Representations
in
Deep Reinforcement Learning

Pieter Abbeel
Open AI / Berkeley AI Research Lab

Many slides made in collaboration with John Schulman and Sergey Levine
Reinforcement Learning

[Figure source: Sutton & Barto, 1998]
Some Reinforcement Learning Success Stories

Kohl and Stone, 2004
Ng et al, 2004
Tedrake et al, 2005
Kober and Peters, 2009

Mnih et al, 2015 (A3C)
Silver et al, 2014 (DPG)
Lillicrap et al, 2015 (DDPG)

Schulman et al, 2016 (TRPO + GAE)
Levine*, Finn*, et al, 2016 (GPS)

Silver*, Huang*, et al, 2016 (AlphaGo)

John Schulman & Pieter Abbeel – OpenAI + UC Berkeley
RL Algorithms Landscape
RL Algorithms Landscape

Policy Optimization
- DFO / Evolution
- Policy Gradients
- Actor-Critic Methods

Dynamic Programming
- Policy Iteration
- Value Iteration
- Q-Learning

modified policy iteration
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration

- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks

- Meta learning
  - MAML
  - RL2
Policy Optimization

[Figure source: Sutton & Barto, 1998]
Policy Optimization

- Consider control policy parameterized by parameter vector $\theta$

$$\max_{\theta} \mathbb{E} \left[ \sum_{t=0}^{H} R(s_t) \mid \pi_\theta \right]$$

- Often stochastic policy class (smoothes out the problem):

$$\pi_\theta(u \mid s) : \text{probability of action } u \text{ in state } s$$

[Figure source: Sutton & Barto, 1998]
Likelihood Ratio Policy Gradient

We let $\tau$ denote a state-action sequence $s_0, u_0, \ldots, s_H, u_H$. We overload notation: $R(\tau) = \sum_{t=0}^{H} R(s_t, u_t)$.

$$U(\theta) = \mathbb{E}\left[\sum_{t=0}^{H} R(s_t, u_t) ; \pi_{\theta}\right] = \sum_{\tau} P(\tau; \theta) R(\tau)$$

In our new notation, our goal is to find $\theta$:

$$\max_{\theta} U(\theta) = \max_{\theta} \sum_{\tau} P(\tau; \theta) R(\tau)$$
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[ \nabla_{\theta} U(\theta) = \nabla_{\theta} \sum_{\tau} P(\tau; \theta) R(\tau) \]

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[ \nabla_{\theta} U(\theta) = \nabla_{\theta} \sum_{\tau} P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \nabla_{\theta} P(\tau; \theta) R(\tau) \]

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[ \nabla_{\theta} U(\theta) = \nabla_{\theta} \sum_{\tau} P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \nabla_{\theta} P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_{\theta} P(\tau; \theta) R(\tau) \]

[Aleksandrov, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta)R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[ \nabla_\theta U(\theta) = \nabla_\theta \sum_{\tau} P(\tau; \theta)R(\tau) \]

\[ = \sum_{\tau} \nabla_\theta P(\tau; \theta)R(\tau) \]

\[ = \sum_{\tau} \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_\theta P(\tau; \theta)R(\tau) \]

\[ = \sum_{\tau} P(\tau; \theta) \frac{\nabla_\theta P(\tau; \theta)}{P(\tau; \theta)} R(\tau) \]

[Aleksandrov, Sysloyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glenn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[
\nabla_\theta U(\theta) = \nabla_\theta \sum_{\tau} P(\tau; \theta) R(\tau)
\]

\[ = \sum_{\tau} \nabla_\theta P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_\theta P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} P(\tau; \theta) \frac{\nabla_\theta P(\tau; \theta)}{P(\tau; \theta)} R(\tau) \]

\[ = \sum_{\tau} P(\tau; \theta) \nabla_\theta \log P(\tau; \theta) R(\tau) \]
Likelihood Ratio Policy Gradient

\[ U(\theta) = \sum_{\tau} P(\tau; \theta) R(\tau) \]

Taking the gradient w.r.t. \( \theta \) gives

\[ \nabla_\theta U(\theta) = \nabla_\theta \sum_{\tau} P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \nabla_\theta P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} \frac{P(\tau; \theta)}{P(\tau; \theta)} \nabla_\theta P(\tau; \theta) R(\tau) \]

\[ = \sum_{\tau} P(\tau; \theta) \nabla_\theta \log P(\tau; \theta) R(\tau) \]

Approximate with the empirical estimate for \( m \) sample paths under policy \( \pi^*_\theta \):

\[ \nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)}) \]

[Alekseev, Sysoyev, & Shemeneva, 1968]
[Rubinstein, 1969]
[Glynn, 1986]
[Reinforce, Williams 1992]
[GPOMDP, Baxter & Bartlett, 2001]
Likelihood Ratio Gradient: Validity

\[ \nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_{\theta} \log P(\tau^{(i)}; \theta) R(\tau^{(i)}) \]

- Valid even if R is discontinuous, and unknown, or sample space (of paths) is a discrete set
Likelihood Ratio Gradient: Intuition

$$\nabla U(\theta) \approx \hat{g} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^{(i)}; \theta) R(\tau^{(i)})$$

- Gradient tries to:
  - Increase probability of paths with positive R
  - Decrease probability of paths with negative R

! Likelihood ratio changes probabilities of experienced paths, does not try to change the paths
Let’s Decompose Path into States and Actions

\[ \nabla_\theta \log P(\tau^{(i)}; \theta) = \nabla_\theta \log \prod_{t=0}^{H} P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)}) \cdot \pi_\theta(u_t^{(i)}|s_t^{(i)}) \]
Let’s Decompose Path into States and Actions

\[ \nabla_\theta \log P(\tau^{(i)}; \theta) = \nabla_\theta \log \left[ \prod_{t=0}^{H} P(s_{t+1}^{(i)} | s_t^{(i)}, u_t^{(i)}) \cdot \pi_\theta (u_t^{(i)} | s_t^{(i)}) \right] \]

\[ = \nabla_\theta \left[ \sum_{t=0}^{H} \log P(s_{t+1}^{(i)} | s_t^{(i)}, u_t^{(i)}) + \sum_{t=0}^{H} \log \pi_\theta (u_t^{(i)} | s_t^{(i)}) \right] \]
Let’s Decompose Path into States and Actions

\[ \nabla_\theta \log P(\tau^{(i)}; \theta) = \nabla_\theta \log \left[ \prod_{t=0}^{H} \left( \frac{P(s_{t+1}^{(i)}|s_{t}^{(i)}, u_{t}^{(i)}) \cdot \pi_{\theta}(u_{t}^{(i)}|s_{t}^{(i)})}{\text{dynamics model}} \cdot \text{policy} \right) \right] \\
= \nabla_\theta \left[ \sum_{t=0}^{H} \log P(s_{t+1}^{(i)}|s_{t}^{(i)}, u_{t}^{(i)}) + \sum_{t=0}^{H} \log \pi_{\theta}(u_{t}^{(i)}|s_{t}^{(i)}) \right] \\
= \nabla_\theta \sum_{t=0}^{H} \log \pi_{\theta}(u_{t}^{(i)}|s_{t}^{(i)}) \]
Let’s Decompose Path into States and Actions

\[ \nabla_\theta \log P(\tau^{(i)}; \theta) = \nabla_\theta \log \left[ \prod_{t=0}^{H} P(s^{(i)}_{t+1} | s^{(i)}_t, u^{(i)}_t) \cdot \pi_\theta(u^{(i)}_t | s^{(i)}_t) \right] \]

\[ = \nabla_\theta \left[ \sum_{t=0}^{H} \log P(s^{(i)}_{t+1} | s^{(i)}_t, u^{(i)}_t) + \sum_{t=0}^{H} \log \pi_\theta(u^{(i)}_t | s^{(i)}_t) \right] \]

\[ = \nabla_\theta \sum_{t=0}^{H} \log \pi_\theta(u^{(i)}_t | s^{(i)}_t) \]

\[ = \sum_{t=0}^{H} \nabla_\theta \log \pi_\theta(u^{(i)}_t | s^{(i)}_t) \]

no dynamics model required!!
Likelihood Ratio Gradient Estimate

\[ \hat{g} = \frac{1}{m} \sum_{k=1}^{m} \nabla_\theta \log P(\tau^{(k)}; \theta) R(\tau^{(k)}) \]
Likelihood Ratio Gradient Estimate

\[ \hat{g} = \frac{1}{m} \sum_{k=1}^{m} \nabla_\theta \log P(\tau^{(k)}; \theta) R(\tau^{(k)}) \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(u_t^{(k)} | s_t^{(k)}) R(\tau^{(k)}) \]
Likelihood Ratio Gradient Estimate

\[ \hat{g} = \frac{1}{m} \sum_{k=1}^{m} \nabla_{\theta} \log P(\tau^{(k)}; \theta) R(\tau^{(k)}) \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta}(u_{t}^{(k)} | s_{t}^{(k)}) R(\tau^{(k)}) \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta}(u_{t}^{(k)} | s_{t}^{(k)}) \sum_{t'=t}^{H-1} r_{t'}^{(k)} \]
Likelihood Ratio Gradient Estimate

\[ \hat{g} = \frac{1}{m} \sum_{k=1}^{m} \nabla_{\theta} \log P(\tau^{(k)}; \theta) R(\tau^{(k)}) \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta}(u_{t}^{(k)} | s_{t}^{(k)}) R(\tau^{(k)}) \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta}(u_{t}^{(k)} | s_{t}^{(k)}) \sum_{t'=t}^{H-1} r_{t'}^{(k)} \]

\[ = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta}(u_{t}^{(k)} | s_{t}^{(k)}) \left( \sum_{t'=t}^{H-1} r_{t'}^{(k)} - b(s_{t'}^{(k)}) \right) \]

\[ b(s_{t'}^{(k)}) = \frac{1}{m} \sum_{k=1}^{m} \sum_{t'=t}^{H-1} r_{t'}^{(k)} \]
Likelihood Ratio Policy Gradient

- Init $\pi \theta_0$
- Collect data $\{s, u, s', r\}$
- $\theta_{i+1} \leftarrow \theta_i + \alpha \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_{\theta_i}(u_t^{(k)}|s_t^{(k)}) \left( \sum_{t'=t}^{H-1} r_{t'}^{(k)} - b(s_t^{(k)}) \right)$

$$b(s_t^{(k)}) = \frac{1}{m} \sum_{k=1}^{m} \sum_{t'=t}^{H-1} r_{t'}^{(k)}$$

$\rightarrow$ Increase logprob of action proportionally to how much its returns are better than the expected return under the current policy

- Can we get a better $b$? Yes! $V^{\pi}$ $[\rightarrow \text{“actor-critic”}]$
Estimation of $V^\pi$

- Bellman Equation for $V^\pi$

  $$V^\pi(s) = \sum_u \pi(u|s) \sum_{s'} P(s'|s,u)[R(s,u,s') + \gamma V^\pi(s')]$$

- Fitted V iteration:
  - Init $V^\pi_{\phi_0}$
  - Collect data $\{s, u, s', r\}$
  - $\phi_{i+1} \leftarrow \min_{\phi} \sum_{(s,u,s',r)} \|r + V^\pi_{\phi_i}(s') - V^\pi_{\phi}(s)\|_2^2 + \lambda \|\phi - \phi_i\|_2^2$
Actor-Critic

- Policy Gradient + Fitted V iteration:
  - Init $\pi \theta_0$, $V^{\pi}_{\phi_0}$
  - Collect data \{s, u, s', r\}
  - $\phi_{i+1} \leftarrow \min_{\phi} \sum_{(s,u,s',r)} \|r + V^{\pi}_{\phi_i}(s') - V_{\phi}(s)\|_2^2 + \lambda \|\phi - \phi_i\|_2^2$
  - $\theta_{i+1} \leftarrow \theta_i + \alpha \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_{\theta} \log \pi_{\theta_i}(u_t^{(k)}|s_t^{(k)}) \left( \sum_{t'=t}^{H-1} r_{t'}^{(k)} - V^{\pi}_{\phi_i}(s_t^{(k)}) \right)$

- Generalized Advantage Estimation (Schulman et al, 2016) also uses learned value function to better estimate first term $\sum_{t'=t}^{H-1} r_{t'}^{(k)}$
Bellman equation for Q*:

\[ Q^*(s, u) = \sum_{s'} P(s' \mid s, u) \left[ R(s, u, s') + \max_{u'} Q^*(s', u') \right] \]

Fitted Q iteration:
- Init \( Q_{\phi_0} \)
- Collect data \( \{s, u, s', r\} \)
- \( \phi_{i+1} \leftarrow \min_{\phi} \sum_{s,u,s',r} \| r + \max_{u'} Q_{\phi_i}(s', u') - Q_{\phi}(s, u) \|_2^2 + \lambda \| \phi - \phi_i \|_2^2 \)
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration

- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks

- Meta learning
  - MAML
  - RL2
DQN

Conv1: 32 8x8 filters w/stride 4
Conv2: 64 4x4 filters with stride 2
Conv3: 64 3x3 filters with stride 1

[Mnih et al, Nature 2015]
DQN

[Mnih et al, Nature 2015]
TRPO + GAE  [Schulman et al, 2015]
A3C

[Mnih et al, 2016]
Dueling Network

\[ Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + \left( A(s, a; \theta, \alpha) - \frac{1}{|A|} \sum_{a'} A(s, a'; \theta, \alpha) \right) \]
Feedforward: h100 – h50 – h25 for both policy and value function
Learning Locomotion (TRPO + GAE)

Iteration 0

[Schulman, Moritz, Levine, Jordan, Abbeel, 2016]
Real Robots

[Levine*, Finn*, Darrell, Abbeel, JMLR 2016]
Guided Policy Search [Levine, Finn, Darrell, Abbeel, JMLR 2016]
Learned Skills

[Levine*, Finn*, Darrell, Abbeel, JMLR 2016]
A3C with LSTMs [Mnih et al, 2016]

Memory [Oh et al, 2016]

Auxiliary Losses

Unreal Agent [Jaderberg, Mnih, Czarnecki et al, 2016]
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration
- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks
- Meta learning
  - RL2
  - MAML
Exploration through Reward Bonuses

- Idea: infuse additional reward for encountering novelty / learning something about the environment
  - VIME: Bayesian Neural Net – Houthooft et al, 2016
  - Pixel-CNN density estimation – Ostrovski et al, 2017
  - ...

Pieter Abbeel -- UC Berkeley / OpenAI / Gradescope
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration
- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks
- Meta learning
  - RL2
  - MAML
Value Iteration Networks

[Tamar, Wu, Thomas, Levine, Abbeel, NIPS 2016]
Value Iteration Network (VIN)

VIN Architecture:

- Observation: $\phi(s)$
- Attention: $\psi(s)$
- Reactive Policy: $\pi_{re}(a|\phi(s), \psi(s))$
- Plan on MDP $\bar{M}$
- VI Module: $\bar{R}$, $\bar{P}$
- End-to-end differentiable planner!

Pieter Abbeel – OpenAI / UC Berkeley / Gradescope
VIN -- Evaluation

<table>
<thead>
<tr>
<th>Network</th>
<th>Train Error</th>
<th>Test Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIN</td>
<td>0.30</td>
<td>0.35</td>
</tr>
<tr>
<td>CNN</td>
<td>0.39</td>
<td>0.58</td>
</tr>
</tbody>
</table>

Gridworld navigation

Mars navigation

Continuous domain

[Tamar, Wu, Thomas, Levine, Abbeel, NIPS 2016]
Continuous Domain -- Video
1st Person Mapping + Navigation with VIN

[Gupta, Davidson, Levine, Sukthankar, Malik, 2017]
Most Closely Related Work

- Embed to Control – Watter, Springenberg, Boedecker, Riedmiller, 2015
- Hindsight MPC – Aviv Tamar et al., ICRA 2017
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration

- Different Approaches / Architectures
  - Value Iteration Networks
  - *Predictron*
  - *Modular Networks*
  - *Option-Critic*
  - *Feudal Networks*
- Meta learning
  - RL2
  - MAML
Predictron

[Silver et al, 2016]
Predictron

[Silver et al, 2016]

Pieter Abbeel -- UC Berkeley / OpenAI / Gradescope
Modular Networks

[Devin, Gupta et al, 2016]
Robot 1: 3 DoF arm
Robot 2: 4 DoF arm
Task 1: Opening a drawer
Task 2: Pushing a block

[Devin, Gupta et al, 2016]
Robot 1

Robot 2

Task 1

Task 2

[Devin, Gupta et al, 2016]
Robot 1

Robot 2

Task 1

Task 2

Task 2

[Devin, Gupta et al, 2016]
Robot 1

Robot 2

Task 1

Task 2

[Devin, Gupta et al, 2016]
Robot 1

Robot 2

Task 1

Task 2

[Devin, Gupta et al, 2016]
[Devin, Gupta et al, 2016]
Option-Critic Architecture

[Methods and equitations from Bacon, Harb, Precup, 2017]
Feudal Networks

$$\nabla g_t = A_t^M \nabla_\theta d_{cos}(s_{t+c} - s_t, g_t(\theta))$$

$$\nabla \pi_t = A_t^D \nabla_\theta \log \pi(a_t|x_t; \theta)$$

[Dayan and Hinton, 1993; Vezhnevets et al, 2017]
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration

- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks

- Meta learning
  - MAML
  - RL2
Learning useful representations with deep learning

Where are the “ImageNet” features of motor control?
The trouble with RL

- Large-scale
- Emphasizes diversity
- Evaluated on generalization

- Small-scale
- Emphasizes mastery
- Evaluated on performance
- Can we force RL to generalize?
Multi-task training for adaptability

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} R(\theta)$$

$$\sum_i L_i(\theta + \alpha \nabla_{\theta} R_i(\theta))$$

[Finnt, Abbeel, Levine, 2017]
Model-agnostic meta-learning: forward/backward locomotion

after MAML training

after 1 gradient step (forward reward)

after 1 gradient step (backward reward)
Model-agnostic meta-learning benchmark results

Omniglot Few-Shot Classification

Omniglot Dataset: 1200 training classes, 423 test classes

Model-agnostic meta-learning: forward/backward locomotion

- After MAML training
- After 1 gradient step (backward reward)
- After 1 gradient step (forward reward)
Talk Outline

- Classical RL
  - Algorithms
    - Policy Gradients
    - Actor-Critic
    - Q-learning
  - Representation
- Representation in exploration

- Different Approaches / Architectures
  - Value Iteration Networks
  - Predictron
  - Modular Networks
  - Option-Critic
  - Feudal Networks

- *Meta learning*
  - MAML
  - *RL2*
## Speed of Learning

### Deep RL (DQN) vs. Human

<table>
<thead>
<tr>
<th></th>
<th>Score: 18.9</th>
<th>Score: 9.3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>#Experience</strong> measured in real time:</td>
<td>40 days</td>
<td>2 hours</td>
</tr>
<tr>
<td></td>
<td><strong>“Slow”</strong></td>
<td><strong>“Fast”</strong></td>
</tr>
</tbody>
</table>

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]

Pieter Abbeel – OpenAI / UC Berkeley / Gradescope
Starting Observations

- TRPO, DQN, A3C are fully general RL algorithms
  - i.e., for any MDP that can be mathematically defined, these algorithms are equally applicable

- MDPs encountered in real world
  = tiny, tiny subset of all MDPs that could be defined

- Can we design “fast” RL algorithms that take advantage of such knowledge?

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Research Questions

- How to acquire a good prior for real-world MDPs?
  - Or for starters, e.g., for real-games MDPs?

- How to design algorithms that make use of such prior information?

Key idea: Learn a fast RL algorithm that encodes this prior

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Formulation

- Given: Distribution over relevant MDPs
- Train the fast RL algorithm to be fast on a training set of MDPs

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Learning the Fast RL Algorithm

- Representation of the fast RL algorithm:
  - RNN = generic computation architecture
  - different weights in the RNN means different RL algorithm
  - different activations in the RNN means different current policy

- Training setup:

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Formulation

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Alternative View on RL2

- RNN = policy for acting in a POMDP
  - Part of what’s not observed in the POMDP is which MDP the agent is in

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Related Work

- Wang et al., (2016) Learning to Reinforcement Learn, in submission to ICLR 2017,
- Chen et al. (2016) Learning to Learn for Global Optimization of Black Box Functions
- Andrychowicz et al., (2016) Learning to learn by gradient descent by gradient descent
- Santoro et al., (2016) One-shot Learning with Memory-Augmented Neural Networks
- Younger et al. (2001), Meta learning with backpropagation
- Schmidhuber et al. (1996), Simple principles of metalearning
Evaluation

- Multi-Armed Bandits
- Provably (asymptotically) optimal RL algorithms have been invented by humans: Gittins index, UCB1, Thompson sampling, ...

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]

Pieter Abbeel – OpenAI / UC Berkeley / Gradescope
## Evaluation

- Multi-Armed Bandits

<table>
<thead>
<tr>
<th>Setup</th>
<th>Random</th>
<th>Gittins</th>
<th>TS</th>
<th>OTS</th>
<th>UCB1</th>
<th>$\varepsilon$-Greedy</th>
<th>Greedy</th>
<th>$RL^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 10, k = 5$</td>
<td>5.0</td>
<td>6.6</td>
<td>5.7</td>
<td>6.5</td>
<td>6.7</td>
<td>6.6</td>
<td>6.6</td>
<td>6.7</td>
</tr>
<tr>
<td>$n = 10, k = 10$</td>
<td>5.0</td>
<td>6.6</td>
<td>5.5</td>
<td>6.2</td>
<td>6.7</td>
<td>6.6</td>
<td>6.6</td>
<td>6.7</td>
</tr>
<tr>
<td>$n = 100, k = 5$</td>
<td>5.1</td>
<td>6.5</td>
<td>5.2</td>
<td>5.5</td>
<td>6.6</td>
<td>6.5</td>
<td>6.5</td>
<td>6.8</td>
</tr>
<tr>
<td>$n = 100, k = 10$</td>
<td>49.9</td>
<td>78.3</td>
<td>74.7</td>
<td>77.9</td>
<td>78.0</td>
<td>75.4</td>
<td>74.8</td>
<td>78.7</td>
</tr>
<tr>
<td>$n = 100, k = 50$</td>
<td>49.9</td>
<td>82.8</td>
<td>76.7</td>
<td>81.4</td>
<td>82.4</td>
<td>77.4</td>
<td>77.1</td>
<td>83.5</td>
</tr>
<tr>
<td>$n = 500, k = 5$</td>
<td>49.8</td>
<td>85.2</td>
<td>64.5</td>
<td>67.7</td>
<td>84.3</td>
<td>78.3</td>
<td>78.0</td>
<td>84.9</td>
</tr>
<tr>
<td>$n = 500, k = 10$</td>
<td>249.8</td>
<td>405.8</td>
<td>402.0</td>
<td>406.7</td>
<td>405.8</td>
<td>388.2</td>
<td>380.6</td>
<td>401.6</td>
</tr>
<tr>
<td>$n = 500, k = 50$</td>
<td>249.0</td>
<td>437.8</td>
<td>429.5</td>
<td>438.9</td>
<td>437.1</td>
<td>408.0</td>
<td>395.0</td>
<td>432.5</td>
</tr>
</tbody>
</table>

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Evaluation

- Multi-Armed Bandits

\[ \text{[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]} \]

Pieter Abbeel – OpenAI / UC Berkeley / Gradescope
Evaluation: Tabular MDPs

- Provably (asymptotically) optimal algorithms:
  - BEB, PSRL, UCRL2, ...

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Evaluation: Tabular MDPs

<table>
<thead>
<tr>
<th>Setup</th>
<th>Random</th>
<th>PSRL</th>
<th>OPSRL</th>
<th>UCRL2</th>
<th>BEB</th>
<th>$\epsilon$-Greedy</th>
<th>Greedy</th>
<th>$RL^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 10$</td>
<td>100.1</td>
<td>138.1</td>
<td>144.1</td>
<td>146.6</td>
<td>150.2</td>
<td>132.8</td>
<td>134.8</td>
<td><strong>156.2</strong></td>
</tr>
<tr>
<td>$n = 25$</td>
<td>250.2</td>
<td>408.8</td>
<td>425.2</td>
<td>424.1</td>
<td>427.8</td>
<td>377.3</td>
<td>368.8</td>
<td><strong>445.7</strong></td>
</tr>
<tr>
<td>$n = 50$</td>
<td>499.7</td>
<td>904.4</td>
<td><strong>930.7</strong></td>
<td>918.9</td>
<td>917.8</td>
<td>823.3</td>
<td>769.3</td>
<td><strong>936.1</strong></td>
</tr>
<tr>
<td>$n = 75$</td>
<td>749.9</td>
<td>1417.1</td>
<td><strong>1449.2</strong></td>
<td>1427.6</td>
<td>1422.6</td>
<td>1293.9</td>
<td>1172.9</td>
<td>1428.8</td>
</tr>
<tr>
<td>$n = 100$</td>
<td>999.4</td>
<td>1939.5</td>
<td><strong>1973.9</strong></td>
<td>1942.1</td>
<td>1935.1</td>
<td>1778.2</td>
<td>1578.5</td>
<td>1913.7</td>
</tr>
</tbody>
</table>

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Evaluation: Tabular MDPs

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Evaluation: Visual Navigation

(built on top of ViZDoom)

Agent’s view  Small maze  Large maze

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]

Pieter Abbeel – OpenAI / UC Berkeley / Gradescope
Evaluation: Visual Navigation

Before learning

After learning

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
Evaluation

Figure 5: RL² learning curves for visual navigation. Each curve shows a different random initialization of the RNN weights. Performance varies greatly across different initializations.

[Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016]
OpenAI Universe
How to Learn More and Get Started?

- **(1) Deep RL Courses**
  
  - COMPM050/COMPGI13 Reinforcement Learning (UCL): [http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html](http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html) by David Silver
(2) Deep RL Code Bases

- **rllab**: [https://github.com/openai/rllab](https://github.com/openai/rllab)
  Duan, Chen, Houthooft, Schulman et al.

- **GPS**: [http://rll.berkeley.edu/gps/](http://rll.berkeley.edu/gps/)
  Finn, Zhang, Fu, Tan, McCarthy, Scharff, Stadie, Levine

  Geramifard, Klein, Dann, Dabney, How
How to Learn More and Get Started?

(3) Environments

- Arcade Learning Environment (ALE) (Bellemare et al, JAIR 2013)
- MuJoCo: [http://mujoco.org](http://mujoco.org) (Todorov)
- Minecraft (Microsoft)
- Deepmind Lab / Labyrinth (Deepmind)
- OpenAI Gym: [https://gym.openai.com/](https://gym.openai.com/)
- Universe: [https://universe.openai.com/](https://universe.openai.com/)
Current / Future Directions

- Faster learning / Hierarchy
  - Exploration (Stadie, Levine, Abbeel 2015; Houthooft, Duan, Chen, Schulman Abbeel, 2016)
  - Meta-learning: RL2 (Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel, 2016); MAML (Finn, Abbeel, Levine, 2017)

- Transfer learning
  - Modular networks (Devin, Gupta, Darrell, Abbeel, Levine, 2017); Invariant feature spaces (Gupta Devin, Liu, Abbeel, Levine, 2017)
  - Domain randomization (Tobin, Fong, Schneider, Zaremba, Abbeel, 2017)

- Safe learning

- Unsupervised / Semisupervised learning
  - InfoGAN (Chen, Duan, Houthooft, Schulman, Sutskever, Abbeel 2016), VLAE (Chen, Kigma, Salimans, Duan, Dhariwal, Schulman, Sutskever, Abbeel, 2017)
  - Semisupervised RL (Finn, Yu, Fu, Abbeel, Levine, 2017)

- Grounded language / Multi-agent
  - “Inventing” language (Mordatch & Abbeel, 2017)

- Imitation
  - First-person from VR Tele-op (McCarthy, Zhang, Jow, Lee, Goldberg, Abbeel, 2017)
  - Third-person (Stadie, Abbeel, Sutskever, 2017)

- Value alignment / AI Safety
  - Communication (Huang, Held, Abbeel, Dragan, 2017)