Towards Faster Learning of Good Decisions

Emma Brunskill
Carnegie Mellon University
Work described down in collaboration with:
Daniel Guo, Dexter Lee, Lihong Li, Yun-En Liu, Travis Mandel, Joseph Runde, Zoran Popovic
Want autonomous agents that act well
(make good sequences of decisions)
Renewable resource allocation

Marketing

Machine repair

Server job scheduling

Automated customer support

Inventory ordering

Renewable resource allocation

Marketing

Machine repair

Server job scheduling

Automated customer support

Inventory ordering
Most AI algorithms developed for robots
Want algorithms to enable:
Agents Making Decisions as Interact with People
Data = People
Towards Faster Learning

• Transfer learning
• Building on offline data
• Leveraging expert input
Formalizing Sample Efficiency of RL Algorithms

• One measure: is it Probably Approximately Correct?
  – Makes good decisions on all but the sample complexity number of steps
  – Sample complexity is polynomial function of problem parameters
  – $E^3$ (Kearns & Singh), R-MAX (Brafman & Tennenholtz)
Sample complexity:
number of actions may choose whose value is potentially far from optimal action’s value (informally: # of mistakes made by algorithm)

Unfortunately, this can be a lot

E.g. (Azar et al. 2013) lower bound

\[ O\left(\frac{SA\log(SA/\delta)}{(1-\gamma)^3 \epsilon^2}\right) \]

Reality check:
\[ \epsilon=0.1, \gamma=0.9: 10^5 \text{ samples per state} \]
\[ \epsilon=0.1, \gamma=0.99: 10^8 \text{ samples per state} \]
Approach: Share Knowledge

• Leverage provided information
  – Given input policy set (regret guarantees relative to best input policy, ECML 2013)
  – Given finite model sets (AAMAS 2012)

• Learn and transfer useful knowledge
Transfer / Multitask / Lifelong Learning

Each task involves a sequence of decisions.
Leverage related tasks to improve performance.
Transfer: Fundamental Questions

• Can learning be sped up across multiple tasks?
• Can computational costs be reduced when doing multiple tasks?
• Is different behavior optimal when an agent is maximizing performance across a set of tasks?

* task = reinforcement learning in a MDP
Sample complexity:
number of actions may choose whose value is potentially far from optimal action’s value

Can sample complexity get smaller by leveraging prior tasks?
Multi-Task Reinforcement Learning with a Finite Set of Models

Sample a task from finite set of MDPs

Brunskill & Li, UAI 2013
Multi-Task Reinforcement Learning with a Finite Set of Models

Act in it for $H$ steps

$<s_1,a_1,r_1,s_2,a_2,r_2,s_3,\ldots,s_H>$
Multi-Task Reinforcement Learning with a Finite Set of Models
Multi-Task Reinforcement Learning with a Finite Set of Models

Act in it for $H$ steps

$\langle s_1, a_1, r_1, s_2, a_2, r_2, s_3, a_3, \ldots, s_H \rangle$
Multi-Task Reinforcement Learning with a Finite Set of Models

Series of tasks
Act in each task for H steps
If Knew Identity of Each Task and the MDP Parameters, No Learning Needed!

Series of tasks
Act in each task for H steps
But Don’t Know the Identity of Each Task (which MDP it is)
And Don’t Know the MDP Model Parameters
Two Key Questions

1) Is learning faster than single-task RL algorithms if known models, but not identity of current task?
Two Key Questions

1) Is learning faster than single-task RL algorithms if know models, but not identity of current task?

2) If yes, can we achieve similar results even if start off not knowing the MDP model parameters?
Idea: Model Identification Can be Easier than Learning Policy From Scratch

- Assume know task is 1 of $M$ MDPs, where parameters of each MDP is specified
- Identify current task
  - Track which of $M$ models are most likely given observed tuples (state, action, reward, next state)
- Proved sample complexity reduction from $|S|/|A|$ → $M$ dependence
Two Key Questions

1) Is learning faster than single-task RL algorithms if know models, but not identity of current task?

2) If yes, can we achieve similar results even if start off not knowing the MDP model parameters?
Learning Set of MDPs’ Models

<\text{s}_{11}^{1}, \text{a}_{11}^{1}, \text{r}_{11}^{1}, \text{s}_{12}^{1}, \text{a}_{12}^{1}, \ldots, \text{s}_{1H}^{1}>$

$\rightarrow$

$\rightarrow$

<\text{s}_{21}^{2}, \text{a}_{21}^{2}, \text{r}_{21}^{2}, \text{s}_{22}^{2}, \text{a}_{22}^{2}, \ldots, \text{s}_{2H}^{2}>$

<\text{s}_{31}^{3}, \text{a}_{31}^{3}, \text{r}_{31}^{3}, \text{s}_{32}^{3}, \text{a}_{32}^{3}, \ldots, \text{s}_{3H}^{3}>$

<\text{s}_{41}^{4}, \text{a}_{41}^{4}, \text{r}_{41}^{4}, \text{s}_{42}^{4}, \text{a}_{42}^{4}, \ldots, \text{s}_{4H}^{4}>$

MDP R
$T_{R}, R_{R}$

MDP Y
$T_{Y}, R_{Y}$

MDP G
$T_{G}, R_{G}$
Learning Set of MABs’ Models

\[<s_{11}', a_{11}', r_{11}', s_{12}', a_{12}', r_{12}', s_{13}', a_{13}', \ldots s_{1H}'>\]

\[<s_{21}', a_{21}', r_{21}', s_{22}', a_{22}', r_{22}', s_{23}', a_{23}', \ldots s_{2H}'>\]

\[<s_{31}', a_{31}', r_{31}', s_{32}', a_{32}', r_{32}', s_{33}', a_{33}', \ldots s_{3H}'>\]

\[<s_{41}', a_{41}', r_{41}', s_{42}', a_{42}', r_{42}', s_{43}', a_{43}', \ldots s_{4H}'>\]

\[\text{MDP } R\]
\[T_R', R_R\]

\[\text{MDP } Y\]
\[T_Y', R_Y\]

\[\text{MDP } G\]
\[T_G', R_G\]
Identity of Each Task Unknown to the Learner: Latent Variable Estimation

- Latent Variable Estimation
  - Observed
    - Underlying MAB identity
      - \(<s_{11}'a_{11}', r_{11}', s_{12}', a_{12}', \ldots s_{1H}'>\)
      - \(<s_{21}'a_{21}', r_{21}', s_{22}', a_{22}', \ldots s_{2H}'>\)
      - \(<s_{31}'a_{31}', r_{31}', s_{32}', a_{32}', \ldots s_{3H}'>\)
      - \(<s_{41}'a_{41}', r_{41}', s_{42}', a_{42}', \ldots s_{4H}'>\)

- MDP R
  - \(T_{R'}, R_R\)
- MDP Y
  - \(T_Y, R_Y\)

- MDP G
  - \(T_G, R_G\)

Latent variable: Underlying MAB identity
Latent Variable Estimation

• Generally hard
• Standard techniques like expectation maximization have no finite sample guarantees & can converge to local optima
Assume Have (Sufficiently) Long Horizon Per Task

- If know all of the MDP parameters of each task perfectly, cluster tasks with identical parameters
- Know different underlying MDPs have to differ in their model parameters in at least one (s,a) pair by some difference $d$, so use to inform how well need to estimate parameters
Assume Have (Sufficiently) Long Horizon Per Task*

* In other work on transfer across multi-armed bandits, we relax this assumption, and use a Methods of Moments approach to derive & use finite sample bounds on latent variable estimation quality (see NIPS 2013)
Sequential Multi-Task PAC RL in a Finite Set of Models

• For tasks $i=1:T_1$
  – Use single-task PAC RL algorithm $E^3$ in task $i$
• Cluster tasks into set of $C$ MDPs
• For all further tasks $i$
  – Do model identification on $i$ given $C$ models
Key Result

- Can significantly speed learning if \# MDPs $<|S||A|$  
- First formal result, to our knowledge, that sequential multi-task reinforcement learning can enable faster learning (reduced sample complexity)  
- No negative transfer in terms of sample complexity!  
  - Negative transfer is when transfer can lead to worse results than if did non-transfer setting  
  - Here have no such issues, at least in terms of theoretical analysis
Concurrent Reinforcement Learning

Guo and Brunskill, AAAI 2015
Class of Students

Or all customers using Amazon, or patients in a hospital, …
Concurrent but Independent

One agent doing many tasks at once

Not multiple agents doing a single task

• Very little prior work on concurrent RL, except encouraging empirical paper that might be very useful for customers (Silver et al. 2013)
Concurrent RL

• Best could hope for: linear improvement
• Result is quite close to this!
Concurrent RL in a Finite Set of MDPs: Algorithm

• For $t=1:T$
  – Explore state-action space in each MDP

• Cluster tasks

• Run concurrent MBIE

• Similar to sequential task, but now doing clustering and sharing while acting as act in a single task
Concurrent RL in a Finite Set of MDPs: Intuition

• If time to cluster is small relative to experience needed to learn a good policy

• Then get approximately linear speedup (in terms of sample complexity) over not sharing information
Building on Offline Data
Off Policy Reinforcement Learning

Data gathered using previous policy (could be stochastic policy or multiple policies)

Want to output an optimal or good policy for future use
Increasing Amount of Decision Data, Increasing Opportunity for Better New Policies

- Electronic medical record systems
- Massive open online classes, tutoring systems
- Consumer marketing
- Home energy monitoring
Want Good Estimates of Generalization Performance
Challenge:
Use old data to figure out good policies to deploy

Mandel, Liu, Brunskill & Popović, AAMAS 2014
State Representation

- Vector of feature values
  - \(<\text{NumberHintsRequested}=1, \text{ProblemsSinceHintRequested}=1, \text{TotalElapsedTime}=56\text{s}, \text{NumberTimesGotCorrectWithoutHint}=1, \text{TotalNumberOfProblemsDone}=2, \ldots>\)
State Representation

- Vector of feature values
  - \(<\text{NumberHintsRequested}=1, \text{ProblemsSinceHintRequested}=1, \text{TotalElapsedTime}=56\text{s}, \text{NumberTimesGotCorrectWithoutHint}=1, \text{TotalNumberOfProblemsDone}=2, \ldots>\>

- Probability distribution over latent state
  - \(\text{Prob}(\text{GraphicalComparison \& SymbolicComparison}) = 0.27, \text{Prob}(\text{GraphicalComparison \& Not SymbolicComparison}) = .63, \text{Prob}(\text{Not GraphicalComparison \& SymbolicComparison}) = 0.03, \text{Prob}(\text{not GraphicalComparison \& Not SymbolicComparison}) = .07\)

Asked for a hint after 20s
Got correct after 40s
Got wrong after 15s
Got correct after 16s

Mandel, Liu, Brunskill & Popović, AAMAS 2014
State Representation

- **Vector of feature values**
  - \(<\text{NumberHintsRequested}=1, \text{ProblemsSinceHintRequested}=1, \text{TotalElapsedTime}=56s, \text{NumberTimesGotCorrectWithoutHint}=1, \text{TotalNumberOfProblemsDone}=2, \ldots>\>

- **Probability distribution over latent state**
  - \(\text{Prob}(\text{GraphicalComparison} \& \text{SymbolicComparison}) = 0.27, \text{Prob}(\text{GraphicalComparison} \& \neg \text{SymbolicComparison}) = 0.63, \text{Prob}(\neg \text{GraphicalComparison} \& \text{SymbolicComparison}) = 0.03, \text{Prob}(\neg \text{GraphicalComparison} \& \neg \text{SymbolicComparison}) = 0.07\)

- **Prediction over responses to future activities**
  - \(\text{Prob} \text{ get next graphical activity right } = 0.9, \text{Prob} \text{ get next improper fraction activity right } = 0.4\)

Mandel, Liu, Brunskill & Popović, AAMAS 2014
General Formulation: Unbiased Offline Policy Evaluation Across Representations for Short Horizons

activity_{11}', observation_{12}', reward_{12}', activity_{12}', observation_{13}', ...
activity_{21}', observation_{22}', reward_{22}', activity_{22}', observation_{23}', ...

Unbiased Estimate of Policy Performance
Guarantees

• Unbiased estimate of expected future performance of input representation—policies
  – Importance sampling to compare policies generated from variety of representations
  – Cross validation to predict generalization

Mandel, Liu, Brunskill & Popović, AAMAS 2014
Deployment: Refraction Game

• Find adaptive policy for concept to give to a player to maximize total number of concepts complete before quit
Deployment:
Refraction Game Offline Data

• 180 features of game per level
• Collected 11,000 players’ data using random level ordering
Used Offline Evaluation Method to Compare Many Representation-Policies

- Best policy in offline evaluation is adaptive policy using PCA+neural network representation
- Previously used expert ordering is estimated as being worse than random!
- Highlights non-trivial nature of designing good policies
... and Best Scoring Offline Policy Improved Concept Completion by 32%

- Tried 4 policies with 2000 new learners
- Compared to random & expert
Towards Faster Learning

• Transfer learning
• Building on offline data
• Leveraging expert input
Incorporate Heuristics
Utilize Prior Data
Evaluate Algorithms

Multi-Armed Bandits

Mandel, Liu, Brunskill & Popović, AAAI 2015
Overcome Delay

Evaluate Algorithms

Queue Method

Incorporate Heuristics

Utilize Prior Data

Multi-Armed Bandits

Mandel, Liu, Brunskill & Popović, AAAI 2015
Queue Method

• Store data (outcomes of pulling arms) in queues, one per arm
• Update a base algorithm (often which has formal performance guarantees) only using queues
  • Formal bounds
• Sampling distribution
Leveraging Heuristics While Maintaining Guarantees

• Given
  • Base algorithm (w/formal performance guarantees)
  • Heuristic algorithm
• Store data (outcomes of pulling arms) in queues, one per arm
• Update base algorithm only using queue data
• Sampling distribution: mixes heuristic and base but bounds lengths of queues
Evaluate Algorithms Using Prior Data

• Store prior data (outcomes of pulling arms) in queues, one per arm
• Draw outcome from queue according to action requested by algorithm to evaluate
• Update algorithm given queue outcome
• Halt when reach an empty queue
Evaluate Algorithms Using Prior Data

• Store prior data (outcomes of pulling arms) in queues, one per arm
• Draw outcome from queue according to action requested by algorithm to evaluate
• Update algorithm given queue outcome
• Halt when reach an empty queue
• More efficient than rejection sampling
• Conditionally unbiased*
Regret Bounds

**Theorem 1.** For algorithm 1 with any choice of procedure \textsc{getSamplingDist} and any online bandit algorithm \textsc{base}, $\mathbb{E} [R_T] \leq \mathbb{E} [R_T^{\textsc{base}}] + \sum_{i=1}^{N} \Delta_i \mathbb{E} [S_{i,T}]$ where $S_{i,T}$ is the number of elements pulled for arm $i$ by time $T$, but not yet shown to \textsc{base}. 
Blue walls are slippery!
Representation of Fractions?

Symbolic target?

Additional Motivation?

Music?
Offline Data Efficiency
Algorithm Performance Evaluated Using Offline Data
Queue Method

- Overcome Delay
- Incorporate Heuristics/Expert Input
- Utilize Prior Data
- Evaluate Algorithms

Multi-Armed Bandits

Mandel, Liu, Brunskill & Popović, AAAI 2015
Towards Faster Learning

- Transfer learning
- Building on offline data
- Leveraging expert input
Faster Learning With Policy Advice

- Online reinforcement learning
- Computational and speed of learning (sample complexity, regret) tend to scale with size of state/action space
- Policy advice: given finite set of policies
- Objective: perform as well as best policy

Azar, Lazaric, Brunskill, ECML 2013
RL with Policy Advice (RLPA)

\[ \pi_1 \quad \pi_2 \quad \pi_3 \]
RL with Policy Advice (RLPA)

- Keep upper bound on avg. reward per policy
- Use to optimistically select policy

Azar, Lazaric, Brunskill, ECML 2013
RLPA Regret Bounds

For any $T \geq T^+ = f^{-1}(H^+)$ the regret of RLPA is bounded as

$$\Delta(s) \leq f(T) \sqrt{Tm(\log(T/\delta))} + f(T)m(\log_2(T^+) + 2\log_2(T)))$$

$$\tilde{O}(\sqrt{mT}) \text{ (with } f(T) = \log(T), \ T^+ = \exp(H^+))$$

- No dependence on size of state-action space
- Sqrt dependence on $m$, # of input policies
- Dependent only on span $H^+$ of best policy

Azar, Lazaric, Brunskill, ECML 2013
Closing the Graveyard of Ambitions
Closing the Graveyard of Ambitions

<table>
<thead>
<tr>
<th></th>
<th>Celtics</th>
<th>Nets</th>
<th>Knicks</th>
<th>76ers</th>
<th>Raptors</th>
<th>Bulls</th>
<th>Cavaliers</th>
<th>Pistons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rockets</td>
<td>9</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td>14</td>
<td>2</td>
<td>0</td>
<td>-3</td>
</tr>
<tr>
<td>Magic</td>
<td>17</td>
<td>11</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>7</td>
<td>-11</td>
<td>-17</td>
</tr>
<tr>
<td>Lakers</td>
<td>-17</td>
<td>-12</td>
<td>-1</td>
<td>8</td>
<td>-12</td>
<td>22</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Blazers</td>
<td>-4</td>
<td>-3</td>
<td>-18</td>
<td>-1</td>
<td>8</td>
<td>13</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Score Difference</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>[-20, -16]</td>
<td>3</td>
</tr>
<tr>
<td>[-15, -11]</td>
<td>3</td>
</tr>
<tr>
<td>[-10, -6]</td>
<td>9</td>
</tr>
<tr>
<td>[-5, -1]</td>
<td>7</td>
</tr>
<tr>
<td>[0, 4]</td>
<td>4</td>
</tr>
<tr>
<td>[5, 9]</td>
<td>5</td>
</tr>
<tr>
<td>[10, 14]</td>
<td>3</td>
</tr>
<tr>
<td>[15, 19]</td>
<td>1</td>
</tr>
<tr>
<td>[20, 24]</td>
<td>1</td>
</tr>
</tbody>
</table>

Difference in Game Points: 2007

- Randomly generated tables
- Dynamic histograms
- Adaptive Hints

Hint: Check your answer for the range [-10, -6], and [5, 9].
* Remember that we are counting the number of times that the score difference in a game falls within these ranges.
Sample Efficient Online RL

• State space size influences how quickly can learn a good policy

• What representation should we use?
  • One that enables us to represent good policy
Abstract from Demonstration

• Cobo et al. (2011, 2014)
• Leverage expert input
• Without being bounded by expert performance
• Identify which features used by experts
• Do RL on that feature space
Learning the Student Features Used to Decide How To Teach

6 features of student learning process better than 70 at predicting teacher’s decisions!
Towards Faster RL