Doing the best with the data we have

Adith Swaminathan,
Microsoft Research

Rossi Seminar, December 4 2020
ML in User-Interactive Systems

- Decisions, not predictions
- Problem: Learn from logged data

The data we collect from interactive systems...

- Framing: Contextual Bandit/Markov Decision Process
- Solution: Causal Reasoning + Pessimism!
- Many open questions...
User-Interactive Systems

We collect user interactions for
- Personalization
- Evaluating performance
- Improving systems
- ...

Search

Recommendations

Healthcare

Control Systems
A Recommendation Example

Goal: Recommend content that maximizes user engagement
Contextual Bandit

- Context $s \sim \text{Pr}(S)$
  - Actions do not affect future contexts
- Action $a \in A$
- Reward $r \sim \text{Pr}(R|s, a)$
- Policy $\pi : S \mapsto \Delta(A)$
- Using logged data
  $(s_i, a_i \sim \mu, r_i)$

Markov Decision Process

- Context $s_t \sim \text{Pr}(S|s_{t-1}, a_{t-1})$
  - State summarizes interaction history
- Goal: Maximize $\mathbb{E}_{s_t,a_t \sim \pi} [\sum_t r_t]$

Preamble

Deep Dive
Non-solutions

- Ignore logged data
- Explore-exploit algorithms
  - Warm-start CB algorithms
  - Batch RL algorithms
- Annotate logged data
- Supervised ML algorithms
  - without requiring $a^*$

Logged data = Prior exploration
How best to exploit?

Challenges: Non-interactive. Reward is unknown for the actions not taken

Distribution shift
Challenge: Distribution Shift

\[ \hat{V}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \frac{\pi(a_i|s_i)}{\mu(a_i|s_i)} r_i \]

Importance Sampling / Inverse Propensity Scoring gives unbiased value estimates despite distribution shift
Failure of ML with IPS objectives

$$\operatorname{argmax}_\pi \hat{V}(\pi)$$

Can we detect, avoid, remain robust to IPS failure during learning?
Solution: Pessimism

Which policy should we pick?

\[ \pi_1 = \mu \]

\[ \pi_2 \]

\[ \pi_3 \]

\[ \pi_4 \]
Pessimism works for Contextual Bandits

- Detect IPS failure: Compute diagnostics (eff. sample size, variance)
- Avoid: Use variance regularization in ML training
- Sample code: [POEM](http://www.cs.cornell.edu/~adith/POEM/)
- Widely deployed and useful in practice

\[
\arg\max_{\pi} \hat{V}(\pi) - \lambda \sqrt{\sum_{i=1}^{n} \left( \frac{\pi(a_i|s_i)}{\mu(a_i|s_i)} r_i - \hat{V}(\pi) \right)^2}
\]
Contextual Bandit

- Context $s \sim \Pr(S)$
- Actions do not affect future contexts

- Action $a \in A$

- Reward $r \sim \Pr(R|s, a)$

- Policy $\pi: S \mapsto \Delta(A)$

- Goal: Maximize $\mathbb{E}_{S, \pi(s)}[r]$  

- Using logged data  
  $(s_i, a_i \sim \mu, r_i)$

---

Markov Decision Process

- Context $s_t \sim \Pr(S|s_{t-1}, a_{t-1})$
- State summarizes interaction history

- Action $a \in A$

- Reward $r \sim \Pr(R|s, a)$

- Policy $\pi: S \mapsto \Delta(A)$

- Goal: Maximize $\mathbb{E}_{S_t, a_t \sim \pi} [\sum_t r_t]$  

---

Deep Dive
What about MDPs?

$r_t = r(s_t, a_t)$

$S_{t+1}$

$a_t$

$\pi: S_t \rightarrow a_t$

![Diagram](image.png)

- Evaluation Policy, $\pi$
- Behavior Policy, $\mu$

Probability of trajectory
What about MDPs?

\[ r_t = r(s_t, a_t) \]

Dataset drawn from
\[ D \sim \mu(s, a) \times r \times P(s' | s, a) \]

Find \( \hat{\pi} = \arg\max_{\pi \in \Pi} \mathbb{E}_{s_t, a_t \sim \pi} \left[ \sum_t \gamma^t r_t \right] \)
A Prototypical RL algorithm: API

- Fit a Q-function $f: S \times A \mapsto [0, V_{max}]$ to approximate cumulative reward

- Q-function is the fixed point of Bellman evaluation operator:

$$T^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s', a' \sim P}[f(s', a')]$$

- Approximate Policy Iteration (API):

  1. $f_{i+1} \leftarrow \hat{T}^\pi_k f_i$ till convergence to $f_*^k$ [Evaluate $\pi_k$ in inner loop]
  2. $\pi_{k+1}(s) = \arg\max_a f_*^k(s, a)$ [Update $\pi_{k+1}$ in outer loop]
  3. Collect data using $\pi_{k+1}$
Why is Batch RL hard?

- Algorithm encounters different $\pi$‘s during learning
- Each $\pi$ induces different $(s, a)$ distribution than $\mu$. 

![Diagram showing evaluation policy $\pi$ and behavior policy $\mu$ with probability of trajectory]
When does API for batch RL work?

Theoretical results assume $\mu$ “covers” any admissible distribution in the policy class: **concentrability**

$$\forall \pi \in \Pi: \max_{s,a} \frac{\Pr(s,a|\pi)}{\mu(s,a)} \leq \text{Constant}$$

In CB, we needed

$$\max_a \frac{\pi(a|s)}{\mu(a|s)} \leq \text{Constant}$$
Concentrability is an Impractical Assumption

Batch RL algos fail when assumption is violated
Doing the Best with What We’ve Got

Can we have guarantee that is adaptive to $\mu$ we are given?  
(by competing with best policy covered by $\mu$)

Want: Reliably return $\pi^*$

Yes! *Pessimism* accomplishes that!
Why did API fail without concentrability?

\[ T^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P, a' \sim \pi}[f(s', a')] \]

- Consider \((s', a')\) which are rare under \(\mu(s, a)\)
- \(f(s', a')\) is unreliable when estimated from \(D\)
- Over-estimate of \(f(s', a')\) cascades to \(f(s, a)\) via Bellman backup
- Greedy policy update \(\pi_k\) picks \(a\) with erroneously over-estimated value
- If we gathered data using \(\pi_k\), we could correct over-estimates...

- Idea from Pessimism toy example: Conservative estimate in Bellman backup?
Algorithm: Pessimistic Policy Iteration

· Modify Bellman evaluation operator in policy iteration:

\[ T^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s', a'} [1(\hat{\mu}(s', a') \geq b)f(s', a')] \]

· \( \hat{\mu} \) is an approximate estimate of \( \mu(s, a) \)
· b controls the amount of pessimism
Pessimistic Policy Iteration

\[ \mathcal{T}^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P, a' \sim \pi} \left[ 1(\hat{\mu}(s', a') \geq b) f(s', a') \right] \]

1. Estimate \( \hat{\mu}(s, a) \) using \( D \)
2. \( f_{i+1} \leftarrow \hat{f}^{\pi_k} f_i \) till convergence to \( f^*_k \) \hspace{1cm} \text{[Evaluate } \pi_k \text{ in inner loop]} \]
3. \( \pi_{k+1}(s) = \arg\max_{a: \hat{\mu}(s, a) \geq b} f^*_k(s, a) \) \hspace{1cm} \text{[Update } \pi_{k+1} \text{ in outer loop]} \]

When \( \Pi \) does not contain all deterministic policies, use classification-based API to find \( \pi_{k+1} \in \Pi \)
Theoretical Result

We bound the error with the best policy in the following policy set:
{all policies such that $\Pr(\mu(s, a) < b|\pi) \leq \epsilon$}

Error bounds:

$$O\left(\frac{V_{\text{max}}}{(1-\gamma)^3 b} \sqrt{\frac{\ln(|\mathcal{F}||\Pi|/\delta)}{n}}\right) + \frac{V_{\text{max}}\epsilon}{1-\gamma}$$

$$\pi: \Pr(\mu(s, a) < b|\pi) \leq \epsilon$$
Practical Considerations

- Three strategies to set $b$:
  - If $n_0$ is #samples we think we need for a reliable estimate at any $(s, a)$, set $b \leftarrow \frac{n_0}{n}$
  - Set $b$ to a percentile threshold of $\hat{\mu}(s, a)$
  - Post-hoc diagnostic on returned $\pi^*$: if $\Pr(\hat{\mu}(s, a) < b | \pi)$ is too large, reduce $b$ and re-run PPI

- Many approaches for density/distribution estimation $\hat{\mu}(s, a)$
  - Our theory only assumes that $\|\hat{\mu} - \mu\|_{TV} \leq \epsilon_{\mu}$. Standard max likelihood gives $\epsilon_{\mu} \sim \frac{1}{\sqrt{n}}$
Prior Batch RL results

- Pessimism in batch RL [1,2] estimate dynamics $\hat{\gamma}(s'|s,a)$ which can be harder to do than Q-functions [rather than $\hat{\mu}(s,a)$]

- Pessimism in batch RL [3,4] censor $\mu(a|s)$ which is easy to do but insufficient [rather than $\mu(s,a)$]

- We recover the same bounds as assuming concentrability

Pessimism works; prior Batch RL algos fail.
Experiments in Cartpole (Discrete $s, a$ space)

Actions: Move left/right
Goal: Stay upright as long as possible

Even if $\mu$ close to optimal,
batch RL algos known to be unstable

Data ($n = 10^4$) from $\epsilon$-Greedy policy; $\hat{\mu}$ using counts
Experiments in MuJoCo (Continuous $s, a$ space)

Data ($n = 10^5$) from imperfect $\mu$; $\hat{\mu}$ using VAE density estimation

Only difference from BCQ: censor also based on $\hat{\mu}(s) \rightarrow$ Much more stable!
Experiments on D4RL (batch RL benchmark)

<table>
<thead>
<tr>
<th>D4RL datasets name</th>
<th>MBS-QL</th>
<th>BCQ</th>
<th>BEAR</th>
<th>BEAR (a recent new implementation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hopper-medium</td>
<td>75.2</td>
<td>54.5</td>
<td>47.6</td>
<td>52.1</td>
</tr>
<tr>
<td>HalfCheetah-medium</td>
<td>38.4</td>
<td>40.7</td>
<td>38.6</td>
<td>41.7</td>
</tr>
<tr>
<td>Walker2d-medium</td>
<td>68.1</td>
<td>53.1</td>
<td>33.2</td>
<td>59.1</td>
</tr>
</tbody>
</table>

- Code: [https://github.com/yaoliucs/PQL](https://github.com/yaoliucs/PQL)
Take-aways

- Less pessimistic Bellman backup?
- Combine strengths of model-based and model-free batch RL?
- Scale $\hat{\mu}$ to high-dimensional state spaces
Take-aways

- Short term proxies -> long term rewards? Trade-offs?
- Partial observability/exogenous variation/unobserved confounding
- Strategic exploration?
Take-aways

- Mechanisms, not policies! Users co-evolve with system!
- “Cold-start” without a production system $\mu$?
- Are we optimizing the wrong thing?

- What are the implications for marketing applications?

Thanks!  adswamin@microsoft.com