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

Wenyan Yang[1], **Nataliya Strokina**[1], **Joni Pajarainen**[2], **and Joni-Kristian Kämäräinen**[1]
[1]Tampere University, [2]Aalto University
Finland
first.surname@{tuni.fi,aalto.fi}

Abstract

We propose constrained Earth Mover's Distance (CEMD) Imitation Q-learning that combines exploration of Reinforcement Learning (RL) and the sample efficiency of Imitation Learning (IL). Sample efficiency makes CEMD suitable for robot learning. Immediate rewards can be efficiently computed by a greedy Earth Mover's Distance (EMD) variant between observed state-action pairs and state-actions in the stored expert demonstrations. In CEMD, we constrain the previously proposed 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.

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 [4, 17]. 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 then further improved the policy based on feedback received from the environment [27, 31]. More recently, methods based on Inverse Reinforcement Learning (IRL) have been proposed [11, 6, 9]. 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 a computational reward function. A computational reward is computed 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 [1, 8, 4]. 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. EMD has been used in recent works of Imitation Learning [16, 8, 4]

The main contribution of this work is **1)** a new RL-based imitation learning method, CEMD, that uses Earth Mover's Distance (EMD) to compute the RL reward. The EMD reward reflects how well the obtained actions and states correspond to those in the expert demonstrations. As the starting point, we

36th Conference on Neural Information Processing Systems (NeurIPS 2022) Deep RL Workshop.

adopt the greedy EMD from [4] which can be computed instantly after each new exploration sample. Our EMD reward is *tightened* by more strict upper and lower bounds. We **2)** derive a greedy EMD reward upper bound and **3)** a generic Q-learning lower bound. The lower bound is adapted from [30]. The tight reward constraints make CEMD sample efficient during exploration, reduce its variance, and achieve better overall performance. These findings are experimentally validated with PyBullet continuous control benchmarks.

2 Preliminaries

2.1 Reinforcement learning

In the classic RL setting [27], 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 [27, 21]:

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

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 [7]. 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 [18, 15, 7, 2, 10].

2.2 Earth Mover's Distance (EMD)

The Earth Mover Distance (EMD, a.k.a Wasserstein metric or distance) [25] 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. 1(a) visualizes the original EMD that we denote as the *optimal coupling EMD*.

Greedy EMD As mentioned in [4], 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, Q-function updates using the MSBE loss (1) and EMD can be computed only after an exploration episode is completed. This leads to sample inefficient episodic learning which in [4] is solved by proposing a *greedy coupling EMD*.

To bring EMD to the Imitation Learning (IL) context, we define X as the agent's sample trajectory after M samples, and Y of N elements is the static set of trajectories obtained through expert demonstrations. The sets X and Y are sequences of observed states, $\{s_0, s_1, ..., s_{N-1}\}$ for X, or state-action pairs $\{\langle s_0, a_0\rangle, \langle s_1, a_1\rangle, ..., \langle s_{N-1}, a_{N-1}\rangle\}$, depending on the problem. The greedy EMD (EMD^g) is defined for horizon of M observations 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 sample is $w_i = \frac{1}{M}$. The greedy EMD cost e_i^g for the sample s_i is defined as:

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

The first greedy EMD constraint ensures that all the agent's *dirt* (EMD terminology) appearing at the time step i is transported. The second constraint guarantees that each of the expert *holes* cannot take more dirt load than its capacity allows. Greedy EMD provides a sub-optimal transport solution, and is an *upper bound estimate* of the true EMD costs $e_i^\star$ [4]:

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

The RL agent objective is to minimize the upper bound estimate of the EMD distance between the expert and the agent policies. Figure 1(b) 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 MSBE loss (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. A common mapping is the exponential function $g(x) = e^{-x}$.

3 Method

In [4] the greedy coupling EMD (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 (5). Q-function updates are computed using the Mean Squared Bellman Error (MSBE) loss (1). We denote this greedy EMD reward [4] loss as $\mathcal{L}_g$. Next, we ropose tighter bounds for the loss by introducing the greedy EMD reward upper bound (Section 3.1) and a general Q-learning lower bound (Section 3.2). The new loss term is defined in Section 3.3

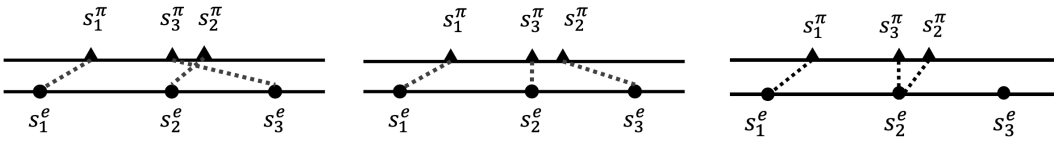


Figure 1: (a) optimal vs. (b) greedy coupling vs. (c) lower bound (LB) greedy coupling EMD. Optimal coupling moves dirt from the agent states s_i^π to the closest holes in the expert demonstration states s_j^e. Greedy coupling moves dirt sequentially and thus sometimes moves it further if the closest hole is occupied. In this example, the vector s_2^π occurs temporally before s_3^π, and therefore the closest position s_2^e is occupied and the dirt needs to be moved to s_3^π. In our greedy EMD lower bound s_3^π is moved to the closest hole s_2^e even if it is already occupied.

3.1 Upper bound of the Greedy EMD reward

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 (the proof is in our supplementary material). To improve the reward computation to boost learning by a tighter estimate, we introduce the lower bound e^g_{LB} of the greedy EMD effort e^g. The lower bound is 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*

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

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, the lower bound greedy EMD is allowed to allocate new observations x_t to the nearest expert hole even if the 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.

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

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

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

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 (a replay buffer). This line of works is often called as *Self-Imitation Learning* (SIL) [20, 30].

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. Therefore, by identifying a similar lower bound, it is

possible to tighten the estimator from the both directions. It turns out that we can form the greedy EMD lower bound by utilizing the generic Q-learning lower bound.

The *lower bound Q-learning* was studied in [30] 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 [30] 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 is formulated inside the 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 [30], this n-step clipped loss brings positive bias for value estimation, and thus makes it suitable for reducing underestimation.

3.3 Constrained EMD Loss

Imitation learning with the proposed 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 [4], $\mathcal{L}_{g,UB}$ is the greedy EMD reward upper bound loss defined in (9), and $\mathcal{L}_{LB}$ is the greedy EMD reward lower bound loss 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 [7] which is a popular choice for off-policy RL. The implementation is based on [22]. Experiments were conducted using the continuous control environments of PyBullet [5]. The task horizon is 1000 steps in all experiments. For each task, only five expert demonstrations were recorded and each of them is sub-sampled to the horizon length of 50 steps. The purpose of sub-sampling is to simulate the low data regime. The same setting was used in [4].

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 [4], 2) CEMD-2 using the greedy EMD and the upper bound loss ($\lambda_1 = 0.3$), 3) CEMD-3 using the greedy EMD and the generic lower bound loss ($\lambda_2 = 0.3$), and 4) CEMD-4 using all three losses ($\lambda_1 = \lambda_2 = 0.15$). 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 10 random seeds.

Imitation Learning research is divided to multiple lines of work, such as Behavior Cloning (BC) [24], Inverse Reinforcement Learning (IRL) [12, 19, 11, 9, 6], adversarial IL [11, 9, 6], and expert support estimation [26, 13, 3]. We chose the following off-policy methods as strong baselines to compare with: Behavior Cloning (BC) [24], Discriminator-Actor-Critic (DAC) [14], soft-Q Imitation Learning (SQIL) [23], and Adversarial Reward-Moment Imitation Learning [29]. 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. 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 [29]. For BC, SQIL and AdRIL, we used the code provided by the authors of [29, 28] who conducted similar experiments. All hyper-parameters remain the same as in the experiments in [29]. The same five expert demonstrations were provided to all IL methods.

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 [4]. CEMD-2 in addition uses 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 [4] according to the t-test with p-value < 0.05.

Methods	CEMD-1 aka [4]	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

We first evaluated the performance by computing the cumulative test rewards in five environments (see Table 1). Our CEMD outperformed the original greedy EMD [4] in all tasks. In HalfCheetah, all variants had similar performance. Among the variants, the one with all three loss terms was the best for Humanoid, Walker2D, and Hopper. For the most challenging tasks (Ant and Humanoid), the greedy upper bound variants obtained the best results. Overall, the proposed CEMD variants achieve performance gains 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 [4] remains substantially better than the others. Clearly, the proposed tightening bounds improve Q-learning using the greedy EMD based loss.

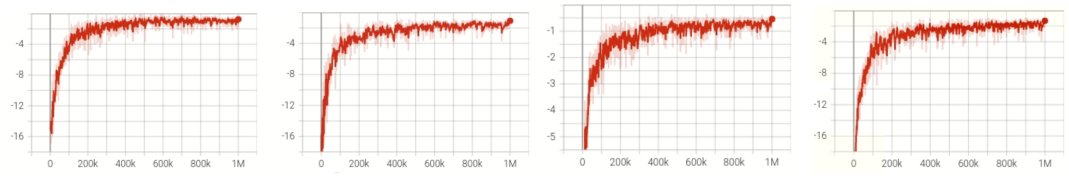
4.3 Empirical verification of the greedy EMD upper bound

It is important 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. Verification is important as the Q-learning is based on updates using the greedy reward r^g. This was made by running the CEMD method with the original greedy reward in [4] (CEMD-1). During training, number of episodes were unrolled and the average differences between the greedy EMD reward and its upper bound estimates were computed. At every 1k environment exploration steps, a 10-episode random seed roll-out was performed. 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 $r_{g,UB}$ were computed. The mean difference between the two, $r_{g,UB} - r_g$, were 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 whether the upper bound holds for the greedy estimate.

Figure 2 shows the differences during training. In all tasks, the differences are negative, verifying that $r_{g,UB}$ indeed is an upper bound estimate. Moreover, there are clear differences between the two values, that confirms that the upper bound reward affects learning performance. When the policy gradually improves, and starts to converge near the optimal solution, the difference asymptotically approaches zero, and verifies negligible bias in the estimate.

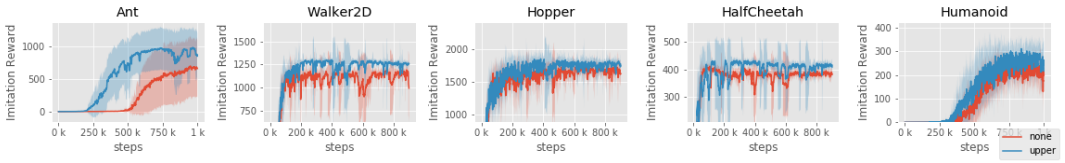
Table 2: The mean and standard deviation of Earth Mover's Distances to the expert demonstrations. Bold denotes the best performance according to the t-test with p-value < 0.05.

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	$\mathbf{1.49 \pm 0.06}$

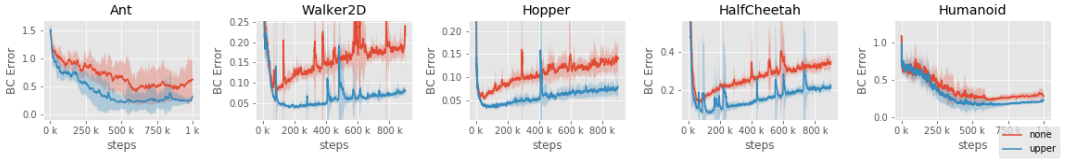


(a) Ant (b) Walker2D (c) Hopper (d) Halfcheetah

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.



(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 Earth Mover's Distance to expert demonstrations); (b) plots the BC error during training.

4.4 Benefits of the bounded reward for Q-learning

To verify that the bounded reward converges faster than the unbounded greedy EMD reward the two were compared. The comparison was made in a realistic case where the number of expert demonstrations was low (five). 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.

The mean test rewards, variances, and behavior cloning errors are shown in Figure 3. The results verify that Imitation Q-learning with bounded rewards converges faster (particularly evident in the Ant task), and achieves better performance than the unbounded reward. This systematically holds for all tested environments. Moreover, the variance is reduced (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 demonstrations are performed in an environment which is not identical to the actual environment, running RL in the actual environment may be needed. Nevertheless, imitation learning serves as initialization of the policy.

Since we use RL to minimize the distance to expert demonstrations, the performance depends on the selected RL. TD3 performed well in the experiments, but other methods should be considered in the future work.

Our approach builds on Earth Mover's Distance (EMD) bounds. While the current bounds already provide a performance boost, better bounds should be investigated.

6 Conclusion

In this work, we proposed an Imitation Q-learning method, CEMD, that learns to solve a control task using expert demonstrations to guide its learning. The expert demonstrations make Imitation Learning more sample efficient than plain Reinforcement Learning (RL), and therefore more suitable for real robot cases. Our main contributions are the tighter bounds for Earth Mover's Distance (EMD) based reward computation. The bounds boost Q-learning to converge faster, to achieve higher performance, and to reduced variance. The results were verified by multiple continuous control tasks where the proposed method systematically outperforms its competitors. All code will be made publicly available.

Acknowledgments

Wenyan Yang received funding from the Doctoral School of Industry Innovations of Tampere University, and Cargotec Ltd.

References

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

[2] Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

[3] Kianté Brantley, Wen Sun, and Mikael Henaff. Disagreement-regularized imitation learning. In *International Conference on Learning Representations*, 2019.

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

[5] Benjamin Ellenberger. Pybullet gymperium. `https://github.com/benelot/pybullet-gym`, 2019.

[6] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[7] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

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

[9] Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*, pages 1259–1277. PMLR, 2020.

[10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey 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.

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

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

[13] Kee-Eung Kim and Hyun Soo Park. Imitation learning via kernel mean embedding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[14] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan 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`.

[15] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

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

[17] 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.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[19] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[20] Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International Conference on Machine Learning (ICML)*, 2018.

[21] Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *ICML*, pages 417–424, 2001.

[22] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3. `https://github.com/DLR-RM/stable-baselines3`, 2019.

[23] Siddharth Reddy, Anca D Dragan, and Sergey Levine. Sqil: Imitation learning via regularized behavioral cloning. *arXiv preprint arXiv:1905.11108*, page 88, 2019.

[24] Stéphane Ross, Geoffrey Gordon, and Drew 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.

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

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

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

[28] Gokul Swamy, Sanjiban Choudhury, J Andrew Bagnell, and Steven Wu. pillbox. `https://github.com/gkswamy98/pillbox.git`, 2021.

[29] Gokul Swamy, Sanjiban Choudhury, J Andrew Bagnell, and Steven Wu. Of moments and matching: A game-theoretic framework for closing the imitation gap. In *International Conference on Machine Learning*, pages 10022–10032. PMLR, 2021.

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

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