No Imitation for Me: Advice-based Offline Reinforcement Learning for Aerial Control

Karush Suri, Mhd Ali Alomrani, Reza Moravej
University of Toronto, Canada
{karush.suri,mohammad.alomrani,mreza.moravej}@mail.utoronto.ca

Proof Of Concept: Colab Notebook (3 minute runtime)

Abstract: Recent success in offline Reinforcement Learning (RL) is highlighted by its adaptability to novel scenarios. One of the key reasons behind this success is the readily available nature of behavior transitions. Practical applications, on the other hand, consist of behavior policy as a sequence of demonstrations rather than a dataset of transitions. This allows one to rethink behavior transitions through the lens of imitation. We steer research towards this direction by answering the central question of *how can demonstrations be utilized for behavior transitions?* To this end, we formulate advice as a combination of expert demonstrations and behavior transitions within the Conservative Q-Learning (CQL) framework. ADVice-based Conservative Q-Learning (ADV-CQL) maximizes advice as a variational lower bound, resulting in consistent behaviors on a suite of challenging aerial control tasks. Additionally, ADV-CQL agents demonstrate suitable offline transfer to novel scenarios indicating reduced overfitting towards prior behaviors.

Keywords: Advice, Offline, Behaviors.

1 Introduction

Consider the scenario wherein a human learns to drive a car. The driver observes a teacher driving the car. This involves paying attention to crucial insights of controlling the vehicle such as steering while making a turn, accelerating during a green light and monitoring mirrors during brakes. Increasing amount of observations by the driver result in learning finer details of the task which are not available *apriori*. For instance, the driver may never learn to make a U-turn if the teacher never encountered a U-turn crossing.

Offline RL [1] addresses this intuitive gap in learning by equipping an agent (the *driver* in above example) with the ability to stitch together portions of observations by making use of a dataset of transitions. For instance, the driver may learn to make a U-turn on its own if it observes the teacher making sharp turns and slowing down the vehicle at intersections. Adoption of transitions in the offline setting allows the agent to efficaciously tackle distributional shift between the teacher's policy and the agent's policy. To this end, it is reasonable to ask the question *how would the agent learn to make a U-turn if it never saw the teacher make sharp turns or slow down the vehicle?* More specifically, how does the agent stitch together portions of transitions with limited data?

Various scenarios [2] make data collection imperative in the face of uncertainty. Lack of optimal behavior transitions observed by an offline RL agent may cripple its policy and result in sub-optimal convergence. This allows one to rethink offline RL as an abstraction of *behavior transitions* and *learning* problems. In the first stage, the agent desires a suitable initialization point (a behavior policy or static dataset of transitions) which would serve as a guiding principle for agent's policy. The second stage comprises of learning optimal behaviors based on initialization point. The direct dependence of offline mechanism on behavior transitions highlights its pivotal role in the learning pipeline.

Modern offline RL [1, 3, 4, 5] methods resort to a black-box dataset of transitions as the initialization point. This often limits optimal behavior at the cost of data collection (as observed in *driving* exam-

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

ple). A suitable alternative to address this limitation is by utilizing demonstrations as initialization point for the agent's policy. Similar to static transitions in offline RL, expert demonstrations provide a guiding mechanism for learner's policy in the IL setup.

Our work aims to answer the central question of *how can demonstrations be utilized for behavior transitions?* Towards this solution, we formulate advice as a combination of expert demonstrations and behavior transitions within the Conservative Q-Learning (CQL) [3] framework. The agent, pre-initialized with behavior transitions, queries a trained online RL expert to seek advice in the form of raw actions. Query samples from the expert act as a *soft* generalization of the *hard* dataset transitions provisioned in the CQL framework. ADVice-based Conservative Q-Learning (ADV-CQL) agents are trained to maximize advice as a variational lower bound, resulting in consistent behaviors on a suite of challenging aerial control tasks. Additionally, provision of an explicit advice demonstrates suitable offline transfer to novel scenarios which is found akin to reduced overfitting towards the behavior policy.

2 Background

Offline Reinforcement Learning: We review the offline RL setup wherein an agent interacts with the environment in order to transition to new states and observe rewards by following a sequence of actions. The problem is modeled as a finite-horizon Markov Decision Process(MDP) [6] defined by the tuple $(\mathcal{S}, \mathcal{A}, r, P, \gamma)$ where the state space is denoted by $\mathcal{S}$ and action space by $\mathcal{A}$, r presents the reward observed by agent such that $r : \mathcal{S} \times \mathcal{A} \to [r_{min}, r_{max}]$, $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$ presents the unknown transition model consisting of the transition probability to the next state $s_{t+1} \in \mathcal{S}$ given the current state $s_t \in \mathcal{S}$ and action $a_t \in \mathcal{A}$ at time step t and γ is the discount factor. We consider the agent's policy $\pi_\theta(a_t|s_t)$ as a function of its parameters θ and a behavior policy $\pi_b(a_t|s_t)$ with the discounted marginal state distribution $d^{\pi_b}(s_t)$. A dataset $\mathcal{D} \sim d^{\pi_b}(s_t)\pi_b(a_t|s_t)$ consists of finite (s_t, a_t, s_{t+1}) transitions. Offline RL defines the agent's objective to maximize the expected discounted reward expressed in Equation 1.

$$\mathbb{E}_{s_t, a_t \sim \mathcal{D}}[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)] \tag{1}$$

Soft Actor-Critic: SAC [7] is an off-policy RL algorithm which defines an entropy-based objective presented in Equation 2.

$$J(\pi_\theta) = \sum_{t=0}^{T} \gamma^t [r(s_t, a_t) + \lambda \mathcal{H}(\pi_\theta(\cdot|s_t))] \tag{2}$$

Here λ is the temperature coefficient and $\mathcal{H}(\pi_\theta(\cdot|s_t))$ is the entropy exhibited by the policy $\pi(\cdot|s_t)$ in s_t. For a fixed policy, the soft Q-value function can be computed iteratively, starting from any function $Q : \mathcal{S} \times \mathcal{A}$ and repeatedly applying a modified Bellman backup operator $\hat{B}^\pi$ given by Equation 3. Here $V(s_t)$ is the soft state value function as expressed in Equation 4.

$$\hat{B}^\pi Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim P}[V(s_{t+1})] \tag{3}$$

$$V(s_t) = \mathbb{E}_{a_t \sim \pi}[Q(s_t, a_t) - \log(\pi(a_t|s_t))] \tag{4}$$

We consider a parameterized state value function $V_\psi(s_t)$, a soft Q-function $Q_\phi(s_t, a_t)$ and a policy $\pi_\theta(a_t|s_t)$ which can be represented with neural networks with ψ, ϕ and θ being the parameters of these networks.

Conservative Q-Learning: Conservative Q-learning (CQL) is an offline RL algorithm which addresses the accumulating bootstrapping error due to action distribution shift by obtaining a lower bound on the Q-values. The empirical Q-values of agent's policy, $\hat{Q}^k$, are underestimated at each policy iteration k which results in an expected lower bound. The CQL objective $CQL(\mathcal{R})$, with a suitable choice of regularizer $\mathcal{R}(\mu)$, is expressed in Equation 5.

$$CQL(\mathcal{R}) = \min_Q \max_\mu \alpha(\mathbb{E}_{s_t \sim \mathcal{D}}[\mathbb{E}_{a_t \sim \mu(a_t|s_t)}[Q] - \mathbb{E}_{a_t \sim \pi_b(a_t|s_t)}[Q]]) + \frac{1}{2}\mathbb{E}_{s_t, a_t, s_{t+1} \sim \mathcal{D}}[(Q - \hat{B}^\pi \hat{Q}^k)^2] + \mathcal{R}(\mu) \tag{5}$$

Practical implementation of $CQL(\mathcal{R})$ utilizes soft-maximum over Q-values resulting in $CQL(\mathcal{H})$ objective expressed in Equation 6.

$$CQL(\mathcal{H}) = \arg\min_{Q} \mathbb{E}_{s_t \sim \mathcal{D}}[\log \sum_{a_t} \exp Q(s_t, a_t) - \mathbb{E}_{a_t \sim \pi_b(a_t|s_t)}[Q]] + \frac{1}{2}\mathbb{E}_{s_t, a_t, s_{t+1} \sim \mathcal{D}}[(Q - \hat{B}^{\pi}\hat{Q}^k)^2] \quad (6)$$

3 Related Work

Offline Reinforcement Learning: Recent advances in offline reinforcement learning offer an appealing alternative to online RL by utilizing large, previously-collected datasets for learning with minimal online interactions. Unfortunately, this approach suffers from distributional shift [8] during training as a result of the Q-function trained on trajectories solely from the dataset. Consequently, the policy π, which is trained to maximize the Q-values, will be biased to out-of-distribution (OOD) actions with erroneous Q-values. Several works tackle this problem by forcing the learned policy to steer away from OOD actions by staying close to the behavior policy π_b [9]. This is typically done by minimizing the KL-divergence or Wasserstein distance to the behavior policy and penalizing large value estimates. In contrast, some methods require training an estimate of the behavior policy to query actions on unseen states [8, 10]. Our work is parallel to aforementioned approaches.

Conservative Q Learning: Kumar et al. [3] propose the CQL framework which learns a *conservative* value estimate from offline transitions. Theoretical analysis of CQL highlights that the learned Q-value estimate is a lower bound on the expected Q-value under the agent's policy. An expected lower bound is akin to conservative behavior which addresses overestimation in value estimates [11]. Additionally, CQL is compatible with online RL algorithms and outperforms other offline RL and behavior cloning methods on a number of complex tasks.

Imitation Learning: In real-world automated navigation scenarios, the agent does not have access to behavior transitions. This makes training of agent using reinforcement learning difficult. To this end, sophisticated methods which learn a policy from visual input and sensors have become popular for such applications. Prior approaches formulate the problem as a supervised learning setting, wherein neural networks are trained on datasets consisting of images recorded from a human driver [12, 13]. The model is then trained to output a steering direction given an image. Ross et al. [2] suggest that a key problem with this approach is the distribution shift in the dataset. It is very likely for the vehicle to encounter an unseen dataset during test time. To alleviate this issue, the work introduces DAgger, which iteratively collects a dataset under the current policy and trains the next policy under the aggregate of all collected datasets. Dagger, being one of the seminal works in accumulation of behavior transitions, demonstrates the necessity for optimal data collection and is orthogonal to our work.

4 A Motivating Example: The Lineworld MDP

In order to highlight the necessity for behavior initialization, we turn our attention to a small toy task. The implementation is available as a Colab Notebook which can be executed within 3 minutes online without downloading. We consider a deterministic counterpart of the Lineworld MDP from Kumar et. al. [14] presented in Figure 1 (top). The agent starts at a start location S and is tasked to reach the goal location G by continuously moving right. For each step in the right direction,

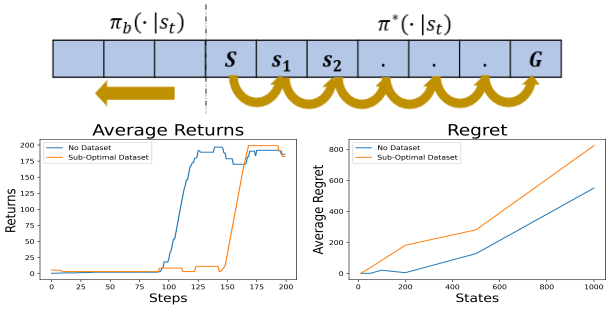

Figure 1: (top) The Lineworld MDP, (bottom-left) Average Returns over 200 steps, (bottom-right) Average Regret over number of states

the agent observes a reward of $+1$. For each step of left action the agent observes a -1 reward resulting in the termination of the episode. The total number of states in the Lineworld MDP correspond to the number of steps between the start and goal states. We consider two agents, namely *no dataset* (online) and *sub-optimal dataset* (offline) agents. The *no dataset* agent does not have a preinitialized dataset of transitions and it collects its own data from the MDP. The *sub-optimal dataset* agent is initialized with a dataset of sub-optimal transitions wherein the behavioral policy always chooses to go left. In departure from the original framework, we allow the *sub-optimal dataset* agent to collect data at selective interval after first 20 steps.

Figure 1 (bottom-left) presents the comparison of average returns achieved by the *no dataset* and *sub-optimal dataset* agents in the Lineworld MDP having 200 states. The online agent, by virtue of data collection, learns from transitions observed in the MDP. As a result of this, the online agent depicts sample-efficient learning in the number of environment interactions. On the other hand, the offline agent learns slowly due to the sub-optimal preferences of behavioral policy. Once the agent starts collecting data, it updates its estimates based on observed transitions and retrieves optimal behavior. Provision of a sub-optimal dataset, in comparison to the absence of a dataset, hinders data-efficient learning by steering the agent away from the path to optimality. While the agent observes ample amount of behavior transitions from the dataset, these fail to motivate learning and act as a procedural bottleneck.

We further study the scalability of problem by varying number of states in the Lineworld MDP. Figure 1 (bottom-right) presents the variation of average regret with the number of states in the MDP. In the case of both agents, regret scales sub-linearly and increases with the number of states. However, the offline agent observes a significantly faster increase in comparison to the online agent. Additionally, the gap between the regrets of offline and online agents widens with the former accumulating more regret over larger state spaces. Provision of a sub-optimal dataset not only hinders data-efficiency, but it also restricts the scalability of agents towards large number of states.

Availability of a sub-optimal dataset, in general, can do more harm than the absence of behavioral data. The above experiments demonstrate that having a dataset does not necessarily motivate efficient learning. Moreover, the learning of an agent primarily depends on the quality of transitions observed in the dataset. Thus, behavior initialization requires attention from a critical standpoint.

5 Advice-based CQL

5.1 Learning from Expert Demonstrations

We first train an agent (teacher) in the online setting with no access any privileged information. In the second stage, we train a sensorimotor agent (student) using the first agent as a *behavior policy*. The sensorimotor agent does not cheat by accessing the direct observation of the environment but can only depend on the visual sensors (cameras) and "advice" from the teacher (first agent). Our approach is similar to the one presented by Learning by Cheating [15], the pivotal difference is that the first agent is used to collect a training dataset for an offline RL agent, rather than a supervised learning agent. Moreover, the student agent seeks behavior advice from the teacher as part of the reward signal. Offline RL utilizes large, previously-collected datasets [4, 5] for sequential decision-making problems to train policies that can reason over temporal horizons. Modern offline methods have been shown to outperform simple behavior cloning on a number of real-world datasets and can often learn better policies than present in the data. Hence, leveraging a teacher agent in offline RL can help alleviate the difficulties that come along with offline data collection and provide more faster learning capabilities compared to behavior cloning due to teacher "advice".

5.2 Maximum Likelihood Advice

We assume the student has access to the teacher policy π_b, and can query it anytime during training. The teacher provides advice to the student policy π_θ through the mutual information measure $\mathbb{MI}(\pi_\theta, \pi_b)$ defined by:

$$\mathbb{MI}(\pi_\theta, \pi_b) = \int p(\pi_\theta, \pi_b) \log \frac{p(\pi_\theta, \pi_{exp})}{p(\pi_\theta)p(\pi_b)} d\pi_\theta d\pi_b$$

We aim to maximize this "intrinsic" reward during training to bring the student policy closer to the teacher. It is important to note that ultimate measure of performance we care about improving is the

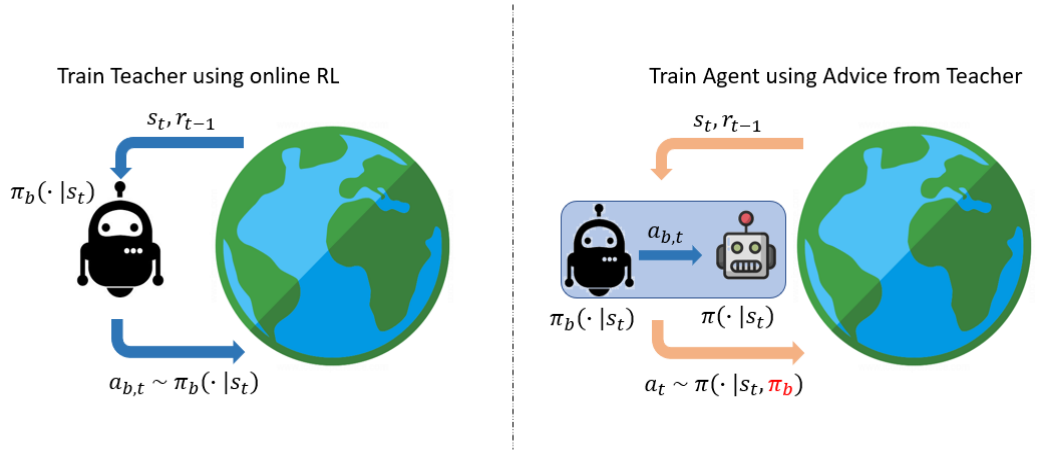

Figure 2: Offline RL through the lens of behavior transitions. (Left) A teacher agent exploits privileged knowledge to learn a behavioral policy π_b. (Right) The offline agent learns with unprivileged knowledge and teacher's policy π_b as behavior advice.

value of the extrinsic rewards achieved by the agent; the intrinsic reward serves only to motivate the agent to learn desired behavior. Unfortunately, computing the MI is challenging in large continuous action spaces, as we do not have access to the underlying distributions, making sample-based estimators brittle. To overcome these difficulties, we employ variational lower bounds combined with deep learning to enable a differentiable and tractable estimation of the MI. That is, we maximize the following lower bound instead,

$$\mathbb{MI}(\pi_\theta, \pi_b) \geq \mathbb{E}_{p(\pi_\theta, \pi_b)}[\log q(\pi_\theta|\pi_b)] + \mathcal{H}(p(\pi_\theta)) \tag{7}$$

where $q(\pi_\theta|\pi_b)$ is a differentiable approximation [1] of the posterior $q(\pi_\theta|\pi_b)$. See Appendix A for the derivation. We train the agent to maximize the sum of the intrinsic reward r_{int} and extrinsic reward r_{ext} weighted by a hyper-parameter α:

$$r = r_{ext} + \alpha r_{int} \tag{8}$$

In addition to utilizing advice, we found the training to be more stable by periodically collecting more demonstrations from the teacher policy during training. This ensures that we do not overfit to the current dataset and have a consistent and safe policy improvement during training. The full training procedure can be found in 1.

Algorithm 1 Training Stages

Initialize $D \leftarrow \phi$
Initialize expert policy π_b
Initialize K
Initialize offline student policy π_θ
Train expert policy π_b online using SAC to maximize Equation 2.
$D \leftarrow$ Sample T-step trajectories using trained policy π_b
for $i = 0$ **to** N **do**
  Update parameters θ to minimize CQL objective in Equation 6 with reward in Equation 8
  if i % K = 0 **then**
    $D \leftarrow$ aggregate dataset with more demonstrations from teacher policy π_b
  end if
end for

[1] via a neural network

6 Experiments

6.1 Advice-based Learning

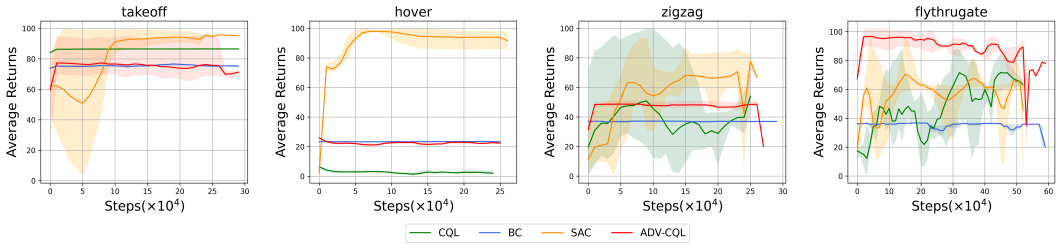

Figure 3: Task rewards learned for Drone control scenarios. Returns are averaged over 4 random runs.

We evaluate Advice-based CQL on 4 different tasks in the Gym PyBullet Drones [16] environment. Namely, learning to takeoff vertically, hover at a particular position, fly in a zig-zag pattern, and fly through a gate. See Appendix B for task details. At each timestep, agents control the RPMs of the 4 motors of a quadcopter.

As we can see in Figure 3, advice-based CQL performs competitively compared to vanilla CQL and SAC, while behavior cloning performs poorly on almost all tasks. In the flythrugate task, the performance of ADV-CQL is substantially better than all baselines. Moreover, our method displays much more consistent results across different runs than CQL and SAC. This demonstrates the effectiveness of advice in high-dimensional and stochastic environments.

6.2 Ablation Study

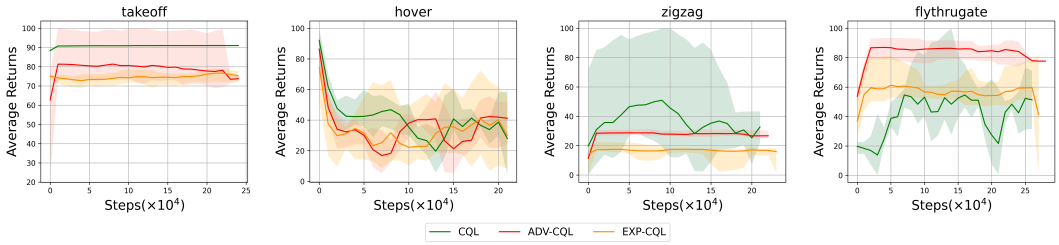

Figure 4: CQL ablations observed on task rewards for Drone control scenarios. Returns are averaged over 4 random runs.

In the ablation study from Figure 4, we evaluate the importance of advice by comparing our method to vanilla CQL and EXP-CQL, where the agent is trained on a dataset of expert demonstrations. Our results show that ADV-CQL outperforms EXP-CQL on almost all tasks, further demonstrating the advantage of *soft* advice over a *hard* dataset which can act as a bottleneck during training in achieving optimal behavior. In the flythrugate task, we observe that both baselines do learn to fly closer to the poles of the gate but fail to pass through the opening. ADV-CQL, on the other hand, learns to quickly pass through the gate without making contact with the poles.

We notice some inconsistencies in the results. For example, ADV-CQL performs worse than both CQL and SAC on the takeoff task. Another example is the zigzag task, wherein ADV-CQL performs worse than CQL while SAC demonstrated optimal performance. These cases indicate that not only the advice provided by SAC is not utilized by ADV-CQL, but it has made it perform worse than CQL. The central question to ask is *why do agents not exploit advice consistently?*

We suspect that the problem lies in collecting too narrow of a dataset using the SAC behaviour policy. Transitions collected from online SAC agent may be constrained to specific behavior modes which prevent effective exploration of ADV-CQL agents towards unseen states. The error may propagate and cause ADV-CQL agents to fall short midway during their ascent. We further solidify this claim by highlighting similar performance drops in the case of EXP-CQL agents. To address this problem, the dataset is repeatedly aggregated every 10,000 steps with 10 trajectories from the behaviour policy. This leads to consistent improvements in most tasks due to fast and optimal data collection. However, this enforces the agents to become data-dependent which motivates additional data collection.

6.3 Offline Transfer

In the online setting, the behavior model must be updated on-the-fly to track incoming data. In the offline setting behavior policy is only trained once. While offline RL algorithms with constraints demonstrate consistent behaviors , their success is largely associated with adaptibility to novel scenarios [17]. In light of adaptibility, we ask the question of *how does the generalization of ADV-CQL compare to CQL on unseen scenarios?*

We answer this question by constructing a novel task called *hover_downfall*. The agent must hover above a given location while it is faces obstacles (a rain of rubber ducks) falling downwards. As shown in Figure 5, additional fine-tuning hinders the CQL agent to obtain suitable behaviors. The CQL policy follows a similar behavior as the original behavior policy and overfits with a large policy loss. ADV-CQL, on the other hand, suitably adapts to the rain of obstacles by utilizing a more generalizable transfer policy. The policy for ADV-CQL does not overfit to prior behaviors and hence, obtains a lower policy loss across its random runs. Another way to interpret this finding is in light of constrained optimization. The behavior policy is unable to model new data well which results in overly-conservative behaviors of CQL agents [17]. ADV-CQL leverages the continued stream of advice from the online teacher and hence, does not suffer from over-conservatism.

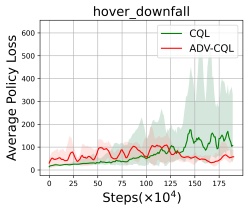

Figure 5: Average policy loss for CQL and ADV-CQL in *hover_downfall*. ADV-CQL presents improved generalization resulting in robust behaviors.

7 Discussion

We presented ADV-CQL, an offline RL algorithm which learns behaviors by maximizing advice conditioned on the behaviour policy provided by an online SAC agent. ADV-CQL performs competitive to SAC and CQL on aerial control tasks. We further show the advantage of *soft* advice over a *hard* dataset by achieving competitive performance compared to CQL agents with a dataset of expert demonstrations. ADV-CQL depicts generalization to novel scenarios and addresses over-conservatism as a result of reduced overfitting towards the behavior policy.

Provision of advice as an information-theoretic objective in offline RL provides several directions for future work. We highlight 3 potential avenues for future development. (1) As opposed to low-dimensional propioceptive information as advice, one can utilize high-dimensional visual information (such as waypoints and depth maps) for guiding agent behaviors. This would facilitate the extension of advice-based learning to high-dimensional state spaces such as raw pixels. (2) From a theoretical perspective, a tighter upper bound on the CQL objective would facilitate in addressing over-conservatism. Our theoretical analysis (deferred to Appendix A) is a step towards this direction. (3) Lastly, an accurate sampling scheme for the CQL objective could potentially provide consistent unimodal behavior distributions which are otherwise to approximate non-trivial due to soft Q value approximations in the CQL objective.

References

[1] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[2] S. Ross, G. J. Gordon, and J. A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning, 2011.

[3] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning.

[4] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg. Conservative safety critics for exploration. *arXiv preprint arXiv:2010.14497*, 2020.

[5] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

[6] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. 2018.

[7] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018.

[8] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. 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, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[9] N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, A. Lapedriza, N. Jones, S. Gu, and R. Picard. Way off-policy batch deep reinforcement learning of human preferences in dialog, 2020.

[10] Y. Wu, G. Tucker, and O. Nachum. Behavior regularized offline reinforcement learning, 2020.

[11] P. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence policy improvement. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2380–2388, Lille, France, 07–09 Jul 2015. PMLR.

[12] D. A. Pomerleau. *ALVINN: An Autonomous Land Vehicle in a Neural Network*, page 305–313. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989. ISBN 1558600159.

[13] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars, 2016.

[14] A. Kumar, J. Fu, G. Tucker, and S. Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

[15] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating. In *Proceedings of the Conference on Robot Learning*, pages 66–75, 2020.

[16] E. Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, SIGGRAPH '15, 2015.

[17] A. Nair, M. Dalal, A. Gupta, and S. Levine. Accelerating online reinforcement learning with offline datasets, 2020.

[18] D. Böhning. Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1):197–200, March 1992.

[19] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok, and A. P. Schöllig. Learning to fly: a pybullet gym environment to learn the control of multiple nano-quadcopters, 2020.

[20] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving sample efficiency in model-free reinforcement learning from images. *arXiv preprint arXiv:1910.01741*, 2019.

A Proofs and Derivations

A.1 Variational Lower Bound

Here we will derive the lower bound on Mutual Information between agent's and expert's policies $\mathbb{MI}(\pi, \pi_{exp})$. We begin by expanding $\mathbb{MI}(\pi, \pi_{exp})$,

$$\begin{aligned}
\mathbb{MI}(\pi, \pi_{exp}) &= \int p(\pi, \pi_{exp}) \log \frac{p(\pi, \pi_{exp})}{p(\pi)p(\pi_{exp})} d\pi d\pi_{exp} \\
&= \int p(\pi, \pi_{exp}) \log \frac{p(\pi|\pi_{exp})p(\pi_{exp})}{p(\pi)p(\pi_{exp})} d\pi d\pi_{exp} \\
&= \int p(\pi, \pi_{exp}) \log \frac{p(\pi|\pi_{exp})}{p(\pi)} d\pi d\pi_{exp}
\end{aligned}$$

We seek a tractable approximation $q(\pi|\pi_{exp})$ to the posterior $p(\pi|\pi_{exp})$,

$$\begin{aligned}
&= \int p(\pi, \pi_{exp})(\log \frac{q(\pi|\pi_{exp})}{p(\pi)} + \log \frac{p(\pi|\pi_{exp})}{q(\pi|\pi_{exp})}) d\pi d\pi_{exp} \\
&= \int p(\pi, \pi_{exp}) \log \frac{q(\pi|\pi_{exp})}{p(\pi)} d\pi d\pi_{exp} + \int p(\pi, \pi_{exp}) \log \frac{p(\pi|\pi_{exp})}{q(\pi|\pi_{exp})} d\pi d\pi_{exp} \\
&= \int p(\pi, \pi_{exp}) \log \frac{q(\pi|\pi_{exp})}{p(\pi)} d\pi d\pi_{exp} + \int p(\pi|\pi_{exp})p(\pi_{exp}) \log \frac{p(\pi|\pi_{exp})}{q(\pi|\pi_{exp})} d\pi d\pi_{exp}
\end{aligned}$$

The second term simplifies to $\mathbb{E}_{\pi_{exp}}[\mathbb{KL}(p(\pi|\pi_{exp})||q(\pi|\pi_{exp}))]$ which must be non-negative,

$$\geq \int p(\pi, \pi_{exp}) \log q(\pi|\pi_{exp}) d\pi d\pi_{exp} - \int p(\pi|\pi_{exp})p(\pi_{exp}) \log p(\pi) d\pi d\pi_{exp}$$

Marginalizing the second term over π_{exp} yields the lower bound,

$$= \mathbb{E}_{p(\pi, \pi_{exp})}[\log q(\pi|\pi_{exp})] + \mathcal{H}(p(\pi))$$

A.2 Tighter Bound on CQL($\mathcal{H}$) (Tutorial-Style)

This section derives a tighter bound on the CQL($\mathcal{H}$) variant of CQL. Prior to the complete derivation, we will show that the CQL($\mathcal{H}$) variant may not always underestimate the true Q values.

Underestimation in CQL: Here we present the proof of underestimation in CQL Q values as derived in [3]. Note that the original CQL objective, in the absence of regularization, is given by Equation 9.

$$\arg\min_{Q} \alpha \mathbb{E}_{\mu}[Q] + \frac{1}{2}\mathbb{E}[(Q - \hat{B}^{\pi}\hat{Q}^{k})^2] \tag{9}$$

Optimizing over Equation 9 by taking the derivative w.r.t Q and setting it to 0, we get,

$$\begin{aligned}
\alpha\mu + (Q - \hat{B}^{\pi}\hat{Q}^{k})\hat{\pi}_{\beta} &= 0 \\
Q &= \hat{B}^{\pi}\hat{Q}^{k} - \alpha\frac{\mu}{\hat{\pi}_{\beta}}
\end{aligned}$$

Since $\hat{B}^{\pi}\hat{Q}^{k} \geq Q$, CQL underestimates Q values at each subsequent iteration.

Underestimation in CQL($\mathcal{H}$): We now follow the same process as described above and show that CQL($\mathcal{H}$) does not always underestimate Q values. Recall that the CQL($\mathcal{H}$) objective is expressed as per Equation 10.

$$\min_{Q} \alpha \mathbb{E}_{\mathcal{D}}[\log \sum_{a} \exp(Q) - \mathbb{E}_{\hat{\pi}_{\beta}}[Q]] + \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(Q - \hat{B}^{\pi}\hat{Q}^{k})^2] \tag{10}$$

Optimizing over Equation 10 by setting the derivative w.r.t Q to 0,

$$\alpha[\frac{\exp(Q)}{\sum_a \exp(Q)} - \hat{\pi}_\beta] + (Q - \hat{B}^\pi \hat{Q}^k)\hat{\pi}_\beta = 0$$
$$= \alpha[\frac{\text{softmax}(Q)}{\hat{\pi}_\beta} - 1] + Q = \hat{B}^\pi \hat{Q}^k$$
$$= Q = \hat{B}^\pi \hat{Q}^k - \alpha[\frac{\text{softmax}(Q)}{\hat{\pi}_\beta} - 1]$$

Since $\hat{B}^\pi \hat{Q}^k \geq Q$ only if $\text{softmax}(Q) \geq \hat{\pi}_\beta$, the CQL($\mathcal{H}$) does not underestimate Q values at each iteration. However, the relation holds with high probability for small values of α.

The derivation for a tighter bound consists of two stages, (1) first we will show that advice-based Q values form an upper bound on the CQL Q values $Q_{CQL} \leq Q_{ADV}$. (2) The second step consists of obtaining advice-based Q values as a lower bound on the true Q values $Q_{ADV} \leq Q$.

Upper Bound on Q_{CQL}: We begin by writing the first expectation exclusively on both its terms,

$$\alpha(\mathbb{E}_{\mathcal{D}}[\log \sum_a \exp(Q)] - \mathbb{E}_{\mathcal{D},\hat{\pi}_\beta}[Q]) + \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(Q - \hat{B}^\pi \hat{Q}^k)^2]$$

By applying Jensen's inequality twice to the first expectation, we get,

$$\leq \alpha(\log \sum_a \mathbb{E}_{\mathcal{D}}[\exp(Q)] - \mathbb{E}_{\mathcal{D},\hat{\pi}_\beta}[Q]) + \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(Q - \hat{B}^\pi \hat{Q}^k)^2]$$

Utilizing Hoeffding's inequality in the first inner expectation for the difference $(\hat{Q}^k - Q_{min})^2$ wherein Q_{min} denotes the minimum Q value up till iteration k, we obtain the following upper bound,

$$\leq \alpha(\log \sum_a \exp(\frac{1}{8}(\hat{Q}^k - Q_{min})^2) - \mathbb{E}_{\mathcal{D},\hat{\pi}_\beta}[Q]) + \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(Q - \hat{B}^\pi \hat{Q}^k)^2]$$

Lower Bound on Q: We follow the same process of optimizing over the upper bound by taking the derivative w.r.t Q and setting it to 0,

$$\alpha[2(\hat{Q}^k - Q_{min})\,\text{softmax}(\hat{Q}^k - Q_{min}) - \hat{\pi}_\beta] + (Q - \hat{B}^\pi \hat{Q}^k)\hat{\pi}_\beta = 0$$
$$= (Q - \hat{B}^\pi \hat{Q}^k)\hat{\pi}_\beta = -\alpha[2(\hat{Q}^k - Q_{min})\,\text{softmax}(\hat{Q}^k - Q_{min}) - \hat{\pi}_\beta]$$
$$Q = \hat{B}^\pi \hat{Q}^k - \frac{\alpha}{\hat{\pi}_\beta}[2(\hat{Q}^k - Q_{min})\,\text{softmax}(\hat{Q}^k - Q_{min}) - \hat{\pi}_\beta]$$

Since $\hat{Q}^k - Q_{min} \geq 0$, $\text{softmax}(\hat{Q}^k - Q_{min}) \geq 0$ and $2(\hat{Q}^k - Q_{min})\,\text{softmax}(\hat{Q}^k - Q_{min}) \geq \hat{\pi}_\beta$ with high probability, Q values at each iteration k are underestimated resulting in $Q \leq \hat{B}^\pi \hat{Q}^k$.

A.3 Upper Bound on CQL($\mathcal{H}$) using Bohning Approximation

Here we derive the upper bound on CQL($\mathcal{H}$) variant of the CQL objective. We begin by defining the shorthand notation $\text{lse}(Q) = \log \sum_a \exp(Q(s_t, a_t))$ wherein we have used Q as shorthand for $Q(s_t, a_t)$. Consider a second order Taylor's series expansion of $\text{lse}(Q)$ around $\psi \in \mathbb{R}^{|\mathcal{A}|}$,

$$\text{lse}(Q) = \text{lse}(\psi) + (Q - \psi)^T g(\psi) + \frac{1}{2}(Q - \psi)^T H(\psi)(Q - \psi)$$

where $g(\psi) = \exp(\psi - \text{lse}(\psi))$ (gradient of $\text{lse}(\psi)$),
$H(\psi) = \text{diag}(g(\psi)) - g(\psi)g(\psi)^T$ (hessian of $\text{lse}(\psi)$)

We form an upper bound on the hessian $H(\psi) < \hat{H}$ such that,

$$\hat{H} = \frac{1}{2}[I_{|\mathcal{A}|} - \frac{1}{|\mathcal{A}| + 1} 1_{|\mathcal{A}|} 1_{|\mathcal{A}|}^T]$$

Here $\hat{H}$ is the Bohning Approximation [18]. This leads to the following formulation,

$$\begin{aligned}\text{lse}(Q) &\leq \text{lse}(\psi) + (Q-\psi)^T g(\psi) + \frac{1}{2}(Q-\psi)^T \hat{H}(Q-\psi) \\ &= \frac{1}{2}Q^T\hat{H}Q - (\hat{H}\psi - g(\psi))Q + \frac{1}{2}\psi^T\hat{H}\psi - g(\psi)^T g(\psi) + \text{lse}(\psi) \\ &= \frac{1}{2}Q^T\hat{H}Q - bQ + c\end{aligned}$$

We denote the above quadratic expression with $\text{Bohn}(Q)$ where,
$b = -(\hat{H}\psi - g(\psi))$
and $c = \frac{1}{2}\psi^T\hat{H}\psi - g(\psi)^T\psi + \text{lse}(\psi)$.

Since $\text{Bohn}(Q)$ is an upper bound on $\text{lse}(Q)$, $\text{lse}(Q) \leq \text{Bohn}(Q)$, plugging it into CQL($\mathcal{H}$) objective yields the final result.

$$Q_{CQL} \leq Q_{ADV} = \arg\min_Q \mathbb{E}_{s\sim\mathcal{D}}[\text{Bohn}(Q) - \mathbb{E}_{a_t\sim\pi_b(a_t|s_t)}[Q]] + \frac{1}{2}\mathbb{E}_{s_t,a_t,s_{t+1}\sim\mathcal{D}}[(Q - \hat{B}^\pi\hat{Q}^k)^2]$$

The bound obtained above arises from a tractable quadratic approximation to the $\text{lse}(Q)$ function. Facilitation of an approximation does away with the need for complex sampling techniques from $\text{lse}(Q)$ resulting in high variance. However, **the bound trades off sampling variance with (1) a lower bound on true Q value and (2)** $\text{Bohn}\, q$ **as a contraction**. Below we throw light on these hindrances by showing that the former is challenging and the latter is only valid for some values of ψ.

(1) a lower bound on true Q value: We follow the process from the original CQL formulation [3] and optimize over the Q values in the upper bound by setting the derivative to 0,

$$\begin{aligned}&\alpha(g(\psi) + \hat{H}(Q-\psi) - \hat{\pi}_\beta) + (Q - \hat{B}^\pi\hat{Q}^k)\hat{\pi}_\beta = 0 \\ &= \alpha(\frac{g(\psi) + \hat{H}(Q-\psi) - \hat{\psi}_\beta}{\hat{\psi}_\beta}) = (\hat{B}^\pi\hat{Q}^k)\hat{\pi}_\beta - Q) \\ &= (\frac{\hat{H}}{\hat{\pi}_\beta} + 1)Q = \hat{B}^\pi\hat{Q}^k - \alpha[\frac{\text{softmax}(\psi)}{\hat{\pi}_\beta} - 1 - \frac{\hat{H}\psi}{\hat{\pi}_\beta}] \\ &= Q = \frac{1}{\hat{H} + \hat{\pi}_\beta}[\hat{\pi}_\beta\hat{B}^\pi\hat{Q}^k - \alpha(\text{softmax}(\psi) - \hat{\pi})_\beta - \hat{H}\psi)] \end{aligned} \tag{11}$$

Equation 11 denotes that the lower bound underestimates Q values at each iteration. However, the underestimation additionally scales down the subsequent values as per the entries of the matrix $\hat{H} + \hat{\pi}_\beta$. If the entries of the matrix lie outside the unit circle, then the Q values result in collapsing estimates. In case the values lie within the unit circle, the Q values will result in overestimation. Thus, a tractable lower bound would require all entries of $\hat{H} + \hat{\pi}_\beta$ on the unit circle.

(2) $\text{Bohn}(Q)$ **as a contraction**: We will now theoretically show that it is possible to preserve the contractive nature of softmax $\text{lse}(Q)$ upon replacing it with $\text{Bohn}(Q)$ in CQL($\mathcal{H}$). More specifically, we will show that the quadratic approximation $\text{Bohn}(Q)$ can be a contraction under special circumstances. We consider two Q-value functions, Q and $Q^{'}$ and define a norm $||\cdot||$. Consider the following expression,

$$\begin{aligned}&||\,\text{Bohn}(Q) - \text{Bohn}(Q^{'})|| \\ &= ||\frac{1}{2}Q^T\hat{H}Q - bQ + c - \frac{1}{2}Q^{'T}\hat{H}Q^{'} + bQ^{'} - c|| \\ &= ||\frac{1}{2}(Q^T\hat{H}Q - Q^{'T}\hat{H}Q^{'}) + b(Q^{'} - Q)|| \\ &\leq \frac{1}{2}||Q^T\hat{H}Q - Q^{'T}\hat{H}Q^{'}|| + b||Q^{'} - Q|| \end{aligned} \tag{12}$$

Wherein the first inequality results from the triangle inequality. Considering the first term in Equation 12,

$$= \frac{1}{4}||Q^T(I_{|\mathcal{A}|} - \frac{1}{|\mathcal{A}|+1}1_{|\mathcal{A}|}1_{|\mathcal{A}|}^T)Q - Q^{'T}(I_{|\mathcal{A}|} - \frac{1}{|\mathcal{A}|+1}1_{|\mathcal{A}|}1_{|\mathcal{A}|}^T)Q^{'}||$$

$$= \frac{1}{4}||Q^TQ - \frac{1}{|\mathcal{A}|+1}Q^TI_{|\mathcal{A}|}I_{|\mathcal{A}|}Q - Q^{'T}Q^{'} + \frac{1}{|\mathcal{A}|+1}Q^{'T}I_{|\mathcal{A}|}I_{|\mathcal{A}|}^TQ^{'}||$$

$$= \frac{1}{4}||Q^TQ - Q^{'T}Q^{'} + \frac{1}{|\mathcal{A}|+1}((I_{|\mathcal{A}|}^TQ^{'})^T(Q^{'T}I_{|\mathcal{A}|})^T - (I_{|\mathcal{A}|}^TQ)^T(Q^TI_{|\mathcal{A}|})^T)||$$

$$= \frac{1}{4}||Q^TQ - Q^{'T}Q^{'} + \frac{1}{|\mathcal{A}|+1}(Q^{'T}Q^{'} - Q^TQ)||$$

$$= \frac{1}{4(|\mathcal{A}|+1)}|||\mathcal{A}|Q^TQ - |\mathcal{A}|Q^{'T}Q^{'}||$$

$$= \frac{|\mathcal{A}|}{4(|\mathcal{A}|+1)}||Q^TQ - Q^{'T}Q^{'}||$$

Using this in Equation 12, we obtain the following,

$$= \frac{|\mathcal{A}|}{4(|\mathcal{A}|+1)}||Q^TQ - Q^{'T}Q^{'}|| + b|| - (Q - Q^{'})||$$

$$= \frac{|\mathcal{A}|}{4(|\mathcal{A}|+1)}||(Q - Q^{'})^T(Q + Q^{'})|| + b||(Q - Q^{'})||$$

$$\leq \frac{|\mathcal{A}|}{4(|\mathcal{A}|+1)}||(Q - Q^{'})||||(Q + Q^{'})|| + b||(Q - Q^{'})||$$

$$= (\frac{|\mathcal{A}|||Q + Q^{'}||}{4(|\mathcal{A}|+1)} + b)||Q - Q^{'}||$$

Wherein the first inequality results from Cauchy-Schwartz inequality. The above result would be a contraction if $(\frac{|\mathcal{A}|||Q+Q^{'}||}{4(|\mathcal{A}|+1)} + b) < 1$. More concretely,

$$\frac{|\mathcal{A}|||Q + Q^{'}||}{4(|\mathcal{A}|+1)} + b < 1$$

$$= ||Q + Q^{'}|| < \frac{4(1-b)(|\mathcal{A}|+1)}{|\mathcal{A}|}$$

$$= ||Q + Q^{'}|| < \frac{4(1-\hat{H}\psi - g(\psi))(|\mathcal{A}|+1)}{|\mathcal{A}|}$$

Thus, a suitable choice of ψ would allow the upper bound to hold, making the Bohning approximation a contraction.

B Implementation Details

B.1 Environment Details

Our setup makes use of the Gym PyBullet Drones [19] library which utilizes Bullet physics simulator [16] to train agents. In addition to the standard tasks of *Takeoff*, *Hover* and *Flythrugate*, we constructed an additional *ZigZag* task. All tasks simulate the agent as a CF2X quadcopter drone operating at a frequency of 240 Hz. The drone consists of 4 motor fins placed at the edges of central axes of body frame. At each timestep, the agent observes a 20 dimensional vector as its state consisting of the drones' position, translational and angular velocities, orientation and quaternion representations in 3D cartesian system. Following are the task setup corresponding to each environment for our experiments-

Takeoff: The agent is tasked to lift the drone along Z-axis of world frame. The agent controls the 4 motors by applying appropriate torques in their respective directions. Reward awarded to the agent is inversely proportional to the distance of drone's center of mass from the goal position $(0, 0, 1)$.

Hover: The agent is tasked to lift the drone along Z-axis of world frame and maintain its position above the origin. The agent controls the 4 motors by applying appropriate torques in their respective directions. Reward awarded to the agent is proportional to the norm distance of drone's center of mass from the goal position

Flythrugate: The agent is tasked to fly through a gate opening placed at a wide angle on the drone's X-axis. Note that this is a significantly challenging scenario in comparison to other tasks as it requires the agent to solve two problems. Firstly, the agent must learn identify the location of the gate. And secondly, the agent must learn to control the drone (by taking off and moving forward). The agent controls the 4 motors by applying appropriate torques in their respective directions. Reward awarded to the agent is proportional to the norm distance of drone's center of mass from the gate opening and inversely proportional to the time spent during simulation.

ZigZag: This is a novel scenario wherein the agent is tasked to move in a zig-zag flight pattern during simulation. The agent must first travel leftwards and then move rightwards towards its final goal location. The agent controls the 4 motors by applying appropriate torques in their respective directions. Reward awarded to the agent is proportional to the norm distance of drone's center of mass from the intermediate location for first half of simulation, and the distance from goal location for second half of simulation.

HoverDownfall: We modify the *Hover* task to assess the offline transfer of agent's behaviors by utilizing the downfall model. The agent, while hovering, now encounters a rain of rubber ducks which may randomly make contact with the body of the drone and alter its current position. We evaluate the robustness of drone towards the sudden contact and monitor how frequently the drone retains its hovering position . Reward for this task is kept same as the original *Hover* task.

B.2 Hyperparameters

All agents are trained for 2×10^5 (5×10^5 for *flythrugate*) timesteps and evaluated over 5 episodes every 1000 timesteps. Experiments are carried out over 16 random seeds . Agents make use of a fixed architecture setup. The agent's policy consists of two hidden layers of 1024 units each with ReLU [] activations and an output layer of 1024 units with tanh nonlinearity. Following are training details corresponding to each algorithm-

Behavior Cloning: The BC implementation is based on the conventional setup of dataset aggregation [2]. The expert, being a trained SAC agent, optimally executes actions in the environment for 5 episodes, resulting in state transitions. These transitions are stored in the dataset every 10000 steps. The agent, being the SAC learner, observes states from the dataset and yields actions based on its policy $\pi(\cdot|s_t)$. The training objective minimizes the squared norm between agent's and expert's actions and backpropagates the gradient's into agent's policy. BC agents were trained with a fixed batch size of 1024 and learning rate of 0.001 as these present the best results for all tasks.

Soft Actor-Critic: Our implementation of SAC makes use of the PyTorch implementation introduced by Yarats et. al. [20]. We utilize the diagonal Gaussian policy with standard deviation bounds rescaled in the range [2,-10] and entropy temperature tuned manually to 0.001. We additionally tried lower values of temperature but it led to deteriorating performance during training. Learning rate for all tasks was tuned to 0.00003 with higher values presenting significant variance across random seeds. Additionally, agents execute 2 SAC updates per timestep in case of *takeoff* task.

Conservative Q-Learning: Our CQL implementation is based on the original implementation of CQL-v2 [3] and makes use of the same architecture setup. The agent is trained with an increased batch size of 1024 and reduced learning rate of 0.0001 for our experiments as this presents marginal improvement. Our experiments additionally tried tuning the temperature parameters but agents were found robust to these components.

Advice-based CQL: In the interest of a fair comparison with CQL, we keep the parameter values of ADV-CQL unchanged. The temperature parameter for extrinsic motivation was kept same as the entropy temperature for CQL.