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

Stéphane Ross, Geoffrey J. Gordon, J. Andrew Bagnell

by Kyutae Sim

10/7/2023

Main Problem

- Sequence Prediction Problems
  - Perform sequence of actions when given a sequence of observations
  - Future observations affected by previous actions
  - Dependence on input
- Typical Imitation learning approach violates i.i.d assumption made in statistical learning problems
  - Independent and Identical random variables
  - Poor Performance

Motivation

- Sequence prediction problems relevant in robotics
- Autonomous systems must be able to take action and adapt to the environment affected by their actions

- Stationary Policy - same policy for each timestep
- No-regret algorithm: produces a sequence of policies

Problem Setting

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

ℓ: the observed surrogate loss function

=> expected 0-1 loss of π with respect to π^* in state s,

or a squared/hinge loss of π with respect to π^* in s

$d_\pi = \frac{1}{T} \sum_{t=1}^{T} d_\pi^t$ average distribution of states

Problem Setting

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

- Hard to know true cost C(s,a)
- Upper bound J(π) (total expected cost of executing policy for T-steps) with surrogate loss function

Limitations of Prior Work - Ross and Bagnell, 2010

Forward training algorithm - non-stationary policy trained to mimic expert on the distribution resulting from all previous iterations

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 $\pi_i^i = \text{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(e_\pi(s))$.
  $\pi_j^i = \pi_j^{i-1}$ for all $j \neq i$
end for
Return $\pi_1^T, \ldots, \pi_T^T$

Algorithm 3.1: Forward Training Algorithm.

Limitations of Prior Work - Ross and Bagnell, 2010

Forward training algorithm

- Non-stationary policy: different policy for each time step in sequence
- Impractical when T is large, cannot be stopped until all T iterations are complete
- Number of mistakes grows linearly with T
- Not ideal for most real-world applications

Limitations of Prior Work - Ross and Bagnell, 2010

SMILe (Stochastic Mixing Iterative Learning):

Trained to mimic the expert under the distribution of the previous iteration

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 $\hat{\pi}^{*i} = \text{argmin}_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(e_\pi(s))$.
  $\pi^i = (1-\alpha)^i \pi^* + \alpha \sum_{j=1}^{i} (1-\alpha)^{j-1} \hat{\pi}^{*j}$.
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.

Limitations of Prior Work - Ross and Bagnell, 2010

SMILe - Stochastic Mixing Iterative Learning

- Stationary stochastic policy (a finite mixture of policies)
- Alleviates problems presented by the forward training algorithm
- Some policies in the mixture are worse than others
- Learned controller may be unstable
- Guarantees near linear regret in T and ϵ

Algorithm

DAGGER (Dataset Aggregation) - Stationary deterministic policy

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.

DAGGER Algorithm

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.

- First iteration uses expert's policy to gather trajectories D and trains the next policy that best mimics the expert
- Next iteration, the policy collects more trajectories and adds it to D

DAGGER Algorithm

- Building up set of inputs that the learned policy might encounter based on previous iterations
- Follow-The-Leader: At iteration n, we pick best policy π_n

$$\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i.$$

DAGGER Algorithm

$$\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i.$$

- Use modified policy to better leverage the presence of the expert

DAGGER Algorithm

Theorem 3.1. *For* DAGGER, *if* N *is* $\tilde{O}(T)$ *there exists a policy* $\hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.* $\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \epsilon_N + O(1/T)$

Theorem 3.2. *For* DAGGER, *if* N *is* $\tilde{O}(uT)$ *there exists a policy* $\hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.* $J(\hat{\pi}) \leq J(\pi^*) + uT\epsilon_N + O(1)$.

- Guarantees near linear total cost if infinite samples are taken

$\epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{s \sim d_{\pi_i}}[\ell(s, \pi)]$: True loss of the best policy

DAGGER Algorithm

Theorem 3.3. *For* DAGGER, *if N is $O(T^2 \log(1/\delta))$ and m is $O(1)$ then with probability at least $1-\delta$ there exists a policy $\hat{\pi} \in \hat{\pi}_{1:N}$ s.t. $\mathbb{E}_{s\sim d_{\hat{\pi}}}[\ell(s,\hat{\pi})] \leq \hat{\epsilon}_N + O(1/T)$*

Theorem 3.4. *For* DAGGER, *if N is $O(u^2 T^2 \log(1/\delta))$ and m is $O(1)$ then with probability at least $1-\delta$ there exists a policy $\hat{\pi} \in \hat{\pi}_{1:N}$ s.t. $J(\hat{\pi}) \leq J(\pi^*) + uT\hat{\epsilon}_N + O(1)$.*

- Tighter bounds can be achieved using the strong convexity of the loss function

Theoretical Analysis

- Reduction of imitation learning to no-regret online learning
- Online learning produces a policy π_n that induces a loss $\ell_n(\pi_n)$
- No-regret algorithm produces a sequence of policies such that the average regret with respect to the best policy goes to 0 as N goes to infinity:

$$\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi_i) - \min_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi) \leq \gamma_N$$

Theoretical Analysis

- Uses no-regret property of the underlying Follow-The-Leader algorithm on strongly convex losses (Kakade and Tewari, 2009)
- Guarantees good performance under its own distribution

Theoretical Analysis

Lemma 4.1. $||d_{\pi_i} - d_{\hat{\pi}_i}||_1 \leq 2T\beta_i$.

Theorem 4.1. *For* DAGGER, *there exists a policy* $\hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.* $\mathbb{E}_{s\sim d_{\hat{\pi}}}[\ell(s,\hat{\pi})] \leq \epsilon_N + \gamma_N + \frac{2\ell_{\max}}{N}[n_\beta + T\sum_{i=n_\beta+1}^{N}\beta_i]$, *for* γ_N *the average regret of* $\hat{\pi}_{1:N}$.

Theorem 4.2. *For* DAGGER, *with probability at least* $1-\delta$, *there exists a policy* $\hat{\pi} \in \hat{\pi}_{1:N}$ *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* $\hat{\pi}_{1:N}$.

=> Bounding the total variation distance between the distribution of states encountered
=> Guaranteed to find a policy that achieves ϵ surrogate loss under its own distribution in the limit (we can bound policy with the loss of best policy)

=> True loss under on finite sample of trajectories

Experimental Setup: Super Tux Kart

Figure 1: Image from Super Tux Kart's Star Track.

Experimental Setup: Super Tux Kart

- Human expert used for demonstrations
- Linear controller as base learner
- Average falls per lap
- 1 lap of training per iteration (~1000 data points) and run SMILe and Dagger for 20 iterations

Experimental Results: Super Tux Kart

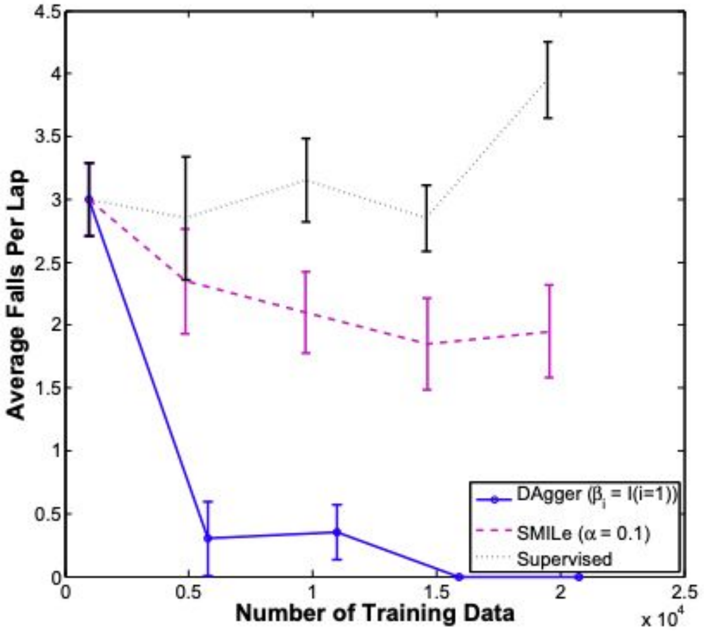

Figure 2: Average falls/lap as a function of training data.

- Baseline supervised approach does not improve as more data is collected as training laps are similar
- DAGGER never falls of after 15 iterations, smooth and qualitatively better than others

Experimental Setup: Super Mario Bros

Figure 3: Captured image from Super Mario Bros.

Experimental Setup: Super Mario Bros

- Expert: Near-optimal planning algorithm with full access to game state, can simulate consequences of actions exactly
- 4 independent linear SVM as the base learner (left, right, up, speed)
- Average distance travelled before dying, running out of time, or completing the stage
- 5000 data points per iteration (each stage is about 150 data points if run to completion) and run the methods for 20 iterations for each approach
- Tested p values in $\beta_i = p^{i-1}$

Experimental Results: Super Mario Bros

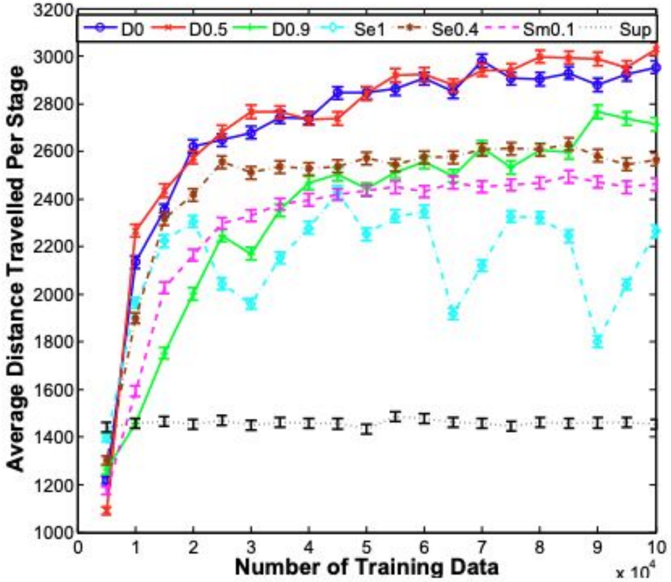

Figure 4: Average distance/stage as a function of data.

- p=0.5 performed the best
- Performance for supervised approach stagnates, often gets stuck
- Other methods learn to unstuck
- DAGGER outperforms all others

Experimental Setup: Handwriting Recognition

- Dataset containing approximately 6600 words, total of over 52,000 characters
- Predicts character in a word from left to right, using previously predicted character and a linear SVM
- Baselines: SMILe, SEARN, non-structured approach, and supervised training approach

Experimental Results: Handwriting Recognition

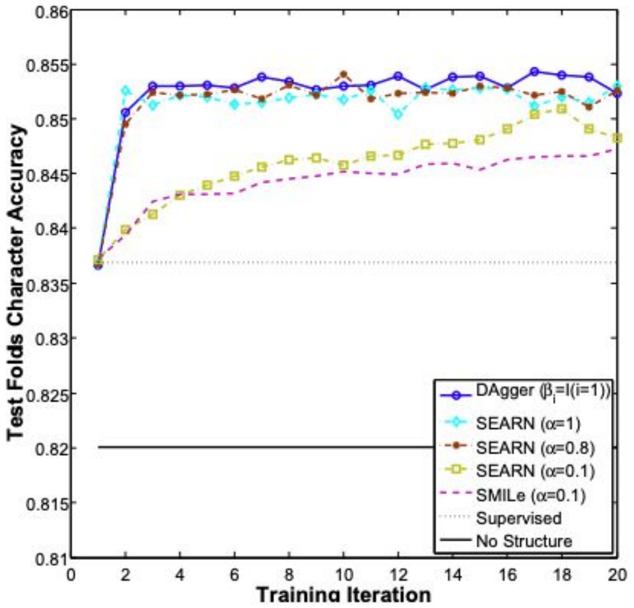

Figure 5: Character accuracy as a function of iteration.

- DAGGER achieves 85.5% accuracy
- Supervised approach performed better than the no-structure approach (83.6% vs 82.2%)

Discussion of Results

- No-regret methods can provide a learning reduction with strong performance guarantees in imitation learning and structure prediction
- Super Tux Kart shows how stochasticity of SMILe leads to less smooth/bad actions
- Super Mario Bros shows how choosing the right balance between expert and nonexpert trajectories is important in situations where it learns to unstuck Mario
- DAGGER performs well in problems with less state change

Critique / Limitations / Open Issues

- How generalizable is the reduction framework to the real-world, complex tasks with high-dimensional trajectories?
- What are some challenges when implementing the reduction framework in different problem settings?
- The paper mentions two types of loss function: 0-1 loss or squared loss. What other loss functions can the algorithm be extended to?

Future Work

- More sophisticated strategies than simple greedy forward for decoding structured prediction
- Using base classifiers relay on Inverse Optimal Control techniques to learn a cost function
- Scalability of DAGGER to real-world scenarios

Extended Readings

- Forward Training and SMILe: S. Ross and J. A. Bagnell. Efficient reductions for imitation learning. In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), 2010.
- Inverse Optimal Control: P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In Proceedings of the 21st International Conference on Machine Learning (ICML), 2004

Summary

- Addresses the problem of compounding errors in sequential prediction problems
- Robots deployed in the real world will face these problems of taking actions in a changing environment where their actions could affect the environment
- Prior works are based on nonstationary policies that includes different policies for each timestep, or stochastic policy of mixing policies => poor performance with compounding errors
- DAGGER, a no-regret method, aggregates data from learned policy
- Instead of compounding errors, DAGGER error rate is linear
- DAGGER performed better than other approaches in various imitation learning and sequential prediction tasks

Thanks!