Appendices

A Implementation details of Cal-QL

Our algorithm, Cal-QL is illustrated in Algorithm 1. A complete implementation of the functions in python-style is provided in Appendix A.2.

A.1 Cal-QL Algorithm

We use $J_Q(\theta)$ to denote the calibrated conservative regularizer for the Q network update:

$$J_Q(\theta) := \alpha \underbrace{\left(\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[\max\left(Q_\theta(s,a), Q^\mu(s,a)\right)\right] - \mathbb{E}_{s,a\sim\mathcal{D}}\left[Q_\theta(s,a)\right]\right)}_{\text{Calibrated conservative regularizer } \mathcal{R}(\theta)} \tag{A.1}$$

$$+ \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - \mathcal{B}^\pi\bar{Q}(s,a)\right)^2\right]. \tag{A.2}$$

Algorithm 1 Cal-QL pseudo-code

1: Initialize Q-function, Q_θ, a policy, π_ϕ
2: **for** step t in $\{1, \ldots, \mathrm{N}\}$ **do**
3: Train the Q-function using the conservative regularizer in Eq. A.1:

$$\theta_t := \theta_{t-1} - \eta_Q \nabla_\theta J_Q(\theta) \tag{A.3}$$

4: Improve policy π_ϕ with SAC-style update:

$$\phi_t := \phi_{t-1} + \eta_\pi \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\phi(\cdot|s)}[Q_\theta(s,a) - \log \pi_\phi(a|s)] \tag{A.4}$$

5: **end for**

A.2 Python Implementation

Listing 1: Training Q networks given a batch of data

```
q_data = critic(batch['observations'], batch['actions'])

next_dist = actor(batch['next_observations'])
next_pi_actions, next_log_pis = next_dist.sample()

target_qval = target_critic(batch['observations'], next_pi_actions)
target_qval = batch['rewards'] + self.gamma * (1 - batch['dones']) * target_qval

td_loss = mse_loss(q_data, target_qval)

num_samples = 4
random_actions = uniform((num_samples, batch_size, action_dim), min=-1, max=1)
random_pi = 0.5 ** batch['actions'].shape[-1]

pi_actions, log_pis = actor(batch['observations'])

q_rand_is = critic(batch['observations'], random_actions) - random_pi
q_pi_is = critic(batch['observations'], pi_actions) - log_pis

# Cal-QL's modification
mc_return = batch['mc_return'].repeat(num_samples)
q_pi_is = max(q_pi_is, mc_return)

cat_q = concatenate([q_rand_is, q_pi_is], new_axis=True)
cat_q = logsumexp(cat_q, axis=0) # sum over num_samples
critic_loss = td_loss + ((cat_q - q_data).mean() * cql_alpha)

critic_optimizer.zero_grad()
critic_loss.backward()
critic_optimizer.step()
```

Listing 2: Training the policy (or the actor) given a batch of data

```
# return distribution of actions
pi_actions, log_pis = actor(batch['observations'])

# calculate q value of actor actions
qpi = critic(batch['observations'], actions)
qpi = qpi.min(axis=0)

# same objective as CQL (kumar et al.)
actor_loss = (log_pis * self.alpha - qpi).mean()

# optimize loss
actor_optimizer.zero_grad()
actor_loss.backward()
actor_optimizer.step()
```

B Regret Analysis of Cal-QL

We provide a theoretical version of Cal-QL in Algorithm 2. Policy fine-tuning has been studied in different settings [60, 52, 55]. Our analysis largely adopts the settings and results in Song et al. [52], with additional changes in Assumption B.1, Assumption B.3, and Definition B.4. Note that the goal of this proof is to demonstrate that *a pessimistic functional class (Assumption B.1)* allows one to utilize the offline data efficiently, rather than providing a new analytical technique for regret analysis. See comparisons between Section B.3 and Section C.1. Note that we use f instead of Q_θ in the main text to denote the estimated Q function for notation simplicity.

Algorithm 2 Theoretical version of Cal-QL

Input: Value function class $\mathcal{F}$, # total iterations K, offline dataset $\mathcal{D}^\nu_h$ of size m_{off} for $h \in [H-1]$.
Initialize $f^1_h(s,a) = 0, \forall (s,a)$.
for $k = 1, \ldots, K$ **do**
  Let π^t be the greedy policy w.r.t. f^k ▷ I.e., $\pi^k_h(s) = \arg\max_a f^k_h(s,a)$.
  For each h, collect m_{on} online tuples $\mathcal{D}^k_h \sim d^{\pi^k}_h$ ▷ online data collection
  Set $f^{k+1}_H(s,a) = 0, \forall (s,a)$.
  for $h = H-1, \ldots 0$ **do** ▷ FQI with offline and online data
    Estimate f^{k+1}_h using conservative least squares on the aggregated data: ▷ I.e., CQL regularized class $\mathcal{C}_h$

$$f^{k+1}_h \leftarrow \arg\min_{f \in \mathcal{C}_h} \left\{ \widehat{\mathbb{E}}_{\mathcal{D}^\nu_h} \left[f(s,a) - r - \max_{a'} f^{k+1}_{h+1}(s',a') \right]^2 + \sum_{\tau=1}^{K} \widehat{\mathbb{E}}_{\mathcal{D}^\tau_h} \left[f(s,a) - r - \max_{a'} f^{k+1}_{h+1}(s',a') \right]^2 \right\} \tag{B.1}$$

    $f^{k+1}_h = \max\{f^{k+1}_h, Q^{\text{ref}}_h\}$ ▷ Set a reference policy for calibration (Definition 4.1)
  end for
end for
Output: π^K

B.1 Preliminaries

In this subsection, we follow most of the notations and definitions in Song et al. [52]. In particular, we consider the finite horizon cases, where the value function and Q function are defined as:

$$V^\pi_h(s) = \mathbb{E}\left[\sum_{\tau=h}^{H-1} r_\tau | \pi, s_h = s \right] \tag{B.2}$$

$$Q^\pi_h(s,a) = \mathbb{E}\left[\sum_{\tau=h}^{H-1} r_\tau | \pi, s_h = s, a_h = a \right]. \tag{B.3}$$

We also define the Bellman operator $\mathcal{T}$ such that $\forall f : \mathcal{S} \times \mathcal{A}$:

$$\mathcal{T} f(s,a) = \mathbb{E}_{s,a}[R(s,a)] + \mathbb{E}_{s' \sim P(s,a)} \max_{a'} f(s',a'), \ \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \tag{B.4}$$

where $R(s,a) \in \Delta[0,1]$ represents a stochastic reward function.

B.2 Notations

- Feature covariance matrix $\Sigma_{k;h}$:

$$\mathbf{\Sigma}_{k;h} = \sum_{\tau=1}^{k} X_h(f^\tau)(X_h(f^\tau))^\top + \lambda \boldsymbol{I} \tag{B.5}$$

- Matrix Norm Zanette et al. [63]: for a matrix Σ, the matrix norm $\|\boldsymbol{u}\|_{\mathbf{\Sigma}}$ is defined as:

$$\|\boldsymbol{u}\|_{\mathbf{\Sigma}} = \sqrt{\boldsymbol{u}\mathbf{\Sigma}\boldsymbol{u}^\top} \tag{B.6}$$

- Weighted ℓ^2 norm: for a given distribution $\beta \in \Delta(\mathcal{S} \times \mathcal{A})$ and a function $f : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, we denote the weighted ℓ^2 norm as:

$$\|f\|_{2,\beta}^2 := \sqrt{\mathbb{E}_{(s,a)\sim\beta} f^2(s,a)} \tag{B.7}$$

- A stochastic reward function $R(s,a) \in \Delta([0,1])$
- For each offline data distribution $\nu = \{\nu_0, \ldots, \nu_{H-1}\}$, the offline data set at time step h (ν_h) contains data samples (s,a,r,s'), where $(s,a) \sim \nu_h, r \in R(s,a), s' \sim P(s,a)$.
- Given a policy $\pi := \{\pi_0, \ldots, \pi_{H-1}\}$, where $\pi_h : \mathcal{S} \mapsto \Delta(\mathcal{A})$, $d_h^\pi \in \Delta(s,a)$ denotes the state-action occupancy induced by π at step h.
- We consider the value-based function approximation setting, where we are given a function class $\mathcal{C} = \mathcal{C}_0 \times \ldots \mathcal{C}_{H-1}$ with $\mathcal{C}_h \subset \mathcal{S} \times \mathcal{A} \mapsto [0, V_{\max}]$.
- A policy π^f is defined as the greedy policy w.r.t. f: $\pi_h^f(s) = \arg\max_a f_h(s,a)$. Specifically, at iteration k, we use π^k to denote the greedy policy w.r.t. f^k.

B.3 Assumptions and Defintions

Assumption B.1 (Pessimistic Realizability and Completeness)**.** *For any policy π^e, we say $\mathcal{C}_h$ is a pessimistic function class w.r.t. π^e, if for any h, we have $Q_h^{\pi^e} \in \mathcal{C}_h$, and additionally, for any $f_{h+1} \in \mathcal{C}_{h+1}$, we have $\mathcal{T} f_{h+1} \in \mathcal{C}_h$ and $f_h(s,a) \leq Q_h^{\pi^e}(s,a), \forall (s,a) \in \mathcal{S} \times \mathcal{A}$.*

Definition B.2 (Bilinear model Du et al. [9])**.** *We say that the MDP together with the function class $\mathcal{F}$ is a bilinear model of rand d of for any $h \in [H-1]$, there exist two (known) mappings $X_h, W_h : \mathcal{F} \mapsto \mathbb{R}^d$ with $\max_f \|X_h(f)\|_2 \leq B_X$ and $\max_f \|W_h(f)\|_2 \leq B_W$ such that*

$$\forall f, g \in \mathcal{F} : \quad \left|\mathbb{E}_{s,a\sim d_h^{\pi^f}}[g_h(s,a) - \mathcal{T} g_{h+1}(s,a)]\right| = |\langle X_h(f), W_h(g)\rangle| . \tag{B.8}$$

Assumption B.3 (Bilinear Rank of Reference Policies)**.** *Suppose $Q^{\text{ref}} \in \mathcal{C}_{\text{ref}} \subset \mathcal{C}$, where $\mathcal{C}_{\text{ref}}$ is the function class of our reference policy, we assume the Bilinear rank of $\mathcal{C}_{\text{ref}}$ is d_{ref} and $d_{\text{ref}} \leq d$.*

Definition B.4 (Calibrated Bellman error transfer coefficient)**.** *For any policy π, we define the calibrated transfer coefficient w.r.t. to a reference policy π^{ref} as*

$$C_\pi^{\text{ref}} := \max_{f\in\mathcal{C}, f(s,a)\geq Q^{\text{ref}}(s,a)} \frac{\sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d_h^\pi}[\mathcal{T} f_{h+1}(s,a) - f_h(s,a)]}{\sqrt{\sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim\nu_h}(\mathcal{T} f_{h+1}(s,a) - f_h(s,a))^2}}, \tag{B.9}$$

where $Q^{\text{ref}} = Q^{\pi^{\text{ref}}}$.

B.4 Discussions on the Assumptions

The pessimistic realizability and completeness assumption (Assumption B.1) is motivated by some theoretical studies of the pessimistic offline methods [59, 6] with regularizers:

$$\min_\theta \alpha \underbrace{\left(\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[Q_\theta(s,a)\right] - \mathbb{E}_{s,a\sim\mathcal{D}}\left[Q_\theta(s,a)\right]\right)}_{\text{Conservative regularizer } \mathcal{R}(\theta)} + \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - \mathcal{B}^\pi \bar{Q}(s,a)\right)^2\right]. \tag{B.10}$$

Since the goal of the conservative regularizer $\mathcal{R}(\theta)$ intrinsically wants to enforce

$$Q_\theta(s, \pi(s)) \leq Q_\theta(s, \pi^e(s)), \tag{B.11}$$

where π is the training policy and π^e is the reference (behavior) policy. One can consider (B.10) as the Lagrange duality formulation of the following primal optimization problem:

$$\min_\theta \mathbb{E}_{s,a,s' \sim \mathcal{D}} \left[\left(Q_\theta(s,a) - \mathcal{B}^\pi \bar{Q}(s,a) \right)^2 \right], \text{ subject to } \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi} \left[Q_\theta(s,a) \right] \leq \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi^e} \left[Q_\theta(s,a) \right], \tag{B.12}$$

where the constraint set is equivalent to Assumption B.1. Although Assumption B.1 directly characterizes the constraint set of the primal form of (B.10) the exact theoretical connection between the pessimistic realizability and the regularized bellman consistency equation is beyond the scope of this work and we would like to leave that for future studies.

Assumption B.1 allows us to restrict the functional class of interest to a smaller conservative function class $\mathcal{C} \subset \mathcal{F}$, which leads to a smaller Bellman rank of the reference policy ($d_{\text{ref}} \leq d$) suggested in Assumption B.3, and a smaller concentrability coefficient ($C_\pi^{\text{ref}} \leq C_\pi$) defined in Definition B.4, and C.2. Assumption B.3 and Definition C.2 provide the Bellman Bilinear rank and Bellman error transfer coefficient of the pessimistic functional class $\mathcal{C}$ of interest.

B.5 Proof Structure Overview

We provide an overview of the proof structure and its dependency on different assumptions below:

- Theorem B.5: the total regret is decomposed into *offline regrets* and *online regrets*.
  - Bounding *offline regrets*, requiring Definition B.4 and the following lemmas:
    - Performance difference lemma w.r.t. a comparator policy (Lemma C.5).
    - Least square generalization bound (Lemma C.4), requiring Assumption B.1.
  - Bounding *online regrets*, requiring Definition B.2
    - Performance difference lemma for the online error (Lemma C.6).
    - Least square generalization bound (Lemma C.4), requiring Assumption B.1.
    - Upper bounds with the bilinear model assumption (Lemma C.7).
    - Applying Elliptical Potential Lemma [35] with bellman rank d and d_{ref} (Lemma C.8), requiring Assumption B.3.

B.6 Our Results

Theorem B.5 (Formal Result of Theorem 6.1)**.** *Fix $\delta \in (0,1), m_{\text{off}} = K, m_{\text{on}} = 1$, suppose and the function class $\mathcal{C}$ follows Assumption B.1 w.r.t. π^e. Suppose the underlying MDP admits Bilinear rank d on function class $\mathcal{C}$ and d_{ref} on $\mathcal{C}_{\text{ref}}$, respectively, then with probability at least $1-\delta$, Algorithm 2 obtains the following bound on cumulative suboptimality w.r.t. any comparator policy π^e:*

$$\sum_{t=1}^K V^{\pi^e} - V^{\pi^k} = \widetilde{O}\left(\min\left\{ C_{\pi^e}^{\text{ref}} H \sqrt{dK \log\left(|\mathcal{F}|/\delta\right)},\ K\left(V^{\pi^e} - V^{\text{ref}}\right) + H\sqrt{d_{\text{ref}} K \log\left(|\mathcal{F}|/\delta\right)} \right\} \right). \tag{B.13}$$

Note that Theorem B.5 provides a guarantee for *any* comparator policy π^{e}, which can be directly applied to $\pi^\star$ described in our informal result (Theorem 6.1). We also change the notation for the reference policy from μ in Theorem 6.1 to π^{ref} (similarly, $d_{\text{ref}}, V^{\text{ref}}, C_{\pi^e}^{\text{ref}}$ correspond to $d_\mu, V^\mu, C_{\pi^e}^\mu$ in Theorem 6.1) for notation consistency in the proof. Our proof of Theorem B.5 largely follows the proof of Theorem 1 of [52], and the major changes are caused by Assumption B.1, Assumption B.3, and Definition B.4.

Proof. Let $\mathcal{E}_k$ denote the event that $\left\{f_0^k(s,a) \le Q^{\text{ref}}(s,a)\right\}$ and $\bar{\mathcal{E}}_k$ denote the event that $\left\{f_0^k(s,a) > Q^{\text{ref}}(s,a)\right\}$. Let $V^{\text{ref}}(s) = \max_a Q^{\text{ref}}(s,a)$, we start by noting that

$$\begin{aligned}
&\sum_{k=1}^{K} V^{\pi^e} - V^{\pi^{f^k}} = \sum_{k=1}^{K} \mathbb{E}_{s\sim\rho}\left[V_0^{\pi^e}(s) - V_0^{\pi^{f^k}}(s)\right] \\
&= \underbrace{\sum_{k=1}^{K} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\left(V_0^{\pi^e}(s) - V^{\text{ref}}(s)\right)\right]}_{\Gamma_0} + \underbrace{\sum_{k=1}^{K} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\left(V^{\text{ref}}(s) - \max_a f_0^k(s,a)\right)\right]}_{=0, \text{ by the definition of } \bar{\mathcal{E}}_k \text{ and line 9 of Algorithm 2}} \\
&\quad + \underbrace{\sum_{t=1}^{K} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\left(\max_a f_0^k(s,a) - V_0^{\pi^{f^k}}(s)\right)\right]}_{\Gamma_1} + \underbrace{\sum_{k=1}^{K} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\mathcal{E}_k\right\}\left(V_0^{\pi^e}(s) - \max_a f_0^k(s,a)\right)\right]}_{\Gamma_2} \\
&\quad + \underbrace{\sum_{t=1}^{T} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\mathcal{E}_k\right\}\left(\max_a f_0^k(s,a) - V_0^{\pi^{f^k}}(s)\right)\right]}_{\Gamma_3}.
\end{aligned} \tag{B.14}$$

Let $K_1 = \sum_{k=1}^{K} \mathbb{1}\left\{f_0^k(s,a) > Q^{\text{ref}}(s,a)\right\}$ and $K_2 = \sum_{k=1}^{K} \mathbb{1}\left\{f_0^k(s,a) \le Q^{\text{ref}}(s,a)\right\}$ (or equivalently $K_1 = \sum_{k=1}^{K} \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}$, $K_2 = \sum_{k=1}^{K} \mathbb{1}\left\{\mathcal{E}_k\right\}$). For Γ_0, we have

$$\Gamma_0 = K_2 \mathbb{E}_{s\sim\rho}\left(V^{\pi^e}(s) - V^{\text{ref}}(s)\right). \tag{B.15}$$

For Γ_2, we have

$$\begin{aligned}
\Gamma_2 &= \sum_{k=1}^{K} \mathbb{E}_{s\sim\rho}\left[\mathbb{1}\left\{\mathcal{E}_k\right\}\left(V_0^{\pi^e}(s) - \max_a f_0^k(s,a)\right)\right] \\
&\overset{(i)}{\le} \sum_{k=1}^{K} \mathbb{1}\left\{\mathcal{E}_k\right\} \sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d_h^{\pi^e}}\left[\mathcal{T} f_{h+1}^k(s,a) - f_h^k(s,a)\right] \\
&\overset{(ii)}{\le} \sum_{k=1}^{K}\left[C_{\pi^e}^{\text{ref}} \cdot \mathbb{1}\left\{\mathcal{E}_k\right\} \sqrt{\sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim\nu_h}\left[\left(f_h^k(s,a) - \mathcal{T} f_{h+1}^k(s,a)\right)^2\right]}\right] \\
&\overset{(iii)}{\le} K_1 C_{\pi^e}^{\text{ref}} \sqrt{H \cdot \Delta_{\text{off}}},
\end{aligned} \tag{B.16}$$

where Δ_{off} is similarly defined as Song et al. [52] (See (C.3) of Lemma C.4). Inequality (i) holds because of Lemma C.5, inequality (ii) holds by the definition of $C_{\pi^e}^{\text{ref}}$ (Definition B.4), inequality (iii) holds by applying Lemma C.4 with the function class satisfying Assumption B.1, and Definition B.4. Note that the telescoping decomposition technique in the above equation also appears in [58, 24, 9]. Next, we will bound $\Gamma_1 + \Gamma_3$:

$$\begin{aligned}
\Gamma_1 + \Gamma_3 &= \sum_{k=1}^{K}\left(\mathbb{1}\left\{\mathcal{E}_k\right\} + \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\right) \mathbb{E}_{s\sim d_0}\left[\max_a f_0^k(s,a) - V_0^{\pi^{f^k}}(s)\right] \\
&\overset{(i)}{\le} \sum_{k=1}^{K}\left(\mathbb{1}\left\{\mathcal{E}_k\right\} + \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\right) \sum_{h=0}^{H-1}\left|\mathbb{E}_{s,a\sim d_h^{\pi^{f^k}}}\left[f_h^k(s,a) - \mathcal{T} f_{h+1}^k(s,a)\right]\right| \\
&\overset{(ii)}{=} \sum_{t=1}^{K}\left[\left(\mathbb{1}\left\{\mathcal{E}_k\right\} + \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\right) \sum_{h=0}^{H-1}\left|\left\langle X_h(f^k), W_h(f^k)\right\rangle\right|\right] \\
&\overset{(iii)}{\le} \sum_{k=1}^{K}\left[\left(\mathbb{1}\left\{\mathcal{E}_k\right\} + \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\}\right) \sum_{h=0}^{H-1}\left\|X_h(f^k)\right\|_{\mathbf{\Sigma}_{k-1;h}^{-1}} \sqrt{\Delta_{\text{on}} + \lambda B_W^2}\right],
\end{aligned} \tag{B.17}$$

where Δ_{on} is similarly defined as Song et al. [52] (See (C.4) of Lemma C.4). Inequality (i) holds by Lemma C.6, equation (ii) holds by the definition of Bilinear model ((B.8) in Definition B.2), inequality (iii) holds by Lemma C.7 and Lemma C.4 with the function class satisfying Assumption B.1. Using Lemma C.8, we have that

$$\begin{aligned}
&\Gamma_1 + \Gamma_3 \\
&\leq \sum_{k=1}^{K} \left[\left(\mathbb{1}\left\{\mathcal{E}_k\right\} + \mathbb{1}\left\{\bar{\mathcal{E}}_k\right\} \right) \sum_{h=0}^{H-1} \left\| X_h(f^k) \right\|_{\mathbf{\Sigma}_{k-1;h}^{-1}} \sqrt{\Delta_{\text{on}} + \lambda B_W^2} \right] \\
&\overset{(i)}{\leq} H \sqrt{2d \log\left(1 + \frac{K_1 B_X^2}{\lambda d}\right) \cdot (\Delta_{\text{on}} + \lambda B_W^2) \cdot K_1} + H \sqrt{2d_{\text{ref}} \log\left(1 + \frac{K_2 B_X^2}{\lambda d_{\text{ref}}}\right) \cdot (\Delta_{\text{on}} + \lambda B_W^2) \cdot K_2} \\
&\overset{(ii)}{\leq} H \left(\sqrt{2d \log\left(1 + \frac{K_1}{d}\right) \cdot (\Delta_{\text{on}} + B_X^2 B_W^2) \cdot K_1} + \sqrt{2d_{\text{ref}} \log\left(1 + \frac{K_2}{d_{\text{ref}}}\right) \cdot (\Delta_{\text{on}} + B_X^2 B_W^2) \cdot K_2} \right),
\end{aligned} \tag{B.18}$$

where the first part of inequality (i) holds by the assumption that the underlying MDPs have bellman rank d (Definition B.2) when $\bar{\mathcal{E}}_k$ happens, and the second part of inequality (i) holds by the assumption that $\mathcal{C}_{\text{ref}}$ has bilinear rank d_{ref} (Assumption B.3) $\mathcal{C}_{\text{ref}}$ has bellman rank d_{ref} when $\mathcal{E}_k$ happens. Inequality (ii) holds by plugging in $\lambda = B_X^2$. Substituting (B.15), inequality B.16, and inequality (B.18) into (B.14), we have

$$\begin{aligned}
&\sum_{t=1}^{K} V^{\pi^e} - V^{\pi^{f^k}} \leq \Gamma_0 + \Gamma_2 + \Gamma_1 + \Gamma_3 \leq K_2 \left(V^{\pi^e}(s) - V^{\text{ref}}(s) \right) + K_1 C_{\pi^e}^{\text{ref}} \sqrt{H \cdot \Delta_{\text{off}}} \\
&+ H \left(\sqrt{2d \log\left(1 + \frac{K_1}{d}\right) \cdot (\Delta_{\text{on}} + B_X^2 B_W^2) \cdot K_1} + \sqrt{2d_{\text{ref}} \log\left(1 + \frac{K_2}{d_{\text{ref}}}\right) \cdot (\Delta_{\text{on}} + B_X^2 B_W^2) \cdot K_2} \right)
\end{aligned} \tag{B.19}$$

Plugging in the values of $\Delta_{\text{on}}, \Delta_{\text{off}}$ from (C.3) and (C.4), and using the subadditivity of the square root function, we have

$$\begin{aligned}
&\sum_{k=1}^{K} V^{\pi^e} - V^{\pi^{f^k}} \\
&\leq K_2 \left(V^{\pi^e}(s) - V^{\text{ref}}(s) \right) + 16 V_{\max} C_{\pi^e}^{\text{ref}} K_1 \sqrt{\frac{H}{m_{\text{off}}} \log\left(\frac{2HK_1|\mathcal{F}|}{\delta}\right)} \\
&\quad + \left(16 V_{\max} \sqrt{\frac{1}{m_{\text{on}}} \log\left(\frac{2HK_1|\mathcal{F}|}{\delta}\right)} + B_X B_W \right) \cdot H \sqrt{2dK_1 \log\left(1 + \frac{K_1}{d}\right)} \\
&\quad + \left(16 V_{\max} \sqrt{\frac{1}{m_{\text{on}}} \log\left(\frac{2HK_2|\mathcal{F}|}{\delta}\right)} + B_X B_W \right) \cdot H \sqrt{2d_{\text{ref}} K_2 \log\left(1 + \frac{K_2}{d_{\text{ref}}}\right)}.
\end{aligned} \tag{B.20}$$

Setting $m_{\text{off}} = K, m_{\text{on}} = 1$ in the above equation completes the proof, we have

$$\begin{aligned}
&\sum_{k=1}^{K} V^{\pi^e} - V^{\pi^k} \\
&\leq \widetilde{O}\left(C_{\pi^e}^{\text{ref}} \sqrt{HK_1 \log\left(|\mathcal{F}|/\delta\right)} \right) + \widetilde{O}\left(H \sqrt{dK_1 \log\left(|\mathcal{F}|/\delta\right)} \right) \\
&\quad + K_2 \left(V^{\pi^e}(s) - V^{\text{ref}}(s) \right) + \widetilde{O}\left(H \sqrt{d_{\text{ref}} K_2 \log\left(|\mathcal{F}|/\delta\right)} \right) \\
&\leq \begin{cases} \widetilde{O}\left(C_{\pi^e}^{\text{ref}} H \sqrt{dK_1 \log\left(|\mathcal{F}|/\delta\right)} \right) & \text{if } K_1 \gg K_2, \\ \widetilde{O}\left(K_2 \left(V^{\pi^e} - V^{\text{ref}} \right) + H \sqrt{d_{\text{ref}} K_2 \log\left(|\mathcal{F}|/\delta\right)} \right) & \text{otherwise.} \end{cases} \\
&\leq \widetilde{O}\left(\min\left\{ C_{\pi^e}^{\text{ref}} H \sqrt{dK \log\left(|\mathcal{F}|/\delta\right)},\ K \left(V^{\pi^e} - V^{\text{ref}} \right) + H \sqrt{d_{\text{ref}} K \log\left(|\mathcal{F}|/\delta\right)} \right\} \right),
\end{aligned} \tag{B.21}$$

where the last inequality holds because $K_1, K_2 \leq K$, which completes the proof. □

C Key Results of HyQ [52]

In this section, we restate the major theoretical results of Hy-Q [52] for completeness.

C.1 Assumptions

Assumption C.1 (Realizability and Bellman completeness)**.** *For any h, we have $Q_h^\star \in \mathcal{F}_h$, and additionally, for any $f_{h+1} \in \mathcal{F}_{h+1}$, we have $\mathcal{T} f_{h+1} \in \mathcal{F}_h$.*

Definition C.2 (Bellman error transfer coefficient)**.** *For any policy π, we define the transfer coefficient as*

$$C_\pi := \max\left\{0, \max_{f\in\mathcal{F}} \frac{\sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d_h^\pi}[\mathcal{T} f_{h+1}(s,a) - f_h(s,a)]}{\sqrt{\sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim\nu_h}(\mathcal{T} f_{h+1}(s,a) - f_h(s,a))^2}}\right\}. \tag{C.1}$$

C.2 Main Theorem of Hy-Q

Theorem C.3 (Theorem 1 of Song et al. [52])**.** *Fix $\delta \in (0,1), m_{\text{off}} = K, m_{\text{on}} = 1$, and suppose that the underlying MDP admits Bilinear rank d (Definition B.2), and the function class $\mathcal{F}$ satisfies Assumption C.1. Then with probability at least $1-\delta$, HyQ obtains the following bound on cumulative suboptimality w.r.t. any comparator policy π^e:*

$$\text{Reg}(K) = \widetilde{O}\left(\max\{C_{\pi^e}, 1\} V_{\max} B_X B_W \sqrt{dH^2 K \cdot \log(|\mathcal{F}|/\delta)}\right). \tag{C.2}$$

C.3 Key Lemmas

C.3.1 Least Squares Generalization and Applications

Lemma C.4 (Lemma 7 of Song et al. [52], Online and Offline Bellman Error Bound for FQI)**.** *Let $\delta \in (0,1)$ and $\forall h \in [H-1], k \in [K]$, let f_h^{k+1} be the estimated value function for time step h computed via least square regression using samples in the dataset $\{\mathcal{D}_h^\nu, \mathcal{D}_h^1, \ldots, \mathcal{D}_h^T\}$ in (B.1) in the iteration t of Algorithm 2. Then with probability at least $1-\delta$, for any $h \in [H-1]$ and $k \in [K]$, we have*

$$\left\|f_h^{k+1} - \mathcal{T} f_{h+1}^{k+1}\right\|_{2,\nu_h}^2 \le \frac{1}{m_{\text{off}}} 256 V_{\max}^2 \log(2HK|\mathcal{F}|/\delta) =: \Delta_{\text{off}} \tag{C.3}$$

and

$$\sum_{\tau=1}^{k} \left\|f_h^{k+1} - \mathcal{T} f_{h+1}^{k+1}\right\|_{2,\mu_h^\tau}^2 \le \frac{1}{m_{\text{on}}} 256 V_{\max}^2 \log(2HK|\mathcal{F}|/\delta) =: \Delta_{\text{on}}, \tag{C.4}$$

where ν_h denotes the offline data distribution at time h, and the distribution $\mu_h^\tau \in \Delta(s,a)$ is defined such that $s,a \sim d_h^{\pi^\tau}$.

C.3.2 Bounding Offline Suboptimality via Performance Difference Lemma

Lemma C.5 (Lemma 5 of Song et al. [52], performance difference lemma of w.r.t. π^e)**.** *Let $\pi^e = (\pi_0^e, \ldots, \pi_{H-1}^e)$ be a comparator policy and consider any value function $f = (f_0, \ldots, f_{H-1})$, where $f_h : \mathcal{S}\times\mathcal{A} \mapsto \mathbb{R}$. Then we have*

$$\mathbb{E}_{s\sim d_0}\left[V_0^{\pi^e}(s) - \max_a f_0(s,a)\right] \le \sum_{i=1}^{H-1} \mathbb{E}_{s,a\sim d_i^{\pi^e}}\left[\mathcal{T} f_{i+1}(s,a) - f_i(s,a)\right], \tag{C.5}$$

where we define $f_H(s,a) = 0, \forall (s,a)$.

C.3.3 Bounding Online Suboptimality via Performance Difference Lemma

Lemma C.6 (Lemma 4 of Song et al. [52], performance difference lemma)**.** *For any function $f = (f_0, \ldots, f_{H-1})$ where $f_h : \mathcal{S}\times\mathcal{A} \mapsto \mathbb{R}$ and $h \in [H-1]$, we have*

$$\mathbb{E}_{s\sim d_0}\left[\max_a f_0(s,a) - V_0^{\pi^f}(s)\right] \le \sum_{h=0}^{H-1} \left|\mathbb{E}_{s,a\sim d_h^{\pi^f}}\left[f_h(s,a) - \mathcal{T} f_{h+1}(s,a)\right]\right|, \tag{C.6}$$

where we define $f_H(s,a) = 0, \forall s,a$.

Lemma C.7 (Lemma 8 of Song et al. [52], upper bounding bilinear class)**.** *For any $k \geq 2$ and $h \in [H-1]$, we have*

$$\left|\left\langle W_h(f^k), X_h(f^k)\right\rangle\right| \leq \left\|X_h(f^k)\right\|_{\mathbf{\Sigma}_{k-1;h}^{-1}} \sqrt{\sum_{i=1}^{k-1} \mathbb{E}_{s,a\sim d_h^{f^i}}\left[\left(f_h^k - \mathcal{T} f_{h+1}^k\right)^2\right] + \lambda B_W^2}, \tag{C.7}$$

where $\mathbf{\Sigma}_{k-1;h}$ is defined as (B.5) and we use $d_h^{f^k}$ to denote $d_h^{\pi_{f^k}}$.

Lemma C.8 (Lemma 6 of Song et al. [52], bounding the inverse covariance norm)**.** *Let $X_h(f^1), \ldots, X_h(f^K) \in \mathbb{R}^d$ be a sequence of vectors with $\left\|X_h(f^k)\right\|_2 \leq B_X < \infty, \forall k \leq K$. Then we have*

$$\sum_{k=1}^{K} \left\|X_h(f^k)\right\|_{\mathbf{\Sigma}_{k-1;h}^{-1}} \leq \sqrt{2dK \log\left(1 + \frac{KB_X^2}{\lambda d}\right)}, \tag{C.8}$$

where we define $\mathbf{\Sigma}_{k;h} := \sum_{\tau=1}^{k} X_h(f^\tau) X_h(f^\tau)^T + \lambda \boldsymbol{I}$ and we assume $\lambda \geq B_X^2$ holds $\forall k \in [K]$.

D Environment Details

D.1 Antmaze

The Antmaze navigation tasks aim to control an 8-DoF ant quadruped robot to move from a starting point to a desired goal in a maze. The agent will receive sparse +1/0 rewards depending on whether it reaches the goal or not. We study each method on "medium" and "hard" (shown in Figure 5) mazes which are difficult to solve, using the following datasets from D4RL [10]: `large-diverse`, `large-play`, `medium-diverse`, and `medium-play`. The difference between "diverse" and "play" datasets is the optimality of the trajectories they contain. The "diverse" datasets contain the trajectories commanded to a random goal from random starting points, while the "play" datasets contain the trajectories commanded to specific locations which are not necessarily the goal. We used an episode length of 1000 for each task. For Cal-QL, CQL, and IQL, we pre-trained the agent using the offline dataset for 1M steps. We then trained online fine-tuning for 1M environment steps for each method.

D.2 Franka Kitchen

The Franka Kitchen domain require controlling a 9-DoF Franka robot to arrange a kitchen environment into a desired configuration. The configuration is decomposed into 4 subtasks, and the agent will receive rewards of $0, +1, +2, +3$, or $+4$ depending on how many subtasks it has managed to solve. To solve the whole task and reach the desired configuration, it is important to learn not only how to solve each subtask, but also to figure out the correct order to solve. We study this domain using datasets with three different optimalities: `kitchen-complete`, `kitchen-partial`, and `kitchen-mixed`. The "complete" dataset contains the trajectories of the robot performing the whole task completely. The "partial" dataset partially contains some complete demonstrations, while others are incomplete demonstrations solving the subtasks. The "mixed" dataset only contains incomplete data without any complete demonstrations, which is hard and requires the highest degree of stitching and generalization ability. We used an episode length of 1000 for each task. For Cal-QL, CQL, and IQL, we pre-trained the agent using the offline dataset for 500K steps. We then performed online fine-tuning for 1.25M environment steps for each method.

D.3 Adroit

The Adroit domain involves controlling a 24-DoF shadow hand robot. There are 3 tasks we consider in this domain: `pen-binary`, `relocate-binary`, `relocate-binary`. These tasks comprise a limited set of narrow human expert data distributions (~ 25) with additional trajectories collected by a behavior-cloned policy. We truncated each trajectory and used the positive segments (terminate when the positive reward signal is found) for all methods. This domain has a very narrow dataset distribution and a large action space. In addition, learning in this domain is difficult due to the sparse reward, which leads to exploration challenges. We utilized a variant of the dataset used in prior work [44] to have a standard comparison with SOTA offline fine-tuning experiments that consider this domain. For the offline learning phase, we pre-trained the agent for 20K steps. We then performed online fine-tuning for 300K environment steps for the `pen-binary` task, and 1M environment steps for the `door-binary` and `relocate-binary` tasks. The episode length is 100, 200, and 200 for `pen-binary`, `door-binary`, and `relocate-binary` respectively.

D.4 Visual Manipulation Domain

The Visual Manipulation domain consists of a pick-and-place task. This task is a multitask formulation explored in the work, Pre-training for Robots (PTR) [33]. Here each task is defined as placing an object in a bin. A distractor object was present in the scene as an adversarial object which the agent had to avoid picking. There were 10 unique objects and no overlap between the task objects and the interfering/distractor objects. The episode length is 40. For the offline phase, we pre-trained the policy with offline data for 50K steps. We then performed online fine-tuning for 100K environment steps for each method, taking 5 gradient steps per environment step.

E Experiment Details

E.1 Normalized Scores

The `visual-manipulation`, `adroit`, and `antmaze` domains are all goal-oriented, sparse reward tasks. In these domains, we computed the normalized metric as simply the goal achieved rate for each method. For example, in the visual manipulation environment, if the object was placed successfully in the bin, a $+1$ reward was given to the agent and the task is completed. Similarly, for the `door-binary` task in the adroit tasks, we considered the success rate of opening the door. For the `kitchen` task, the task is to solve a series of 4 sub-tasks that need to be solved in an iterative manner. The normalized score is computed simply as $\frac{\#\text{tasks solved}}{\text{total tasks}}$.

E.2 Mixing Ratio Hyperparameter

In this work, we explore the mixing ratio parameter m, which is used during the online fine-tuning phase. The mixing ratio is either a value in the range $[0, 1]$ or the value -1. If this mixing ratio is within $[0, 1]$, it represents what percentage of offline and online data is seen in each batch when fine-tuning. For example, if the mixing ratio $m = 0.25$, that means for each batch we sample 25% from the offline data and 75% from online data. Instead, if the mixing ratio is -1, the buffers are appended to each other and sampled uniformly.

E.3 Details and Hyperparameters for CQL and Cal-QL

We list the hyperparameters for CQL and Cal-QL in Table 3. We utilized a variant of Bellman backup that computes the target value by performing a maximization over target values computed for k actions sampled from the policy at the next state, where we used $k = 4$ in visual pick and place domain and $k = 10$ in others. In the Antmaze domain, we used the dual version of CQL [32] and conducted ablations over the value of the threshold of the CQL regularizer $\mathcal{R}(\theta)$ (target action gap) instead of α. In the visual-manipulation domain which is not presented in the original paper, we swept over the alpha values of $\alpha = 0.5, 1, 5, 10$, and utilized separate α values for offline ($\alpha = 5$) and online ($\alpha = 0.5$) phases for the final results. We built our code upon the CQL implementation from `https://github.com/young-geng/JaxCQL` [14]. We used a single NVIDIA TITAN RTX chip to run each of our experiments.

E.4 Details and Hyperparameters for IQL

We list the hyperparameters for IQL in Table 4. To conduct our experiments, we used the official implementation of IQL provided by the authors [30], and primarily followed their recommended parameters, which they previously ablated over in their work. In the visual-manipulation domain which is not presented in the original paper, we performed a parameter sweep over expectile $\tau = 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99$ and temperature $\beta = 1, 3, 5, 10, 25, 50$ and selected the best-performing values of $\tau = 0.7$ and $\beta = 10$ for our final results. In addition, as the second best-performing method in the visual-manipulation domain, we also attempted to use separate β values for IQL, for a fair comparison with CQL and Cal-QL. However, we found that it has little to no effect, as shown in Figure 9.

E.5 Details and Hyperparameters for AWAC and ODT

We used the JAX implementation of AWAC from `https://github.com/ikostrikov/jaxrl` [29]. We primarily followed the author's recommended parameters, where we used the Lagrange multiplier $\lambda = 1.0$ for the Antmaze and Franka Kitchen domains, and $\lambda = 0.3$ for the Adroit domain. In

the visual-manipulation domain, we performed a parameter sweep over $\lambda = 0.1, 0.3, 1, 3, 10$ and selected the best-performing value of $\lambda = 1$ for our final results. For ODT, we used the author's official implementation from https://github.com/facebookresearch/online-dt, with the author's recommended parameters they used in the Antmaze domain. In addition, in support of our result of AWAC and ODT (as shown in Table 1), the poor performance of Decision Transformers and AWAC in the Antmaze domain can also be observed in Table 1 and Table 2 of the IQL paper [30].

E.6 Details and Hyperparameters for SAC, SAC + Offline Data, Hybrid RL and CQL + SAC

We used the standard hyperparameters for SAC as derived from the original implementation in [19]. We used the same other hyperparameters as CQL and Cal-QL. We used automatic entropy tuning for the policy and critic entropy terms, with a target entropy of the negative action dimension. For SAC, the agent is only trained with the online explored data. For SAC + Offline Data, the offline data and online explored data is combined together and sampled uniformly. For Hybrid RL, we use the same mixing ratio used for CQL and Cal-QL presented in Table 3. For CQL + SAC, we first pre-train with CQL and then run online fine-tuning using both offline and online data, also using the same mixing ratio presented in Table 3.

Table 3: CQL, Cal-QL Hyperparameters

Hyperparameters	**Adroit**	**Kitchen**	**Antmaze**	**Manipulation**
α	1	5	-	5 (online: 0.5)
target action gap	-	-	0.8	-
mixing ratio	-1, 0.25, **0.5**	-1, **0.25**, 0.5	0.5	0.2, **0.5**, 0.7, 0.9

Table 4: IQL Hyperparameters

Hyperparameters	**Adroit**	**Kitchen**	**Antmaze**	**Manipulation**
expectile τ	0.8	0.7	0.9	0.7
temperature β	3	0.5	10	10
mixing ratio	-1, **0.2**, 0.5	-1, **0.25**, 0.5	0.5	0.2, **0.5**, 0.7, 0.9

F Extended Discussion on Limitations of Existing Fine-Tuning Methods

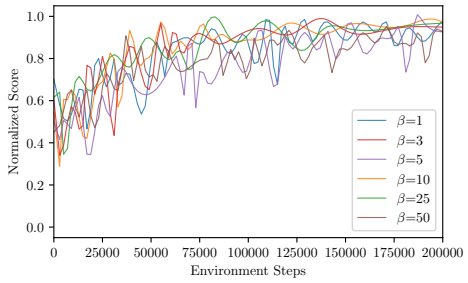

Figure 9: **Abalation on IQL's online temperature values**: The change in the temperature β used in online fine-tuning phase has little to no effect on the sample efficiency.

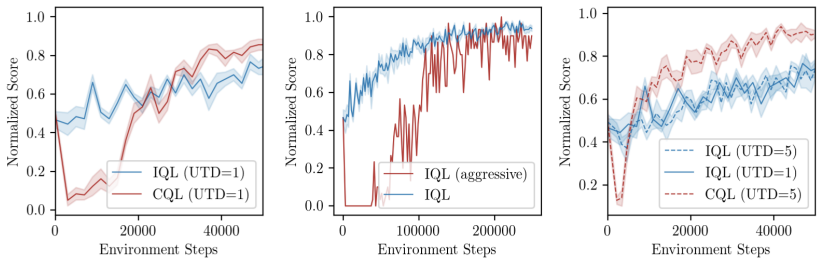

Figure 10: **IQL and CQL:** Step 0 on the x-axis is the performance after offline pre-training. Observe while CQL suffers from initial policy unlearning, IQL improves slower throughout fine-tuning.

In this section, we aim to highlight some potential reasons behind the slow improvement of other methods in our empirical analysis experiment in Section 4.1, and specifically, we use IQL for the analysis. We first swept over the temperature β values used in the online fine-tuning phase for IQL, which controls the constraint on how closely the learned policy should match the behavior policy. As shown in Figure 9, the change in the temperature β has little to no effect on the sample efficiency. Another natural hypothesis is that IQL improves slowly because we are not making enough updates per unit of data collected by the environment. To investigate this, we ran IQL with **(a)** five times as many gradient steps per step of data collection (UTD $= 5$), and **(b)** with a more aggressive policy update. Observe in Figure 10 that **(a)** does not improve the asymptotic performance of IQL, although it does improve CQL meaning that there is room for improvement on this task by making more gradient updates. Observe in Figure 10 that **(b)** often induces policy unlearning, similar to the failure mode in CQL. These two observations together indicate that a policy constraint approach can slow down learning asymptotically, and we cannot increase the speed by making more aggressive updates as this causes the policy to find erroneously optimistic out-of-distribution actions, and unlearn the policy learned from offline data.

G Impact of Estimation Errors in the Reference Value Function

In our experiments, we compute the reference value functions using Monte-Carlo return estimates. However, this may not be available in all tasks. How does Cal-QL behave when reference value functions must be estimated using the offline dataset itself? To answer this, we ran an experiment on the `kitchen` domain, where instead of using an estimate for Q^μ based on the Monte-Carlo return, we train a neural network function approximator Q^μ_θ to approximate Q^μ via supervised regression on to Monte-Carlo return, which is then utilized by Cal-QL. Observe in Figure 11, that the performance of Cal-QL largely remains unaltered. This implies as long as we can obtain a reasonable function approximator to estimate the Q-function of the reference policy (in this case, the behavior policy), errors in the reference Q-function do not affect the performance of Cal-QL significantly.

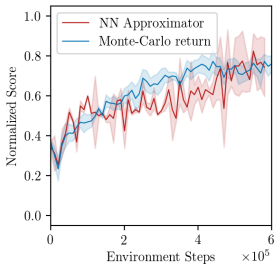

Figure 11: The performance of Cal-QL using a neural net approximator for the reference value function is comparable to using the Monte-Carlo return.

H Initial Unlearning of CQL on Multiple Tasks

In this section, we show the learning curves of CQL and Cal-QL from Figure 6 and zoom in on the x-axis to provide a clearer visualization of CQL's initial unlearning in the Franka Kitchen, Adroit, and the visual-manipulation domains. As depicted in Figure 12, it is evident across all tasks that while CQL experiences initial unlearning, Cal-QL can effectively mitigate it and quickly recovers its performance. Regarding the Antmaze domain, as we discussed in section 7.3, CQL does not exhibit initial unlearning since the default dataset has a high coverage of data. However, we can observe a similar phenomenon if we narrow down the dataset distribution (as shown in Figure 8).

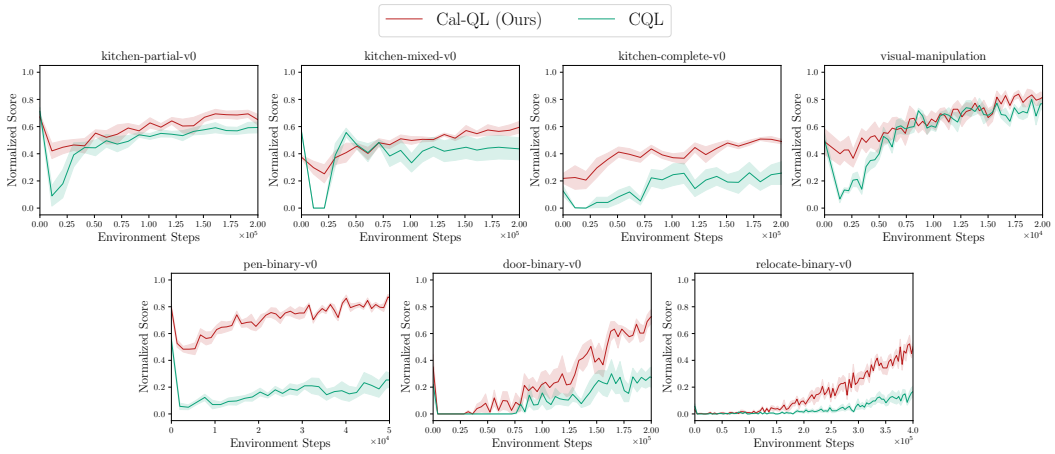

Figure 12: While CQL experiences initial unlearning, Cal-QL effectively mitigates it and quickly recovers its performance.