Reduction of Imitation Learning to No-Regret Online Learning

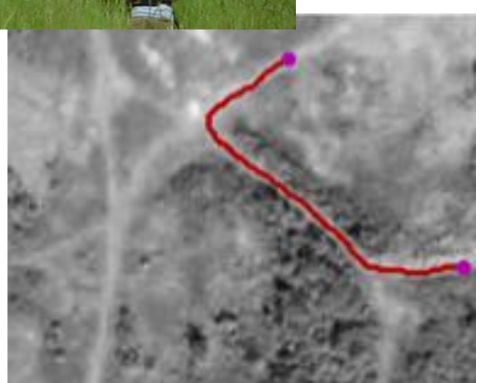
Stephane Ross

Joint work with Drew Bagnell & Geoff Gordon

Carnegie Mellon
THE ROBOTICS INSTITUTE

Imitation Learning

Expert Demonstrations → Machine Learning Algorithm → Policy



Imitation Learning

- Many successes:
  - Legged locomotion [Ratliff 06]
  - Outdoor navigation [Silver 08]
  - Helicopter flight [Abbeel 07]
  - Car driving [Pomerleau 89]
  - etc...

Example Scenario

Learning to drive from demonstrations

Input:

Camera Image

Policy

Output:

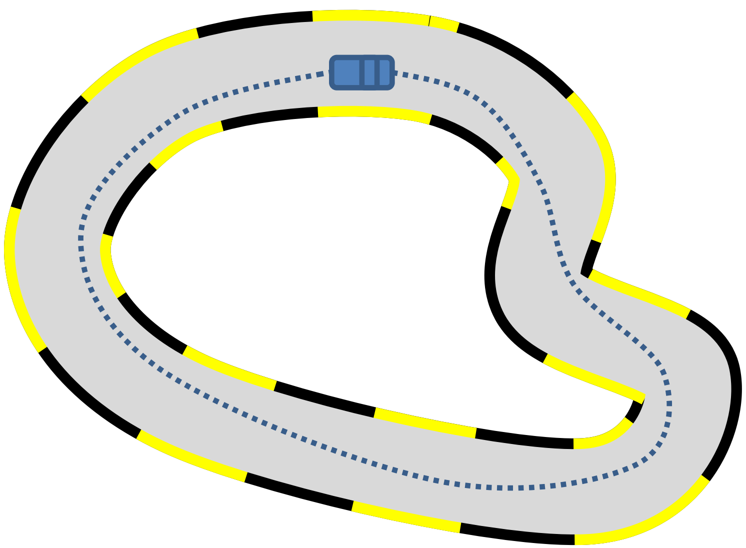
Steering in [-1,1]

Hard left turn

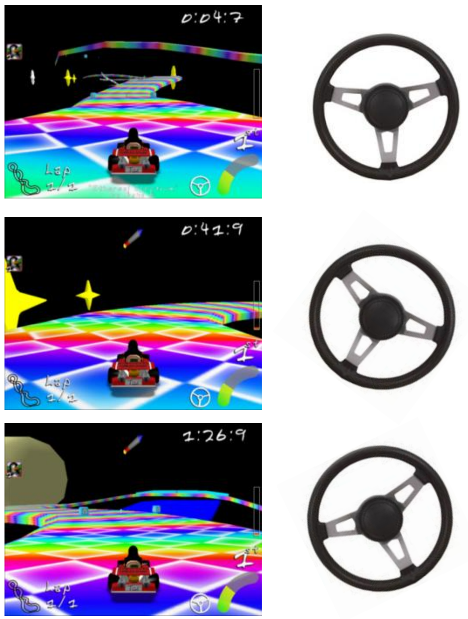
Hard right turn

Supervised Training Procedure

Expert Trajectories



Dataset



Learned Policy: $\hat{\pi}_{\mathbf{sup}} = \mathbf{arg\,min}_{\pi\in\Pi} \mathbf{E}_{\mathbf{s\sim D(\pi^*)}}[\ell(\pi,\mathbf{s},\pi^*(\mathbf{s}))]$

Poor Performance in Practice

Mistakes Grows Quadratically in T!

[Ross 2010]

$$\mathbf{J}(\hat{\pi}_{\mathbf{sup}}) \leq \mathbf{T}^2 \varepsilon$$

Exp. # of mistakes
over T steps

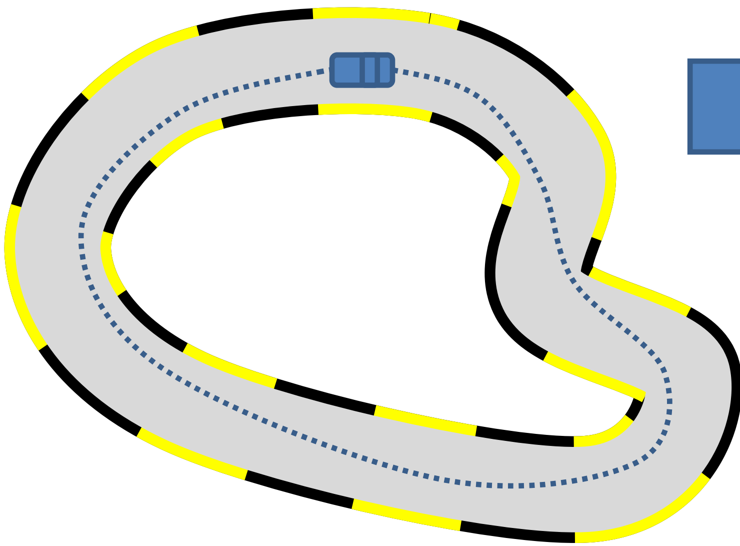
time steps

Avg. loss on D(π*)

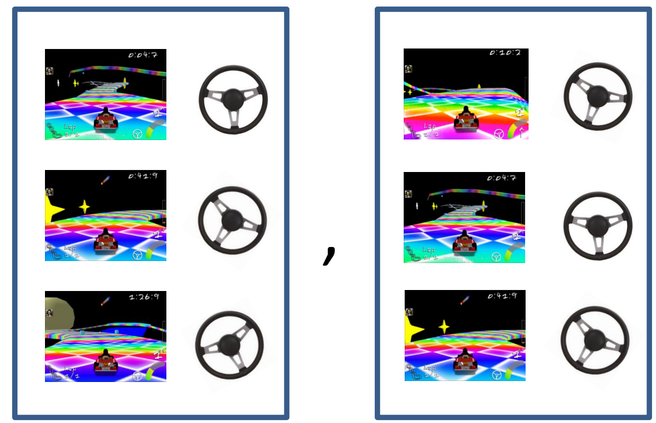
Reason: Doesn't learn how to recover from errors!

Reduction-Based Approach & Analysis

Hard Learning Problem



Easier Related Problem(s)



, , …

Performance: f(ε)

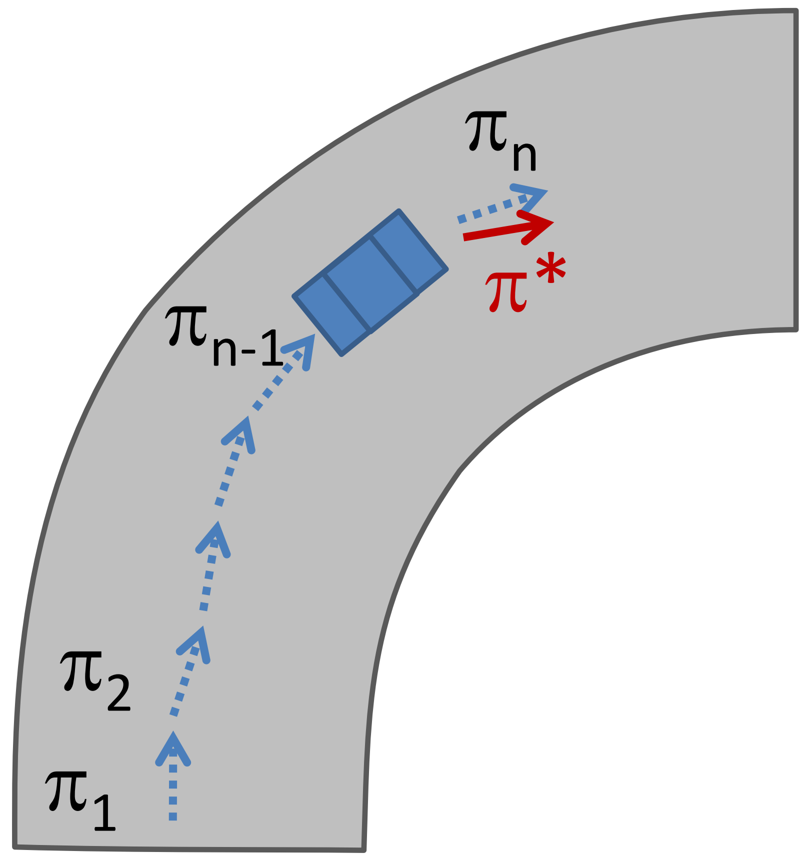
Performance: ε

Example: Cost-sensitive Multiclass classification to Binary classification [Beygelzimer 2005]

Previous Work: Forward Training

[Ross 2010]

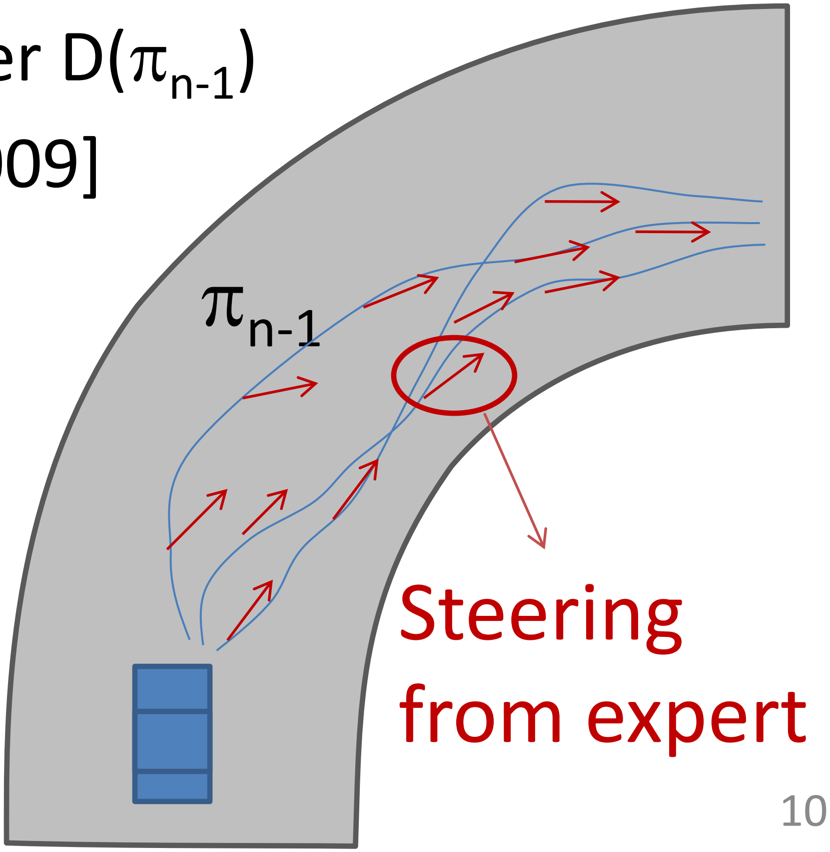
- Sequentially learn one policy/step
- # mistakes grows linearly:
  - $J(\pi_{1:T}) \leq T\varepsilon$
- **Impractical if T large**



Previous Work: SMILe

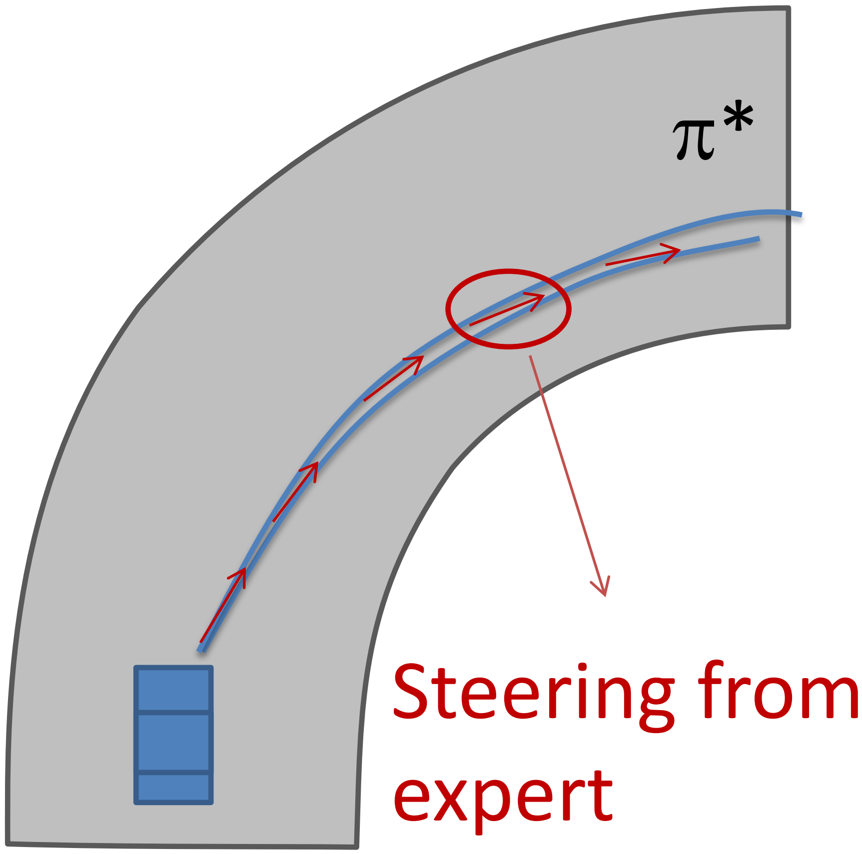
[Ross 2010]

- Learn stochastic policy, changing policy slowly
  - $\pi_n = \pi_{n-1} + \alpha_n(\pi'_n - \pi^*)$
  - π'_n trained to mimic π^* under $D(\pi_{n-1})$
  - Similar to SEARN [Daume 2009]
- Near-linear bound:
  - $J(\pi) \leq O(T\log(T)\varepsilon + 1)$
- **Stochasticity undesirable**



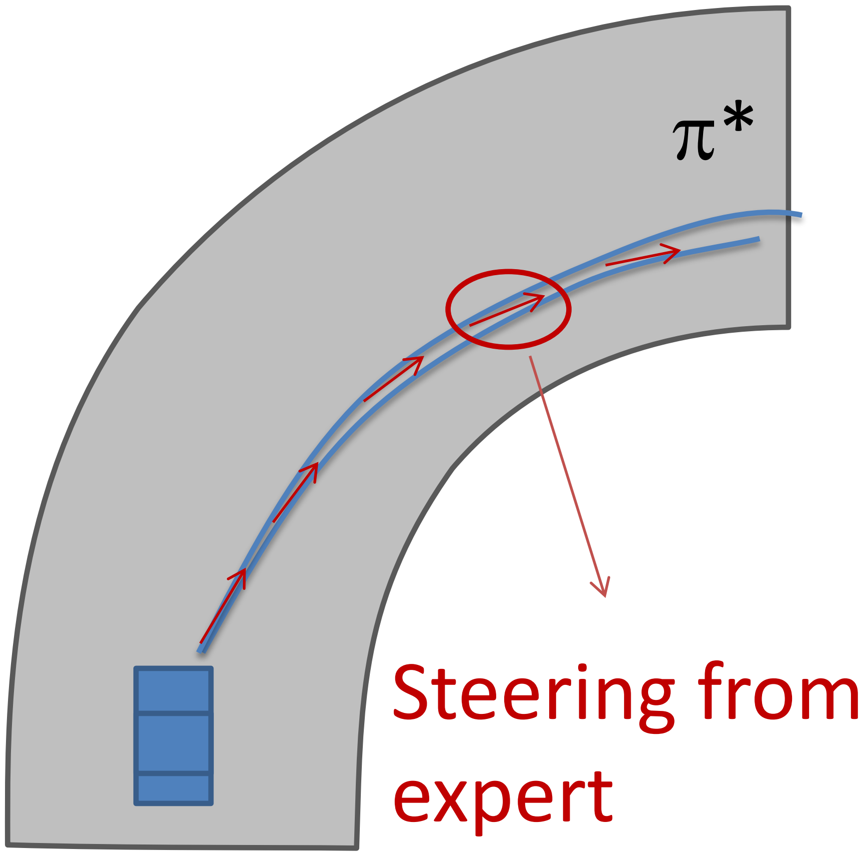
DAgger: Dataset Aggregation

- Collect trajectories with expert π^*



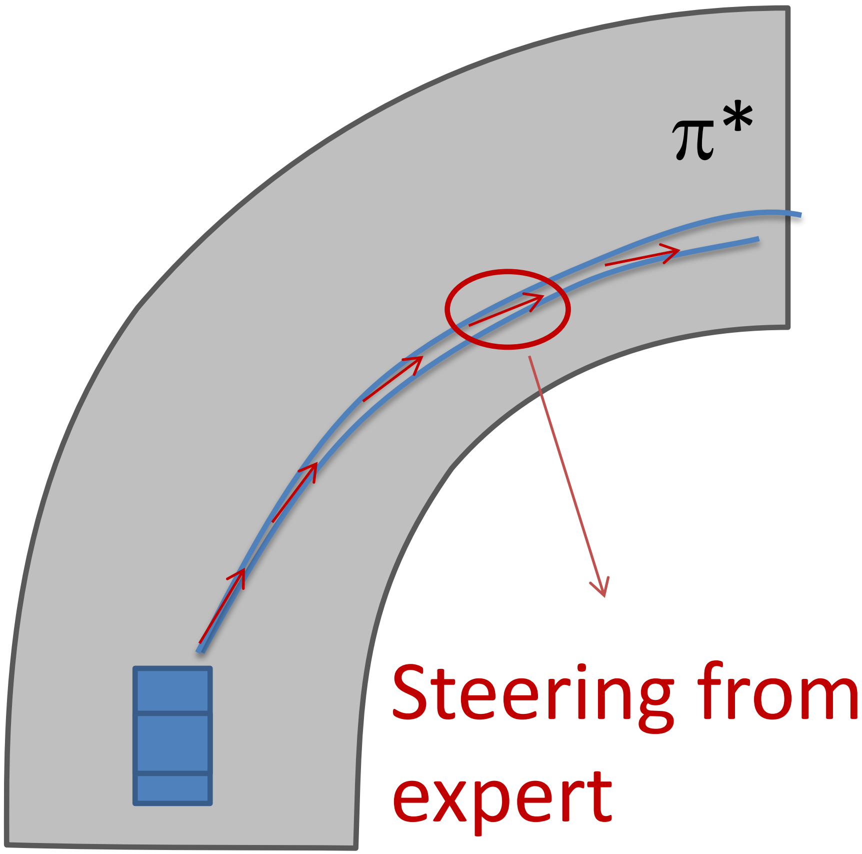
DAgger: Dataset Aggregation

- Collect trajectories with expert π^*
- Dataset $D_0 = \{(s, \pi^*(s))\}$



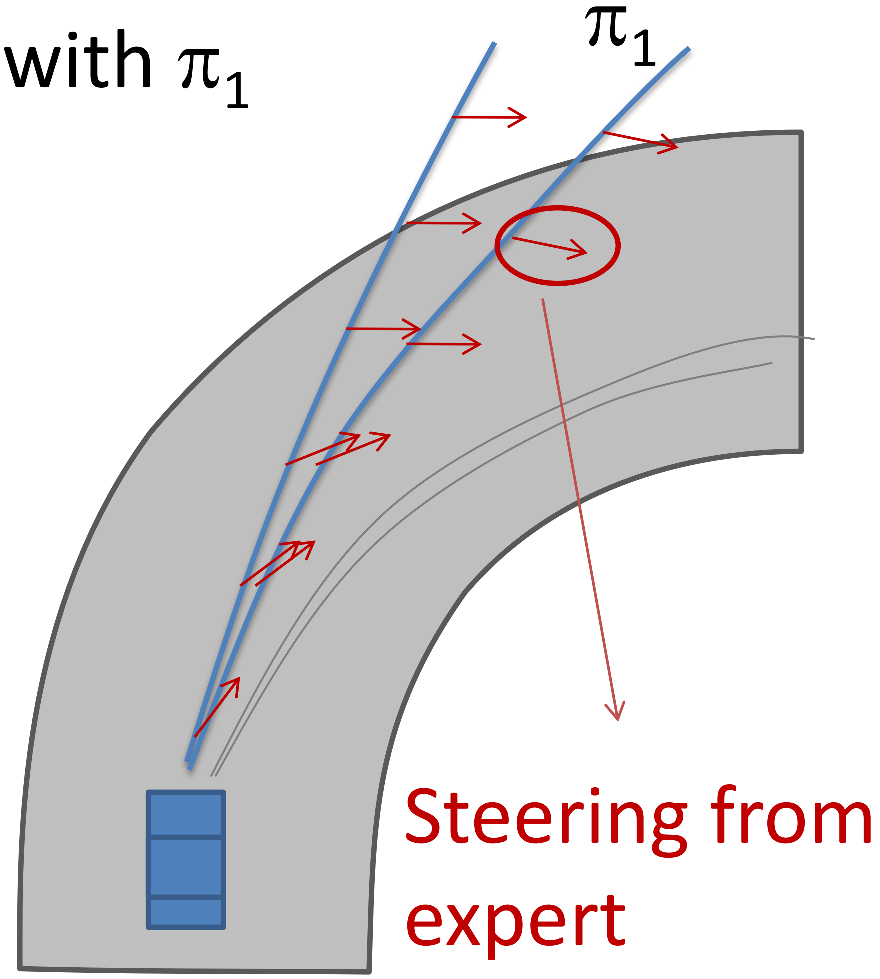
DAgger: Dataset Aggregation

- Collect trajectories with expert π^*
- Dataset $D_0 = \{(s, \pi^*(s))\}$
- Train π_1 on D_0



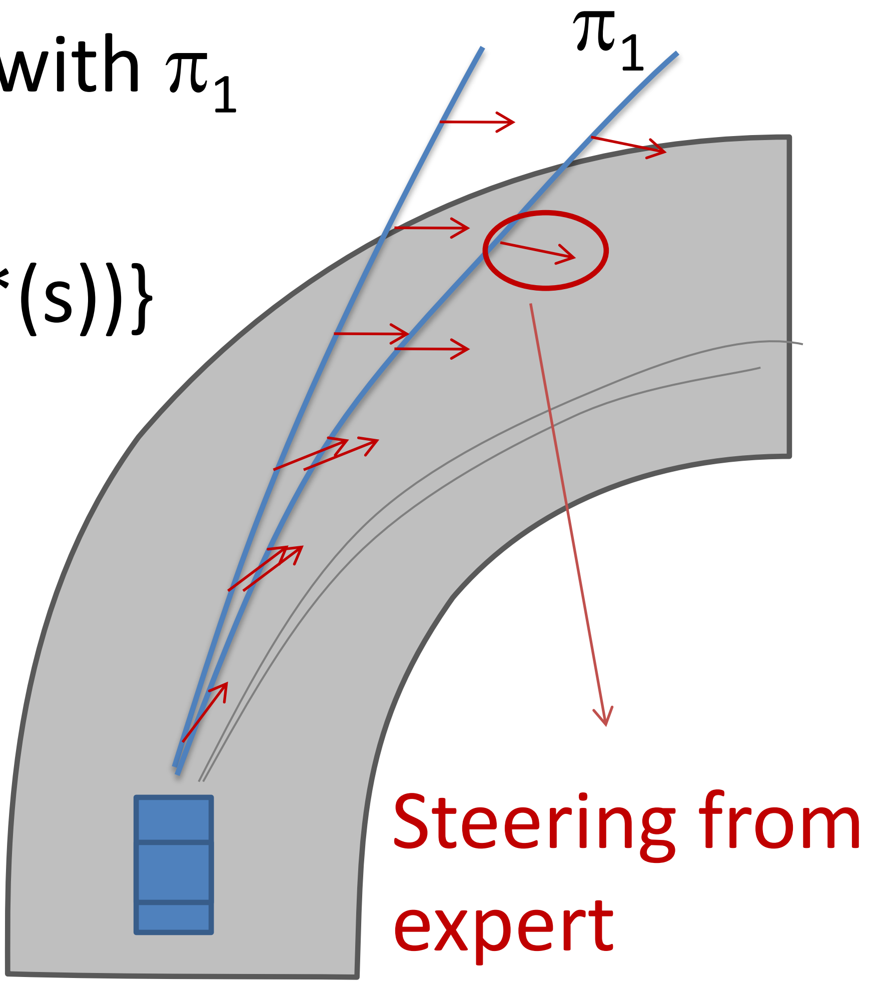
DAgger: Dataset Aggregation

- Collect new trajectories with π_1



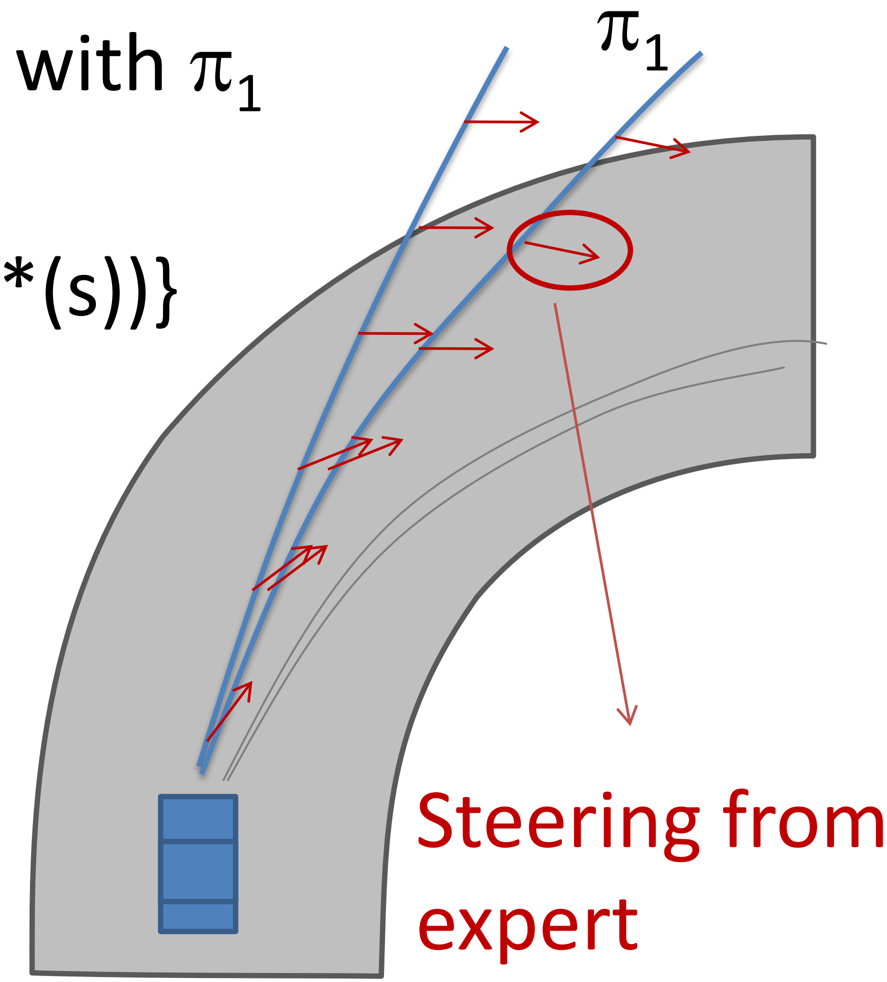
DAgger: Dataset Aggregation

- Collect new trajectories with π_1
- New Dataset $D_1' = \{(s, \pi^*(s))\}$



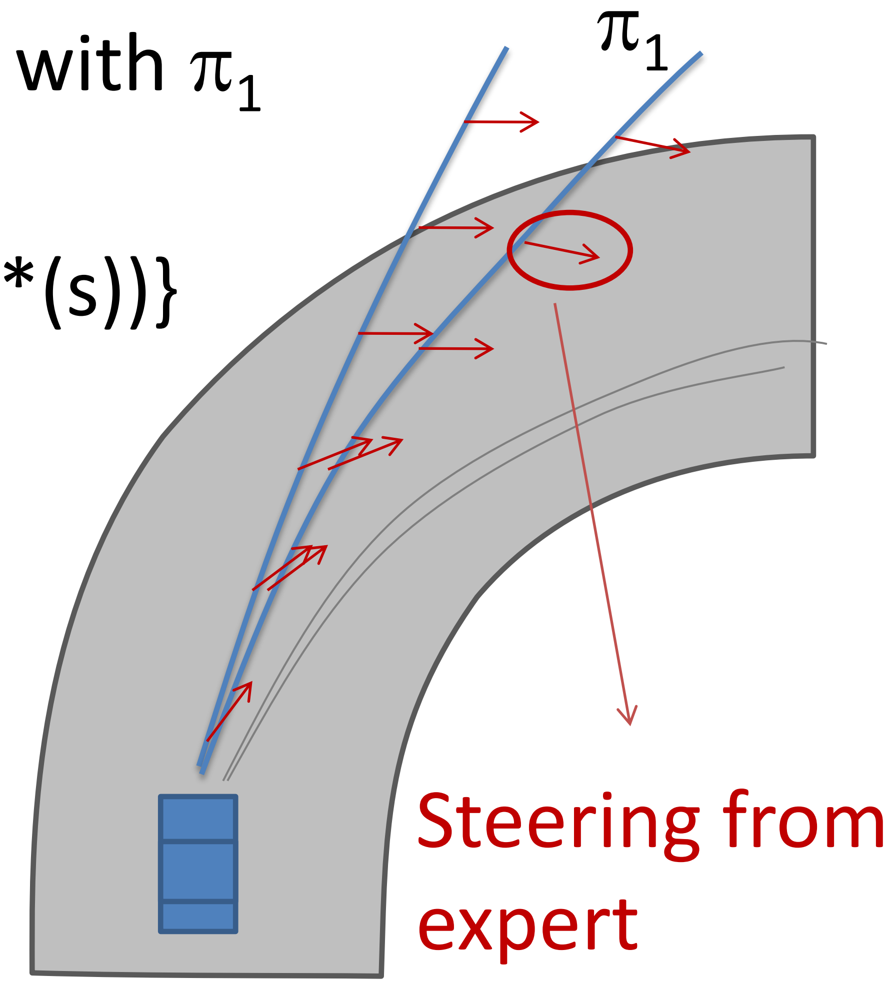
DAgger: Dataset Aggregation

- Collect new trajectories with π_1
- New Dataset $D_1' = \{(s, \pi^*(s))\}$
- Aggregate Datasets:
  $D_1 = D_0 \cup D_1'$



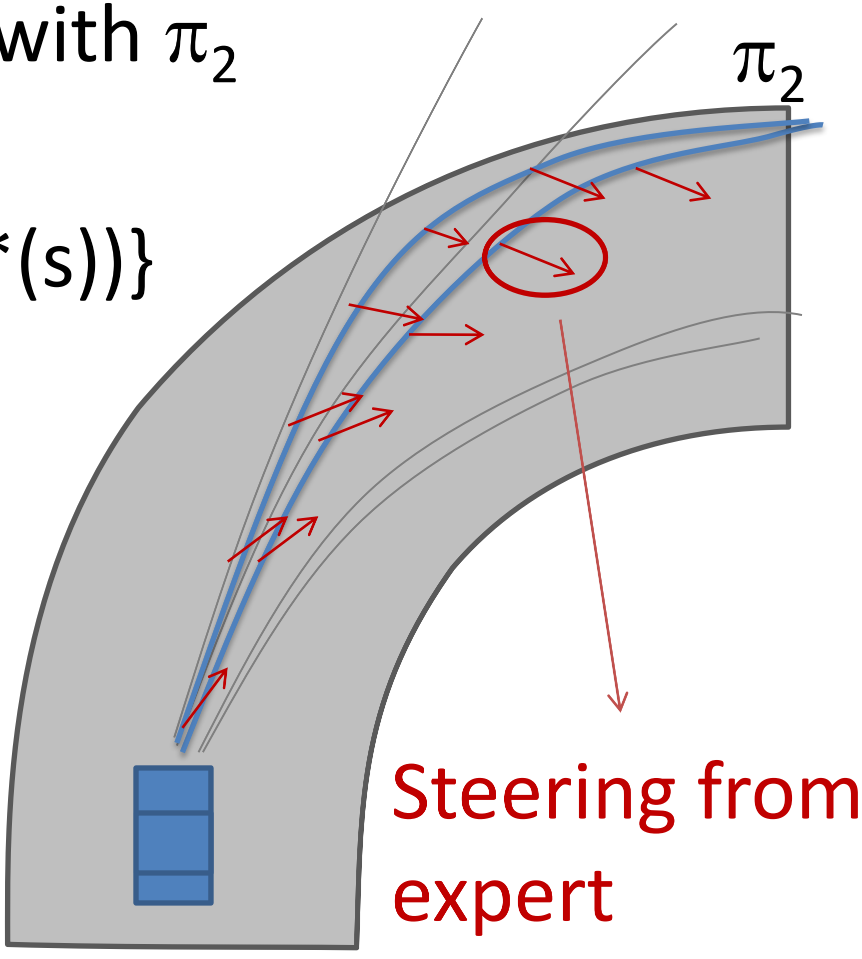
DAgger: Dataset Aggregation

- Collect new trajectories with π_1
- New Dataset $D_1' = \{(s, \pi^*(s))\}$
- Aggregate Datasets:
  $D_1 = D_0 \cup D_1'$
- Train π_2 on D_1



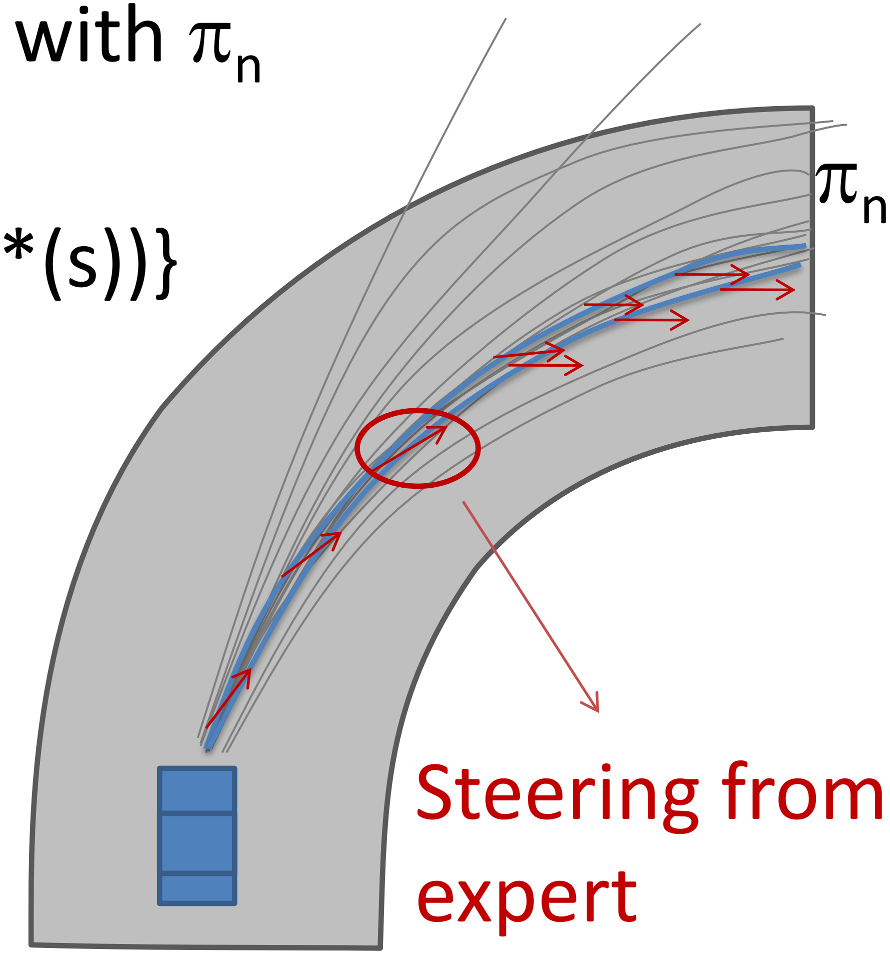
DAgger: Dataset Aggregation

- Collect new trajectories with π_2
- New Dataset $D_2' = \{(s, \pi^*(s))\}$
- Aggregate Datasets:
  $D_2 = D_1 \cup D_2'$
- Train π_3 on D_2



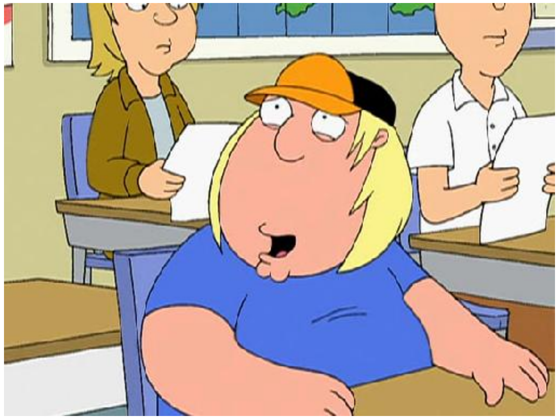
DAgger: Dataset Aggregation

- Collect new trajectories with π_n
- New Dataset $D_n' = \{(s, \pi^*(s))\}$
- Aggregate Datasets:
  $D_n = D_{n-1} \cup D_n'$
- Train π_{n+1} on D_n



Online Learning

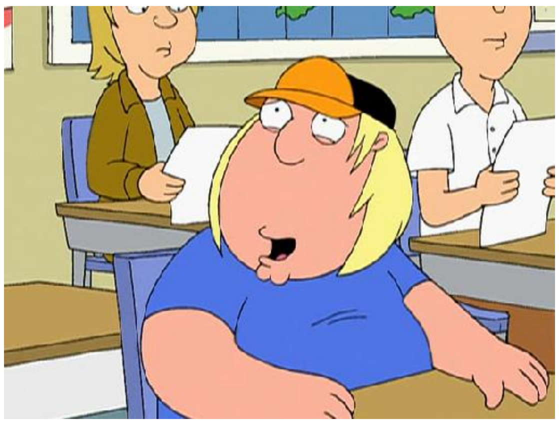
Learner



Adversary

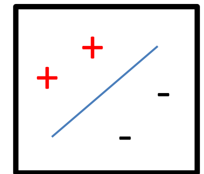
Online Learning

Learner



⋮

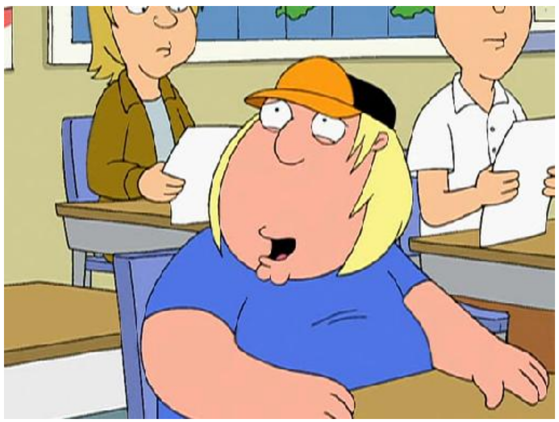
Current Hypothesis h_n

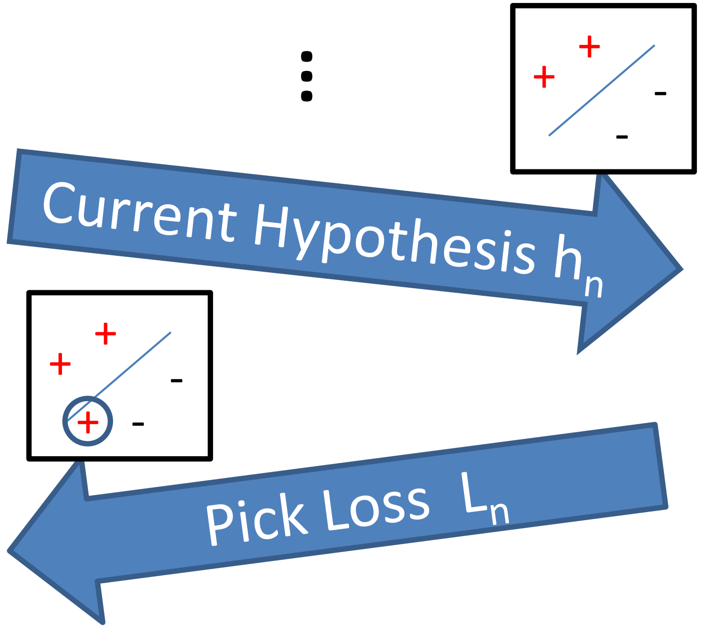


Adversary

Online Learning

Learner

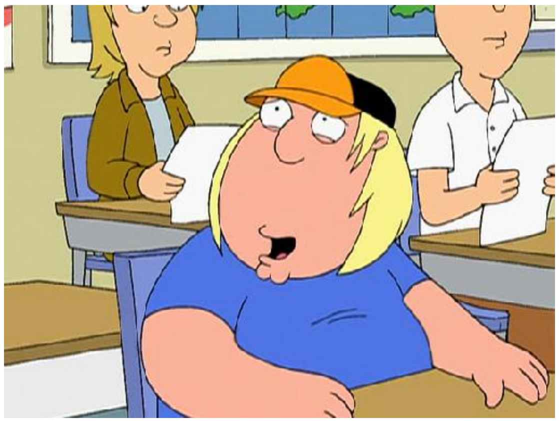




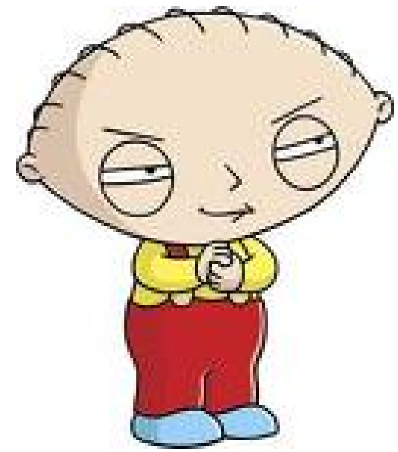
Adversary

Online Learning

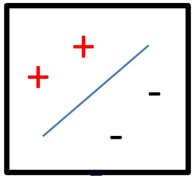
Learner



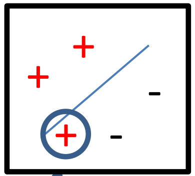
Adversary



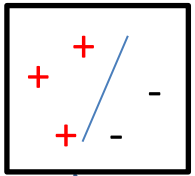
⋮



Current Hypothesis h_n

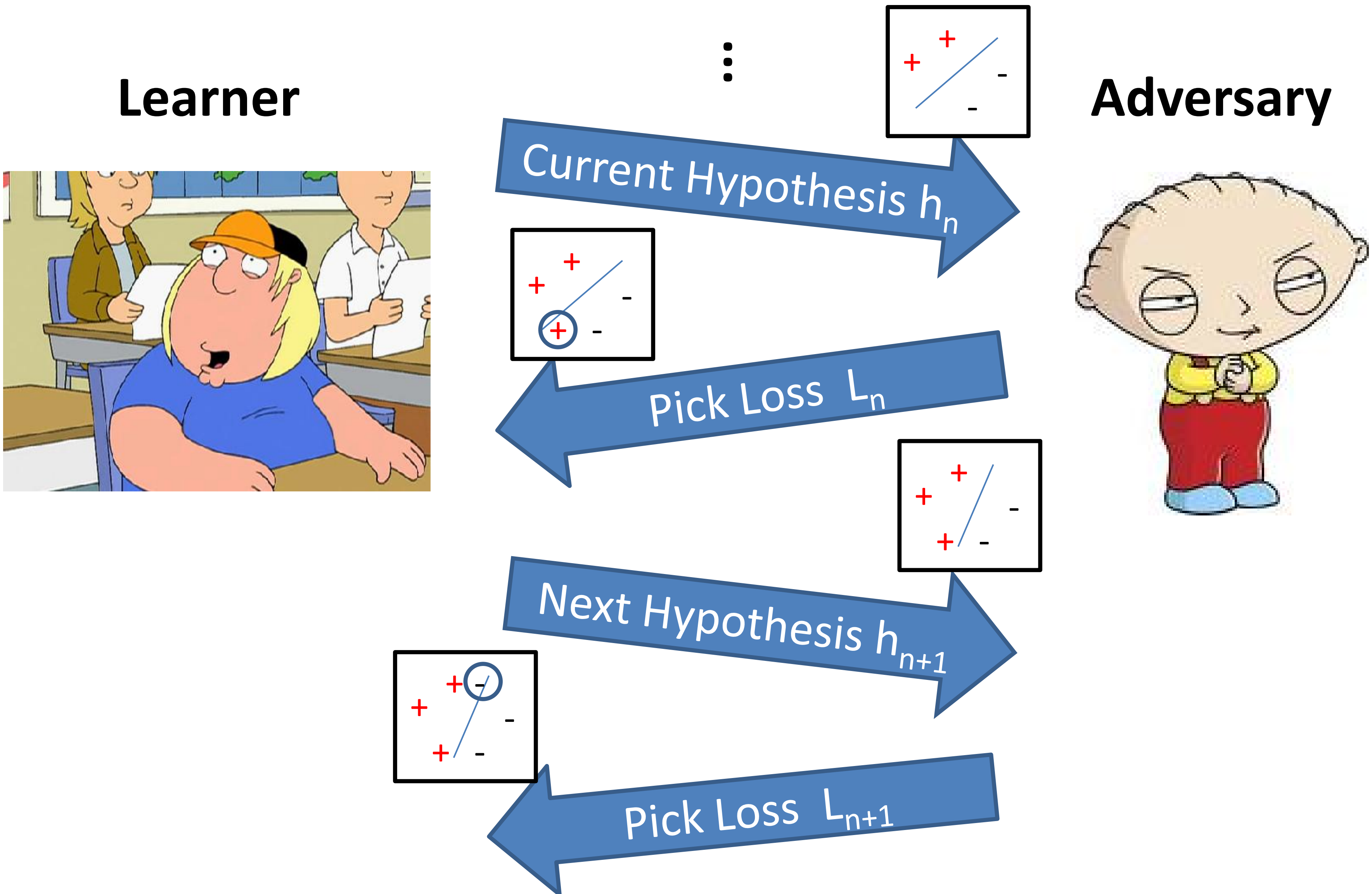


Pick Loss L_n



Next Hypothesis h_{n+1}

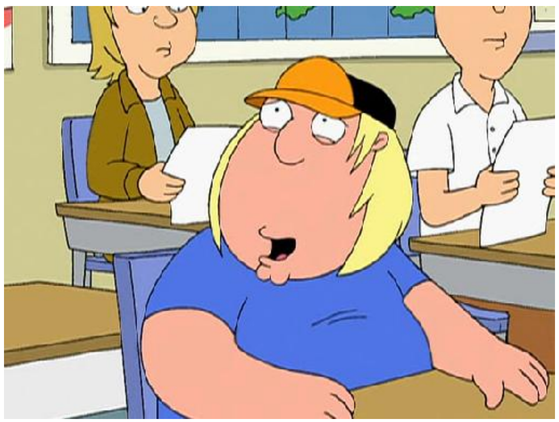
Online Learning



Online Learning

⋮

Learner



Current Hypothesis h_n

Pick Loss L_n

Next Hypothesis h_{n+1}

Pick Loss L_{n+1}

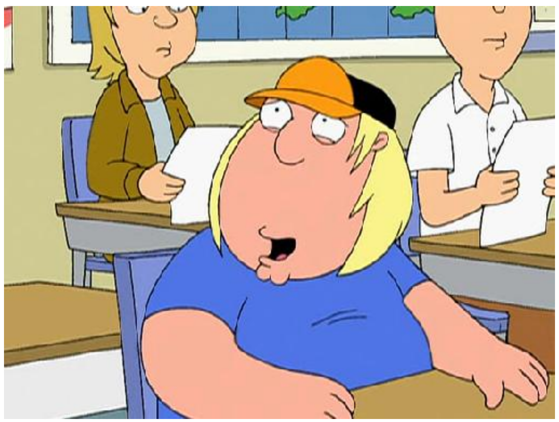
Adversary

Avg. Regret: $\gamma_n = \frac{1}{n}\left[\sum_{i=1}^{n} L_i(h_i) - \min_{h \in H} \sum_{i=1}^{n} L_i(h)\right]$

DAgger as Online Learning

⋮

Learner



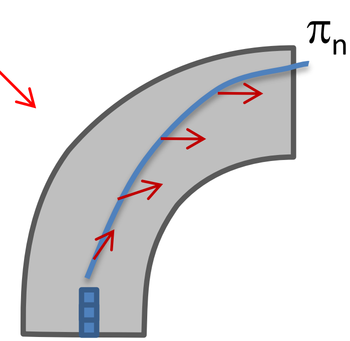
Adversary

Current Policy π_n

Pick Loss L_n

Next Policy π_{n+1}

Pick Loss L_{n+1}

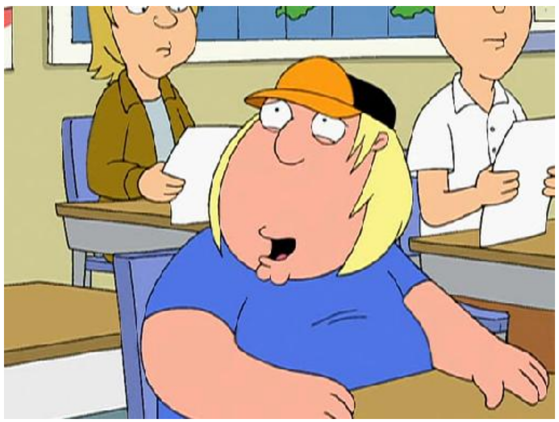


$$L_n(\pi) = \mathop{\mathbf{E}}_{s \sim D(\pi_n)} [\ell(\pi, s, \pi^*(s))]$$

DAgger as Online Learning

⋮

Learner



Adversary

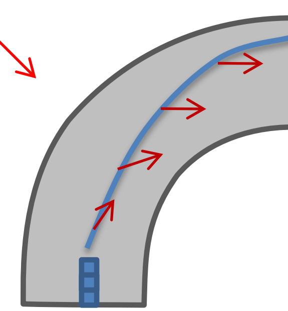
Current Policy π_n

Pick Loss L_n

Next Policy π_{n+1}

Pick Loss L_{n+1}

$$\pi_{n+1} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} L_i(\pi)$$

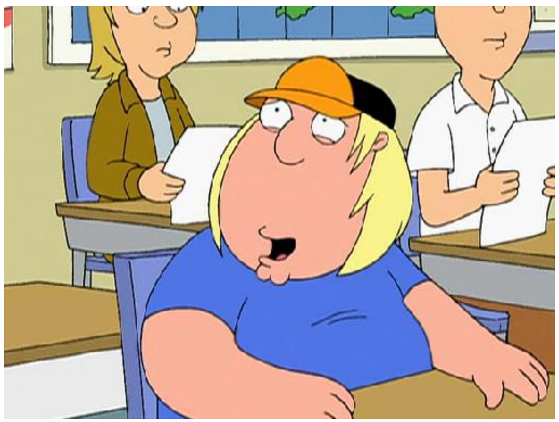


$$L_n(\pi) = \mathop{E}_{s \sim D(\pi_n)}[\ell(\pi, s, \pi^*(s))]$$

DAgger as Online Learning

⋮

Learner



Adversary

Current Policy π_n

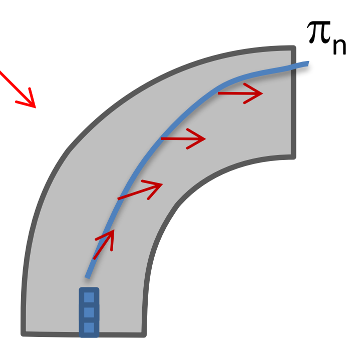
Pick Loss L_n

Next Policy π_{n+1}

Pick Loss L_{n+1}

$$\pi_{n+1} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} L_i(\pi)$$

Follow-The-Leader (FTL)



$$L_n(\pi) = \mathop{E}_{s \sim D(\pi_n)} [\ell(\pi, s, \pi^*(s))]$$

Theoretical Guarantees of DAgger

- Best policy π in sequence $\pi_{1:N}$ guarantees:

$$\mathbf{J}(\pi) \leq \mathbf{T}(\varepsilon_{\mathbf{N}} + \gamma_{\mathbf{N}}) + \mathbf{O}(\mathbf{T}/\mathbf{N})$$

Avg. Loss on Aggregate Dataset

Avg. Regret of $\pi_{1:N}$

Iterations of DAgger

Theoretical Guarantees of DAgger

- Best policy π in sequence $\pi_{1:N}$ guarantees:

$$\mathbf{J}(\pi) \leq \mathbf{T}(\varepsilon_{\mathbf{N}} + \gamma_{\mathbf{N}}) + \mathbf{O}(\mathbf{T}/\mathbf{N})$$

Avg. Loss on Aggregate Dataset

Avg. Regret of $\pi_{1:N}$

Iterations of DAgger

- For strongly convex loss, **N = O(TlogT) iterations**:

$$\mathbf{J}(\pi) \leq \mathbf{T}\varepsilon_{\mathbf{N}} + \mathbf{O}(1)$$

Theoretical Guarantees of DAgger

- Best policy π in sequence $\pi_{1:N}$ guarantees:

$$\mathbf{J}(\pi) \leq \mathbf{T}(\varepsilon_{\mathbf{N}} + \gamma_{\mathbf{N}}) + \mathbf{O}(\mathbf{T}/\mathbf{N})$$

Avg. Loss on Aggregate Dataset

Avg. Regret of $\pi_{1:N}$

Iterations of DAgger

- For strongly convex loss, **N = O(TlogT) iterations**:

$$\mathbf{J}(\pi) \leq \mathbf{T}\varepsilon_{\mathbf{N}} + \mathbf{O}(1)$$

- Any No-Regret algorithm has same guarantees

Theoretical Guarantees of DAgger

- If sample **m trajectories** at each iteration, w.p. 1-δ:

$$\mathbf{J}(\pi) \leq \mathbf{T}(\hat{\varepsilon}_{\mathbf{N}} + \gamma_{\mathbf{N}}) + \mathbf{O}(\mathbf{T}\sqrt{\mathbf{log}(1/\delta)} / \sqrt{\mathbf{Nm}})$$

Empirical Avg. Loss on Aggregate Dataset

Avg. Regret of $\pi_{1:N}$

Theoretical Guarantees of DAgger

- If sample **m trajectories** at each iteration, w.p. 1-δ:

$$\mathbf{J}(\pi) \leq \mathbf{T}(\hat{\varepsilon}_{\mathbf{N}} + \gamma_{\mathbf{N}}) + \mathbf{O}(\mathbf{T}\sqrt{\mathbf{log}(1/\delta)} / \sqrt{\mathbf{Nm}})$$

Empirical Avg. Loss on Aggregate Dataset

Avg. Regret of $\pi_{1:N}$

- For strongly convex loss, **N = O(T^{2}log(1/δ)) , m=1**, w.p. 1-δ:

$$\mathbf{J}(\pi) \leq \mathbf{T}\hat{\varepsilon}_{\mathbf{N}} + \mathbf{O}(1)$$

Experiments: 3D Racing Game

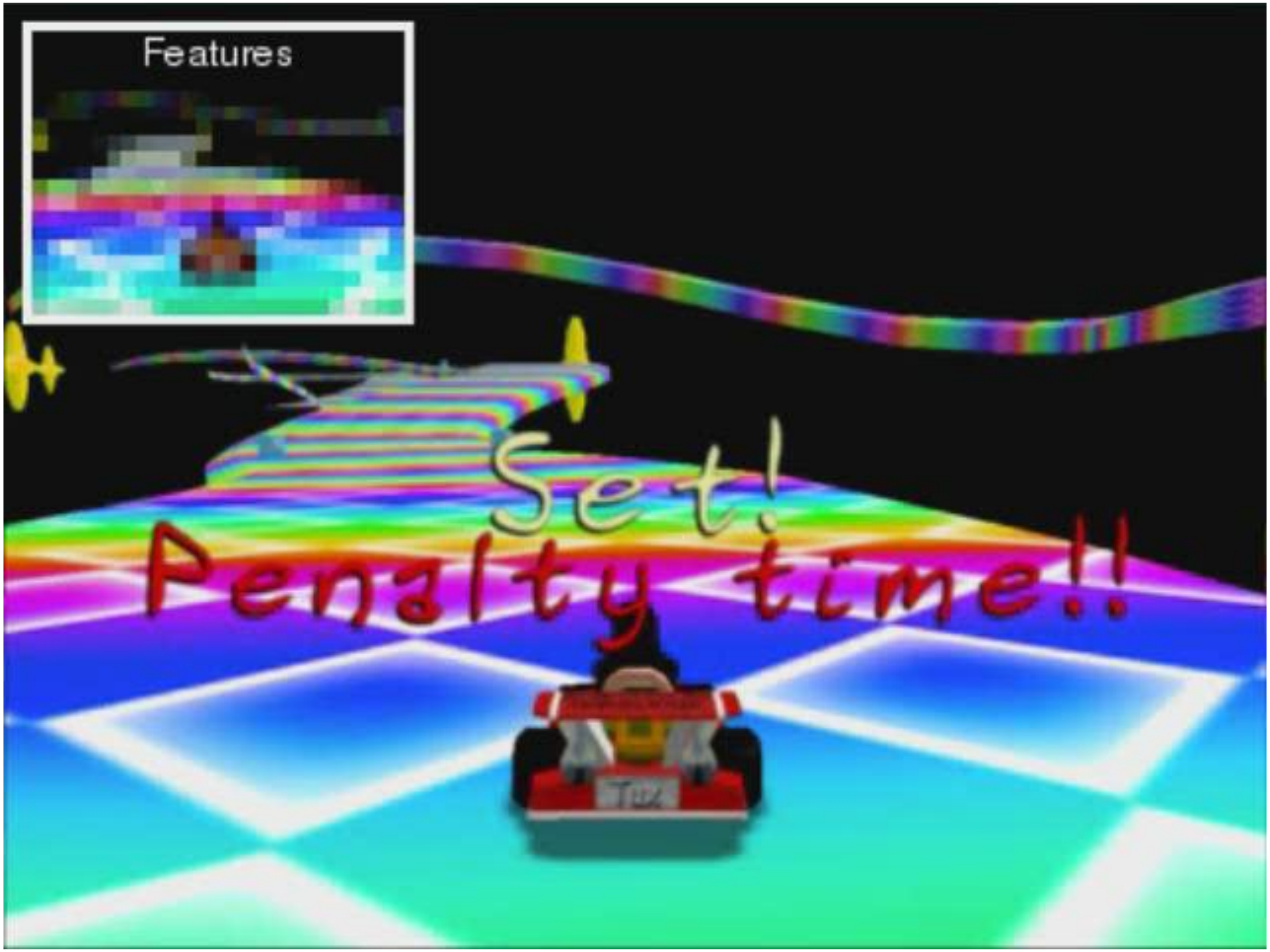
Input:

Resized to 25x19 pixels (1425 features)

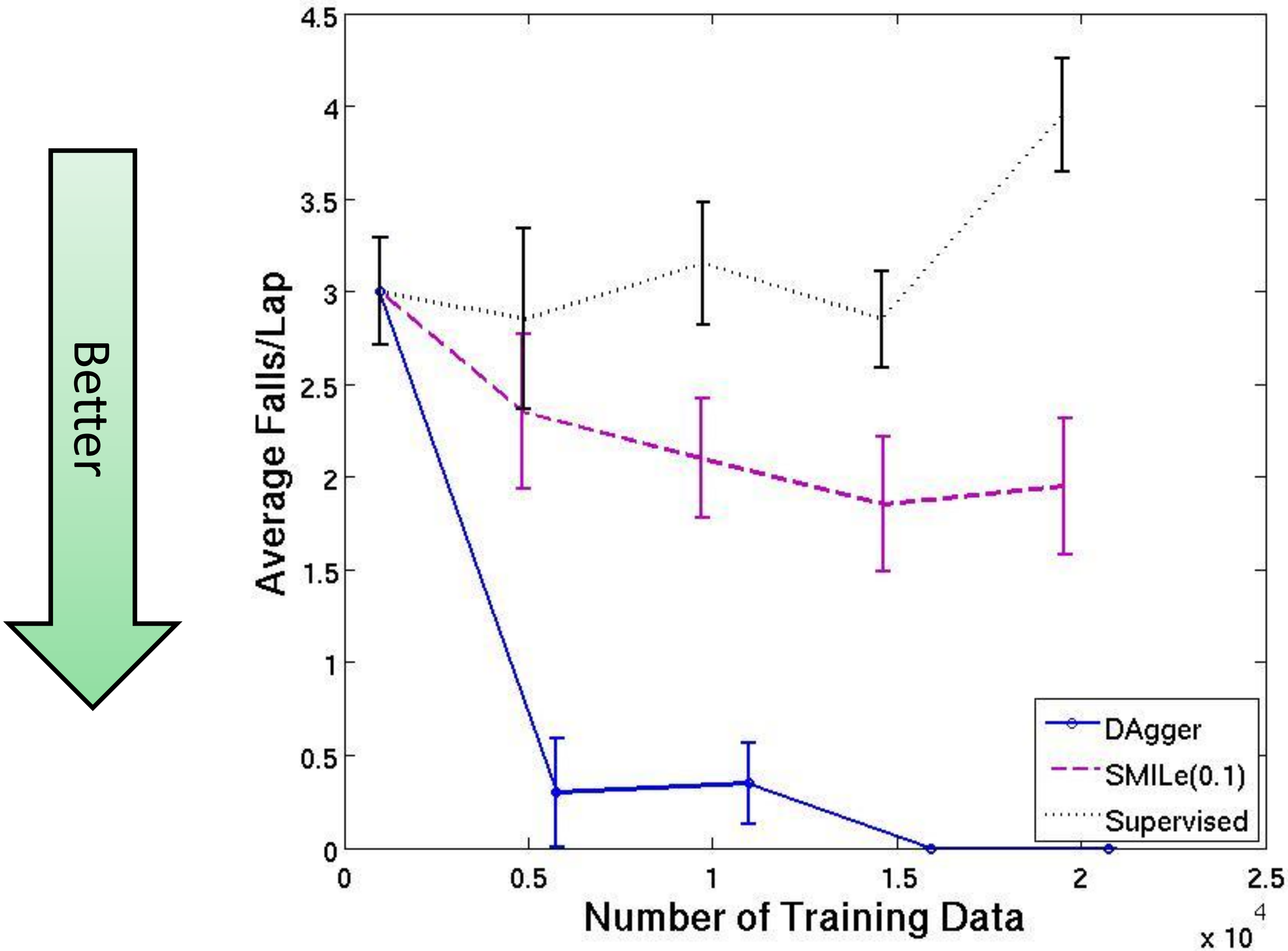
Output:

Steering in [-1,1]

DAgger Test-Time Execution



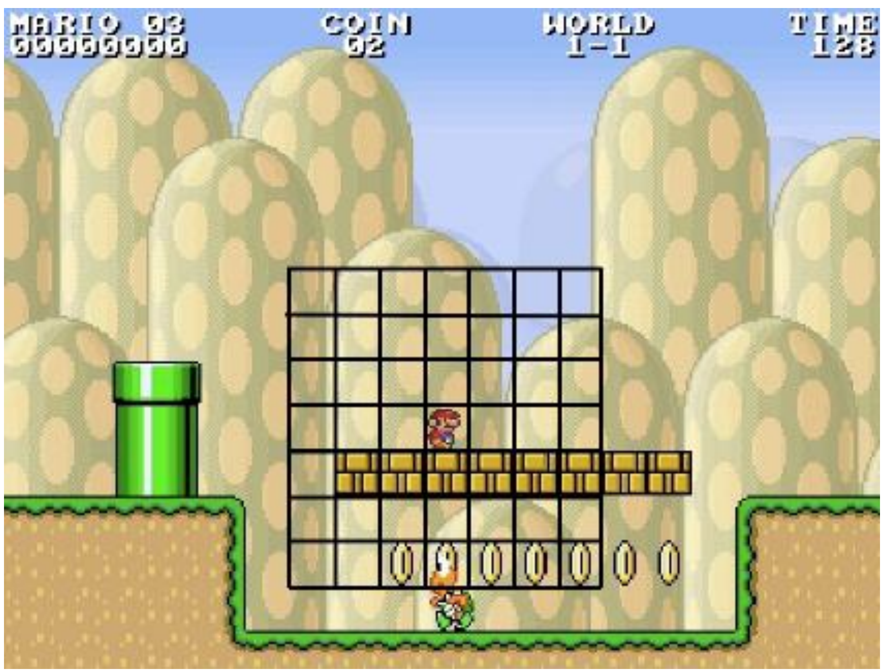
Average Falls/Lap



Experiments: Super Mario Bros

From Mario AI competition 2009

Input:



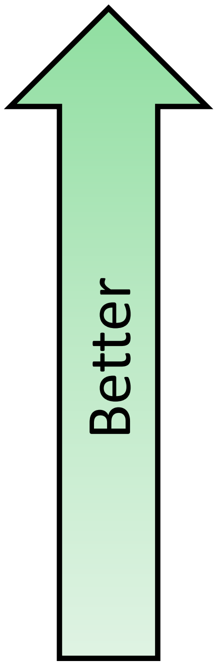
Output:

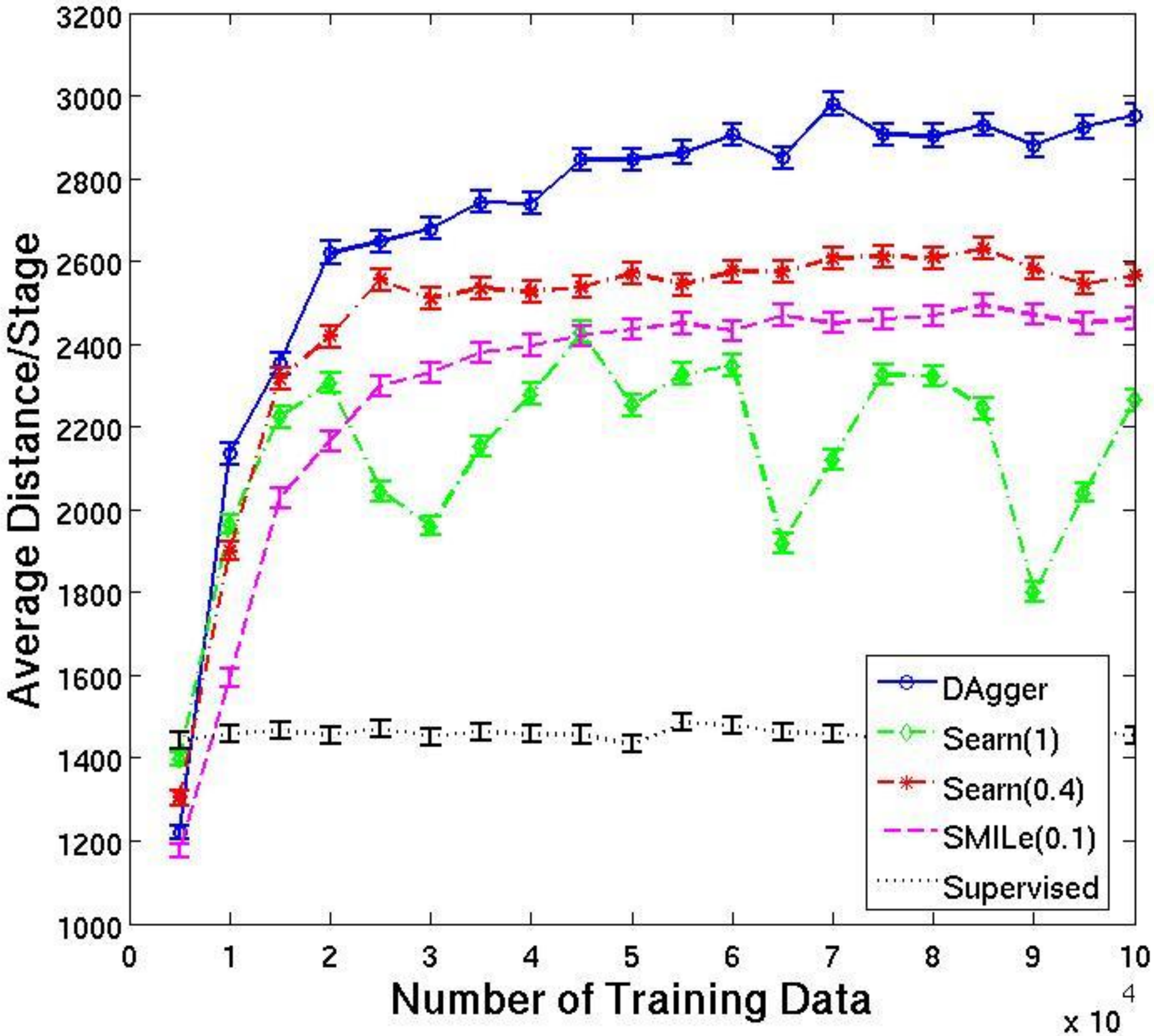
Jump in {0,1}
Right in {0,1}
Left in {0,1}
Speed in {0,1}

Extracted 27K+ binary features from last 4 observations (14 binary features for every cell)

Test-Time Execution

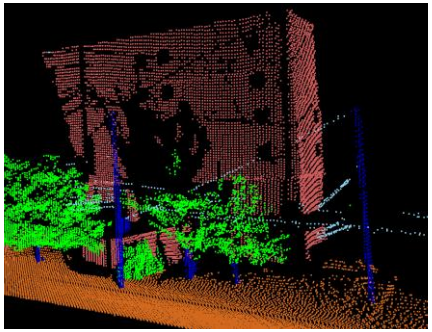
Average Distance/Stage





Conclusion

- Take-Home Message
  - Simple iterative procedures can yield much better performance.
- Can also be applied for **Structured Prediction**:
  - NLP (e.g. Handwriting Recognition)
  - Computer Vision [Ross & al., CVPR 2011]



- Future Work:
  - Combining with other Imitation Learning techniques [Ratliff 06]
  - Potential extensions to Reinforcement Learning?

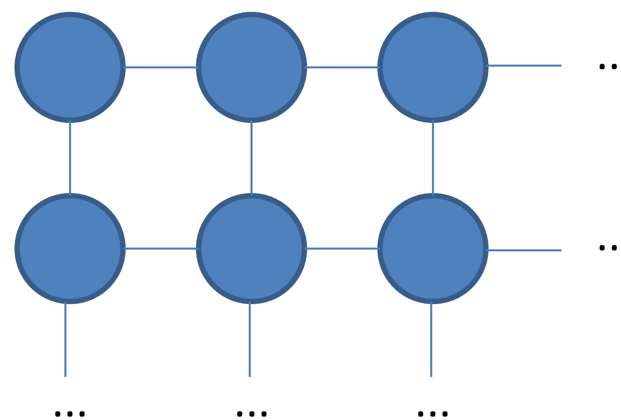
Questions

Structured Prediction

- Example: Scene Labeling

Image

Graph Structure over Labels

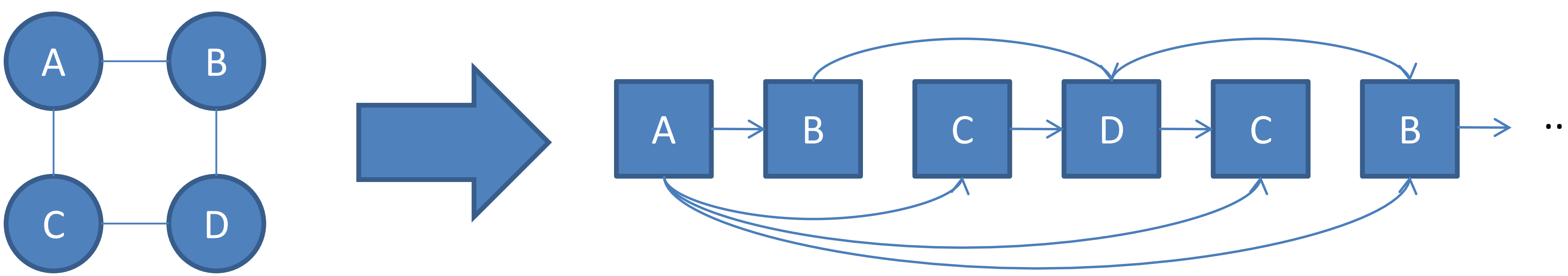


Structured Prediction

- Sequentially label each node using neighboring predictions
  - e.g. In Breath-First-Search Order (Forward & Backward passes)

Graph

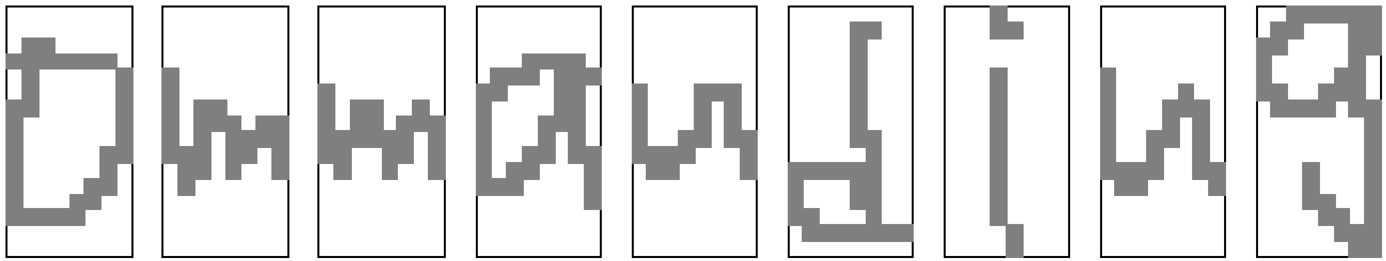
Sequence of Classifications



Structured Prediction

- Input to Classifier:
  - Local image features in neighborhood of pixel
  - Current neighboring pixels' labels
- Neighboring labels depend on classifier itself
- DAgger finds a classifier that does well at predicting pixel labels given the neighbors' labels it itself generates during the labeling process.

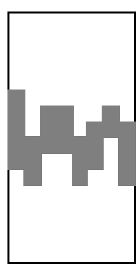
Experiments: Handwriting Recognition



[Taskar 2003]

Input:

Image current letter:



Previous predicted letter: **o**

Output:

Current letter in {a,b,...,z}

Test Folds Character Accuracy