BRAC+: GOING DEEPER WITH BEHAVIOR REGULARIZED OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Online interactions with the environment to collect data samples for training a Reinforcement Learning agent is not always feasible due to economic and safety concerns. The goal of Offline Reinforcement Learning (RL) is to address this problem by learning effective policies using previously collected datasets. Standard off-policy RL algorithms are prone to overestimations of the values of out of distribution (less explored) actions and are hence unsuitable for Offline RL. Behavior regularization, which constraints the learned policy within the support set of the dataset, has been proposed to tackle the limitations of standard off-policy algorithms. In this paper, we improve the behavior regularized offline reinforcement learning and propose *BRAC+*. We use an analytical upper bound on KL divergence as the behavior regularizor to reduce variance associated with sample based estimations. Additionally, we employ state-dependent Lagrange multipliers for the regularization term to avoid distributing KL divergence penalty across all states of the sampled batch. The proposed Lagrange multipliers allow more freedom of deviation to high probability (more explored) states leading to better rewards while simultaneously restricting low probability (least explored) states to prevent out of distribution actions. We also propose several practical enhancements to further improve the performance. On challenging locomotion offline RL benchmarks, BRAC+ matches the state-of-the-art approaches on single-modal datasets and outperforms them on multi-modal datasets.

1 INTRODUCTION

Reinforcement Learning (RL) has shown great success in a wide range of applications including board games (Silver et al., 2016), strategy games (Vinyals et al., 2019), energy systems (Zhang et al., 2019), robotics (Lin, 1992), recommendation systems (Choi et al., 2018), etc. The success of RL relies heavily on extensive online interactions with the environment for exploration. However, this is not always feasible in the real world as it can be expensive or dangerous (Levine et al., 2020).

Offline RL, also known as batch RL, avoids online interactions with the environment by learning from a static dataset that is collected in an offline manner (Levine et al., 2020). While standard off-policy RL algorithms (Mnih et al., 2013; Lillicrap et al., 2016; Haarnoja et al., 2018a) can, in theory, be employed to learn from an offline data, in practice, they perform poorly due to distributional shift between the behavior policy (probability of actions conditional on states as observed in the dataset) of the collected dataset and the learned policy (Levine et al., 2020). The distributional shift manifests itself in form of overestimation of the out of distribution (OOD) actions leading to erroneous Bellman backups.

Prior works tackle this problem via behavior regularization (Fujimoto et al., 2018b; Kumar et al., 2019; Wu et al., 2019; Siegel et al., 2020). This ensures that the learned policy stays "close" to the behavior policy. This is achieved by adding a regularization term that calculates the f-divergence between the learned policy and the behavior policy. Kernel Maximum Mean Discrepancy (MMD) (Gretton et al., 2007), Wasserstein distance and KL divergence are widely used (Wu et al., 2019). The regularization term is either fixed (Wu et al., 2019), or tuned via dual gradient descent (Wu et al., 2019), or applied using a trust region objective (Siegel et al., 2020).

In this paper, we propose improvements to the Behavior Regularized Actor Critic (BRAC) algorithm proposed in (Wu et al., 2019). To obtain the same, we observe that sample based estimation of di-

vergence measures is computationally expensive and prone to higher variance. Therefore, to reduce variance, we use an analytical function that provides an upper bound on the KL divergence measure as the regularization term in the objective function. Moreover, we show that current works that apply the regularization term i.e. the divergence measure, on the entire batch, end up distributing the penalty over all states in the batch in amounts inversely proportional to the state's probability of occurrence in the batch. This needlessly restricts the deviation of highly explored states while allowing less explored ones to deviate farther leading to OOD actions. To address the same, we employ state dependent Lagrange multipliers for the regularization terms and automatically tune their strength using state-wise dual gradient descent. In addition, we use several practical enhancements such as the use of entropy regularization, suitable policy initialization to avoid difficulties in convergence, and policy evaluation masking to allow longer training without the convergence being impacted due to less explored states in the dataset.

We call our improved algorithm BRAC+ following (Wu et al., 2019). Our experiments suggest that BRAC+ outperforms existing state-of-the-art model-free and model-based offline RL algorithms in multi-modal datasets on the D4RL benchmark (Fu et al., 2020).

2 Background

Markov Decision Process RL algorithms aim to solve Markov Decision Process (MDP) with unknown dynamics. A Markov decision process (MDP) (Sutton & Barto, 2018) is defined as a tuple $< \mathcal{S}, \mathcal{A}, R, P, \mu >$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $R(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ defines the intermediate reward when the agent transitions from state s to s' by taking action a, $P(s'|s, a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the probability when the agent transitions from state s to s' by taking action a, $\mu : \mathcal{S} \to [0, 1]$ defines the starting state distribution. The objective of reinforcement learning is to select policy $\pi : \mu \to P(A)$ to maximize the following objective:

$$J(\pi) = \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu, a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim P(\cdot|s_t, a_t)}} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})] \tag{1}$$

Offline Reinforcement Learning The goal of offline RL is to learn policy π_θ from a fixed dataset $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}_{i=1}^N$ consisting of single step transitions $\{(s_i, a_i, s'_i, r_i)\}$. The dataset is assumed to be collected using a behavior policy π_β which denotes the conditional distribution $p(a|s)$ observed in the dataset. Note that π_β may consist of multi-modal policy distribution. In principle, standard off-policy RL algorithms using a replay buffer (Mnih et al., 2013; Lillicrap et al., 2016; Haarnoja et al., 2018a) can directly learn from $\mathcal{D}$. The key challenge resides in the policy evaluation step:

$$Q_\psi = \arg\min_\psi [(Q_\psi(s, a) - (r(s, a) + \gamma \mathbb{E}_{a' \sim \pi_\theta} Q_{\psi'}(s', a')))]^2 \qquad \text{(policy evaluation)} \tag{2}$$

In this step, the target Q value depends on the learned policy. If the learned policy distribution π_θ diverges from the data distribution π_β, it results in evaluation of target Q values using out-of-distribution (OOD) actions. Such evaluations are prone to errors. Erroneous overestimation of values get exploited by the policy update preventing the algorithm from learning useful policies. In order to avoid such cases, behavior regularization is adopted to force the learned policy to stay "close" to the behavior policy (Fujimoto et al., 2018b; Kumar et al., 2019; Wu et al., 2019).

3 Improving Behavior Regularized Offline Reinforcement Learning

In this section, we discuss and propose two non-trivial improvements to the Behavior Regularized Actor Critic (BRAC) offline reinforcement learning (Wu et al., 2019). BRAC (Wu et al., 2019) augments either the policy improvement or the policy evaluation step with a penalty term to constrain the policy within the support set of the behavior policy. For simplicity of illustration, we only consider the policy improvement step (Wu et al., 2019):

$$\max_{\pi_\theta} \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} [\mathbb{E}_{a'' \sim \pi_\theta(\cdot|s)} [Q_\psi(s, a'')] - \alpha \hat{D}(\pi_\theta(\cdot|s), \pi_\beta(\cdot|s))] \tag{3}$$

Here, $\hat{D}$ is a selected f-divergence (Csiszár, 1972) measure used for behavior regularization. Zooming into the regularization term, we get:

$$\sum_{s \in \mathcal{S}} (P_s \alpha) \hat{D}(\pi_\theta(\cdot|s), \pi_\beta(\cdot|s)) \tag{4}$$

In the equation above, $\mathcal{S}$ denotes the states that occur in a single sampled batch and P_s is the probability of state s in the sampled batch. The improvements that we propose are targeted towards the following two terms in Equation 4: (1) The divergence term $\hat{D}$ which for each fixed π_θ determines the impact actions have on the overall objective conditional on the state, and (2) the term $(P_s \alpha)$ which determines the impact the "divergence penalty" of each state will have on the overall objective function. Note that this term is dependent upon the probability of the occurrence of state in the sampled batch. For (1), we propose to use an analytical upper bound of KL divergence to reduce variance. For (2), we propose to decouple the impact of "divergence penalty" of each state from the probability of its occurrence. Detailed explanation of the two improvements and the rationale behind them follows in the next two sub-sections.

3.1 Regularization Method

Behavior regularization is used to constrain the learned policy with the support set of the behavior policy. In other words, it ensures that the "difference" between the probability distributions of the learned policy $\pi_\theta(\cdot|s)$ and behavior policy $\pi_\beta(\cdot|s)$ is small. The following divergence measures are most widely used in the community:

Kernel MMD Kernel Maximum Mean Discrepancy (MMD) (Gretton et al., 2007) was first introduced in (Kumar et al., 2019) to penalize the policy from diverging from the behavior policy:

$$\text{MMD}_k^2(\pi(\cdot|s), \pi_b(\cdot|s)) = \mathop{\mathbb{E}}_{x,x' \sim \pi(\cdot|s)}[K(x,x')] - 2 \mathop{\mathbb{E}}_{\substack{x \sim \pi(\cdot|s) \\ y \sim \pi_b(\cdot|s)}}[K(x,y)] + \mathop{\mathbb{E}}_{y,y' \sim \pi_b(\cdot|s)}[K(y,y')] \tag{5}$$

where K is a kernel function. Symmetric kernel functions such as Laplacian and Gaussian kernels are typically used (Gretton et al., 2007).

Wasserstein distance The p-Wasserstein distance between probability measures μ and ν on $\mathbb{R}^d$ is defined as:

$$W_p(\mu, \nu) = \inf_{X \sim \mu, Y \sim \nu} (\mathbb{E}||X - Y||^p)^{1/p}, \qquad p \geq 1 \tag{6}$$

The Wasserstein distances W_p are proper distances in that they are nonnegative, symmetric in X and Y, and satisfy the triangle inequality.

KL divergence For two probability distributions P and Q on some probability space χ, the KL-Divergence from Q to P is given as

$$\mathcal{D}_{\text{KL}}(P, Q) = \int_{x \sim \chi} P(x) \log \frac{P(x)}{Q(x)} dx \tag{7}$$

KL divergence is assymetric and both $\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s), \pi_b(\cdot|s))$ and $\mathcal{D}_{\text{KL}}(\pi_b(\cdot|s), \pi_\theta(\cdot|s))$ are valid. However, $\mathcal{D}_{\text{KL}}(\pi_b(\cdot|s), \pi_\theta(\cdot|s))$ is not suitable because it requires $\pi_\theta(a|s) \neq 0$ when $\pi_b(a|s) \neq 0$, for each a. It may assign $\pi_\theta(a|s) > 0$ when $\pi_b(a|s) = 0$, producing out of distribution actions. Therefore, $\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s), \pi_b(\cdot|s))$ is used for regularization.

3.1.1 Analytical KL divergence upper bound

All the existing behavior regularized methods estimate the divergence via samples (Kumar et al., 2019; Wu et al., 2019). While in theory it produces an unbiased estimator, it requires large samples to reduce the variance. Our experimental results (Figure 1a), as well as (Kumar et al., 2020) show that the performance deteriorates as the training proceeds longer.

To stabilize the performance, the key idea is to have a low variance estimator for KL divergence. We obtain this by deriving an upper bound on the KL divergence between the learned policy π_θ and the behavior policy π_β which can be computed analytically.

Assume that we learn π_β using a variational inference model (e.g. VAE (Kingma & Welling, 2014)) with latent variable Z. According to the variational lower bound, we obtain $\log \pi_\beta(a|s) \geq \mathbb{E}_{z\sim q(z)}[\log p(a|s,z)] - \mathcal{D}_{\text{KL}}(q(z)||p(z))$, where $q(z)$ is a variational distribution and $p(z)$ is the prior. Then, the KL divergence is bounded by:

$$\begin{aligned}\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s)||\pi_\beta(\cdot|s)) &= \mathbb{E}_{a\sim\pi_\theta}[\log \pi_\theta(a|s)] - \mathbb{E}_{a\sim\pi_\theta}[\log \pi_\beta(a|s)] \\ &\leq \mathbb{E}_{a\sim\pi_\theta}[\log \pi_\theta(a|s)] - \mathbb{E}_{a\sim\pi_\theta}[\mathbb{E}_{z\sim q(z)}[\log p(a|s,z)] - \mathcal{D}_{\text{KL}}(q(z)||p(z))] \\ &= \mathbb{E}_{a\sim\pi_\theta, z\sim q(z)}[\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s)||p(\cdot|s,z)) + \mathcal{D}_{\text{KL}}(q(z)||p(z))] \end{aligned} \tag{8}$$

Thus, if we choose $p(\cdot|s,z)$ to be a tractable distribution, Equation 8 can be estimated analytically. To increase the expressiveness, we choose $p(\cdot|s,z)$ to be mixture of Gaussian distributions (GMM). The KL divergence between a Gaussian distribution and a GMM is analytically intractable. Fortunately, a variational upper bound is available (Hershey & Olsen, 2007):

$$\mathcal{D}_{\text{KL}}(\mathcal{N}(\mu,\sigma)||\sum_{i=1}^{C}\omega_i\mathcal{N}(\mu_i,\sigma_i)) \leq \sum_{i=1}^{C}\omega_i\,\mathcal{D}_{\text{KL}}(\mathcal{N}(\mu,\sigma)||\mathcal{N}(\mu_i,\sigma_i)) \tag{9}$$

Please refer to (Hershey & Olsen, 2007) for more details on approximation of the KL divergence between Gaussian mixture models. Equation 8 and 9 provide a computationally efficient and low variance estimator of the KL divergence upper bound. Our experimental results in Section 5 indicate that the performance achieved using this estimator is much more stable than the performance achieved using samples.

3.2 State Dependent Lagrange Multiplier

From the term $(P_s\alpha)$ in Equation 4, one can infer that the "penalty budget" of divergence measure gets distributed among the states in amounts inversely proportional to the probability of the occurrence of the states in the sampled batch. If we assume the batches are sampled uniformly, this implies that the states which are more numerous in the offline dataset $\mathcal{D}$ i.e. the states that have been explored more thoroughly are restricted from deviating from the behavior policy. Whereas the less numerous state that have had limited exploration enjoy more freedom for deviation. This is undesirable as restricting highly explored states is overly conservative while allowing less explored states to deviate can lead to OOD actions.

To address this we add a state wise constraint for KL divergence as follows:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim\mathcal{D}}[\mathbb{E}_{a'\sim\pi_\theta(\cdot|s)}[Q_\psi(s,a')]] \;\text{ s.t. }\; \mathcal{D}_{KL}(\pi_\theta(\cdot|s),\pi_\beta(\cdot|s)) \leq \epsilon_{\text{KL}}, \forall s \quad \text{(policy improvement)}$$

This translates to assignment of state dependent Lagrange multipliers when solved using dual gradient descent:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim\mathcal{D}}[\mathbb{E}_{a'\sim\pi_\theta(\cdot|s)}[Q_\psi(s,a') - \alpha(s)\,\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s),\pi_\beta(\cdot|s))]] \tag{10}$$

Maintaining a Lagrange multiplier for each state is impractical for large datasets. In practice, we parameterize $\alpha(s)$ with a neural network.

3.3 Other Practical Enhancements

Entropy regularization The KL divergence is the sum of negative entropy of the learned policy plus the cross entropy between the learned policy and the behavior policy: $\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s),\pi_\beta(\cdot|s)) = -\mathcal{H}(\pi_\theta(\cdot|s)) + \mathcal{H}(\pi_\theta(\cdot|s),\pi_\beta(\cdot|s))$. When the learned policy distribution violates the KL constraints, the KL divergence between the policy distribution and the behavior distribution is decreased by the optimizer. This is equivalent to increasing the entropy of the learned policy and decreasing the cross entropy between the learned policy and the behavior policy. In soft actor-critic (Haarnoja et al., 2018a), the minimum entropy of the learned policy is enforced to encourage exploration. However, due to the absence of exploration, stochastic policy with large entropy will sample out-of-distribution actions when computing the target Q values in Equation 2. If such values are overestimated, the policy will exploit the erroneous Q values when performing the policy improvement in Equation 10 and lead to failure, which can't be corrected without more data. Thus, we maintain the maximum entropy of the learned policy using the technique proposed in (Haarnoja et al., 2018b).

Algorithm 1 BRAC+: Improved Behavior Regularized Actor Critic

Train the behavior policy π_β on the offline dataset $\mathcal{D} = \{(s_i, a_i, r_i, s_i^{'})\}_{i=1}^N$ via maximum likelihood estimation: $\pi_\beta = \arg\max_{\pi_\beta} \sum_{i=1}^N \log \pi_\beta(a_i|s_i)$
Train initial policy: $\pi_\theta = \arg\min_{\pi_\theta \in \Pi} \mathcal{D}_{\text{KL}}(\pi, \pi_\beta)$
Train initial Q network: $Q_\psi = \arg\min_\psi [(Q_\psi(s,a) - (r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_\theta} Q_{\psi'}(s', a')))]^2$
for $e = 1 : E$ **do**
  for $t = 1 : T$ **do**
    Update Q network: $\min_{Q_\psi} \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}}[(Q_\psi(s,a) - (r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_\theta} Q_{\psi'}(s', a')))^2]$
    Update the policy: $\max_{\pi_\theta} \mathbb{E}_{s \sim \mathcal{D}}[\mathbb{E}_{a' \sim \pi_\theta(\cdot|s)}[Q_\psi(s, a') - \alpha(s)\, \mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s), \pi_\beta(\cdot|s)) - \beta \mathcal{H}(\pi_\theta(\cdot|s)]]$, where $\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s), \pi_\beta(\cdot|s))$ is computed analytically using Equation 8
    Update α via dual gradient descent: $\alpha(s) \leftarrow \alpha(s) + \lambda_\alpha(\mathcal{D}_{\text{KL}}(\pi_\theta(\cdot|s), \pi_\beta(\cdot|s)) - \epsilon_{\text{KL}})$
    Update β via dual gradient descent: $\beta \leftarrow \beta + \lambda_\beta(\mathcal{H}(\pi_\theta(\cdot|s)) - \mathcal{H}_0)$
    Update the target network $\psi^{'} = \tau\psi + (1-\tau)\psi^{'}$
  end for
end for

Initialization If the dataset is collected using a narrow policy distribution in a high dimensional space (e.g. human demonstration), the constrained optimization problem using dual gradient descent finds it difficult to converge if random initialization is used for the policy network. To mitigate this issue, we start with a policy that has the minimum KL divergence with the behavior policy: $\pi_\theta = \arg\min_{\pi_\theta \in \Pi} \mathcal{D}_{\text{KL}}(\pi, \pi_\beta)$, where Π represents a family of policy types. In this work, we consider Π as Gaussian policies. Correspondingly, we initialize the Q network to Q^{π_θ}.

Policy Evaluation Masking In our experiments, we observe that no matter how high the Lagrange multipliers of behavior regularization terms are, there always exists a small number of states where the KL divergence between the policy distribution and the behavior distribution is much larger than the threshold. We hypothesize that this issue arises because: 1) the capacity of the neural network is limited, 2) there are always some states in the dataset which are not sufficiently explored (e.g. the behavior distribution density is a Dirac function for them.). This leads to out-of-distribution actions if the training is continued for sufficiently long time. To mitigate this problem, we apply zero masking to prevent the target Q values evaluated at the out-of-distribution actions from updating the Q network:

$$Q_\psi = \arg\min_\psi [(\mathbb{1}[\mathcal{D}_{\text{KL}}(\pi(\cdot|s'), \pi_\beta(\cdot|s'))] < \epsilon'_{\text{KL}})(Q_\psi(s,a) - (r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_\theta} Q_{\psi'}(s', a')))]^2 \tag{11}$$

where ϵ'_{KL} decides the hard constraints. In practice, we set $\epsilon'_{\text{KL}} = \epsilon_{\text{KL}} + \sigma_{\text{KL}}$, where σ_{KL} is the empirical standard deviation of the KL divergence between the learned policy and the behavior policy at all the states in the dataset.

4 Related Work

We briefly summarize prior works in offline RL and discuss their relationship with our approach. As discussed in Section 1, the fundamental challenge in learning from a static data is to avoid out-of-distribution actions (Levine et al., 2020). This requires solving two problems: 1) estimation of behavior policy, 2) quantification of out-of-distribution actions. BCQ (Fujimoto et al., 2018b), BEAR (Kumar et al., 2019) and BRAC (Wu et al., 2019) learn the behavior policy using a conditional VAE (Kingma & Welling, 2014). This suffers from epistemic uncertainty due to limited data. In contrast, we adopt the ideas from model-based RL Chua et al. (2018); Kurutach et al. (2018); Janner et al. (2019) to learn ensembles of the behavior policies and randomly select one to perform inference at test time. To avoid out-of-distribution actions, BCQ generates actions in the target values by perturbing the behavior policy. However, this is over-pessimistic in most of the cases. BRAC (Wu et al., 2019) constrains the policy using various sample-based f-divergence measures including MMD, Wasserstein distance and KL divergence with penalized policy improvement or policy evaluation. BEAR (Kumar et al., 2019) is an instance of BRAC with penalized policy improvement using

MMD (Gretton et al., 2007). Sample-based estimation is computationally expensive and suffers from high variance. In contrast, our method uses an analytical upper-bound of the KL divergence to constrain the distance between the learned policy and the behavior policy. It is both computationally efficient and has low variance. (Siegel et al., 2020) solves trust-region objective instead of using penalty. CQL (Kumar et al., 2020) avoids estimating the behavior policy by learning a conservative Q function that lower-bounds its true value. Hyperparameter search is another challenging problem in offline RL. (Lee et al., 2020) uses a gradient-based optimization of the hyperparameter using held-out data. MOPO (Yu et al., 2020) follows MBPO Janner et al. (2019) with additional reward penalty on unreliable model-generated transitions. MBOP (Argenson & Dulac-Arnold, 2020) learns the dynamics mode, the behavior policy and a truncated value function to perform online planning. (Kidambi et al., 2020) learns a surrogate MDP using the dataset, such that taking out-of-distribution actions transit to the terminal state. The out-of-distribution actions are detected using the agreement of model ensembles.

5 Experiments

Our experiments aim to answer the following questions: 1) How does the performance of our improvements compare with state-of-the-art model-free and model-based offline RL methods? 2) How does the use of analytical variational upper bound on KL divergence for regularization term compare with sampled KL? 3) How does the state-dependent Lagrange multiplier based regularization (state-wise regularizor) compare with fixed and global policy regularization? 4) What is the relationship between the KL divergence bound (ϵ_{KL}) and the performance of the learned policy? To answer these questions, we evaluate our methods on a subset of the D4RL (Fu et al., 2020) benchmark. We consider three locomotion tasks (hopper, walker2d, and halfcheetah) and four types of datasets: 1) random (rand): collect the interactions of a run of random policy for 1M steps to create the dataset, 2) medium (med): collect the interactions of a run of medium quality policy for 1M steps as the dataset, 3) medium-expert (med-exp): run a medium quality policy and an expert quality policy for 1M steps, respectively, and combine their interactions to create the dataset, 4) mixed (medium-replay): train a policy using SAC (Haarnoja et al., 2018a) until the performance of the learned policy exceeds a pre-determined threshold, and take the replay buffer as the dataset.

We compare against state-of-the-art model-free and model-based baselines, including BEAR (Kumar et al., 2019) that constrains the learned policy within the support of the behavior policy using sampled MMD, BRAC-p/v (Wu et al., 2019) that constrains the learned policy within the support of the behavior policy using various sample-based f-divergences to penalize either the policy improvement (p) or the policy evaluation (v), CQL($\mathcal{H}$) (Kumar et al., 2020) that learns a Q function that lower-bounds its true value. We also compare against model-based approaches including MOPO (Yu et al., 2020) that follows MBPO (Janner et al., 2019) with additional reward penalties and MBOP (Argenson & Dulac-Arnold, 2020) that learns an offline model to perform online planning.

5.1 Comparative Results

Performance on multi-modal datasets We first compare the performance on multi-modal datasets i.e. med-exp and mixed datasets. Results shown in Table 1 suggest that our method outperforms various model-free baselines on most of the multi-modal datasets, especially on hopper-mix and walker2d-mix by up to 1.5x$\sim$2x. BEAR and BRAC suffers from high variance due to out-of-distribution action estimations. The choice of parameters in CQL($\mathcal{H}$) makes it too conservative to achieve higher performance. The performance of our method on the two tasks matches with MOPO, which relies on the generalization of the learned dynamics model to provide additional data to learn better Q functions. This implies that the learned Q function in our method generalizes to these actions without any additional data.

Performance on single-modal dataset In general, the performance of our method on single-modal (rand and med) dataset matches with baseline methods as evident from Table 1. In two of the tasks (hopper-rand and hopper-med), our method outperforms the baselines whereas in the other four, our method performs worse. We observe that the performance is very sensitive to the choice of target policy entropy. We hypothesize that the behavior distribution is too narrow such that policies with large entropy easily samples out-of-distribution actions.

Table 1: Results for D4RL (Fu et al., 2020) datasets. Each number is the normalized score as proposed in (Fu et al., 2020) by running the policy obtained at the last epoch of the training for 100 episodes, averaged over 4 random seeds with standard deviation. Please refer to (Fu et al., 2020) for results on more baselines.

Task Name	Model-Free				Model-Based	
	BEAR	BRAC-p/v	CQL($\mathcal{H}$)	BRAC+ (Ours)	MOPO	MBOP
halfcheetah-rand	25.1	24.1/31.2	**35.4**	25.5 ± 2.4	31.9±2.8	6.3±4.0
walker2d-rand	**7.3**	-0.2/1.9	7.0	4.5± 2.5	**13.0±2.6**	8.1± 5.5
hopper-rand	11.4	11.0/12.2	10.8	**13.9±0.4**	**13.3±1.6**	10.8± 0.3
halfcheetah-med	41.7	43.8/**46.3**	44.4	43.6 ± 4.2	40.2± 2.7	**44.6±0.8**
walker2d-med	59.1	77.5/81.1	**79.2**	65.5± 3.2	14.0±10.1	41.0 ± 29.4
hopper-med	52.1	32.7/31.1	**58.0**	**59.7 ± 3.6**	26.5±3.7	48.8 ± 26.8
halfcheetah-med-exp	53.4	44.2/41.9	**62.4**	46.5± 6.4	57.9± 24.8	**105.9± 17.8**
walker2d-med-exp	40.1	76.9/81.6	**98.7**	90.6± 15.6	55.0± 19.1	70.2 ± 36.2
hopper-med-exp	96.3	1.9/0.8	**111.0**	**112.0± 0.1**	51.7±42.9	55.1 ± 44.3
halfcheetah-mixed	38.6	45.4/**47.7**	46.2	45.5±0.4	**54.0±2.6**	42.3 ± 0.9
walker2d-mixed	19.2	-0.3/0.9	26.7	**41.5± 4.6**	**42.7±8.3**	9.7 ± 5.3
hopper-mixed	33.7	0.6/0.6	48.6	**92.82 ± 3.4**	**92.5±6.3**	12.4 ± 5.8

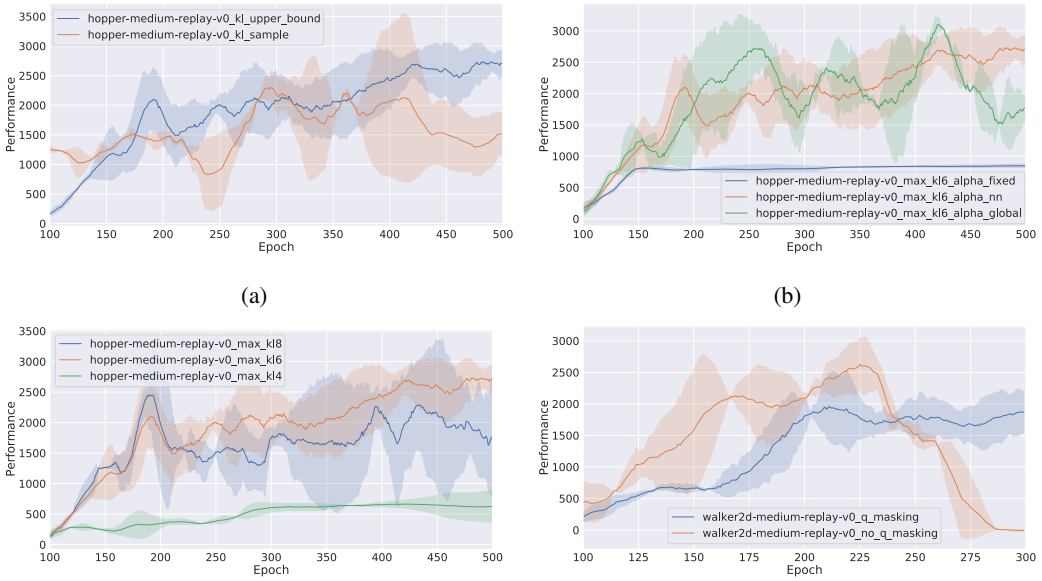

Figure 1: Figures of ablation study. Each setting is repeated for 4 random seeds. The curve is the mean and the shaded area is the standard deviation. The number of samples to estimate the KL divergence is 10. The name of the legend indicates the task and the parameter choices. For example, "hopper-medium-replay-v0_max_kl6_alpha_nn" means that we evaluate our method on hopper-medium-replay (mixed) task with $\epsilon_{\text{KL}} = 6$ and with neural network alpha.

5.2 Ablation Study

To answer question (2), (3) and (4), we conduct a thorough ablation study on BRAC+.

Sampled KL vs. analytical upper bound KL We show the performance of the hopper-mixed (identical to the hopper-medium-replay in (Fu et al., 2020)) task by using sampled KL divergence and the analytical upper bound KL divergence in Figure 1a. The performance of the agent trained

using analytical upper bound KL divergence is much more stable than the performance of the agent trained using sampled KL divergence.

State-wise vs. global regularization We show the performance of the hopper-mixed task by using fixed (alpha_fixed), global (alpha_global) and state-wise regularization (alpha_nn) in Figure 1b. The performance of fixed α does not improve much as it is too restrictive for some states and too loose for others. The performance of a globally updated α fluctuates. This is because it's hard to find an α that works for all the sampled batches unless we use the whole batch to perform policy improvement. However, this is impractical for large datasets. On the contrary, the performance learned with state-wise regularization (legend - alpha_nn in the figure) converges.

Performance vs. KL divergence threshold Selecting the right KL divergence threshold is challenging. If we start with $\epsilon_{KL} = \min_{\pi_\theta \in \Pi} \mathcal{D}_{\text{KL}}(\pi, \pi_\beta)$ and gradually increase the threshold, the performance first increases and then decreases. This is because increasing the threshold allows exploration in the multi-modal datasets while simultaneously increasing the risk of sampling out out-of-distribution actions. We show the performance of hopper-mixed task by setting the KL divergence threshold to 4, 6, and 8, respectively in Figure 1c. It's noted that the performance by setting $\epsilon_{\text{KL}} = 4$ is too restrictive. Setting the $\epsilon_{\text{KL}} = 8$ allows too much out-of-distribution data that hinders the performance as the training progresses.

Stability analysis Even with policy regularization, the out-of-distribution actions cannot be fully avoided due to the representation capacity of the policy network. It is important to prevent the target Q values evaluated at those actions from updating the Q network. Figure 1d shows the performance on walker2d-mix with and without policy evaluation masking (q-masking). The performance of training without q-masking increases rapidly at the beginning due to aggressive Q updates. However, as the training stabilizes, the performance drops significantly. In contrast, the performance of training with q-masking steadily increases and converges.

6 Conclusion

In this paper, we improved the behavior regularized offline reinforcement learning by proposing a low-variance upper bound of the KL divergence estimator to reduce variance, state-dependent Lagrange multiplier to allow more freedom of deviation to high probability states while restricting low probability states, entropy regularization, better initialization and policy evaluation masking. Our experimental results on challenging benchmarks illustrate the benefits of our improvements. Future work includes conducting experiments on more tasks and have better hyper-parameter tuning procedure.

References

Abien Fred Agarap. Deep learning using rectified linear units (relu). *ArXiv*, abs/1803.08375, 2018.

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. *ArXiv*, abs/2008.05556, 2020.

Sungwoon Choi, Heonseok Ha, Uiwon Hwang, Chanju Kim, Jung-Woo Ha, and S. Yoon. Reinforcement learning based recommender system using biclustering technique. *ArXiv*, abs/1801.05532, 2018.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *CoRR*, abs/1805.12114, 2018. URL http://arxiv.org/abs/1805.12114.

I. Csiszár. A class of measures of informativity of observation channels. *Periodica Mathematica Hungarica*, 2(1):191–213, 1972. doi: 10.1007/BF02018661. URL https://doi.org/10.1007/BF02018661.

Justin Fu, Aviral Kumar, Ofir Nachum, G. Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *ArXiv*, abs/2004.07219, 2020.

Scott Fujimoto, H. V. Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *ArXiv*, abs/1802.09477, 2018a.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *CoRR*, abs/1812.02900, 2018b. URL http://arxiv.org/abs/1812.02900.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Scholkopf, and Alexander J. Smola. A kernel approach to comparing distributions. In *AAAI*, pp. 1637–1641, 2007. URL http://www.aaai.org/Library/AAAI/2007/aaai07-262.php.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018a. URL http://arxiv.org/abs/1801.01290.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018b. URL http://arxiv.org/abs/1812.05905.

J. R. Hershey and P. A. Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, pp. IV–317–IV–320, 2007.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *CoRR*, abs/1906.08253, 2019. URL http://arxiv.org/abs/1906.08253.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel : Model-based offline reinforcement learning, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Diederik P. Kingma and M. Welling. An introduction to variational autoencoders. *Found. Trends Mach. Learn.*, 12:307–392, 2019.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *CoRR*, abs/1906.00949, 2019. URL http://arxiv.org/abs/1906.00949.

Aviral Kumar, Aurick Zhou, G. Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *ArXiv*, abs/2006.04779, 2020.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *CoRR*, abs/1802.10592, 2018. URL http://arxiv.org/abs/1802.10592.

Byung-Jun Lee, Jongmin Lee, Peter Vrancx, DongHo Kim, and Kee-Eung Kim. Batch reinforcement learning with hyperparameter gradients. 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv*, abs/2005.01643, 2020.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2016.

Long-Ji Lin. *Reinforcement Learning for Robots Using Neural Networks*. PhD thesis, USA, 1992.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL http://arxiv.org/abs/1312.5602.

Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, T. Lampe, Roland Hafner, and Martin A. Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *ArXiv*, abs/2002.08396, 2020.

David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojtek Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.

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

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization, 2020.

Chi Zhang, Sanmukh R. Kuppannagari, Rajgopal Kannan, and Viktor K. Prasanna. Building hvac scheduling using reinforcement learning via neural network based model approximation. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '19, pp. 287–296, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450370059. doi: 10.1145/3360322.3360861. URL https://doi.org/10.1145/3360322.3360861.