Quantile Q-Learning: Revisiting Offline Extreme Q-Learning with Quantile Regression

Anonymous authors
Paper under double-blind review

Abstract

Offline reinforcement learning (RL) enables policy learning from fixed datasets without further environment interaction, making it particularly valuable in high-risk or costly domains. Extreme Q-Learning (XQL) is a recent offline RL method that models Bellman errors using the Extreme Value Theorem, yielding strong empirical performance. However, XQL and its stabilized variant MXQL suffer from notable limitations: both require extensive hyperparameter tuning specific to each dataset and domain, and also exhibit instability during training. To address these issues, we proposed a principled method to estimate the temperature coefficient β via quantile regression under mild assumptions. To further improve training stability, we introduce a value regularization technique with mild generalization, inspired by recent advances in constrained value learning. Experimental results demonstrate that the proposed algorithm achieves competitive or superior performance across a range of benchmark tasks, including D4RL and NeoRL2, while maintaining stable training dynamics and using a consistent set of hyperparameters across all datasets and domains.

1 Introduction

Deep reinforcement learning (DRL) has achieved impressive results across a broad range of domains, including navigation (Mirowski et al., 2018), healthcare (Yu et al., 2021a), robotics (Haarnoja et al., 2018), and games (Mnih et al., 2015; Silver et al., 2016). Recent advances in offline reinforcement learning (offline RL) (Kumar et al., 2020; Levine et al., 2020; Kostrikov et al., 2021; Garg et al., 2023) have extended the capability of DRL by enabling agents to learn solely from static datasets, without requiring further interaction with the environment. This paradigm shift is particularly promising in domains where data collection is expensive, risky, or impractical.

Among the recent developments, the offline version of extreme Q-learning (XQL) (Garg et al., 2023) introduces a novel perspective by modeling the Bellman error distribution using the Extreme Value Theorem (EVT), assuming it follows a Gumbel distribution. This theoretical insight leads to an in-sample learning algorithm that has demonstrated competitive performance on standard offline RL benchmarks such as D4RL. However, XQL exhibits notable instability during training, as reported in follow-up work (Omura et al., 2024). To address this issue, MXQL (Omura et al., 2024) introduces a variant that stabilizes training through a Maclaurin-series-based approximation of the original XQL loss.

Despite these efforts, several challenges remain. Both XQL and MXQL are sensitive to hyperparameter choices—particularly the temperature coefficient—which often require dataset-specific tuning to achieve optimal performance, making them impractical for real-world applications. This sensitivity can lead to significant performance degradation when hyperparameters are changed, as demonstrated in Figure 1. Besides, even the stabilized MXQL variant can experience training instability under certain conditions (see Figure 2a). These limitations motivate our central research question:

Can we design an algorithm based on offline XQL that maintains stability and strong performance across diverse domains and offline datasets, using a single, consistent set of hyperparameters?

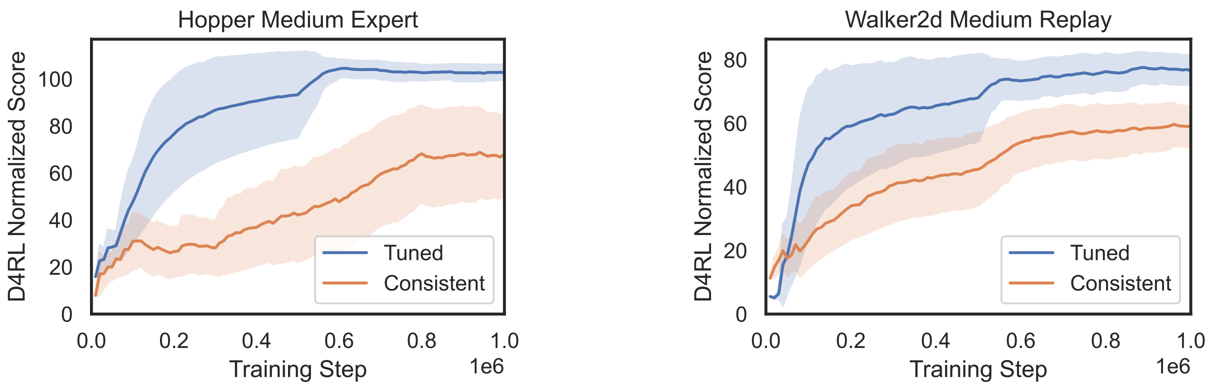


Figure 1: Comparison of XQL performance using dataset-specific tuning versus consistent hyperparameters across the domain. In some scenarios, performance degrades notably without dataset-specific tuning.

To address this issue, we propose a principled method for estimating the temperature coefficient β via quantile regression under mild assumptions, thereby eliminating the need for dataset-specific tuning. Building on this, we introduce a value regularization approach with mild generalization, inspired by prior work on doubly constrained value learning (Mao et al., 2024). The performance of the proposed algorithm matches or exceeds that of the state-of-the-art model-free offline RL algorithms across a variety of domains and dataset types in diverse offline RL benchmarks, including D4RL (Fu et al., 2020) and NeoRL2 (Gao et al., 2025), without the need for domain-specific hyperparameter tuning.

2 Related Works

Our work builds primarily on the literature of offline reinforcement learning (offline RL), a subfield of reinforcement learning that aims to learn optimal policies from a fixed dataset without any further interaction with the environment. A central challenge in offline RL is that many online off-policy methods tend to underperform due to extrapolation error (Fujimoto et al., 2019) or distributional shift (Levine et al., 2020). To address these issues, offline RL algorithms often augment standard off-policy methods with a penalty term that measures the divergence between the learned policy and the behavior policy used to collect the data (Fujimoto & Gu, 2021). Existing offline RL approaches vary in how they formulate the problem, including methods that constrain the learned policy to remain close to the behavior policy (Zhang et al., 2023; Fujimoto & Gu, 2021; Li et al., 2022), approaches that incorporate pessimistic value estimation to avoid overestimation of out-of-distribution actions (An et al., 2021; Yu et al., 2021b; Kostrikov et al., 2021; Kumar et al., 2020), and techniques that leverage deep architectures such as large autoregressive models (Chen et al., 2021; Janner et al., 2021) or generative models (Janner et al., 2022; Ajay et al., 2022; Wang et al., 2022; Zhang et al., 2025). In our work, we specifically focus on improving an existing offline RL method, Extreme-Q Learning (XQL) (Garg et al., 2023), which is potentially unstable and sensitive to hyperparameters as discussed in the introduction. Prior work (Omura et al., 2024) proposed a method to stabilize XQL by applying a Maclaurin expansion to the original objective function.

3 Preliminaries

The problem is formulated as an Markov decision process (MDP), which is represented by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ represents the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ characterizes the transition dynamics; $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function; $\gamma \in [0, 1)$ is the discount factor. We aim to learn a policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ to maximize cumulative discounted rewards (Sutton & Barto, 2018). In offline RL setting, the policy learning is conducted on a dataset $\mathcal{D}$ consisting a set of trajectories $(\tau_1, ..., \tau_T)$. Each trajectory is composed of a group of transitions $\{(s_t, a_t, r_t, s'_t)\}$.

Extreme Q-Learning Extreme Q-Learning (XQL) (Garg et al., 2023) is an algorithm being capable of solving both online and offline RL tasks. It directly models the maximal value using Extreme Value Theory (EVT) with the assumption that the error in Q-functions follow a Gumbel distribution. In our work, we especially focus on the offline part of XQL. Specifically, offline XQL leverages an objective for optimizing a soft value function V over dataset $\mathcal{D}$:

$$\mathcal{J}(V) = \mathbb{E}_{(s,a)\sim\mathcal{D}} \exp\left((Q(s,a) - V(s))/\beta\right) - (Q(s,a) - V(s)/\beta) - 1, \tag{1}$$

where Q is the Q-function optimized via minimizing mean-squared Bellman error and β is the temperature hyperparameter. Furthermore, XQL learns a policy with an advantage-weighted regression (Peng et al., 2019; Nair et al., 2020) (AWR) style update.

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[e^{(Q(s,a)-V(s))/\beta} \log \pi \right]. \tag{2}$$

In practice, we find that XQL is sensitive to the hyperparameter β, and its optimal value varies significantly across environments and offline datasets, as shown in Appendix C.1 and Figure 1.

4 Revisiting Extreme Q-Learning with Quantile Regression

4.1 Estimating β under Mild Assumptions

Since the formulation of XQL is sensitive to the hyperparameter β, we show in this section that β can be generalized to a state-dependent function $\beta(s)$ and learned under mild assumptions. We begin by stating the assumptions used in our subsequent analysis involving the estimated Q-function Q, the optimal Q-function $Q^\star$ and the soft value function V. It is worth noting that these assumptions are either identical or closely aligned with those used in prior works (Hui et al., 2023; Garg et al., 2023).

Assumption 1 ((Hui et al., 2023))**.** *Given a state-dependent function $\beta(s)$ and a heteroscedastic Gumbel noise $\epsilon(s,a) \sim \mathcal{G}(0, \beta(s))$:*

$$\epsilon(s,a) = Q^\star(s,a) - Q(s,a).$$

Assumption 2. *Given a state-dependent Gumbel noise model $\mathcal{G}(0, \beta(s))$:*

$$(Q(s,a) - V(s)) \sim -\mathcal{G}(0, \beta(s)).$$

Remark on Assumption 2 Assumption 2 is similar to the one used in XQL (Garg et al., 2023), except that it allows for a state-dependent noise model. It reduces exactly to the XQL assumption when the noise scale is constant, i.e., $\beta(s) = \beta$.

With the above assumptions in place, we now present Proposition 1, followed by Proposition 2, which provides key theoretical support for estimating $\beta(s)$.

Proposition 1. *Under Assumptions 1 and 2, given an optimal Q-function $Q^\star$ and $V^\star(s) = \max_a Q^\star(s,a)$, the following equation holds:*

$$V^\star(s) = \beta(s) \log \int \exp\left(\frac{Q^\star(s,a)}{\beta(s)}\right) da.$$

Proof. See Appendix B.1. □

Proposition 1 establishes a relationship between the optimal Q-function and value function. To extend this relationship to the estimated functions used in practice, we define an estimated value function inspired by the structure in Proposition 1.

Definition 1. *Given a Q-function estimation and a state-dependent temperature parameter $\beta(s)$, we define the following value function:*

$$\tilde{V}(s) = \beta(s) \log \int \exp\left(\frac{Q(s,a)}{\beta(s)}\right) da.$$

Remark on Definition 1 $\tilde{V}(s)$ can be interpreted as a SoftArgMax (Hui et al., 2023) of $Q(s,a)$, aligning with the principles of MaxEnt RL and Extreme Q-Learning. Since we assume the $\beta(s)$ scale is consistent across Assumptions 1 and 2, it enables a coherent formulation of both $V(s)$ and $\tilde{V}(s)$ (Definition 1). For simplicity, we use the notation $V(s)$ throughout the paper. Empirical evidence supporting the consistency of the $\beta(s)$ scale is presented in the experimental section.

Based on Proposition 1 and Definition 1, we can construct an estimation of $\beta(s)$ based on the value functions:

Proposition 2 (Estimating $\beta(s)$)**.** *Under Assumptions 1 and 2, given $\hat{V}(s) = \mathbb{E}[V^\star(s)]$, there exists:*

$$\omega\beta(s) = \hat{V}(s) - V(s),$$

where $\omega \approx 0.57721$ is identified as the Euler–Mascheroni constant.

Proof. See Appendix B.2. □

In this case, $\beta(s)$ can be estimated by computing $[\hat{V}(s) - V(s)]/\omega$. However, how to estimate both $\hat{V}(s)$ and $V(s)$ remains an open question. In the next section, we propose an estimation method based on quantile regression.

4.2 Learning Value Functions with Quantile Regression

In this section, we demonstrate how the learning of $V(s)$ and $\hat{V}(s)$ can be formulated as a quantile regression problem. Under Assumption 2, the cumulative distribution function (CDF) of the Gumbel distribution enables us to interpret the value functions as quantiles, as established in Propositions 3 and 4.

Proposition 3. *For a Q-function and value function satisfying Assumption 2, and with $\alpha_1 = 1 - \exp(-1)$, there exists:*

$$P(Q(s,a) \leq V(s)) = \alpha_1,$$

which shows that $V(s)$ corresponds to the α_1-quantile of $Q(s,a)$ under its CDF.

Proof. See Appendix B.3. □

Similarly, by applying the results from Proposition 2, we obtain the following for $\hat{V}$.

Proposition 4. *Under Assumptions 1 and 2, for a Q-function, let $\hat{V}(s) = \mathbb{E}[V^\star(s)]$, and define $\alpha_2 = 1 - \exp(-\exp(\omega))$. Then, there exists:*

$$P(Q(s,a) \leq \hat{V}(s)) = \alpha_2,$$

which shows that $\hat{V}(s)$ corresponds to the α_2-quantile of $Q(s,a)$ under its CDF.

Proof. See Appendix B.4. □

Based on the preceding propositions, we have established that $V(s)$ and $\hat{V}(s)$ correspond to the α_1- and α_2-quantiles of $Q(s,a)$ under its cumulative distribution function. Consequently, it is natural to estimate $V(s)$ and $\hat{V}(s)$ using quantile regression, and to estimate $\beta(s)$ based on the resulting estimates of $V(s)$ and $\hat{V}(s)$.

Algorithm 1 Quantile Q-Learning

```
Require: Initialized networks $Q_\theta$, $V_{\psi_1}$, $V_{\psi_2}$, dataset $\mathcal{D}$, mild generalization and policy constraint hyperparameters $\lambda$ and $\zeta$
1: for each gradient step do
2:     Sample a batch of transitions from dataset $\mathcal{D}$
3:     Update value networks by minimizing Eqs. 7.
4:     Update $Q$-function $Q_\theta$ by minimizing Eq. 5
5:     Update policy $\pi_\phi$ by maximizing Eq. 6
6: end for
```

Definition 2 (Quantile Regression)**.** *The quantile regression objective is defined as:*

$$\mathcal{L}_{qr}(u,\tau) = u \cdot (\tau - \mathbb{I}(u<0)),$$

where $u = y - \hat{y}$ is the residual, $\tau \in [0,1]$ is the target quantile level, and $\mathbb{I}(\cdot)$ denotes the indicator function.

To learn $V(s)$ and $\hat{V}(s)$, we parameterize them using ψ_1 and ψ_2, respectively. Leveraging the quantile regression objective defined in Definition 2, we formulate the learning objectives for $V_{\psi_1}(s)$ and $\hat{V}_{\psi_2}(s)$ as follows:

$$\mathcal{J}(\psi_1) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\, \mathcal{L}_{qr}(Q_\theta(s,a) - V_{\psi_1}(s), \alpha_1). \tag{3}$$

$$\mathcal{J}(\psi_2) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\, \mathcal{L}_{qr}(Q_\theta(s,a) - \hat{V}_{\psi_2}(s), \alpha_2). \tag{4}$$

For a Q-function satisfying Assumption 1, $Q(s,a) + \omega\beta(s)$ is an unbaised estimation of the optimal Q-function $Q^\star(s,a)$. By Proposition 2, the estimator for $Q^\star$ can be further expressed as $Q(s,a) + (\hat{V}(s) - V(s))$. Therefore, the Q-function, parameterized by θ, is learned using the mean-squared Bellman error :

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[Q_\theta(s,a) + (\hat{V}_{\psi_2}(s) - V_{\psi_1}(s)) - r(s,a) - \gamma\hat{V}_{\psi_2}(s')\right]^2. \tag{5}$$

Using the estimator $\beta(s) = (\hat{V}_{\psi_2}(s) - V_{\psi_1}(s))/\omega$ and a KL-constrained policy objective, we can obtain a β-free AWR-style policy objective with a policy constraint weight ζ:

$$\mathcal{L}_\pi(\phi) = \mathbb{E}_{(s,a)\sim\mathcal{D}} \exp\Big(\frac{\omega(Q_\theta(s,a) - V_{\psi_2}(s))}{\zeta(\hat{V}_{\psi_2}(s) - V_{\psi_1}(s))} + \frac{\omega(Q_\theta(s,a) - V_{\psi_1}(s))}{\hat{V}_{\psi_2}(s) - V_{\psi_1}(s)}\Big) \log \pi_\phi(s,a). \tag{6}$$

The detailed derivation can be found in Appendix A. Thus far, we have established the basic formulation of our proposed algorithm, a β-free variant of XQL, which we refer to as **Quantile Q-Learning**. Further enhancements to this approach will be presented in the following sections.

4.3 Regulating Value Functions with Mild Generalization

By estimating β via quantile regression, we construct a β-free variant of XQL. However, similarly to the original XQL, this approach remains fully in-sample and has been shown to be overly conservative (Mao et al., 2024). To address this crucial issue, we introduce mild generalization into the quantile regression objective.

A straightforward approach is to apply Doubly Mild Generalization (DMG) (Mao et al., 2024) directly to Eq.3 and Eq.4. However, approximating $V^\star(s') = \max_{a'\sim\pi(\cdot|s')} Q^\star(s',a')$ by sampling actions from $\pi(\cdot \mid s')$ and taking the α_2-quantile of $Q_\theta(s',a')$ can introduce extrapolation error (Fujimoto et al., 2019), which may propagate throughout the learning process. To mitigate this, we conservatively estimate $V^\star(s')$ by subtracting $\omega\beta(s')$, with $\beta(s')$ providing an estimation of uncertainty. Applying this conservative estimation systematically to all value functions throughout mild generalization leads to $V(s')$ as the α_0-quantile of $Q(s',a')$, and $\hat{V}(s)$ as the α_1-quantile of $Q(s',a')$, whereby $\alpha_0 = 1 - \exp(-\exp(-\omega))$. Ablation results in the experiment section demonstrate the effectiveness of this modification. The updated objective for training the value networks incorporating mild generalization can therefore be expressed as:

$$\begin{aligned}\mathcal{J}(\psi_1) &= \mathbb{E}_{(s,a)\sim\mathcal{D}}\, \mathcal{L}_{qr}(Q_\theta(s,a) - V_{\psi_1}(s), \alpha_1) + \mathbb{E}_{s'\sim\mathcal{D}, a'\sim\pi(\cdot|s')}\, \lambda\, \mathcal{L}_{qr}(Q_\theta(s',a') - V_{\psi_1}(s'), \alpha_0),\\ \mathcal{J}(\psi_2) &= \mathbb{E}_{(s,a)\sim\mathcal{D}}\, \mathcal{L}_{qr}(Q_\theta(s,a) - \hat{V}_{\psi_2}(s), \alpha_2) + \mathbb{E}_{s'\sim\mathcal{D}, a'\sim\pi(\cdot|s')}\, \lambda\, \mathcal{L}_{qr}(Q_\theta(s',a') - \hat{V}_{\psi_2}(s'), \alpha_1).\end{aligned} \tag{7}$$

In this objective, a mild generalization hyperparameter λ is introduced to control the degree of generalization applied. Empirically, we find that $\lambda = 1$ works well in most cases.

5 Experiments

5.1 Experimental Setting and Baseline Methods

Benchmark Datasets We conduct experiments on two offline RL benchmarks: the widely used D4RL suite (Fu et al., 2020) and the more challenging, near–real-world NeoRL2 benchmark (Gao et al., 2025), which incorporates real-world complexities such as high-latency transitions and global safety constraints. For D4RL, our evaluation spans locomotion tasks (Hopper, HalfCheetah, and Walker2d) across various dataset types, including medium, medium-replay, and medium-expert. We also evaluate on the Adroit manipulation tasks (Pen, Door, and Hammer), as well as the AntMaze-Umaze task under both fixed and randomized initial states and goals. For NeoRL2, we assess performance on the RocketRecovery and SafetyHalfCheetah tasks.

Baseline Methods Our approach is compared against a diverse set of offline RL baselines, encompassing various design principles. These include policy-constrained methods such as Behavior Cloning (BC) and TD3+BC (Fujimoto & Gu, 2021); conservative model-free algorithms like CQL (Kumar et al., 2020); in-sample learning methods including IQL (Kostrikov et al., 2021) and XQL (Garg et al., 2023); a stabilized variant of XQL, MXQL (Omura et al., 2024); an uncertainty-aware method, EDAC (An et al., 2021); and the batch-constrained off-policy algorithm BCQ (Fujimoto et al., 2019).

Hyperparameter Setting For every task–dataset combination we fix the generalization coefficient to $\lambda = 1.0$ and the policy constraint weight to $\zeta = 1.0$. Using this single hyperparameter setting for all of our experiments (Table 1) highlights the robustness of our method, whereas baseline algorithms are evaluated with the dataset-specific, tuned hyperparameters reported in prior work. We also compare our approach and XQL under a consistent hyperparameter setting across tasks, revealing that the original XQL algorithm suffers from performance degradation without per-task tuning, while our method maintains strong performance (Table 3). The detailed hyperparameter settings for the baseline algorithms are provided in Appendix C.

5.2 Main Results

In this section, the results of the proposed algorithm are presented on D4RL datasets, and its performance is compared with baseline methods. Our approach consistently outperforms or matches state-of-the-art in-sample and model-free offline RL baselines under dataset-specific hyperparameter tuning using a unified hyperparameter setting across diverse domains (Locomotion, Adroit, and AntMaze) and dataset types. The results are summarized in Table 1. Furthermore, training curve comparisons are presented between two improved versions of XQL: MXQL and our proposed QQL. As shown in Figure 2a, QQL demonstrates more stable training dynamics, with lower standard deviation, and achieves more optimal performance.

5.3 Comparison with Consistent Hyperparameter Setting

We compare the performance of our method with XQL (Garg et al., 2023) under a consistent hyperparameter setting across diverse datasets and domains. As shown in Table 3, XQL suffers significant performance degradation when using the consistent hyperparameters across settings, while our method maintains strong performance. This highlights the robustness and generality of our approach.

5.4 Ablation Studies

Ablations on Value Regulation and Conservative Estimation Ablation studies are performed to isolate the contributions of Value Regulation (VR) and Conservative Estimation (CE) to the performance and stability of the QQL algorithm. Experiments on HalfCheetah Medium, Walker2d Medium Replay, and

Domain	Dataset	BC	TD3+BC	CQL	IQL	XQL	MXQL	QQL (Ours)
Gym Locomotion	hopper-med-exp	52.5	98.0	105.4	91.5	111.2	110.7	**112.5±1.3**
	hopper-med	52.9	59.3	58.5	66.3	74.2	**80.9**	77.3 ± 3.8
	hopper-med-rep	18.1	60.9	95.0	94.7	100.7	**102.7**	101.1±2.1
	halfcheetah-med-exp	55.2	90.7	91.6	86.7	94.2	92.1	**94.3 ± 1.8**
	halfcheetah-med	42.6	48.3	44.0	47.4	48.3	47.7	**49.5 ± 0.3**
	halfcheetah-med-rep	36.6	44.6	45.5	44.2	45.2	45.7	**46.6 ± 0.3**
	walker2d-med-exp	107.5	110.1	108.8	112.7	112.7	111.2	**113.2±0.3**
	walker2d-med	75.3	83.7	72.5	78.3	84.2	83.8	**85.2 ±1.3**
	walker2d-med-rep	26.0	81.8	77.2	73.9	82.2	83.6	**90.2 ± 2.1**
Adroit	pen-human	99.7	10.0	58.9	106.2	105.3	122.1	**128.3 ± 4.1**
	pen-cloned	99.1	52.7	14.7	114.1	112.6	**117.4**	115.2 ± 4.7
	door-human	9.4	-0.1	13.3	13.5	13.2	18.3	**21.1 ± 5.6**
	door-cloned	3.4	-0.2	-0.1	**9.0**	1.1	1.8	8.8 ± 4.1
	hammer-human	12.6	2.4	0.3	6.9	7.3	**14.7**	8.4 ± 3.1
	hammer-cloned	8.9	0.9	0.3	**11.6**	1.1	11.1	8.0 ± 4.4
AntMaze	antmaze-umaze	68.5	**98.5**	94.8	84.0	90.3	88.3	88.5 ± 6.1
	antmaze-umaze-diverse	64.8	71.3	53.8	79.5	77.2	53.2	**81.3 ± 4.1**

Table 1: **Main Experimental Results on D4RL.** Average normalized performance on Gym Locomotion, Adroit, and AntMaze tasks. All hyperparameters for our QQL method are kept **consistent** across all environments and datasets, while baseline algorithms are individually **tuned** for each setting. Results are averaged over 5 random seeds.

Dataset	DATA	BC	CQL	EDAC	BCQ	TD3+BC	XQL	QQL (Ours)
RocketRecovery	75.27	72.75	74.32	65.65	76.46	79.74	81.1	**83.2±4.7**
SafetyHalfCheetah	**73.56**	70.16	71.18	53.11	54.65	68.58	66.2	60.3±4.1

Table 2: **Main Experimental Results on NeoRL2.** Average normalized performance on RocketRecovery and SafetyHalfCheetah tasks. All hyperparameters for our QQL method are kept consistent, while baseline algorithms are individually **tuned** for each setting. The **DATA** column reports the normalized scores of the trajectories contained in the offline datasets. Results are averaged over 3 random seeds.

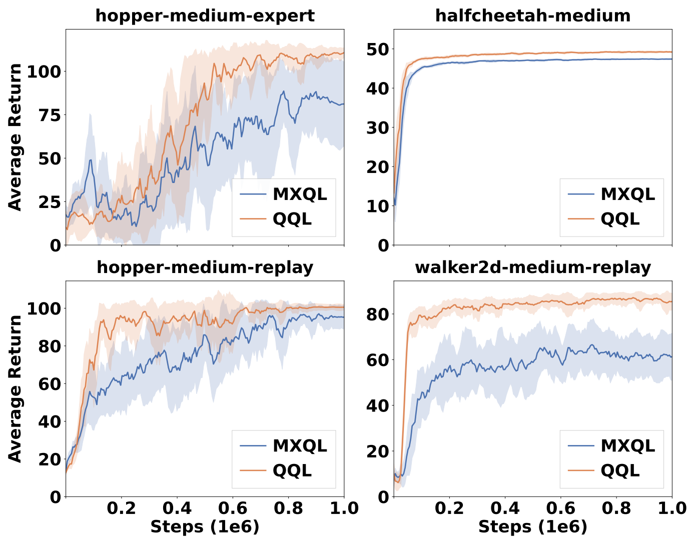
Hopper Medium Expert (Figure 3) compare the full QQL method with variants that exclude value regulation (QQL w/o VR) or conservative estimation (QQL w/o CE), as described in the methodology section.

Removing either component results in noticeable performance drops, while omitting both leads to substantial degradation. This highlights the role of value regulation in mitigating overestimation and stabilizing value estimates for robust offline learning, and the importance of conservative estimation in encouraging pessimistic value functions to better handle distributional shifts and limited coverage.

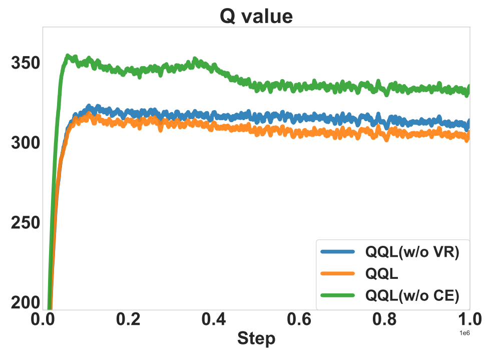
These results demonstrate that both VR and CE are essential for effective offline RL. Their complementary effects yield a more stable and performant learning algorithm. All ablation experiments use consistent hyperparameter settings.

Additionally, a comparison of the Q-values from the original QQL method against those from QQL without value regularization and without conservative estimation was plotted. The results demonstrate that both value regularization and conservative estimation help stabilize Q-value estimation and improve conservativeness. The ablation study is performed on the Hopper Medium Expert dataset, with the results shown in Figure 2b.

Ablations on the Hyperparameters λ and ζ Ablation studies are conducted to investigate the impact of the mild generalization coefficient λ and the policy constraint weight ζ. Experiments are performed on three Gym Locomotion tasks: HalfCheetah Medium, Walker2d Medium Replay, and Hopper Medium Expert.



(a) **Training curve comparison between QQL and MXQL.** QQL demonstrates more stable training behavior across tasks compared to MXQL.



(b) **Q-value plots for value regularization and conservative estimation ablation.** We visualize the Q-value estimates of the original QQL method, its variant without value regularization (w/o VR), and its variant without conservative estimation (w/o CE), as described in the methodology section. The ablation results demonstrate that the proposed approach effectively stabilizes Q-value estimation.

Figure 2: **QQL Performance and Ablation Analysis.** We visualize the following results: (a) Training stability comparison between QQL and MXQL. (b) Q-value plots demonstrating the stabilization effect of value regularization and conservative estimation.

Domain	Gym Locomotion	Adroit	AntMaze
XQL (Dataset-specific tuning)	83.7±2.7	40.6±6.3	83.8±6.9
XQL (Consistent per domain)	75.5±3.6	38.6±4.5	69.6±10.7
XQL (Consistent across all domains)	73.6±4.1	37.8±4.1	67.3±9.2
QQL (Consistent across all domains)	**85.6±1.4**	**48.3±4.3**	**84.9±5.1**

Table 3: **Comparisons with Consistent Hyperparameters** We compare our approach—using a single, consistent set of hyperparameters—against three XQL variants: (1) with dataset-specific tuning, (2) with domain-level consistent hyperparameters, and (3) with a single set shared across all domains. Performance is reported as the average D4RL score across all dataset types and tasks within each domain. Detailed hyperparameter settings for the different XQL variants are provided in Appendix C.1. All results are averaged over 5 random seeds.

For both λ and ζ, we fix one hyperparameter at 1.0 while varying the other over the set $[0.25, 0.5, 1.0, 2.0, 4.0]$. The results are summarized in Table 4.

The ablation results show that the scores exhibit minimal variation with changes in the hyperparameters λ and ζ, demonstrating the robustness of the proposed QQL method to these hyperparameters. The ablation results show that the scores exhibit minimal variation with changes in the hyperparameters λ and ζ, demonstrating the robustness of the proposed QQL method to these hyperparameters.

5.5 Experiments on $\beta(s)$ Scale

In Definition 1, we actually assume that the $\beta(s)$ scale of Assumption 1 and Assumption 2 is similar, as discussed in the remark of Definition 1. In this section, we provide empirical evidence by leveraging the results from a toy example.

Dataset	Setting	0.25	0.5	1.0	2.0	4.0
hopper-med-exp	$\lambda = 1.0$ (ζ varies)	113.2±0.3	113.1±0.3	112.5±1.3	112.9±0.3	112.6±0.6
	$\zeta = 1.0$ (λ varies)	110.7±3.2	111.6±2.1	112.5±1.3	112.1±1.7	112.6±0.3
halfcheetah-med	$\lambda = 1.0$ (ζ varies)	50.0±0.1	49.6±0.1	49.5±0.3	49.4±0.1	49.1 ± 0.1
	$\zeta = 1.0$ (λ varies)	49.6±0.1	49.6±0.1	49.5±0.3	49.6±0.1	49.6±0.2
walker2d-med-rep	$\lambda = 1.0$ (ζ varies)	90.9±0.8	90.4±1.2	90.2±2.1	89.9±1.5	88.4±0.4
	$\zeta = 1.0$ (λ varies)	88.8 ± 1.4	90.5 ± 1.0	90.2±2.1	88.5 ± 0.6	89.3 ± 0.6

Table 4: **Ablations on λ and ζ.** Average normalized performance on D4RL tasks. The first row for each task shows performance when varying ζ with fixed $\lambda = 1.0$, and the second row shows performance when varying λ with fixed $\zeta = 1.0$.

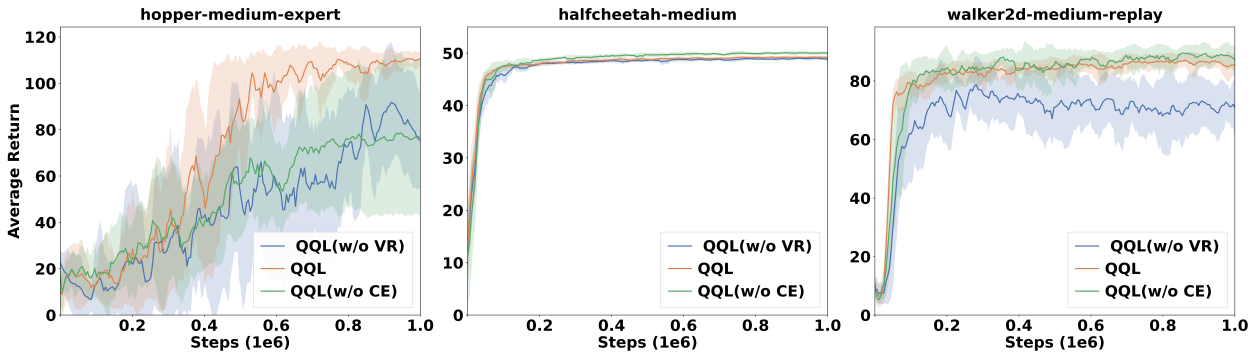


Figure 3: **Ablation Studies on Value Regulation and Conservative Estimation** Comparison of QQL performance to its variant without value regulation (w/o VR) and its variant without conservative estimation (w/o CE) adapted in the methodology section. The hyperparameters are consistent across variants.

Toy Example Construction. To illustrate that Assumption 2 can be approximately satisfied given Assumption 1, we construct a toy example where the Q-value estimation process aligns with the desired Gumbel properties. Under Assumption 1, we model $-Q(s,a)$ as a Gumbel-distributed random variable with location $-Q^\star(s,a)$ and scale $\beta(s)$. We fix the state s and generate $-Q^\star(s,a)$ using a randomly initialized neural network that takes action a as input. To simulate $-Q(s,a)$, we add Gumbel noise $g(s) \sim \mathcal{G}(0, \beta(s))$ to the output of the network. To test Assumption 2, we sample actions from a Gaussian policy and analyze the resulting distribution of $-Q(s,a)$ values to check whether it retains the Gumbel form. The procedure involves computing $-Q^\star(s,a)$ for each sampled action, adding Gumbel noise, and comparing the empirical distribution of $-Q(s,a)$ to a theoretical Gumbel distribution. We fix the Gumbel noise scale at $\beta(s) = 1$ and vary the standard deviation of the Gaussian policy over a range of values.

Results and Discussion. Figure 4 displays the empirical distributions of $-Q(s,a)$ under varying Gaussian policy variances, overlaid with Gumbel distributions fitted via maximum likelihood estimation. The strong alignment between the empirical histograms and the fitted Gumbel curves, which is confirmed with high p-values in a goodness-of-fit test, shows that the distribution of $-Q(s,a)$ maintains its Gumbel form despite the stochasticity of Gaussian sampling. The shape and scale of these fitted distributions are primarily controlled by the additive Gumbel noise used in the simulation. This supports our premise that the scale parameter $\beta(s)$ stays consistent between Assumptions 1 and 2, even when actions are sampled from a distribution rather than being deterministically fixed.

6 Conclusion and Future Work

In this work, we have addressed critical limitations of prior Extreme Q-Learning (XQL) approaches—namely, the need for dataset-specific hyperparameter tuning and the instability of training. We have introduced

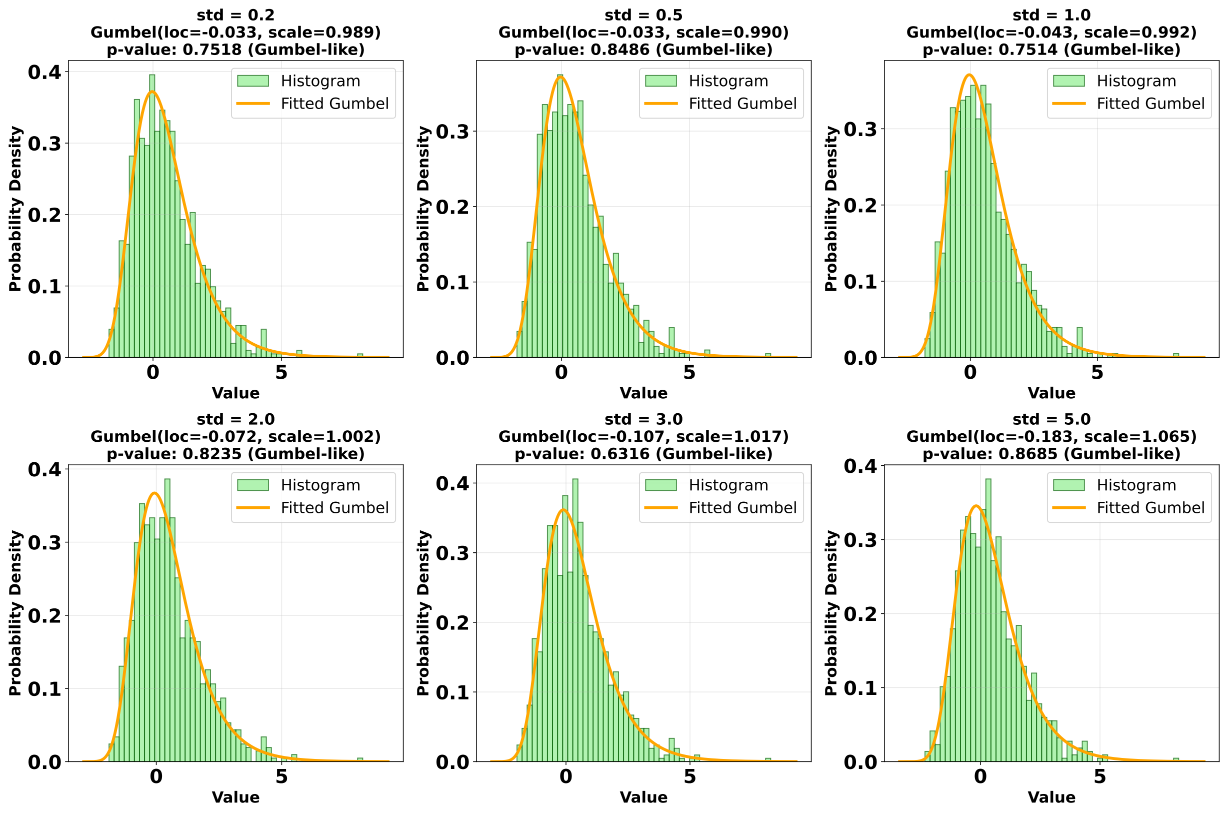


Figure 4: **Toy Example for $\beta(s)$ Scale.** Empirical distributions of $-Q(s,a)$, along with the corresponding fitted Gumbel distributions and their estimated location, scale, and goodness-of-fit p-values.

a novel method for estimating the temperature coefficient β via quantile regression, requiring only mild statistical assumptions. To further bolster stability, we have incorporated a value regularization mechanism inspired by constrained value learning that encourages mild generalization while preserving performance.

Empirical results across diverse offline RL benchmarks, including D4RL and NeoRL2, validate our approach: it achieves performance that is competitive with—or even surpasses—XQL and its stabilized variant MXQL, while avoiding the hyperparameter overfitting and erratic training behavior they exhibit. Crucially, our method maintains robust performance using a consistent set of hyperparameters across all tasks and domains, highlighting its practicality and general applicability in real-world high-stakes settings.

Looking ahead, a natural extension of this work is to adapt our approach to the online reinforcement learning setting by leveraging developments in online variants of Extreme Q-Learning. Integrating our quantile-based temperature estimation and value regularization into the online learning framework could improve training stability in interactive environments. This generalization has the potential to make Extreme Q-Learning more robust and widely applicable in real-time decision-making scenarios.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making?, 2022.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 7436–7447, 2021.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 15084–15097, 2021.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pp. 2052–2062. PMLR, 2019.

Chen-Xiao Gao and Kong Rui. OfflineRL-Lib: Benchmarked Implementations of Offline RL Algorithms, feb 2023. URL `https://github.com/typoverflow/OfflineRL-Lib`.

Songyi Gao, Zuolin Tu, Rong-Jun Qin, Yi-Hao Sun, Xiong-Hui Chen, and Yang Yu. Neorl-2: Near real-world benchmarks for offline reinforcement learning with extended realistic scenarios, 2025.

Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme q-learning: Maxent reinforcement learning without entropy, 2023.

E. J. Gumbel. Les valeurs extrêmes des distributions statistiques. *Annales de l'institut Henri Poincaré*, 5: 115–158, 1935.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 1861–1870. PMLR, 2018.

David Yu-Tung Hui, Aaron Courville, and Pierre-Luc Bacon. Double gumbel q-learning. In *Proceedings of the 37th Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 2580–2616, 2023. URL `https://openreview.net/forum?id=UdaTyy0BNB`.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 1273–1286, 2021.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pp. 9902–9915. PMLR, 2022.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Proceedings of the 34th Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 1179–1191, 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.

Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. When data geometry meets deep function: Generalizing offline reinforcement learning, 2022.

Yixiu Mao, Qi Wang, Yun Qu, Yuhang Jiang, and Xiangyang Ji. Doubly mild generalization for offline reinforcement learning. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 51436–51473, 2024.

Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. In *Proceedings of the 32nd Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

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

Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. Accelerating online reinforcement learning with offline datasets, 2020.

Motoki Omura, Takayuki Osa, Yusuke Mukuta, and Tatsuya Harada. Stabilizing extreme q-learning by maclaurin expansion. In *Proceedings of the 1st Reinforcement Learning Conference (RLC)*, 2024. URL `https://openreview.net/forum?id=LNCjWk859A`.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL `https://arxiv.org/abs/1910.00177`.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. doi: 10.1038/nature16961.

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

Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. CORL: Research-oriented deep offline reinforcement learning library. In *Proceedings of the 3rd Offline RL Workshop: Offline RL as a "Launchpad"*, 2022. URL `https://openreview.net/forum?id=SyAS49bBcv`.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning, 2022.

Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021a.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. In *Proceedings of the 35th Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 28954–28967, 2021b.

Jing Zhang, Chi Zhang, Wenjia Wang, and Bingyi Jing. Constrained policy optimization with explicit behavior density for offline reinforcement learning. In *Proceedings of the 37th Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 5616–5630, 2023.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning, 2025.

A Derivation of Policy Objective

A common challenge in policy learning arises when the sampling policy $\mu(\cdot \mid \cdot)$ is suboptimal, which often results in overly conservative estimates from in-sample approaches. To address this issue, existing methods like Advantage-Weighted Regression (AWR) aim to mitigate such inherent conservativeness. Specifically, the AWR-style policy objective minimizes the cross-entropy loss $\mathbb{C}(\mu^* \,||\, \pi)$, where $\mu^*(a \mid s) \propto \mu(a \mid s)\exp(Q(s,a)/\beta(s))$. We define the normalizer $V(s) = \beta(s)\int \mu(a \mid s)\exp(\frac{Q(s,a)}{\beta(s)})da$. Thus, μ^* can be expressed as $\mu^*(a \mid s) = \mu(a \mid s)\exp(\frac{Q(s,a)-V(s)}{\beta(s)})$.

Given that μ^* shares the same support as μ and is considered a superior policy, we construct a less conservative policy π by remaining close to μ^*. Consequently, the following objective is maximized:

$$\max_{\phi} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\phi(\cdot|s)} Q_\theta(s,a) - \nu \mathbb{E}_{(s,a)\sim\mathcal{D}} \mathbf{KL}(\mu^* \mid\mid \pi_\phi(\cdot \mid s)). \tag{8}$$

This objective is equivalent to:

$$\max_{\phi} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\phi(\cdot|s) Q_\theta(s,a)} - \nu \mathbb{E}_{(s,a)\sim\mathcal{D}}[\exp((Q_\theta(s,a) - V_{\psi_1})/\beta(s)) \log \pi_\phi(a \mid s)]. \tag{9}$$

Here, $\nu = \zeta\beta(s)$, where ζ is a positive, adjustable parameter referred to as the policy constraint weight, balancing policy optimization and the conservative constraint.

Furthermore, Equation equation 9 resembles the MaxEnt RL objective and possesses a closed-form solution:

$$\begin{aligned}
\pi^*(a \mid s) &= \mu^*(a \mid s) \exp\left(\frac{Q(s,a) - V'(s)}{\beta(s)}\right) \\
&= \mu(a \mid s) \exp\left(\frac{Q(s,a) - V'(s)}{\zeta\beta(s)}\right) \exp\left(\frac{Q(s,a) - V(s)}{\beta(s)}\right),
\end{aligned}$$

where $V'(s) = \beta(s) \int \mu^*(a \mid s) \exp(\frac{Q(s,a)}{\zeta\beta(s)})da$ serves as a normalizer. Based on this, the policy optimization objective is modified to minimize the KL divergence between π^* and π. In practice, V_{ψ_1} and V_{ψ_2} approximate the normalizers $V(s)$ and $V'(s)$, respectively, leading to the following policy optimization objective:

$$\max_{\phi} \mathbb{E}_{(s,a)\sim\mathcal{D}} \exp\left(\frac{Q_\theta(s,a) - V_{\psi_2}(s)}{\zeta((V_{\psi}2(s) - V_{\psi}1(s))/\omega)} + \frac{Q_\theta(s,a) - V_{\psi_1}(s)}{(V_{\psi}2(s) - V_{\psi}1(s))/\omega}\right) \log \pi_\phi(s,a).$$

B Ommited Proofs

B.1 Proof for Proposition 1

To begin with, we introduce the following lemmas:

Lemma 1 (Gumbel (1935)). *Given a continuous random variable α and a Gumbel noise $g \sim \mathcal{G}(0,\beta)$, the following identity exists:*

$$\beta \log \int \exp\left(\frac{\alpha + g}{\beta}\right) d\alpha = \beta \log \int \exp\left(\frac{\alpha}{\beta}\right) d\alpha + g.$$

Proof.

$$\begin{aligned}
\beta \log \int \exp\left(\frac{\alpha + g}{\beta}\right) d\alpha &= \beta \log \int \exp\left(\frac{\alpha}{\beta} + \frac{g}{\beta}\right) d\alpha \\
&= \beta \log(\exp\left(\frac{g}{\beta}\right) \int \exp\left(\frac{\alpha}{\beta}\right) d\alpha) \\
&= \beta \log \exp\left(\frac{g}{\beta}\right) + \beta \log \int \exp\left(\frac{\alpha}{\beta}\right) d\alpha \\
&= \beta \log \int \exp\left(\frac{\alpha}{\beta}\right) d\alpha + g.
\end{aligned}$$

□

Lemma 2 (Hui et al. (2023)). *Given a optimal Q-function $Q^\star$ and a state-dependent gumbel noise $g(s) \sim \mathcal{G}(0,\beta(s))$, the following equation holds:*

$$\max_a Q^\star(s,a) = \beta(s) \log \int \exp\left(\frac{Q(s,a)}{\beta(s)}\right) \mathrm{d}a + g(s). \tag{10}$$

Proof. See prior work (Hui et al., 2023). □

With the lemmas above, we're ready to provide the proof for Proposition 1:

Proof. By definition, $V^\star(s) = \max_a Q^\star(s,a)$, and according to Lemma 2, we have:

$$V^\star(s) = \beta(s) \log \int \exp\left(\frac{Q(s,a)}{\beta(s)}\right) \mathrm{d}a + g(s).$$

And by Lemma 1, it exists:

$$\begin{aligned} V^\star(s) &= \beta(s) \log \int \exp\left(\frac{Q(s,a)+g(s,a)}{\beta(s)}\right) da \\ &= \beta(s) \log \int \exp\left(\frac{Q^\star(s,a)}{\beta(s)}\right) da. \end{aligned}$$

The last part is completed by directly applying Assumption 1. □

B.2 Proof for Proposition 2

Given $\hat{V}(s) = \mathbb{E}[V^\star(s)]$, Proposition 1 and Assumption 1, we can construct the proof for Proposition 2:

Proof.

$$\begin{aligned} \hat{V}(s) - V(s) &= \mathbb{E}\left[\beta(s) \log \int \exp\left(\frac{Q^\star(s,a)}{\beta(s)}\right) da - \beta(s) \log \int \exp\left(\frac{Q^\star(s,a)-g(s)}{\beta(s)}\right) da\right] \\ &= \mathbb{E}\beta(s)\left[\log \int \exp\left(\frac{Q^\star(s,a)}{\beta(s)}\right) da - \log \int \exp\left(\frac{Q^\star(s,a)}{\beta(s)}\right) da + \frac{g(s)}{\beta(s)}\right] \\ &= \mathbb{E}[g(s)] = \omega\beta(s). \end{aligned}$$

The final equation leverages a property of the Gumbel distribution to compute the expectation:

$$\mathbb{E}[g] = \omega\beta, \quad \text{where} \quad g \sim \mathcal{G}(0,\beta).$$

□

B.3 Proof for Proposition 3

Leveraging Assumption 2 and the properties of the Gumbel distribution, we can construct the following proof:

Proof.

$$\begin{aligned} P(Q(s,a) \le V(s)) &= 1 - P((Q(s,a) > V(s)) \\ &= 1 - P(-g(s) > 0) \\ &= 1 - P(g(s) < 0) \\ &= 1 - \exp(-1) = \alpha_1, \end{aligned}$$

where we denote $g(s) \sim \mathcal{G}(0, \beta(s))$.

□

B.4 Proof for Proposition 4

Similar to the proof of Proposition 3, the proof of Proposition 4 can be constructed using Assumption 2, Proposition 2, and the properties of the Gumbel distribution:

Proof.

$$\begin{aligned} P(Q(s,a) \leq \hat{V}(s)) &= 1 - P((Q(s,a) > \hat{V}(s)) \\ &= 1 - P(-g(s) - \omega\beta(s) > 0) \\ &= 1 - P(g(s) < -\omega\beta(s)) \\ &= 1 - \exp(-\exp(\omega)), \end{aligned}$$

where we denote $g(s) \sim \mathcal{G}(0, \beta(s))$. □

C Baseline Hyperparameter Settings

C.1 Extreme Q-Learning

Task (Variant)	**halfcheetah**	**hopper**	**walker2d**
medium	1.0	5.0	10.0
medium-rep	1.0	2.0	5.0
medium-exp	1.0	2.0	2.0

Table 5: XQL temperature settings (β) with dataset-specific tuning on D4RL locomotion datasets.

Task (Variant)	**pen**	**hammer**	**door**
human	5.0	0.5	1.0
cloned	0.8	5.0	5.0

Table 6: XQL temperature settings (β) with dataset-specific tuning on D4RL Adroit datasets.

Hyperparameter	**antmaze-umaze**	**antmaze-umaze-diverse**
β	1.0	5.0

Table 7: XQL temperature settings (β) with dataset-specific tuning on D4RL AntMaze datasets.

This section illustrates the temperature hyperparameter β settings used in the Extreme Q-Learning (XQL) baseline (Garg et al., 2023) across various offline reinforcement learning (RL) tasks from the D4RL and NeoRL2 benchmark. Dataset-specific β values are listed in Tables 5, 6, 7 and 8, corresponding to the results in Table 1 and the first row of results in Table 3 in the experiment section. Domain-consistent β settings are shown in Table 9 and relate to the second row of Table 3, while a fully consistent setting across all domains uses $\beta = 2.0$, as reported in the third row. The variability in optimal β values across tasks highlights XQL's sensitivity to its temperature parameter and the importance of tuning it carefully for each environment, as also emphasized in (Garg et al., 2023). For all other hyperparameter settings, we follow the XQL reproduction provided by (Gao & Rui, 2023).

Hyperparameter	**RocketRecovery**	**SafetyHalfCheetah**
β	2.0	2.0

Table 8: XQL temperature settings (β) with dataset-specific tunning on NeoRL2 datasets.

Hyperparameter	**D4RL Locomotion**	**D4RL Adroit**	**D4RL AntMaze**
β	2.0	5.0	0.6

Table 9: XQL consistent temperature settings (β) per domain.

C.2 MXQL and Other Baselines

For MXQL (Omura et al., 2024), we adopt the original JAX implementation and use the exact hyperparameter settings as reported in the paper. For the remaining baseline algorithms, we use the reproduced results provided by CORL (Tarasov et al., 2022).

D Implementation Details

This section provides a comprehensive overview of the hyperparameters and implementation specifics of our proposed QQL algorithm.

Our implementation of QQL is built upon the publicly available PyTorch implementation of Implicit Q-Learning (IQL) from CORL (Tarasov et al., 2022). We adopt the identical neural network architecture and retain the exact hyperparameter settings used for network updates in the original IQL implementation.

A crucial aspect of our approach is ensuring that $\beta(s)$ remains positive during policy optimization. To achieve this, we define $\beta(s)$ as the absolute value of $(\hat{V}_{\psi_2}(s) - V_{\psi_1}(s))/\omega$. For all other procedural steps, the direct value of $(\hat{V}_{\psi_2}(s) - V_{\psi_1}(s))/\omega$ is utilized. To prevent $\beta(s)$ from becoming infinitesimally small, we apply a lower bound clipping, setting $\beta(s) \geq \beta_{low} = 0.1$ specifically within the policy optimization routine. For consistency across all evaluated task-dataset combinations, we fix the generalization coefficient λ to 1.0 and the policy constraint weight ζ to 1.0.

Table 10 summarizes the detailed hyperparameters employed in our QQL algorithm.

Table 10: Detailed Hyperparameters of the QQL Algorithm.

Parameter	**Value**
V-function learning rate (α_V)	3×10^{-4}
Q-function learning rate (α_Q)	3×10^{-4}
Policy learning rate (α_π)	3×10^{-4}
Discount factor (γ)	0.99
Target network update rate (τ)	0.005
Batch size	256
β_{low}	0.1
λ	1.0
ζ	1.0
Hidden layer dimension	256
Number of hidden layers	2
Activation function	ReLU