

Lecture 5: Imitation Learning

Admin:

- Waitlist: If you were pulled from the waitlist, enroll yourself. Sorry for the confusion about this.
- HW2 was just released.
- Late day policy. Please read the syllabus about this. Small revision: imputed grade will be a *weighted* average, with the same weights used for the final grades.
- Logistical question?

1 Examples from slides

- ALVINN
- TRI
- NVIDIA car
- Human imitation

2 Outline for Today

- Reviewing the MDP
- What is imitation learning, and why are we talking about it in an RL course?
- Behavioral cloning.
- Intuition for the objective
- The main challenge with behavioral cloning: distribution shift
- A few solutions to distribution shift
- Another, subtle problem: implicit smoothing of multi-modal action distributions
- Outlook
- Doing better than imitation learning

3 Review: Markov Decision Process (MDP)

- Agent – Environment loop
- States s_t , actions a_t , rewards $r(s_t, a_t)$.
- Key quantity is the policy $\pi(a | s)$. This is what we're trying to learn.
- *Discounted* objective: $\mathbb{E}_\pi [\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$.
- Dynamics $p(s_{t+1} | s_t, a_t)$ and reward function are unknown.

Questions:

- Q: What is a problem that you'd apply a bandit algorithm to?
- Q: What is a problem that you'd apply an RL algorithm to?
- Q: What is the discount factor?

4 Imitation Learning \neq Reinforcement Learning

Today's class will focus on mimicking previously-seen policies and strategies. It will not focus on finding better strategies. The imitation learning concepts that we discuss today will lay the groundwork for building (actual) RL methods in coming weeks. In fact, we'll see that one of the most common RL methods is a very small changes from the behavioral cloning method that we will introduce today.

- Goal: mimicking behaviors. Recall examples from slides. The goal is *not* to maximize rewards. The imitation learning problem is different from RL.
- Intuitively, we want to learn a policy that visits the same states and actions as an observed expert.
- There are many different types of imitation learning. E.g., do you observe *how* the expert does it (i.e., their actions) or just their states? Can you interact with the environment to try out possible mimicry strategies? Do you try to infer what reward function the expert is optimizing, or just learn their policy?

5 Introducing Behavioral Cloning [?]

This is by far the most common imitation learning algorithm. It doesn't require interactions; it will learn purely from a dataset of states and actions.

- The inputs are a dataset $\{(s, a)\}$ of state-action pairs from an expert. This could be some learned policy. It could be a human expert. E.g., for ALVINN, this was the grad student driving the car. For TRI, it was the human teleoperator.
- The output is a policy $\pi(a | s)$.
- The objective is to maximize the likelihood of the observed actions.

$$\max_{\pi} \mathbb{E}_{(s,a) \sim \mathcal{D}} [\log \pi(a | s)] \quad (1)$$

$$\max_{\theta} \mathbb{E}_{(s,a) \sim \mathcal{D}} [\log \pi_{\theta}(a | s)]. \quad (2)$$

Note that we can write this either as an optimization problem over a policy, or over the parameters of the policy. In practice, we just have samples from this distribution, so we write this objective as

$$\max_{\pi} \frac{1}{|\mathcal{D}|} \sum_{(s,a) \sim \mathcal{D}} [\log \pi(a | s)]. \quad (3)$$

- Q: Would it be useful to talk about parametrizations of the policy?

6 Understanding the Objective

- BC is an example of maximum likelihood. Q: Who has seen maximum likelihood before? How do you think about maximum likelihood?
- Intuition: Your model is some probability distribution over the actions. It's parameters are given by some function (e.g., a neural network) of the input observation. For example, think of a Gaussian distribution, so your neural network is predicting the mean and variance. In BC, your aim is to tune those neural network weights so that the predicted mean and variance result in assigning high likelihood to the observed data points.
- Is this just supervised learning? Yes, but the evaluation will be different.

To build more intuition for this objective, let's consider the special case where your policy is a Gaussian over 1-dimensional actions:

$$\pi_{\theta}(a | s) = \mathcal{N}(a; \mu_{\theta}(s), \sigma = 1) \quad (4)$$

$$= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\mu_{\theta}(s) - a)^2}. \quad (5)$$

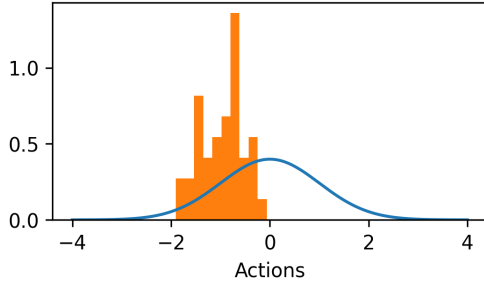


Figure 1: Maximum likelihood involves changing the parameters of a density function (blue) so that it assigns higher weight to the observed data (orange).

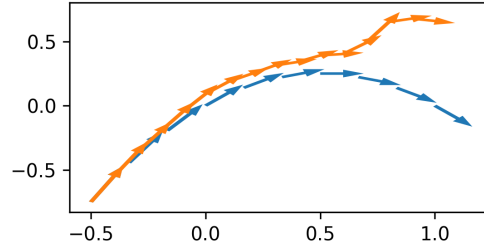


Figure 2: Distribution shift: A policy that achieves low loss on the training might have a much higher loss on different states, including those states that it would actually visit if evaluated in the environment.

We can then write the maximum likelihood objective as

$$\mathbb{E}_{(s,a) \sim \mathcal{D}} [\log \pi(a | s)] = \mathbb{E}_{(s,a) \sim \mathcal{D}} \left[-\frac{1}{2} (\mu_\theta(s) - a)^2 - \log 2 \right]. \quad (6)$$

Note that this is just minimizing the MSE between the predicted mean and the actual action.

7 The main challenge with BC: Distribution Shift

The main challenge with behavioral cloning is distribution shift: while the policy is trained to select actions on states from a certain training set, using that policy to interact with the world may result in visiting states that were unseen during training.

The root cause is a difference between training and evaluation. During training, states are sampled from the dataset (one distribution); during evaluation, we let the policy select actions, which can lead to visiting states (second distribution). While the policy may achieve a low loss on one of these state distributions, it's not guaranteed to achieve a low loss on the other distribution. Q: Ideas for solutions?

8 A Few Solutions

1. Data augmentation [1]. Requires domain knowledge, but can be quite effective. See Fig. 3 for an example. For every state-action pair, we augment the state (e.g., shift it up) and (often) augment the action to compensate (e.g., shift it down). This is often done in autonomous driving applications. The car might drive off the road very infrequently, so it's hard to collect data where the car takes *corrective* actions. This data augmentation exactly serves that purpose.
2. DART [2]. Add noise while collecting the data. This *broadens* the data distribution, increasing the likelihood that the learned policy will continue to visit states that it has seen before in the dataset.

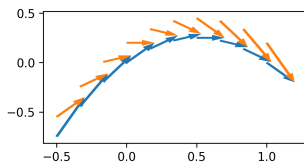


Figure 3: Data augmentation for behavioral cloning

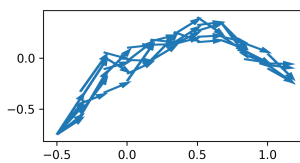


Figure 4: Injecting noise during data collection

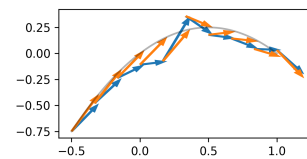


Figure 5: DAGGER

3. DAGGER [3]. This is the most principled approach, it has the strongest theoretical guarantees, but it's pretty hard to use in practice. As visualized in Fig. 5, it entails three steps:
 - (a) Run BC on an initial dataset
 - (b) Collect data with the learned policy
 - (c) Ask an expert for action labels: $(s, a) \rightarrow (s, a^*)$. Go back to step 1.

9 Another Challenge: Implicit Smoothing

A second, subtle challenge with behavioral cloning is that the true distributions over actions are often multi-modal. See, e.g., Fig. 6. In this example, what would happen if you attempted to fit a unimodal policy? (Your learned policy would frequently select actions that are never seen in the training data.)

One natural solution to this is to use more expressive policy classes. This is *not* saying that we're adding more layers to our neural network. Rather, we're saying that the output distribution has to be more expressive. Examples include Gaussian mixture models, discretization + softmax (Google's robots use this), diffusion models (TRI's robots use this).

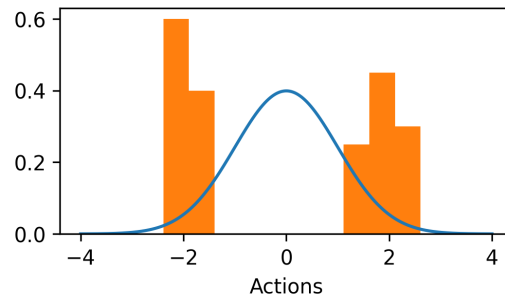


Figure 6: Fitting a multi-modal distribution over actions with a uni-modal policy can result in selecting actions that rarely occur in the data.

Historical note and open question. BC methods went out of style for several years, but are not back in vogue. We can even interpret today's LLMs as doing BC. Is the poor performance typical ascribed to BC caused by (1) compounding errors (the first problem we talked about) or (2) this implicit smoothing? The recent success of discretized and diffusion models hints at the latter.

10 Outlook

Types of learning:

- Imitation learning. We talked about BC today, but this also includes IRL.
- Supervised learning. This intersects with imitation learning.
- Reinforcement learning. This is about going beyond mimicry. It won't necessarily require that we start with data (but it will be able to make use of preexisting data).

References

- [1] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.
- [2] Laskey, M., Lee, J., Fox, R., Dragan, A., and Goldberg, K. (2017). Dart: Noise injection for robust imitation learning. In *Conference on robot learning*, pages 143–156. PMLR.
- [3] Ross, S., Gordon, G., and Bagnell, D. (2011). 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.