Constrained Imitation Q-learning with Earth Mover's Distance reward

Anonymous Author(s)
Affiliation
Address
email

Abstract: We propose constrained Earth Mover's Distance (CEMD) Imitation Q-learning that combines the exploration power of Reinforcement Learning (RL) and the sample efficiency of Imitation Learning (IL). Sample efficiency makes Imitation Q-learning a suitable approach for robot learning. For Q-learning, immediate rewards can be efficiently computed by a greedy variant of Earth Mover's Distance (EMD) between the observed state-action pairs and state-actions in stored expert demonstrations. In CEMD, we constrain the otherwise non-stationary greedy EMD reward by proposing a greedy EMD upper bound estimate and a generic Q-learning lower bound. In PyBullet continuous control benchmarks, CEMD is more sample efficient, achieves higher performance and yields less variance than its competitors.

Keywords: Imitation learning, Reinforcement Learning, Wasserstein Distance, Q-learning

1 Introduction

Recently, approaches combining Reinforcement Learning (RL) and Imitation Learning (IL) have gained momentum as RL and IL have complementary strengths and weaknesses for robot learning [1, 2]. Reinforcement Learning enables robots to improve autonomously, but introduces significant challenges with exploration and stable learning. Imitation learning provides more stable learning from expert demonstrations, but cannot improve from failures. It is unclear how the two approaches should be combined to take the best of the both worlds.

There have been multiple attempts to combine RL and IL. The first methods simply learned preliminary policy or its parts from expert demonstrations and the further improved the policy based on feedback received from the environment [3, 4]. More recently, methods based on Inverse Reinforcement Learning (IRL) have been proposed [5, 6, 7]. IRL methods iterate between learning the underlying reward function for the expert demonstrations and running RL to find the optimal policy for the estimated rewards by interacting with the environment. The IRL methods suffer from sample inefficiency which makes them less suitable for robot learning.

The sample hungry inverse reward optimization can be avoided by defining an explicit reward function that computes a computational reward using the difference between states and actions obtained from the environment and those in the expert demonstrations. This line of work has been found more sample efficient, and the proposed methods mainly differ in how they compute the reward [8, 9, 1]. The proposed method in this work belongs to this group of methods. The expert demonstrations are treated as finite distributions and the reward is computed via distance metric between the expert and agent obversation distributions. In our work, we adopt the Earth Mover's Distance (EMD) as the basic framework as it has been successfully used in recent works of Imitation Learning [10, 9, 1]

The main contribution of this work is **1)** a new RL-based imitation learning method, *Imitation Q-learning*, that uses Earth Mover's Distance (EMD) to compute the RL reward. The EMD reward

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

reflects how well the obtained actions and states correspond to those in the expert demonstrations. As the starting point, we adopt the greedy EMD from [1] which can be computed instantly after each new exploration sample. Our EMD reward is "tightened", i.e., upper and lower bounded, by **2)** a greedy EMD reward upper bound derived in this work and **3)** a generic Q-learning lower bound adapted from [11]. It is the tight reward constraints that make CEMD sample efficient during exploration, reduce its variance and to converge to better overall performance. These findings are experimentally validated with PyBullet continuous control benchmarks, and they make ILQ an attractive choice for robot learning.

2 Preliminaries

2.1 Reinforcement learning

In the classic RL setting [3], Markov Decision Processes (MDPs) are formalized as a tuple $M := (S, A, r, P, \gamma, P_0)$ with the state space S and the action space A. A reward function $r : S \times A \to \mathbb{R}$ maps a state-action pair to a reward. $P(s_{t+1}|s_t, a_t)$ denotes the probability of transferring from state s_t to state s_{t+1} under action a_t. $P_0(s)$ is the initial state distribution and $\gamma \in [0, 1]$ is a discount factor. At each time step t, an agent observes a state $s_t \in S$ and chooses an action $a_t \in A$ based on a policy $\pi(a_t|s_t) : S \to \mathcal{P}(\mathcal{A})$. The return $R(\tau)$ of an MDP is defined as the discounted sum of rewards $r(s_t, a_t)$ along a trajectory $\tau := (s_0, a_0, ..., s_{t-1}, a_{t-1}, s_t)$. The classic objective of RL is to learn an optimal policy π^* that maximizes the expected cumulative discounted reward:

$$\pi^* = \arg\max_{\pi} \mathop{\mathbb{E}}_{(s_t,a_t)\sim\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \quad .$$

The state-action value function, Q-function $Q^\pi(s, a)$, provides the expected cumulative discounted reward of a state s if an action a is taken and the trajectory τ according to the policy π is followed. Formally, Q-function is defined as $Q^\pi(s, a) = \mathbb{E}_\pi [R(\tau)|s, a]$. The value function obeys the Bellman equation: $Q^\pi(s, a) = \mathbb{E}_{s_{t+1}\sim P(s_{t+1}|s,a)} \left[r(s, a) + \gamma \mathbb{E}_{a_{t+1}\sim\pi}[Q^\pi(s_{t+1}, a_{t+1})] \right]$. The Bellman operator $\mathcal{B}_\pi : \mathbb{R}^n \to \mathbb{R}^n$ describes a single step of dynamic programming [3, 12]:

$$\mathcal{B}\left(Q(s, a)\right) = r(s, a) + \gamma \mathbb{E}\left[Q(s_{t+1}, a_{t+1})\right] \quad .$$

Q^π is the fixed point of the operator $\mathcal{B}_\pi(Q)$ to which the process $Q^{k+1} := \mathcal{B}_\pi(Q)^k$ converges. For high-dimensional and continuous state and action spaces, we need to approximate $Q(s, a)$ to efficiently solve for Q^π. Let's denote with θ and ω the parameters of a Q-function $Q_\theta(s, a)$ and the associated parameterized policy π_ω, correspondingly. In addition, a double target function $Q_{\bar{\theta}}$ is often used to reduce value overestimation [13]. Now, improving $Q_\theta(s, a)$ relies on finding values θ that minimize the Mean Squared Bellman Error (MSBE). Since the approximation function often takes the form of a neural network, it is convenient to define the error as a loss term:

$$L_{MSBE} = \mathbb{E}_\pi \left[(Q_\theta(s, a) - Q_{target}(s, a))^2 \right], \tag{1}$$

where $Q_{target} = r(s, a) + Q_\theta(s_{t+1}, \pi_\omega(s_{t+1}))$ is the target for minimizing MSBE, or alternatively $Q_{target} = r(s, a) + Q_{\bar{\theta}}(s_{t+1}, \pi_\omega(s_{t+1}))$ in the double target function formulation. Optimization of the MSBE loss (1) is the fundamental step in many SotA off-policy RL methods [14, 15, 13, 16, 17].

2.2 Earth Mover's Distance (EMD)

The Earth Mover Distance (EMD, a.k.a Wasserstein metric/distance) [18] is based on the minimal cost that must be paid to transform one distribution into another. Consider two finite distributions: $X = \{(x_1, w_1), ..., (x_M, w_M)\}$ with points $x_i \in \mathbb{R}^d$ and $Y = \{(y_1, u_1), ..., (y_N, u_N)\}$ with $y_j \in \mathbb{R}^d$, where $w_i \in \mathbb{R}$ and $u_j \in \mathbb{R}$ represent the weights of each point in X and Y. The flow matrix $F = (f_{ij}) \in \mathbb{R}^{m\times n}$ has elements f_{ij} that denote the amount of weight at x_i matched to the amount of weight at y_j. The EMD between two finite distributions is cast as a linear programming problem to find F that minimizes

$$\mathrm{EMD}(F, X, Y) = \min_{f_{ij}} \sum_{i=1}^{M} \sum_{j=1}^{N} f_{ij} d_{ij}, \tag{2}$$

where $d_{ij} = d(x_i, y_j)$ is the metric distance between x_i and y_j in the d-dimensional feature (or state-action) space. In this case, the flow F is a solution to the optimal transport problem such that the overall transportation cost is minimized.

The EMD linear programming problem has the following constraints: **(i)** $f_{ij} \geq 0$ requires the all weights of x_i matched to y_j be non-negative; **(ii)** $\sum_{j=1}^{N} f_{ij} \leq w_i, i = 1, 2, ..., M$ ensures that the transported weight from each x_i over all y_j does not exceed w_i; **(iii)** $\sum_{i=1}^{M} f_{ij} \leq u_j, j = 1, 2, ..., N$ ensures that weights over all x_i transported to each y_j do not exceed u_j; and (iv) $\sum_{i,j} f_{ij} = \min(\sum w_i, \sum u_j)$ forces the total flow to be equal to the total weights of the smaller of the two distributions. Fig. **??** visualizes the original EMD that we denote as the *optimal coupling EMD*.

Greedy EMD As mentioned in [1], the original optimal coupling EMD has an undesirable property to require that the two distributions X an Y are known in order to solve the optimal weight transportation problem. Consequently, with EMD the Q-function updates using the MSBE loss in (1) can be computed only after a full exploration episode is finished. This leads to sample inefficient episodic learning which in [1] is solved by *greedy coupling EMD*.

To discuss EMD in the Imitation Learning (IL) context we define X as the agent's sample trajectory after M samples and Y as the static set of trajectories obtained through expert demonstrations with N elements. The sets X and Y are either sequences of observed states, $\{s_0, s_1, ..., s_{N-1}\}$ for X, or state-action pairs $\{\langle s_0, a_0\rangle, \langle s_1, a_1\rangle, ..., \langle s_{N-1}, a_{N-1}\rangle\}$. The greedy EMD ($\text{EMD}^g$) is defined for the horizon of M observed samples in X as

$$\text{EMD}^g(F^g, X, Y) = \sum_{i=1}^{M} e_i^g = \sum_{i=1}^{M} \sum_{j=1}^{N} f_{ij}^g d_{ij} \ .$$

Let's further assume that N expert trajectory points have the capacity of $u_i = \frac{1}{N}$ and the mass of the each obtained agent's sample is $w_i = \frac{1}{M}$. The greedy EMD cost e_i^g for the sample s_i (or $\langle s_i, a_i\rangle$) is defined as:

$$\begin{gathered} e_i^g = \arg\min_{f_{ij}^g} \sum_{j=1}^{N} d(x_i, y_j) f_{ij}^g \\ \underbrace{\sum_{j=1}^{N} f_{ij}^g = \frac{1}{M}}_{\text{Constraint 1}}, \ \underbrace{\forall k \in [1:N] : \sum_{i'=1}^{i-1} f_{i'k}^g + f_{ik}^g \leq \frac{1}{N}}_{\text{Constraint 2}} . \end{gathered} \tag{3}$$

The first greedy EMD constraint ensures that all the agent's "dirt" appearing at each time step i is transported. The second constraint guarantees that each of the expert "holes" cannot be filled with more load than its capacity allows. Greedy EMD provides a sub-optimal transport solution and is an upper bound estimate of the true EMD costs $e_i^\star$ [1]:

$$\text{EMD}^g(F^g, X, Y) = \sum_{i=1}^{M} e_i^g \geq \sum_{i=1}^{M} e_i^\star \ . \tag{4}$$

The RL agent's objective is to minimize the upper bound estimate of the EMD distance between the expert and the agent policies. Figure **??** illustrates how the agent mass s_i^π is assigned to the expert holes s_j^e under the greedy-coupling EMD. The reward $r_i = r(s, a)$ in the MSBE loss equation in (1) is computed from e_i^g using some monotonically decreasing mapping $g(\cdot)$,

$$r_i = g(e_i^g) \ , \tag{5}$$

that converts small distances to large rewards. The standard mapping in this kind of conversions is the exponential function $g(x) = e^{-x}$.

3 Method

In [1] the greedy coupling EMD in (3) is used to compute the sample distance to the expert demonstrations after each sample, and the computed distance is converted to a greedy reward r^g by the exponential mapping in (5). Q-function updates are computed using the Mean Squared Bellman Error (MSBE) loss in (1). We denote the greedy EMD reward [1] based loss as $\mathcal{L}_g$ in our work. However, we next propose tighter bounds for this loss by introducing the greedy EMD reward upper bound (Section 3.1) and a general Q-learning lower bound (Section 3.2). The formulation of the final loss term using the greedy EMD reward with the lower and upper bounds is given in Section 3.3

3.1 Greedy EMD reward upper bound

It turns out that the greedy reward r^g has two undesirable properties, i) it is non-stationary and ii) it underestimates the reward by over-estimating the EMD distance (see the proofs in our supplementary material). To improve the reward computation and consequently boost learning, we introduce the lower bound e^g_{LB} of the greedy EMD effort e^g, to make a "tighter" reward estimate. A lower bound can be constructed by removing the second constraint in (3).

Lemma 1 *By removing the constraint 2 in Eq. 3, the earth moving effort e^g_{LB} for the dirt x_t appearing at time step t becomes*

$$e^g_{LB,t} = \arg\min_{f^{LB}_{t,j}} \sum_{j=1}^{N} d_{t,j} f^{LB}_{t,j}, \ s.t. \ \underbrace{\sum_{j=1}^{N} f^{LB}_{t,j} = \frac{1}{M}}_{constraint\ 1}. \tag{6}$$

and this new effort e^g_{LB} is a lower bound for the greedy coupling EMD effort e^g in (3), i.e.,

$$\forall t, \ e^g_{LB,t} \leq e^g_t \ . \tag{7}$$

By removing the second constraint from the greedy EMD equations, the lower bound greedy EMD is allowed to allocate new observations x_t to the nearest expert hole even if this hole is already occupied by the previous observations. This is not possible in the original greedy EMD formulation. The differences between the optimal EMD, greedy EMD and Lower Bound greedy EMD couplings are visualized in Figure 1.

For Q-learning the main result is that the lower bound greedy EMD formulation produces an *upper bound estimate of the greedy EMD reward*, i.e.

$$r^g_{UB,t} = g(e^g_{LB,t}) \geq r^g_t = g(e^g_t) \ . \tag{8}$$

Consequently, using the new *upper bound greedy EMD reward* we may define a new loss term $\mathcal{L}_{g,LB}$ and combine it with the original greedy loss term $\mathcal{L}_g$ to define a tighter EMD-based loss for Q-function optimization

$$\mathcal{L} = \mathcal{L}_g + \lambda \mathcal{L}_{g,UB} \ . \tag{9}$$

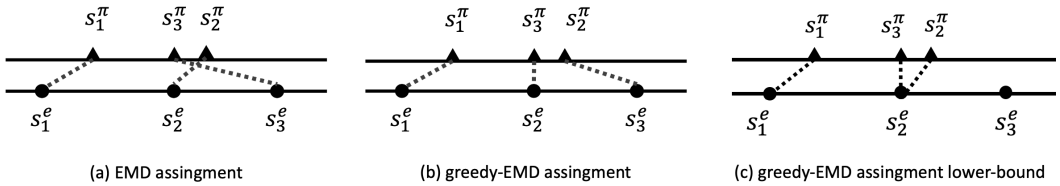


Figure 1: Optimal vs. greedy coupling EMD vs. lower bound (LB) greedy coupling EMD. (a) optimal coupling moves "dirt" from the agent states s^π_i to the closest "holes" in expert demonstrations states s^e_j. (b) greedy coupling moves dirt sequentially and thus sometimes moves it further if the closest hole is already occupied. In this example, the vector s^π_2 occurs temporally before s^π_3 and therefore the closest position s^e_2 is occupied and the dirt needs to be moved to s^π_3. (c) greedy EMD lower bound example: s^π_3 is moved to the closest hole s^e_2 even if it is already occupied by s^π_2.

3.2 Generic Q-learning Lower Bound

In off-policy Reinforcement Learning one of the main challenges is how to facilitate efficient learning from off-policy samples that can be very noisy, especially in the beginning of learning. One of the recently proposed approaches tries to benefit from a small number of particularly good episodes stored in the learning memory (aka replay buffer). This line of works is often called as *Self-Imitation Learning* (SIL) [19, 11].

The main theoretical foundation of the recent SIL methods is in tightening the Q-learning bounds. Intuitively, we learn only from samples that produce better Q-values than the current lower bound. This is particularly important in our case as the previously defined greedy EMD Lower Bound (6) forms an upper bound for the greedy EMD based reward. Therefore, by identifying a lower bound, it is possible to tighten the estimator bounds from the both sides. It turns out that we can form the greedy EMD based reward lower bound by utilizing the generic Q-learning lower bound.

The *lower bound Q-learning* was studied in [11] where they introduced an n-step bootstrapped Q-value as the general lower bound for the true Q-value under the optimal policy π^*

$$Q_{LB}(s_t, a_t) = r_t + \sum_{i=t+1}^{n-1} \gamma^t r_{t+1} + \gamma^n Q^{\pi}(s_n, a_n) \leq Q^{\pi^*}(s_t, a_t) \ .$$

In [11] the Q-function $Q_\theta(s, a)$ is updated only when $Q_\theta(s, a)$ is greater than the lower bound $Q_{LB}(s, a)$ and in our case the rule can be formulated as new loss function

$$\mathcal{L}_{LB} = \mathbb{E}\left[||(Q_\theta(s, a) - Q_{LB}(s, a))_+||^2\right] \ , \tag{10}$$

where $||\cdot||_+$ is the $\max(\cdot, 0)$ operator. As shown in [11], this n-step clipped loss brings positive bias for value estimation, and thus makes it suitable for reducing underestimation.

3.3 CEMD Loss

Our Imitation learning with constrained EMD (CEMD) Q-estimation uses a combination of the three loss terms that represent the greedy EMD based reward, and its upper and lower bounds,

$$\mathcal{L}_{CEMD} = \mathcal{L}_g + \lambda_1 \mathcal{L}_{g,UB} + \lambda_2 L_{LB} \ . \tag{11}$$

$\mathcal{L}_g$ is the greedy EMD reward loss from [1], $\mathcal{L}_{g,UB}$ is the greedy EMD reward upper bound loss defined in (9) and $\mathcal{L}_{LB}$ is the greedy EMD reward lower bound loss adapted from [11] and defined in (10). The experiments in the next section verify that the compound loss results to better overall performance, better sample efficiency (faster convergence) and smaller variance than the greedy EMD based reward loss without the bounds. In addition, CEMD is superior to other competing IL methods.

4 Experiments

4.1 Tasks and settings

The RL baseline of our method is TD3 [13] which is a popular choice for off-policy RL. The implementation is based on [20]. Experiments were conducted using the continuous control environments of PyBullet [21]. The task horizon is 1000 steps in all experiments. For each task, only five expert demonstrations were recorded and each of them is a sub-sampled trajectory with the horizon length of 50 steps. The purpose of sub-sampling is to simulate the low data regime similar and the same setting was used in [1].

4.2 Method comparison

In the first experiment, variants of the proposed CEMD are compared to SotA RL and IL methods. The tested CEMD variants are: 1) CEMD-1 using the greedy EMD loss as in [1], 2) CEMD-2 using

Table 1: Off-policy imitation learning method comparison on PyBullet environments. CEMD results include four variants of the different loss terms. CEMD-1 uses only the greedy EMD loss ($\mathcal{L}_g$) and thus corresponds to PWIL [1]. CEMD-2 uses also the greedy upper bound loss ($\mathcal{L}_g + \mathcal{L}_{g,UB}$) and CEMD-3 uses the generic lower bound ($\mathcal{L}_g + \mathcal{L}_{LB}$). Finally, CEMD-4 contains all three loss terms ($\mathcal{L}_g + \mathcal{L}_{g,UB} + \mathcal{L}_{LB}$). **Bold** values denote the best and underlined values that outperform the recent baseline [1] according to the t-test with p-value < 0.05.

Methods	CEMD-1 aka [1]	CEMD-2	CEMD-3	CEMD-4	DAC	BC	SQIL	AdRIL
Humanoid	1649.5 ± 965	$\mathbf{1931.9 \pm 881}$	$\mathbf{1921.6 \pm 840}$	$\mathbf{1982.1 \pm 826}$	953.0 ± 609	49.4 ± 10	255.7 ± 301	-31.2 ± 28
Walker2D	1985.3 ± 50	*2061.2 ± 104*	2062.0 ± 21	$\mathbf{2076.0 \pm 106}$	2049.2 ± 44	1162.4 ± 334	536.3 ± 152	1552.8 ± 584
Hopper	2406.4 ± 91	2422.0 ± 43	2466.0 ± 74	$\mathbf{2483.3 \pm 35}$	2379.1 ± 292	1362.4 ± 603	746.0 ± 67	1650.7 ± 439
Ant	1644.4 ± 1058	2326.9 ± 769	2529.6 ± 284	2566.1 ± 33	$\mathbf{2735.0 \pm 51}$	318.9 ± 46	494.5 ± 39	781.8 ± 480

the greedy EMD and its upper bound losses ($\lambda_1 = 0.3$), 3) CEMD-3 using the greedy EMD with the generic lower bound loss ($\lambda_2 = 0.3$), and 4) CEMD-4 using all three loss terms ($\lambda_1 = \lambda_2 = 0.15$). For the generic lower bound loss, $\mathcal{L}_{LB}$, the bootstrapping step size of 6 was used. We trained all variants with the limit of 1M environment interactions and on 10 random seeds.

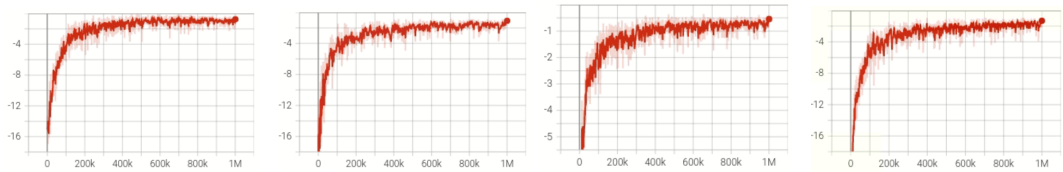
Imitation Learning research is divided to multiple lines of work, such as Behavior Cloning (BC) [22], Inverse Reinforcement Learning (IRL) [23, 24, 5, 7, 6], adversarial IL [5, 7, 6], and expert support estimation [25, 26, 27]. We chose the following off-policy methods as strong baselines to compare with: Behavior Cloning (BC) [22], Discriminator-Actor-Critic (DAC) [28], soft-Q Imitation Learning (SQIL) [29], and Adversarial Reward-Moment Imitation Learning [30]. DAC uses an off-policy RL method (TD3) to learn the imitation policy. SQIL learns to recover expert behavior by balancing the expert transitions with the agent transitions in the replay buffer of a value-based agent using off-policy RL. AdRIL was selected for the comparison as it uses trajectory matching for reward computation and at the moment it is largely considered as the SotA IL method [30]. For BC, SQIL and AdRIL, we used the code provided by the authors of [30, 31] who conducted similar experiments to us. All hyper-parameters remain the same as in the experiments in [30]. The same five expert demonstrations were used with all imitation learning methods.

We first evaluated the performance by computing the cumulative test rewards in five environments (see Table 1). Our CEMD outperformed the original greedy EMD [1] in all tasks. In HalfCheetah, all variants had similar performance. Among the variants, the one with all three loss terms has the best performance for task Humanoid, Walker2D, and Hopper. For the most challenging tasks (Ant and Humanoid), the greedy upper bound variants obtain the best rewards. Overall, the proposed CEMD variants achieve performance gains of at least 0.5%-41.0% depending on the task.

Table 2 shows the true EMD distances to the expert demonstrations. The results of Table 2 verify the results in Table 1. 400 rollouts are collected for each task in this experiment. CEMD-4 that includes all three loss terms has the smallest EMDs for all tasks. The importance of the greedy upper and generic lower bound losses is particularly evident for the Humanoid and Ant tasks for which the original greedy EMD loss in [1] remains clearly above as compared to others. Clearly, the proposed tightening bounds improve Q-learning using the greedy EMD based loss.

Table 2: The mean and standard deviation of Eearth Mover's Distances to the expert demonstrations. Bold denotes the best performance according to the t-test with p-value < 0.05.

Methods	CEMD-1	CEMD-2	CEMD-3	CEMD-4
Humanoid	14.59 ± 11.46	10.60 ± 14.43	9.93 ± 11.44	$\mathbf{8.82 \pm 9.31}$
Walker2D	1.81 ± 0.13	1.71 ± 0.72	1.65 ± 0.04	$\mathbf{1.60 \pm 0.04}$
Hopper	1.12 ± 0.07	1.30 ± 0.04	1.07 ± 0.15	$\mathbf{1.00 \pm 0.04}$
Ant	15.41 ± 20.69	6.60 ± 12.87	3.13 ± 9.33	$\mathbf{2.31 \pm 0.12}$
HalfCheetah	1.62 ± 0.16	1.58 ± 0.16	1.57 ± 0.12	$\mathbf{1.49 \pm 0.06}$



(a) Ant (b) Walker2D (c) Hopper (d) Halfcheetah

Figure 2: Difference of the greedy EMD reward and its upper bound reward, $r_g - r_{g,UB}$, during training. The negative difference verifies that $r_{g,UB}$ indeed is an upper bound estimate and as the policy converges close to the optimal, then the difference becomes zero.

4.3 Empirical verification of the greedy EMD upper bound

We wanted to empirically verify that the greedy EMD lower bound estimate defined in (6) and its corresponding reward upper bound r^g_{UB} in (8) indeed represent upper bounds for the greedy EMD. This is important as the Q-learning is based on updates using the greedy reward r^g. This experimental evaluation can be made by running the CEMD method with the original greedy reward in [1] (CEMD-1) and during the training, unroll a number of episodes and compute the average differences between the greedy EMD reward and its upper bound estimates. At every 1k environment exploration steps, a 10-episode random seed rollout is performed following the current Q-function trained with the original greedy EMD and without exploration noise. It means that every evaluation stop generates ten trajectories and each has 1000 steps horizon. Using these steps the greedy reward r_g and its greedy upper bound reward $r_{g,UB}$ are computed. The mean difference between the two, $r_{g,UB} - r_g$, is recorded and stored as a single evaluation value. These evaluation values over the increasing number of training episodes should reveal the assymptotic behaviour between the two rewards and reveal whether the upper bound holds for the greedy estimate.

Figure 2 plots the difference during training. The results in all tasks verify that the difference is negative, verifying that $r_{g,UB}$ indeed is an upper bound estimate. Moreover, there is clear difference between the two values that confirms that the upper bound reward affect learning performance. When the policy gradually improves and starts to converge near the optimal solution, the difference of the estimate and its upper bound asymptotically approaches zero and thus verifies that the estimate is unbiased.

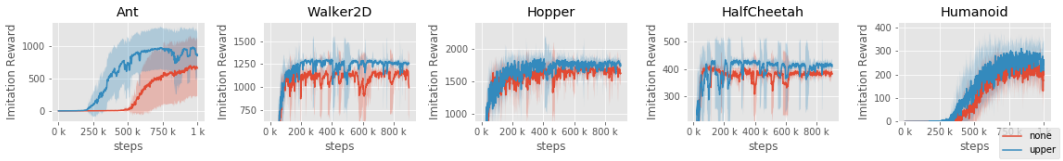
4.4 Benefits of the bounded reward for Q-learning

Finally, we wanted to verify that the bounded reward converges faster than the unbounded greedy EMD reward and is thus more sample efficient, and it achieves better performance and lower variance than the non-stationary unbounded reward. These should be verified for a realistic case where the number of expert demonstrations is low, only as few as five as in all experiments. To verify these, we computed the overall task rewards and the behavior cloning errors for the greedy EMD reward r_g Q-learning (CEMD-1) and its bounded version as proposed in this work.

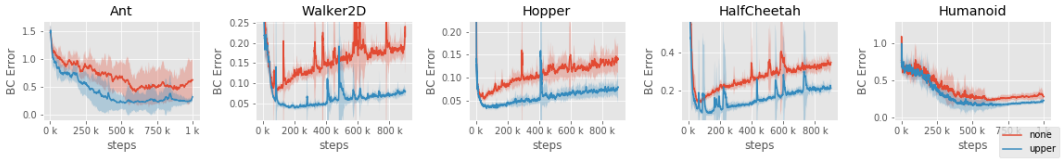
The mean test rewards, variances and behavior cloning errors are plotted in Figure 3. The results verify that Imitation Q-learning with bounded rewards converges faster (particularly evident for the Ant task) and achieves better performance than the unbounded reward estimator only. This result systematically holds for all tested environments. Moreover, the variance is reduced (see also the standard deviations in Table 1).

5 Limitations

Similarly to other imitation learning methods, the reliance on expert demonstrations is a limitation of our approach. In tasks where experts do not provide optimal demonstrations, or, where the demon-



(a) Imitation reward



(b) Behavior cloning (BC) error

Figure 3: Comparing the unbounded greedy EMD loss (red) and the bounded greedy EMD loss proposed in this work (blue): (a) the training performance measured with imitation reward (the higher the reward the smaller the Eearth Mover's Distance to expert demonstrations); (b) plots the BC error during training.

strations are performed in an environment which is not identical to the actual environment, running RL in the actual environment may be needed. Nevertheless, even in these cases, imitation learning often serves as crucial initialization of the policy due to RL needing more data than can be collected in robotic scenarios.

Since we use RL to minimize the distance to expert demonstrations (note that we do not use RL in the environment), our approach depends to an extent on the used RL method. TD3 performed well in the experiments but other methods could be also used in the future.

Our approach builds on Earth Mover's Distance (EMD) bounds. While the bounds yield significant performance increases, future work could improve the bounds further.

6 Conclusion

In this work, we proposed Imitation Q-learning (CEMD) method that learns to solve a control task using expert demonstrations to guide its exploration during learning. Exploitation of the expert demonstrations make Imitation Learning more sample efficient than Reinforcement Learning (RL) methods and thus more suitable for real robot learning. Our main contributions are in tightening Earth Mover's Distance (EMD) based reward computation that significantly boosts Q-learning by converging faster, achieving higher performance and with reduced variance - all properties that benefit robot learning. The results were verified by multiple continuous control tasks where the proposed method systematically outperforms prior arts. All code will be made publicly available to facilitate more research on distance measure based Imitation Learning.

References

[1] R. Dadashi, L. Hussenot, M. Geist, and O. Pietquin. Primal wasserstein imitation learning. In *International Conference on Learning Representations (ICLR)*, 2021.

[2] Y. Lu, K. Hausman, Y. Chebotar, M. Yan, E. Jang, A. Herzog, T. Xiao, A. Irpan, M. Khansari, D. Kalashnikov, and S. Levine. AW-Opt: Learning robotic skills with imitation and reinforcement at scale. In *Conference on Robot Learning (CoRL)*, 2021.

[3] R. S. Sutton. Introduction: The challenge of reinforcement learning. In *Reinforcement Learning*, pages 1–3. Springer, 1992.

[4] G. Tesauro et al. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[5] J. Ho and S. Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

[6] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[7] S. K. S. Ghasemipour, R. Zemel, and S. Gu. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*, pages 1259–1277. PMLR, 2020.

[8] *arXiv preprint arXiv:1911.10947*, 2019.

[9] T. Gangwani and J. Peng. State-only imitation with transition dynamics mismatch. 2020.

[10] F. Liu, Z. Ling, T. Mu, and H. Su. State alignment-based imitation learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[11] Y. Tang. Self-imitation learning via generalized lower bound q-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[12] D. Precup, R. S. Sutton, and S. Dasgupta. Off-policy temporal-difference learning with function approximation. In *ICML*, pages 417–424, 2001.

[13] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[15] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[16] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. Tb, A. Muldal, N. Heess, and T. Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[18] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

[19] J. Oh, Y. Guo, S. Singh, and H. Lee. Self-imitation learning. In *International Conference on Machine Learning (ICML)*, 2018.

[20] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann. Stable baselines3. https://github.com/DLR-RM/stable-baselines3, 2019.

[21] B. Ellenberger. Pybullet gymperium. https://github.com/benelot/pybullet-gym, 2019.

[22] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[23] J. Ho, J. Gupta, and S. Ermon. Model-free imitation learning with policy optimization. In *International Conference on Machine Learning*, pages 2760–2769. PMLR, 2016.

[24] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[25] Y. Schroecker and C. Isbell. State aware imitation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2915–2924, 2017.

[26] K.-E. Kim and H. S. Park. Imitation learning via kernel mean embedding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] K. Brantley, W. Sun, and M. Henaff. Disagreement-regularized imitation learning. In *International Conference on Learning Representations*, 2019.

[28] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Hk4fpoA5Km.

[29] S. Reddy, A. D. Dragan, and S. Levine. Sqil: Imitation learning via regularized behavioral cloning. *arXiv preprint arXiv:1905.11108*, page 88, 2019.

[30] G. Swamy, S. Choudhury, J. A. Bagnell, and S. Wu. Of moments and matching: A game-theoretic framework for closing the imitation gap, 2021.

[31] G. Swamy, S. Choudhury, J. A. Bagnell, and S. Wu. pillbox. https://github.com/gkswamy98/pillbox.git, 2021.