Interactive Imitation Learning
Announcement

This Thursday:

lecture will start at 9:50am
and office hour will end at 11:15am
Recap

Offline IL
Recap

Offline IL

Ground truth reward \( r(s, a) \in [0,1] \) is unknown; assume expert is a near optimal policy \( \pi^* \)
Recap

Offline IL

Ground truth reward \( r(s, a) \in [0,1] \) is unknown; assume expert is a near optimal policy \( \pi^* \)

We have a dataset \( \mathcal{D} = (s_i^*, a_i^*)^M_{i=1} \), \( s_i^* \sim \mu^{\pi^*} \), \( a_i^* \sim \pi^*(\cdot|s_i^*) \)
Recap

The Behavior Cloning algorithm:

Choose regression (for continuous action) or classification loss $\ell(\pi(s), a)$, and perform SL:

$$\hat{\pi} = \min_{\pi \in \Pi} \sum_{i=1}^{M} \ell(\pi^*(s^*_i), a^*_i)$$
Recap

The Behavior Cloning algorithm:

Choose regression (for continuous action) or classification loss $\ell(\pi(s), a)$, and perform SL:

$$\hat{\pi} = \min_{\pi \in \Pi} \sum_{i=1}^{M} \ell(\pi^*(s_i^*), a_i^*)$$

Pros:

*Simple, flexible*, and often just works reasonably well
Recap

The Behavior Cloning algorithm:

Choose regression (for continuous action) or classification loss $\ell(\pi(s), a)$, and perform SL:

$$\hat{\pi} = \min_{\pi \in \Pi} \sum_{i=1}^{M} \ell(\pi^*(s_i^*), a_i^*)$$

**Pros:**

*Simple, flexible*, and often just works reasonably well

**Cons:**

*Distribution shift issue*: $\hat{\pi}$ does not know what to do outside expert’s states
Question for today:

How to mitigate the distribution shift issue?
Solution:

Interactive Imitation Learning Setting
Solution:

Interactive Imitation Learning Setting

Key assumption:
we can query expert $\pi^*$ at any time and any state during training
Solution:

Interactive Imitation Learning Setting

Key assumption:
we can query expert $\pi^*$ at any time and any state during training

(Recall that previously we only had an offline dataset $\mathcal{D} = (s^*_i, a^*_i)_{i=1}^M \sim d_\mu^{\pi^*}$)
Outline for today:

1. The DAgger (Data Aggregation) Algorithm

2. Analysis of DAgger: a quick intro to Online Learning
Recall the Main Problem from Behavior Cloning:

- Learned Policy
- Expert’s trajectory
- No training data of “recovery” behavior
Intuitive solution: **Interaction**

Use interaction to collect data where learned policy goes.
General Idea: Iterative Interactive Approach

Collect Data through Interaction → New Data

Update Policy → Updated Policy

All DAgger slides credit: Drew Bagnell, Stephane Ross, Arun Venktraman
DAgger: Dataset Aggregation

0th iteration

Expert Demonstrates Task

Dataset

Supervised Learning

1st policy $\pi_1$
DAgger: Dataset Aggregation
1st iteration

Execute $\pi_1$ and Query Expert

Steering from expert

[Ross11a]
DAgger: Dataset Aggregation
1st iteration

Execute $\pi_1$ and Query Expert

Steering from expert

New Data

[Ross11a]
DAgger: Dataset Aggregation
1st iteration

Execute $\pi_1$ and Query Expert

Steering from expert

New Data

States from the learned policy

[Ross11a]
DAgger: Dataset Aggregation
1st iteration

Execute $\pi_1$ and Query Expert

Steering from expert

New Data

All previous data

[Ross11a]
DAgger: Dataset Aggregation
1st iteration

Execute $\pi_1$ and Query Expert

Steering from expert

New policy $\pi_2$

Supervised Learning

Aggregate Dataset

All previous data

New Data

[Ross11a]
DAgger: Dataset Aggregation
2nd iteration

Execute $\pi_2$ and Query Expert

New Data

Aggregate Dataset

All previous data

Supervised Learning

New policy $\pi_3$

Steering from expert

[Ross11a]
DAgger: Dataset Aggregation

$n$th iteration

Execute $\pi_{n-1}$ and Query Expert

Steering from expert

New policy $\pi_n$

Supervised Learning

Aggregate Dataset

New Data

All previous data

[Ross11a]
Success!

Set!

Penalty time!!
Success!

Features

Set!
Penalty time!!
Success!

Features

Set!

Penalty time!!
Average Falls/Lap

![Graph showing the average falls per lap against the number of training data points, with DAgger, SMILE(0.1), and Supervised lines compared. The graph indicates improved performance as the number of training data points increases.]
FPS: 24
Attempt: 1 of 1
AgentLinear
Selected Actions:

RIGHT

SPEED
FPS: 24
Attempt: 1 of 1
AgentLinear
Selected Actions:

RIGHT

SPEED
FPS: 24
Attempt: 1 of 1
AgentLinear
Selected Actions:

RIGHT
SPEED
More fun than Video Games…
More fun than Video Games…
More fun than Video Games…
Forms of the Interactive Experts

Interactive Expert is expensive, especially when the expert is human…

But expert does not have to be human…
Forms of the Interactive Experts

Interactive Expert is expensive, especially when the expert is human…

But expert does not have to be human…

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Fig. 4: The AutoRally car and the test track.
Forms of the Interactive Experts

Interactive Expert is expensive, especially when the expert is human…

But expert does not have to be human…

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Goal: learn a racing control policy that maps from data on cheap on-board sensors (raw-pixel imagine) to low-level control (steer and throttle)

Fig. 4: The AutoRally car and the test track.
Forms of the Interactive Experts

Interactive Expert is expensive, especially when the expert is human...

But expert does not have to be human...

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Goal: learn a racing control policy that maps from data on cheap on-board sensors (raw-pixel imagine) to low-level control (steer and throttle)

Fig. 4: The AutoRally car and the test track.

(a) raw image
Forms of the Interactive Experts

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]
Forms of the Interactive Experts

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Their Setup:
At Training, we have expensive sensors for accurate state estimation and we have computation resources for MPC (i.e., high-frequency replanning)
Forms of the Interactive Experts

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Their Setup:
At Training, we have expensive sensors for accurate state estimation and we have computation resources for MPC (i.e., high-frequency replanning)

The MPC is the expert in this case!
Forms of the Interactive Experts

Example: high-speed off-road driving
[Pan et al, RSS 18, Best System Paper]

Their Setup:
At Training, we have expensive sensors for accurate state estimation and we have computation resources for MPC (i.e., high-frequency replanning)

The MPC is the expert in this case!
Distribution Shift: Example

Initial state

\( s_0 \rightarrow s_1 \rightarrow s_2 \)

Transitions:
- \( a_1 \) from \( s_0 \) to \( s_1 \)
- \( a_2 \) from \( s_0 \) to \( s_2 \)
- \( a_1 \) from \( s_1 \) to \( s_2 \)
- \( a_1 \) from \( s_2 \) to \( s_1 \)
- \( a_2 \) from \( s_2 \) to \( s_0 \)
Distribution Shift: Example

Initial state
Distribution Shift: Example

Initial state

\[
d^{\pi}_{s_0}(s_0) = 1 - \gamma, \quad d^{\pi}_{s_0}(s_1) = \gamma, \quad d^{\pi}_{s_0}(s_2) = 0
\]
Distribution Shift: Example

Initial state

\[ r(s_1) = 1 \]

\[ d_{s_0}^\pi(s_0) = 1 - \gamma, \quad d_{s_0}^\pi(s_1) = \gamma, \quad d_{s_0}^\pi(s_2) = 0 \]
Distribution Shift: Example

Initial state

\[ r(s_1) = 1 \]

\[ V^{\pi^*}_{s_0} = \frac{\gamma}{1 - \gamma} \]

\[ d^\pi_{s_0}(s_0) = 1 - \gamma, \ d^\pi_{s_0}(s_1) = \gamma, \ d^\pi_{s_0}(s_2) = 0 \]
Distribution Shift: Example

Assume SL returned such policy \( \hat{\pi} \)

\[
\hat{\pi}(s_0) = \begin{cases} 
  a_1 & \text{w/ prob } 1 - \epsilon/(1 - \gamma) \\
  a_2 & \text{w/ prob } \epsilon/(1 - \gamma) 
\end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2
\]

Initial state

\[
\begin{align*}
  r(s_1) &= 1 \\
  d_{s_0}^\pi(s_0) &= 1 - \gamma, \quad d_{s_0}^\pi(s_1) = \gamma, \quad d_{s_0}^\pi(s_2) = 0 \\
  V_{s_0}^\pi &= \frac{\gamma}{1 - \gamma}
\end{align*}
\]
Distribution Shift: Example

Assume SL returned such policy $\hat{\pi}$

$$\hat{\pi}(s_0) = \begin{cases} 
    a_1 & \text{w/ prob } 1 - \epsilon/(1 - \gamma) \\
    a_2 & \text{w/ prob } \epsilon/(1 - \gamma)
\end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2$$

We will have good supervised learning error:

$$\mathbb{E}_{s \sim d_{s_0}^\pi} \mathbb{E}_{a \sim \hat{\pi}(\cdot|s)} 1(a \neq \pi^*(s)) = \epsilon$$

$$d_{s_0}^\pi(s_0) = 1 - \gamma, \quad d_{s_0}^\pi(s_1) = \gamma, \quad d_{s_0}^\pi(s_2) = 0$$

$$V_{s_0}^\pi = \frac{\gamma}{1 - \gamma}$$

Initial state

$r(s_1) = 1$

$\begin{array}{c}
    S_0 \\
    \xrightarrow{a_2} S_1 \\
    \xrightarrow{a_1} S_2 \\
    \xrightarrow{a_2} S_1 \\
    \xrightarrow{a_1} S_2 \\
    \xrightarrow{a_2} S_2
\end{array}$
Distribution Shift: Example

Assume SL returned such policy \( \hat{\pi} \)

\[
\hat{\pi}(s_0) = \begin{cases} 
    a_1 \text{ w/ prob } 1 - \epsilon/(1 - \gamma), \\
    a_2 \text{ w/ prob } \epsilon/(1 - \gamma) 
\end{cases}, \quad \hat{\pi}(s_1) = a_2, \quad \hat{\pi}(s_2) = a_2
\]

We will have good supervised learning error:

\[
\mathbb{E}_{s \sim d_{\pi}^{s_0}} \mathbb{E}_{a \sim \hat{\pi}(.|s)} 1(a \neq \pi^*(s)) = \epsilon
\]

But we have quadratic error in performance:

\[
V_{\hat{\pi}}^{s_0} = V_{\pi}^{s_0} - \frac{\epsilon \gamma}{(1 - \gamma)^2}
\]

Initial state

\[
d_{s_0}^\pi(s_0) = 1 - \gamma, \quad d_{s_0}^\pi(s_1) = \gamma, \quad d_{s_0}^\pi(s_2) = 0
\]

\[
V_{s_0}^{\pi} = \frac{\gamma}{1 - \gamma}
\]
Distribution Shift: Example

Assume SL returned such policy $\hat{\pi}$

\[
\hat{\pi}(s_0) = \begin{cases} 
    a_1 \text{ w/ prob } 1 - \epsilon/(1 - \gamma) \\
    a_2 \text{ w/ prob } \epsilon/(1 - \gamma)
\end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2
\]

\[
r(s_1) = 1
\]

Initial state

\[
d^{\pi}_s(s_0) = 1 - \gamma, \quad d^{\pi}_s(s_1) = \gamma, \quad d^{\pi}_s(s_2) = 0
\]

\[
V^{\pi}_s = \frac{\gamma}{1 - \gamma}
\]
Distribution Shift: Example

Assume SL returned such policy \( \hat{\pi} \)

\[
\hat{\pi}(s_0) = \begin{cases} 
    a_1 \text{ w/ prob } 1 - \epsilon/(1 - \gamma) \\
    a_2 \text{ w/ prob } \epsilon/(1 - \gamma)
\end{cases}
\]

\( \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2 \)

Why DAgger can fix this problem?

Initial state

\[
d_{s_0}^\ast(s_0) = 1 - \gamma, \quad d_{s_0}^\ast(s_1) = \gamma, \quad d_{s_0}^\ast(s_2) = 0
\]

\[
V_{s_0}^\ast = \frac{\gamma}{1 - \gamma}
\]
**Distribution Shift: Example**

Assume SL returned such policy \( \hat{\pi} \)

\[
\hat{\pi}(s_0) = \begin{cases} 
    a_1 & \text{w/ prob } 1 - \frac{\epsilon}{1 - \gamma} \\
    a_2 & \text{w/ prob } \frac{\epsilon}{1 - \gamma}
\end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2
\]

Why DAgger can fix this problem?

\( \hat{\pi} \) will visit \( s_2 \), and we collect \( \pi^*(s_2) = a_1 \)

**Initial state**

\[
\begin{align*}
    V_{s_0}^{\pi^*} &= \frac{\gamma}{1 - \gamma} \\
    d_{s_0}^{\pi^*}(s_0) &= 1 - \gamma, \ d_{s_0}^{\pi^*}(s_1) = \gamma, \ d_{s_0}^{\pi^*}(s_2) = 0
\end{align*}
\]
Distribution Shift: Example

Assume SL returned such policy $\hat{\pi}$

$$\hat{\pi}(s_0) = \begin{cases} a_1 \text{ w/ prob } 1 - \epsilon/(1 - \gamma) \\ a_2 \text{ w/ prob } \epsilon/(1 - \gamma) \end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2$$

Why DAgger can fix this problem?

$\hat{\pi}$ will visit $s_2$, and we collect $\pi^*(s_2) = a_1$

By data aggregation, our new dataset will contain $(s_2, a_1)$ pairs

\[d_{s_0}^\pi(s_0) = 1 - \gamma, \quad d_{s_0}^\pi(s_1) = \gamma, \quad d_{s_0}^\pi(s_2) = 0\]

\[V_{s_0}^\pi = \frac{\gamma}{1 - \gamma}\]
Distribution Shift: Example

Assume SL returned such policy $\hat{\pi}$

$$\hat{\pi}(s_0) = \begin{cases} a_1 & \text{w/ prob } 1 - \epsilon/(1 - \gamma) \\ a_2 & \text{w/ prob } \epsilon/(1 - \gamma) \end{cases}, \quad \hat{\pi}(s_1) = a_2, \hat{\pi}(s_2) = a_2$$

Why DAgger can fix this problem?

$\hat{\pi}$ will visit $s_2$, and we collect $\pi^*(s_2) = a_1$

By data aggregation, our new dataset will contain $(s_2, a_1)$ pairs

Thus, our new learned policy will know what to do at $s_2$
Outline for today:

1. The DAgger (Data Aggregation) Algorithm

2. Quick intro on Online Learning
Online Learning

Learner

convex Decision set $\Theta$

Adversary

[Vovk92, Warmuth94, Freund97, Zinkevich03, Kalai05, Hazan06, Kakade08]
Online Learning

[Learner picks a decision $\theta_0$]

convex Decision set $\Theta$

...
Online Learning

Learner picks a decision $\theta_0$

Adversary picks a loss $\ell_0 : \Theta \rightarrow \mathbb{R}$

convex Decision set $\Theta$
Online Learning

Learner

Adversary

convex Decision set $\Theta$

Learner picks a decision $\theta_0$

Adversary picks a loss $\ell_0 : \Theta \rightarrow \mathbb{R}$

Learner picks a new decision $\theta_1$

...
Online Learning

Adversary

Learner picks a decision $\theta_0$

Adversary picks a loss $\ell_0 : \Theta \rightarrow \mathbb{R}$

Learner picks a new decision $\theta_1$

Adversary picks a loss $\ell_1 : \Theta \rightarrow \mathbb{R}$

convex Decision set $\Theta$

[Vovk92, Warmuth94, Freund97, Zinkevich03, Kalai05, Hazan06, Kakade08]
Online Learning

Learner picks a decision $\theta_0$

Adversary picks a loss $\ell_0 : \Theta \rightarrow \mathbb{R}$

Learner picks a new decision $\theta_1$

Adversary picks a loss $\ell_1 : \Theta \rightarrow \mathbb{R}$

...$

\text{Regret} = \sum_{t=0}^{T-1} \ell_t(\theta_t) - \min_{\theta \in \Theta} \sum_{t=0}^{T-1} \ell_t(\theta)$
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:

1. Learner first picks $\theta_t \in \text{Ball} \subset \mathbb{R}^d$
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:

1. Learner first picks $\theta_t \in \text{Ball} \subset \mathbb{R}^d$

2. Adversary then picks $x_t \in \mathcal{X} \subset \mathbb{R}^d$, $y_t \in [a, b]$
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:

1. Learner first picks $\theta_t \in \text{Ball} \subset \mathbb{R}^d$

2. Adversary \textbf{then} picks $x_t \in \mathcal{X} \subset \mathbb{R}^d, y_t \in [a, b]$

3. Learner suffers loss $\ell_t(\theta_t) = (\theta_t^T x_t - y_t)^2$
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:

1. Learner first picks $\theta_t \in \text{Ball} \subseteq \mathbb{R}^d$

2. Adversary then picks $x_t \in \mathcal{X} \subseteq \mathbb{R}^d, y_t \in [a, b]$

3. Learner suffers loss $\ell_t(\theta_t) = (\theta_t^T x_t - y_t)^2$

Learner has to make decision $\theta_t$ based on history up to $t - 1$, while adversary could pick $(x_t, y_t)$ even after seeing $\theta_t$
Example: online linear regression

Can we perform linear regression in online fashion with non i.i.d (or even adversary) data?

Every iteration $t$:

1. Learner first picks $\theta_t \in \text{Ball} \subset \mathbb{R}^d$

2. Adversary then picks $x_t \in \mathcal{X} \subset \mathbb{R}^d, y_t \in [a, b]$

3. Learner suffers loss $\mathcal{L}_t(\theta_t) = (\theta_t^T x_t - y_t)^2$

Learner has to make decision $\theta_t$ based on history up to $t - 1$, while adversary could pick $(x_t, y_t)$ even after seeing $\theta_t$

Adversary seems too powerful…
Example: online linear regression

BUT, a very intuitive algorithm actually achieves no-regret property:
Example: online linear regression

BUT, a very intuitive algorithm actually achieves no-regret property:

Every iteration $t$:

1. Learner first picks $\theta_t$ that minimizes the aggregated loss

$$\theta_t = \arg \min_{\theta \in \text{Ball}} \sum_{i=0}^{t-1} (\theta^T x_i - y_i)^2 + \lambda \|\theta\|_2^2$$
Example: online linear regression

BUT, a very intuitive algorithm actually achieves no-regret property:

Every iteration $t$:

1. Learner first picks $\theta_t$ that minimizes the aggregated loss

$$\theta_t = \arg \min_{\theta \in \text{Ball}} \sum_{i=0}^{t-1} (\theta^T x_i - y_i)^2 + \lambda \|\theta\|_2^2$$

This is called Follow-the-Regularized-Leader (FTRL), and it achieves no-regret property:
Example: online linear regression

BUT, a very intuitive algorithm actually achieves no-regret property:

Every iteration $t$:

1. Learner first picks $\theta_t$ that minimizes the aggregated loss

$$\theta_t = \arg\min_{\theta \in \text{Ball}} \sum_{i=0}^{t-1} (\theta^T x_i - y_i)^2 + \lambda \|\theta\|_2^2$$

This is called Follow-the-Regularized-Leader (FTRL), and it achieves no-regret property:

$$\sum_{i=0}^{T-1} \ell_i(\theta_t) - \min_{\theta \in \text{Ball}} \sum_{i=0}^{T-1} \ell_i(\theta) = O\left(\frac{1}{\sqrt{T}}\right)$$
Generally, Follow-the-Regularized-Leader is no-regret

At time step $t$, learner has seen $\ell_0, \ldots \ell_{t-1}$, which new decision she could pick?

\[
\text{FTL: } \theta_t = \min_{\theta \in \Theta} \sum_{i=0}^{t-1} \ell_i(\theta) + \lambda R(\theta)
\]
Generally, Follow-the-Regularized-Leader is no-regret

At time step $t$, learner has seen $\mathcal{L}_0, \ldots \mathcal{L}_{t-1}$, which new decision she could pick?

**FTL:** $\theta_t = \min_{\theta \in \Theta} \sum_{i=0}^{t-1} \mathcal{L}_i(\theta) + \lambda R(\theta)$

**Theorem (FTL) (optional):** if $\Theta$ is convex, and $\mathcal{L}_t$ is convex for all $t$, and $R(\theta)$ is strongly convex, then for regret of FTL, we have:

$$\frac{1}{T} \left[ \sum_{t=0}^{T-1} \mathcal{L}_t(\theta_t) - \min_{\theta \in \Theta} \sum_{t=0}^{T-1} \mathcal{L}_t(\theta) \right] = O \left( \frac{1}{\sqrt{T}} \right)$$
Any questions about no-regret online learning?

Online learning is a very rich research area — details are out of scope

Key message:

Learner has to make a decision before Adversary picks a loss function, yet it is possible to do as well as the best decision in hindsight if we had access to all the loss functions beforehand
DAgger Revisit

At iteration $t$:

- **Steering from expert**
- **New policy** $\pi_n$
- **Supervised Learning**
- **Aggregate Dataset**

**New Data**

- **All previous data**

---

**Steering from expert**

**New policy** $\pi_n$

**Supervised Learning**
DAgger Revisit

At iteration $t$:

New Data

Aggregate Dataset

All previous data

Supervised Learning

New policy $\pi_n$

Steering from expert

...
DAgger Revisit

At iteration $t$:

$$
\ell_t(\pi) = \sum_{i=1}^{m} \| \pi(s^i) - \pi^*(s^i) \|_2^2
$$

New Data

Steering from expert

New policy $\pi_n$

Aggregate Dataset

All previous data

Supervised Learning
DAgger Revisit

At iteration $t$:

$$\ell_t(\pi) = \sum_{i=1}^{m} \| \pi(s^i) - \pi^*(s^i) \|_2^2$$

New Data

Aggregate Dataset

All previous data

Supervised Learning

New policy $\pi_n$

Steering from expert
DAGGER Revisit

At iteration $t$:

$\ell_t(\pi) = \sum_{i=1}^{m} \left\| \pi(s^i) - \pi^*(s^i) \right\|^2_2$

New Data

Aggregate Dataset

All previous data

$\sum_{i=0}^{i-1} \ell_i(\pi)$

Steering from expert

New policy $\pi_n$

Supervised Learning
DAgger Revisit

At iteration t:

\[ \ell_t(\pi) = \sum_{i=1}^{m} \| \pi(s^i) - \pi^*(s^i) \|_2^2 \]

New Data

Aggregate Dataset

All previous data

\[ \sum_{i=0}^{i-1} \ell_i(\pi) \]

Supervised Learning

Steering from expert

New policy \( \pi_n \)
DAgger Revisit

At iteration $t$:

New Data

\[
\ell_t(\pi) = \sum_{i=1}^{m} \| \pi(s^i) - \pi^*(s^i) \|_2^2
\]

Aggregate Dataset

All previous data

\[
\sum_{i=0}^{t-1} \ell_i(\pi) + \lambda R(\pi)
\]

New policy $\pi^*_n$

\[
\pi^*_t = \arg \min_{\pi} \sum_{i=0}^{t-1} \ell_i(\pi)
\]

Steering from expert

Supervised Learning
DAgger Revisit

At iteration $t$:

New Data

$$\ell_t(\pi) = \sum_{i=1}^{m} \| \pi(s^i) - \pi^*(s^i) \|_2^2$$

Aggregate Dataset

$\bar{\pi}_t = \arg\min_{\pi} \sum_{i=0}^{t-1} \ell_i(\pi) + \lambda R(\pi)$

Supervised Learning

Data Aggregation = Follow-the-Regularized-Leader Online Learner
Summary for Today

1. The DAgger algorithm

Initialize $\pi^0$, and dataset $\mathcal{D} = \emptyset$

For $t = 0 \rightarrow T - 1$:

1. W/ $\pi^t$, generate dataset $\mathcal{D}^t = \{s_i, a_i^*\}, s_i \sim d^{\pi^t}_\mu, a_i^* = \pi^*(s_i)$

2. Data aggregation: $\mathcal{D} = \mathcal{D} + \mathcal{D}^t$

3. Update policy via Supervised-Learning: $\pi^{t+1} = \text{SL} \left( \mathcal{D} \right)$
Summary for Today

1. The DAgger algorithm

Initialize $\pi^0$, and dataset $\mathcal{D} = \emptyset$

For $t = 0 \rightarrow T - 1$:

1. W/ $\pi^t$, generate dataset $\mathcal{D}^t = \{s_i, a^*_i\}, s_i \sim d^\pi_{\mu}, a^*_i = \pi^*(s_i)$

2. Data aggregation: $\mathcal{D} = \mathcal{D} + \mathcal{D}^t$

3. Update policy via Supervised-Learning: $\pi^{t+1} = \text{SL} \left( \mathcal{D} \right)$

2. We can see that DAgger is essentially an online-learning algorithm (FTRL)