Towards Better Understanding of Kernel-based Imitation Learning

Donghun Lee
dhfromkorea@gmail.com

Abstract

Imitation learning aims to learn a policy from expert demonstrations. Recently in 2018, Generative Moment Matching Imitation Learning, the first kernel-based imitation learning method, was proposed. It is not well studied yet whether its state-of-the-art results can be easily reproduced and to what extent the method is advantageous. To the end, we briefly review the new method and evaluate it on a synthetic task and OpenAI control benchmarks. We present the empirical results in terms of the method's reproducibility, stability, sample-efficiency, sensitivity to data dimensionality, and robustness for noise.

1 Introduction

We consider the problem of recovering an optimal policy from demonstrations. Typical reinforcement learning algorithms learn a policy by optimizing a pre-defined cost function. Since it is difficult to engineer a good cost function, [1, 2] proposed an inverse reinforcement learning (IRL) method that aims to learn a linear cost function the expert demonstration. Though the method became popular, it has two main issues. The first is the dependence on the choice of a feature map which restricts the cost function hypothesis spaces. If the desired behavior is complex and the hypothesis space is restrictive, the method would not produce a good policy. The second main issue is that the method is slow due to its computationally expensive loop of solving a new reinforcement learning problem.

A simpler alternative is called imitation learning (IL) [3] that bypasses the computationally expensive loop. The simplest IL method would arguably be to reduce the main problem to a supervised learning problem. It has been shown the simple reduction is rarely sufficient due to the accumulation of errors. [4] showed a lower bound on the test loss of a simple supervised IL algorithm is $\tilde{O}(\epsilon T^2)$. It tells even with a small classification error at each time step $\epsilon > 0$, the total error will accumulate quadratically in the length of a trajectory T.

The method of interest in this report, Generative Moment Matching Imitation Learning (GMMIL), on the other hand, tries to sit between IRL and IL. First, the method aims to make IRL computationally tractable by not performing the loop in the full scale. Second, its use of kernels may equip the cost function hypothesis spaces with a richer set of choices on the implicit feature maps by the reproducing property. Third, the possible use of *characteric* kernels gives a theoretical guarantee in terms of moment matching of a distribution, making the method better than the simple supervised IL algorithm. Finally, GMMIL maintains a simple formulation and is easier to train than other sophisticated algorithms like the one based on Generative Adversarial Networks [5].

2 Background

In this section, we establish the basic notations and the related concepts that will be used throughout the report. An Markov Decision Process (MDP) is a tuple (S, A, T, C, γ) parameterized by a set of states $s \in S$, a set of actions $a \in A$, a transition probability $T = \mathbb{P}(s'|s, a)$, a cost function

$C : S \times A \to (-\infty, 0]$, and a discount factor $\gamma \in [0, 1)$. A policy $\pi(a|s)$ is a probability distribution over actions conditioned on states $\pi : S \times A \to [0, 1)$. Let $d(s_0)$ be a probability distribution over initial states $\{s_0\} \subset S$. Let $\mathbb{E}_\pi[c(s, a)]$ be a shorthand for $\mathbb{E}_{s_0,a_0,s_1,a_2,\ldots}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t)|\pi]$ where $s_0 \sim d(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \in T(s'|s, a)$. It denotes the expected total discounted cost over a trajectory distributed by π. Let π_E always mean the expert (optimal) policy and π_θ the candidate policy we are to optimize. let $\phi : S \times A \to \mathbb{R}^d$ denote a feature map. Unless otherwise specified, we consider linear cost functions $c \in \mathcal{C} \triangleq \{c \,|\, c = w^\top \phi(s, a), w \in \mathbb{R}^d\}$.

2.1 The Strong Duality of IRL and Occupancy Measure Matching

In this section, we establish the connection between IRL and Occupancy Measure Matching; concretely, the strong duality (refer to [5] for the proof). IRL aims to learn a cost function that consequently will induce the optimal policy whereas IL focuses on mimicking the expert policy [6], both from expert demonstrations denoted by $D_E \triangleq \{\tau \,|\, \tau = (s_0, a_0, s_1, \ldots)\}$. Concretely, IRL is a loop where each iteration solves:

$$RL \circ IRL \triangleq \arg\min_\theta \left\{ \max_{||\phi|| \leq 1} \mathbb{E}_{\pi_\theta}[c(s, a)] - \mathbb{E}_{\pi_E}[c(s, a)] \right\} \tag{1}$$

IRL searches for a cost function that assigns low cost to the expert policy and high cost to the candidate policy (the inner) and then, searches for a policy that minimizes the cost (the outer).

Occupancy Measure and Feature Expectations Define the occupancy measure $\rho : S \times A \to \mathbb{R}$ where $\rho_\pi(s, a) \triangleq \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \mathbb{I}(s_t = s, a_t = a)] = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s|\pi)$ where $\mathbb{I}$ is an indicator function. This can be interpreted as the unnormalized distribution function over (s, a) given π or as an expected number of visits to (s, a). In the subsequent discussion, we assume ρ to be a finite measure so it is treated as a probability measure on S and A (the detailed justification is in [7]). It allows us to write $\mathbb{E}_\pi[c(s, a)] = \sum_{s,a} c(s, a)\rho_\pi(s, a)$. It is known that there is one-to-one correspondence between the policy and its occupancy measure [8] such that $\pi(a|s)$ can be identified with $\frac{\rho(s,a)}{\sum_a' \rho(s,a')}$. Hence, matching the occupancy measure would imply matching the policy that corresponds to it. Utilitizing this property, many IRL approaches use a metric known as *feature expectation* that gives rise to the key equation for this paper:

$$\mu^\pi \triangleq \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t)|\pi] = \mathbb{E}_\pi[\phi(s, a)] = \mathbb{E}_{s,a \sim \rho(s,a)}[\phi(s, a)] \tag{2}$$

2.2 Maximum Mean Discrepancy for Matching the Moments of the Expert Policy

In this section, we establish the need and advantage of computing Maximum Mean Discrepancy with a characteristic kernel mean embedding [9]. By Eq. (2), we notice the feature expectation is indeed the mean embedding of the distribution $\rho_\pi(s, a)$ with the feature map $\phi(s, a)$. The observation motivates the use of a kernel to represent the feature expectation.

Kernel Mean Embedding Let P, Q be Borel probability measures on $X, Y \in (S, A)$. The mean of a feature map $\mu \in \mathcal{H}$ is $\mu_p = [\ldots \mathbb{E}_p[\phi(X)] \ldots]$ and for a positive definite kernel $k : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, it gives $\mathbb{E}_{P,Q}[k(x, y)] = \langle \mu_P, \mu_Q \rangle_\mathcal{H}$. Also we get $\mathbb{E}_p[\phi(X)] = \langle \mu_P, \phi(\cdot) \rangle_\mathcal{H}$. In relation to the main topics of the paper, the last equivalence implies the reproducing property of k where choosing the kernel gives rise to an implicit feature map $\phi : S \times A \to \mathcal{H}$ with μ_ρ^π being an element of an RHKS $\mathcal{H}$ [10]. That is:

$$\mu_\rho \triangleq \mathbb{E}_{X \sim \rho}[k(x, \cdot)] = \mathbb{E}_{X \sim \rho}[\phi(x)] = \int_\mathcal{X} \phi(x) d\rho(x) \tag{3}$$

Maximum Mean Discrepancy (MMD) Let $D_x \triangleq \{x_i\}_{i:1:n}, D_y \triangleq \{y_i\}_{i:1:m}$ be the two sets of samples and suppose we would like to ask whether $P_X = P_Y$. MMD [11] is a distance measure between two distributions $P(X)$ and $Q(Y)$ defined as $\text{MMD}(P, Q, \mathcal{H}) \triangleq ||\mu_p - \mu_q||_\mathcal{H}^2$. We observe the inner risk in Eq. (1) is indeed $\text{MMD}[\rho_{\pi_\theta}, \rho_{\pi_E}, \mathbb{R}^d] = \sup_{||\phi|| \leq 1} \langle \mu^{\pi_\theta} - \mu^{\pi_E}, \phi(s, a) \rangle$. By Cauchy–Schwarz inequality, it is easy to notice the optimum is $||\mu^{\pi_\theta} - \mu^{\pi_E}||_2$ with $\frac{\mu^{\pi_\theta} - \mu^{\pi_E}}{||\mu^{\pi_\theta} - \mu^{\pi_E}||}$ as the maximizer. Finally, define an empirical biased estimate of MMD: $\widehat{MMD}[D_x, D_y, \mathcal{H}] = ||\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\phi(y_j)||^2 = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{i'=1}^{n}k(x_i, x_i') + \frac{1}{m^2}\sum_{j=1}^{m}\sum_{j'=1}^{m}k(y_j, y_j') - \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}k(x_i, y_j)$.

The connection is especially useful for kernels that are injective, known as *characteristic kernels*. i.e. $||\rho_p - \rho_q||_{\mathcal{H}} = 0$ if and only if $P(X) = Q(Y)$. In relation to the main topic, this means if two policies have the same feature expectations then all of moments of the two distributions are equal. Notice we can exchange π and ρ by the one-on-one correspondence between policy and occupancy measure. This motivates the comparison of feature expectations of two policies as a metric [12].

3 Method: Generative Moment Matching Imitation Learning (GMMIL)

GMMIL (Algo. 1) directly follows from our earlier observation that occupancy measure matching induces the same optimal policy as would be given by IRL and that minimizing MMD using a characterisic kernel is essentially the same as minimizing a distance between all moments of a candidate policy and the expert policy. [7] suggested using the sum of two Gaussian kernels that has the all-moment-matching property $k(x, x') = \exp(-\frac{||x-x'||_2^2}{\sigma^2})$ to have $\rho_{\pi_\theta} = \rho_{\pi_E}$ when $\text{MMD}(\cdot) = 0$. Let $D_E = D_{\pi_E} = \{(s_j, a_j)^E\}_{j:1:m}$ and $D_\theta = D_{\pi_\theta} = \{(s_i, a_i)\}_{i:1:n}$. The authors suggested the median heuristics for the bandwidth parameters $\{\sigma_1, \sigma_2\}$ where $\sigma_1 = median(\{||x_\theta^{(0)} - x_E||_2^2 \mid (x_\theta^{(0)}, x_E) \in G\})$ and $G = \{(x_\theta^{(0)}, x_E) \mid x_\theta^{(0)} \in D_\theta, x_E \in D_E\}$. For σ_2, only with D_E.

Algorithm 1: Generative Moment Matching Imitation Learning

```
Input: $\pi_\theta^{(0)}, k, D_E, \epsilon > 0$
set $i = 0$;
while $\widehat{MMD}[D_\theta, D_E] \leq \epsilon$ do
    Sample $D_\theta$ with $\pi_\theta^{(i)}$;
    Compute $\widehat{MMD}[D_\theta, D_E]$ for all $(s, a) \in D_E$;
    Set the optimal cost function $c^*(s, a) = \widehat{MMD}$;
    Update $\theta$ with $c^*$ using TRPO algorithm and set $i = i + 1$;
end
```

4 Experiments

We evaluated GMMIL on 5 different environments. First, we tested the reproducibility of the algorithm on the two OpenAI control tasks for the comparison to the original paper. This was a valuable first step as the poor reproducibility is a big problem in the field of reinforcement learning [13]. Second, we designed a simple navigation task to better understand the behavior of GMMIL under the varying input dimensions and noise levels.

The baseline was a Behavior Cloning (BC) algorithm that employs supervised learning. i.e. it simply tries to learn a map $S \rightarrow A$ with either the mean squared error loss for continuous actions or the cross-entropy loss for discrete actions. Since the original implementation details of GMMIL are not fully available, we based our project off the OpenAI Baseline library like the authors did to make the experimental setup comparable [1]. Due to the computational and time constraint, our experiments were limited to 5 independent trials.

4.1 OpenAI Continuous-state Control Tasks

Can the results be easily reproduced? According to the reported results, the method outperformed the behavior cloning algorithm in terms of sample efficiency, total costs and stability (low variance).

[1] `github.com/openai/baselines/`

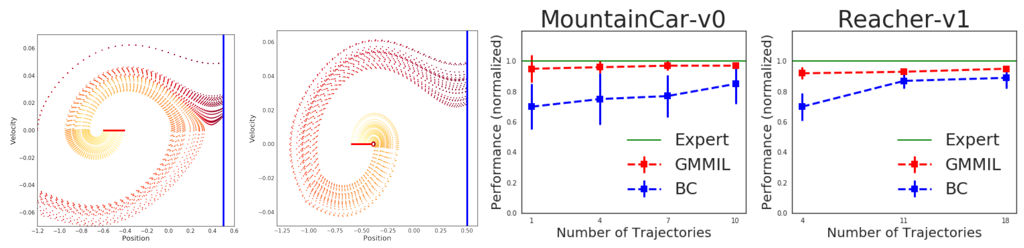

Figure 1: **Reproducibility of the Algorithm on OpenAI tasks** The first two illustrate the trajectories (red) of the expert policy (the first) and the policy learned by GMMIL (the second). We observed the GMMIL policy learned to reach the goal (blue) but generate slightly different trajectory patterns. This is likely due to the stochasticity of the initial state distribution that pushed the agent to unseen situations. The last two plots show the normalized performance as a function of the number of training trajectories. Overall, we observed our result is on par with the original paper's result.

We chose a discrete-action problem, MountainCar-v0 ($dim(S) = 2, dim(A) = 3$), and a continuous-action problem, Reacher-v1 ($dim(S) = 11, dim(A) = 2$). The demonstration data for MountainCar-v0 was generated by the expert policy obtained with Deep Q Learning. For Reacher-v1, we used the demonstration data made available by OpenAI that is known to be obtained using TRPO [14].

4.2 Synthetic Continuous-state Navigation Task

We designed a simple continuous-state discrete-action stochastic MDP where the state is a unit hypercube $S \in [0, 1]^d$. We fixed the action set to the four directions where each action would move the agent by a constant step in the chosen direction. The agent starts at the origin and the goal is fixed on the opposite side of the diagonal which is $\sqrt{d}$ away. The dynamics is stochastic where a uniformly random action is taken instead of the action sampled by the policy with a small probability. The expert policy was hand-crafted by alternating the right and up actions towards the goal. Subsequently, the trajectories were sampled around the diagonal of the state space.

Does the performance degrade as the input dimension or the noise increases? It is known that the MMD suffers the curse of dimensionality [15]. However, the potential downside was not strongly exhibited in the test results of the original paper, even though some tasks have high dimensional inputs (e.g. 376-dimensional state variables in *Humanoid*). We hypothesized the dimensionality issue had been mitigated by the fact a subset of the high-dimensional state variables was essentially nuisance information or that the issue arises for a very large scale problem. To the end, we varied the dimension of the states $d = \{5, 50, 500\}$ and also checked how GMMIL behaves to the noise and the nuisance information. We used an isotropic Gaussian noise $\epsilon \sim N(0, \delta^2 I)$ and varied $\delta = [0.01, 1.0]$ for $d = 50$. For the nuisance factors, we concatenated a near-constant random vector sampled independently of the states with the true state vector for $d = 5$.

Dim.	GMMIL (t)	BC (t)
5	1.11 ± 0.03	1.29 ± 0.13
50	1.17 ± 0.05	1.34 ± 0.17
500	1.36 ± 0.04	1.55 ± 0.22

Noise	GMMIL (t)	BC (t)
Low	1.15 ± 0.05	1.32 ± 0.17
High	2.93 ± 0.16	3.95 ± 1.30
Nuisance	1.27 ± 0.05	1.42 ± 0.16

Table 1: **Sensitivity to Input Dimension and Noise on the Synthetic task**: The left table shows the behavior of GMMIL with varying input dimensions. The reported numbers refer to the average time step at which the agent reached the goal (normalized by the expert's record) with one standard error. We observed GMMIL's performance degrades as the input dimension increases but with a slower rate than BC. The right table shows the robustness to the noise. We observed that both methods are not so robust even for the low noise setting. GMMIL consistently showed a lower variance than BC.

5 Discussion and Conclusion

We empirically evaluated GMMIL to better understand its advantages and disadvantages. We reproduced its superior performance on OpenAI benchmark control tasks, compared to the supervised learning baseline (BC) and the random policy. We confirmed GMMIL is especially superior in the low training data regime which is in line with the original authors' claim that GMMIL is robust for sparse demonstrations. We observed that GMMIL showed a low variance, making it deemed stable. Also we observed that GMMIL is not particularly robust to the noise, but that it performed reasonably well when the nuisance information was added, which can potentially explain the GMMIL's good performance even in some high-dimensional tasks. Moreover, we observed GMMIL's performance degraded as the input dimension increased. For the future work, it would be interesting to study the sensitivity of GMMIL to the choice of kernels including non-characteristic and its hyper-parameters and also the compatibility with policy optimization algorithms other than TRPO.

References

[1] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, pages 663–670, 2000.

[2] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first International Conference on Machine learning*, page 1. ACM, 2004.

[3] 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, 2011.

[4] Matti Kääriäinen. Lower bounds for reductions. In *Atomic Learning Workshop*, 2006.

[5] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pages 4565–4573, 2016.

[6] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668, 2010.

[7] Kee-Eung Kim and Hyun Soo Park. Imitation learning via kernel mean embedding. *AAAI*, 2018.

[8] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[9] Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*, pages 13–31. Springer, 2007.

[10] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[11] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[12] Kenji Fukumizu, Francis R Bach, and Michael I Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5(Jan): 73–99, 2004.

[13] Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *arXiv preprint arXiv:1708.04133*, 2017.

[14] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[15] Aaditya Ramdas, Sashank Jakkam Reddi, Barnabás Póczos, Aarti Singh, and Larry A Wasserman. On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. In *AAAI*, pages 3571–3577, 2015.