The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Imitation Learning via Kernel Mean Embedding

Kee-Eung Kim
School of Computer Science
KAIST
kekim@cs.kaist.ac.kr

Hyun Soo Park
Department of Computer Science and Engineering
University of Minnesota
hspark@umn.edu

Abstract

Imitation learning refers to the problem where an agent learns a policy that mimics the demonstration provided by the expert, without any information on the cost function of the environment. Classical approaches to imitation learning usually rely on a restrictive class of cost functions that best explains the expert's demonstration, exemplified by linear functions of pre-defined features on states and actions. We show that the kernelization of a classical algorithm naturally reduces the imitation learning to a distribution learning problem, where the imitation policy tries to match the state-action visitation distribution of the expert. Closely related to our approach is the recent work on leveraging generative adversarial networks (GANs) for imitation learning, but our reduction to distribution learning is much simpler, robust to scarce expert demonstration, and sample efficient. We demonstrate the effectiveness of our approach on a wide range of high-dimensional control tasks.

In imitation learning, an agent learns to behave by mimicking the demonstration provided by the expert, situated in an environment with an unknown cost function. A classical approach to imitation learning is behavioral cloning, where the policy is directly learned to map from states to actions by supervised learning (Sammut 2010). Unfortunately, this straightforward approach does not generalize well to unseen states, often requiring a large amount of training data. A more principled approach is apprenticeship learning (AL), where the policy is sought that is guaranteed to perform *at least as well as* the expert (Russell 1998; Ng, Russell, and others 2000; Abbeel and Ng 2004). However, to formally meet the guarantee, AL algorithms typically assume a restrictive class of cost functions and a planner that yields a sufficiently accurate optimal policy for a cost function. This does not reflect the complex nature of high-dimensional dynamics in real-world problems.

On the other hand, deep neural networks have been shown strong predictive power to model complex functions: the parametric function via networks is highly flexible and expressive, which can be efficiently trained by stochastic gradient descent. Representing the cost function and the agent policy using neural networks shall yield a plausible policy that faithfully imitates the expert's demonstrated behaviors in high-dimensional control tasks. In this line of thought, Ho and Ermon (2016) presented generative adversarial imitation learning (GAIL), which casts the objective of imitation learning as the training objective of generative adversarial networks (GANs) (Goodfellow et al. 2014). The key insight behind GAIL is that imitation learning reduces to matching the state-action visitation distributions (i.e. occupancy measure) of the learned policy to that of the expert policy, under a suitable choice of the penalty on the cost function. However, GAIL often exhibits unstable training in practice due to the alternating optimizations of the generator and discriminator networks to address the minimax objective function, a well-known challenge in training GANs.

In this work, we show that extending the class of cost functions to the reproducing kernel Hilbert space (RKHS) alternatively reduces the imitation learning to the distribution learning problem under the maximum mean discrepancy (MMD), a metric on probability distributions defined in the RKHS. However, our derivation is much simpler and more natural. Although the derivation is almost immediate, our work is the first to present the kernelization of a classical AL algorithm (Abbeel and Ng 2004), and establish analogies with the state of the art imitation learning algorithm, i.e. GAIL. The advantage of our work is that the training becomes simpler yet robust to local optima since the hard minimax optimization is avoided. As an end result, our work becomes closely related to generative moment matching networks (GMMNs) (Li, Swersky, and Zemel 2015) and MMD nets (Dziugaite, Roy, and Ghahramani 2015), two approaches to training deep generative neural networks using the MMD. Our experiments on the same set of high-dimensional control imitation tasks with the identical settings as in the GAIL paper, with the largest task involving 376 observation and 17 action dimensions, demonstrate that the proposed approach performs better than or on a par with GAIL, and significantly outperforms GAIL particularly when the expert demonstration is scarce, with performance gain up to 41%.

Background

MDPs and Imitation Learning We define basic notation for our problem setting and briefly review relevant work in the literature. We assume learning in an environment that can be modeled as a Markov decision process (MDP), with state space S, action space A, transition model $p(s'|s,a)$, initial state distribution $p_0(s)$, and cost function $c(s,a)$. We assume the total discounted cost with discount rate γ, so that the long-term cost of a policy $\pi : S \to p(A)$ is defined as

$$J_c(\pi) \triangleq E_{s_0,a_0,\cdots}[\sum_{t=0}^{\infty} \gamma^t c(s_t,a_t)],$$

where the trajectory $[s_0, a_0, \cdots]$ is generated by $s_0 \sim p_0(s_0), s_{t+1} \sim p(s_{t+1}|s_t,a_t), a_t \sim \pi(a_t|s_t), \forall t \geq 0$. We also define the state-action value function $Q_c^\pi(s,a) \triangleq E_{s_t=s,a_t=a,\cdots}[\sum_{t'=t}^{\infty} \gamma^{t'-t} c(s_{t'},a_{t'})]$, the state value function $V_c^\pi(s) \triangleq E_{a\sim\pi(\cdot|s)}[Q_c^\pi(s,a)]$ and the advantage function $A_c^\pi(s,a) \triangleq Q_c^\pi(s,a) - V_c^\pi(s)$.

For any policy π, there is a one-to-one correspondence between the policy and its *occupancy measure* (Puterman 1994)

$$\begin{aligned}\rho_\pi(s,a) &\triangleq E_{s_0,a_0,\cdots}[\sum_{t=0}^{\infty} \gamma^t \delta_{s,a}(s_t,a_t)] \\ &= \sum_{t=0}^{\infty} \gamma^t p(s_t = s, a_t = a) \qquad (1)\end{aligned}$$

where δ is the Kronecker delta function. Essentially, this is an unnormalized visitation distribution over states and actions, which sums to $1/(1-\gamma)$. The long-term cost is then simply the expected cost under the occupancy measure,

$$J_c(\pi) = \sum_{s,a} c(s,a)\rho_\pi(s,a) \triangleq E_{s,a\sim\rho_\pi}[c(s,a)].$$

When the state and the action spaces are large or continuous, the classical approach is to work with pre-defined features $\vec{\phi}(s,a) \in \Re^K$ and assume a linear cost function in terms of features: $c(s,a) \triangleq \vec{w}^\top \vec{\phi}(s,a)$. In this case, it is easy to see that using the *feature expectation*

$$\vec{\mu}_\pi \triangleq E_{s_0,a_0,\cdots}[\sum_{t=0}^{\infty} \gamma^t \vec{\phi}(s_t,a_t)] = E_{s,a\sim\rho_\pi}[\vec{\phi}(s,a)], \qquad (2)$$

the long-term cost is $J_c(\pi) = \vec{w}^\top \vec{\mu}_\pi$ by the linearity of expectation.

In imitation learning, we assume demonstration dataset D_{π_E} provided by the expert, whose behavior is governed by policy π_E unknown to the agent. The goal is to compute policy π that best approximates π_E given the demonstration dataset, without any further information about the underlying MDP model, except a simulator that can sample trajectories given any policy π.

Apprenticeship learning (AL) is an approach to imitation learning, which formalizes this problem by defining a class of cost functions $\mathcal{C}$ and seeks policy π that performs as well as π_E for all $c \in \mathcal{C}$, i.e.

$$J_c(\pi) \leq J_c(\pi_E) \quad \text{for all } c \in \mathcal{C}$$

Note that if the true cost function is in $\mathcal{C}$, the policy π that satisfies this inequality is guaranteed to perform as well as, or better than, π_E. This constraint satisfaction problem can be reformulated as an optimization problem, with the objective

$$\min_\pi \psi_{\mathcal{C}}^*(\pi,\pi_E) \text{ where } \psi_{\mathcal{C}}^*(\pi,\pi_E) = \max_{c\in\mathcal{C}} [J_c(\pi) - J_c(\pi_E)] \qquad (3)$$

The AL algorithm by Abbeel and Ng (2004) can be seen as choosing the cost function class

$$\mathcal{C}_{\ell_2} = \left\{ c(s,a) = \vec{w}^\top \vec{\phi}(s,a) \;\middle|\; \|\vec{w}\|_2 \leq 1 \right\}$$

which yields

$$\psi_{\mathcal{C}_{\ell_2}}^*(\pi,\pi_E) = \max_{\|\vec{w}\|_2\leq 1} [\vec{w}^\top \vec{\mu}_\pi - \vec{w}^\top \vec{\mu}_{\pi_E}] \qquad (4)$$

whereas Multiplicative Weights Apprenticeship Learning (MWAL) by Syed and Schapire (2007) is associated with the cost function class

$$\mathcal{C}_\Delta = \left\{ \vec{w}^\top \vec{\phi}(s,a) \;\middle|\; \vec{w} \in \Delta \right\}$$

where Δ denotes the simplex constraint, i.e. $\vec{w} \geq \vec{0}$ and $\sum w_i = 1$. This yields

$$\psi_{\mathcal{C}_\Delta}^*(\pi,\pi_E) = \max_{\vec{w}\in\Delta} [\vec{w}^\top \vec{\mu}_\pi - \vec{w}^\top \vec{\mu}_{\pi_E}] \qquad (5)$$

The minimax optimization problem in Eqn. (3) typically involves repetitive computations of optimal policies for intermediate cost functions (Abbeel and Ng 2004; Syed and Schapire 2007). This is intractable for large-scale problems, especially those with continuous action spaces. Ho, Gupta, and Ermon (2016) presented a gradient-based stochastic optimization approach, where the parameterized policy (typically a neural network) π_θ is found by alternating between computing the cost function c^* that attains maximum in $\psi_{\mathcal{C}}^*$ with fixed π_θ, and then improving π_θ by gradient descent using $\nabla_\theta \psi_{c^*}^*(\pi_\theta,\pi_E)$. This approach can be used with different cost function classes, e.g. $\mathcal{C}_{\ell_2}$ or $\mathcal{C}_\Delta$. In the experiments, we will refer to these two gradient-based versions of AL as Feature Expectation Matching (FEM) and Game-Theoretic Apprenticeship Learning (GTAL).

Kernel Mean Embedding and MMD This paper seeks a kernel-based imitation learning algorithm that does not rely on explicit features $\vec{\phi}(s,a)$. The very first step is to realize that the feature expectation given in Eqn. (2) is actually the mean embedding of the distribution ρ_π using the feature $\vec{\phi}$. This motivates the use of kernel mean embedding, which extends the classical kernel approach to probability distributions (Smola et al. 2007). Specifically, choosing a kernel implies an implicit feature map ϕ that represents a probability distribution as a mean function,

$$\mu_p(\cdot) \triangleq \int_{\mathcal{X}} k(x,\cdot)dp(x) = \int_{\mathcal{X}} \phi(x)dp(x)$$

which is an element in the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ of functions on $\mathcal{X}$ with reproducing kernel k. This approach has a number of useful properties. First, since $\mathcal{H}$ is an RKHS, by the Riesz representation theorem, there is a feature map $\phi(x)$ from $\mathcal{X}$ to $\Re$ such that

$$f(x) = \langle f, \phi(x) \rangle_{\mathcal{H}} = \langle f, k(x,\cdot) \rangle_{\mathcal{H}}$$

when f belongs to $\mathcal{H}$ (Reed and Simon 1980; Steinwart and Christmann 2008). Note that this implies $\langle \phi(x), \phi(y) \rangle_{\mathcal{H}} = k(x,y)$, and in addition, $E_{x\sim p}[f(x)] = \langle f, \mu_p \rangle_{\mathcal{H}}$. Second, for a class of kernels known as characteristic kernels, the mean map ϕ is injective, i.e. $\|\mu_p - \mu_q\|_{\mathcal{H}} = 0$ if and only if p and q are the same distribution ($p = q$). Hence, the kernel mean embedding can be used to define a metric for probability distributions (Fukumizu, Bach, and Jordan 2004; Sriperumbudur et al. 2008).

The maximum mean discrepancy (MMD) (Gretton et al. 2012) is an instance of the integral probability metric (IPM) (Müller 1997) defined with an RKHS:

$$\begin{aligned}\mathrm{MMD}[\mathcal{H}, p, q] &\triangleq \sup_{\|f\|_{\mathcal{H}} \leq 1} \left[\int f(x)dp(x) - \int f(y)dq(y)\right] \\ &= \sup_{\|f\|_{\mathcal{H}} \leq 1} \langle \mu_p - \mu_q, f\rangle_{\mathcal{H}} \qquad (6) \\ &= \|\mu_p - \mu_q\|_{\mathcal{H}}\end{aligned}$$

Hence, $\mathrm{MMD}[\mathcal{H}, p, q] = 0$ if and only if $p = q$ with a characteristic kernel. Given two sets of samples $D_X = \{x_i\}_{i=1}^N$ and $D_Y = \{y_i\}_{i=1}^M$ from p and q respectively, a (biased) empirical estimate of the MMD is obtained by

$$\begin{aligned}\widehat{\mathrm{MMD}}^2[\mathcal{H}, D_X, D_Y] &= \|\hat{\mu}_p - \hat{\mu}_q\|_{\mathcal{H}}^2 \\ &= \Big\| \frac{1}{N}\sum_{i=1}^N \phi(x_i) - \frac{1}{M}\sum_{j=1}^M \phi(y_j)\Big\|^2 \\ &= \frac{1}{N^2}\sum_{i=1}^N\sum_{i'=1}^N k(x_i, x_{i'}) + \frac{1}{M^2}\sum_{j=1}^M\sum_{j'=1}^M k(y_j, y_{j'}) \\ &\quad - \frac{2}{NM}\sum_{i=1}^N\sum_{j=1}^M k(x_i, y_j)\end{aligned}$$

The function f^* that attains the supremum in Eqn. (6) is defined as the *witness function*. Since $f^* \propto \mu_p - \mu_q$, we have

$$f^*(z) \propto \langle \mu_p - \mu_q, \phi(z)\rangle_{\mathcal{H}} = E_{x\sim p}[k(x,z)] - E_{y\sim q}[k(y,z)]$$

$$\hat{f}^*(z) \propto \langle \hat{\mu}_p - \hat{\mu}_q, \phi(z)\rangle_{\mathcal{H}} = \tfrac{1}{N}\sum_{i=1}^N k(x_i, z) - \tfrac{1}{M}\sum_{j=1}^M k(y_j, z)$$

where $\hat{f}^*$ is the empirical estimate of f^*, and the normalizing constant is $\|\mu_p - \mu_q\|_{\mathcal{H}}$.

Generative Moment Matching Imitation Learning

The cost function class in the imitation learning objective (Eqn. (4)), i.e. linear functions on pre-defined features, is too restrictive. We want to learn with a more expressive class of cost functions, yet keep the optimization tractable. The main idea behind our approach is to reformulate the imitation learning objective using kernels. First, we assume the cost function class

$$\mathcal{C}_{\mathcal{H}} = \Big\{c(s,a) = \langle c, \phi(s,a)\rangle \;\Big|\; \|c\|_{\mathcal{H}} \leq 1\Big\}$$

defined in RKHS $\mathcal{H}$. Then, the objective function becomes

$$\begin{aligned}\psi^*_{\mathcal{C}_{\mathcal{H}}}(\pi, \pi_E) &= \sup_{\|c\|_{\mathcal{H}} \leq 1} \Big[\int c(s,a)d\rho_\pi(s,a) \\ &\qquad - \int c(s',a')d\rho_{\pi_E}(s',a')\Big] \\ &= \sup_{\|c\|_{\mathcal{H}} \leq 1} \langle \mu_\pi - \mu_{\pi_E}, c\rangle_{\mathcal{H}} \qquad (7) \\ &= \|\mu_\pi - \mu_{\pi_E}\|_{\mathcal{H}}\end{aligned}$$

which is exactly the MMD between two distributions ρ_π and ρ_{π_E}, i.e. $\mathrm{MMD}[\mathcal{H}, \rho_\pi, \rho_{\pi_E}]$. Hence, imitation learning naturally reduces to matching the state-action distribution of the agent to that of expert, in which we use the MMD as the metric between two distributions.

This also yields an insight on how our approach relates to Generative Adversarial Imitation Learning (GAIL) (Ho and Ermon 2016). GAIL casts the minimax optimization problem in Eqn. (3) as the training objective of Generative Adversarial Networks (GANs). It was shown in the GAIL paper that, instead of assuming a specific cost function class to be searched, a sophisticated penalty function defined directly over generic cost functions leads to the objective function

$$\begin{aligned}\psi^*_{GA}(\pi_\theta, \pi_E) = \max_{d\in(0,1)^{S\times A}} \Big[&E_{s,a\sim\rho_{\pi_\theta}}[\log d(s,a)] \\ &+ E_{s,a\sim\rho_{\pi_E}}[\log(1 - d(s,a))]\Big]\end{aligned} \qquad (8)$$

where d is a binary function defined over all possible states and actions. Hence, optimizing $\min_\theta \psi^*_{GA}(\pi_\theta, \pi_E)$ essentially becomes training a GAN, where the discriminator $d(s,a)$ tries to discriminate between the trajectories from π_θ and those from π_E, and the generator is the policy π_θ that tries to minimize the long-term cost with cost function $c(s,a) = \log d(s,a)$. It can be shown that the optimum of $\psi^*_{GA}(\pi, \pi_E)$ is the Jensen-Shannon divergence between the occupancy measures of π and π_E: $JS(\rho_\pi, \rho_{\pi_E}) \triangleq KL(\rho_\pi \| (\rho_\pi + \rho_{\pi_E})/2) + KL(\rho_{\pi_E} \| (\rho_\pi + \rho_{\pi_E})/2)$ (Goodfellow et al. 2014; Ho and Ermon 2016). In summary, assuming that we can obtain the optimum d inside $\psi^*_{GA}(\pi, \pi_E)$, the optimization objective of GAIL becomes

$$\min_\theta JS(\rho_{\pi_\theta}, \rho_{\pi_E})$$

where we are trying to match the state-action distribution of the agent to that of expert, in which the probability metric is Jensen-Shannon divergence, instead of MMD.

Our reformulation of the imitation learning objective using kernels has three immediate advantages over GAIL. First, we have shown that the imitation learning can be reduced to matching two probability distributions, via a much simpler argument using kernel mean embedding. Second, we have avoided alternating minimax optimization in GAN, which is known to be prone to local convergence. In GAIL, the cost function is defined as the discriminator in GAN, which is not an exact optimum of Eqn. (8) but an approximation using a neural network. In contrast, the cost function in our approach is the witness function of MMD, which is the exact closed-form optimum of Eqn. (7). Lastly, and practically, we can monitor the progress via MMD, which indicates how well the policy is imitating. In contrast, the JS estimate from GANs is known to be not much meaningful.

Our algorithm is shown in Algorithm 1, which we call generative moment matching imitation learning (GMMIL). We assume a multi-layer neural network for representing the policy, e.g. a Gaussian distribution with its mean and precision parameterized by neural networks that take raw state encodings as input, and use Trust Region Policy Optimiza-

Algorithm 1: Generative Moment Matching Imitation Learning

```
Input : Expert dataset of trajectories $D_{\pi_E} = \{(s_i^E, a_i^E)\}_{j=1}^M$
Initialize policy parameter $\theta$
for $iter = 0, 1, \dots$ do
    Sample trajectories $D_{\pi_\theta} = \{(s_i, a_i)\}_{i=1}^N$ by executing $\pi_\theta$
    Calculate the empirical estimate of the witness function $\hat{f}^*(s,a)$ of MMD$[D_{\pi_\theta}, D_{\pi_E}]$ for all $(s,a) \in D_{\pi_\theta}$:
        $\hat{f}^*(s,a) = \frac{1}{N}\sum_{i=1}^N k((s_i,a_i),(s,a)) - \frac{1}{M}\sum_{j=1}^M k((s_j^E,a_j^E),(s,a))$
    Update the policy parameter $\theta$ using the TRPO gradient update with cost function $c^*(s,a) = \hat{f}^*(s,a)$.
end
```

tion (TRPO) (Schulman et al. 2015) for optimizing the policy with the witness function taken as the cost function.

We remark that our approach closely resembles Generative Moment Matching Networks (GMMNs) (Li, Swersky, and Zemel 2015) and MMD nets (Dziugaite, Roy, and Ghahramani 2015), where a neural network generator is trained to minimize the MMD between the training instances and generated samples. In our case, the neural network generator is the imitation policy, and the training instances are the expert trajectories.

Occupancy measure as sum of distributions Cautious readers may notice that the occupancy measure in Eqn. (1) is actually a discounted sum of probability distributions for each time step, rather than a stationary, unnormalized distribution independent of time steps. This is indeed true, and a simple re-calculation of the imitation loss (i.e. revised MMD) becomes

$$\psi^*_{\mathcal{C}_\mathcal{H}}(\pi, \pi_E) = \text{MMD}[\mathcal{H}, \rho_\pi, \rho_{\pi_E}] = \|\mu_\pi - \mu_{\pi_E}\|_\mathcal{H}$$
$$\mu_\pi = \textstyle\sum_{t=0}^\infty \gamma^t \int \phi(s,a) dp_\pi(s_t = s, a_t = a)$$
$$\mu_{\pi_E} = \textstyle\sum_{t=0}^\infty \gamma^t \int \phi(s,a) dp_{\pi_E}(s_t = s, a_t = a)$$

Assuming that the lengths of trajectories are uniformly T and the trajectory data are split into time steps each containing N/T instances, i.e. $D_\pi^{(t)} = \{(s_i,a_i)^{(t)}\}_{i=1}^{N/T}$ and $D_{\pi_E}^{(t)} = \{(s_j^E, a_j^E)^{(t)}\}_{j=1}^{N/T}$ for $t = 0, \dots, T$, the empirical estimate becomes

$$\begin{aligned}
&\widehat{\text{MMD}}^2[\mathcal{H}, D_\pi, D_{\pi_E}] \\
&\quad = \tfrac{1}{(N/T)^2}\sum_{t,t'}\gamma^{t+t'}\sum_{i,i'} k((s_i,a_i)^{(t)}, (s_{i'},a_{i'})^{(t')}) \\
&\quad - \tfrac{2}{(N/T)^2}\sum_{t,t'}\gamma^{t+t'}\sum_{i,j} k((s_i,a_i)^{(t)}, (s_j^E,a_j^E)^{(t')}) \\
&\quad + \tfrac{1}{(N/T)^2}\sum_{t,t'}\gamma^{t+t'}\sum_{j,j'} k((s_j^E,a_j^E)^{(t)}, (s_{j'}^E,a_{j'}^E)^{(t')})
\end{aligned}$$

which requires $O(N^2)$ kernel evaluations. The evaluation of the cost function at a state-action pair requires $O(N)$ kernel evaluations, the same computational requirement as in the stationary unnormalized distribution assumption. Our implementation takes the latter approach, as the implementation was much simpler and the performance difference was negligible.

Kernel selection It is straightforward to see that the classical imitation learning in Eqn. (4) corresponds to using the linear kernel $k((s,a),(s',a')) = \vec{\phi}(s,a)^\top\vec{\phi}(s',a')$, which makes $\mu_{\rho_\pi} = E_\pi[\vec{\phi}(s,a)]$, i.e. retaining the first moment of ρ_π in the feature space $\vec{\phi}$. The polynomial kernel of degree d, given by $k((s,a),(s',a')) = (\vec{\phi}(s,a)^\top\vec{\phi}(s',a') + 1)^d$, retains moments up to d-th order. The Gaussian kernel,

$$k((s,a),(s',a')) = \exp(-\sigma^{-1}\|[s,a] - [s',a']\|_2^2)$$

where we simply concatenate the raw encodings of states and actions, is a well-known example of characteristic kernels, which ensures that $\rho_\pi = \rho_{\pi_E}$ if and only if MMD$[\mathcal{H}, \rho_\pi, \rho_{\pi_E}] = 0$.

In our implementation, we used the sum of two Gaussian kernels with different bandwidth parameters $\{\sigma_1, \sigma_2\}$ to span multiple ranges of data points, similar in spirit to (Li, Swersky, and Zemel 2015). However, rather than using the fixed values for the bandwidth parameters, we employed the median heuristics which is theoretically well justified (Schölkopf and Smola 2002; Ramdas et al. 2015). The first bandwidth parameter σ_1 was selected as the median of the pairwise squared-ℓ_2 distances among the data points from the expert policy and from the initial policy, since the initial policy would be vastly different from the expert policy. The second bandwidth parameter σ_2 was selected as the median of the pairwise squared-ℓ_2 distances among the data points only from the expert policy, since the learned policy should be indistinguishable from the expert policy for a successful imitation. Although it remains as a future work to investigate the effectiveness of more sophisticated optimization of kernels, e.g. (Sutherland et al. 2017), we found that these two bandwidth parameters were sufficient for successful imitation in all tasks in our experiments.

Multiple TRPO gradient updates per iteration Algorithm 1 performs single TRPO gradient update on π_θ per iteration, in order to make a direct comparison with GAIL. However, it turns out that we can perform multiple TRPO gradient updates per iteration without re-generating trajectories, allowing for more sample efficient learning.

In (Ho, Gupta, and Ermon 2016), it was shown that each TRPO step that improves π_θ using the trajectories $D_{\pi_0} = \{(s_i, a_i)\}_{i=1}^N$ sampled by some other policy π_0 can be formulated by minimizing the local approximation to the imitation learning objective [1]

$$\begin{aligned}\psi^*_{\mathcal{C}}(\pi_\theta, \pi_E) &= \max_{c\in\mathcal{C}} [J_c(\pi_\theta) - J_c(\pi_E)] \\ &\approx \max_{c\in\mathcal{C}} [J_c(\pi_0) + A^{\pi_{\theta_0}}(\pi_\theta) - J_c(\pi_E)] \triangleq \tilde{\psi}_{\mathcal{C}}(\theta)\end{aligned}$$

where $A^{\pi_0}(\pi_\theta) \triangleq E_{s\sim\rho_{\pi_0}} E_{a\sim\pi_\theta(\cdot|s)}[A^{\pi_0}(s,a)]$. The key step in reusing the trajectories is to re-formulate the objective using importance sampling

$$\begin{aligned}\tilde{\psi}_{\mathcal{C}}(\theta) &= \max_{c\in\mathcal{C}} E_{\rho_{\pi_0}}[c(s,a)] - E_{\rho_{\pi_E}}[c(s,a)] \\ &\qquad + E_{\rho_{\pi_0}}\left[\frac{\pi_\theta(a|s)}{\pi_0(a|s)}(Q^{\pi_0}_c(s,a) - V^{\pi_0}_c(s))\right] \\ &= \max_{c\in\mathcal{C}} E_{\rho_{\pi_0}}[c(s,a)] - E_{\rho_{\pi_E}}[c(s,a)] \\ &\qquad + E_{\rho_{\pi_0}}\left[\left(\frac{\pi_\theta(a|s)}{\pi_0(a|s)} - 1\right) Q^{\pi_0}_c(s,a)\right]\end{aligned}$$

Assuming the cost function class $\mathcal{C}_{\mathcal{H}}$, and introducing notations $\mu_{\pi_0} \triangleq E_{\rho_{\pi_0}}[\phi(s,a)]$, $\mu_{\pi_E} \triangleq E_{\rho_{\pi_E}}[\phi(s,a)]$, and $\mu'_{\pi_\theta} \triangleq E_{\rho_{\pi_0}}\left[\left(\frac{\pi_\theta(a|s)}{\pi_0(a|s)} - 1\right)\phi(s,a)\right]$, we have

$$\tilde{\psi}_{\mathcal{C}}(\theta) = \sup_{\|c\|_{\mathcal{H}}\le 1} \langle \mu_{\pi_0} - \mu_{\pi_E} + \mu'_{\pi_\theta}, c\rangle_{\mathcal{H}}$$

which results in the closed-form optimal cost function

$$c^*(s,a) = \frac{\langle \mu_{\pi_0} - \mu_{\pi_E} + \mu'_{\pi_\theta}, \phi(s,a)\rangle_{\mathcal{H}}}{\|\mu_{\pi_0} - \mu_{\pi_E} + \mu'_{\pi_\theta}\|_{\mathcal{H}}}$$

The empirical estimate of the numerator in the optimal cost function is obtained by

$$\begin{aligned}\hat{c}^*(s,a) \propto\ & \tfrac{1}{N}\textstyle\sum_{i=1}^N k((s_i,a_i),(s,a)) \\ & - \tfrac{1}{M}\textstyle\sum_{j=1}^N k((s^E_j,a^E_j),(s,a)) \\ & + \tfrac{1}{N}\textstyle\sum_{i=1}^N \left(\tfrac{\pi_\theta(a_i|s_i)}{\pi_0(a_i|s_i)} - 1\right) k((s_i,a_i),(s,a))\end{aligned}$$

Thus, we can perform multiple TRPO gradient updates using the trajectories gathered initially at each iteration. We can similarly use the importance sampling to perform multiple updates in GAIL, but this often makes the algorithm unstable as we show in the experiments.

Experiments

We evaluated GMMIL against other imitation learning algorithms on 9 control tasks in the OpenAI Gym (Brockman et al. 2016). The control tasks include 3 low-dimensional classic control tasks (Cartpole, Acrobot, and Mountain Car), and 6 high-dimensional robotic control tasks (Reacher, HalfCheetah, Hopper, Walker, Ant, and Humanoid) that use the MuJoCo simulator (Todorov, Erez, and Tassa 2012). The largest control task, i.e. Humanoid, has 376 observation and 17 action dimensions.

The imitation learning algorithms that we evaluated against are: Behavioral Cloning that uses supervised learning; feature expectation matching (FEM) which corresponds to the cost function class $\mathcal{C}_{\ell_2}$ (Abbeel and Ng 2004); game-theoretic apprenticeship learning (GTAL) which corresponds to the cost function class $\mathcal{C}_\Delta$ (Syed and Schapire 2007); and Generative Adversarial Imitation Learning (GAIL) (Ho and Ermon 2016). All the algorithms used raw state and action encoding as features, and we mostly leveraged the GAIL source code[2] for implementing GMMIL and conducting experiments. For fair comparison, we used the same experimental settings as in (Ho and Ermon 2016), including the exactly same neural network architectures for the policies and the optimizer parameters for TRPO.

Fig. 1 reports the performance of each imitation algorithm, under varying numbers of expert trajectories (the tabularization of the results are provided in the supplementary material). The results show that while algorithms such as behavioral cloning, FEM and GTAL suffered from poor performance, especially in high-dimensional MuJoCo tasks, GAIL and GMMIL attain near-expert performance. In particular, as observed in Reacher, Walker, and Ant tasks, the performance of GAIL significantly degraded when the expert trajectory is scarce. In contrast, GMMIL did not show any significant performance degradation. We suspect that the performance degradation of GAIL is due to a number of issues related to training GANs. First, scarce expert trajectory poses severe label imbalance for training the discriminator. The original implementation of GAIL simply scales the instance weights to mitigate the issue, but it seems that this is not enough. Second, the support overlap of two distributions may become so small that the gradient of the imitation policy almost vanishes. This is known to be the problem associated with using the Jensen-Shannon divergence as the probability metric (Huszár 2015; Nowozin, Cseke, and Tomioka 2016; Arjovsky, Chintala, and Bottou 2017). Although GAIL could be improved via more recent work on GANs that addresses this issue, it is beyond the scope of this paper.

Fig. 2 shows how the discriminator (i.e. cost function) learning rate in GAIL affects learning the imitation policy. We report the performance of GAIL learned with the largest number of expert trajectories under varying learning rate. We can clearly observe that too large or too small learning rates often results in sub-optimal performance. On the other hand, GMMIL does not involve any learning of the cost function, but obtained directly from MMD. This figure shows another practical advantage of using GMMIL.

Fig. 3 compares how quickly GMMIL and GAIL learn to imitate the expert policy, measured in terms of the return attained by the intermediate policies at each iteration. Each iteration amounts to sampling a set of trajectories from

[1]The trust region constraint is not relevant to the discussion.

[2]https://github.com/openai/imitation

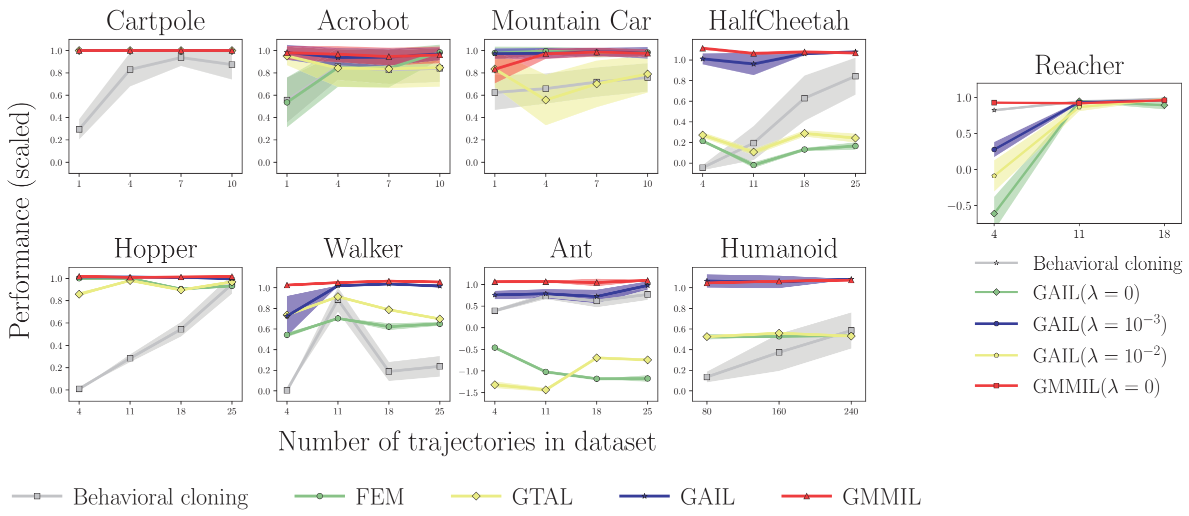

Figure 1: Performance of the learned policy with varying amount of expert data. The x-axis is the number of expert trajectories used for learning and the y-axis is the return of imitation policy, normalized so that the return of the expert policy is 1 and that of the random policy is 0. λ is the penalty weight on the causal entropy of the policy, used in GAIL (Ho and Ermon 2016).

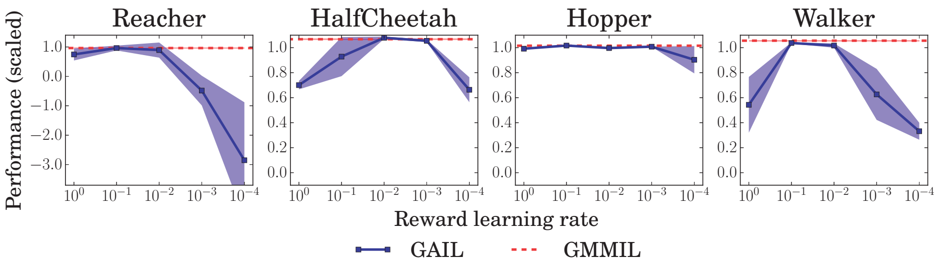

Figure 2: Performance evaluations of GAIL with different learning rates for the cost function. We used the settings with the largest number of expert trajectories, i.e. 18 for reacher; 25 for half cheetah, hopper and walker tasks.

the environment and performing a TRPO gradient update. Again, we show the results with the largest number of expert trajectories, but the results were similar for other settings. We omitted the results from classic control tasks since they were too easy for both algorithms. Note that in most of the tasks, GMMIL converges in fewer iterations than GAIL, exhibiting that GMMIL is more sample-efficient than GAIL requiring fewer samples from the environment. In terms of the computation time, evaluating the witness function on N samples takes $O(N^2)$, but our GPU-based implementation takes overall time on par with GAIL.

Fig. 4 compares the convergence rates of GMMIL and GAIL, both performing 5 TRPO gradient updates per iteration using importance sampling to reuse the trajectories, gathered at the beginning of each iteration. We stress that importance sampling does not in principle guarantee improving sample efficiency, as its effectiveness depends on a number of factors, such as the variance of importance weights. The figure shows the results on higher dimensional tasks without tuning the parameters. GMMIL always showed stable convergence, and it was slower than the single update method in only one of the tasks (i.e. ant). In contrast, GAIL failed to show meaningful training progress in many of the tasks. This also highlights the advantage of using the MMD instead of GAN, since the best cost function is found exactly in each iteration rather than gradually fit using a neural network.

Summary and Future Work

In this paper, we presented GMMIL, a simple yet effective imitation learning algorithm that is essentially a kernelized version of the classical apprenticeship learning algorithm (Abbeel and Ng 2004). Compared to GAIL, our approach provides an alternative reduction of imitation learning to a distribution matching problem, with a much simpler argument. It is shown that the training objective is to minimize the loss measured by the MMD between the occupancy measures of the learned policy and the expert policy. This allows our approach to avoid the hard minimax optimization of GAN inherent to training in GAIL, which results in more robust and sample-efficient imitation learning. As an end result, our approach becomes an imitation learning version of GMMNs (Li, Swersky, and Zemel 2015) and MMD nets (Dziugaite, Roy, and Ghahramani 2015).

Through an extensive set of experiments on high-

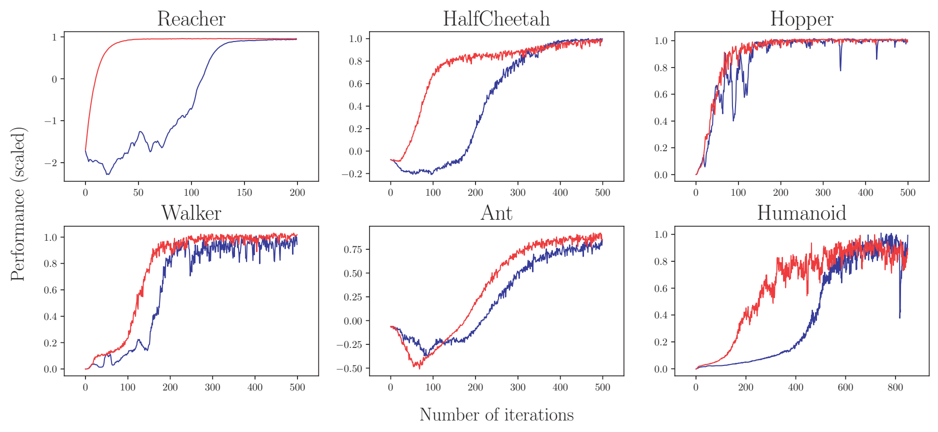

Figure 3: Performance of the learned policy during training iterations, from GMMIL (red) and GAIL (blue). The x-axis is the iteration, and the y-axis is the scaled return of the policy. We used the settings with the largest number of expert trajectories, i.e. 25 for half cheetah, hopper, walker, and ant; 240 for humanoid tasks. These results are from single TRPO gradient update per iteration.

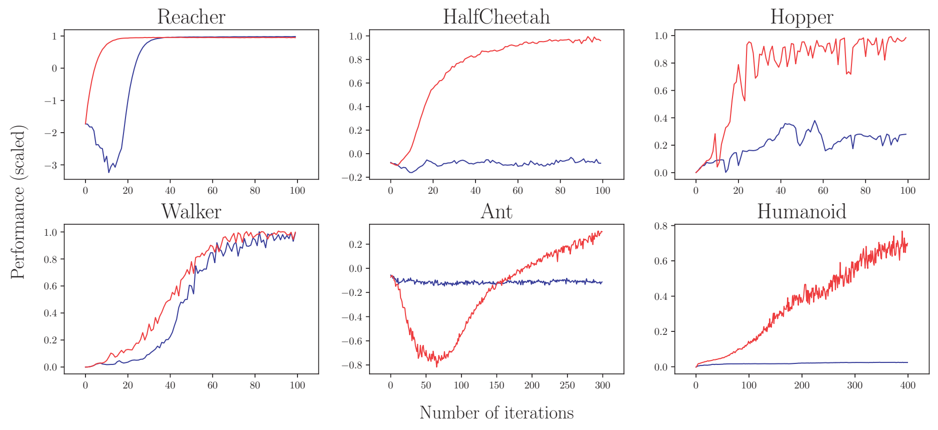

Figure 4: Performance of the learned policy during training iterations using importance sampling to perform 5 TRPO gradient updates per iteration, both for GMMIL (red) and GAIL (blue). Note that the x-axis scales are different from Fig. 3.

dimensional robotic imitation tasks with up to 376 state variables and 17 action variables (i.e. Humanoid), we showed that GMMIL successfully imitates expert policies, even when the expert trajectory was scarcely provided. The returns obtained by the learned policy exhibited lower variances, hinting that using MMD makes the overall optimization much more stable compared to the minimax optimization in GAIL. In addition, we showed that we can naturally reuse the trajectories by importance sampling, allowing for further improving the sample efficiency.

As for the future work, we would like to address many aspects in which our formulation could be improved. First of all, it is well known that the test power of MMD degrades with the dimensionality of the data (Ramdas et al. 2015). Although we did not suffer from this issue in our experiments, this could be true with visual domains. Second, even though TRPO was mostly robust to the variance of the cost function, we still observed some cases where this was somewhat problematic in the last few iterations, both for GAIL and GMMIL. It would be interesting to develop a principled scheme for the cost function that warrants a stable convergence of policy search algorithms.

Acknowledgements

Kee-Eung Kim is supported by IITP/MSIT (2017-0-01778) and DAPA/ADD via KAIST HSVRC. Hyun Soo Park is supported by MnDrive Robotics, Sensing, and Advanced Manufacturing and Oculus/Facebook Research.

References

Abbeel, P., and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning*.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym.

Dziugaite, G. K.; Roy, D. M.; and Ghahramani, Z. 2015. Training generative neural networks via maximum mean discrepancy optimization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, 258–267.

Fukumizu, K.; Bach, F. R.; and Jordan, M. I. 2004. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research* 5:73–99.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13:723–773.

Ho, J., and Ermon, S. 2016. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*.

Ho, J.; Gupta, J.; and Ermon, S. 2016. Model-free imitation learning with policy optimization. In *Proceedings of the 33rd International Conference on Machine Learning*.

Huszár, F. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*.

Li, Y.; Swersky, K.; and Zemel, R. S. 2015. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning*.

Müller, A. 1997. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability* 29:429–443.

Ng, A. Y.; Russell, S. J.; et al. 2000. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*.

Nowozin, S.; Cseke, B.; and Tomioka, R. 2016. f-GAN: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*.

Puterman, M. L. 1994. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Ramdas, A.; Reddi, S. J.; Póczos, B.; Singh, A.; and Wasserman, L. A. 2015. On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

Reed, M., and Simon, B. 1980. *Methods of modern mathematical physics. Functional analysis*, volume 1. Academic.

Russell, S. 1998. Learning agents for uncertain environments. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*.

Sammut, C. 2010. Behavioral cloning. In Sammut, C., and Webb, G. I., eds., *Encyclopedia of Machine Learning*. Springer US. 93–97.

Schölkopf, B., and Smola, A. J. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M. I.; and Moritz, P. 2015. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*.

Smola, A.; Gretton, A.; Song, L.; and Schölkopf, B. 2007. A Hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*. Springer.

Sriperumbudur, B. K.; Gretton, A.; Fukumizu, K.; Lanckriet, G.; and Schölkopf, B. 2008. Injective Hilbert space embeddings of probability measures. In *Proceedings of the 20th Annual Conference on Computational Learning Theory*.

Steinwart, I., and Christmann, A. 2008. *Support vector machines*. Springer Science & Business Media.

Sutherland, D. J.; Tung, H.-Y.; Strathmann, H.; De, S.; Ramdas, A.; Smola, A.; and Gretton, A. 2017. Generate Models and Model Criticism via Optimized Maximum Mean Discrepancy. In *Proceedings of the 5th International Conference on Learning Representations*.

Syed, U., and Schapire, R. E. 2007. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* 5026–5033.