DAgger: Taming Covariate Shift with No Regret

Sanjiban Choudhury
Recap in 60 seconds!
SUPERVISED LEARNING

1. Get Data
   - Input (s)
   - Output (a)

2. Train Policy
   - $\pi : s \rightarrow a$

3. Deploy!
Behavior Cloning crashes into a wall

Train ≠ Test
Why did the robot crash?
Why did the robot crash?

Error: \( \epsilon \)

No training data
Error: 1.0

Demonstrations
Why did the robot crash?

Train ≠ Test

No training data
Error: 1.0

Demonstrations

Error: ε
Activity!
Think-Pair-Share!

Think (30 sec): What are ALL the different sources of train-test mismatch?

Pair: Find a partner

Share (45 sec): Partners exchange ideas
Case 1: Data changes over time
Case 2: Data changes with robot behavior
Case 3: Data changes adversarially (game)
Challenge:
Don’t know the test distribution upfront

Learner  Collect Data
Introducing an *interactive* expert!
To know the distribution, you need a learner
To train a learner, you need a distribution
Let’s work out a simple algorithm
Be stable

Slowly change predictions
DAGGER

Episode IV

A NEW HOPE
DAGGER: A meta-algorithm for imitation learning

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning

Stéphane Ross  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA  
stephaneross@cmu.edu

Geoffrey J. Gordon  
Machine Learning Department  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA  
ggordon@cs.cmu.edu

J. Andrew Bagnell  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA  
dbagnell@ri.cmu.edu
DAgger: Iteration 0

Data

Human drives

Policy $\pi_0$
DAgger: Iteration 1

Data

Old Data

Robot $\pi_0$ drives

Human corrects!

AGGREGATE DATA

Policy $\pi_1$
DAgger: Iteration 2

Robot $\pi_1$ drives

[Ross et al'11]

AGGREGATE DATA
DAgger: Iteration 1

Robot $\pi_N$ drives

After many iterations .... we are able to drive like a human!
Dagger (Dataset Aggregation)

- Aggregated Training Distribution
- Test Distribution
- Human Distribution
Supervised Approach after 20 Laps of Training

Original results from DAGGER!
DAGGER after 20 Iterations (20 Laps of Training)

Original results from DAGGER!
DAgger here reacts dynamically to an untrained obstacle
A brief history of DAGGER

NLP folks (Hal Daume III in particular) were first to identify feedback effects in sequential prediction tasks.
DAGGER is a foundation

Imitation under uncertainty

SAIL
ExPLORE
STROLL
Counterfactual Teaching

Agnostic
SysID
DaaD
Model learning

DPI
LOLS
NRPI
Reinforcement Learning

DAEQUIL
AGGREVATE(D)
Imitation learning

EIL
HG-DAGGER
SHIV
Query efficient imitation learning

DAGGER
Many cool applications of DAGGER in robotics

Lee et al, Learning quadrupedal locomotion over challenging terrain (2020)

Choudhury et al, Data Driven Planning via Imitation Learning (2018)

Chen et al, Learning by Cheating (2020)

Pan et al, Imitation learning for agile autonomous driving (2019)
But why does *aggregating* data work?
Think of the human ... 

... as an adversary
Interactive Learning

Learner → Adversary → Learner
The Imitation Game

Learner

Initialize policy

Update policy

Adversary

Chooses loss

π₁ [policy]

l₁(.) [loss]

π₂

l₂(.)
Let’s formalize!
How do we design algorithms that are no-regret?

\[
\text{Regret} = \sum_{t=1}^{T} l_t(\pi_t) - \min_{\pi^*} \sum_{t=1}^{T} l_t(\pi^*)
\]

(Learner) (Best in hindsight)
At every round $t$, choose the best expert in hindsight

$$
\pi_t = \arg \min_{\pi} \sum_{i=1}^{t-1} l_i(\pi)
$$

(lowest total loss)
$\sum l_t$  

<table>
<thead>
<tr>
<th>Expert 1</th>
<th>$l_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expert 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expert 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5</td>
</tr>
</tbody>
</table>

Avg. Regret: $\mathbf{- -}$
<table>
<thead>
<tr>
<th>Expert</th>
<th>(l_1)</th>
<th>(l_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert 1</td>
<td>1.0</td>
<td>0.5</td>
</tr>
<tr>
<td>Expert 2</td>
<td>0.2</td>
<td>0.5</td>
</tr>
<tr>
<td>Expert 3</td>
<td>0.5</td>
<td>0.2</td>
</tr>
</tbody>
</table>

\[ \sum l_t \]

Avg. Regret: 0.80
\[ \sum l_t \begin{array}{c|c|c|c}
\text{Expert 1} & l_1 & l_2 & l_3 \\
1.5 & 1.0 & 0.5 & 0.5 \\
\end{array} \]

\[ \begin{array}{c|c|c|c}
\text{Expert 2} & l_1 & l_2 & l_3 \\
0.7 & 0.2 & 0.5 & 1.0 \\
\end{array} \]

\[ \begin{array}{c|c|c|c}
\text{Expert 3} & l_1 & l_2 & l_3 \\
0.7 & 0.5 & 0.2 & 0.2 \\
\end{array} \]

Avg. Regret: 0.40
<table>
<thead>
<tr>
<th>$\sum l_t$</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert 1</td>
<td>2.0</td>
<td>1.0</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>Expert 2</td>
<td>1.7</td>
<td>0.2</td>
<td>0.5</td>
<td>1.0</td>
</tr>
<tr>
<td>Expert 3</td>
<td>0.9</td>
<td>0.5</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Avg. Regret: 0.53
$$\sum l_t$$ | \(l_1\) | \(l_2\) | \(l_3\) | \(l_4\) | \(l_5\) \\
---|---|---|---|---|---
Expert 1 | 3.0 | 1.0 | 0.5 | 0.5 | 1.0 | 0.5 \\
Expert 2 | 1.9 | 0.2 | 0.5 | 1.0 | 0.2 | 1.0 \\
Expert 3 | 1.4 | 0.5 | 0.2 | 0.2 | 0.5 | 0.2 \\

Avg. Regret: 0.40
<table>
<thead>
<tr>
<th>$l_t$</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
<th>$l_5$</th>
<th>$l_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert 1</td>
<td>3.5</td>
<td>1.0</td>
<td>0.5</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
</tr>
<tr>
<td>Expert 2</td>
<td>2.9</td>
<td>0.2</td>
<td>0.5</td>
<td>1.0</td>
<td>0.2</td>
<td>1.0</td>
</tr>
<tr>
<td>Expert 3</td>
<td>1.6</td>
<td>0.5</td>
<td>0.2</td>
<td>0.2</td>
<td>0.5</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Avg. Regret: 0.32
$$\sum l_t$$

<table>
<thead>
<tr>
<th>Expert 1</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
<th>$l_5$</th>
<th>$l_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5</td>
<td>1.0</td>
<td>0.5</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expert 2</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
<th>$l_5$</th>
<th>$l_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4</td>
<td>0.2</td>
<td>0.5</td>
<td>1.0</td>
<td>0.2</td>
<td>1.0</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expert 3</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
<th>$l_5$</th>
<th>$l_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.8</td>
<td>0.5</td>
<td>0.2</td>
<td>0.2</td>
<td>0.5</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Avg. Regret: **0.26**
FTL appears to be no regret ...

Can you think of an example that breaks FTL?
Loss = 0.75  Avg. Regret = 0.5

Choose $\pi^1$  Choose $\pi^2$
Loss = 1.0  Avg. Regret = 0.5

Choose $\pi^1$

Choose $\pi^2$
Choose $\pi^1$  
Choose $\pi^2$ 

Loss = 1.0  
Avg. Regret = 0.5
Loss

Choose $\pi_1$

Loss = 1.0  Avg. Regret = 0.5

Choose $\pi_2$
Loss = 1.0  Avg. Regret = 0.5

Choose $\pi^1$  Choose $\pi^2$
Be stable

Slowly change predictions

Achieve no-regret
Follow the Regularized Leader

\[ \pi_t = \arg \min_{\pi} \sum_{i=1}^{t-1} l_i(\pi) + \eta_t R(\pi) \]

Strong regularization!
Loss = 0.5  Avg. Regret = 0.25

Choose $\pi^1$

Choose $\pi^2$
Loss = 0.6  Avg. Regret = 0.17

Choose $\pi^1$  

Choose $\pi^2$
Loss = 0.78
Avg. Regret = 0.21

Choose $\pi^1$

Choose $\pi^2$
Loss = 0.78  Avg. Regret = 0.2

Choose $\pi^1$

Choose $\pi^2$
To know the distribution, you need a learner
To train a learner, you need a distribution

The Imitation Game

Learner
Initialize policy
Update policy

Adversary
Chooses loss

Data + Old Data
AGGREGATE DATA

Robot $a_t$ drives
Human corrects!

Policy $\pi_1$ [Ross et al'11]