ASYMMETRIC ACTOR-CRITIC PRETRAINING FOR EFFICIENT FLOW-BASED ONLINE RL FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline-to-online (O2O) reinforcement learning employs synchronized, symmetric offline training of the actor and critic for downstream online fine-tuning. Our analysis reveals that this symmetric training process impairs performance during online fine-tuning: while the policy network acquires a sampling prior through offline training to ensure efficient online fine-tuning, the value network may lose plasticity due to prolonged offline training, thereby failing to rapidly adapt to new environments when faced with distribution shifts between the offline and online settings. Based on this distinction, we identify a trade-off mechanism regarding critic offline training and investigate the impact of offline data coverage on the actor's online performance. Our research reveals an asymmetry in offline training: the critic is highly sensitive to the extent of offline training, whereas the actor relies more heavily on the coverage of effective regions within the offline data. Leveraging this asymmetry, we propose **C**ritic-**F**ree **P**retraining—a simple, asymmetric training framework widely applicable to various algorithms. CFP involves pretraining the actor without a critic, subsequently initializing a fresh critic with a joint offline warm-up before entering the online interaction phase. Across 8 challenging manipulation domains, spanning OGBench and Robomimic, CFP achieves performance comparable to or superior to naive O2O methods.

1 INTRODUCTION

Offline-to-online reinforcement learning leverages static datasets to prepare agents for subsequent learning through environmental interaction (Levine et al., 2020). In actor-critic methods, a common practice is to optimize the policy and value function offline and then use the resulting parameters for online fine-tuning. This procedure transfers two different objects: a policy that determines which states and actions the agent visits, and a critic that guides policy improvement. Whether these objects should receive the same offline training schedule is therefore a consequential design question.

Directly transitioning offline-trained models to the online phase often results in severe performance drops and inefficient fine-tuning. Prevailing explanations attribute this difficulty primarily to the explicit pessimism introduced during offline critic training (Nakamoto et al., 2023; Ball et al., 2023; Xiao et al., 2025). Traditional offline RL methods—which typically employ Gaussian policies—rely on conservative penalty terms to constrain critic estimates for out-of-distribution (OOD) actions, thereby preventing overestimation (Kumar et al., 2020). While unlearning this explicit pessimism is a recognized bottleneck, it does not fully explain the O2O transition problem. For instance, in Flow-based RL methods such as Flow Q-learning (Park et al., 2025b), the critic is trained via standard Bellman regression. Yet, even in these penalty-free settings, an inherited critic often adapts slowly and hinders policy improvement when the training distribution shifts online. This strongly indicates that the O2O bottleneck stems from a deeper structural issue in how pretrained critics behave, rather than merely from explicit conservatism.

To uncover this underlying issue, we reconsider the distinct benefits and adverse impacts of offline training on the actor and the critic. We identify a trade-off in the duration of offline critic training: both insufficient and excessive training can reduce the efficiency of subsequent online fine-tuning. In contrast, the actor primarily benefits from offline training by learning a prior policy, a property that remains crucial for downstream exploration.

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.

Motivated by these observations, we propose a novel O2O training paradigm, Critic-Free Pretraining (CFP), an asymmetric training workflow which deliberately customizes the preparation of the actor and the critic. By strategically managing the offline training schedules, CFP prevents the critic from over-anchoring to the offline dataset while preserving the actor's exploratory advantages. Furthermore, we theoretically examine how the pretrained actor's coverage of the demonstrator trajectories influences downstream fine-tuning. Our analysis gives a conditional return guarantee for critic-free actor pretraining, making the implementation of CFP a method that is both theoretically grounded and highly effective at the empirical level. Our main contributions are listed below:

- **A systematic analysis of the offline training mechanisms for the actor and the critic.** We identify a fundamental cause of low fine-tuning efficiency in O2O models: the loss of plasticity in critic networks subjected to excessive offline training. Ablation studies regarding the extent of offline training confirm the tradeoff mechanism inherent in offline critic training, which is not observed during actor pretraining. This highlights the fundamental requirement of the actor-critic architecture for an asymmetric training process.
- **A theory regarding the impact of pretrained actors on downstream fine-tuning.** We derive a bound on the conditional return that links the performance of downstream policies to the extent to which the pretrained actor covers the states visited by high-return policies, thereby theoretically quantifying the value of critic-free actor pretraining for downstream fine-tuning. Comparative experiments conducted on datasets with varying levels of high-quality trajectory coverage are consistent with the coverage perspective suggested by the theory.
- **An asymmetric training paradigm.** We present CFP, a novel paradigm for efficient online fine-tuning leveraging the training asymmetry between the actor and the critic. CFP combines critic-free actor pretraining with joint offline warm-up and subsequent online fine-tuning. Since it involves modifications solely at the workflow level, it can be widely applied to flow algorithms. We conduct extensive experiments across 28 manipulation tasks spanning 8 domains. CFP consistently demonstrates superior efficiency during offline stage and achieves online fine-tuning performance that is either comparable to or superior to that of O2O across all domains.

2 PRELIMINARIES

Flow Q-learning Flow-based policies are generally trained via Flow Matching (Lipman et al., 2023; 2024; Liu et al., 2022; McAllister et al., 2026; Yang et al., 2026). The objective is to minimize the loss function:

$$L_{Flow}(\theta) = \mathbb{E}_{\substack{x_0\sim\mathcal{N}(0,I_d),\\(x_1=a,s)\sim\mathcal{D},\\t\sim\text{Unif}([0,1])}}[\|v_\theta(t,s,x_t) - (x_1 - x_0)\|_2^2] \tag{1}$$

where $\mathcal{D}$ denotes the static dataset and

$$x_t = (1-t)\cdot x_0 + t\cdot x_1, \qquad x_0 \sim \mathcal{N}(0, I). \tag{2}$$

Flow Q-learning (FQL) is a strong flow-based algorithm. It applies Equation 1 as the update target of the velocity network. In actual implementation, rather than learning the velocity field directly, FQL uses a single-step network as the actor to distill the final actions produced by the velocity network. This allows actions that previously required multiple computations to be obtained in a single pass, significantly improving efficiency and training stability (Park et al., 2025b; Zhang et al., 2026a;b; Li & Levine, 2026; Li et al., 2026; Doo et al., 2026). The overall loss function of FQL is as below:

$$L_{Total}(\theta,\omega) = L_{Flow}(\theta) + L_\pi(\omega) \tag{3}$$

where

$$\mathcal{L}_\pi(\omega) = \underbrace{\mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\omega(\cdot|s)}\big[-Q_\phi(s,a)\big]}_{\text{Q loss}} + \underbrace{\alpha\,\mathbb{E}_{s\sim\mathcal{D},z\sim\mathcal{N}(0,I^d)}\big[\|\mu_\omega(s,z) - \mu_\theta(s,z)\|_2^2\big]}_{\text{BC loss}}. \tag{4}$$

FQL employs standard Bellman regression as the update target for its critic; unlike the penalties on out-of-distribution (OOD) actions commonly found in Gaussian offline RL (Kumar et al., 2020), the critic update process in FQL incorporates no explicit conservative value regularizer, as shown below.

$$L(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D},a'\sim\pi_\omega(\cdot|s')}[(Q_\phi(s,a) - (r + \gamma Q_{\bar{\phi}}(s',a')))^2] \tag{5}$$

FQL can be applied in both offline and O2O settings. In the O2O scenario, the actor and critic are trained synchronously on the dataset using the aforementioned equations; at the onset of the online phase, the model utilizes the pretrained actor and critic. This enables the model to undergo online fine-tuning while retaining the prior knowledge embedded in the dataset.

3 Efficient Online Fine-tuning via CFP

To understand why transferring an offline-trained actor-critic pair often leads to inefficient online fine-tuning, we must isolate the behavior of the transferred critic. We design a diagnostic baseline: instead of co-training the actor and critic synchronously throughout the entire offline phase (as in standard O2O), we pretrain the actor alone. Then, we initialize a fresh critic and expose it to only a brief joint offline warm-up just before online interaction. We denote this asymmetric diagnostic approach as Critic-Free Pretraining (CFP). By comparing standard O2O with this CFP variant, we can uncover the hidden bottlenecks introduced by prolonged offline critic training.

3.1 Why does a pretrained critic hinder online fine-tuning?

Loss of plasticity and critic mismatch Neural networks may gradually lose the ability to rapidly adapt to new data distributions and changes during prolonged training—a phenomenon known as "loss of plasticity" or "primacy bias" (Nikishin et al., 2022; Lyle et al., 2023; Abbas et al., 2023). We hypothesize that prolonged offline optimization causes the critic to specialize in the offline state occupancy measure and Bellman targets, while simultaneously reducing its ability to adapt to rapidly changing targets in an online environment. Distribution shift does not inherently cause a loss of plasticity; rather, it exposes the limited adaptability developed during offline training. A newly initialized critic avoids this offline specialization, thereby potentially adapting more rapidly to online environments.

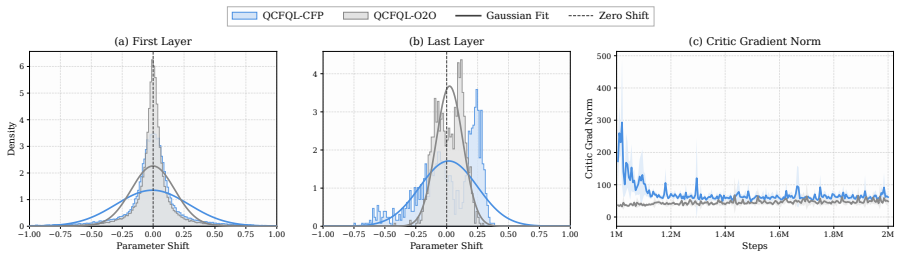


Figure 1: Evaluation of the parameter shifts of the O2O and CFP critics before and after online training; specifically, we selectively monitor the parameter shifts—calculated as $\sigma_{after} - \sigma_{before}$—of the first and last layers of the MLP networks. We also monitor the gradient norm of the critic loss throughout the online training process. *Left:* parameter shift of the first layer. *middle:* parameter shift of the last layer. *Right:* gradient norm during the online stage.

To verify our hypothesis, we measure the parameter shifts of the O2O and CFP critics before and after online training. We separately inspect the input layer, which determines how state-action inputs are embedded into the critic representation, and the output layer, which maps this representation to scalar Q-values (Yosinski et al., 2014; Sokar et al., 2023). Figure 1 shows that, following online training, the parameter shift in the CFP critic is significantly greater than that in the O2O critic. This indicates that online training has a more pronounced impact on the CFP critic, suggesting a superior ability to adapt to new environments. Figure 1 also illustrates that the CFP's critic gradient norm fluctuates significantly during the initial online phase and remains consistently higher than that of O2O throughout the process. This indicates that the fresh critic rapidly adapts to shifting data distributions and that this capability does not diminish over online training.

Merely demonstrating a decline in the plasticity of the critic network is insufficient to prove that offline training is detrimental to online fine-tuning. After all, if the critic has already converged to the optimal point during offline training, there is no need for online training. Datasets like OGBench (Park et al., 2025a), characterized by long horizons and sparse rewards, comprise agent trajectories

and data from completing various tasks within the same domain; their purpose is to evaluate whether a model can activate task-specific problem-solving capabilities when interacting with an online environment. In other words, offline training data does not necessarily facilitate online fine-tuning and may even prove counterproductive. Furthermore, the inclusion of numerous failure trajectories prevents the model from simply learning highly efficient paths during the offline phase. These biases hinder the critic from learning the true values of state-action pairs encountered in the subsequent real-world environment during the offline stage itself. We call this phenomenon critic mismatch and design a toy example to visualize the critic mismatch produced during offline training, see Appendix F.

The simultaneous occurrence of the critic's mismatch and reduced plasticity prevents it from efficiently adapting during online interaction, ultimately resulting in poor online performance.

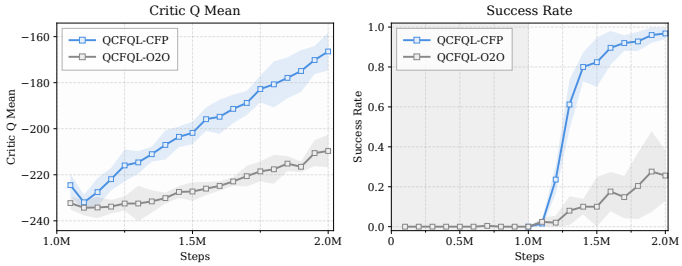


Figure 2: Critic Q-mean estimates and success rates on Cube Triple task 4. *Left:* at the beginning of online fine-tuning, the output level of the CFP critic is nearly identical to that of the O2O critic; however, as fine-tuning progresses, the CFP critic's output rises rapidly, indicating that the CFP critic maintains high plasticity. *Right:* CFP achieves a success rate of nearly 100% on this challenging task, whereas O2O demonstrates a very low success rate.

Figure 2 further illustrates the mean Q-values output by the CFP critic and the O2O critic, along with the corresponding agent success rate (SR). It is evident that the outputs of the two critics are very close at the onset of online training; however, as fine-tuning progresses, the Q-mean of the CFP critic rises rapidly, whereas that of the O2O critic increases slowly. This further confirms that the low fine-tuning efficiency of O2O flow RL is not due to critic pessimism, but is instead a consequence of prolonged offline training under distribution shifts.

3.2 The CFP workflow

Our diagnostic experiments reveal a critical flaw in the standard O2O paradigm: prolonged offline training causes the critic to suffer from a severe loss of plasticity, crippling its ability to adapt online. In contrast, limiting the critic's offline exposure to a brief warm-up preserves its plasticity while still providing necessary initial calibration. Motivated by this finding, we elevate our diagnostic baseline into a generalized training paradigm, formally introduced as Critic-Free Pretraining (CFP).

As shown in Algorithms 1 and 2, CFP includes three main stages in its workflow.

- **Critic-free actor pretraining.** We initialize the actor and train it on the offline dataset. During this stage, the actor follows the behavioral cloning (BC) update target—that is, by omitting the Q-loss component from Equation 4 (Bhatt et al., 2026; Torabi et al., 2018; Foster et al., 2024; Sasaki & Yamashina, 2020).
- **Offline warm-up.** After actor pretraining, we initialize a fresh critic and its target network. We then optimize the policy and critic on the offline dataset using the objectives of the base algorithm.
- **Online fine-tuning.** We continue from the warmed-up policy and critic, initialize the replay buffer with offline transitions, and apply the base algorithm's online updates.

Throughout this paper, pretraining refers specifically to the initial training of a freshly initialized actor without critic guidance. Warm-up denotes the subsequent joint offline optimization of the actor and a newly initialized critic. Offline training denotes all optimization performed on the static dataset before online interaction.

Algorithm 1 Traditional O2O

Init: Critic Q_ϕ, Policy π_θ, Buffer $\mathcal{B} \leftarrow \mathcal{D}$
▷ Offline Stage
for $\ell = 1 \dots L_{\text{off}}$ **do**
  Sample batch $\{(s, a, r, s')\} \sim \mathcal{B}$
  Update π_θ via offline actor loss
  Update Q_ϕ via TD loss
end for
▷ Online Stage
for $\ell = 1 \dots L_{\text{on}}$ **do**
  Collect $(s, a, r, s') \sim \pi_\theta$, append to $\mathcal{B}$
  Sample batch $\{(s, a, r, s')\} \sim \mathcal{B}$
  Update π_θ via online actor loss
  Update Q_ϕ via TD loss
end for

Algorithm 2 CFP (Ours)

Init: Policy π_θ, Buffer $\mathcal{B} \leftarrow \mathcal{D}$
▷ Pretraining Stage (Critic-Free)
for $\ell = 1 \dots L_{\text{off}}$ **do**
  Sample batch $\{(s, a, r, s')\} \sim \mathcal{B}$
  Update π_θ via BC Flow loss
end for
▷ **Warm-up Stage**
Init fresh critic Q_ϕ from scratch
for $\ell = 1 \dots L_{\text{warm-up}}$ **do**
  Sample batch $\{(s, a, r, s')\} \sim \mathcal{B}$
  Update π_θ via online actor loss
  Update Q_ϕ via TD loss
end for
▷ Online Stage
  Same as the O2O workflow.

Offline training improves the efficiency with which RL models learn effective policies in complex environments (Levine et al., 2020; Ball et al., 2023; Zhang et al., 2026b). However, we have shown that offline training can actually be detrimental to the critic. This anomaly raises two questions: Do we still need offline training? And if the answer is yes, what exactly do we gain from it?

3.3 Does the critic still require offline training?

We evaluate the effectiveness of CFP on a massive dataset (100M). Unlike in the smaller-scale environments (1M) used commonly, CFP training is unstable and exhibits high variance when warm-up steps are set to 0 or 10k, as shown in Figure 3. While CFP achieves excellent performance with a 1M-step warm-up, this does not imply that increasing warm-up steps further enhances fine-tuning efficiency; indeed, when we employ an extended warm-up of 10M steps, we observe a collapse in performance during the online phase. This phenomenon indicates that offline training continues to play a crucial role in accelerating fine-tuning, yet this benefit coexists with potential offline-related risks; while offline calibration enables the critic to provide a reasonably accurate estimate of true Q-values, prolonged offline training exposes the neural network to overfitting issues exacerbated by distribution shift.

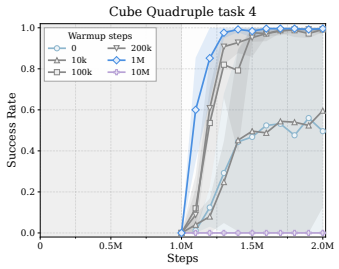


Figure 3: The impact of varying offline warm-up steps on subsequent fine-tuning performance: The model achieves optimal fine-tuning performance with 1M warm-up steps of QCFQL-nstep-CFP; offline training durations that are either too short or too long lead to training instability or even complete collapse.

Therefore, we conclude that excessively long offline training is the reason why the online fine-tuning efficiency of O2O is significantly lower than that of CFP. However, when dealing with large volumes of data, offline training remains a key means of facilitating fine-tuning.

4 Understanding Critic-Free Actor Pretraining

While Section 3 demonstrates that the critic benefits from delayed and abbreviated offline training, CFP introduces a radical change for the actor: it is pretrained entirely without critic guidance. A natural skepticism arises: Does omitting value-guided policy improvement during offline pretraining cripple the agent's downstream capabilities?

CFP uses supervised actor pretraining to acquire a behavioral prior before value-guided improvement. We analyze when this prior is sufficient to support downstream learning. The key requirement is that imitation preserves the task-relevant coverage of the offline behavior distribution, rather than that the pretrained actor already maximizes the deployment reward (Wagenmaker et al., 2026; Uchendu et al., 2023; Li et al., 2023).

Setup. Consider a finite-horizon MDP with horizon H, discount $\gamma \in (0, 1]$, and bounded rewards $r_{\min} \le r_h(s, a) \le r_{\max}$. Let β denote the offline behavior policy, μ the BC-pretrained actor, $\pi^\dagger$ a comparator, and $\widehat{\pi}$ the policy returned by downstream learning. All policies are evaluated from a common initial-state distribution ρ. Write d_h^π for state occupancy and

$$B_h := (r_{\max} - r_{\min}) \sum_{j=0}^{H-h} \gamma^j. \tag{6}$$

Assumption 1. *For each step h, suppose a region G_h satisfies the restricted measure inequality*

$$d_h^{\pi^\dagger}|_{G_h} \le C_h d_h^\beta|_{G_h}, \qquad m_h := d_h^{\pi^\dagger}(G_h^c). \tag{7}$$

The coefficient C_h measures frequency mismatch within the covered region, whereas m_h measures comparator mass absent from this guarantee. This assumption allows unsuccessful and task-irrelevant demonstrations, provided the relevant part of the behavior distribution retains sufficient coverage.

We measure imitation-induced state-distribution error using squared Hellinger distance (Foster et al., 2024),

$$\zeta_h = D^2_{\mathrm{Hel}}(d_h^\beta, d_h^\mu), \qquad D^2_{\mathrm{Hel}}(P, Q) = \int (\sqrt{dP} - \sqrt{dQ})^2. \tag{8}$$

Lemma 1. *Under Assumption 1, for any $\tau \in (0, 1)$, there exist measurable regions $\widetilde{G}_h \subseteq G_h$ such that*

$$d_h^{\pi^\dagger}|_{\widetilde{G}_h} \le \frac{C_h}{\tau} d_h^\mu|_{\widetilde{G}_h}, \qquad d_h^{\pi^\dagger}(\widetilde{G}_h^c) \le m_h + \frac{C_h \zeta_h}{(1 - \sqrt{\tau})^2}. \tag{9}$$

The lemma turns distribution fitting into retained coverage: accurate imitation preserves a bounded density ratio after discarding a controlled amount of comparator mass. Allowing this discarded mass is essential, because a small average distribution error does not generally imply a uniformly bounded density ratio over the entire state space.

Assumption 2. *The downstream learner controls its local decision regret under the pretrained actor's occupancy, simultaneously for all h, with probability at least $1 - \delta$:*

$$\mathbb{E}_{d_h^\mu}[\Delta_h(s)] \le \varepsilon_h, \qquad \Delta_h(s) = \sup_a Q_h^{\widehat{\pi}}(s, a) - \mathbb{E}_{a \sim \widehat{\pi}_h(\cdot|s)} Q_h^{\widehat{\pi}}(s, a). \tag{10}$$

Bounded rewards ensure that $0 \le \Delta_h(s) \le B_h$. Obtaining a small ε_h is a separate condition on downstream learning, depending on value estimation and policy optimization.

Theorem 1. *Under Assumptions 1 and 2, for every $\tau \in (0, 1)$, on the downstream learning event,*

$$J(\pi^\dagger) - J(\widehat{\pi}) \le \sum_{h=1}^{H} \gamma^{h-1} \left[\frac{C_h}{\tau} \varepsilon_h + B_h m_h + \frac{B_h C_h \zeta_h}{(1 - \sqrt{\tau})^2} \right]. \tag{11}$$

The three terms account for downstream decision error, coverage missing from the offline behavior distribution, and additional coverage loss due to imitation. No pretrained critic is required to control the third term: it is a distribution-learning problem. Value-guided improvement remains necessary downstream to control the first term. Thus, the theorem provides a conditional sufficiency statement for critic-free actor pretraining. For the proofs, see Appendix A.

To empirically verify the above theorem, we construct several datasets while we keep the dataset size fixed and collect all trajectories using the same expert actor. The only difference lies in the number of initial seeds used to construct each dataset. For instance, single-reset-seed implies that

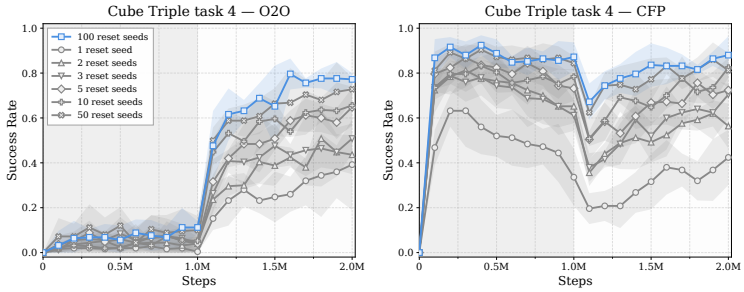


Figure 4: Fine-tuning performance of QCFQL after pretraining on datasets with varying diversity. N-reset-seeds indicates that the dataset consists of trajectories sampled by an expert actor across environments with N different reset seeds. *Left:* O2O scenario. *Right:* CFP scenario. In both scenarios, models pretrained on datasets with higher expert trajectory coverage demonstrate better performance during online fine-tuning.

all trajectories are collected using a single seed, whereas 100-reset-seeds indicates that the data is gathered from 100 seeds. Since variations in seeds lead to slight differences in details, such as initial positions and cube locations, trajectories collected from multiple seeds naturally cover a broader range of the expert's state space. Figure 4 demonstrates that an actor with high state coverage significantly outperforms one with low state coverage during the online phase. These results are consistent with the coverage perspective in Theorem 1, confirming that the actor is sensitive to the quality of pretraining data, and its impact on downstream fine-tuning is positively correlated with the extent of high-quality trajectory coverage.

5 Experiments

We conduct extensive experiments on manipulation tasks under different environments. We further conduct ablation experiments, providing solid evidence for the implementation of CFP. All the curves and shaded regions in this paper represent the mean and 95% confidence interval across 5 random seeds.

Settings. We conduct our experiments on 8 sparse reward domains, including 5 domains from OGBench (Park et al., 2025a): `Cube-Double/Triple/Quadruple`, `Puzzle-4×4`, `Scene`, and 3 tasks in Robomimic (Mandlekar et al., 2021): `Lift`, `Can` and `Square`. For details about the benchmark and dataset, see Appendix B.

Baselines. We select 4 representative flow-based algorithms as our baselines, including **(1) FQL** (Park et al., 2025b), **(2) QC** (Ghasemipour et al., 2021; Li et al., 2026), **(3) QCFQL** (Li et al., 2026), **(4) QCFQL-nstep** (Zhang et al., 2026b). We further implement CFP on all of them and compare the online performance among them. For more details, see Appendix D.

5.1 Results and Q&As

Q: How effective is CFP for online fine-tuning?

A: Compared to O2O, CFP demonstrates comparable or superior fine-tuning performance across **all** domains.

Figure 5 reports the main comparison between CFP and O2O. For OGBench, we select `Task 4` from each domain as the representative task. CFP achieves significantly more efficient fine-tuning results than O2O across nearly all domains. For instance, on `Double Task 4`, FQL-CFP achieves an SR of 0.85, an improvement of nearly **11**-fold compared to FQL-O2O. On `Triple Task 4`, CFP successfully fine-tunes QCFQL, which originally achieves an SR of only 0.27, to 0.97, representing a **3.5**-fold increase. Overall, CFP's average fine-tuning performance on OGBench is 1.35× that of O2O. This result is achieved even when including domains like `Scene`, where all algorithms

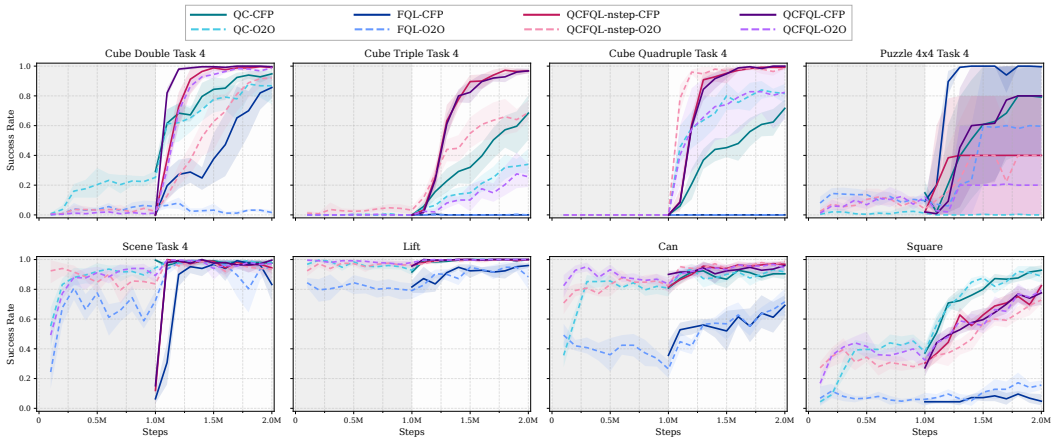


Figure 5: Performance across 8 domains in OGBench and Robomimic.

attain perfect scores. On Robomimic, CFP achieves performance comparable to O2O across all domains. For full results, see Appendix E.2.

Q: How efficient is CFP for offline training?

A: Compared to O2O, CFP demonstrates higher efficiency during offline training.

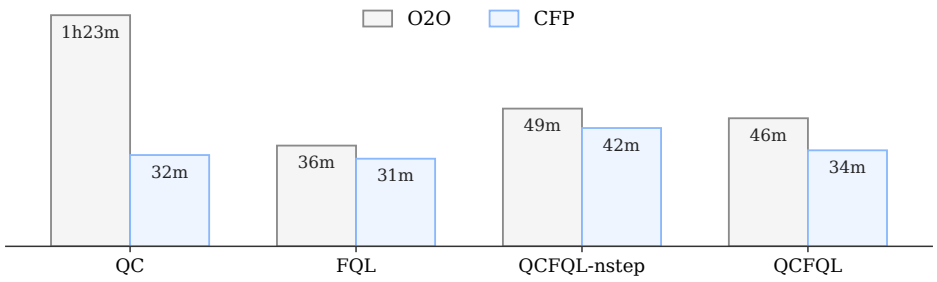


Figure 6: Comparison of GPU-hours required for offline training between CFP and O2O: CFP requires significantly fewer GPU-hours than O2O. All training is performed on NVIDIA A800 GPUs.

As shown in Figure 6, CFP demonstrates exceptional efficiency during the offline training phase (pretraining + warm-up). Overall, the total offline training time for the 4 CFP variants is only 0.65× that of O2O, significantly reducing the cost and overhead of offline training. In particular, QC (Best-of-N) evaluates 32 or even more action chunks with the critic ensemble. Removing these critic updates during actor pretraining reduces the measured training time from 83 to 32 minutes in the reported setting, corresponding to a 2.59× speedup.

Q: Do we still need offline training?

A: Yes, offline training still plays a significant role in CFP workflow.

We conduct ablation studies on CFP, completely removing offline training (offline pretraining + warm-up) and additionally include RLPD variants for each algorithm (Ball et al., 2023).

As shown in Figure 7, across all four domains, the online fine-tuning performance follows the order: CFP > CFP w/o offline > RLPD. The first inequality further demonstrates that offline training helps improve fine-tuning efficiency, while the second confirms that offline data may impede online fine-tuning (Zhou et al., 2025). All variants achieve fine-tuning performance equal to or superior to that of O2O, intuitively illustrating the detrimental impact of prolonged offline pretraining on the critic (given that none of these three variants uses prolonged critic pretraining).

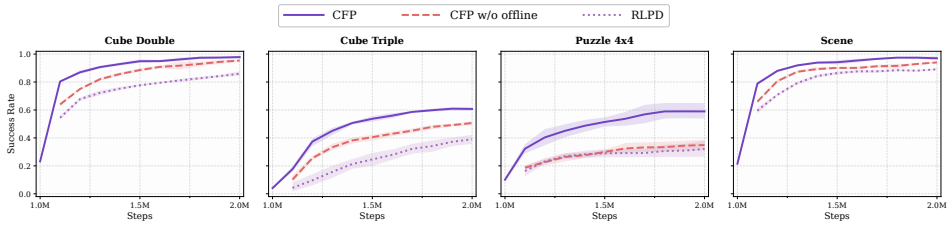


Figure 7: Aggregated performance of 4 algorithms across 4 OGBench domains. Performance across all domains follows the pattern: CFP $>$ CFP w/o offline $>$ RLPD.

Q: Is CFP sensitive to actor pretraining?

A: Relatively no, but a certain degree of actor pretraining is still recommended.

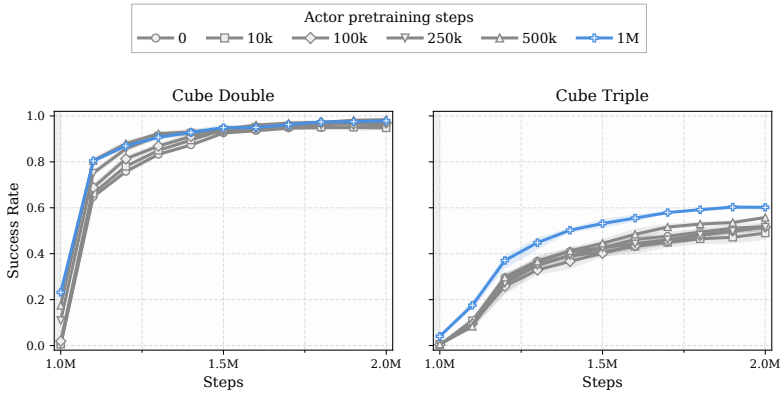


Figure 8: Aggregated performance of 4 algorithms across 2 OGBench domains. Online performance demonstrates strong robustness regarding the degree of actor pretraining.

We are also interested in whether CFP is sensitive to actor pretraining. We vary the number of training steps for the pretrained actors. As shown in Figure 8, overall, CFP remains relatively robust to actor pretraining degree. While this implies that actor does not require the same strict control over the extent of pretraining as critic does, the final fine-tuning performance remains positively correlated with the number of pretraining steps within a small range. This is one aspect of the asymmetric behavior observed between actor and critic offline training. We provide more interesting ablation experiments worth discussing in Appendix E.3.

6 Conclusion and Discussion

In this article, we attempt to achieve more efficient online fine-tuning using an asymmetric CFP training framework. Several questions warrant further investigation: Is CFP applicable to accelerating real robot fine-tuning? Do the training mechanisms related to value network observed with CFP still hold true as the network scales up? Furthermore, our work explores how to achieve efficient fine-tuning at the workflow level and offers new insights for future algorithm design, though we do not propose a new algorithm ourselves. We consider the design of new algorithms based on the inherent asymmetry between the actor and the critic to be a key focus for future research.

AI USE STATEMENT

In this work we only use generative AI tools to polish the language to improve readability. We have reviewed all AI-assisted work. We have carefully cross-checked all AI-edited text against our original drafts sentence-by-sentence to ensure that technical meanings remained completely accurate and no unverified claims or ideas were introduced. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our experiments are conducted exclusively on public benchmarks and datasets. This work does not involve human subjects or sensitive data and presents no foreseeable direct ethical concerns.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, additional implementation details, hyperparameters, and experimental results are provided in the Appendix. All experiments leverage public benchmarks and datasets. The anonymous code is available at: `https://anonymous.4open.science/r/CFP`

REFERENCES

Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C Machado. Loss of plasticity in continual deep reinforcement learning. In *Conference on lifelong learning agents*, pp. 620–636. PMLR, 2023.

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

Dwait Bhatt, Shih-Chieh Chou, and Nikolay Atanasov. Rainbow-demorl: Combining improvements in demonstration-augmented reinforcement learning. *arXiv preprint arXiv:2603.27400*, 2026.

JaeHyeok Doo, Byeongguk Jeon, Seonghyeon Ye, Kimin Lee, and Minjoon Seo. Q-flow: Stable and expressive reinforcement learning with flow-based policy. In *International Conference on Machine Learning*, 2026.

Dylan J Foster, Adam Block, and Dipendra Misra. Is behavior cloning all you need? understanding horizon in imitation learning. *Advances in Neural Information Processing Systems*, 37:120602–120666, 2024.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062, 2019.

Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pp. 3682–3691, 2021.

Siyuan Guo, Lixin Zou, Hechang Chen, Bohao Qu, Haotian Chi, Philip S Yu, and Yi Chang. Sample efficient offline-to-online reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering*, 36(3):1299–1310, 2023.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

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.

Gen Li, Wenhao Zhan, Jason D Lee, Yuejie Chi, and Yuxin Chen. Reward-agnostic fine-tuning: Provable statistical benefits of hybrid reinforcement learning. *Advances in Neural Information Processing Systems*, 36:55582–55615, 2023.

Qiyang Li and Sergey Levine. Q-learning with adjoint matching. In *International Conference on Learning Representations*, 2026.

Qiyang Li, Zhiyuan Paul Zhou, and Sergey Levine. Reinforcement learning with action chunking. *Advances in Neural Information Processing Systems*, 38:55518–55553, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, pp. 23190–23211. PMLR, 2023.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *International Conference on Learning Representations*, volume 2026, pp. 36352–36372, 2026.

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

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.

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*, pp. 16828–16847, 2022.

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

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning*, 2025b.

Fumihiro Sasaki and Ryota Yamashina. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*, 2020.

Yongjae Shin, Jeonghye Kim, Whiyoung Jung, Sunghoon Hong, Deunsol Yoon, Youngsoo Jang, Geon-Hyeong Kim, Jongseong Chae, Youngchul Sung, Kanghoon Lee, and Woohyung Lim. Online pre-training for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, 2025.

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

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

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*, pp. 34556–34583, 2023.

Andrew Wagenmaker, Perry Dong, Raymond Tsao, Chelsea Finn, and Sergey Levine. Posterior behavioral cloning: Pretraining BC policies for efficient RL finetuning. In *International Conference on Machine Learning*, 2026.

Wei Xiao, Jiacheng Liu, Zifeng Zhuang, Runze Suo, Shangke Lyu, and Donglin Wang. Efficient online rl fine tuning with offline pre-trained policy only. *arXiv preprint arXiv:2505.16856*, 2025.

Tengyang Xie, Nan Jiang, Huan Wang, Caiming Xiong, and Yu Bai. Policy finetuning: Bridging sample-efficient offline and online reinforcement learning. *Advances in neural information processing systems*, 34:27395–27407, 2021.

Shunpeng Yang, Ben Liu, and Hua Chen. Policyflow: Policy optimization with continuous normalizing flow in reinforcement learning. In *International Conference on Learning Representations*, 2026.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, 27, 2014.

Zishun Yu and Xinhua Zhang. Actor-critic alignment for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, pp. 40452–40474. PMLR, 2023.

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

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. *Advances in Neural Information Processing Systems*, 38: 106282–106319, 2026a.

Yixian Zhang, Shu-ang Yu, Tonghe Zhang, Mo Guang, Haojia Hui, Kaiwen Long, Yu Wang, Chao Yu, and Wenbo Ding. Sac flow: Sample-efficient reinforcement learning of flow-based policies via velocity-reparameterized sequential modeling. In *International Conference on Learning Representations*, volume 2026, pp. 121983–122016, 2026b.

Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. In *International Conference on Learning Representations*, volume 2025, pp. 32343–32368, 2025.

A PROOFS

We use the squared Hellinger convention in Eq. (8). For probability measures P, Q and a common dominating measure λ, writing $p = dP/d\lambda$ and $q = dQ/d\lambda$,

$$D^2_{\mathrm{Hel}}(P, Q) = \int (\sqrt{p} - \sqrt{q})^2 \, d\lambda. \tag{12}$$

This expression is independent of the choice of dominating measure.

A.1 PROOF OF LEMMA 1

Fix a step h and a threshold $\tau \in (0, 1)$. Let

$$\lambda_h = d^\beta_h + d^\mu_h, \qquad p_h = \frac{dd^\beta_h}{d\lambda_h}, \qquad q_h = \frac{dd^\mu_h}{d\lambda_h}. \tag{13}$$

Define

$$A_h = \{s : q_h(s) < \tau p_h(s)\}, \qquad \widetilde{G}_h = G_h \cap A^c_h. \tag{14}$$

Thus, $\widetilde{G}_h$ retains the part of G_h on which the pretrained actor assigns at least a fraction τ of the behavior occupancy density.

For any measurable set $E \subseteq \widetilde{G}_h$, Assumption 1 and $p_h \le q_h/\tau$ give

$$\begin{aligned} d^{\pi^\dagger}_h(E) &\le C_h d^\beta_h(E) = C_h \int_E p_h \, d\lambda_h \\ &\le \frac{C_h}{\tau} \int_E q_h \, d\lambda_h = \frac{C_h}{\tau} d^\mu_h(E). \end{aligned} \tag{15}$$

This proves the first restricted measure inequality in Equation 9. On A_h, we have $\sqrt{q_h} < \sqrt{\tau}\sqrt{p_h}$, and hence

$$(\sqrt{p_h} - \sqrt{q_h})^2 \ge (1 - \sqrt{\tau})^2 p_h. \tag{16}$$

Integrating over A_h yields

$$\begin{aligned} d^\beta_h(A_h) &\le \frac{1}{(1-\sqrt{\tau})^2} \int_{A_h} (\sqrt{p_h} - \sqrt{q_h})^2 \, d\lambda_h \\ &\le \frac{\zeta_h}{(1-\sqrt{\tau})^2}. \end{aligned} \tag{17}$$

Since $\widetilde{G}^c_h = G^c_h \cup (G_h \cap A_h)$, with disjoint sets on the right, Assumption 1 implies

$$\begin{aligned} d^{\pi^\dagger}_h(\widetilde{G}^c_h) &= d^{\pi^\dagger}_h(G^c_h) + d^{\pi^\dagger}_h(G_h \cap A_h) \\ &\le m_h + C_h d^\beta_h(G_h \cap A_h) \\ &\le m_h + \frac{C_h \zeta_h}{(1-\sqrt{\tau})^2}. \end{aligned} \tag{18}$$

Together, Equations 15 and 18 establish Equation 9.

A.2 VALUE SPAN AND PERFORMANCE DIFFERENCE

For a policy π, define

$$V^\pi_h(s) = \mathbb{E}_\pi \left[\sum_{t=h}^{H} \gamma^{t-h} r_t \mid s_h = s \right], Q^\pi_h(s, a) = r_h(s, a) + \gamma \mathbb{E}[V^\pi_{h+1}(s') \mid s, a], \tag{19}$$

with $V^\pi_{H+1} \equiv 0$; for stochastic rewards, $r_h(s, a)$ denotes the conditional mean. Bounded rewards imply that both $V^\pi_h(s)$ and $Q^\pi_h(s, a)$ lie in the interval

$$\left[r_{\min} \sum_{j=0}^{H-h} \gamma^j, r_{\max} \sum_{j=0}^{H-h} \gamma^j \right]. \tag{20}$$

Since $V_h^{\widehat{\pi}}(s) = \mathbb{E}_{a\sim\widehat{\pi}_h(\cdot|s)} Q_h^{\widehat{\pi}}(s,a)$, the definition of Δ_h in Equation 10 therefore gives

$$0 \le \Delta_h(s) \le B_h. \tag{21}$$

Next, let $A_h^{\widehat{\pi}}(s,a) = Q_h^{\widehat{\pi}}(s,a) - V_h^{\widehat{\pi}}(s)$. The Bellman equation and telescoping along trajectories of $\pi^\dagger$ yield

$$\begin{aligned}\sum_{h=1}^{H} \gamma^{h-1} \mathbb{E}_{\pi^\dagger}[A_h^{\widehat{\pi}}(s_h,a_h)] &= \mathbb{E}_{\pi^\dagger}\left[\sum_{h=1}^{H} \gamma^{h-1}\left(r_h + \gamma V_{h+1}^{\widehat{\pi}}(s_{h+1}) - V_h^{\widehat{\pi}}(s_h)\right)\right] \\ &= J(\pi^\dagger) - \mathbb{E}_{s_1\sim\rho} V_1^{\widehat{\pi}}(s_1) \\ &= J(\pi^\dagger) - J(\widehat{\pi}).\end{aligned} \tag{22}$$

Conditioning on s_h, and upper bounding the comparator's expected action value by its supremum, gives

$$\begin{aligned}J(\pi^\dagger) - J(\widehat{\pi}) &= \sum_{h=1}^{H} \gamma^{h-1} \mathbb{E}_{s\sim d_h^{\pi^\dagger}}\left[\mathbb{E}_{a\sim\pi_h^\dagger(\cdot|s)} Q_h^{\widehat{\pi}}(s,a) - V_h^{\widehat{\pi}}(s)\right] \\ &\le \sum_{h=1}^{H} \gamma^{h-1} \mathbb{E}_{s\sim d_h^{\pi^\dagger}} \Delta_h(s).\end{aligned} \tag{23}$$

A.3 Proof of Theorem 1

Let $\mathcal{E}$ denote the event in Assumption 2, on which $\mathbb{E}_{d_h^\mu}\Delta_h \le \varepsilon_h$ holds simultaneously for every h. By assumption, $\Pr(\mathcal{E}) \ge 1-\delta$. We work on this event and fix any $\tau \in (0,1)$. For the regions $\widetilde{G}_h$ constructed in Lemma 1, decompose

$$\mathbb{E}_{d_h^{\pi^\dagger}} \Delta_h = \int_{\widetilde{G}_h} \Delta_h \, dd_h^{\pi^\dagger} + \int_{\widetilde{G}_h^c} \Delta_h \, dd_h^{\pi^\dagger}. \tag{24}$$

On the retained region, Lemma 1 and non-negativity of Δ_h imply

$$\begin{aligned}\int_{\widetilde{G}_h} \Delta_h \, dd_h^{\pi^\dagger} &\le \frac{C_h}{\tau} \int_{\widetilde{G}_h} \Delta_h \, dd_h^{\mu} \\ &\le \frac{C_h}{\tau} \mathbb{E}_{d_h^\mu} \Delta_h \le \frac{C_h}{\tau} \varepsilon_h.\end{aligned} \tag{25}$$

On the complement, Equation 21 and Lemma 1 give

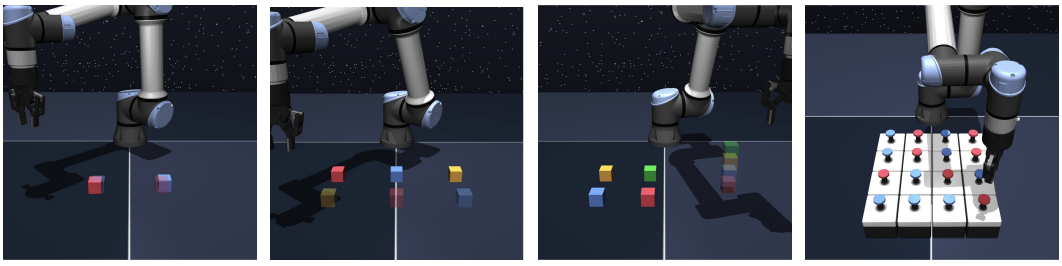
$$\begin{aligned}\int_{\widetilde{G}_h^c} \Delta_h \, dd_h^{\pi^\dagger} &\le B_h d_h^{\pi^\dagger}(\widetilde{G}_h^c) \\ &\le B_h m_h + \frac{B_h C_h \zeta_h}{(1-\sqrt{\tau})^2}.\end{aligned} \tag{26}$$

Substituting Equation 25 and26 into Equation 24, and then into Equation 23, proves

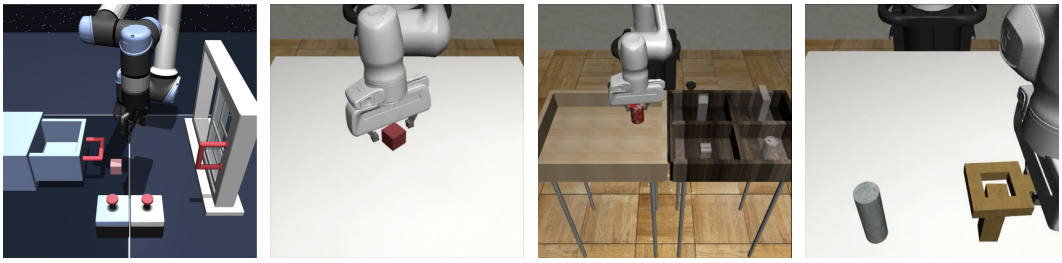
$$J(\pi^\dagger) - J(\widehat{\pi}) \le \sum_{h=1}^{H} \gamma^{h-1} \left[\frac{C_h}{\tau}\varepsilon_h + B_h m_h + \frac{B_h C_h \zeta_h}{(1-\sqrt{\tau})^2}\right], \tag{27}$$

which is Equation 11. The coverage inequalities hold deterministically for every $\tau \in (0,1)$, so the conclusion holds for all such τ on the same event $\mathcal{E}$.

Remark 1. *The proof uses the induced occupancy discrepancy ζ_h, rather than the loss used to train μ. It therefore applies to flow-matching-based actor pretraining whenever the stated coverage and downstream-learning conditions hold. It does not assert that a small empirical pretraining loss alone guarantees a small Hellinger occupancy discrepancy.*



(a) Cube Double (b) Cube Triple (c) Cube Quadruple (d) Puzzle 4×4



(e) Scene (f) Lift (g) Can (h) Square

Figure 9: Visualizations of the environments. Subfigures (a-e) show the OGBench domains. Subfigures (f-h) depict the manipulation tasks from Robomimic.

B ENVIRONMENT

In this section, we present the domains used in our experiments in detail. Table 1 summarizes some information about each domain, including **Reward Type, Dataset Size, Episode Length and Action Dimension.**

B.1 OGBENCH

We choose 5 OGBench manipulation domains and use the single-task version of it. We use the standard **play** version of the dataset, in which trajectories were collected using a non-Markovian expert policy. For `Cube Double/Triple/Quadruple`, the agents need to manipulate an arm and learn a pick-and-place skill to place colored cube blocks to the right locations. Particularly, `Cube Quadruple` is an extremely challenging domain in which the agent must process four cubes in sequence. It is almost impossible to learn an effective policy under standard dataset. Hence, we use the 100M-size dataset provided officially. `Puzzle 4×4` requires solving the "Lights Out" puzzle with a robot arm. For `Scene-sparse`, we use the sparse-reward mode of the reward function such that the agent receives a reward of 0 only upon completing the task, and -1 in all other cases.

B.2 ROBOMIMIC

We select three challenging domains from Robomimic, including `Lift, Can and Square`. We conduct our experiments on Multi-Human datasets. These datasets were collected by 6 operators, including 300 successful trajectories. `Lift` requires the robot arm to pick up a cube. `Can` requires the robot arm to pick up a can and place it into a container. `Square` requires the robot arm to pick up a square sleeve and fit it onto a square post.

Table 1: Specifications of the environments used in our experiments.

Environment	Reward Type	Dataset Size	Episode Length	Action Dimension
OGBench				
cube-double	Sparse	1M	500	5
cube-triple	Sparse	3M	1000	5
cube-quadruple	Sparse	100M	1000	5
puzzle-4x4	Sparse	1M	500	5
scene	Sparse	1M	750	5
Robomimic				
lift	Sparse	31,127	500	7
can	Sparse	62,756	500	7
square	Sparse	80,731	500	7

C Related Work

C.1 Offline-to-Online RL

The primary objective of the O2O RL is to leverage offline pre-training on static datasets to enable more efficient fine-tuning during the subsequent online stage. This approach mitigates the fundamental challenges where training online RL from-scratch often fails to acquire effective policies in complex environments, thereby significantly reducing both computational overhead and time costs (Levine et al., 2020; Xie et al., 2021; Guo et al., 2023; Yu & Zhang, 2023; Zhang et al., 2023; 2026b). However, O2O RL faces the problem of the distribution shift between the dataset and the environment, which causes catastrophic performance drop in online stage. To confront this problem, a few solutions are introduced. In this section, we list some of them that are related to our work.

Pessimism and Conservatism Early algorithms mainly focus on eliminating extrapolation errors. By augmenting the Bellman objective with a conservative regularizer, CQL (Kumar et al., 2020) lowers the estimated values of actions that are poorly supported by the offline dataset, thereby mitigating overestimation caused by distribution shift. Cal-QL (Nakamoto et al., 2023) constrains the learned conservative values to remain above the value of a reference policy, thereby placing the Q-values on a reasonable scale for subsequent online fine-tuning. Some algorithms reach the same goal by explicitly restricting the learned policy from deviating too far from dataset distribution. For instance, BCQ (Fujimoto et al., 2019) uses a conditional generative model and a perturbation model to restrict action selection to actions that are likely under, or close to, the dataset distribution. TD3+BC (Fujimoto & Gu, 2021) explicitly augments the actor objective with a behavior-cloning term, whereas AWAC (Nair et al., 2020) performs an advantage-weighted maximum-likelihood policy update, implicitly constraining the learned policy toward actions supported by the replay distribution.

Replay Design and Critic Calibration Beyond conservatism, recent offline-to-online RL methods improve fine-tuning through replay design and explicit value calibration. For example, RLPD (Ball et al., 2023) trains an off-policy online RL agent with minibatches drawn jointly from fixed offline data and newly collected online experience, together with high update-to-data ratios and critic regularization for stable and sample-efficient learning. WSRL (Zhou et al., 2025) introduces the warm-up stage in which only online data are used to calibrate the pretrained critic network. OPT (Shin et al., 2025) introduces a newly initialized critic and an additional online pre-training phase, during which the new critic is trained with both the offline dataset and a small set of online transitions. During fine-tuning, policy improvement is guided by a scheduled weighted combination of the offline-pretrained critic and the online-pretrained critic.

Xiao et al. (2025) present Policy-only Reinforcement Learning (PORL) that is closely related to our work. Xiao et al. (2025) also discover the phenomenon that the pretrained critic could potentially hinder online fine-tuning. Their work focuses primarily on the specific case of the Gaussian algorithm; PORL studies online fine-tuning from a pretrained policy without retaining a pretrained

critic and emphasizes persistent value underestimation associated with offline training. This conclusion differs from the finding of reduced plasticity associated with CFP. While PORL offers valuable insights by focusing on initializing the critic through online sampling, thereby bypassing offline critic training entirely, our work complements this perspective by examining the effect of prolonged critic training in flow-based methods and by studying a short offline warm-up stage before online fine-tuning.

D IMPLEMENTATION

In this section, we present the details of our implementation of CFP on all the baselines.

Flow Q-Learning. FQL is a strong baseline for offline reinforcement learning. Its main innovation is to exploit the expressive power of a multi-step flow network to approximate the behavior distribution in the offline dataset, while using a Q-loss to propagate gradients from the critic to the actor. Directly backpropagating through the multi-step flow generation process, however, may lead to unstable gradients. To avoid this issue, FQL introduces a one-step noise-conditioned policy that directly distills the actions generated by the multi-step behavior flow. Although the one-step policy may sacrifice some of the expressive capacity of the original multi-step flow, it enables more stable policy optimization.

More concretely, FQL consists of the following three networks:

1. a behavior flow policy $\mu_\theta(s, z)$, which generates an action from a state s and Gaussian noise $z \sim \mathcal{N}(0, I^d)$ through multi-step flow integration;
2. a one-step noise-conditioned policy $\mu_\omega(s, z)$, which distills the output of the behavior flow policy; and
3. a critic $Q_\phi(s, a)$, which estimates the expected return of a state–action pair.

The behavior flow is trained with the flow-matching objective in Equation 1, while the one-step actor is trained by distilling the flow output using the same noise input. The teacher output is detached in the distillation update,

$$L_\pi(\omega) = \alpha \mathbb{E}_{\substack{s\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^d)}} \left[\|\mu_\omega(s,z) - \mu_\theta(s,z)\|_2^2 \right] + \mathbb{E}_{\substack{s\sim\mathcal{D},\\ a\sim\pi_\omega(\cdot|s)}} \left[-Q_\phi(s,a) \right], \tag{28}$$

where α controls the strength of behavior regularization. The first term distills the output of the multi-step behavior flow into the one-step policy, whereas the second term propagates the critic gradient to the one-step actor.

The critic is trained with the temporal-difference objective

$$L_Q(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\left(Q_\phi(s,a) - \left[r + \gamma Q_{\bar{\phi}}(s', a') \right] \right)^2 \right], \tag{29}$$

where $a' \sim \pi_\omega(\cdot \mid s')$ and $Q_{\bar{\phi}}$ denotes the target critic.

QCFQL. QCFQL extends FQL to temporally extended action spaces by jointly generating and evaluating an action chunk $\mathbf{a}_t = (a_t, \ldots, a_{t+h-1})$. The QCFQL actor objective is

$$L_\pi(\omega) = \alpha \mathbb{E}_{\substack{s_t\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^{hd})}} \left[\|\mu_\omega(s_t,z) - \mu_\theta(s_t,z)\|_2^2 \right] + \mathbb{E}_{\substack{s_t\sim\mathcal{D},\\ \mathbf{a}_t\sim\pi_\omega(\cdot|s_t)}} \left[-Q_\phi(s_t,\mathbf{a}_t) \right]. \tag{30}$$

The QCFQL critic is trained using an h-step temporal-difference target:

$$L_Q(\phi) = \mathbb{E} \left[\left(Q_\phi(s_t,\mathbf{a}_t) - \left[r_t^{(h)} + \gamma^h Q_{\bar{\phi}}(s_{t+h}, \mathbf{a}_{t+h}) \right] \right)^2 \right], \tag{31}$$

where

$$r_t^{(h)} = \sum_{j=0}^{h-1} \gamma^j r_{t+j}, \qquad \mathbf{a}_{t+h} \sim \pi_\omega(\cdot \mid s_{t+h}). \tag{32}$$

QCFQL-nstep. QCFQL-nstep is a similar implementation of QCFQL. QC-FQL-nstep is essentially derived from SAC-Flow. It adopts the core concept that flow represents sequential modeling. By directly utilizing backpropagation through a multi-layer network and employing techniques such as tanh function compression, QC-FQL-nstep preserves the high expressive power of multi-step flow while partially mitigating the problems of vanishing or exploding gradients. The QCFQL-nstep actor objective is

$$L_\pi(\theta) = \alpha \mathbb{E}_{\substack{(s_t,\mathbf{a}_t)\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^{hd})}} \left[\|\mu_\theta(s_t, z) - \mathbf{a}_t\|_2^2 \right] + \mathbb{E}_{\substack{s_t\sim\mathcal{D},\\ \mathbf{a}_t\sim\pi_\theta(\cdot|s_t)}} \left[-Q_\phi(s_t, \mathbf{a}_t) \right]. \tag{33}$$

The remaining components of QCFQL-nstep are identical to those of QCFQL.

QC. QC uses the same flow-matching behavior policy μ_θ as FQL, but does not introduce an additional one-step noise-conditioned actor. Instead, QC implicitly defines its policy through best-of-N candidate selection. Specifically, for a given state s, we first sample N independent Gaussian noise vectors,

$$z^1, z^2, \ldots, z^N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I^{hd}), \tag{34}$$

and generate N candidate actions using the multi-step behavior flow:

$$\mathbf{a}_t^i = \mu_\theta(s_t, z^i), \qquad i \in \{1, \ldots, N\}. \tag{35}$$

The policy then selects the candidate with the largest estimated Q-value:

$$i^* = \arg\max_{i\in\{1,\ldots,N\}} Q_\phi(s_t, \mathbf{a}^i), \qquad \pi_{\mathrm{QC}}(s) = \mathbf{a}^{i^*}. \tag{36}$$

Unlike FQL and QCFQL, QC does not backpropagate the critic gradient. The selected best-of-N action is directly used to construct the temporal-difference target. Accordingly, the critic is optimized by

$$L_Q(\phi) = \mathbb{E}_{(s,\mathbf{a}_t,r,s')\sim\mathcal{D}} \left[\left(Q_\phi(s_t, \mathbf{a_t}) - \left[r_t^{(h)} + \gamma^h Q_{\bar{\phi}} \left(s_{t+h}, \mathbf{a}_{t+h}^{i^*} \right) \right] \right)^2 \right], \tag{37}$$

where $\mathbf{a}_{\mathbf{t+h}}^{\mathbf{i}^*} \sim \pi_{QC}(\cdot|s_{t+h})$.

Next, we demonstrate how we incorporate CFP into the original algorithm.

FQL-CFP. Since the implementations of all the algorithms are largely similar, we primarily describe how FQL-CFP is implemented.

The objective of FQL-CFP during the pretraining stage is:

$$L_\pi(\omega) = \alpha \mathbb{E}_{\substack{s\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^d)}} \left[\|\mu_\omega(s, z) - \mu_\theta(s, z)\|_2^2 \right]. \tag{38}$$

CFP performs no critic updates during the actor-pretraining stage.

During the warm-up phase, the actor and the initialized critic undergo brief updates and calibration using the offline dataset, with the following update objective:

$$L_\pi(\omega) = \alpha \mathbb{E}_{\substack{s\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^d)}} \left[\|\mu_\omega(s, z) - \mu_\theta(s, z)\|_2^2 \right] - \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\omega(\cdot|s)}[Q_\phi(s, a)]. \tag{39}$$

The critic is trained with the temporal-difference objective

$$L_Q(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\left(Q_\phi(s, a) - \left[r + \gamma Q_{\bar{\phi}}(s', a') \right] \right)^2 \right], \tag{40}$$

where $a' \sim \pi_\omega(\cdot \mid s')$ and terminal transitions are masked in the bootstrap target. During the online phase, the agent samples trajectories in the real environment and stores the transitions in the replay buffer. The objectives are:

$$L_\pi(\omega) = \alpha \mathbb{E}_{\substack{s\sim\mathcal{B},\\ z\sim\mathcal{N}(0,I^d)}} \left[\|\mu_\omega(s, z) - \mu_\theta(s, z)\|_2^2 \right] - \mathbb{E}_{s\sim\mathcal{B},\, a\sim\pi_\omega(\cdot|s)}[Q_\phi(s, a)], \tag{41}$$

$$L_Q(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{B}}\left[\left(Q_\phi(s,a) - \left[r + \gamma Q_{\bar{\phi}}(s', a')\right]\right)^2\right]. \tag{42}$$

Since the objectives for the warm-up and online phases are identical—differing only in the source of the updates (the warm-up phase relies entirely on the dataset, whereas the online phase utilizes batches containing online data)—no further distinction will be made between the two.

QCFQL-CFP. The update method is very similar to that of FQL-CFP; QCFQL simply incorporates an action chunk. Its update objective during the pretraining stage is

$$L_\pi(\omega) = \alpha\mathbb{E}_{\substack{s_t\sim\mathcal{D},\\ z\sim\mathcal{N}(0,I^{hd})}}\left[\|\mu_\omega(s_t,z) - \mu_\theta(s_t,z)\|_2^2\right] \tag{43}$$

For both the warm-up and online stage, the targets are:

$$L_\pi(\omega) = \alpha\mathbb{E}_{\substack{s_t\sim\mathcal{M},\\ z\sim\mathcal{N}(0,I^{hd})}}\left[\|\mu_\omega(s_t,z) - \mu_\theta(s_t,z)\|_2^2\right] + \mathbb{E}_{\substack{s_t\sim\mathcal{M},\\ \mathbf{a}_t\sim\pi_\omega(\cdot|s_t)}}\left[-Q_\phi(s_t,\mathbf{a}_t)\right], \tag{44}$$

$$L_Q(\phi) = \mathbb{E}\left[\left(Q_\phi(s_t,\mathbf{a}_t) - \left[r_t^{(h)} + \gamma^h Q_{\bar{\phi}}(s_{t+h},\mathbf{a}_{t+h})\right]\right)^2\right], \tag{45}$$

where $\mathcal{M} = \begin{cases}\mathcal{D}, & \text{warm-up,}\\ \mathcal{B}, & \text{online fine-tuning,}\end{cases}$

QCFQL-nstep-CFP. QCFQL-nstep-CFP differs slightly from its O2O variant. During the pretraining phase, the QCFQL-nstep-CFP actor learns the velocity field via Flow Matching:

$$L_\pi(\theta) = \mathbb{E}_{\substack{\mathbf{x_0}\sim\mathcal{N}(0,I_{hd}),\\ (\mathbf{x_1}=a,s)\sim\mathcal{D},\\ t\sim\text{Unif}([0,1])}}[\|\pi_\theta(t,s,\mathbf{x_t}) - (\mathbf{x_1}-\mathbf{x_0})\|_2^2] \tag{46}$$

For warm-up and online updates, the objectives are identical to those of its O2O variant:

$$L_\pi(\theta) = \alpha\mathbb{E}_{\substack{(s_t,\mathbf{a}_t)\sim\mathcal{M},\\ z\sim\mathcal{N}(0,I^{hd})}}\left[\|\mu_\theta(s_t,z) - \mathbf{a}_t\|_2^2\right] + \mathbb{E}_{\substack{s_t\sim\mathcal{M},\\ \mathbf{a}_t\sim\pi_\theta(\cdot|s_t)}}\left[-Q_\phi(s_t,\mathbf{a}_t)\right], \tag{47}$$

$$L_Q(\phi) = \mathbb{E}\left[\left(Q_\phi(s_t,\mathbf{a}_t) - \left[r_t^{(h)} + \gamma^h Q_{\bar{\phi}}(s_{t+h},\mathbf{a}_{t+h})\right]\right)^2\right]. \tag{48}$$

QC-CFP. QC naturally shares the same training approach as CFP—specifically, using only the actor network to approximate the velocity field. So its training objective during pretraining is:

$$L_\pi(\theta) = \mathbb{E}_{\substack{\mathbf{x_0}\sim\mathcal{N}(0,I_{hd}),\\ (\mathbf{x_1}=a,s)\sim\mathcal{D},\\ t\sim\text{Unif}([0,1])}}[\|\pi_\theta(t,s,\mathbf{x_t}) - (\mathbf{x_1}-\mathbf{x_0})\|_2^2]. \tag{49}$$

For warm-up and online training, the objectives are:

$$L_\pi(\theta) = \mathbb{E}_{\substack{\mathbf{x_0}\sim\mathcal{N}(0,I_{hd}),\\ (\mathbf{x_1}=a,s)\sim\mathcal{M},\\ t\sim\text{Unif}([0,1])}}[\|\pi_\theta(t,s,\mathbf{x_t}) - (\mathbf{x_1}-\mathbf{x_0})\|_2^2], \tag{50}$$

$$L_Q(\phi) = \mathbb{E}\left[\left(Q_\phi(s_t,\mathbf{a_t}) - \left[r_t^{(h)} + \gamma^h Q_{\bar{\phi}}\left(s_{t+h},\mathbf{a}_{t+h}^{i^*}\right)\right]\right)^2\right], \tag{51}$$

where $\mathbf{a}_{\mathbf{t+h}}^{\mathbf{i^*}}$ follows the Best-of-N sampling.

-RLPD. We develop corresponding RLPD variants for all the algorithms used in this article. While the original RLPD utilizes a Gaussian-based algorithm, we adopt its concept of leveraging offline data; specifically, during the online phase, we maintain two replay buffers of equal size—one for offline data and one for online data—and construct training batches by sampling equal amounts of data from both buffers.

Table 2: Common hyperparameters used in our experiments.

Parameter	Value
Batch size (M)	256
Discount factor (γ)	0.99
Optimizer	Adam
Learning rate	3×10^{-4}
Target network update rate (τ)	5×10^{-3}
Critic ensemble size	2
Number of flow integration steps (T)	4 for QCFQL-nstep; 10 otherwise
Action chunk length (h)	1 for FQL; 5 otherwise
Number of offline training steps	1×10^{6}
Number of online environment steps	1×10^{6}
Network width	512
Network depth	4 hidden layers
Number of best-of-N candidates (QC)	32

E EXPERIMENT

E.1 HYPERPARAMETERS

Unless otherwise specified, we use the same hyperparameters across all environments and algorithms. The common hyperparameters are summarized in Table 2.

For specific hyperparameters of α values for backpropagation algorithms and N for Best-of-N algorithms, see Table 3. We tune the hyperparameters based on previous works (Park et al., 2025b; Li et al., 2026; Li & Levine, 2026). With the exception of the `puzzle 4×4 task 3`, we keep the hyperparameters identical for both CFP and O2O to ensure that any differences in fine-tuning performance are not attributable to hyperparameter settings. For the QCFQL and QCFQL-nstep algorithms in this specific task, we find that using the α value specified in the original paper causes extreme instability during CFP training, so we make appropriate adjustments to this parameter. It should be noted that this inconsistency does not confer an advantage to CFP, as the O2O variants of both algorithms converge extremely rapidly to a success rate of 1 on this task.

Regarding the hyperparameter of warm-up steps, we conduct an exhaustive search within the range of [5k, 10k, 30k, 50k, 0.1M, 0.2M] and select the value that achieves the best balance between final performance and computational efficiency, see Table 4.

Table 3: Task-specific hyperparameters for all O2O and CFP variants.

Environment	Task	QC (N)		FQL (α)		QCFQL-nstep (α)		QCFQL (α)	
		O2O	CFP	O2O	CFP	O2O	CFP	O2O	CFP
Cube Double	Task 1	32	32	300	300	100	100	100	100
	Task 2	32	32	300	300	100	100	100	100
	Task 3	32	32	300	300	100	100	100	100
	Task 4	32	32	300	300	100	100	100	100
	Task 5	32	32	300	300	100	100	100	100
Cube Triple	Task 1	32	32	300	300	100	100	100	100
	Task 2	32	32	300	300	100	100	100	100
	Task 3	32	32	300	300	100	100	100	100
	Task 4	32	32	300	300	100	100	100	100
	Task 5	32	32	300	300	100	100	100	100
Cube Quadruple	Task 1	32	32	300	300	100	100	100	100
	Task 2	32	32	300	300	100	100	100	100
	Task 3	32	32	300	300	100	100	100	100
	Task 4	32	32	300	300	100	100	100	100
	Task 5	32	32	300	300	100	100	100	100
Puzzle 4×4	Task 1	64	64	1000	1000	1000	1000	1000	1000
	Task 2	64	64	1000	1000	1000	1000	1000	1000
	Task 3	64	64	1000	1000	1000	100	1000	100
	Task 4	64	64	1000	1000	1000	1000	1000	1000
	Task 5	64	64	1000	1000	1000	1000	1000	1000
Scene	Task 1	32	32	300	300	300	300	300	300
	Task 2	32	32	300	300	300	300	300	300
	Task 3	32	32	300	300	300	300	300	300
	Task 4	32	32	300	300	300	300	300	300
	Task 5	32	32	300	300	300	300	300	300
Lift	–	16	16	10000	10000	10000	10000	10000	10000
Can	–	16	16	10000	10000	10000	10000	10000	10000
Square	–	16	16	10000	10000	10000	10000	10000	10000

Table 4: Task-specific critic warm-up steps for the CFP variants.

Environment	Task	QC-CFP	FQL-CFP	QCFQL-nstep-CFP	QCFQL-CFP
Cube Double	Task 1	10000	10000	10000	10000
	Task 2	10000	10000	10000	10000
	Task 3	10000	10000	10000	10000
	Task 4	10000	10000	10000	10000
	Task 5	10000	10000	10000	10000
Cube Triple	Task 1	10000	10000	10000	10000
	Task 2	10000	10000	10000	10000
	Task 3	10000	10000	10000	10000
	Task 4	10000	10000	10000	10000
	Task 5	10000	10000	0.1M	10000
Cube Quadruple	Task 1	0.2M	0.2M	0.2M	0.2M
	Task 2	0.2M	0.2M	0.2M	0.2M
	Task 3	0.2M	0.2M	0.2M	0.2M
	Task 4	0.2M	0.2M	0.2M	0.2M
	Task 5	0.2M	0.2M	0.2M	0.2M
Puzzle 4×4	Task 1	10000	10000	10000	10000
	Task 2	10000	10000	10000	10000
	Task 3	50000	50000	50000	50000
	Task 4	10000	50000	10000	10000
	Task 5	10000	50000	10000	10000
Scene	Task 1	10000	30000	10000	10000
	Task 2	10000	30000	10000	10000
	Task 3	10000	30000	10000	10000
	Task 4	10000	30000	10000	10000
	Task 5	10000	30000	10000	10000
Lift	Task 1	10000	5000	10000	10000
Can	Task 1	10000	5000	10000	10000
Square	Task 1	10000	5000	10000	5000

E.2 Full Results

OGBench. Figure 10 shows the full results on OGBench.

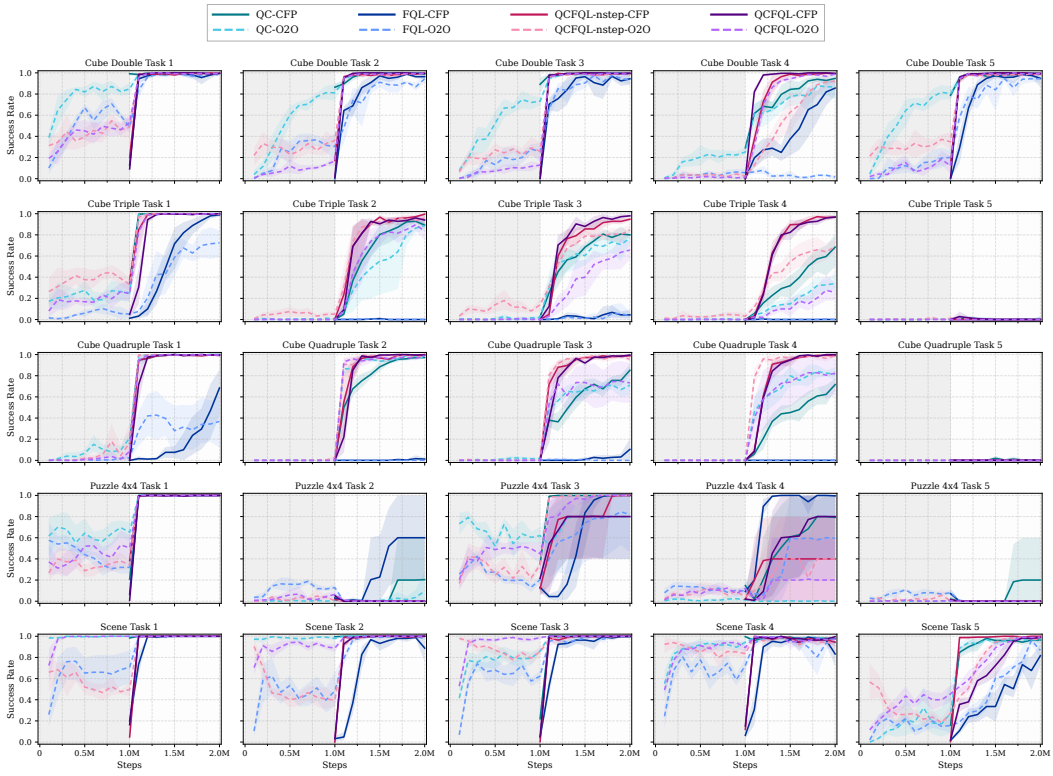


Figure 10: Full OGBench results by task.

Robomimic. Figure 5 shows the full results on Robomimic.

E.3 Ablation Study

We present more results of the ablation study here.

Q: Do we still need offline training?

We present task-specific and algorithm-specific results of this question in the following figures.

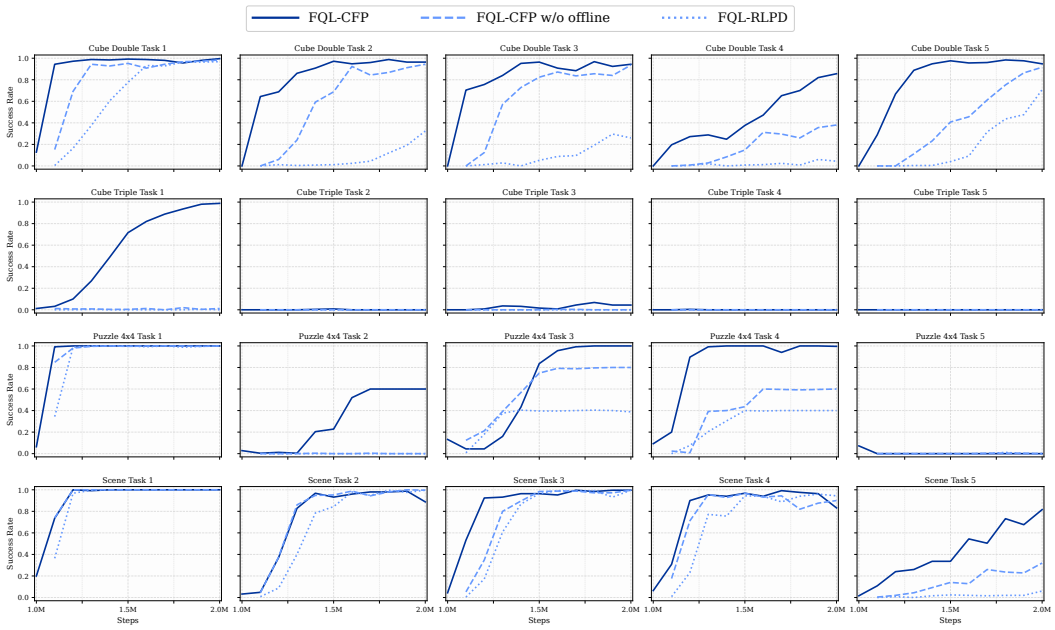


Figure 11: Full training curves of the ablation study on offline stage of FQL.

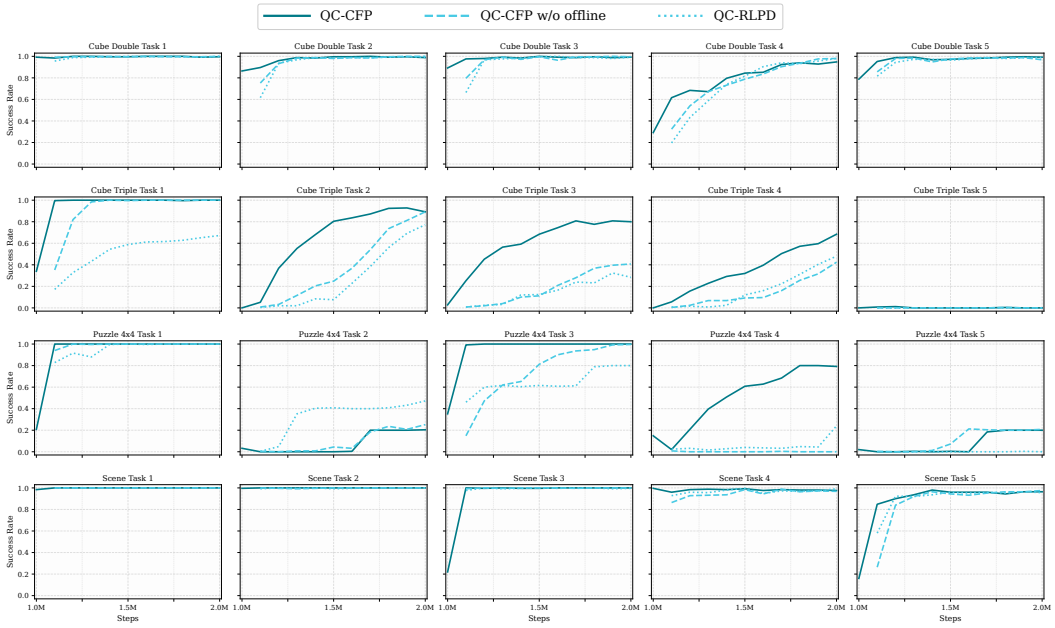


Figure 12: Full training curves of the ablation study on offline stage of QC.

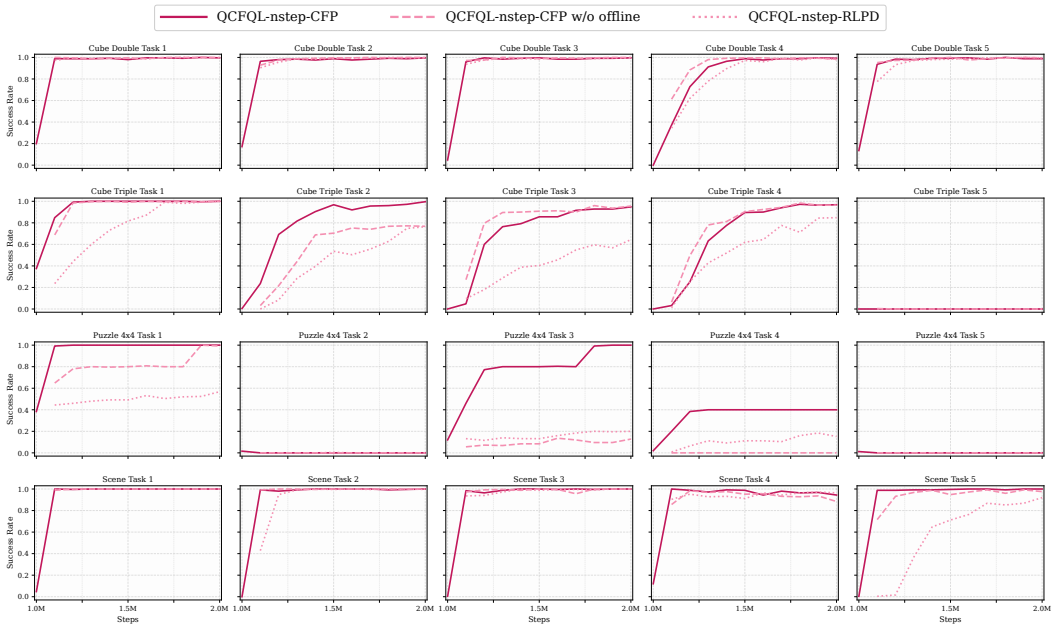


Figure 13: Full training curves of the ablation study on offline stage of QCFQL-nstep.

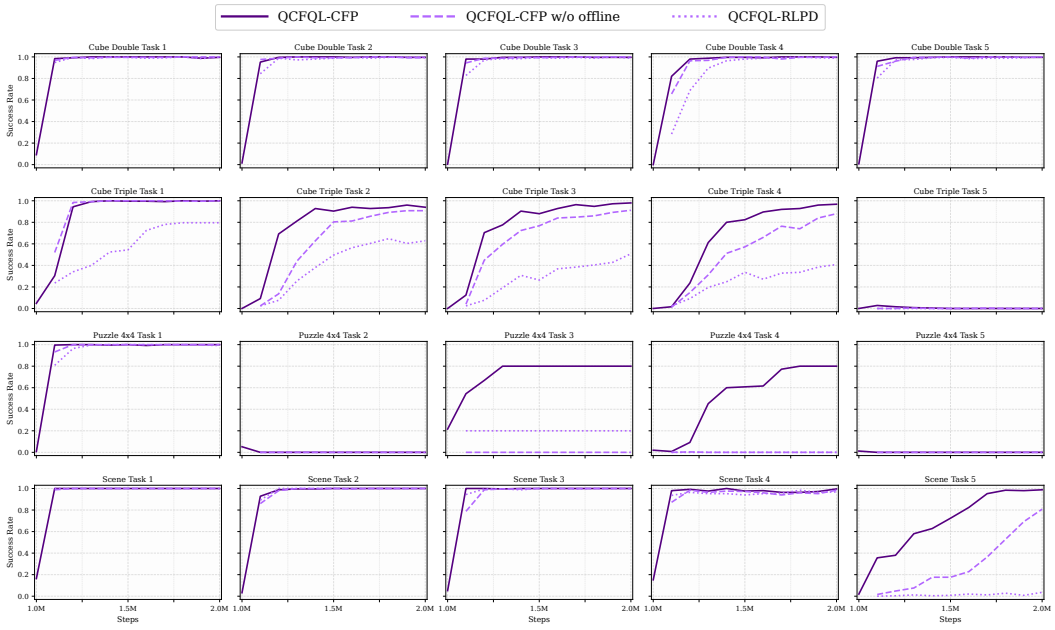


Figure 14: Full training curves of the ablation study on offline stage of QCFQL.

Q: Is CFP sensitive to actor pretraining?

We present task-specific results of this question in Figure 15. CFP is relatively robust to the number of pretraining steps of the actor.

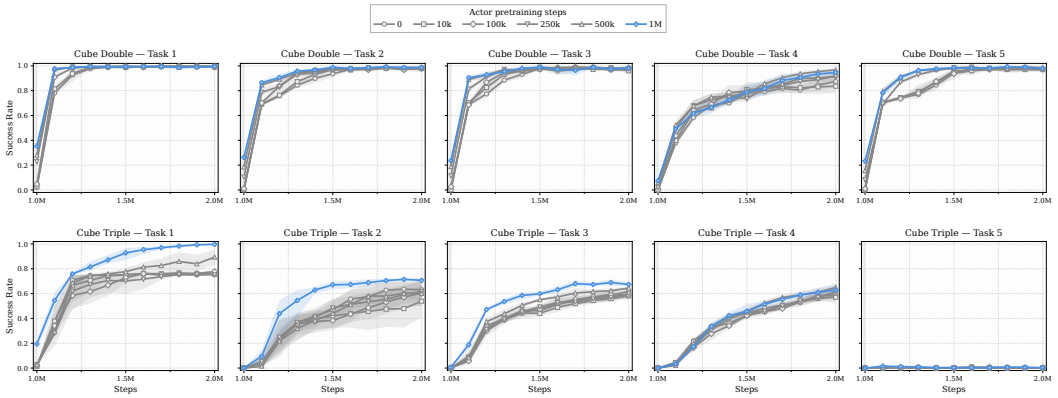


Figure 15: Full training curves of the ablation study on actor pretraining steps. CFP is relatively robust to the degree of the pretrained actor.

We further provide algorithm-specific results. See the following figures.

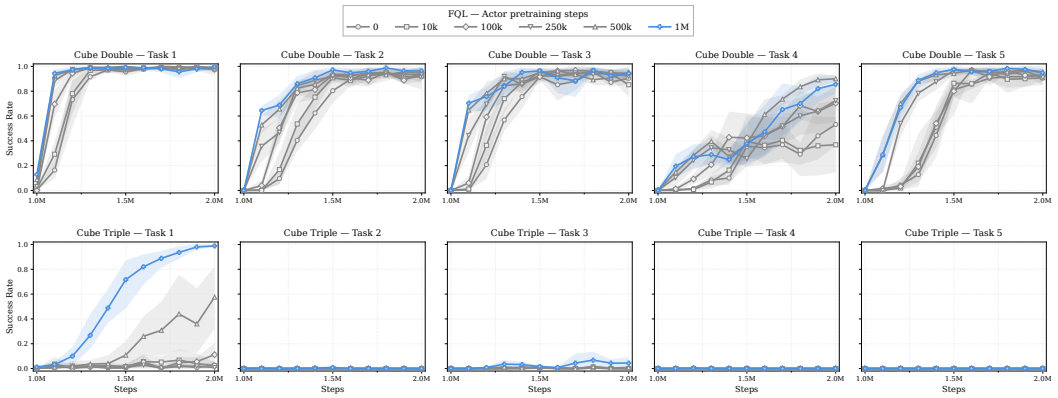


Figure 16: Full training curves of the ablation study on actor pretraining steps of FQL.

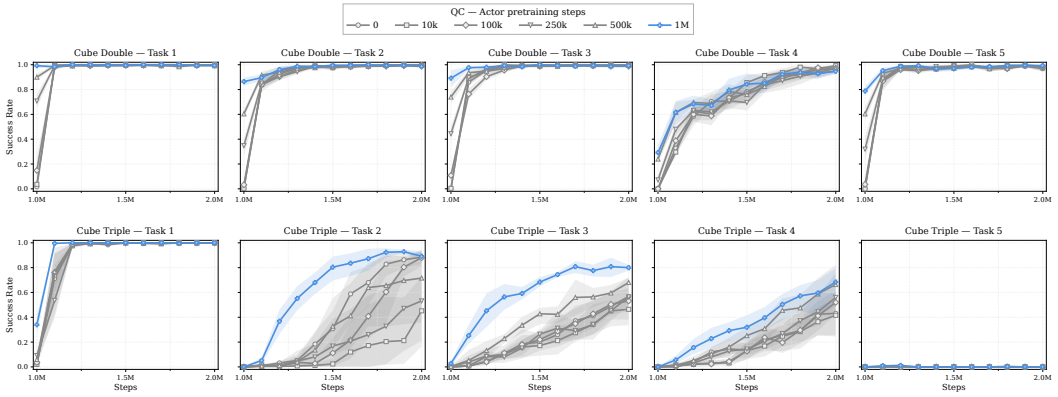


Figure 17: Full training curves of the ablation study on actor pretraining steps of QC.

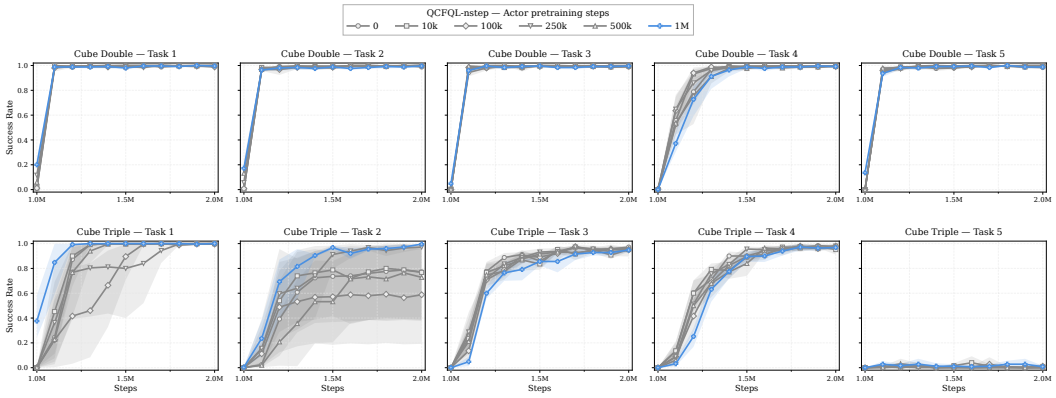


Figure 18: Full training curves of the ablation study on actor pretraining steps of QCFQL-nstep.

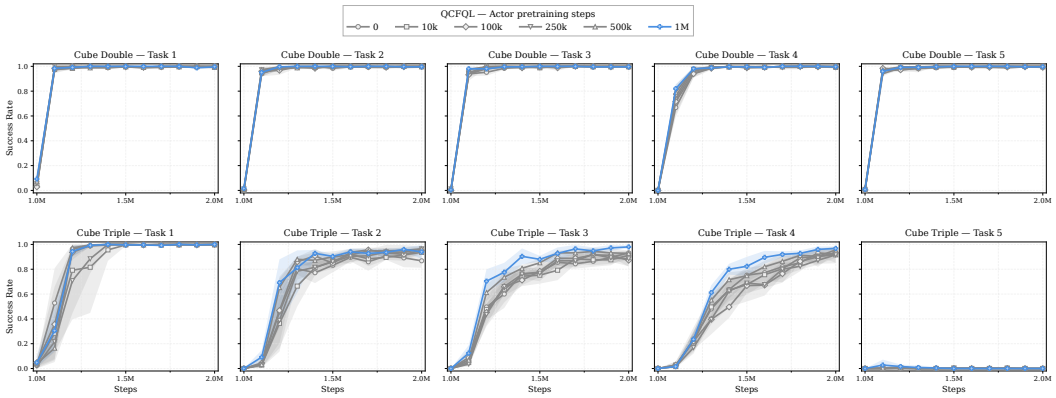


Figure 19: Full training curves of the ablation study on actor pretraining steps of QCFQL.

As shown in the figures above, FQL and QC exhibit relative sensitivity to the extent of actor pretraining, showing a positive correlation between the number of pretraining steps and subsequent fine-tuning performance. In contrast, QCFQL and QCFQL-nstep show little such correlation, though using 1M pretraining steps remains the most stable choice. These results reaffirm that a sufficient level of actor pretraining is crucial for stable online fine-tuning.

Q: Should the actor receive Q-guided updates during warm-up?

A: Probably yes.

This is an interesting question. The question here is essentially whether the actor should be guided by the fresh critic during the warm-up phase—that is, whether the update formula should incorporate the Q-loss from Equation 4. Initially, we hypothesize the actor should not be directly guided by the fresh critic, given that the gradients it provides are stochastic and uncontrollable; however, our experimental results indicate otherwise.

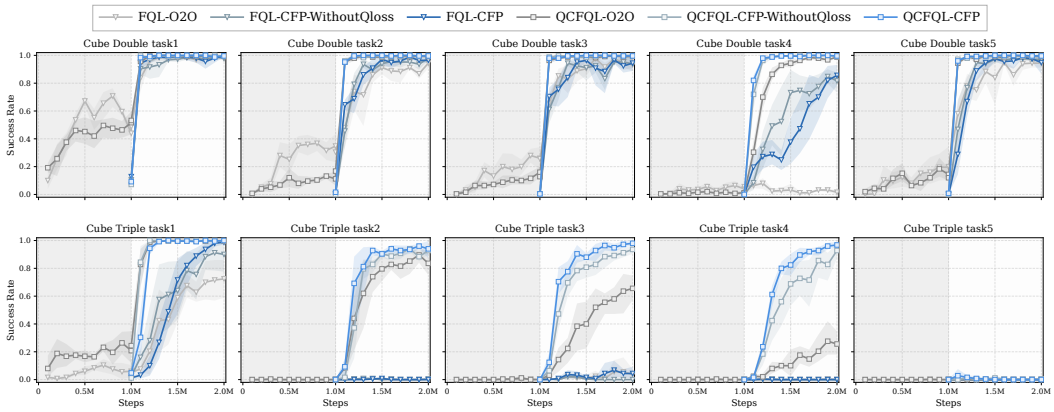


Figure 20: Ablation of the Q-loss during warm-up on OGBench.

Figure 20 shows that in some tasks, such as `Cube Triple task3` and `Cube Triple task4`, comparing the two warm-up approaches for QCFQL-CFP—one involving Q-loss updates and the other without—the former demonstrates superior online fine-tuning performance. In other environments and with other algorithms, however, the two warm-up methods show little difference. This result suggests that the fresh critic converges within a short warm-up phase, to the extent that it influences subsequent fine-tuning performance. It further confirms that even during very short offline training sessions, the critic can acquire valuable information and effectively leverage that value to influence the actor. Naturally, within the actor-critic coupling framework, the actor's behavior significantly influences the training of the critic; we further compare what happens if the Q-loss is omitted during the warm-up phase.

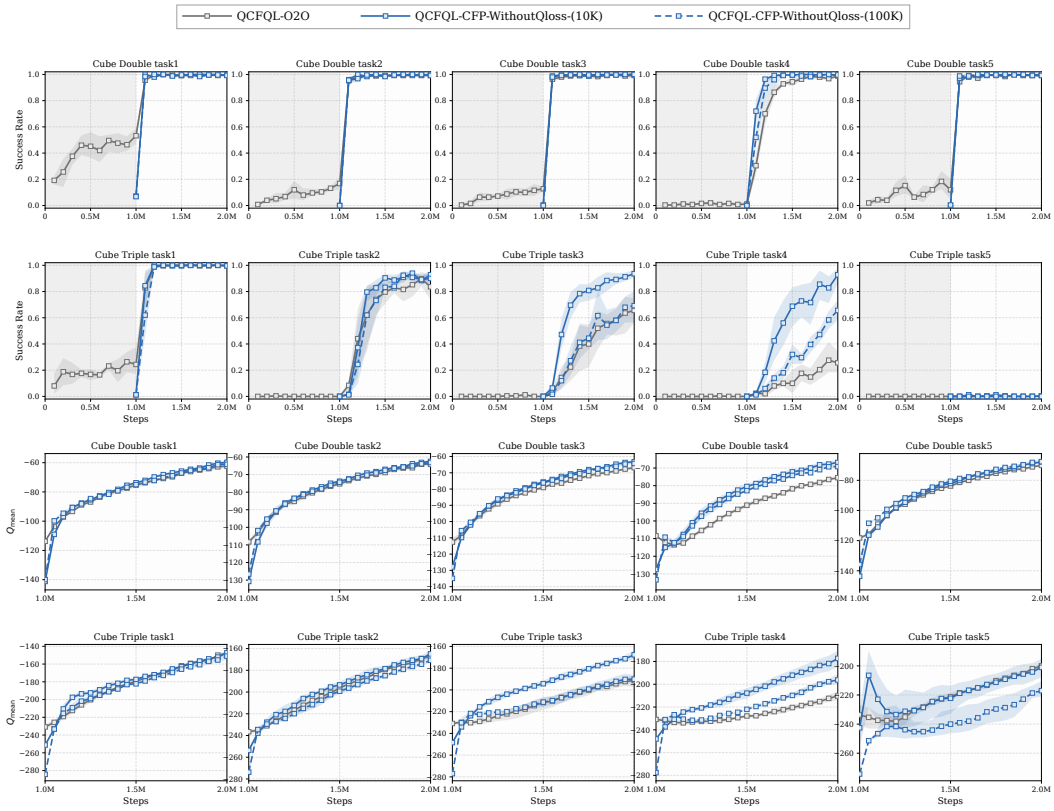


Figure 21: Ablation study of the sensitivity of CFP (without Q-loss) to warm-up steps.

Figure 21 shows that CFP without Q-loss during warm-up stage is rather sensitive to the length of warm-up steps. Overall, the longer the warm-up period, the worse the performance of online fine-tuning of CFP. We examine the corresponding mean Q-values and find that the Q mean assigned by the critic decreased as the warm-up length increased. We attribute this phenomenon to the interplay within the actor-critic architecture. We have demonstrated that a fresh critic can provide guidance even during the warm-up phase, enabling the actor to output higher-value actions; conversely, since the target action in the critic's update formula is provided by the actor, higher-quality actions from the actor result in higher action ratings from the critic. Without the inclusion of Q-loss, the actor remains confined to the low-quality actions present in the dataset, producing progressively lower critic estimates of state-action pairs. This result further confirms that offline datasets—characterized by low environmental coverage and limited trajectory success rates—exacerbate the critic's mismatch, thereby hindering generalization during the online phase.

F Toy Example

In this section, we introduce the toy example we implement to visualize the critic mismatch in detail.

We design an MDP table and simulate the complete O2O phase as well as the CFP implementation during the actual training process. In our experiments, we rigorously control the variables associated with O2O and CFP, ensuring that the sole difference between the two was whether the critic was initialized during the online phase.

F.1 Environment

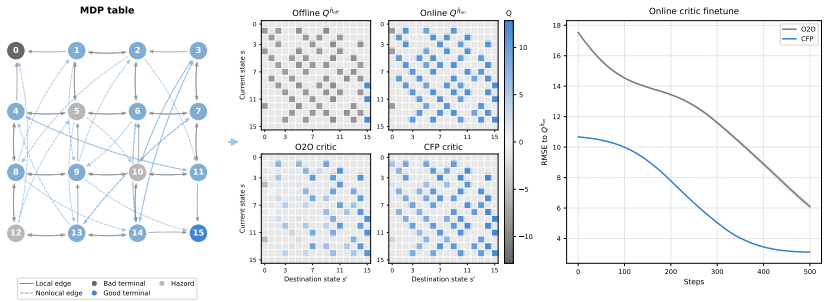


Figure 22: A toy example of an MDP table; heatmaps showing the Q-values from O2O and CFP critics, the dataset, and the true environment; and a comparison of the RMSE between the O2O/CFP critic outputs and the true Q-values during online fine-tuning.

The environment is a finite MDP with 16 states as shown in Figure 22. For visualization, the states are placed on a 4×4 canvas,

$$\begin{array}{cccc} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & 15, \end{array} \tag{52}$$

but these coordinates do not define the transition dynamics. The dynamics are instead specified by an irregular directed graph. An action at state s selects one of its valid destination states $s' \in \mathcal{N}^+(s)$; therefore the action-value function is represented as a matrix $Q(s, s')$. The graph includes both local edges and nonlocal directed shortcuts, and the existence of $s \to s'$ does not imply the existence of $s' \to s$. Table 5 provides specific information. The left sub-figure in Figure 22 also demonstrates the same dynamics in a visual way.

Table 5: Possible transitions of the sixteen-state directed MDP.

s	$\mathcal{N}^+(s)$	s	$\mathcal{N}^+(s)$
0	$\varnothing$	8	$\{1, 4, 9, 12, 14\}$
1	$\{0, 2, 5, 9\}$	9	$\{2, 5, 8, 10, 15\}$
2	$\{1, 6, 8, 11\}$	10	$\{3, 6, 9, 11, 14\}$
3	$\{2, 7, 10, 14\}$	11	$\{4, 7, 10, 15\}$
4	$\{0, 5, 8, 11\}$	12	$\{0, 5, 8, 13\}$
5	$\{1, 4, 6, 10, 13\}$	13	$\{4, 7, 9, 12, 14\}$
6	$\{2, 5, 7, 9, 14\}$	14	$\{3, 6, 10, 13, 15\}$
7	$\{3, 6, 11, 13\}$	15	$\varnothing$

State 0 is a bad terminal with reward -8, whereas state 15 is a good terminal with reward $+12$. Every other transition has a base reward of -0.05. Entering one of the hazard states $\mathcal{H} = \{5, 10, 12\}$ incurs an additional penalty of -1.25. Several directed shortcuts also have edge-specific costs, listed in Table 6. Thus, for a nonterminal destination,

$$r(s, s') = -0.05 - 1.25\,\mathbb{I}[s' \in \mathcal{H}] + c(s, s'), \tag{53}$$

while transitions into states 0 and 15 receive their terminal rewards. The discount factor is $\gamma = 0.97$.

Table 6: Edge-specific costs $c(s, s')$.

Edge	Cost	Edge	Cost	Edge	Cost
$1 \to 9$	-0.70	$2 \to 11$	$+0.15$	$3 \to 14$	-0.40
$4 \to 11$	-0.20	$5 \to 13$	-0.60	$6 \to 14$	$+0.10$
$7 \to 13$	-0.30	$8 \to 1$	-0.50	$9 \to 2$	-0.40
$10 \to 3$	-0.20	$11 \to 4$	-0.60	$12 \to 5$	-0.25
$13 \to 4$	-0.35	$14 \to 3$	-0.45		

To simulate the inability of the offline dataset to provide optimal or suboptimal action coverage in complex environments, we design the offline dataset to tend to reach the bad terminal. Specifically, for every nonterminal state, we compute the shortest directed distance from each valid successor to state 0. A probability mass of 0.985 is distributed among the successors with minimum distance to state 0, while the remaining mass is distributed uniformly over all valid successors. This policy generates an offline dataset whose dominant trajectories terminate at state 0.

Furthermore, to simulate a policy that more closely resembles the one operating in the real world environment—rather than the dataset policy—after the actor has undergone fine-tuning during the online phase, we implement an online sampling strategy. Specifically, the prescribed online behavior is an ϵ-greedy policy with respect to the exact optimal Q-function of the finite MDP, with $\epsilon = 0.05$. It therefore predominantly follows high-value routes toward the positive terminal state 15. This policy is used only to generate state-action samples that model the rapid actor-distribution shift observed after a small number of online updates.

For clarity, the implementation uses three distinct datasets:

1. **Offline dataset** $\mathcal{D}_{\text{off}}$: generated by the policy attracted to terminal state 0; used to fit the offline flow and the offline critic.
2. **Post-shift actor dataset** $\mathcal{D}_{\text{actor}}$: generated by the prescribed near-optimal online behavior.
3. **Online critic replay** $\mathcal{D}_{\text{replay}}$: generated by

$$\pi_{\text{replay}} = 0.70\,\pi_{\text{near-optimal}} + 0.30\,\pi_{\text{uniform}}, \tag{54}$$

where π_{uniform} is uniform over valid outgoing edges.

By decoupling the online actor and the online critic (regardless of whether offline pre-training is employed), we ensure that the actor remains high-performing and reliable during the online phase;

this prevents low-quality offline data from impairing the actor's exploration capabilities—and, consequently, the training of the online critic. Similarly, by incorporating a uniform policy into the online critic's training data, we simulate the presence of low-quality data that might otherwise affect the critic's training.

Each dataset contains 35,000 episodes. An episode begins from a uniformly sampled nonterminal state and ends at a terminal state or after 80 transitions. Truncated transitions are treated as terminal in the TD mask.

F.2 Actor

Each discrete destination state is embedded using its normalized two-dimensional canvas coordinate in $[-1, 1]^2$. Given a state s, Gaussian noise x_0, and flow time t, the flow network predicts a velocity field. For a dataset action x_1, Flow Matching minimizes

$$x_t = (1 - t)x_0 + tx_1, \tag{55}$$

$$L_{\text{Flow}} = \mathbb{E}\Big[\|v_\theta(s, x_t, t) - (x_1 - x_0)\|_2^2\Big]. \tag{56}$$

The flow network contains two hidden layers of width 128. At inference time, we use 10 Euler integration steps, clip the continuous action to $[-1, 1]^2$, and project it to the closest valid destination of the current state.

We first train an offline flow $\hat{\pi}_{\text{off}}$ on $\mathcal{D}_{\text{off}}$. Starting from this fitted flow, we then perform 500 additional flow updates on $\mathcal{D}_{\text{actor}}$ and freeze the resulting post-shift policy $\hat{\pi}_{\text{on}}$.

F.3 Critic

The critic concatenates a one-hot state vector with the two-dimensional destination embedding, applies a single hidden layer of width 48 with a $\tanh$ activation, and outputs one scalar Q-value. The critic minimizes

$$L_Q(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D},a'\sim\hat{\pi}(\cdot|s')}\left[\left(Q_\phi(s, a) - \left[r + \gamma Q_{\bar{\phi}}(s', a')\right]\right)^2\right] \tag{57}$$

The target parameters are updated with $\tau = 0.02$.

F.4 Implementation

The complete training pipeline is as follows.

1. Initialize one flow network and fit it to $\mathcal{D}_{\text{off}}$, obtaining $\hat{\pi}_{\text{off}}$.
2. Initialize a small critic once and save its random parameters ϕ_{init}.
3. Train that critic for 500 TD updates on $\mathcal{D}_{\text{off}}$ using $\hat{\pi}_{\text{off}}$ in the TD target, obtaining ϕ_{off}.
4. Adapt the flow policy on $\mathcal{D}_{\text{actor}}$ and freeze the resulting post-shift policy $\hat{\pi}_{\text{on}}$.
5. Draw one common sequence of online-replay minibatches and one common sequence of Gaussian policy-noise keys.
6. Run O2O from ϕ_{off} and CFP from the saved ϕ_{init}. Then perform 500 TD updates on the same replay using the same frozen $\hat{\pi}_{\text{on}}$, minibatches, and policy noises.

F.5 Hyperparameters

Table 7 summarizes the hyperparameters used in our toy example.

F.6 Results and Interpretation

We calculated the target Q-values for the offline and online phases based on offline and online strategies, respectively. We also evaluate the Q-values produced by the O2O and CFP critics. The left panels of Figure 23 display the theoretical target Q-values for the offline and online settings, as well as the Q-values output by the O2O and CFP critics, respectively. The right side shows heatmaps

Table 7: Hyperparameters of the toy example.

Hyperparameter	Value	Hyperparameter	Value
Batch size	256	Discount factor γ	0.97
Offline episodes	35,000	Online actor episodes	35,000
Online replay episodes	35,000	Maximum episode length	80
Offline-policy greed	0.985	Online-policy ϵ	0.05
Uniform replay mixture	0.30	Flow steps	10
Actor hidden widths	[128, 128]	Actor learning rate	3×10^{-4}
Critic hidden width	48	Critic learning rate	10^{-3}
Offline updates	500	Online updates	500
Target update τ	0.02	RMSE recording interval	10

representing the absolute differences between pairs of the plots on the left; areas appearing more gray indicate greater similarity, while darker blue areas indicate larger differences. The O2O critic output is significantly closer to the target value of the offline dataset than the CFP critic output. Conversely, the CFP critic output is closer to the online target Q-value.

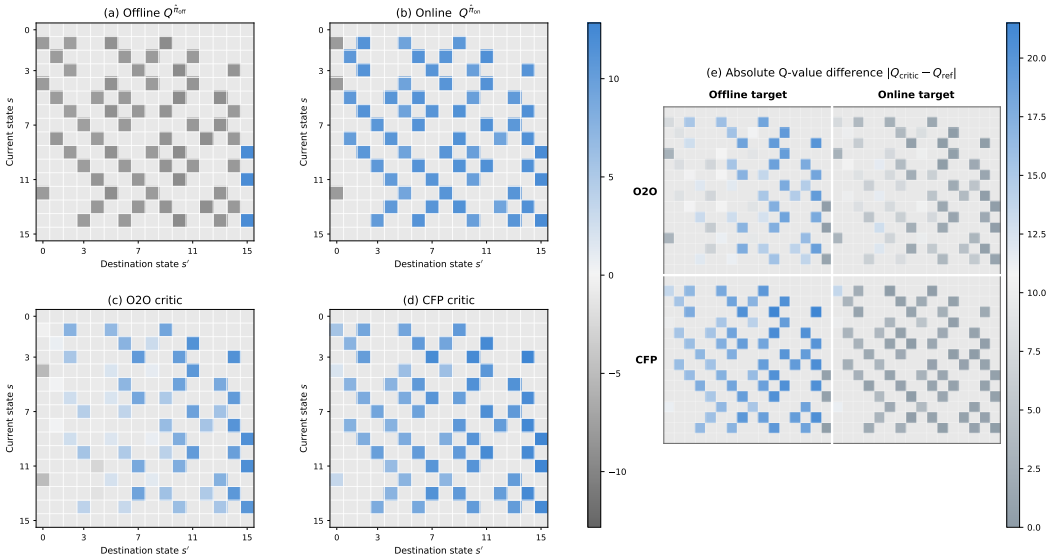


Figure 23: Results of toy example.