Stabilizing Extreme Q-learning by Maclaurin Expansion

Motoki Omura
omura@mi.t.u-tokyo.ac.jp
The University of Tokyo

Takayuki Osa
osa@mi.t.u-tokyo.ac.jp
The University of Tokyo
RIKEN

Yusuke Mukuta
mukuta@mi.t.u-tokyo.ac.jp
The University of Tokyo
RIKEN

Tatsuya Harada
harada@mi.t.u-tokyo.ac.jp
The University of Tokyo
RIKEN

Abstract

In offline reinforcement learning, in-sample learning methods have been widely used to prevent performance degradation caused by evaluating out-of-distribution actions from the dataset. Extreme Q-learning (XQL) employs a loss function based on the assumption that Bellman error follows a Gumbel distribution, enabling it to model the soft optimal value function in an in-sample manner. It has demonstrated strong performance in both offline and online reinforcement learning settings. However, issues remain, such as the instability caused by the exponential term in the loss function and the risk of the error distribution deviating from the Gumbel distribution. Therefore, we propose Maclaurin Expanded Extreme Q-learning to enhance stability. In this method, applying Maclaurin expansion to the loss function in XQL enhances stability against large errors. This approach involves adjusting the modeled value function between the value function under the behavior policy and the soft optimal value function, thus achieving a trade-off between stability and optimality depending on the order of expansion. It also enables adjustment of the error distribution assumption from a normal distribution to a Gumbel distribution. Our method significantly stabilizes learning in online RL tasks from DM Control, where XQL was previously unstable. Additionally, it improves performance in several offline RL tasks from D4RL.

1 Introduction

Deep reinforcement learning has demonstrated good performance in many tasks, including robotics (Schulman et al., 2017; Haarnoja et al., 2018) and games (Mnih et al., 2013; 2015; Silver et al., 2016). During learning, the goal is to acquire the optimal policy by learning a value function, and the learning of the value function involves the Bellman update. Recently, Garg et al. (2023) proposed that based on the Extreme Value Theorem, the Bellman error follows a Gumbel distribution rather than the normal distribution assumed by traditional least squares methods. Consequently, they proposed an algorithm called Extreme Q-learning (XQL), which employs Gumbel Regression, a maximum likelihood estimation assuming a Gumbel error distribution. This method demonstrated excellent performance, primarily in offline RL. However, a significant issue with Gumbel Regression is its instability. The loss function contains an exponential term that can lead to too large or too small gradients, resulting in divergence or slow convergence. As a remedy, they used stabilization measures such as clipping and the max-normalization trick, but it was still unstable, particularly in online RL. Another problem is that the error distribution may not exactly match the Gumbel

distribution. Garg et al. (2023) demonstrated that based on the i.i.d. among state-action pairs and time steps, it becomes a Gumbel distribution, but in reality, the independence is not guaranteed due to the use of the same neural network for all states and actions. Additionally, in the actual Bellman updates of algorithms, many elements such as entropy maximization, target networks, and Clipped Double Q-learning are incorporated, which also affect the error distribution. As suggested in Garg et al. (2023), the resulting distribution may resemble a mix of normal and Gumbel distributions.

Thus, in this study, we propose a both simple and practical algorithm, Maclaurin Expanded Extreme Q-learning (MXQL), which not only stabilizes the Gumbel loss but also allows for the adjustment of the error distribution assumption from normal to Gumbel. The proposed method uses Expanded Gumbel loss, which is a Maclaurin expansion of the Gumbel loss. By reducing the order of expansion n, this loss, as shown to the left of Figure 1, mitigates excessively large or small gradients compared to the Gumbel loss.

As n increases, the loss function converges to the Gumbel loss, while for $n = 2$, it becomes the L2 loss. With the Gumbel loss, the estimated value function becomes a soft optimal value, and with the L2 loss, it becomes the value function under the behavior policy. These learning methods correspond to soft Q-learning (Haarnoja et al., 2017) and SARSA, respectively. By adjusting n between 2 and ∞, the method of estimating the value function can be adjusted between soft Q-learning and SARSA. In other words, a larger n leads to an unstable but optimal value function estimation, while a smaller n results in a stable but non-maximizing estimation, offering a trade-off between stability and optimality. This is analogous to adjusting the parameter τ in IQL's expectile loss from 1 to a smaller value for stabilization. IQL adjusts between SARSA and Q-learning using the parameter τ, while MXQL adjusts between SARSA and soft Q-learning using the parameter n.

The assumed error distribution follows a normal distribution when $n = 2$ because the loss is the L2 loss. When n is large, it follows a Gumbel distribution. Therefore, by adjusting n between 2 and a larger value, the assumed error distribution can be adjusted between the normal and Gumbel distributions. In practice, the Bellman error is influenced by various factors such as the target network and double Q-learning, which suggests that, according to the central limit theorem, it may approximate a normal distribution. The assumed error distribution is illustrated to the right of Figure 1, showing that as n increases, the distribution transitions from normal to Gumbel.

In the experiments, similar to Garg et al. (2023), we compared performance using DM Control tasks (Tassa et al., 2018; Tunyasuvunakool et al., 2020) for online RL and D4RL tasks (Fu et al., 2020) for offline RL. In online RL scenarios, where XQL was previously unstable, we observed improved stability. Furthermore, in offline RL, our method demonstrated superior performance compared to existing methods, including XQL, across several tasks.

The contributions of this study are as follows:

- Verifying XQL's instability through preliminary experiments and demonstrating its instability when the data distribution significantly deviates from the assumed error distribution.
- Proposing a novel loss function, the Expanded Gumbel loss, by applying a Maclaurin expansion to the Gumbel loss in XQL, aimed at stabilizing XQL.
- Demonstrating improved stability and performance by using Maclaurin Expanded Extreme Q-learning, which incorporates the Expanded Gumbel loss into XQL, across numerous tasks in both online RL and offline RL.

2 Background

2.1 Reinforcement Learning

We address the reinforcement learning (RL) problem within the framework of a Markov decision process (MDP) represented by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, d)$. Here, $\mathcal{S}$ represents the state space, where s denotes

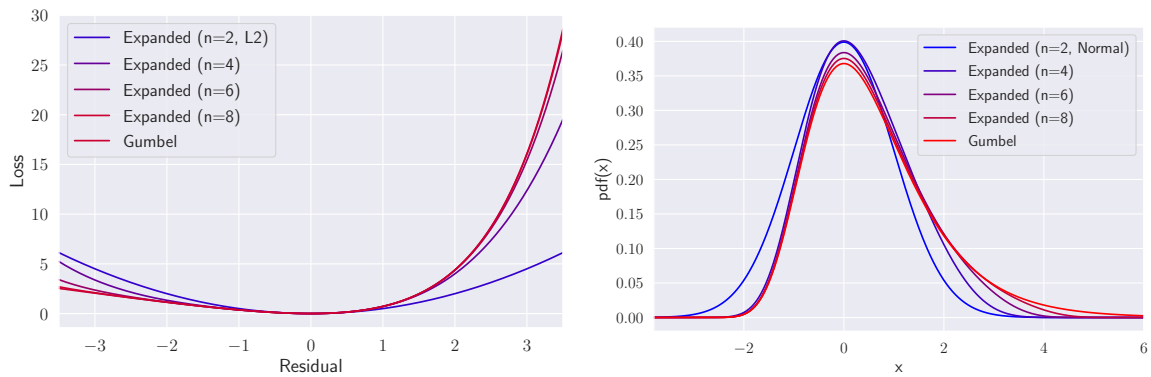

Figure 1: **Left**: Illustration of Gumbel loss ($\beta = 1$) alongside Expanded Gumbel loss derived from it. When $n = 2$, the loss aligns identically with the L2 loss. **Right**: Depiction of the error distribution assumed by Gumbel loss and Expanded Gumbel loss. Specifically, when $n = 2$ (corresponding to L2 loss), a normal distribution is assumed.

a specific state. $\mathcal{A}$ represents the action space, with a denoting a specific action. $\mathcal{P}(s_{t+1}|s_t, a_t)$ defines the transition probability from one state to another given an action. $r(s, a)$ is the reward function, representing the reward received after taking action a in state s. γ represents the discount rate. Lastly, $d(s_0)$ represents the initial state's probability density. The policy $\pi(a|s)$ is defined as the probability of taking a specific action given the state. The objective in RL is to discover the policy that maximizes the expected sum of discounted rewards, denoted as $\mathbb{E}[R_0|\pi]$. The return R_t is calculated as $R_t = \sum_{k=t}^{T} \gamma^{k-t} r(s_k, a_k)$, representing the total discounted rewards from time t.

2.2 Extreme Q-learning

In this section, we explain XQL, which forms the foundation of this research. First, based on the Gumbel Error Model, we explain how the Bellman error follows the Gumbel distribution. Then, we outline the Gumbel Regression for learning the value function, effective for addressing the error distribution of the Gumbel distribution. Subsequently, we examine the role of the hyperparameter β and discuss the inherent limitations of XQL.

2.2.1 Gumbel Error Model

The Gumbel Error Model suggests that the Bellman error follows a Gumbel distribution. Given multiple instances of $\hat{Q}$, each representing a different simulation of the model, the target defined by the Bellman optimal operator is given as:

$$\hat{\mathcal{B}}^* \hat{Q}_t = r + \gamma \max_{a'} \hat{Q}_t(s', a') = r + \gamma \max_{a'} (\bar{Q}_t(s', a') + \epsilon_t(s', a')) \tag{1}$$

Here, $\hat{Q}_t(s, a) = \bar{Q}_t(s, a) + \epsilon_t(s, a)$. The error ϵ_t, an i.i.d. variable with zero mean, is combined with the mean value $\bar{Q}_t$. Thus, the mean value $\bar{Q}_{t+1}$ for the estimated value $\hat{Q}_{t+1}$ is given by:

$$\bar{Q}_{t+1}(s, a) = r + \gamma \mathbb{E}_{s'|s,a}[\mathbb{E}_{\epsilon_t}[\max_{a'} (\bar{Q}_t(s', a') + \epsilon_t(s', a'))]] \tag{2}$$

Despite ϵ_t having a zero mean, the maximization operation introduces a positive expectation bias into $\bar{Q}_{t+1}$, resulting in an overestimation bias in Q-learning, specifically when using function approximators. Furthermore, if ϵ_t is uncorrelated with s_t, a_t, and across different time steps, according to the Extreme Value Theorem, ϵ_t will converge to a Gumbel distribution over time. The error distribution violates the least squares method's assumption that the error follows a normal distribution.

2.2.2 Gumbel Regression

Based on the properties of the Gumbel Error Model, they proposed Gumbel Regression, a maximum likelihood estimation assuming a Gumbel distribution as the error distribution. The loss function in Gumbel Regression is as follows:

$$L(h) = \mathbb{E}_{x_i \sim \mathcal{D}}[e^{(x_i - h)/\beta} - (x_i - h)/\beta - 1] \tag{3}$$

where x_i is the samples and h is the parameter to be estimated. By minimizing this loss function, the estimated parameters h are found as $h = \beta \log \mathbb{E}_{x_i \sim \mathcal{D}}[e^{x_i/\beta}]$. This expression is analogous to the log-partition function (LogSumExp), where the summation is replaced by an expectation.

In XQL, this property is applied to Q-learning, directly estimating the soft-value function ($V^*(s) = \beta \log \sum_a \mu(a \mid s) \exp(Q(s,a)/\beta)$) in maximum entropy RL. The loss function becomes:

$$L(V) = \mathbb{E}_{s,a \sim \mu}[e^{(Q(s,a) - V(s))/\beta} - (Q(s,a) - V(s))/\beta - 1] \tag{4}$$

where μ is the behavior policy that generated the sampled state-action pairs. As the estimation of the value function eliminates the need for samples from the current policy, it avoids the computation of Q-values using out-of-distribution actions, thereby enhancing performance, particularly in offline RL scenarios. In practical online RL applications, the last policy is used for μ, and a trust region update (Schulman et al., 2015) is performed.

The Q-function is learned using the least squares method as follows:

$$L(Q) = \mathbb{E}_{s,a,s' \sim \mathcal{D}}[(r(s,a) + \gamma V(s') - Q(s,a))^2] \tag{5}$$

2.2.3 Role of β

The soft optimal value estimated in Eq. (4) is derived from the following objective function in Maximum Entropy RL.

$$J(\pi) = \mathbb{E}_{s,a \sim \pi}\left[Q(s,a)\right] - \beta D_{\mathrm{KL}}(\pi || \mu) \tag{6}$$

Here, $\beta D_{\mathrm{KL}}(\pi||\mu)$ serves as a conservative term against deviations from the behavior policy, with β modulating the level of conservatism. As can be seen from Eq. (4), this parameter, β, also impacts learning stability: small β values can increase gradients, risking divergence, while large β values may reduce gradient magnitudes, decelerating convergence. Therefore, while β modifies conservatism, it does not ensure stability across all values.

2.2.4 Limitations

The first limitation concerns instability, which complicates the adjustment of conservatism through the parameter β. Techniques such as clipping and max normalization were employed to achieve stabilization:

```
def gumbel_loss(pred, label, beta, clip):
    z = (label - pred)/beta
    z = torch.clamp(z, -clip, clip)
    max_z = torch.max(z)
    max_z = torch.where(max_z < -1.0, torch.tensor(-1.0), max_z)
    max_z = max_z.detach()
    loss = torch.exp(z - max_z) - z*torch.exp(-max_z) - torch.exp(-max_z)
    return loss.mean()
```

Nevertheless, instability persists, and in certain tasks, learning can collapse, necessitating a restart (Garg et al., 2023). This instability is demonstrated in the following section through preliminary experiments.

The second limitation is that the error distribution may not be a Gumbel distribution. The Gumbel Error Model assumes independence between state-action pairs and steps; however, maintaining this independence becomes challenging when the same network is utilized for all training samples. Moreover, practical algorithms integrate techniques such as target networks and Clipped Double Q-learning, which additionally influence the error distribution. Plots in the appendix of Garg et al. (2023) demonstrate distributions that deviate from the Gumbel distribution. They also suggest that the error distribution might combine normal and Gumbel elements. We hypothesize that this mixed distribution, influenced by various factors and the Central Limit Theorem, resembles an intermediate between normal and Gumbel distributions.

3 Maclaurin Expanded Extreme Q-learning

As mentioned above, XQL has limitations; it can be unstable, and its assumptions regarding the error distribution may not be met. Therefore, we propose a method named Maclaurin Expanded Extreme Q-learning (MXQL), which stabilizes XQL and facilitates adjustment between normal and Gumbel distributions in the error distribution assumption.

3.1 Instability related to β

In Extreme Q-learning, β was adjusted for conservatism, learning efficiency, and stability. In other words, even if one aims to adjust the level of conservatism using β, certain β values may prove unstable and unusable depending on the data. To illustrate this, we conducted a simple preliminary experiment.

For scalar data represented as x_i, parameter h, and Gumbel noise ϵ_i, we perform Gumbel Regression using Eq. (3) as $x_i = h + \epsilon_i$ where $\epsilon_i \sim -\mathcal{G}(0, \beta_{reg}), x_i \sim -\mathcal{G}(0, \beta_{data})$. This implies that when β_{reg} and β_{data} are equal, the assumptions concerning the error distribution in Gumbel Regression are fully met. We varied the β of the data (β_{data}) and the β used in Gumbel Regression (β_{reg}) respectively, estimated them, and analyzed how differences between β_{data} and β_{reg} affected the estimation results. The estimated h was assessed based on its deviation from the true value ($\log \sum e^{x_i/\beta_{reg}}$). The results are presented in Figure 2.

When β_{data} and β_{reg} are equal, that is, when the distribution of the data assumed in Gumbel Regression is the same as the distribution of the data used for estimation, the accurate estimation can be achieved with fewer updates. When β_{data} and β_{reg} are different, the required number of updates increases. Furthermore, if β_{reg} is too small relative to β_{data}, the gradient diverges and learning collapses. Similar outcomes have been noted in higher-dimensional and online RL tasks, demonstrated by Garg et al. (2023), with learning collapse requiring restarts in specific scenarios.

These results suggest that in Gumbel Regression, it is necessary to choose a β that approximates the distribution of the learning data closely, thereby narrowing the range within which conservativity can be adjusted. Therefore, we propose a method to stabilize Gumbel Regression adaptable to diverse β values.

3.2 Expanded Gumbel Loss

We introduce the Expanded Gumbel loss, which applies a Taylor (Maclaurin) expansion to the Gumbel loss at the point where the residual $(x_i - h)$ equals zero.

$$L(h) = \mathbb{E}_{x_i \sim \mathcal{D}} \left[\sum_{j=2}^{n} \frac{(x_i - h)^j}{j! \beta^j} \right] \tag{7}$$

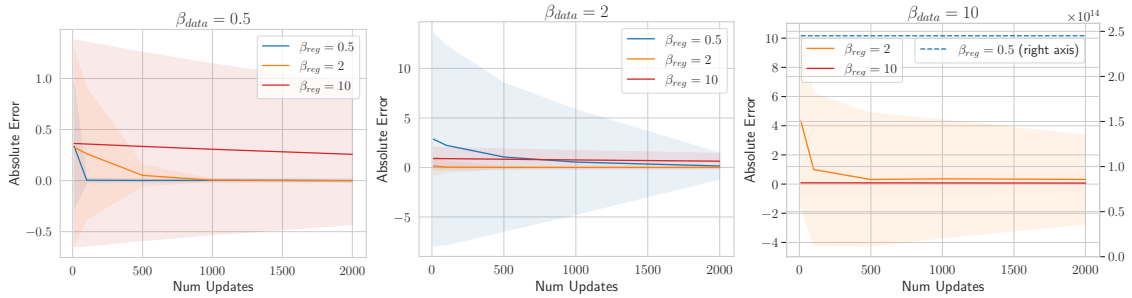

Figure 2: Results of Gumbel Regression with varying parameters β_{data} and β_{reg} set to (0.5, 2, 10). The absolute error refers to the absolute difference between the estimated parameter h and the true value, defined as $\log \sum e^{x_i/\beta_{reg}}$. This error was measured at update counts of [10, 100, 500, 1000 2000]. The experiment was conducted 100 times to obtain average values, and the standard deviation is depicted as shaded areas.

where n is the order of expansion. As shown on the left side of Figure 1, the Expanded Gumbel loss exhibits a less steep gradient on the right side and a steeper gradient on the left side compared to the traditional Gumbel loss. This modification resolves issues such as the collapse of learning and slow convergence observed in the preliminary experiments. Moreover, by adjusting the expansion order n, the degree of stabilization can be altered: smaller values of n lead to greater deviation from the Gumbel loss but enhance stability. It should be noted that n is selected from even numbers to ensure the loss remains positive. In MXQL, similar to XQL, the Expanded Gumbel loss is employed for estimating the value function, and the loss function is defined as follows:

$$L(V) = \mathbb{E}_{s,a\sim\mu}\left[\sum_{j=2}^{n} \frac{(Q(s,a) - V(s))^j}{j!\beta^j}\right] \tag{8}$$

Learning the Q-function uses Eq. (5) in the same manner as in XQL.

As shown in Eq. (8), when $n = 2$, the loss function becomes the L2 loss. The V-function represents the expected value of Q for actions taken under the behavior policy. In other words, the learning of the value function follows a SARSA-based approach, and the estimated value function is the value under the behavior policy rather than the optimal policy. When n approaches ∞, Eq. (8) transforms into the Gumbel loss. The value function estimated by the Gumbel loss corresponds to the soft optimal value in maximum entropy RL. Thus, the learning of the value function follows a soft Q-learning-based approach. This indicates a trade-off between stability and optimality: when n is large, the estimated value function is more optimal but less stable, whereas when n is small, the value function is more stable but but does not fully maximize.

This trade-off is similar to the parameter τ in IQL (Kostrikov et al., 2022), which adjusts the expectile loss. In IQL, when $\tau = 0.5$, the loss becomes the L2 loss, corresponding to $n = 2$ in MXQL, while $\tau = 1$ in IQL corresponds to $n = \infty$ in MXQL. IQL adjusts between SARSA and Q-learning using the parameter τ, while MXQL adjusts between SARSA and soft Q-learning using the parameter n. Since IQL typically uses values of τ less than 1, such as 0.7 or 0.9, this suggests the necessity of stabilizing the loss in XQL towards the L2 loss.

In the following section, we present the analysis of the error distribution from the perspective of the Expanded Gumbel loss.

3.3 Perspective of Error Distribution

The Gumbel loss was derived from the maximum likelihood estimation that assumes a Gumbel distribution for the error distribution. Next, we consider the error distribution assumed by the

proposed Expanded Gumbel loss. The error distribution for each n is shown on the right side of Figure 1. When n is large, the Expanded Gumbel loss approaches the Gumbel loss due to the nature of the Taylor expansion, and consequently, the assumed error distribution also approaches the Gumbel distribution. When $n = 2$, which is the smallest value, the Expanded Gumbel loss takes a form equivalent to L2 loss. This loss corresponds to the least squares method, which assumes a normal distribution for the error distribution. As n increases, the error distribution shifts from a normal to a Gumbel distribution. In other words, by adjusting the expansion order, it is possible to modulate the error distribution between a normal and a Gumbel distribution. It is conceivable that error distributions influenced by various factors tend to have properties close to a normal distribution due to the Central Limit Theorem. Moreover, preliminary experiments confirmed the importance of the closeness between the assumed error distribution and the data distribution, suggesting that adjusting the distribution using n is likely to be effective.

We name our method, which substitutes XQL's Gumbel loss with Expanded Gumbel loss, Maclaurin Expanded Extreme Q-learning (MXQL). Our approach is based on XQL, simply changing the loss function for V-function. The implementation details are provided in the appendix. In MXQL, the expansion order n is a hyperparameter. However, it should be noted that the clipping size is also a hyperparameter in XQL, and the number of hyperparameters has not changed compared to XQL.

4 Experiments

In our experiments, we assessed the stability and performance of Maclaurin Expanded Extreme Q-learning (MXQL) in comparison with Extreme Q-learning (XQL) across both online RL and offline RL scenarios for various β values. The implementation is based on the official XQL framework, and we adhered to the same hyperparameters as those used in the XQL setup. Further details are provided in the appendix.

4.1 Online RL

In online RL, experiments were conducted using five tasks from the DM control suite. These tasks include Quadruped-run, Hopper-hop, Walker-run, Cheetah-run, as experimented in Garg et al. (2023), in addition to Humanoid-walk. Both XQL and MXQL used SAC (Haarnoja et al., 2018) as the base algorithm. Experiments were conducted with a range of β values ([0.1, 0.5, 1, 2, 5]), which includes additional smaller values ([0.1, 0.5]) not used in the experiments by Garg et al. (2023) ([1, 2, 5]). In Garg et al. (2023), when XQL became unstable, it was restarted, but in this study, no restarts were performed for a fair comparison. The plot of average returns when the expansion order is 8 is shown in Figure 3.

In most tasks, MXQL demonstrated superior performance compared to XQL. Particularly in cases of small β, XQL often failed to learn adequately. This implies that XQL could not learn with a small trust region, whereas MXQL was able to stabilize learning even for small β. In other words, MXQL allows for a wider range of β selections. The scores averaged across all tasks are shown in Table 2. Experiments were conducted with $n = 4, 8, 12$, and stable learning was achieved across all values. The final average scores and standard deviations for each task when using the best β in XQL and MXQL ($n = 8$) are shown in Table 3, demonstrating improved performance with the tuned β.

4.2 Offline RL

To ensure a fair comparison with XQL, experiments were conducted on the same Gym locomotion, AntMaze, and Franka Kitchen tasks as those reported in Garg et al. (2023). The scores are displayed in Table 1 and are compared with those from Brandfonbrener et al. (2021); Fujimoto & Gu (2021); Kumar et al. (2020); Kostrikov et al. (2022); Garg et al. (2023). The proposed method, MXQL, demonstrated the best scores in several tasks compared to the scores of existing methods reported in Kostrikov et al. (2022); Sikchi et al. (2024), and outperformed XQL in most tasks. In the AntMaze

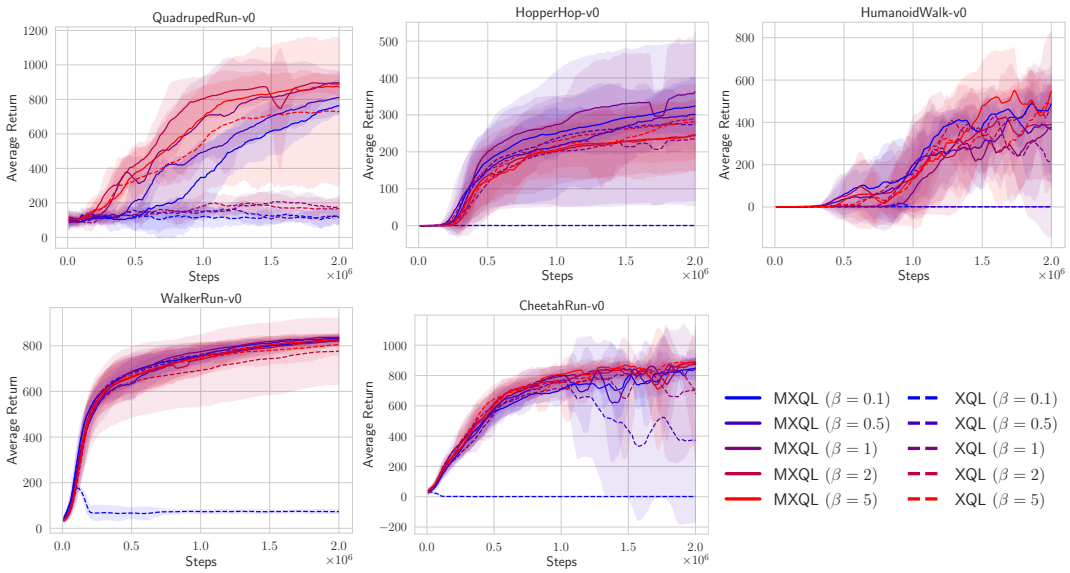

Figure 3: Performance comparison of MXQL (n=8) and XQL on DM Control tasks for online RL.

medium and large tasks, MXQL underperforms compared to IQL, suggesting that the soft optimal value estimated in both XQL and MXQL may not be well-suited for these particular tasks.

Dataset	BC	Onestep RL	TD3+BC	CQL	IQL	XQL	MXQL
halfcheetah-med	42.6	**48.4**	**48.3**	44.0	47.4	47.4	46.5 ± 0.3
hopper-med	52.9	59.6	59.3	58.5	66.3	**68.5**	**68.3** ± 7.3
walker2d-med	75.3	81.8	**83.7**	72.5	78.3	81.4	71.9 ± 3.9
halfcheetah-med-rep	36.6	38.1	**44.6**	**45.5**	44.2	44.1	44.1 ± 0.3
hopper-med-rep	18.1	**97.5**	60.9	95.0	94.7	95.1	**98.2** ± 3.2
walker2d-med-rep	26.0	49.5	**81.8**	77.2	73.9	58.0	57.9 ± 8.8
halfcheetah-med-exp	55.2	**93.4**	90.7	91.6	86.7	90.8	88.2 ± 4.4
hopper-med-exp	52.5	103.3	98.0	**105.4**	91.5	94.0	**105.1** ± 10.1
walker2d-med-exp	107.5	**113.0**	110.1	108.8	109.6	110.1	109.9 ± 0.1
antmaze-umaze	54.6	64.3	78.6	74.0	87.5	47.7	**88.3** ± 2.1
antmaze-umaze-div	45.6	60.7	71.4	**84.0**	62.2	51.7	53.2 ± 9.7
antmaze-med-play	0.0	0.3	10.6	61.2	**71.2**	31.2	50.8 ± 2.7
antmaze-med-div	0.0	0.0	3.0	53.7	**70.0**	0.0	52.2 ± 6.6
antmaze-large-play	0.0	0.0	0.2	15.8	39.6	10.7	18.7 ± 5.9
antmaze-large-div	0.0	0.0	0.0	14.9	**47.5**	31.3	14.3 ± 6.4
kitchen-complete	**65.0**	-	-	43.8	62.5	56.7	**64.2** ± 10.3
kitchen-partial	38.0	-	-	**49.8**	46.3	48.6	47.1 ± 8.7
kitchen-mixed	51.5	-	-	51.0	51.0	40.4	**71.9** ± 3.6

Table 1: Average normalized scores on Gym locomotion, AntMaze and Kitchen tasks. Highlighted results are within one performance point of those achieved by the best-performing algorithm, and the standard deviation across six seeds is displayed as ± for MXQL.

5 Related Work

XQL (Garg et al., 2023) has evolved based on MaxEnt RL (Bloem & Bambos, 2014; Haarnoja et al., 2017; 2018) and allowed for the estimation of soft-values through Gumbel loss without access to entropy, similar to Heess et al. (2015); Haarnoja et al. (2017). XQL has been successfully combined

β	XQL	MXQL n=4	MXQL n=8	MXQL n=12
0.1	38.9	619.0	648.0	642.9
0.5	319.2	665.4	634.8	598.9
1	432.1	653.9	669.9	622.8
2	474.1	626.5	643.3	643.6
5	638.7	654.3	674.7	642.0

Table 2: Average scores of XQL and MXQL with various n values for five tasks from DM Control in online RL. Scores underlined with dashes represent significant differences between XQL and MXQL, as determined by a t-test with a significance level of 0.05.

with existing methods widely used in online RL, such as Fujimoto et al. (2018); Haarnoja et al. (2018). However, its instability and sensitivity to the value of β necessitate careful tuning. In MXQL, the loss function has been stabilized through Maclaurin expansion. Bas-Serrano et al. (2021); Hui et al. (2023) have also used loss functions different from L2.

In offline RL, some methods regularize through conservatism (Wu et al., 2019; Kumar et al., 2019; Fujimoto et al., 2019; Kumar et al., 2020; Fujimoto & Gu, 2021; Nair et al., 2021) and others that directly model the greedy policy (Peng et al., 2019; Brandfonbrener et al., 2021; Chen et al., 2021). XQL was able to learn without direct access to the current policy, similar to Kostrikov et al. (2022); Xu et al. (2023), while introducing conservatism. Similar to online RL, methods for stabilization have been employed in offline RL; notably, the stabilization in MXQL has shown performance improvements in certain offline RL tasks. Recently, there have been methods that have led to performance improvements by using Diffusion Models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) as the policy (Wang et al., 2023; Hansen-Estruch et al., 2023), and applying this approach in MXQL could be a future direction.

6 Conclusion

A recent study (Garg et al., 2023) suggested that the Bellman error might follow a Gumbel distribution, challenging traditional least squares methods. In response, XQL, utilizing Gumbel Regression, was proposed but encountered stability issues in online RL due to extreme gradient values and the often incorrect assumption that state-action pairs are i.i.d.. The study introduces Maclaurin Expanded Extreme Q-learning (MXQL), stabilizing Gumbel loss and allowing adjustment between normal and Gumbel error distributions using Expanded Gumbel loss. This modification offers a practical solution to the instability and error distribution assumption issues seen in XQL, showing improved performance and stability in both online and offline RL scenarios.

Acknowledgments

This work was partially supported by JST Moonshot R&D Grant Number JPMJPS2011, CREST Grant Number JPMJCR2015 and Basic Research Grant (Super AI) of Institute for AI and Beyond of the University of Tokyo. T.O. was partially supported by JSPS KAKENHI Grant Number JP23K18476.

References

Joan Bas-Serrano, Sebastian Curi, Andreas Krause, and Gergely Neu. Logistic q-learning. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 3610–3618. PMLR, 13–15 Apr 2021.

Michael Bloem and Nicholas Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. In *53rd IEEE Conference on Decision and Control*, pp. 4911–4916, 2014. doi: 10.1109/CDC.2014.7040156.

David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 4933–4946. Curran Associates, Inc., 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 M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 15084–15097. Curran Associates, Inc., 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 Gu. A minimalist approach to offline reinforcement learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 10–15 Jul 2018.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 09–15 Jun 2019.

Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme q-learning: Maxent RL without entropy. In *International Conference on Learning Representations*, 2023.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1352–1361. PMLR, 06–11 Aug 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies, 2023.

Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

David Yu-Tung Hui, Aaron Courville, and Pierre-Luc Bacon. Double gumbel q-learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

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, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets, 2021.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual RL: Unification and new methods for reinforcement and imitation learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=xt9Bu66rqv.

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.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: https://doi.org/10.1016/j.simpa.2020.100022.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning, 2019.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline RL with no OOD actions: In-sample learning via implicit value regularization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ueYYgo2pSSU.

A Experiments

A.1 Online RL

In the experiments of MXQL and XQL, the implementation is based on the official implementation of (Garg et al., 2023), with only the loss function changed. The hyperparameters are the same as in (Garg et al., 2023). In all experiments on online RL, 5 seeds are used, and the 95% confidence interval is shown as a shaded area. The results of changing the order n of expansion in MXQL are shown in Figure 4 and Figure 5.

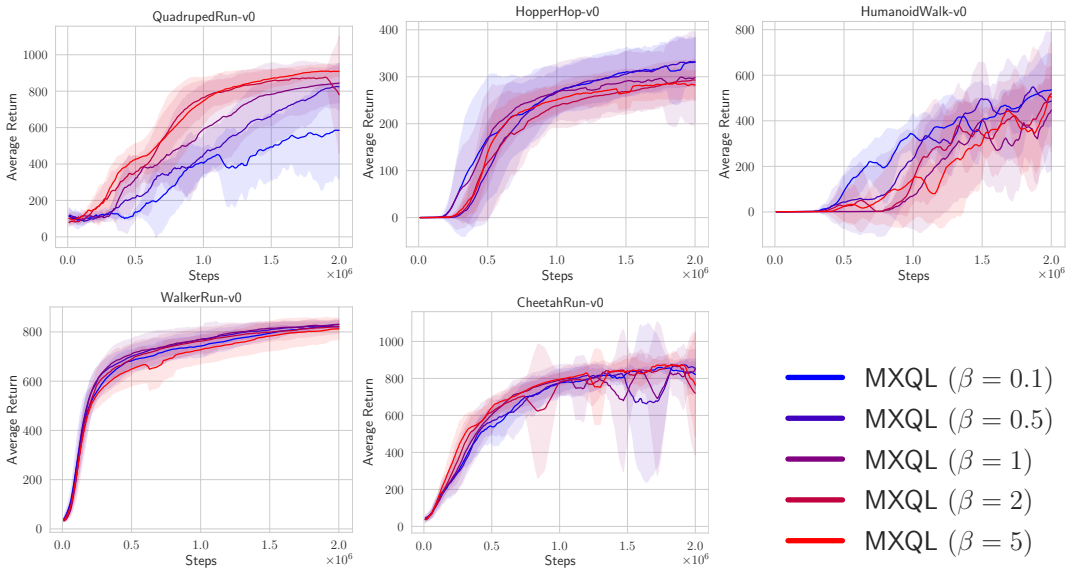

Figure 4: Performance of MXQL (n=4) in the online RL tasks from DM Control.

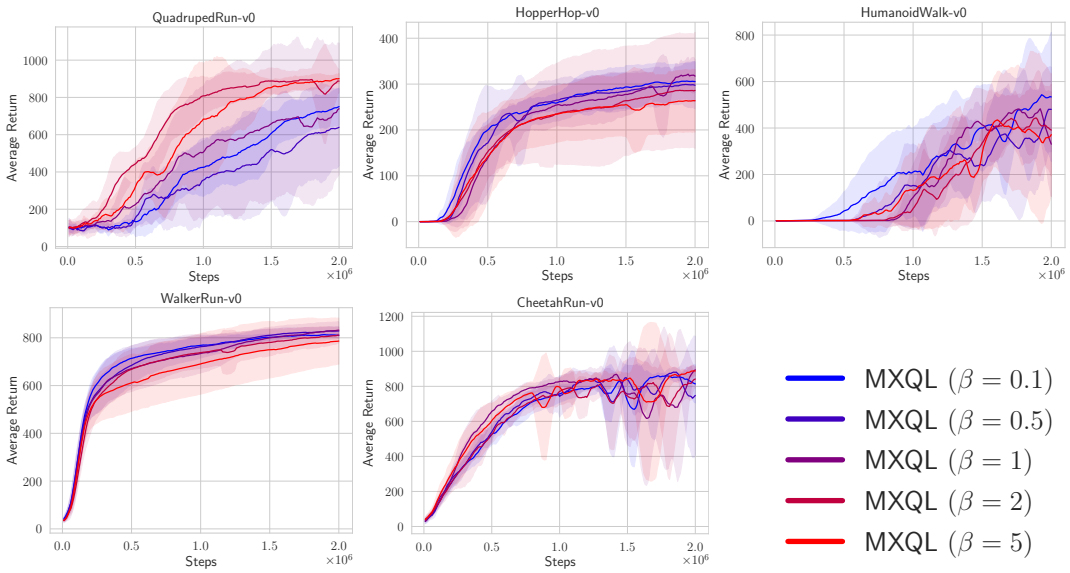

Figure 5: Performance of MXQL (n=12) in the online RL tasks from DM Control.

Task	XQL		MXQL	
	β	Score	β	Score
QuadrupedRun-v0	5	730.2 ± 303.8	1	896.0 ± 51.4
HopperHop-v0	2	287.4 ± 9.1	1	362.7 ± 115.7
HumanoidWalk-v0	5	487.1 ± 60.3	5	546.1 ± 45.0
WalkerRun-v0	0.5	826.0 ± 19.4	1	837.2 ± 5.3
CheetahRun-v0	5	890.0 ± 16.9	1	887.6 ± 7.6

Table 3: The final average score and standard deviation when using the best β in XQL and MXQL (n=8).

A.2 Offline RL

In offline RL, the official implementation is also used, and the hyperparameters are the same as in Garg et al. (2023). In Gym tasks, "-v2" was used, while in AntMaze and Kitchen tasks, "-v0" was employed. The batch size and the update frequency of the V-function, which were tuned in (Garg et al., 2023), are not tuned. In XQL, the β and the size of clipping, and in MXQL, the β and the order n of expansion have been tuned, and a common value for each domain has been used. The range for tuning β is the same as for Garg et al. (2023), which is [0.6, 0.8, 1, 2, 5]. The n was selected from $[4, 8, 12, 16, 20]$. These values are shown in Table 4. In Garg et al. (2023), the evaluation was based on the best score during the learning process rather than the final score, and therefore, the XQL scores in Table 1 are cited from Sikchi et al. (2024). The experiments are conducted using 6 random seeds.

Dataset	XQL		MXQL	
	β	Clip	β	n
Gym	2	7	2	20
AntMaze	0.6	7	1	8
Kitchen	5	7	1	4

Table 4: Hyperparameters in offline RL tasks from D4RL. The hyperparameters for XQL are the same as those used in (Garg et al., 2023).

B Details of the Figures

In the calculation of the error distribution in the right of Figure 1, the loss function, which is the log-likelihood, is applied with "-exp", and a coefficient for normalization is multiplied. This coefficient is calculated to ensure that the integral of the distribution equals one, using "scipy.integrate.quad" (Virtanen et al., 2020).

In the preliminary experiments of Gumbel Regression in Figure 2, the estimation was performed using stochastic gradient descent with 10,000 data. The learning rate was set at 0.02, and the batch size was 32. The mean and standard deviation were calculated across 100 experiments.