Efficient Reductions for Imitation Learning

Stephane Ross
Joint work with Drew Bagnell
Imitation Learning

• Many successes:
  – Legged locomotion [Ratliff 06]
  – Outdoor navigation [Silver 08]
  – Helicopter flight [Abbeel 07]
  – Car driving [Pomerleau 89]
  – etc...
Key Problem:
Different training and testing distribution...

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

- Expert’s state distribution
- Error of \( \pi \) in state \( s \) (0-1 loss)

But testing on \( D(\pi_{\text{sup}}) \)...

Example: 3D racing game

Super Tux Kart

Input: [Image]

Output: Steering in \([-1,1]\)

Hard left turn

Hard right turn

Classifier
Video

- YouTube - Supervised Learning Approach To Imitation Learning
Cost grows quadratically in $T$! 😞

\[ J(\hat{\pi}_{\text{sup}}) \leq J(\pi^*) + T^2 \varepsilon \]

- Exp. T-step cost
- Error on $D(\pi^*)$
- # time steps

Reason: Don’t learn how to repair errors!
First solution:
Train T policies sequentially

• Train $\pi_1$ for 1$^{st}$ time step
Forward Training Algorithm

• Train $\pi_2$ for $2^{nd}$ time step given did $\pi_1$ on $1^{st}$
Forward Training Algorithm

• All $\pi_i$ trained on testing distribution!

$\pi_i$ learns how to repair errors!
Cost grows linearly in $T!$ (often...)

- Linear regret for many classes of problem:

\[
J(\pi_{1:T}) \leq J(\pi^*) + O(T\bar{e})
\]

avg. error
Linear when $\pi^*$ can recover quickly

What should I do now?

BAD!!!

$\pi^*$ can’t recover
Regret is $O(T^2\varepsilon)$

What should I do now?

Not too bad...

$\pi^*$ recovers in few steps
Regret is $O(T\varepsilon)$

Rapidly mixing chain under $\pi^*$ ➔ Linear regret 😊
ImLearn Algorithm
Change stationary policy SLOWLY!

• Set \( \pi_0 = \pi^* \) (query/act like \( \pi^* \) says)
• Train \( \pi'_1 \) to act like \( \pi^* \) under \( D(\pi_0) \)

• Update: \( \pi_1 = \pi_0 + \alpha(\pi'_1 - \pi^*) \)

Similar to:
SEARN [Daume 09]
CPI [Kakade 02]
Retrain under new policy

• Train $\pi'_2$ to act like $\pi^*$ under $D(\pi_1)$

• $\pi_2 = \pi_1 + \alpha(1-\alpha)(\pi'_2 - \pi^*)$
Decrease learning rate exponentially

- Train $\pi'_i$ to act like $\pi^*$ under $D(\pi_{i-1})$

- $\pi_i = \pi_{i-1} + \alpha(1-\alpha)^{i-1}(\pi'_i - \pi^*)$

Again $\pi'_i$ learns how to repair errors!
Remove queries/renormalize

- Finally remove remaining queries to $\pi^*$

- $\pi^*_N \propto \pi_N - (1-\alpha)^N\pi^*$
Near-linear regret on similar class of problems

- For $\alpha = T^{-2}$ and $N = 2\alpha^{-1}\ln T$:

$$J(\pi^\#_N) \leq J(\pi^*) + O(T \log T \overline{\epsilon})$$

- In practice can choose bigger $\alpha$ and smaller $N$
Only need $T$ times more samples, not $N$!

- $N$ iterations requires $\leq (T+1)m$ samples!

Small $\alpha$

$D(\pi^i) \approx D(\pi^{i-1})$

Reuse lots of samples in $D(\pi^{i-1})$
Experiments: Mario Bros
From the recent Mario AI competition

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 obs.
(14 bin. Features for every cell)
Video

• YouTube - Comparison of Imitation Learning approaches in Mario Bros
ImLearn achieves better performance with less samples!
Experiments: 3D racing game

Input:
- Resized to 24x18 pixels (1296 features)

Output:
- Steering in [-1,1]
- Discretized into 15 values
Video

• YouTube - ImLearn Approach to Imitation Learning
Conclusion

• General Idea: **Train, retrain and retrain, while changing policy slowly!**

• Generalizes to other areas of supervised learning (e.g. Structured classification)