Learning-Based Control & Imitation

Designing, Visualizing and Understanding Deep Neural Networks

CS W182/282A

Instructor: Sergey Levine
UC Berkeley
So far: learning to *predict*

What about learning to *control*?
From *prediction to control*: challenges

$$p(\mathcal{D}) = \prod_i p(y_i|x_i)p(x_i)$$

output $y_1$ does not change $x_2$

this is **very** important, because it allows us to just focus on getting the highest **average** accuracy over the whole dataset

making the wrong choice here is a disaster

making the wrong choice here is perhaps OK
From *prediction* to *control*: challenges

**Ground truth labels:**

“puppy”

**Abstract goals:**

“drive to the grocery store”

> what steering command is that?
From *prediction* to *control*: challenges

- i.i.d. distributed data (each datapoint is independent)
- ground truth supervision
- objective is to predict the right label

These are not *just* issues for control: in many cases, real-world deployment of ML has these same *feedback* issues

*Example:* decisions made by a traffic prediction system might affect the route that people take, which changes traffic

- each decision can change future inputs (not independent)
- supervision may be high-level (e.g., a goal)
- objective is to accomplish the task

We will *build up* toward a *reinforcement learning* system that addresses all of these issues, but we’ll do so one piece at a time...
This distinction will very important later, but is not so important today.
Terminology

- $o_t$ - observation
- $a_t$ - action
- $s_t$ - state
- $\pi_\theta(a_t|o_t)$ - policy
- $\pi_\theta(a_t|s_t)$ - policy (fully observed)

Markov property independent of $s_{t-1}$
Aside: notation

\( s_t \) – state
\( a_t \) – action

\( x_t \) – state
\( u_t \) – action

Richard Bellman

Lev Pontryagin
Imitation Learning

\[ o_t \quad \pi_\theta(a_t | o_t) \quad a_t \]

behavioral cloning
Does it work? No!

Where have we seen this before?
Does it work? Yes!

Video: Bojarski et al. ‘16, NVIDIA
Getting behavioral cloning to work
What is the problem?

the problem: \( p_{\text{data}}(o_t) \neq p_{\pi_\theta}(o_t) \)
What is the problem?

The problem: \( p_{\text{data}}(o_t) \neq p_{\pi_\theta}(o_t) \)

This is a training/test discrepancy: the network always saw true sequences as inputs, but at test-time it gets as input its own (potentially incorrect) predictions.

The problem: this is the same problem!

This is called **distributional shift**, because the input distribution shifts from true strings (at training) to synthetic strings (at test time).

we got unlucky, but now the model is completely confused because the network never saw inputs remotely like this.
Why not use the same solution?

the problem: \( p_{\text{data}}(o_t) \neq p_{\pi_\theta}(o_t) \)

**Before:** scheduled sampling

**Now:** control

we could take the predicted action \( a_t \sim \pi_\theta(a_t|o_t) \) and observe the resulting \( o_{t+1} \)

but this requires interacting with the world!

why?

we don’t know \( p(s_{t+1}|s_t, a_t) \)!
Can we **mitigate** the problem?

the problem: $p_{\text{data}}(o_t) \neq p_{\pi_\theta}(o_t)$

if $\pi_\theta(a_t|o_t)$ is very accurate
maybe $p_{\text{data}}(o_t) \approx p_\theta(o_t)$

Why **might** we fail to fit the expert?

1. **Non-Markovian behavior**
2. **Multimodal behavior**

If we see the same thing twice, we do the same thing twice, regardless of what happened before

Often very unnatural for human demonstrators
How can we use the whole history?

variable number of frames, too many weights
How can we use the whole history?

Typically, LSTM cells work better here.
Why might we fail to fit the expert?

1. Non-Markovian behavior
2. Multimodal behavior

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization

Look up some of these:
- Conditional variational autoencoder
- Normalizing flow/realNVP
- Stein variational gradient descent

\[ \xi \sim \mathcal{N}(0, I) \]
Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization

We’ll learn more about better ways to model multi-modal distributions when we cover generative models later.
Does it work? Yes!

Video: Bojarski et al. ‘16, NVIDIA
Why did that work?

Bojarski et al. ‘16, NVIDIA
Summary

• In principle it should not work
  • Distribution mismatch problem
• Sometimes works well
  • Hacks (e.g. left/right images)
  • Models with memory (i.e., RNNs)
  • Better distribution modeling
  • Generally taking care to get high accuracy
A (perhaps) better approach
Can we make it work more often?

\[ \pi_\theta(a_t | o_t) \]

\[ p_{\pi_\theta}(o_t) \]

\[ p_{\text{data}}(o_t) \]

can we make \( p_{\text{data}}(o_t) = p_{\pi_\theta}(o_t) \)?
Can we make it work more often?

can we make $p_{\text{data}}(o_t) = p_{\pi_\theta}(o_t)$?
idea: instead of being clever about $p_{\pi_\theta}(o_t)$, be clever about $p_{\text{data}}(o_t)$!

**DAgger: Dataset Aggregation**

goal: collect training data from $p_{\pi_\theta}(o_t)$ instead of $p_{\text{data}}(o_t)$
how? just run $\pi_\theta(a_t|o_t)$
but need labels $a_t$!

1. train $\pi_\theta(a_t|o_t)$ from human data $\mathcal{D} = \{o_1, a_1, \ldots, o_N, a_N\}$
2. run $\pi_\theta(a_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, \ldots, o_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $a_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Ross et al. ‘11
DAgger Example

Ross et al. ‘11
What’s the problem?

1. train $\pi_\theta(a_t|o_t)$ from human data $D = \{o_1, a_1, \ldots, o_N, a_N\}$
2. run $\pi_\theta(a_t|o_t)$ to get dataset $D_\pi = \{o_1, \ldots, o_M\}$
3. Ask human to label $D_\pi$ with actions $a_t$
4. Aggregate: $D \leftarrow D \cup D_\pi$
Summary and takeaways

- In principle it should not work
  - Distribution mismatch problem
  - DAgger can address this, but requires costly data collection and labeling

- Sometimes works well
  - Requires a bit of (heuristic) hacks, and very good (high-accuracy) models

My recommendation: try behavioral cloning first, but prepare to be disappointed
Next time

• i.i.d. distributed data (each datapoint is independent)
• ground truth supervision
• objective is to predict the right label

• each decision can change future inputs (not independent)
• supervision may be high-level (e.g., a goal)
• objective is to accomplish the task

We’ll tackle these issues with reinforcement learning