual beyond $M = 10$. We therefore use $M = 10$ in the main benchmark comparison as a balance between performance, stability, and critic-training cost.

F.2 Effect of the AWR Inverse Temperature

We examine whether the failure of single-critic Freeze-then-Extract is related to excessive concentration of AWR weights. In AWR-style extraction, the exponential weight $w(s, a) = \exp(\beta A(s, a))$

amplifies advantage differences as the inverse temperature β increases. Reducing β can therefore alleviate sample over-concentration without changing the frozen critic.

For this ablation, we use IQL on `antmaze-large-diverse-v2` and vary β during policy extraction while keeping the frozen critic fixed. Fig. 14 shows that reducing β from the default value of 10 improves the score from 17.0 ± 5.4 to 34.0 ± 3.8 at $\beta = 2$. This result is consistent with sample over-concentration contributing to the failure of single-critic extraction.

However, lowering β alone does not close the gap to Independent Ensemble, which achieves 44.6 ± 5.8 on the same task. Temperature tuning can partially alleviate sample over-concentration, but it does not change the underlying critic or remove critic-specific advantage errors. Moreover, when β is too small, as in $\beta = 0.3$, performance deteriorates substantially. These results motivate using multiple critic judgments rather than only rescaling a single critic's advantages.

F.3 Stochastic Critic Selection versus Weight Averaging

We further examine how multiple frozen critics should be combined during policy extraction. Our default Independent Ensemble (IE) samples one critic member uniformly at each actor update and computes AWR weights from that member's advantages, as described in Eq. 3. As an alternative, we consider ***IE-Mean***, which computes the clipped AWR weight for each ensemble member and then averages these weights across members. Specifically, for each member m,

$$w_m(s,a) = \min\{\exp(\beta A_m(s,a)), w_{\max}\}, \qquad \bar{w}(s,a) = \frac{1}{M}\sum_{m=1}^{M} w_m(s,a). \tag{4}$$

IE-Mean uses $\bar{w}(s,a)$ for policy extraction. Both variants use the same frozen critic ensemble and policy-training settings, differing only in whether actor updates use a stochastically selected member-specific weight or the average weight across all members.

Table 23: Comparison of stochastic critic selection (IE) and averaging of member-specific AWR weights (IE-Mean) with IQL. Values are benchmark-level means over 17 D4RL tasks and 8 OGBench tasks. The overall column averages all 25 task-level means.

Method	D4RL (17 tasks)	OGBench (8 tasks)	Overall (25 tasks)
Alt.	64.71	50.33	60.11
Frozen	58.92	35.75	51.51
IE	65.22	48.60	59.90
IE-Mean	65.20	40.20	57.20

Table 23 shows that stochastic critic selection and weight averaging perform almost identically on D4RL, with mean scores of 65.22 and 65.20, respectively. On OGBench, however, stochastic selection performs substantially better, achieving 48.60 compared with 40.20 for IE-Mean. Across all 25 tasks, the corresponding means are 59.90 and 57.20.

This result suggests that the way ensemble members are combined can affect policy extraction performance. Different critics often assign high weight to different samples, as shown by the diversity of high-weight sample sets in Appendix A.4. Under uniform member sampling, stochastic critic selection and weight averaging yield the same expected actor gradient, since $\bar{w}(s,a) = \mathbb{E}_m[w_m(s,a)]$. However, with a finite number of actor updates, their optimization dynamics can differ: stochastic selection exposes each update to a member-specific weighting pattern, whereas IE-Mean uses the same averaged weights across updates. The results suggest that this member-specific variation across finite actor updates may contribute to the benefit of critic ensembling.

F.4 Ensembling under Alternating Updates

We apply Independent Ensemble to alternating IQL to distinguish its general benefits from its role in frozen-critic extraction. We use the same stochastic critic-selection mechanism, but continue updating the critics during policy learning. Table 24 compares single-critic and ensemble variants under both update protocols.

Table 24: Independent ensembling under alternating and frozen-critic training. Single denotes a single critic and IE denotes Independent Ensemble. The mean row averages the five task means.

	Alternating		Frozen	
Task	Single	IE	Single	IE
antmaze-large-play-v2	41.2 ± 2.1	49.0 ± 2.1	23.6 ± 5.4	49.8 ± 4.7
antmaze-large-diverse-v2	44.8 ± 2.3	45.6 ± 2.9	17.0 ± 5.4	44.6 ± 5.8
antmaze-large-navigate	46.6 ± 6.5	54.2 ± 4.6	15.8 ± 3.0	42.6 ± 5.0
cube-single-play	78.4 ± 4.7	81.2 ± 3.6	58.0 ± 6.0	85.6 ± 2.4
pointmaze-medium-navigate	88.0 ± 5.5	100.0 ± 0.0	60.0 ± 24.5	69.0 ± 18.5
Mean	59.8	66.0	34.9	58.3

Ensembling improves both protocols, but its average gain is larger for Frozen ($+23.4$) than for Alternating ($+6.2$), with a larger Frozen gain on four of the five tasks. This supports the usefulness of critic diversity when extraction would otherwise rely on a single fixed critic. However, frozen ensembling does not eliminate the average gap to alternating ensembling.

F.5 Policy Extraction from a Target Critic

We examine whether temporal smoothing of a single critic is sufficient to stabilize policy extraction by extracting policies from the slowly updated target critic. Table 25 compares this baseline with ordinary single-critic extraction, Independent Ensemble, and alternating IQL on six D4RL AntMaze tasks.

Table 25: Policy extraction from a target critic. Frozen uses the ordinary critic, while Target uses the slowly updated target critic. The mean row averages the six task means.

Task	Alt.	Frozen	Target	Frozen + IE
antmaze-umaze-v2	87.4 ± 1.3	84.0 ± 4.2	83.8 ± 2.7	84.6 ± 0.6
antmaze-umaze-diverse-v2	68.0 ± 3.1	65.4 ± 3.7	60.8 ± 3.0	73.6 ± 2.6
antmaze-medium-play-v2	71.6 ± 1.0	61.6 ± 13.6	63.0 ± 3.5	74.4 ± 1.4
antmaze-medium-diverse-v2	75.2 ± 2.2	54.6 ± 4.7	56.6 ± 4.4	70.4 ± 2.7
antmaze-large-play-v2	41.2 ± 2.1	23.6 ± 5.4	20.6 ± 6.1	49.8 ± 4.7
antmaze-large-diverse-v2	44.8 ± 2.3	17.0 ± 5.4	26.2 ± 4.9	44.6 ± 5.8
Mean	64.7	51.0	51.8	66.2

The target critic achieves a mean score of 51.8, close to Frozen at 51.0 and below Independent Ensemble at 66.2. Although the target critic smooths short-term changes in the critic during training, policy extraction still relies on a single fixed critic and therefore a single fixed AWR weighting signal. It does not directly prevent the actor from repeatedly emphasizing the same critic-specific subset of samples. This helps explain why using the target critic alone does not recover the benefit of exposing the actor to multiple critic signals.

G Experimental Details and Hyperparameters

This appendix provides implementation details that are omitted from the main text. We focus on experimental choices that affect reproducibility: benchmark suites, evaluation protocol, update budgets, critic checkpointing, hyperparameters, compute resources, and code availability. Unless otherwise stated, the benchmark settings below refer to the main single-task D4RL and OGBench comparison; goal-conditioned reuse and VLA experiments use separate configurations.

G.1 Benchmarks and Evaluation Protocol

We evaluate the main single-task results on D4RL and OGBench. The D4RL suite includes AntMaze, locomotion, Kitchen, and Adroit tasks. The OGBench suite includes single-task navigation and

manipulation environments. These benchmarks cover both settings where behavior cloning is already strong, such as several locomotion and Kitchen tasks, and settings where value-guided stitching is more important, such as AntMaze and OGBench navigation tasks.

For the standard offline RL benchmark tables, we report normalized scores. Unless otherwise stated, each number is averaged over five random seeds and reported as mean $\pm$ standard error of the mean (SEM). The evaluation protocol depends on the benchmark family: D4RL AntMaze tasks are evaluated every $50{,}000$ actor updates using 100 episodes, D4RL locomotion tasks every $10{,}000$ actor updates using 20 episodes, D4RL Kitchen and Adroit tasks every $20{,}000$ actor updates using 100 episodes, and OGBench single-task environments every $50{,}000$ actor updates using 100 episodes. For each method and task, we report the final actor evaluation rather than the best intermediate actor checkpoint. The value-learning checkpoint choices are described in Appendix G.2.

G.2 Update Budgets and Checkpointing

For standard alternating training, actor and value-learning updates are interleaved for 10^6 iterations. For all Freeze-then-Extract variants, value learning is performed for 10^6 iterations before freezing the value functions and training the actor for 10^6 updates. Single-critic Freeze and Checkpoint Ensemble use one value-learning trajectory, whereas Independent Ensemble uses $M = 10$ independently trained trajectories.

Single-critic Freeze and Independent Ensemble use the final value checkpoints. For Checkpoint Ensemble, we use a fixed checkpoint schedule in the main comparison and additionally report a tuned variant that selects the best-performing schedule for each algorithm–task pair. The checkpoint schedules and sensitivity results are detailed in Appendix C.

G.3 Models and Hyperparameters

This subsection summarizes the model configurations and hyperparameters used in the main single-task benchmark experiments. We organize them into common settings, task-dependent settings, and algorithm-specific settings. Unless otherwise stated, we follow the hyperparameter choices used in prior work (Kostrikov et al., 2022; Garg et al., 2023; Xu et al., 2023; Park et al., 2025a) for each base algorithm and benchmark, and use the same settings across alternating updates, single-critic Freeze-then-Extract, Checkpoint Ensemble, and Independent Ensemble.

G.3.1 Common Settings

Table 26: Hyperparameters shared across methods in the main single-task benchmark comparison, unless otherwise stated.

Hyperparameter	Value
Random seeds	5
Batch size	256
Optimizer	Adam (Kingma & Ba, 2014)
Discount factor γ	0.99 by default; 0.995 for IQL/SQL on `antmaze-giant` and `humanoidmaze-medium`
Target update rate τ	0.005
Ensemble size M	10 for Checkpoint Ensemble and Independent Ensemble

For IQL and XQL, the actor, Q-function, and value function use an initial learning rate of 3×10^{-4}. The actor learning rate follows cosine decay over the actor-update budget, whereas the Q and V learning rates remain constant. For SQL, the task-dependent learning rate in Table 29 applies to all three networks, with the same scheduling convention.

For IQL and XQL, we compute $A(s,a) = \min\{Q_1(s,a), Q_2(s,a)\} - V(s)$ and use $w(s,a) = \min\{\exp(\beta A(s,a)), 100\}$ for policy extraction. SQL instead uses clipped positive-advantage weighting, $w(s,a) = \min\{\max(A(s,a), 0), 100\}$, without an inverse temperature. In all cases, the policy

objective is the minibatch mean of the weighted negative log likelihood, without additional normalization of the weights.

G.3.2 Task-Dependent Hyperparameters

Table 27 summarizes task-dependent network sizes and policy-extraction temperatures. The inverse temperature β is used only for IQL and XQL; SQL uses the positive-advantage weighting described above.

Table 27: Task-dependent network sizes and IQL/XQL policy-extraction temperatures.

Task group or task	Hidden dimensions	IQL/XQL inverse temperature β
D4RL AntMaze	$[256, 256]$	10.0
D4RL locomotion	$[256, 256]$	3.0
D4RL Kitchen	$[256, 256]$	0.5
D4RL Adroit	$[256, 256]$	0.5
OGBench navigation and maze tasks	$[256, 256, 256, 256]$	10.0
`cube-single-play-singletask-task1-v0`	$[256, 256, 256, 256]$	1.0
`cube-double-play-singletask-task1-v0`	$[256, 256, 256, 256]$	0.3

G.3.3 Algorithm-Specific Hyperparameters

Tables 28, 29, and 30 summarize the hyperparameters used for IQL, SQL, and XQL, respectively. The XQL loss parameter β_{XQL} is distinct from the policy-extraction inverse temperature β in Table 27.

Table 28: IQL-specific hyperparameters.

Task group	Expectile
D4RL AntMaze	0.9
D4RL locomotion	0.7
D4RL Kitchen	0.7
D4RL Adroit	0.7
OGBench single-task	0.9

Table 29: SQL-specific hyperparameters.

Task group or task	α_{SQL}	Learning rate
`antmaze-umaze-v2`	0.1	2×10^{-4}
`antmaze-umaze-diverse-v2`	3.0	2×10^{-4}
D4RL AntMaze, other tasks	0.1	2×10^{-4}
D4RL locomotion	1.0	3×10^{-4}
D4RL Kitchen	0.5	3×10^{-4}
D4RL Adroit	0.5	3×10^{-4}
OGBench single-task	0.1	2×10^{-4}

Table 30: XQL-specific hyperparameters.

Task	XQL loss β_{XQL}	XQL loss clip	Value updates
antmaze-umaze-v2	1.0	7.0	1
antmaze-umaze-diverse-v2	0.8	7.0	1
antmaze-medium-play-v2	0.9	7.0	1
antmaze-medium-diverse-v2	0.6	7.0	4
antmaze-large-play-v2	0.6	5.0	1
antmaze-large-diverse-v2	0.6	5.0	1
halfcheetah-medium-v2	1.0	7.0	1
hopper-medium-v2	5.0	7.0	1
walker2d-medium-v2	10.0	7.0	1
halfcheetah-medium-replay-v2	1.0	5.0	1
hopper-medium-replay-v2	2.0	7.0	1
walker2d-medium-replay-v2	5.0	5.0	1
kitchen-complete-v0	2.0	7.0	1
kitchen-partial-v0	8.0	7.0	1
kitchen-mixed-v0	5.0	7.0	1
pen-human-v1	5.0	7.0	1
pen-cloned-v1	0.8	5.0	2
antmaze-large-navigate-singletask-task1-v0	0.6	7.0	1
antmaze-giant-navigate-singletask-task1-v0	0.6	7.0	1
humanoidmaze-medium-navigate-singletask-task1-v0	0.6	7.0	1
cube-single-play-singletask-task1-v0	5.0	7.0	1
cube-double-play-singletask-task1-v0	5.0	7.0	1
pointmaze-medium-navigate-singletask-task1-v0	0.6	7.0	1
pointmaze-medium-stitch-singletask-task1-v0	0.6	7.0	1
pointmaze-large-navigate-singletask-task1-v0	0.6	7.0	1

G.4 VLA Experimental Details

We compare alternating updates, single-critic Freeze-then-Extract, and ensemble-based Freeze-then-Extract with VLA policies in RoboCasa-based simulation tasks (Nasiriany et al., 2024). We use DEAS (Kim et al., 2026) for value learning and a lightweight Multi-head Ensemble for ensemble-based extraction.

Tasks and datasets. We collect our own offline datasets in RoboCasa-based simulation environments. Following DEAS (Kim et al., 2026), we first pre-finetune GR00T N1.5 on RoboCasa expert demonstrations and use the resulting policy to collect offline rollouts. We consider `CoffeeSetupMug` with 1,500 offline trajectories and the language-conditioned `TurnOffStove` task with 9,000 trajectories and six language goals. Both datasets contain successful and failed trajectories. Coffee contains 806,757 transitions and 270 successful trajectories (18.0%), while Stove contains 3,723,293 transitions and 2,772 successful trajectories (30.8%). We fix the environment layout and style in both tasks. In Coffee, the mug model is also fixed. Both tasks use three RGB views (left, right, and wrist) at 256×256 resolution and 20 Hz, together with a 53-dimensional proprioceptive state and language instruction. The policy outputs 12-dimensional continuous actions using 16-step action chunks. Table 31 summarizes the main dataset and training settings.

Table 31: VLA dataset and training settings.

Setting	CoffeeSetupMug	TurnOffStove
Trajectories	1,500	9,000
Transitions	806,757	3,723,293
Successful trajectories	270 (18.0%)	2,772 (30.8%)
Maximum trajectory length	600	500
Language goals	1	6
Actor/Critic updates	30k	60k
Global batch size	128	256
Approx. data epochs	4.76	4.13
AWR β	3	10
Weight clip	5	5
Base policy success rate	17.6%	28.4%

Policy and critic architectures. DEAS extends IQL to action sequences using distributional critics. The critic uses a 16-step action horizon, double Q-functions with $\min(Q_1, Q_2)$ aggregation, expectile 0.7, within-chunk discount 0.9, bootstrap discount 0.99, and target-network update rate 0.005. Q and V use four-layer MLPs with hidden sizes 512 and 256, respectively. The distributional critic uses 101 atoms over $[-100, 0]$ with HL-Gauss $\sigma = 0.75$. The critic learning rate is 3×10^{-4} with no decay and gradient clipping at 5.0. The GR00T vision and language backbones are frozen. For critic learning, successful terminal transitions receive reward 0, while all other transitions, including failure terminals, receive reward -1. Our Multi-head Ensemble uses five critic heads that share the visual-language features. Each head has independent Q_1, Q_2, V, and target Q networks, and all heads are trained on the same minibatch with their losses averaged. Relative to the single-head critic, this increases the total parameter count by 1.37% and the number of trainable parameters by 17.65%.

Training and evaluation. We initialize the actor from the same GR00T N1.5 checkpoint pre-finetuned on RoboCasa expert demonstrations that is used for offline data collection. Alternating training interleaves critic and policy updates, while the frozen-critic variants train the critic first and then extract the policy with the critic fixed. Coffee uses 30k critic updates and 30k actor updates with global batch size 128, while Stove uses 60k updates for each with global batch size 256.

For policy extraction, we apply advantage weighting to the per-sample flow-matching loss. For an action chunk $\mathbf{a}$, we compute

$$A(s, \mathbf{a}) = \min\{Q_1(s, \mathbf{a}), Q_2(s, \mathbf{a})\} - V(s), \qquad w(s, \mathbf{a}) = \min\{5, \exp(\beta A(s, \mathbf{a}))\},$$

and optimize

$$\mathcal{L}_\pi = \mathbb{E}_{(s,\mathbf{a})\sim\mathcal{D}}\left[w(s, \mathbf{a})\, \mathcal{L}_{\mathrm{FM}}(s, \mathbf{a})\right],$$

where $\mathcal{L}_{\mathrm{FM}}$ denotes the standard flow-matching loss of the VLA policy. We use $\beta = 3$ for Coffee and $\beta = 10$ for Stove, with no advantage normalization. The actor is trained with learning rate 10^{-4} using cosine decay and 5% warmup. The language and vision backbones are frozen, while the projector and flow-matching action head are updated. For the five-head critic, the five heads are used in randomized order during policy extraction.

At inference time, the policy predicts a 16-step action chunk and executes all 16 actions open-loop before replanning from the latest observation. We run each method with five training seeds. Each checkpoint is evaluated over 250 episodes per run, and we report the mean success rate across the five runs.

G.5 Goal-Conditioned Critic Reuse

We evaluate goal-conditioned GCIQL (Park et al., 2023; 2025a) critic reuse on `cube-single` and `antmaze-large-navigate`, comparing single-critic Freeze, Checkpoint Ensemble, and Independent Ensemble. All pretrained value functions remain frozen during downstream actor learning.

Goal representation and construction. Goals are physical cube positions (x, y, z) for Cube and Ant positions (x, y) for AntMaze, and these coordinates are also supplied to the networks. We use a goal-region width scale of 0.5, halving each horizontal coordinate range about its midpoint. Cube's original XY workspace is $[0.30, 0.55] \times [-0.30, 0.30]$, yielding training bounds $[0.3625, 0.4875] \times [-0.15, 0.15]$. The tabletop height is $z = 0.02$ with tolerance ± 0.015; this height band is not scaled. AntMaze's original bounds are the coordinate-wise minima and maxima of dataset XY positions. Training goals are selected from dataset candidates within the scaled region, rather than uniformly from the entire rectangle.

Each run uses eight held-out goals. Before critic pretraining, we remove all training-goal candidates within Euclidean distance 0.06 of the held-out goals for Cube and 1.0 for AntMaze. Thus, the held-out goals and their surrounding neighborhoods are never used for critic pretraining. Cube held-out goals are sampled uniformly in the original XY workspace at tabletop height, while AntMaze goals alternate between valid cells near and far from the maze center. Both are mapped to nearby valid dataset candidates. Held-out goals may lie inside or outside the scaled region.

Critic pretraining and ensemble construction. Table 32 gives the goal-sampling mixture. Current and future branches sample only transitions with eligible current or future goals in the training region. Future goals follow a geometric distribution with parameter $1 - \gamma = 0.01$, renormalized over eligible candidates and including terminal tail probability when applicable. Random goals are sampled uniformly from the training-goal pool. The offline dataset is retained, but the goal restriction also changes the transition-sampling distribution.

Table 32: Goal-sampling probabilities for GCIQL critic pretraining. All sampled goals satisfy the training-goal restriction.

Update	Current	Uniform future	Geometric future	Random
Value update	0.2	0.0	0.5	0.3

Actor, Q, and V networks use three 512-unit hidden layers with GELU and no dropout. LayerNorm is applied to Q and V only. All networks use Adam with constant learning rate 3×10^{-4} and batch size 1,024. Critic learning uses discount 0.99, target-network update rate 0.005, and expectile 0.9.

Freeze uses the final critic after 10^6 updates. Checkpoint Ensemble retains six matched (Q_1, Q_2, V) members at 750k, 800k, 850k, 900k, 950k, and 1M updates from one trajectory. Independent Ensemble uses six independently initialized trajectories of 1M updates each.

Actor extraction and evaluation. We initialize a separate goal-conditioned Gaussian actor for each target goal and train it for 10^6 updates with the goal and critics fixed. Extraction uses AWR inverse temperature 10, weight cap 100, and no weight normalization. At each ensemble-based actor update, one member is sampled uniformly and its matched Q and V estimates are used for the entire minibatch before computing the extraction weights.

The protocol uses five runs, each with eight held-out goals. Methods share the same goal definitions within each run. Cube held-out goals vary across runs, whereas AntMaze uses the same eight held-out goals across runs. The main results in Fig. 9 report performance on these held-out goals. Each goal is evaluated over 20 episodes every 100k actor updates, and we use the final post-training evaluation for the reported scores.

G.6 Compute Resources

The standard D4RL and OGBench benchmark experiments were run on single-GPU machines using NVIDIA A100 or V100 GPUs. Each benchmark run takes no more than three days on a single GPU. The VLA experiments were conducted using NVIDIA GB200 GPUs.

G.7 Code and Data Availability

The standard offline RL benchmark experiments use publicly available datasets from D4RL (Fu et al., 2020) and OGBench (Park et al., 2025a), accessed through the standard benchmark APIs. The structured-goal reuse experiments restrict goal sampling during critic pretraining rather than replacing the underlying offline data. The RoboCasa-based VLA datasets are described in Appendix G.4.

At the time of submission, the code is not publicly released. We plan to clean up and release the code upon publication, including training scripts, configuration files, and hyperparameters needed to reproduce the main experiments. To support reproducibility before the code release, we provide the benchmark list, training protocol, evaluation protocol, hyperparameters, and full per-task results in the appendix.