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

Stephane Ross, Geoffrey J. Gordon and J. Andrew Bagnell [1]

Dennis Aumiller

Heidelberg University
Imitation Learning

October 12, 2018

Table of Contents

Outline

Motivation

"Standard Arguments" for Imitation Learning:

- Generally helpful in sequential prediction problems
- Learn a robust policy that can recover from failure (compare Supervised Learning)
- *Efficiently* learn such a policy (compare Reinforcement Learning)

Further, we have seen shortcomings of previous algorithms:

- Convergence might not be (or only weakly) guaranteed
  - SEARN: Grows quadratically in the number of errors
- Resulting policy might be a stochastic mixture of several policies, or non-stationary

Application Examples

- Autonomous navigation
- POS tagging
- Handwriting recognition

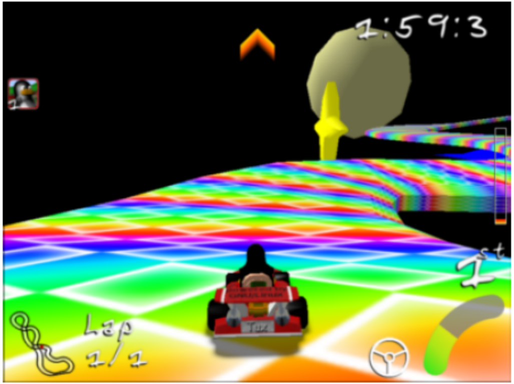

Figure: SuperTux Cart racing game [1].

Problem Formulation

General notation:

- Π: Class of all possible policies, with $\pi \in \Pi$ an arbitrary policy.
- T: Task horizon, $t \in T$ a specific time step.
- d_π^t: Distribution of states in policy π at time step t.
- $d_\pi = \frac{1}{T} \sum_{t=1}^{T} d_\pi^t$: State distribution of policy π across all time steps.
- $C(s, a)$: Immediate cost of an action a under a given state s. Note that C is bound by $[0, 1]$.
- $C_\pi(s) = \mathbb{E}_{a \sim \pi(s)}[C(s, a)]$: Expected immediate cost in state s under policy π.
- $J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{s \sim d_\pi^t}[C_\pi(s)] = T\, \mathbb{E}_{s \sim d_\pi}[C_\pi(s)]$: Total cost of one episode under policy π.
- $\ell(s, \pi)$: Surrogate loss function (possibly with respect to an expert policy).
- $Q_t^{\pi'}(s, \pi)$: t-step cost of executing π from the initial state s and then following π' after.

Goal

- True cost of action $C(s, a)$ is usually unknown. Thus, we use the surrogate loss $\ell(s, \pi)$ instead.
- Find a policy $\hat{\pi}$ that best approximates the expert policy π^* under the distribution of states

$$\hat{\pi} = \text{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim d_\pi}(\ell(s, \pi)) \tag{1}$$

Shortcomings:

- Due to unknown system dynamics, cannot compute d_π.
  $\implies$ non-iid supervised learning problem, since representation of d depends on π!

Outline

Reduction to Behavioral Cloning

Train classifier $\mathcal{D}_{\text{sup}}$ only on states encountered by expert $(= d_{\pi^*})$, which yields policy π_{sup}:

$$\hat{\pi}_{\text{sup}} = \text{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim d_{\pi^*}} [\ell(s, \pi)] \tag{2}$$

Reduction to Behavioral Cloning

Train classifier $\mathcal{D}_{\text{sup}}$ only on states encountered by expert $(= d_{\pi^*})$, which yields policy π_{sup}:

$$\hat{\pi}_{\text{sup}} = \text{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim d_{\pi^*}} [\ell(s, \pi)] \tag{2}$$

Assume $\ell(s, \pi)$ is 0-1 loss, or upper bounded on 0-1 loss, implies:

Theorem (2.1 Error of Behavioral Cloning)

Let $\mathbb{E}_{s \sim d_{\pi^*}} [\ell(s, \pi)] = \epsilon$, then the resulting cost of the episode $J(\pi) \leq J(\pi^*) + T^2 \epsilon$.

For proof, see yesterday's slides, or [2].

Forward Training [1, 2]

- Iteratively trained policy
  - Non-stationary
  - π_t for each time step t
- π_t trained to mimic π^* on state distribution induced by previous policies $\pi_1, \ldots \pi_{t-1}$
- Thus guarantees expected loss to match average loss during training
- Each policy is only adopted on its specific time step!

Initialize $\pi_1^0, \ldots, \pi_T^0$ to query and execute π^*.
for $i = 1$ **to** T **do**
  Sample T-step trajectories by following π^{i-1}.
  Get dataset $\mathcal{D} = \{(s_i, \pi^*(s_i))\}$ of states, actions taken by expert at step i.
  Train classifier $\pi_i^i = \operatorname{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(e_\pi(s))$.
  $\pi_j^i = \pi_j^{i-1}$ for all $j \neq i$
end for
Return $\pi_1^T, \ldots, \pi_T^T$

Algorithm 3.1: Forward Training Algorithm.

Forward Training Guarantee

Theorem (2.1 Error of Behavioral Cloning)

Let π be such that $\mathbb{E}_{s\sim d_\pi}[\ell(s,\pi)] = \epsilon$, and
$Q^{\pi^*}_{T-t+1}(s,a) - Q^{\pi^*}_{T-t+1}(s,\pi^*) \leq u,\ \forall a, t \in \{1,2,...,T\},\ d^t_\pi(s) > 0,$
then it follows that $J(\pi) \leq J(\pi^*) + uT\epsilon$.

Also holds for any general policy π that can guarantee ϵ surrogate loss!

Forward Training Guarantee

Proof: Consider π that executes learned policy for first t steps, then lets the expert policy π^* take over. Then

$$J(\pi) = J(\pi^*) + \sum_{t=0}^{T-1} [J(\pi_{1:T-t}) - J(\pi_{1:T-t-1})] \text{ (deviation cost per time step)} \qquad (3)$$

Forward Training Guarantee

Proof: Consider π that executes learned policy for first t steps, then lets the expert policy π^* take over. Then

$$J(\pi) = J(\pi^*) + \sum_{t=0}^{T-1} [J(\pi_{1:T-t}) - J(\pi_{1:T-t-1})] \text{ (deviation cost per time step)} \quad (3)$$

$$= J(\pi^*) + \sum_{t=1}^{T} \mathbb{E}_{s \sim d_\pi^t} [Q_{T-t+1}^{\pi^*}(s, \pi) - Q_{T-t+1}^{\pi^*}(s, \pi^*)] \text{ (per definition of } J(\pi)) \quad (4)$$

Forward Training Guarantee

Proof: Consider π that executes learned policy for first t steps, then lets the expert policy π^* take over. Then

$$J(\pi) = J(\pi^*) + \sum_{t=0}^{T-1} [J(\pi_{1:T-t}) - J(\pi_{1:T-t-1})] \text{ (deviation cost per time step)} \tag{3}$$

$$= J(\pi^*) + \sum_{t=1}^{T} \mathbb{E}_{s \sim d_\pi^t} [Q^{\pi^*}_{T-t+1}(s, \pi) - Q^{\pi^*}_{T-t+1}(s, \pi^*)] \text{ (per definition of } J(\pi)) \tag{4}$$

$$\leq J(\pi^*) + u \sum_{t=1}^{T} \mathbb{E}_{s \sim d_\pi^t} [\ell(s, \pi)] = J(\pi^*) + uT\epsilon \text{ (inequality from bounding on 0-1 loss).} \tag{5}$$

Stochastic Mixing Iterative Learning (SMILe) [2]

- Strongly related to SEARN
- Start from expert policy π_0
- At step i, $\hat{\pi}_i$ is trained to mimic expert under previous policy π_{i-1}

Initialize $\pi^0 \leftarrow \pi^*$ to query and execute expert.
for $i = 1$ **to** N **do**
 Execute π^{i-1} to get $\mathcal{D} = \{(s, \pi^*(s))\}$.
 Train classifier $\hat{\pi}^{*i} = \mathrm{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(e_\pi(s))$.
 $\pi^i = (1-\alpha)^i \pi^* + \alpha \sum_{j=1}^{i} (1-\alpha)^{j-1} \hat{\pi}^{*j}$.
end for
Remove expert queries: $\tilde{\pi}^N = \frac{\pi^N - (1-\alpha)^N \pi^*}{1-(1-\alpha)^N}$
Return $\tilde{\pi}^N$

Algorithm 4.1: The SMILe Algorithm.

Stochastic Mixing Iterative Learning (SMILe) [2]

- Update can be rewritten as $\pi_i = \pi_{i-1} + \alpha(1-\alpha)^{i-1}(\hat{\pi}_i - \pi_0)$
- Generally $O(T^2)$ regret
- If parameter $\alpha \in O(\frac{1}{T^2})$ guarantees near-linear regret in T and ϵ
- Also needs less iterations than SEARN ($O(T^2(\ln T)^{\frac{3}{2}})$ instead of $O(T^3 \ln T)$)

Data Aggregation (DAgger)

- Choose arbitrary starting policy
- Let policy $\hat{\pi}_i$ run, and flip coin whether $\hat{\pi}_i$ or expert π^* execute current action
- But always record expert decision (in the background)
- Construct new dataset as aggregation of *all* previous samples
- Train new policy, and repeat

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

Data Aggregation (DAgger)

- Avoid mixture of policies
- Follow-the-leader strategy avoids overfitting
- Mixture parameter β_i generally indicator function $I(i = 1)$ or exponentially decaying value $p^{(i-1)}$

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

Outline

Guarantee of Bounds to No-Regret Learning

- Assumes infinte sample trajectories at each iteration
- $\epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{s \sim d_{\pi_i}}[\ell(s, \pi)]$ true loss of best policy

Theorem (3.1 Existence of Optimal Policy for Infinte Sample Case)

For DAgger, if $N \in \tilde{O}(T)$ there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ s.t. $\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \epsilon_N + O(\frac{1}{T})$.

Proof via analysis results in next part.

Guarantee of Bounds to No-Regret Learning

- Holds for policy that performs best under its own distribution:
- $\hat{\pi} = \text{argmin}_{\pi \in \{\hat{\pi}_1, \ldots, \hat{\pi}_N\}} \mathbb{E}_{s \sim d_\pi} [\ell(s, \pi)]$
- Alternatively, pick uniformly at random from $\{\hat{\pi}_1, \ldots, \hat{\pi}_N\}$

Guarantee of Bounds to No-Regret Learning

Combining Theorem 3.1 with Theorem 2.2 yields another result, important for the no-regret convergence. Only requirement is that ℓ upper bounds true cost C:

Theorem (3.2 Convergence with respect to Expert Policy)

For DAgger, if $N \in \tilde{O}(uT)$ there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ s.t. $J(\hat{\pi}) \leq J(\pi^*) + uT\epsilon_N + O(1)$.

Guarantee of Bounds to No-Regret Learning

Combining Theorem 3.1 with Theorem 2.2 yields another result, important for the no-regret convergence. Only requirement is that ℓ upper bounds true cost C:

Theorem (3.2 Convergence with respect to Expert Policy)

For DAgger, if $N \in \tilde{O}(uT)$ there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ s.t. $J(\hat{\pi}) \leq J(\pi^*) + uT\epsilon_N + O(1)$.

Proof:

- 3.1 guarantees policy that satisfies prerequisites for 2.1
- Additional error bound of $O(\frac{1}{T})$ over T time steps is in $O(1)$.

Guarantees for Finite Sample Case

- Usually only limited samples available
- Still guaranteed to find converging policy with certain probability
- Let m be the samples per iteration
- Make use of a special case of the Azuma-Hoeffding inequality

Azuma-Hoeffding Inequality

- Gives probabilty that mean of samples from distribution are ϵ-close to the actual mean of the distribution
- The more samples we have, the closer we can get

It states:

$$X_N = \sum_{i=1}^{N} Y_i \ , \ \mathbb{E}[Y] = \mu$$

$$P[X_N/N - \mu > \epsilon/N] \leq e^{-\frac{\epsilon^2}{2N}} = \delta \tag{6}$$

$$\implies \epsilon = \sqrt{\log \frac{1}{\delta} \cdot 2N} \tag{7}$$

$$\implies P[X_N/N - \mu \leq \frac{\sqrt{2N \log \frac{1}{delta}}}{N}] \geq 1 - \delta \tag{8}$$

Theorems for Finite Sample Case

Theorem (3.3 Convergence for Finite Sample Case)

For DAgger, if $N \in \tilde{O}(T^2 \log(1/\delta))$ *and* $m \in O(1)$, *then with probability of at least* $1 - \delta$ *there exists a policy* $\hat{\pi} \in \{\hat{\pi}_1, \ldots, \hat{\pi}_N\}$ *s.t.* $\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \hat{\epsilon}_N + O(\frac{1}{T})$.

Theorems for Finite Sample Case

Theorem (3.3 Convergence for Finite Sample Case)

For DAgger, if $N \in \tilde{O}(T^2 \log(1/\delta))$ *and* $m \in O(1)$*, then with probability of at least* $1 - \delta$ *there exists a policy* $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ *s.t.* $\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \hat{\epsilon}_N + O(\frac{1}{T})$.

Theorem (3.4 Convergence for Finite Sample Case with respect to Expert)

For DAgger, if $N \in \tilde{O}(u^2 T^2 \log(1/\delta))$ *and* m *in* $O(1)$ *then with probabilty at least* $1 - \delta$ *there exists a policy* $\hat{\pi} \in \{\hat{\pi}_1, ..., \hat{\pi}_N\}$ *s.t.* $J(\hat{\pi}) \leq J(\pi^*) + uT\hat{\epsilon}_N + O(1)$.

No-Regret Algorithms Guarantees

- Hold for any no-regret algorithm, not just Follow-the-leader

Limitation of Average Regret

Further assumptions:

- β_i is non-increasing
- $\ell_{\max} \geq \ell_t(s, \hat{\pi}_t),\ \forall t \in \{1, ..., T\}$
- n_β is largest n s.t. $\beta_n > \frac{1}{T}$

Limitation of Average Regret

Further assumptions:

- β_i is non-increasing
- $\ell_{\max} \geq \ell_t(s, \hat{\pi}_t), \ \forall t \in \{1, ..., T\}$
- n_β is largest n s.t. $\beta_n > \frac{1}{T}$

Lemma (4.1 Bound on Total Variation)

$||d_{\pi_t} - d_{\hat{\pi}_t}||_1 \leq 2T\beta_i$, especially for $\beta_i \leq 1/T$.

Limitation of Average Regret

Further assumptions:

- β_i is non-increasing
- $\ell_{\max} \geq \ell_t(s, \hat{\pi}_t), \ \forall t \in \{1, \ldots, T\}$
- n_β is largest n s.t. $\beta_n > \frac{1}{T}$

Lemma (4.1 Bound on Total Variation)

$||d_{\pi_t} - d_{\hat{\pi}_t}||_1 \leq 2T\beta_i$, *especially for* $\beta_i \leq 1/T$.

Theorem (4.1 Average Regret)

For DAgger, there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, \ldots, \hat{\pi}_N\}$ *s.t.* $\mathbb{E}_{s\sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \hat{\epsilon}_N + \gamma_N + \frac{2\ell_{\max}}{N}[n_\beta + T\sum_{i=n_\beta+1}^{N}\beta_i]$, *for* γ_N *the average regret of* $\{\hat{\pi}_1, \ldots, \hat{\pi}_N\}$

Limitation of Average Regret

$$
\begin{aligned}
&\min_{\hat{\pi} \in \hat{\pi}_{1:N}} \mathbb{E}_{s \sim d_{\hat{\pi}}} [\ell(s, \hat{\pi})] \\
&\leq \tfrac{1}{N} \textstyle\sum_{i=1}^{N} \mathbb{E}_{s \sim d_{\hat{\pi}_i}} (\ell(s, \hat{\pi}_i)) \\
&\leq \tfrac{1}{N} \textstyle\sum_{i=1}^{N} [\mathbb{E}_{s \sim d_{\pi_i}} (\ell(s, \hat{\pi}_i)) + 2\ell_{\max} \min(1, T\beta_i)] \\
&\leq \gamma_N + \tfrac{2\ell_{\max}}{N} [n_\beta + T \textstyle\sum_{i=n_\beta+1}^{N} \beta_i] + \min_{\pi \in \Pi} \tfrac{1}{N} \textstyle\sum_{i=1}^{N} \ell_i(\pi) \\
&= \gamma_N + \epsilon_N + \tfrac{2\ell_{\max}}{N} [n_\beta + T \textstyle\sum_{i=n_\beta+1}^{N} \beta_i]
\end{aligned}
$$

Average Regret of Finite Sampling Case

Theorem (4.2 Average Regret in Finite Sampling Case)

For DAgger, with probability at least $1-\delta$, there exists a policy $\hat{\pi} \in \{\hat{\pi}_1, \dots, \hat{\pi}_N\}$ s.t. $\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \hat{\epsilon}_N + \gamma_N + \frac{2\ell_{\max}}{N}[n_\beta + T\sum_{i=n_\beta+1}^{N} \beta_i] + \ell_{\max}\sqrt{\frac{2\log(1/\delta)}{mN}}$, for γ_N the average regret of $\{\hat{\pi}_1, \dots, \hat{\pi}_N\}$

Average Regret of Finite Sampling Case

$$
\begin{aligned}
&\min_{\hat{\pi}\in\hat{\pi}_{1:N}} \mathbb{E}_{s\sim d_{\hat{\pi}}}[\ell(s,\hat{\pi})] \\
&\leq \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \mathbb{E}_{s\sim d_{\hat{\pi}_i}}[\ell(s,\hat{\pi}_i)] \\
&\leq \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \mathbb{E}_{s\sim d_{\pi_i}}[\ell(s,\hat{\pi}_i)] + \tfrac{2\ell_{\max}}{N}[n_\beta + T\sum_{i=n_\beta+1}^{N}\beta_i] \\
&= \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \mathbb{E}_{s\sim D_i}[\ell(s,\hat{\pi}_i)] + \tfrac{1}{mN}\sum_{i=1}^{N}\sum_{j=1}^{m} Y_{ij} \\
&\quad + \tfrac{2\ell_{\max}}{N}[n_\beta + T\textstyle\sum_{i=n_\beta+1}^{N}\beta_i] \\
&\leq \tfrac{1}{N}\textstyle\sum_{i=1}^{N} \mathbb{E}_{s\sim D_i}[\ell(s,\hat{\pi}_i)] + \ell_{\max}\sqrt{\tfrac{2\log(1/\delta)}{mN}} \\
&\quad + \tfrac{2\ell_{\max}}{N}[n_\beta + T\textstyle\sum_{i=n_\beta+1}^{N}\beta_i] \\
&\leq \hat{\epsilon}_N + \gamma_N + \ell_{\max}\sqrt{\tfrac{2\log(1/\delta)}{mN}} + \tfrac{2\ell_{\max}}{N}[n_\beta + T\textstyle\sum_{i=n_\beta+1}^{N}\beta_i]
\end{aligned}
$$

Comparison to SEARN [3]

SEARN

- Requires large number of iterations to converge
  - Both in theory and practice
- Mixture of policies
- [2] mention $O(T^2 \log T)$ instead of linear scaling in T as presented in [3]

DAgger

- Stronger guarantees due to No-Regret approach
- Follow-the-leader returns single policy
- Requires less queries to expert (although still a lot)

Outline

Super Tux Cart

- Continuous action space (steering angle between [-1,1])
- DAgger performed best with $\beta_i = I(i = 1)$

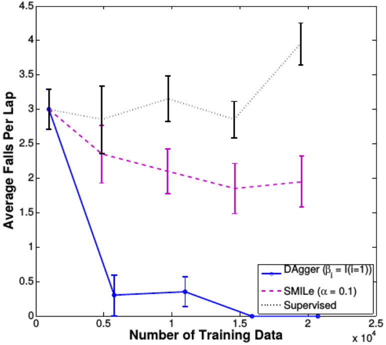

Figure: Convergence of various methods for Super Tux Cart [1].

Super Mario Bros.

- Expert is near-optimal planning algorithm (expensive to query)
- Discrete action space (four buttons to press)
- Very simple levels!

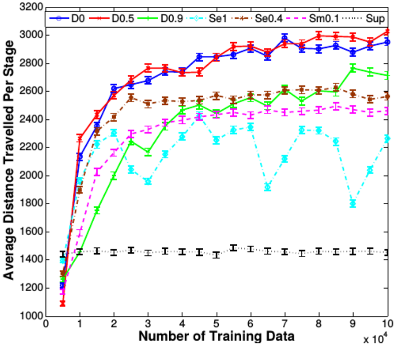

Figure: Performance in Super Mario Bros. for various methods [1].

Handwriting Recognition

- Expert is supervised training data
- Discrete action space (predicted character)
- Probably sub-optimal compared to state-of-the-art neural architectures (RNN/LSTM)

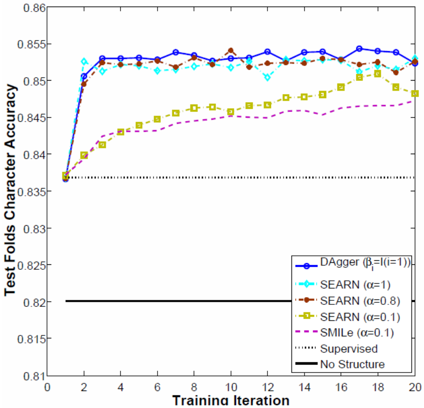

Figure: Performance for handwriting recognition [1].

Outline

Limitations of DAgger

Main problem:

- DAgger still relies extremely heavily on the oracle/expert
- Each query can potentially be expensive, or make the collection of training samples hard

Limitations of DAgger

Main problem:

- DAgger still relies extremely heavily on the oracle/expert
- Each query can potentially be expensive, or make the collection of training samples hard
- Only guaranteed to work for convex loss functions
- Potentially a stronger bound can be given under the assumption of strong convexity

Limitations of DAgger

Main problem:

- DAgger still relies extremely heavily on the oracle/expert
- Each query can potentially be expensive, or make the collection of training samples hard
- Only guaranteed to work for convex loss functions
- Potentially a stronger bound can be given under the assumption of strong convexity
- Instability issues [4]

Limitations of DAgger

Main problem:

- DAgger still relies extremely heavily on the oracle/expert
- Each query can potentially be expensive, or make the collection of training samples hard
- Only guaranteed to work for convex loss functions
- Potentially a stronger bound can be given under the assumption of strong convexity
- Instability issues [4]

Still only at most performance on par with teacher!

Outline

Conclusion

- Presented various methods that are within $O(T^2)$ or lower bounds of an expert solution:
  - Forward Training
  - SMILe / SEARN
- Presented DAgger, a deterministic and stationary solution that alleviates several problems of previous methods by aggregating data across several episodes and querying an oracle/expert
- Provided extensive analysis of bounds for finite and infinite sample case for DAgger, which show nice properties
- Analyzed experiments, and showed some of DAgger's limitations

Possible Extensions

Backplay curriculum learning [5, 6]:

- Instead of starting from initial starting state s_0, run the first iterations from an inverse policy $p_{t:T}$ that runs the expert for iterations 1:t, and **then** the policy π.
- Potentially avoids distribution shift towards uncommon failure cases that appear in first iterations
- Stabler training even with mixed policy for later episodes?
- Building a dense search tree from the bottom up could help to relinquish some queries: Instead of querying the expert, use surrogate loss as distance between expert replay and prediction (without expert)

Use with Deep Neural architectures:

- Experimental setup was conducted with linear SVM classifiers

Bibliography I

- Stéphane Ross, Geoffrey Gordon, and Drew Bagnell.
  A reduction of imitation learning and structured prediction to no-regret online learning.
  In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

- Stéphane Ross and Drew Bagnell.
  Efficient reductions for imitation learning.
  In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668, 2010.

- Hal Daumé, John Langford, and Daniel Marcu.
  Search-based structured prediction.
  Machine learning, 75(3):297–325, 2009.

Bibliography II

Hal Daumé.
A Course in Machine Learning.
http://ciml.info/.

Cinjon Resnick, Roberta Raileanu, Sanyam Kapoor, Alex Peysakhovich, Kyunghyun Cho, and Joan Bruna.
Backplay:" Man muss immer umkehren".
arXiv preprint arXiv:1807.06919, 2018.

Tim Salimans and Richard Chen.
Learning Montezuma's Revenge from a single demonstration.
https://blog.openai.com/learning-montezumas-revenge-from-a-single-demonstration/.

Questions

Thank you for your attention!