Supervised Learning of Behaviors

CS 294-112: Deep Reinforcement Learning
Sergey Levine
Class Notes

1. Make sure you sign up for Piazza!
2. Homework 1 is now out
   - Milestone due soon – good way to check your TensorFlow knowledge
3. Remember to start forming final project groups
4. Waitlist
Today’s Lecture

1. Definition of sequential decision problems
2. Imitation learning: supervised learning for decision making
   a. Does direct imitation work?
   b. How can we make it work more often?
3. Case studies of recent work in (deep) imitation learning
4. What is missing from imitation learning?
   • Goals:
     • Understand definitions & notation
     • Understand basic imitation learning algorithms
     • Understand their strengths & weaknesses
Terminology & notation

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

- \( s_t \) – state
- \( o_t \) – observation
- \( a_t \) – action

- \( \pi_\theta(a_t|o_t) \) – policy
- \( \pi_\theta(a_t|s_t) \) – policy (fully observed)

Markov property independent of \( s_{t-1} \)
Aside: notation

\[ s_t \text{ – state} \]
\[ a_t \text{ – action} \]

\[ x_t \text{ – state} \]
\[ u_t \text{ – action} \]
Imitation Learning

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

Images: Bojarski et al. '16, NVIDIA
Does it work? No!
Does it work? Yes!

Video: Bojarski et al. ‘16, NVIDIA
Why did that work?
Can we make it work more often?

stability
Learning from a stabilizing controller

$p(s)$, a Gaussian distribution obtained using variant of iterative LQR

(more on this later)
Can we make it work more often?

\begin{align*}
\pi_{\theta}(a_t | o_t) \\
\text{can we make } p_{\text{data}}(o_t) = p_{\pi_{\theta}}(o_t)\
\end{align*}
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: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$
DAgger Example

Ross et al. ‘11
What’s the problem?

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
Can we make it work without more data?

• DAgger addresses the problem of distributional “drift”
• What if our model is so good that it doesn’t drift?
• Need to mimic expert behavior very accurately
• But don’t overfit!
Why might we fail to fit the expert?

1. Non-Markovian behavior
   \[ \pi(\mathbf{a}_t | \mathbf{o}_t) \]
   behavior depends only on current observation

2. Multimodal behavior
   \[ \pi(\mathbf{a}_t | \mathbf{o}_1, \ldots, \mathbf{o}_t) \]
   behavior depends on all past observations

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. Implicit density model
3. Autoregressive discretization
Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Implicit density model
3. Autoregressive discretization
Why might we fail to fit the expert?

1. Output mixture of Gaussians
2. Implicit density model
3. Autoregressive discretization

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

1. Output mixture of Gaussians
2. Implicit density model
3. Autoregressive discretization
Imitation learning: recap

- Often (but not always) insufficient by itself
  - Distribution mismatch problem
- Sometimes works well
  - Hacks (e.g. left/right images)
  - Samples from a stable trajectory distribution
  - Add more **on-policy** data, e.g. using Dagger
  - Better models that fit more accurately
Case study 1: trail following as classification

A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots

Alessandro Giusti¹, Jérôme Guzzi¹, Dan C. Ciresan¹, Fang-Lin He¹, Juan P. Rodriguez¹
Flavio Fontana², Matthias Faessler², Christian Forster²
Jürgen Schmidhuber¹, Gianni Di Caro¹, Davide Scaramuzza², Luca M. Gambardella¹
Case study 2: DAgger & domain adaptation

Learning Transferable Policies for Monocular Reactive MAV Control

Shreyansh Daftry, J. Andrew Bagnell, and Martial Hebert

Robotics Institute, Carnegie Mellon University, Pittsburgh, USA
{daftry, dbagnell, hebert}@ri.cmu.edu
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: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$
Fig. 2. Experiments and Results for (Row-1) Transfer across physical systems from ARDrone to ArduCopter, (Row-2) Transfer across weather conditions from summer to winter and (Row-3) Transfer across environments from Univ. of Zurich to CMU.
Case study 3: Imitation with LSTMs

Learning real manipulation tasks from virtual demonstrations using LSTM

Rouhollah Rahmatizadeh¹, Pooya Abolghasemi¹, Aman Behal² and Ladislau Bölöni¹
Learning Manipulation Trajectories Using Recurrent Neural Networks
Follow-up: adding vision

Vision-Based Multi-Task Manipulation for Inexpensive Robots Using End-To-End Learning from Demonstration

Controlling robot arm by PS Move

Demonstrating multiple tasks while recording:
1) Sequence of images, 2) Robot joint commands

Training neural network

Current image
Task selector
Joint command to robot

Environment change

CNN

Robot autonomously performs the selected task by continuously receiving images of the environment
First we demonstrate different tasks to the robot using Leap Motion or PlayStation Move.
Other topics in imitation learning

• Structured prediction
  
  x: where are you
  y: I’m at work

  ▪ See Mohammad Norouzi’s lecture in November!

• Interaction & active learning

• Inverse reinforcement learning
  ▪ Instead of copying the demonstration, figure out the goal
  ▪ Will be covered later in this course
Imitation learning: what’s the problem?

• Humans need to provide data, which is typically finite
  • Deep learning works best when data is plentiful

• Humans are not good at providing some kinds of actions

• Humans can learn autonomously; can our machines do the same?
  • Unlimited data from own experience
  • Continuous self-improvement
Next time: learning without humans
Terminology & notation

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

- \( s_t \) – state
- \( o_t \) – observation
- \( a_t \) – action

- \( c(s_t, a_t) \) – cost function
- \( r(s_t, a_t) \) – reward function

\[
\min_{a_1, \ldots, a_T} \sum_{t=1}^{T} \log p(s_t, a_t \mid s_{t-1}, a_{t-1}) - \lambda I(s_t, a_t) - r(s_t, a_t)
\]
Aside: notation

\[ s_t - \text{state} \]
\[ a_t - \text{action} \]
\[ r(s, a) - \text{reward function} \]

\[ r(s, a) = -c(x, u) \]

\[ x_t - \text{state} \]
\[ u_t - \text{action} \]
\[ c(x, u) - \text{cost function} \]
Cost/reward functions in theory and practice

\[ r(s, a) = \begin{cases} 
1 & \text{if object at target} \\
0 & \text{otherwise} 
\end{cases} \]

\[ r(s, a) = -w_1\|p_{\text{gripper}}(s) - p_{\text{object}}(s)\|^2 + \\
- w_2\|p_{\text{object}}(s) - p_{\text{target}}(s)\|^2 + \\
- w_3\|a\|^2 \]

\[ r(s, a) = \begin{cases} 
1 & \text{if walker is running} \\
0 & \text{otherwise} 
\end{cases} \]

\[ r(s, a) = w_1 v(s) + \\
w_2\delta(|\theta_{\text{torso}}(s)| < \epsilon) + \\
w_3\delta(h_{\text{torso}}(s) \geq h) \]
A cost function for imitation?

$$r(s, a) = \log p(a = \pi^*(s) | s)$$

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: $D \leftarrow D \cup D_\pi$

Ross et al. ‘11
The trouble with cost & reward functions

More on this later...

Sim-to-Real Robot Learning from Pixels with Progressive Nets

Andrei A. Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, Raia Hadsell

Google DeepMind
London, UK

{andrei.rusu, matej.vecerik, tcr, heess, razp, raia}@google.com

Rewards are given automatically by tracking the colored target
A note about terminology...

the “R” word

A bit of history...

Reinforcement learning (the problem statement)

\[
\max_{\pi} \sum_{t=1}^{T} E[r(s_t, a_t)]
\]

\[s_{t+1} \sim p(s_{t+1} | s_t, a_t)\]

Reinforcement learning (the method)

Without using the model

\[s_{t+1} \sim p(s_{t+1} | s_t, a_t)\]