Reinforcement Learning CS285 Notes and Homeworks

Andres Espinosa

December 18, 2024

Contents

1 Lecture 1

No notes taken during this lecture

2 Lecture 2

Supervised learning of behaviors and imitation learning

2.1 Part 1

2.1.1 Notation and Context

Given an observation o, our goal is to find a policy $\pi_\theta(a_t|o_t)$ that maps an observation to an action at a time t.

- $\mathbf{o}_t$ - observation
- $\mathbf{s}_t$ - state (note that this and observation tend to get confounded)
- $\mathbf{a}_t$ - action
- $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ - policy which can be deterministic or stochastic. We can generalize policies to be stochastic and then if we want a deterministic decision we can pick the most probable outcome.
- $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ - policy (fully observed)

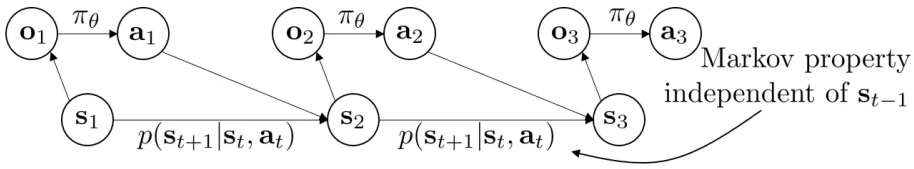

Figure 1: Graphical model showing the transition of observations, actions, and states

2.1.2 Imitation Learning

Imitation learning's goal is to get a policy from looking at observations and actions. Observation and action pairs $(\mathbf{o}_t, \mathbf{a}_t)$ are taken in as training data and fed into some supervised learning algorithm with the goal of outputting a policy $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$. Imitation learning, or behavior cloning, can be very flawed since it is difficult for it to adapt its knowledge to new situations. The supervised learning model will make a mistake that gets compounded as mistakes will cause it to deviate from the original trajectory. Being in a new trajectory increases the likelihood that the model will make more mistakes.
The issue of compounded mistakes is not present in traditional supervised learning since each observation in the data is IID. In a temporal sequence, like a control/reinforcement learning problem, the IID assumption does not hold. This issue can be avoided or mitigated by being smart about the way data is collected and augmented.

2.2 Part 2

2.2.1 Why does behavioral cloning fail?

If we have a policy $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ and a data set $(\mathbf{o}_t, \mathbf{a}_t)$. We can define the probabilities that we pick an action based off of an observation as $p_{data}(\mathbf{o}_t)$ and $p_{\pi_\theta}(\mathbf{o}_t)$. In this case, as we train using supervised

learning, we have the objective function

$$\max_{\theta} \mathbb{E}_{\mathbf{o}_t \ p_{data}(\mathbf{o}_t)}[\log \pi_\theta(\mathbf{a}_t|\mathbf{o}_t)] \tag{1}$$

Since there is a difference in the distribution that the model was trained on and the one that it will experience $p_{data}(\mathbf{o}_t)$ and $p_{\pi_\theta}(\mathbf{o}_t)$, this skews the output. A different objective function can be created using a cost function

$$c(\mathbf{s}_t, \mathbf{a}_t) = \begin{cases} 0 & if \mathbf{a}_t = \pi^*(\mathbf{s}_t) \\ 1 & otherwise \end{cases} \tag{2}$$

and we restructure our objective function to be

$$\text{minimize} \quad \mathbb{E}_{\mathbf{s}_t \ p_{\pi_\theta}(\mathbf{s}_t)}[c(\mathbf{s}_t, \mathbf{a}_t)] \tag{3}$$

We can analyze how effective behavioral cloning by identifying how likely you are at each time step t in a horizon T to make a mistake. If we assume that for any $\mathbf{s} \in D_{train}$, there is a probability $\mathbf{prob}[\pi_\theta(\mathbf{a} \neq \pi^*(\mathbf{s})|\mathbf{s})] \leq \epsilon$. Then if we want to find the expected value of our cost function across all time periods, we can find the below equation. Intuitively, the below equation works by first adding the probability we make a mistake each period to the probability we don't make a mistake on the first step and then every mistake every step after, etc.

$$\mathbb{E}[\sum_t c(\mathbf{s}_t, \mathbf{a}_t)] \leq \epsilon T + (1-\epsilon)(\epsilon(T-1)) \dots \tag{4}$$

This series increases at a rate $O(\epsilon T^2)$ in the worst case. This is a pessimistic worst case, so in order for behavioral cloning to work in practice, the cost functions and problem structure is usually built in a way to make the state easy to recover from.

2.3 Part 3

2.3.1 Data Collection and Augmentation

In order to teach the policy to make corrections well, it is important to collect and augment data in way that shows how to do so.

- Intentionally add mistakes and corrections to collected data
- Augment data to include mistake information.

2.3.2 Model failure types

Some reasons that a model may fail to fit an expert despite good data collection and augmentation processes could be

- Non-Markovian behavior
- Multimodal behavior

In order to model non-markovian behavior, it is possible to use sequence modelling techniques such as LSTMs and transformers to encode history into a current state. In order to model multimodal behavior, it is important to not add merge distributions together because that could cause an incorrect action. Therefore, a mixture of gaussians can be used.
Causal confusion is when a model has too much information and doesn't know exactly what action came from what piece of information.

2.3.3 Model Methods for Improving Failure

Mixture of gaussians:

$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i N(\mu_i, \Sigma_i) \tag{5}$$

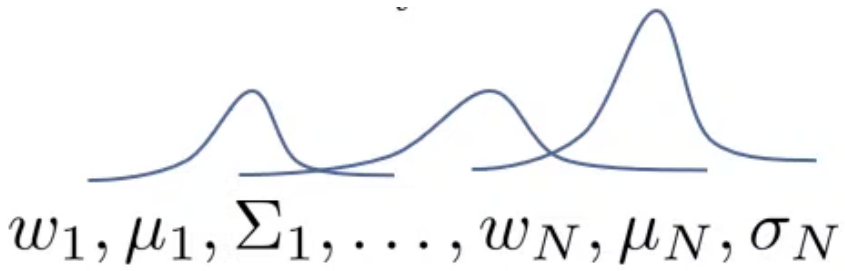

Figure 2: Image of an example gaussian mixture and the corresponding distribution

Latent variable models: Conditional variable autoencoders are latent variable models. The idea is to put in anm additional random vector that will tell it which distribution to output.
Diffusion models: In diffusion models, random noises is added to a noise and then a model is trained to go through the noisy image and predict what the image looked like before the noise was added. This is extended to reinforcement learning by going through each action, adding noise to it and then training a model that attempts to extract what the original action was.
Discretization: Can discretize the actions by binning them into different steps.

2.4 Part 4

2.4.1 Multi-task learning

Learning multiple tasks at the same time can make imitation learning easier. For example, we could create a policy that has multiple outcomes it is trying to reach. The goal of this is to increase the coverage of the states in the training data. Random goals can be placed and data collected over them.

2.5 Part 5

2.5.1 Data Collection Strategy

DAgger algorithm: Dataset Aggregation. The goal with this algorithm is to get $p_{data}(\mathbf{O}_t) = p_{\pi_\theta}(\mathbf{o}_t)$.

- First: train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $D = \{(\mathbf{o}_1, \mathbf{a}_1), \ldots, (\mathbf{o}_N, \mathbf{a}_N)\}$
- run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $D_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
- Ask human to label D_π with actions $\mathbf{a}_t$
- Aggregate: $D \leftarrow D \cup D_\pi$

Basically, the DAgger algorithm has humans retroactively label what the agent should have done and then it adds it to the original dataset.

2.5.2 Cost functions and reward functions

The issue with imitation learning is that humans need to provide data. There are multiple limitations with this as humans may not be the best at providing the data for some kinds of actions. Naturally, we shift away from imitation learning into reinforcement learning to alleviate this issue and have the model itself collect data.