The Three Regimes of Offline-to-Online Reinforcement Learning

Lu Li[1,2] **Tianwei Ni** [1,2] **Yihao Sun**[1,2] **Pierre-Luc Bacon**[1,2,3]

[1]Mila – Québec AI Institute [2]Université de Montréal [3]CIFAR AI Chair

lu.li@mila.quebec, twni2016@gmail.com

Abstract

Offline-to-online reinforcement learning (RL) has emerged as a practical paradigm that leverages offline datasets for pretraining and online interactions for fine-tuning. However, its empirical behavior is highly inconsistent: design choices of online fine-tuning that work well in one setting can fail completely in another. Guided by the stability–plasticity principle, we propose a framework that can explain this inconsistency: We argue that efficient fine-tuning must preserve the utility of the stronger offline prior, whether that is the pretrained policy or the offline dataset, while maintaining sufficient plasticity. This perspective identifies three regimes of online fine-tuning, each requiring distinct stability properties. We validate this framework through a large-scale empirical study, finding that the results strongly align with its predictions in 45 out of 63 cases, with only 3 opposite mismatches. This work provides a framework for guiding design choices in offline-to-online RL based on the relative performance of the offline dataset and the pretrained policy.

1 Introduction

Reinforcement learning (RL) has achieved impressive successes in a variety of domains [1, 2, 3], but its reliance on large amounts of online interaction often makes direct application to real-world problems challenging. To address this challenge, recent research has turned to leveraging pre-collected datasets through offline RL or imitation learning [4, 5], providing a strong initial policy trained from offline data. However, policies trained purely offline are often suboptimal and fail to generalize to states outside the dataset's support, making online fine-tuning essential. Offline-to-online RL [6, 7] addresses this issue by first pretraining an agent on an offline dataset and then fine-tuning it with additional online interactions to further improve performance.

While offline-to-online RL has led to promising results, online RL fine-tuning suffers from highly inconsistent empirical behavior: design choices that work well in one setting can fail completely in another. For example, as shown in Figure 1, on D4RL tasks [8] such as *antmaze-large-play-v2*, Warm-Start RL (WSRL) [9], which relies on the pretrained policy and discards the offline dataset during online fine-tuning, substantially outperforms RL with Prior Data (RLPD) [10], which uses the offline dataset only at the online RL stage. In contrast, on D4RL tasks such as *relocate-binary-v0*, the opposite pattern emerges, with RLPD outperforming WSRL by a wide margin. These seemingly inconsistent outcomes

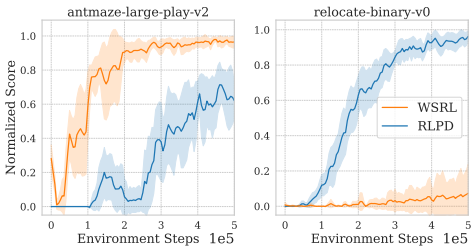

Figure 1: Comparison between **WSRL** (pretrained policy only) and **RLPD** (offline dataset only) on two representative offline-to-online RL tasks. Learning curves are shown as mean ± 95% CI.

Accepted at ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning.

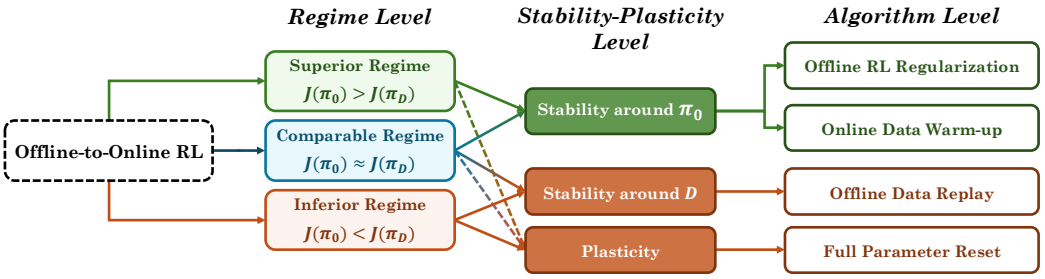

Figure 2: **Overview of the three regimes in offline-to-online RL**, defined based on the *relative* performance of the pretrained policy $J(\pi_0)$ and the offline dataset $J(\pi_\mathcal{D})$. For each regime, our framework indicates which property is most needed during fine-tuning. The boxes at the right show representative design choices that implement these enhancing stability or plasticity. Dashed arrows denote weaker connections than solid arrows.

raise one fundamental question: *What underlying factors cause design choices to succeed in some settings but fail in others?*

To answer this question, we propose a framework for offline-to-online RL that explains these seemingly inconsistent outcomes through the lens of the stability–plasticity principle, a perspective that has been widely studied in neuroscience [11] and machine learning [12, 13, 14]. Guided by the principle, effective fine-tuning requires a careful balance between stability and plasticity. **Stability** refers to the preservation of useful prior knowledge, ensuring that competencies acquired during pretraining are not substantially degraded. **Plasticity**, in contrast, denotes the capacity of the model to adapt flexibly and efficiently to new data. Furthermore, we identify two distinct forms of stability in offline-to-online RL: stability around the *pretrained policy* π_0, which emphasizes preserving knowledge explicitly encoded in the policy parameters, and stability around the *offline dataset* $\mathcal{D}$, which emphasizes retaining knowledge implicitly encoded in offline data. As stability and plasticity are inherently in trade-off, this distinction indicates that fine-tuning is more efficient when stability is enhanced with respect to the stronger source of offline priors, whether it is the pretrained policy or the offline dataset.

Building on this insight, we propose a taxonomy of **three regimes for offline-to-online RL**, each capturing a distinct relationship between the pretrained policy and the offline dataset. As shown in Figure 2, the three regimes are defined based on which source of prior knowledge is stronger, either the pretrained policy or the offline dataset. Moreover, different fine-tuning methods can be systematically categorized according to whether they enhance stability around π_0, stability around $\mathcal{D}$, or plasticity. By first determining the regime, one can select or design fine-tuning strategies that match its stability-plasticity requirements. This yields two *practical* benefits: it helps choose the most suitable method for each setting rather than applying a single uniform state-of-the-art algorithm, and it narrows the search space by indicating whether one should focus on leveraging the pretrained policy or on exploiting the offline dataset, thereby reducing unnecessary trial-and-error.

To validate this framework, we conduct a large-scale empirical study that covers 21 dataset-task compositions across four D4RL domains (MuJoCo locomotion, AntMaze navigation, Adroit manipulation, and Kitchen manipulation) and three representative pretraining algorithms, yielding 63 settings. The results align closely with the predictions of our framework, supporting its utility for guiding design choices of fine-tuning in offline-to-online RL.

Contributions of this paper can be summarized as:

- We develop a diagnostic and predictive framework for offline-to-online RL guided by the stability–plasticity principle. This yields a three-regime taxonomy defined by the relative performance of the initial pretrained policy π_0, and the offline dataset $\mathcal{D}$. We also cast prior disparate fine-tuning algorithms into a unified view based on whether they enhance stability or plasticity.
- To validate this framework, We conduct an extensive empirical study across 63 settings. The results align with our predictions in 45 cases, with only 3 opposite mismatches. To ground these behavioral outcomes, we provide a mechanistic analysis of value learning dynamics during fine-tuning, showing that failures stem from exploding offline TD error and divergent Q-values. Finally, we demonstrate that our raw-return-based taxonomy consistently outperforms alternative taxonomies based on complex, indirect metrics, establishing it as a robust and practical first-order criterion.

2 Preliminary: Offline-to-Online RL

Consider an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where the performance of a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ is measured by its expected discounted return: $J(\pi) = \mathbb{E}_{\pi,\mathcal{M}}[\sum_t \gamma^t r_t]$. Offline-to-online RL begins by pretraining the agent on an offline dataset $\mathcal{D}$, which is collected from $\mathcal{M}$ under an unknown behavior

policy (or mixture of policies), using an offline RL algorithm A_{off}. This yields an offline pretrained agent whose policy is given by $\pi_0 = A_{\text{off}}(\mathcal{D})$. The fine-tuning step consists of using an online algorithm A_{on} that starts from π_0 and interacts with $\mathcal{M}$ to obtain a final policy π_N after N iterations:

$$\pi_N = A_{\text{on}}\left(\mathcal{M}, \mathcal{D}, \pi_0\right). \tag{1}$$

Crucially, we analyze this transition from offline pretraining to online fine-tuning as a fundamental continual learning problem [13]. Even though the underlying MDP remains identical across both phases, the agent faces a severe distributional shift from a static, external offline dataset to actively collected online experiences. This shift forces the agent to balance acquiring new behaviors against the risk of catastrophically forgetting the valuable prior knowledge extracted during pretraining [13, 15].

Although the ideal objective of offline-to-online RL is to co-design $(A_{\text{off}}, A_{\text{on}})$ to maximize $J(\pi_N)$, in this work, we focus on understanding and improving the **online RL fine-tuning** component A_{on} by fixing the offline pretraining component A_{off}.

3 A Decomposition of Performance Based on Stability–Plasticity Principle

This section introduces an analytical framework for reasoning about fine-tuning in offline-to-online RL. Our goal is to characterize when and how online training leads to improvements over the offline initialization or degradations of what was already learned. We define two complementary properties of online fine-tuning: *stability*, the ability to preserve previously acquired performance, and *plasticity*, the capacity to improve further. These are grounded in the notion of a *performance level*, understood as the expected return encoded either in the dataset or in the pretrained policy. We show that performance admits a decomposition into three terms — offline prior, stability, and plasticity. This perspective provides both diagnostic insight and practical guidance.

3.1 Offline Priors for Online Fine-Tuning

We distinguish two offline priors available before online fine-tuning: the offline dataset and the pretrained policy, obtained by running an offline RL algorithm on the dataset. While the ultimate utility of these priors depends on complex factors such as state-action coverage, we abstract this complexity by evaluating both through their empirical performance. Relying on this single, first-order metric ensures *practical simplicity*, providing an objective and readily quantifiable basis to directly compare two fundamentally different sources. In Section 5.5, we compare this simple metric with alternative regime taxonomies, including dense reward proxies.

Performance of the dataset: $J(\pi_{\mathcal{D}})$. Let $\mathcal{D}$ be the offline dataset and $\pi_{\mathcal{D}}$ be an abstract behavior policy representing the data-generating process. While $\mathcal{D}$ may be collected from a mixture of policies, $\pi_{\mathcal{D}}$ serves as a convenient abstraction. Its performance can be estimated by the average return:

$$J(\pi_{\mathcal{D}}) \approx \frac{1}{L}\sum_{k=1}^{L}\sum_{t=1}^{T} r_{k,t},$$

where L is the number of trajectories in $\mathcal{D}$, and $r_{k,t}$ is the reward at time step t in the k-th trajectory. This measure provides a scalar summary of the return encoded in the dataset.

Performance of the pretrained policy: $J(\pi_0)$. The performance $J(\pi_0)$ reflects the inductive biases of the offline RL algorithm A_{off}, as well as the data quality and the underlying complexity of the MDP $\mathcal{M}$. This policy can provide a strong initialization for online fine-tuning, though it does not necessarily dominate the dataset baseline in practice. We therefore consider both $J(\pi_{\mathcal{D}})$ and $J(\pi_0)$ jointly as candidate sources of offline priors.

3.2 Performance Decomposition and Three Regimes

Online fine-tuning produces a sequence of policies $\{\pi_n\}_{n=0}^{N}$ with corresponding performances $\{J(\pi_n)\}_{n=0}^{N}$. Our goal is to understand how these trajectories of performance can be expressed in terms of the offline priors identified above and the two complementary properties of stability and plasticity. This leads to a decomposition of final performance that makes explicit what is preserved from offline training and what is gained during online interaction.

Stability with regard to a performance level. In continual learning, stability is conventionally evaluated through an agent's ability to avoid catastrophic degradation of previously acquired capabilities [12, 16]. Motivated by this established standard, We define the stability of an online RL training

process with respect to a performance level l, as the ability to retain the relative performance:

$$\text{Stability}(l) := \min\left(\min_{0\le n\le N} J(\pi_n) - l, 0\right). \tag{2}$$

This captures the worst-case performance drop during fine-tuning relative to l. A score of zero means no degradation; a negative score measures how much was lost.

In our setting, the appropriate reference level is the best performance available from the offline pretraining phase, either from the dataset or from the pretrained policy:

$$J^*_{\text{off}} := \max\left(J(\pi_0),\ J(\pi_{\mathcal{D}})\right). \tag{3}$$

We refer to this as the *offline performance baseline*, and the stability with respect to it is:

$$\text{Stability}(J^*_{\text{off}}) = \min_{0\le n\le N} J(\pi_n) - J^*_{\text{off}} \le 0. \tag{4}$$

Plasticity. In continual learning, plasticity is characterized by an agent's ability to continually adapt and improve its performance [17, 14]. Consistent with this established notion, we define the plasticity as the ability to improve during online fine-tuning:

$$\text{Plasticity} := \max_{0\le i\le N} J(\pi_i) - \min_{0\le j\le N} J(\pi_j) \ge 0. \tag{5}$$

We quantify it by the largest performance gain attained, namely the gap between the best and worst observed performance.

By relating these concepts through a **performance decomposition**, we have:

$$\underbrace{\max_{0\le n\le N} J(\pi_n)}_{\text{Best Performance}} = \underbrace{J^*_{\text{off}}}_{\text{(1) Prior}} + \underbrace{\text{Stability}(J^*_{\text{off}})}_{\text{(2) Degradation}\le 0} + \underbrace{\text{Plasticity}}_{\text{(3) Online Improvement}\ge 0}.$$

This equation states that the best performance an agent achieves is the outcome of three interacting components. (1) The first term is the baseline performance established by the offline phase, either through the offline dataset or the pretrained policy, whichever is stronger. (2) The second term measures stability, which records whether this baseline is preserved or degraded during fine-tuning; it is always non-positive since performance can at best be maintained but not exceeded by this term. (3) The third term captures plasticity, defined as the performance improvement resulting from online interaction, and is non-negative by definition. Therefore, the *improvement* over the prior is given by the sum of plasticity and stability.

The three regimes of offline-to-online RL. Given a pretrained policy π_0 and an offline dataset $\mathcal{D}$, the objective of online fine-tuning is to improve performance by balancing between stability with respect to $\max(J(\pi_0), J(\pi_{\mathcal{D}}))$ and sufficient plasticity. Based on this perspective, we identify three regimes for the online fine-tuning phase: **Superior**: where $J(\pi_0) > J(\pi_{\mathcal{D}})$; **Comparable**: where $J(\pi_0) \approx J(\pi_{\mathcal{D}})$; **Inferior**: where $J(\pi_0) < J(\pi_{\mathcal{D}})$. These regimes are intended to reflect substantial differences in $J(\pi_0)$ and $J(\pi_{\mathcal{D}})$, since small performance gaps may not be meaningful and therefore should not determine regime assignment.

This regime taxonomy provides a framework for reasoning about the stability–plasticity trade-off in offline-to-online RL. It clarifies which source of prior should anchor stability in a given setting, as shown in Figure 2. In the **Superior** Regime, stability relative to π_0 should be prioritized, because π_0 offers greater utility than $\mathcal{D}$. In the **Inferior** Regime, stability relative to $\mathcal{D}$ should be emphasized, as $\mathcal{D}$ demonstrates more usefulness than π_0. In the **Comparable** Regime, both baselines provide similar utility, so preserving either lead to similar effect. At the same time, maintaining sufficient plasticity across all regimes is essential for efficient online RL fine-tuning.

4 Design Choices in Stability and Plasticity

Building on the stability–plasticity principle and the regime taxonomy, we analyze concrete design choices for online RL fine-tuning. By categorizing methods according to whether they promote stability around the pretrained policy π_0, stability around the offline dataset $\mathcal{D}$, or increased plasticity, our framework organizes previously disparate practices into a structured landscape. Because many existing algorithms entangle these components, we isolate and analyze representative modules individually to clarify their distinct effects.

Minimal baseline. We begin with defining a naive online RL fine-tuning baseline that serves as the reference point for introducing additional components, which is *intentionally minimalist*. It applies

a standard online RL algorithm (e.g., SAC [18] or TD3 [19]) initialized with an offline-pretrained agent, without any further modifications. This baseline anchors the analysis and makes the marginal effect of each added component interpretable.

4.1 Stability Relative to the Offline Dataset $\mathcal{D}$

Design choices in this category promote stability by reusing the offline dataset during online fine-tuning. Incorporating $\mathcal{D}$ into the online learning process helps preserve the knowledge in the offline dataset and mitigates distribution shift between offline and online data.

A common strategy for incorporating offline data during fine-tuning is to reuse the offline dataset together with newly collected online transitions. One approach initializes the replay buffer with the entire offline dataset, after which new online experiences are appended as the agent interacts with the environment. In this case, the ratio of offline to online data is determined by the dataset size and gradually shifts toward online data as training progresses. An alternative approach maintains two separate replay buffers: one fixed buffer containing the offline dataset and another buffer for online experiences. During training, each batch is sampled from both buffers according to a specified offline data ratio α. For instance, CalQL [15] and RLPD [10] use $\alpha = 0.5$, corresponding to a symmetric 50% offline and 50% online sampling ratio.

4.2 Stability with Respect to the Pretrained Policy π_0

Design choices in this category focus on preserving and building upon the knowledge encoded in the pretrained policy π_0. The goal is to reduce the risk of catastrophic forgetting and ensure that fine-tuning does not erase useful behaviors learned during pretraining.

Online data warmup. Before applying gradient updates, the agent π_0 first collects a larger amount of online data (K steps) [9]. This strategy reduces the mismatch between the pretraining distribution and the online data distribution, lowering the chance that early updates overwrite prior knowledge.

Offline RL regularization. Fine-tuning can also reuse the same offline RL algorithm that produced the pretrained policy, thereby inheriting its conservative regularization. This regularization penalizes state-action pairs outside the online data. Since the online data is collected by the sequence of policies from π_0 to π_N, the regularization *implicitly* anchors learning around the region visited by π_0, even if π_0 is not stored during fine-tuning. When offline data $\mathcal{D}$ is also used with regularization, we consider it as promoting stability towards *both* π_0 and $\mathcal{D}$; since π_0 is derived from $\mathcal{D}$, this setting tends to have the strongest stability. Such regularization is widely adopted in prior work [6, 20, 21, 15].

4.3 Plasticity: Parameter Reset

A direct method to increase plasticity is to reset network parameters, as randomly initialized networks tend to exhibit higher plasticity than pretrained ones [17]. In the context of offline-to-online RL, parameter reset can be interpreted as starting from any pretrained agent π_0 and then resetting its weights to a randomly initialized agent π_1. While this approach severely degrades initial performance (i.e., $J(\pi_1) - J(\pi_0)$ is highly negative), it significantly enhance plasticity. RLPD [10] directly trains an online RL agent from random initialization, which can thus be reinterpreted as pretraining followed by full parameter reset before fine-tuning.

5 Empirical Study

We test the validity of our three-regime framework through a large-scale empirical study, examining whether its regime-specific predictions align with observed outcomes. To connect the design modules described above with this framework, we group algorithms by the primary source of stability they emphasize. Methods that preserve knowledge from the pretrained policy π_0 are labeled $\boldsymbol{\pi_0}$**-centric**, while those that anchor stability to the offline dataset $\mathcal{D}$ are labeled $\boldsymbol{\mathcal{D}}$**-centric**. Approaches that combine elements of both are called **mixed** $\boldsymbol{\pi_0 + \mathcal{D}}$ **methods**.

We study a diverse set of benchmark tasks and dataset compositions, following the experimental protocols of prior work [15, 9]. Specifically, we include MuJoCo locomotion, AntMaze navigation, Adroit manipulation, and Kitchen manipulation domains from D4RL [8], covering a total of 21 dataset-task compositions. All experiments are conducted with 10 random seeds to ensure statistical reliability.

Table 1: **Confusion matrix of fine-tuning results across the three regimes.** Green cells: correct predictions (45/63); red cells: opposite mismatches (3/63); gray cells: adjacent mismatches (15/63). Overall, the framework achieves **71%** correct predictions with only **5%** opposite mismatches.

		Pretraining Regime		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**24**	2	1
	π_0-centric $\approx$ $\mathcal{D}$-centric	6	**2**	3
	π_0-centric $<$ $\mathcal{D}$-centric	2	4	**19**

Offline pretraining phase. We employ two representative offline RL algorithms, CalQL [15] and ReBRAC [21], as well as behavior cloning (BC) [22] using a deterministic policy. To pair a behavior-cloned policy with a critic, we pretrain the critic by Fitted Q Evaluation (FQE) [23] after BC. Combining the 21 dataset-task compositions with these 3 pretraining algorithms yields 63 experimental settings in total. Each setting is defined by a specific combination of pretraining algorithm, dataset, and task. In this work, since we focus on cases where the offline dataset and the task are from the same MDP, the pretraining algorithm and dataset are sufficient to uniquely specify a setting.

We use the regime classification introduced in Section 3 to organize our analysis and to interpret the outcomes of the fine-tuning methods. Each of the 63 experimental settings, defined by a unique combination of pretraining algorithm, dataset, and task, is assigned to one of the three regimes based on the relative performance of the pretrained policy and the offline dataset. Specifically, we conduct t-tests with a margin $\delta = 0.05$ to assess whether the difference between $J(\pi_0)$ and $J(\pi_\mathcal{D})$ is statistically significant. The margin δ is introduced for robustness, since $J(\pi_\mathcal{D})$ is approximated by the dataset average return and small gaps between $J(\pi_0)$ and $J(\pi_\mathcal{D})$ may not be meaningful. It prevents over-interpreting numerical noise in regime assignment. The complete set of regime assignments is reported in Table 13 in the appendix.

Online fine-tuning phase. To ensure consistency between offline pretraining phase and online fine-tuning phase, we fine-tune each agent using the corresponding base algorithm. Specifically, we fine-tune CalQL-pretrained agents using SAC [18] and ReBRAC-pretrained agents using TD3 [19]. For the deterministic BC pretraining, we use TD3 for fine-tuning to match its deterministic actor structure, and use ReBRAC when applying regularization.

Since evaluating every possible design and their combinations is infeasible, we have grouped them into four categories: the minimal baseline, π_0-centric methods, $\mathcal{D}$-centric methods, and mixed $\pi_0 + \mathcal{D}$ methods. We then evaluate six representative methods spanning these four categories, which collectively capture the key design choices explored in prior offline-to-online RL literature [15, 9, 10].

- **Baseline:** Fine-tuning the pretrained policy using an online RL algorithm with only online data.
- **π_0-centric methods:** Two variants of such methods are evaluated: the baseline with (i) online data warmup ($K = 5{,}000$ steps) and (ii) offline RL regularization using the pretraining coefficient.
- **$\mathcal{D}$-centric methods:** Two variants of such methods are evaluated: the baseline with (i) offline data replay, and (ii) with offline data replay and reset. Both variants use separate replay buffers with an offline data ratio of $\alpha = 0.5$.
- **Mixed $\pi_0 + \mathcal{D}$ methods:** The baseline combined with offline data replay and offline RL regularization, a combination widely adopted in prior work [6, 20, 21, 15].

In each setting, we focus and compare the strongest π_0-centric and $\mathcal{D}$-centric methods to better approximate the ideal performance achievable by each stability source, while minimizing confounding from implementation details and hyperparameter tuning. In the following subsections, We first present and analyze the fine-tuning results across regimes: the **Superior** regime (Sec. 5.1), the **Inferior** regime (Sec. 5.2), and the **Comparable** regime (Sec. 5.3). Finally, in Sec. 5.4, we examine the value learning dynamics during fine-tuning to uncover the underlying mechanisms. Additionally, Appendix B.2 details the empirical values of stability and plasticity, discussing how they align with intuition.

5.1 Superior Regime: $J(\pi_0) > J(\pi_\mathcal{D})$

In this regime, the pretrained policy π_0 achieves substantially higher performance than the offline dataset, which is common for suboptimal or low-quality datasets. In such cases, the offline dataset offers limited additional value, and the primary concern becomes preserving stability relative to π_0.

Representative results are shown in Figure 3, with the complete results in this regime provided in the appendix (Figure 6). We perform t-tests between the strongest π_0-centric and $\mathcal{D}$-centric methods in each setting. The results indicate π_0-centric methods outperform $\mathcal{D}$-centric methods in 24 out of

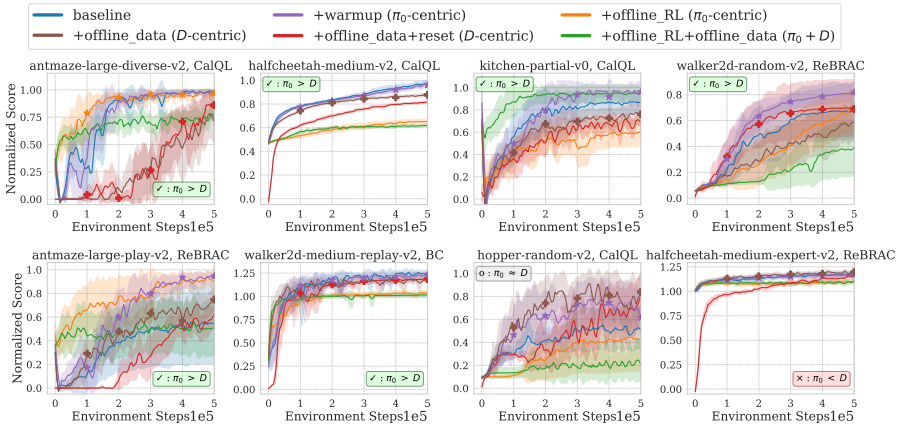

Figure 3: Representative fine-tuning results in the **Superior** regime: the first row and first two subplots in the second row are correct predictions, while the remaining two show an adjacent mismatch and an opposite mismatch. Markers on the curves indicate the better-performing variant within π_0-centric methods and within $\mathcal{D}$-centric methods.

32 settings (75%), while the remaining settings mostly show no statistically significant difference. These statistics correspond to the **Superior** column of the confusion matrix in Table 1.

These aggregate outcomes strongly support our principle that, in the **Superior** regime, π_0-centric methods tend to be more effective than $\mathcal{D}$-centric methods. In other words, when the pretrained policy π_0 already outperforms the dataset, methods that stick close to π_0 work better than those that keep leaning on the offline dataset. While the prediction accuracy is not perfect, such discrepancies are anticipated given the influence of hyperparameters and implementation details. Importantly, the overall observed patterns remain consistent with our principle.

Beyond aggregate comparisons, the analysis of specific design choices highlights key trade-offs between stability and plasticity. Comparing the two π_0-centric methods, online data warmup achieves better performance in 27 out of 32 settings, reflecting its ability to preserve the pretrained policy's knowledge while maintaining sufficient plasticity. Conversely, while offline RL regularization minimizes early performance degradation through stronger stability, it severely restricts the plasticity needed for long-term improvement. Consequently, it only outperforms warmup (5 of 32 settings) when the pretrained policy is already near-optimal. The same applies to the combination of offline RL regularization with offline data replay, which exhibits the strongest stability among all methods considered. These contrasts highlight the importance of considering each setting and identifying the method that best balances the underlying stability–plasticity trade-off.

Takeaway: In the **Superior** regime, where the pretrained policy π_0 substantially outperforms the offline dataset $\mathcal{D}$, π_0-centric methods are typically more effective than $\mathcal{D}$-centric methods. Stronger stability proves beneficial primarily when π_0 is already close to optimal.

5.2 **Inferior** Regime: $J(\pi_0) < J(\pi_\mathcal{D})$

In this regime, the pretrained policy π_0 performs much worse than the $\pi_\mathcal{D}$, which is common for near-expert datasets in sparse-reward domains. Thus, π_0 contributes substantially less useful knowledge than the offline dataset, making it crucial to retain and leverage the offline data.

Representative results are shown in Figure 4, with the complete results in this regime provided in the appendix (Figure 7). $\mathcal{D}$-centric methods outperform π_0-centric methods in 19 out of 23 settings (83%), while the remaining settings mostly show no statistically significant difference. These statistics correspond to the **Inferior** column of the confusion matrix in Table 1. Taken together, these aggregate results show that in the **Inferior** regime, $\mathcal{D}$-centric methods tend to be more effective than π_0-centric methods, consistent with the prediction of our framework.

Notably, offline data replay with reset achieves better performance than offline data replay in 13 out of 24 settings, despite the fact that reset initially causes significant degradation. This indicates that in these cases the offline pretraining phase substantially reduces plasticity while offering limited useful knowledge, and resetting the parameters allows the agent to adapt and acquire new knowledge more effectively. Furthermore, combining offline RL regularization with offline data generally underperforms compared to $\mathcal{D}$-centric methods. Although this design leverages offline data during fine-tuning, which is essential in this regime, the excessive stability limits plasticity and thereby hinders further improvement.

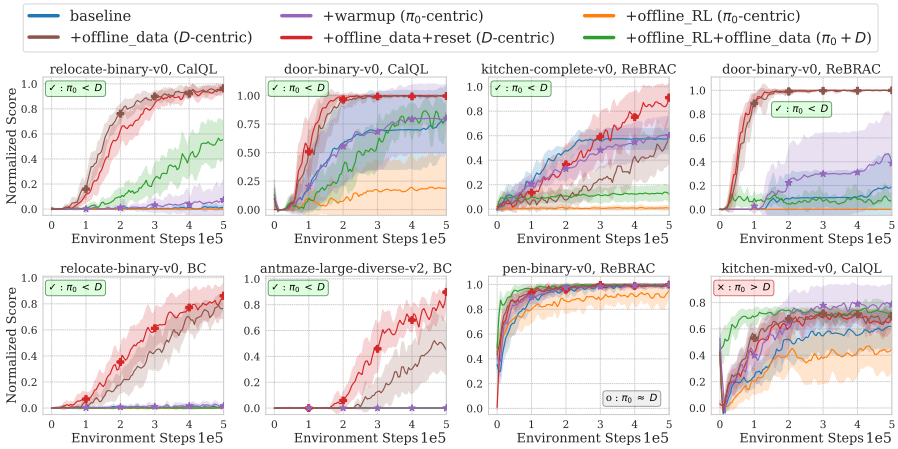

Figure 4: Representative results in the **Inferior** regime: the first six results are correct predictions, while the remaining two show an adjacent mismatch and an opposite mismatch.

Takeaway: In the **Inferior** regime, where the pretrained policy π_0 performs substantially worse than the offline dataset $\mathcal{D}$, $\mathcal{D}$-centric methods typically yield more effective fine-tuning. High plasticity (e.g., full reset) is preferred when the pretrained agent performs poorly.

5.3 **Comparable** Regime: $J(\pi_0) \approx J(\pi_\mathcal{D})$

In this regime, the pretrained policy and the offline dataset yield similar performance. The complete results are provided in the appendix (Figure 8). Our framework predicts that π_0-centric and $\mathcal{D}$-centric methods should yield comparable outcomes once fully optimized. Empirically, only 2 out of 8 settings are statistically indistinguishable under t-tests. This seems at odds with the prediction. However, closer inspection shows that the differences are minor: in 6 of 8 settings the mean gap between categories is less than 0.1. These small gaps indicate that both anchors provide similar prior knowledge, exactly as the framework suggests.

Why, then, do mismatches arise at all? The key is that effect sizes in this regime are small by construction. When π_0 and $\mathcal{D}$ are nearly tied, outcomes become highly sensitive to hyperparameters, initialization, and other implementation details. In our study, we fixed a limited set of representative variants and hyperparameters across all settings to avoid over-tuning. This conservative design choice helps comparability but can also tip results in such close cases.

Takeaway: In the **Comparable** regime, where the pretrained policy π_0 and the offline dataset $\mathcal{D}$ exhibit similar performance, π_0-centric and $\mathcal{D}$-centric methods should in principle yield comparable fine-tuning outcomes when fully optimized, though in practice their relative performance is often sensitive to implementation details.

5.4 Mechanistic Analysis

While we characterize stability and plasticity through performance metrics in our main taxonomy, it is crucial to understand the underlying mechanisms driving these behavioral outcomes. Given that value learning represents a primary bottleneck in offline-to-online RL [15, 9], we analyze how the Q-function evolves during fine-tuning to uncover the mechanisms behind the **Superior** and **Inferior** regimes. Figure 5 illustrates a representative pattern using CalQL pretraining, comparing a **Superior** regime and an **Inferior** regime; more results are provided in Appendix D. In each regime, we evaluate a π_0-centric method (warmup) against a $\mathcal{D}$-centric method (offline data). To understand the mechanism about plasticity, we examine the temporal difference (TD) loss on the newly collected online data during fine-tuning. As shown in the rightmost column of Figure 5, across both regimes, fine-tuning with warmup achieves a lower online TD loss compared to using the offline dataset, mechanistically facilitating its ability to adapt to the online data distribution by imposing weaker regularization.

Conversely, to understand the mechanism behind stability, we evaluate the TD loss and Q-value on data from offline dataset $\mathcal{D}$, as shown in the second and third columns of Figure 5. While fine-tuning with offline data safely anchors the critic, the warmup method lacks this anchoring and suffers a severe optimization breakdown in the **Inferior** regime. Its offline TD loss exceeds 10^3 during the first 100k steps, causing drastic Q-value divergence. Although a similar destabilization occurs in the **Superior** regime, it is significantly milder, with the offline TD loss peaking near 10^2 and avoiding Q-value divergence on the offline data.

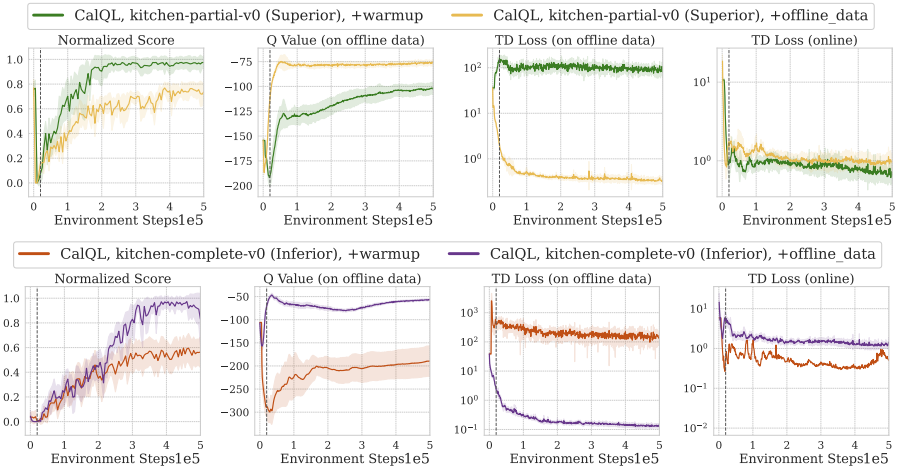

Figure 5: Comparison of value learning dynamics in different regimes. Fine-tuning without offline data results in significantly higher TD loss on the offline dataset in the **Inferior** regime compared to the **Superior** regime, and also leads to divergence of the Q-values on the offline data.

By linking the outcomes to the value learning dynamics, this evaluation mechanistically illustrates why anchoring to the offline dataset $\mathcal{D}$ is crucial in the **Inferior** regime to prevent exploding TD loss and diverging Q-values, whereas such anchoring is not necessary in the **Superior** regime.

5.5 Alternative Taxonomies of Three Regimes

To validate our raw-return metric, we benchmark against alternative regime classifications based on: (1) dense reward proxies for sparse-reward domains (Adroit, AntMaze, and Kitchen), (2) learned Q-values, and (3) BC performance. To ensure a fair comparison with raw returns, the dense rewards are constructed without access to privileged task information (e.g., maze layouts) and are used as coarse progress proxies. Full details are provided in Appendix C. As shown in Table 2, our raw-return-based taxonomy outperforms the alternatives, achieving the best classification accuracy and minimizing opposite mismatches. This demonstrates that raw return provides a significantly more reliable criterion for regime identification, as it directly aligns with the true task objectives rather than relying on potentially divergent heuristic proxies.

Table 2: Results of taxonomies based on alternative metrics beyond raw return, averaged across benchmarks. Dense-reward proxies are applied only to sparse-reward domains.

Domains	Metric	Accuracy ↑	Opposite mismatch ↓
Sparse-reward (27 cases)	Raw return (ours)	**78%**	**4%**
	Dense-reward proxy	65%	11%
All domains (63 cases)	Raw return (ours)	**71%**	**5%**
	Q-function	51%	30%
	BC performance	41%	**5%**

6 Conclusion and Discussion

This paper introduces a framework guided by the stability–plasticity principle to reconcile the puzzling variability of offline-to-online RL. We showed that the key determinant of fine-tuning success is anchoring stability to the dominant offline prior, whether that is the pretrained policy or the offline dataset. From this observation we derived a taxonomy of three regimes, each dictating where stability should be enforced and how plasticity should be managed. The value of this framework is twofold. First, it provides a clear explanation for the conflicting empirical evidence in the literature: design choices that seem inconsistent across benchmarks in fact reflect different underlying regimes. Second, it offers actionable guidance for practitioners. By identifying the regime of a given setting, one can select methods that align with its stability–plasticity requirements, reducing reliance on trial-and-error.

Limitations and Future Work. Our regime taxonomy provides an efficient lens for understanding offline-to-online RL by compressing complex phenomena into a small number of discrete categories. Such taxonomies have been widely used in both the natural sciences and machine learning, for example in imitation learning, where regimes have been proposed based on density ratios [24] or dataset size [25]. We ground our taxonomy in raw return because task performance serves as the dominant, first-order determinant of fine-tuning dynamics in our three regimes framework. Other characteristics, such as state-action coverage of the dataset, may act as second-order factors that influence the learning process. Extending our framework to incorporate these second-order dimensions alongside performance to yield a more comprehensive taxonomy is an important direction for future work.

Acknowledgments

Lu Li would like to thank Guozheng Ma for valuable discussions during the preparation of this work. The research was enabled in part by computational resources provided by the Digital Research Alliance of Canada (`https://alliancecan.ca`) and Mila (`https://mila.quebec`). We acknowledge funding support from CIFAR.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[2] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[3] Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

[4] 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.

[5] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, Jan Peters, et al. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2):1–179, 2018.

[6] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[7] Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR, 2022.

[8] 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.

[9] Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. *arXiv preprint arXiv:2412.07762*, 2024.

[10] Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023.

[11] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[12] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[13] Maciej Wolczyk, Bartłomiej Cupiał, Mateusz Ostaszewski, Michał Bortkiewicz, Michał Zajac, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Fine-tuning reinforcement learning models is secretly a forgetting mitigation problem. In *International Conference on Machine Learning*, pages 53039–53078. PMLR, 2024.

[14] Shibhansh Dohare, J Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A Rupam Mahmood, and Richard S Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026):768–774, 2024.

[15] Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

[16] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[17] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International conference on machine learning*, pages 16828–16847. PMLR, 2022.

[18] 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*, pages 1861–1870. Pmlr, 2018.

[19] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[20] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[21] Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

[22] Stefan Schaal. Learning from demonstration. *Advances in neural information processing systems*, 9, 1996.

[23] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

[24] Jonathan Spencer, Sanjiban Choudhury, Arun Venkatraman, Brian Ziebart, and J Andrew Bagnell. Feedback in imitation learning: The three regimes of covariate shift. *arXiv preprint arXiv:2102.02872*, 2021.

[25] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *Advances in neural information processing systems*, 36:80375–80395, 2023.

[26] Jianxiong Li, Xiao Hu, Haoran Xu, Jingjing Liu, Xianyuan Zhan, and Ya-Qin Zhang. Proto: Iterative policy regularized offline-to-online reinforcement learning. *arXiv preprint arXiv:2305.15669*, 2023.

[27] Xu-Hui Liu, Tian-Shuo Liu, Shengyi Jiang, Ruifeng Chen, Zhilong Zhang, Xinwei Chen, and Yang Yu. Energy-guided diffusion sampling for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, pages 31541–31565. PMLR, 2024.

[28] Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, et al. Jump-start reinforcement learning. In *International Conference on Machine Learning*, pages 34556–34583. PMLR, 2023.

[29] Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2023.

[30] Hao Hu, Yiqin Yang, Jianing Ye, Chengjie Wu, Ziqing Mai, Yujing Hu, Tangjie Lv, Changjie Fan, Qianchuan Zhao, and Chongjie Zhang. Bayesian design principles for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, pages 19491–19515. PMLR, 2024.

[31] Yuda Song, Yifei Zhou, Ayush Sekhari, J Andrew Bagnell, Akshay Krishnamurthy, and Wen Sun. Hybrid rl: Using both offline and online data can make rl efficient. *arXiv preprint arXiv:2210.06718*, 2022.

[32] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, 4:504, 2013.

[33] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

[34] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[35] Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In *International Conference on Machine Learning*, pages 32145–32168. PMLR, 2023.

Contents

A Related Work

A.1 Offline-to-Online RL

Offline-to-online RL seeks to combine the strengths of offline pretraining with the adaptability of online fine-tuning. Early approaches focused on extending offline RL regularization methods into the online regime, constraining fine-tuning updates to remain close to the pretrained policy. For example, Advantage Weighted Actor-Critic (AWAC) [6] and Implicit Q-Learning (IQL) [20] applied offline regularization techniques directly to online fine-tuning. Building on the similar idea, PROTO [26] introduced KL regularization to explicitly constrain the online policy to the pretrained one. Another line of work proposes new replay strategies for incorporating offline data more effectively. Lee et al. [7] propose balanced replay to mitigate distribution shift and bootstrap error when transitioning from offline to online learning, EDIS[27] employs a diffusion model to select or generate samples. Alternative strategies separate the roles of exploration and exploitation during fine-tuning. Jump-Start RL (JSRL) [28], maintains a fixed guided policy from pretraining alongside an exploration policy that is updated online, progressively transferring control from the pretrained policy to the learned one. More recent directions include expanding the action space via policy set expansion (PEX [29]) and Bayesian methods for uncertainty-aware exploration (BOORL [30]). Some approaches skip offline pretraining but still make use of offline data. For example, Hybrid RL [31] and RLPD [10] start training directly with online RL while incorporating offline datasets, providing another way to combine offline data with online interaction.

A.2 Plasticity and Stability

Plasticity and stability have long been recognized as central, often competing, objectives in learning systems. In neuroscience, this tension is formalized as the stability–plasticity dilemma [32], highlighting the challenge of integrating new knowledge without overwriting previously acquired competencies. In machine learning, similar dynamics manifest when agents must adapt to new data while preserving useful prior knowledge. Early work on continual and lifelong learning addressed this challenge via regularization techniques [12], replay buffers [33] and modular architectures [34]. More recently, researchers have observed that insufficient plasticity can also hinder online deep RL, motivating methods designed to enhance plasticity of the neural network during training [17, 35, 14].

We extend this perspective to offline-to-online RL by framing fine-tuning as a stability–plasticity trade-off between preserving knowledge from pretraining and adapting to new online data. While prior work has examined forgetting and plasticity in continual and online RL, their role in the offline-to-online transition has received limited attention. Our framework shows that stability–plasticity is not only an explanatory lens, but also yields actionable guidance by predicting which design choices are effective in different regimes.

B Detailed Experimental Setup and Complete Results

B.1 Offline Pretraining

For offline pretraining, we train CalQL for 1M gradient steps on AntMaze, 20k on Adroit, and 250k on both Kitchen and MuJoCo locomotion tasks. ReBRAC is trained for 1M gradient steps on AntMaze, 100k on Adroit, 250k on Kitchen, and 500k on MuJoCo tasks. For the behavior cloning (BC) baseline, we perform 500K gradient steps of policy learning followed by 100k steps of fitted Q evaluation [23] (FQE) to obtain a Q-function for subsequent RL fine-tuning. Since this work primarily focuses on the online fine-tuning stage of offline-to-online RL, we do not modify the offline pretraining algorithm or its default hyperparameters.

For regime classification, we employ the two one-sided t-test (TOST) procedure with a margin of $\delta = 0.05$ and a significance level of $\alpha = 0.05$. The goal is to formally assess whether the pretrained policy π_0 and the offline dataset $\pi_\mathcal{D}$ are statistically indistinguishable in performance, or whether one is significantly superior. Let μ_0 and $\mu_\mathcal{D}$ denote the mean returns of π_0 and $\pi_\mathcal{D}$. We conduct two one-sided tests for the null hypotheses $H_0 : \mu_0 - \mu_\mathcal{D} \leq -\delta$ and $H_0 : \mu_0 - \mu_\mathcal{D} \geq \delta$. If both null hypotheses are rejected, the difference is within the margin and the two are considered comparable, leading to assignment to the **Comparable** Regime. If only one hypothesis is rejected, the difference is statistically significant and exceeds the margin, and the setting is assigned to either the **Superior** or **Inferior** Regime depending on which policy achieves the higher mean return. The statistics for each dataset and pretraining policy are reported in Table 13.

While we use the margin parameter $\delta = 0.05$ in the main experiments, we further assess the sensitivity to δ by comparing results under $\delta = 0$ and $\delta = 0.1$. The corresponding fine-tuning confusion matrices are reported in Table 3 and Table 4.

Table 3: Confusion matrix of fine-tuning results across the three pretraining regimes with margin $\delta = 0$.

		Pretraining Regime ($\delta = 0$)		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**26**	0	1
	π_0-centric $\approx$ $\mathcal{D}$-centric	8	**0**	3
	π_0-centric $<$ $\mathcal{D}$-centric	4	1	**20**

Table 4: Confusion matrix of fine-tuning results across the three pretraining regimes with margin $\delta = 0.1$.

		Pretraining Regime ($\delta = 0.1$)		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**18**	9	0
	π_0-centric $\approx$ $\mathcal{D}$-centric	4	**5**	2
	π_0-centric $<$ $\mathcal{D}$-centric	2	9	**14**

B.2 Online Fine-Tuning

Across all environments, online fine-tuning is performed for 500k environment steps with UTD=1. The complete results, categorized according to the regime taxonomy, are reported in Figure 6, Figure 7, and Figure 8. To obtain the strongest performance for each class in each settings, we compare the interquartile mean (IQM) of evaluation results of online data warmup and offline RL regularization within π_0-centric methods, and analogously compare offline data replay with and without reset within $\mathcal{D}$-centric methods. We then use the higher value from each class, comparing π_0-centric and $\mathcal{D}$-centric methods using two-sided t-tests with $\alpha = 0.05$. To obtain stable and reliable t-test statistics, we base our analysis on the last 10 evaluation results from each random seed during online fine-tuning, which correspond to the final 50k training steps given our evaluation frequency of every 5k steps. An exception is made for *door-binary-v0* and *pen-binary-v0*, where we instead use results up to 200k steps, since by the end of training nearly all methods achieve a 100% success rate, leaving no differences.

We report the empirical values of stability, plasticity, and improvement during fine-tuning for the **Superior**, **Inferior**, **Comparable**, and all regimes. Since these quantities depend on the fine-tuning steps, we present results at both 50k and 500k environment steps, representing the *early* and *late* stages of fine-tuning, as shown in Table 5–Table 8.

Further discussion on early and late fine-tuning stages. Plasticity often correlates with performance improvement, but it is not sufficient on its own to guarantee strong results. Our analyses reveal the following regime- and stage-dependent patterns:

- **Superior regime.** The "offline data + reset" method attains the highest plasticity, yet yields the *lowest* improvement at 50k steps and only moderate improvement at 500k steps. This shows that when π_0 is strong, plasticity alone does not determine performance; maintaining stability is essential.
- **Inferior regime.** The relationship between stability, plasticity, and improvement depends on the fine-tuning stage. At the late stage (500k steps), methods with the highest plasticity tend to achieve the greatest improvement because π_0 performs poorly and can degrade toward near-zero performance during fine-tuning. In cases where $\min_j J(\pi_j) = 0$, the improvement simplifies to plasticity $- J^*_{\text{off}}$, making plasticity the dominant factor. In contrast, at the early stage (50k steps), the "offline RL + offline data" method attains the highest improvement while also exhibiting the strongest stability, despite having only moderate plasticity. This indicates that stability can have a stronger influence on improvement early in fine-tuning.

Takeaway: Plasticity should be emphasized when π_0 is weak (**Inferior** regime) and the fine-tuning budget is large (500k steps), while stability becomes more important when π_0 is strong (**Superior** regime) and the fine-tuning budget is small (50k steps).

Table 5: Empirical values of stability, plasticity, and improvement of different fine-tuning methods in **Superior** Regime for 50k and 500k environment steps.

Fine-tuning method in **Superior** regime	Stability ↑	Plasticity ↑	Improvement ↑
50k environment steps			
baseline	-0.352 ± 0.344	0.525 ± 0.325	0.172 ± 0.145
+ warmup (π_0-centric)	-0.328 ± 0.348	0.501 ± 0.315	$\mathbf{0.173 \pm 0.122}$
+ offline RL (π_0-centric)	-0.162 ± 0.242	0.289 ± 0.255	0.127 ± 0.151
+ offline data ($\mathcal{D}$-centric)	-0.293 ± 0.351	0.448 ± 0.318	0.155 ± 0.138
+ offline data + reset ($\mathcal{D}$-centric)	-0.615 ± 0.338	$\mathbf{0.581 \pm 0.349}$	-0.034 ± 0.311
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.072 \pm 0.128}$	0.204 ± 0.189	0.132 ± 0.138
500k environment steps			
baseline	-0.399 ± 0.363	0.832 ± 0.275	0.433 ± 0.276
+ warmup (π_0-centric)	-0.394 ± 0.371	0.865 ± 0.251	$\mathbf{0.471 \pm 0.258}$
+ offline RL (π_0-centric)	-0.199 ± 0.268	0.519 ± 0.291	0.319 ± 0.256
+ offline data ($\mathcal{D}$-centric)	-0.376 ± 0.380	0.796 ± 0.282	0.421 ± 0.251
+ offline data + reset ($\mathcal{D}$-centric)	-0.615 ± 0.338	$\mathbf{1.011 \pm 0.183}$	0.396 ± 0.271
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.124 \pm 0.213}$	0.393 ± 0.288	0.270 ± 0.233

Table 6: Empirical values of stability, plasticity and improvement of different fine-tuning methods in **Inferior** Regime for 50k and 500k environment steps.

Fine-tuning method in **Inferior** regime	Stability ↑	Plasticity ↑	Improvement ↑
50k environment steps			
baseline	-0.688 ± 0.348	0.216 ± 0.219	-0.472 ± 0.408
+ warmup (π_0-centric)	-0.671 ± 0.343	0.213 ± 0.217	-0.458 ± 0.422
+ offline RL (π_0-centric)	-0.665 ± 0.342	0.159 ± 0.197	-0.505 ± 0.428
+ offline data ($\mathcal{D}$-centric)	-0.670 ± 0.343	0.268 ± 0.215	-0.402 ± 0.398
+ offline data + reset ($\mathcal{D}$-centric)	-0.832 ± 0.191	$\mathbf{0.417 \pm 0.301}$	-0.415 ± 0.395
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.640 \pm 0.357}$	0.266 ± 0.219	$\mathbf{-0.374 \pm 0.415}$
500k environment steps			
baseline	-0.692 ± 0.307	0.475 ± 0.378	-0.217 ± 0.480
+ warmup (π_0-centric)	-0.677 ± 0.306	0.495 ± 0.389	-0.182 ± 0.490
+ offline RL (π_0-centric)	-0.650 ± 0.322	0.262 ± 0.283	-0.388 ± 0.489
+ offline data ($\mathcal{D}$-centric)	-0.674 ± 0.308	0.696 ± 0.340	0.023 ± 0.451
+ offline data + reset ($\mathcal{D}$-centric)	-0.769 ± 0.275	$\mathbf{0.870 \pm 0.299}$	$\mathbf{0.101 \pm 0.205}$
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.614 \pm 0.354}$	0.440 ± 0.297	-0.174 ± 0.441

B.3 Hyperparameters

We summarize the hyperparameters used in our empirical studies. The hyperparameters for SAC and CalQL are given in Table 9. Common hyperparameters of TD3 and ReBRAC appear in Table 10, while Table 11 contains the task-dependent hyperparameters for ReBRAC. Table 12 reports the hyperparameters for BC and FQE.

Table 7: Empirical values of stability, plasticity and improvement of different fine-tuning methods in **Comparable** Regime for 50k and 500k environment steps.

Fine-tuning method in **Comparable** regime	Stability ↑	Plasticity ↑	Improvement ↑
50k environment steps			
baseline	-0.083 ± 0.084	0.163 ± 0.237	0.080 ± 0.251
+ warmup (π_0-centric)	-0.082 ± 0.085	0.163 ± 0.195	0.081 ± 0.212
+ offline RL (π_0-centric)	-0.079 ± 0.085	0.214 ± 0.264	0.135 ± 0.270
+ offline data ($\mathcal{D}$-centric)	-0.081 ± 0.085	0.147 ± 0.161	0.066 ± 0.166
+ offline data + reset ($\mathcal{D}$-centric)	-0.114 ± 0.147	$\mathbf{0.261 \pm 0.250}$	0.147 ± 0.237
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.057} \pm 0.058$	0.240 ± 0.251	$\mathbf{0.182 \pm 0.258}$
500k environment steps			
baseline	-0.043 ± 0.071	0.560 ± 0.411	0.516 ± 0.452
+ warmup (π_0-centric)	-0.042 ± 0.072	0.543 ± 0.390	0.501 ± 0.423
+ offline RL (π_0-centric)	-0.043 ± 0.072	0.564 ± 0.382	0.521 ± 0.416
+ offline data ($\mathcal{D}$-centric)	-0.042 ± 0.072	0.680 ± 0.338	0.638 ± 0.361
+ offline data + reset ($\mathcal{D}$-centric)	-0.106 ± 0.139	0.623 ± 0.431	0.516 ± 0.431
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	$\mathbf{-0.035 \pm 0.061}$	$\mathbf{0.682 \pm 0.302}$	$\mathbf{0.646 \pm 0.314}$

Table 8: Empirical values of stability, plasticity and improvement of different fine-tuning methods across **all regimes** for 50k and 500k environment steps.

Fine-tuning method	Stability ↑	Plasticity ↑	Improvement ↑
50k environment steps			
baseline	-0.433 ± 0.379	0.397 ± 0.329	-0.036 ± 0.395
+ warmup (π_0-centric)	-0.413 ± 0.380	0.382 ± 0.315	-0.031 ± 0.389
+ offline RL (π_0-centric)	-0.311 ± 0.359	0.242 ± 0.246	-0.069 ± 0.404
+ offline data ($\mathcal{D}$-centric)	-0.391 ± 0.387	0.365 ± 0.298	-0.026 ± 0.357
+ offline data + reset ($\mathcal{D}$-centric)	-0.632 ± 0.349	0.486 ± 0.340	-0.146 ± 0.393
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	-0.247 ± 0.345	0.226 ± 0.206	-0.021 ± 0.357
500k environment steps			
baseline	-0.461 ± 0.381	0.667 ± 0.375	0.206 ± 0.502
+ warmup (π_0-centric)	-0.453 ± 0.382	0.689 ± 0.372	0.236 ± 0.495
+ offline RL (π_0-centric)	-0.344 ± 0.362	0.431 ± 0.327	0.087 ± 0.526
+ offline data ($\mathcal{D}$-centric)	-0.442 ± 0.388	0.745 ± 0.316	0.303 ± 0.415
+ offline data + reset ($\mathcal{D}$-centric)	-0.607 ± 0.360	0.910 ± 0.299	0.304 ± 0.318
+ offline RL + offline data (mixed $\pi_0 + \mathcal{D}$)	-0.291 ± 0.360	0.447 ± 0.307	0.155 ± 0.433

B.4 Compute Details

All experiments were conducted on a single-GPU setup using an NVIDIA L40S GPU, 24 CPU workers, and 20GB of RAM.

Table 9: SAC and CalQL's hyperparameters.

Parameter	**Value**
optimizer	Adam
batch size	1024
learning rate	1e-4
Q-function soft-update rate (τ)	5e-3
discount factor (γ)	0.999 on AntMaze, 0.99 on other
CQL n actions	10
CQL α	1 for Adroit, 5 for others
CQL max target backup	True

Table 10: Common hyperparameters of TD3 and ReBRAC.

Parameter	**Value**
optimizer	Adam
batch size	1024
learning rate	1e-4 on AntMaze, 1e-3 on other
Q-function soft-update rate (τ)	5e-3
discount factor (γ)	0.999 on AntMaze, 0.99 on other

Table 11: ReBRAC hyperparameters used in our experiments. All hyperparameters follow the best hyperparameters reported in ReBRAC [21], except for the Kitchen domain, which was not included; for Kitchen, we performed a hyperparameter search and selected the best configuration.

Task Name	β_1 (actor)	β_2 (critic)
halfcheetah-random	0.001	0.1
halfcheetah-medium	0.001	0.01
halfcheetah-medium-expert	0.01	0.1
halfcheetah-medium-replay	0.01	0.001
hopper-random	0.001	0.01
hopper-medium	0.01	0.001
hopper-medium-expert	0.1	0.01
hopper-medium-replay	0.05	0.5
walker2d-random	0.01	0.0
walker2d-medium	0.05	0.1
walker2d-medium-expert	0.01	0.01
walker2d-medium-replay	0.05	0.01
antmaze-large-play	0.002	0.001
antmaze-large-diverse	0.002	0.002
antmaze-ultra-diverse	0.002	0.002
door-binary	0.1	0.01
pen-binary	0.1	0.01
relocate-binary	0.1	0.01
kitchen-complete	0.1	0.001
kitchen-mixed	0.1	0.001
kitchen-partial	0.1	0.001

Table 12: BC and FQE's hyperparameters.

Parameter	**Value**
optimizer	Adam
batch size	1024
learning rate	3e-4
action	deterministic
FQE steps	1e5
Q-function soft-update rate (τ)	5e-3
discount factor (γ)	0.999 on AntMaze, 0.99 on other

Table 13: Statistics (mean $\pm$ std) of offline datasets and pretrained policies. Pretrained policy scores $J(\pi_0)$ are reported as averages over 10 random seeds.

Dataset	**Dataset**		**CalQL**		**ReBRAC**		**BC**	
	$J(\pi_\mathcal{D})$	#Trajs	$J(\pi_0)$	order	$J(\pi_0)$	order	$J(\pi_0)$	order
halfcheetah-random-v2	-0.001 $\pm$ 0.006	1000	0.248 $\pm$ 0.014	$>$	0.275 $\pm$ 0.011	$>$	0.019 $\pm$ 0.001	$\approx$
halfcheetah-medium-replay-v2	0.271 $\pm$ 0.135	202	0.451 $\pm$ 0.002	$>$	0.504 $\pm$ 0.003	$>$	0.370 $\pm$ 0.007	$>$
halfcheetah-medium-v2	0.406 $\pm$ 0.029	1000	0.470 $\pm$ 0.003	$>$	0.651 $\pm$ 0.010	$>$	0.427 $\pm$ 0.002	$\approx$
halfcheetah-medium-expert-v2	0.643 $\pm$ 0.239	2000	0.519 $\pm$ 0.078	$<$	1.010 $\pm$ 0.019	$>$	0.563 $\pm$ 0.025	$<$
hopper-random-v2	0.012 $\pm$ 0.005	45240	0.091 $\pm$ 0.017	$>$	0.082 $\pm$ 0.036	$\approx$	0.038 $\pm$ 0.022	$\approx$
hopper-medium-replay-v2	0.150 $\pm$ 0.157	2039	1.001 $\pm$ 0.009	$>$	0.965 $\pm$ 0.042	$>$	0.398 $\pm$ 0.037	$>$
hopper-medium-v2	0.443 $\pm$ 0.117	2187	0.672 $\pm$ 0.039	$>$	1.016 $\pm$ 0.012	$>$	0.554 $\pm$ 0.010	$>$
hopper-medium-expert-v2	0.648 $\pm$ 0.319	3214	1.058 $\pm$ 0.108	$>$	1.063 $\pm$ 0.042	$>$	0.556 $\pm$ 0.012	$<$
walker2d-random-v2	0.000 $\pm$ 0.001	48908	0.082 $\pm$ 0.043	$>$	0.058 $\pm$ 0.001	$>$	0.008 $\pm$ 0.001	$\approx$
walker2d-medium-replay-v2	0.148 $\pm$ 0.195	1093	0.843 $\pm$ 0.025	$>$	0.853 $\pm$ 0.045	$>$	0.276 $\pm$ 0.074	$>$
walker2d-medium-v2	0.620 $\pm$ 0.239	1191	0.742 $\pm$ 0.071	$>$	0.845 $\pm$ 0.008	$>$	0.507 $\pm$ 0.073	$<$
walker2d-medium-expert-v2	0.826 $\pm$ 0.285	2191	1.073 $\pm$ 0.035	$>$	1.113 $\pm$ 0.004	$>$	1.075 $\pm$ 0.004	$>$
pen-binary-v0	1.000 $\pm$ 0.000	846	0.657 $\pm$ 0.059	$<$	0.451 $\pm$ 0.086	$<$	0.589 $\pm$ 0.068	$<$
door-binary-v0	1.000 $\pm$ 0.000	82	0.112 $\pm$ 0.123	$<$	0.000 $\pm$ 0.000	$<$	0.000 $\pm$ 0.000	$<$
relocate-binary-v0	1.000 $\pm$ 0.000	36	0.010 $\pm$ 0.011	$<$	0.000 $\pm$ 0.000	$<$	0.000 $\pm$ 0.000	$<$
kitchen-partial-v0	0.586 $\pm$ 0.187	600	0.764 $\pm$ 0.094	$>$	0.133 $\pm$ 0.085	$<$	0.222 $\pm$ 0.079	$<$
kitchen-mixed-v0	0.598 $\pm$ 0.145	600	0.464 $\pm$ 0.092	$<$	0.034 $\pm$ 0.033	$<$	0.275 $\pm$ 0.049	$<$
kitchen-complete-v0	1.000 $\pm$ 0.000	19	0.043 $\pm$ 0.078	$<$	0.002 $\pm$ 0.004	$<$	0.385 $\pm$ 0.202	$<$
antmaze-large-diverse-v2	0.106 $\pm$ 0.308	999	0.305 $\pm$ 0.055	$>$	0.399 $\pm$ 0.080	$>$	0.000 $\pm$ 0.000	$<$
antmaze-large-play-v2	0.105 $\pm$ 0.307	999	0.247 $\pm$ 0.068	$>$	0.351 $\pm$ 0.072	$>$	0.000 $\pm$ 0.000	$<$
antmaze-ultra-diverse-v2	0.053 $\pm$ 0.224	999	0.118 $\pm$ 0.072	$\approx$	0.127 $\pm$ 0.132	$\approx$	0.000 $\pm$ 0.000	$\approx$

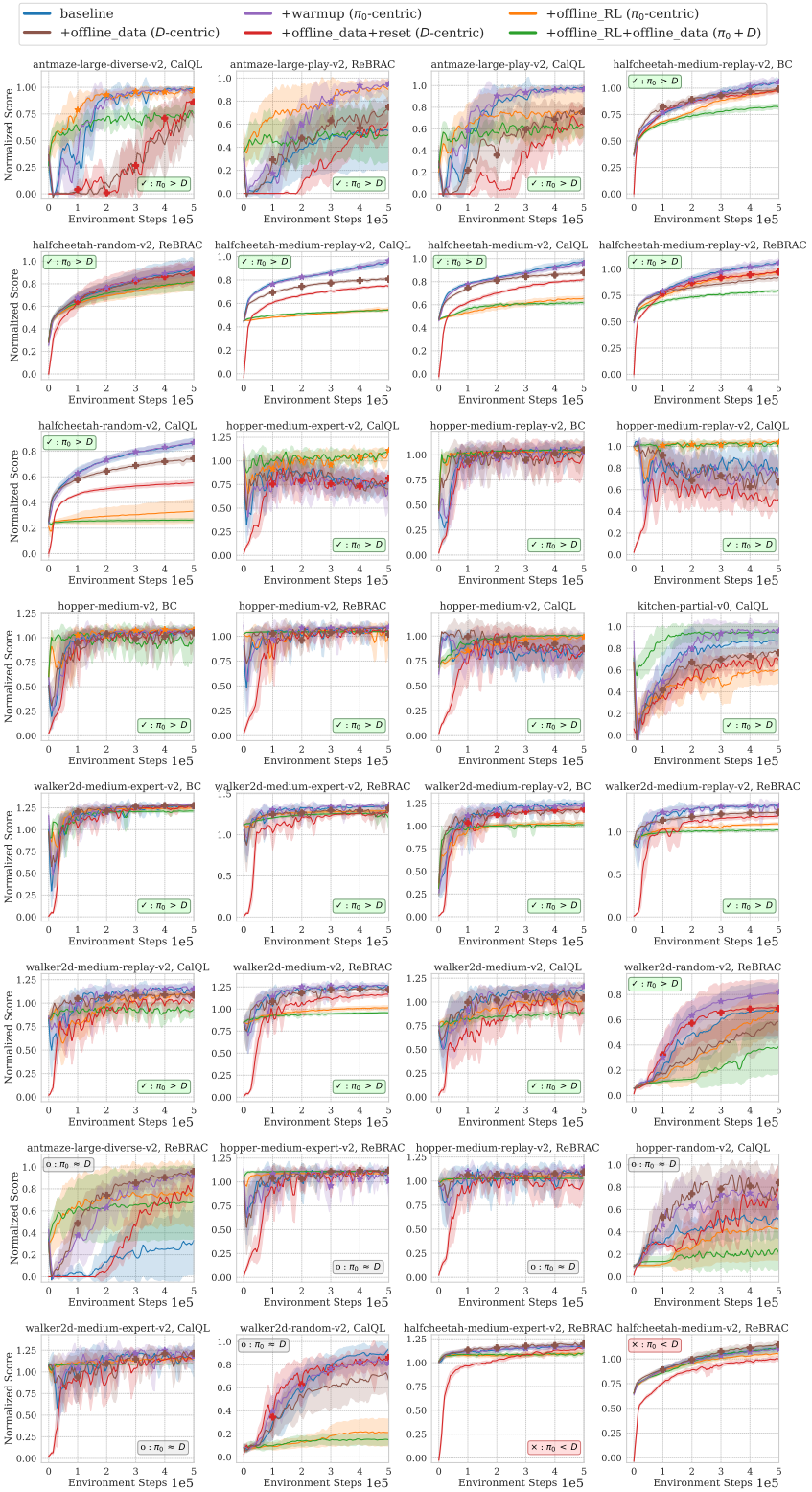

Figure 6: Full fine-tuning results in the **Superior** regime.

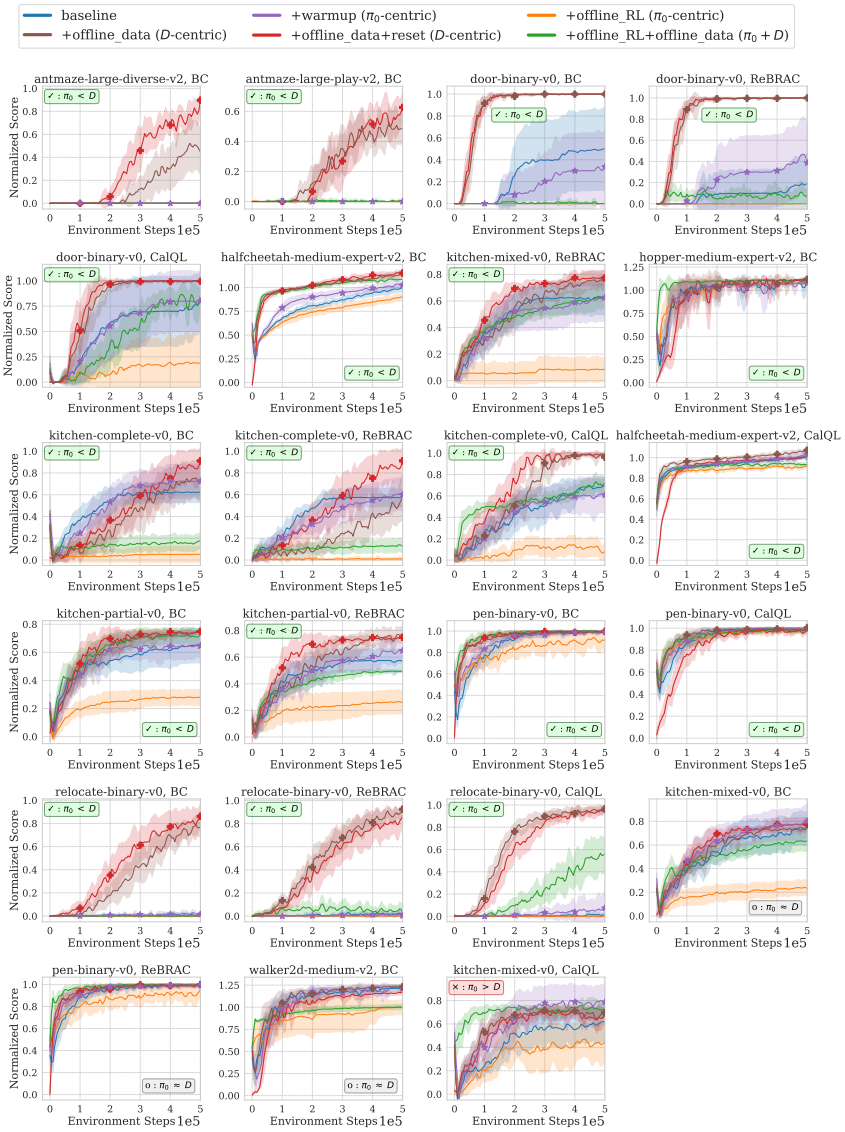

Figure 7: Full fine-tuning results in the **Inferior** regime.

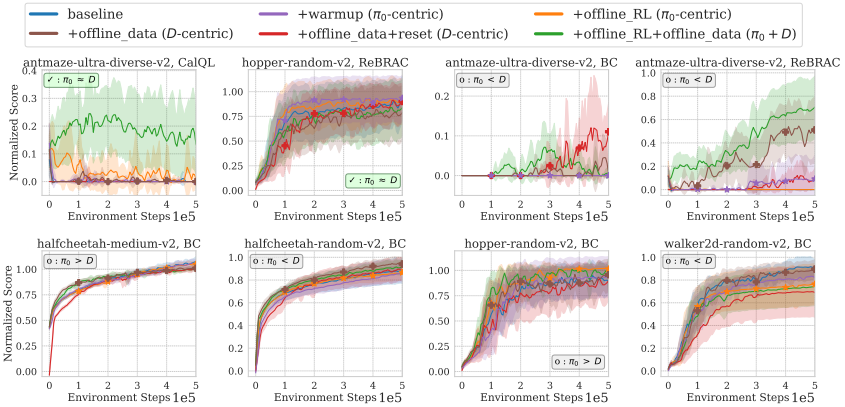

Figure 8: Full fine-tuning results in the **Comparable** regime.

C Alternative Taxonomies of Three Regimes

While our framework is defined using the returns of the pretrained agents and the dataset, it is also useful to consider alternative taxonomies based on different metrics. Below we present three such taxonomies, derived from dense reward proxy, Q-functions, and behavior-cloning performance, respectively.

C.1 Dense-Reward-Based Taxonomy

Several domains in our study (AntMaze, Adroit, and Kitchen) feature sparse rewards, where raw episode returns can be an incomplete proxy for the usefulness of prior knowledge. In particular, a pretrained policy may achieve low or zero success rate while still consistently reaching meaningful intermediate states that help exploration and fine-tuning. In such cases, classifying regimes solely based on sparse returns may underestimate the value of pretrained policy.

To address this limitation, we introduce a dense-reward-based taxonomy as a complementary regime classifier. We construct task-specific *dense reward proxies* using limited environmental knowledge (e.g., distance-to-goal, object proximity, or task progress), without assuming access to oracle dense signals. These dense rewards are used only for regime identification and analysis, not for training.

Dense reward proxy for AntMaze. We define a simple dense proxy as the negative Euclidean distance between the current agent position x and the goal location g:

$$r_{\text{maze}} = -\|x - g\|_2. \tag{6}$$

While AntMaze contains internal walls that make Euclidean distance an imperfect measure of true task progress, we intentionally avoid using shortest-path signals. This choice mirrors the sparse-reward metric by restricting access to privileged environment information, while still providing a coarse indicator of progress toward the goal.

Dense reward proxy for Adroit. We construct dense reward proxies for the Pen, Relocate, and Door tasks based on their respective success criteria. In each case, we formulate the reward as a negative squared penalty for deviations from the success thresholds.

- Pen: Success is defined by an object-goal distance $d < 0.075$ and orientation similarity $s > 0.95$. We define the proxy as:

$$r_{\text{pen}} = -\max(d - 0.075, 0)^2 - \max(0.95 - s, 0)^2. \tag{7}$$

- Relocate: Success requires the object-target distance d to be less than 0.1. The proxy is defined as:

$$r_{\text{relocate}} = -\max(d - 0.1, 0)^2. \tag{8}$$

- Door: The goal is achieved when the door position p exceeds 1.35. We penalize positions below this threshold:

$$r_{\text{door}} = -\min(p - 1.35, 0)^2. \tag{9}$$

Dense reward proxy for Kitchen. Kitchen involves a sequence of subtasks, for which we design a dense reward that incentivizes sequential completion. Let $\mathcal{T}$ be the ordered set of subtasks. For each subtask $i \in \mathcal{T}$, the reward component r_i is determined by its current status:

$$r_i = \begin{cases} 0 & \text{if subtask } i \text{ is completed,} \\ -\min\left(\frac{d_i}{d_{i,\text{init}}}, 1\right) & \text{if subtask } i \text{ is currently active,} \\ -1 & \text{if subtask } i \text{ is a future task.} \end{cases} \tag{10}$$

Here d_i is the current Euclidean distance to the subtask goal, and $d_{i,\text{init}}$ is the distance recorded when the subtask first became active. The total dense reward is the sum over all subtasks: $r_{\text{kitchen}} = \sum_{i \in \mathcal{T}} r_i$.

Accuracy on sparse-reward domains. Following our primary methodology, we employ a t-test to assess whether the performance $J(\pi_0)$ and $J(\pi_{\mathcal{D}})$ differs significantly under the dense reward metric. However, unlike the return-based formulation, we omit the significance margin δ since the widely varying scales of dense rewards across different domains. Based on this t-test, we classify each task into the three regimes. The resulting confusion matrix for the dense-reward taxonomy is presented in Table 14; it correctly identifies the regime in 16 out of 27 cases (59%). For comparison, we evaluate our proposed taxonomy within the same sparse-reward settings using margins of $\delta = 0.05$ and $\delta = 0$ (Table 15a and Table 15b, respectively). Notably, both configurations achieve an accuracy of **78%**, substantially outperforming the dense-reward-based alternative.

Table 14: Confusion matrix of fine-tuning results using **dense-reward-based taxonomy in sparse-reward domains**. Green cells: correct predictions (16/27); red cells: opposite mismatches (3/27); gray cells: adjacent mismatches (8/27). Overall, dense-reward-based taxonomy achieves 59% correct predictions with 11% opposite mismatches.

		dense-reward-based Regime		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**4**	0	1
	π_0-centric $\approx$ $\mathcal{D}$-centric	2	**0**	2
	π_0-centric $<$ $\mathcal{D}$-centric	2	4	**12**

Table 15: Confusion matrices of fine-tuning results using **raw-return-based taxonomy (ours) in sparse-reward domains**. We provide our taxonomy using a margin of $\delta = 0.05$ (Left) and $\delta = 0$ (Right). Green cells indicate correct predictions, red cells denote opposite mismatches, and gray cells represent adjacent mismatches. In both settings, our taxonomy achieves **78% accuracy** (21/27 correct) with only 4% opposite mismatches.

(a) With Margin ($\delta = 0.05$)

		Pretraining Regime (Ours)		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**4**	0	1
	π_0-centric $\approx$ $\mathcal{D}$-centric	1	**1**	2
	π_0-centric $<$ $\mathcal{D}$-centric	0	2	**16**

(b) No Margin ($\delta = 0$)

		Pretraining Regime (Ours)		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**4**	0	1
	π_0-centric $\approx$ $\mathcal{D}$-centric	2	**0**	2
	π_0-centric $<$ $\mathcal{D}$-centric	0	1	**17**

C.2 Q-Function-Based Taxonomy

Conservative offline RL aims to outperform the behavior policy that generates the offline dataset [4]. Ideally, the value function of the pretrained policy π_0 should therefore exceed that of the behavior policy $\pi_{\mathcal{D}}$ on in-dataset state-action pairs. Formally, Kostrikov et al. [20, Lemma 2] show that for in-sample Q-learning, the optimal value function satisfies

$$Q^{\pi_0^*}(s,a) \geq Q^{\pi_{\mathcal{D}}}(s,a), \quad \forall (s,a) \in \mathcal{D}, \tag{11}$$

where π_0^* denotes the optimal pretrained policy and Q^{π} is the ground-truth value function of policy π. A weaker, expectation-based version of this condition is

$$\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[Q^{\pi_0^*}(s,a)\right] \geq \mathbb{E}_{(s,a)\sim\mathcal{D}}[Q^{\pi_{\mathcal{D}}}(s,a)]\,. \tag{12}$$

Motivated by this result, we examine a taxonomy based on comparing $\mathbb{E}_{(s,a)\sim\mathcal{D}}[\hat{Q}^{\pi_0}(s,a)]$ and $\mathbb{E}_{(s,a)\sim\mathcal{D}}[Q^{\pi_{\mathcal{D}}}(s,a)]$, where $\hat{Q}^{\pi_0}$ denotes the pretrained Q-function. The intuition is that if

$$\mathbb{E}_{(s,a)\sim\mathcal{D}}[\hat{Q}^{\pi_0}(s,a)] \geq \mathbb{E}_{(s,a)\sim\mathcal{D}}[Q^{\pi_{\mathcal{D}}}(s,a)],$$

then pretraining approximately satisfies Eq. 12, placing it in the **Superior** or **Comparable** regime; otherwise, it falls into the **Inferior** regime.

In practice, we estimate $Q^{\pi_{\mathcal{D}}}(s,a)$ using the Monte Carlo return $G(s,a)$ from $\mathcal{D}$. We then perform a t-test to assess whether $\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\hat{Q}^{\pi_0}(s,a) - G(s,a)\right]$ is zero. Acceptance of the null corresponds to the **Comparable** Q-regime; a significantly positive difference indicates the **Superior** Q-regime, and a significantly negative difference indicates the **Inferior** Q-regime.

Finally, we evaluate how well this Q-based taxonomy aligns with the fine-tuning results. The corresponding confusion matrix is reported in Table 16. It achieves 32 out of 63 correct predictions (**51%**), 19 opposite mismatches (30%), and 12 adjacent mismatches (19%). Although this accuracy is substantially better than random guessing (33%), it remains noticeably lower than the performance of our framework (**71%**).

C.3 Behavior-Cloning-Based Taxonomy

Instead of relying on the return of the behavior policy $J(\pi_{\mathcal{D}})$, one can define a taxonomy based on the performance of a behavior-cloned policy π_{BC} learned on the offline dataset $\mathcal{D}$.

Table 16: Confusion matrix of fine-tuning results using **Q-based taxonomy**. Green cells: correct predictions (32/63); red cells: opposite mismatches (19/63); gray cells: adjacent mismatches (12/63). Overall, Q-based taxonomy achieves 51% correct predictions with 30% opposite mismatches.

		Pretraining Q-based Regime		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**23**	0	4
	π_0-centric $\approx$ $\mathcal{D}$-centric	8	**0**	3
	π_0-centric $<$ $\mathcal{D}$-centric	15	1	**9**

Similar to our taxonomy, we use a t-test with a margin of $\delta = 0.05$ to assess whether $J(\pi_{\text{BC}})$ and $J(\pi_{\mathcal{D}})$ differ significantly, thereby identifying the three corresponding regimes. The corresponding confusion matrix is shown in Table 17. The BC-based taxonomy achieves 26 out of 63 correct predictions (**41%**), 3 opposite mismatches (5%), and 34 adjacent mismatches (54%). While better than random guessing, this result remains noticeably lower than the performance of our framework (**71%**).

Table 17: Confusion matrix of fine-tuning results using **BC-based taxonomy**. Green cells: correct predictions (26/63); red cells: opposite mismatches (34/63); gray cells: adjacent mismatches (3/63). Overall, BC-based taxonomy achieve 41% correct predictions with 5% opposite mismatches.

		BC-based Regime		
		Superior	**Comparable**	**Inferior**
Fine-tune	π_0-centric $>$ $\mathcal{D}$-centric	**17**	10	0
	π_0-centric $\approx$ $\mathcal{D}$-centric	4	**6**	1
	π_0-centric $<$ $\mathcal{D}$-centric	3	19	**3**

D Extended Mechanistic Analysis

In this section, we extend our mechanistic analysis to a broader range of experimental settings. Figures 9 and 10 present the extended results for the **Superior** and **Inferior** regimes, respectively. Consistent with our main text discussion, these results illustrate that fine-tuning without explicit offline data anchoring leads to diverging Q-values and exploding TD errors on the offline dataset. While a similar pattern also occurs in the **Superior** regime, it is significantly less severe.

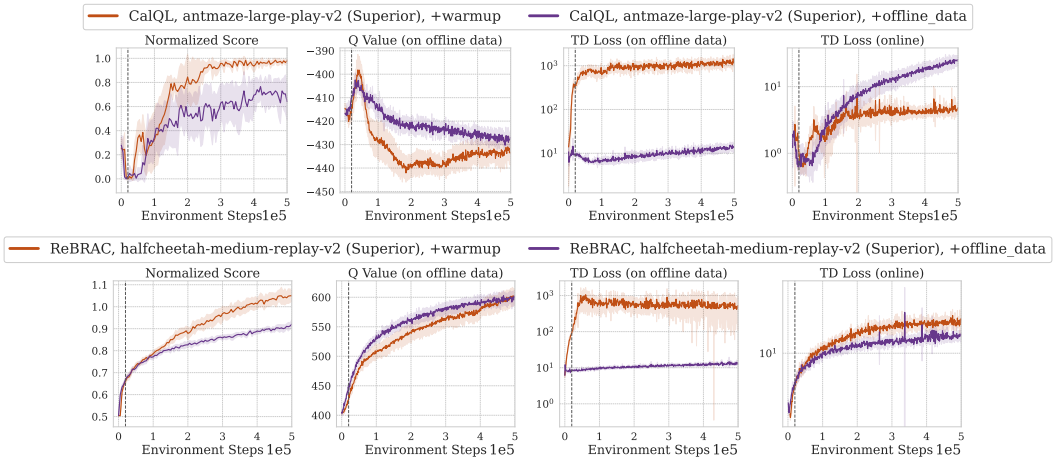

Figure 9: Extended mechanistic analysis in the **Superior** regime.

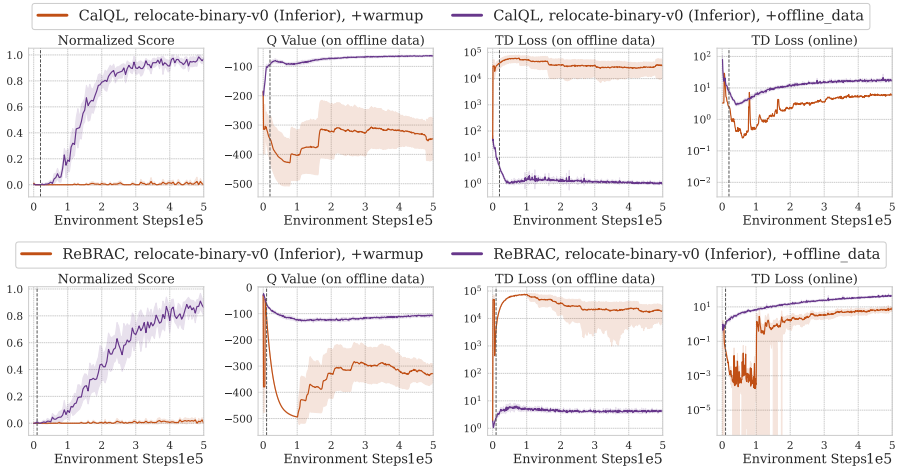

Figure 10: Extended mechanistic analysis in the **Inferior** regime.