Randomized Prior Functions for Deep Reinforcement Learning

Ian Osband, John Aslanides, Albin Cassirer
Reinforcement Learning
Reinforcement Learning

Data & Estimation = Supervised Learning
Reinforcement Learning

Data & Estimation = Supervised Learning
+ partial feedback = Multi-armed Bandit
Reinforcement Learning

- Data & Estimation
- + partial feedback
- = Multi-armed Bandit
- + delayed consequences
- = Reinforcement Learning
Reinforcement Learning

• “Sequential decision making under uncertainty.”
Reinforcement Learning

- “Sequential decision making under uncertainty.”
Reinforcement Learning

• “Sequential decision making under uncertainty.”

• Three necessary building blocks:
Reinforcement Learning

- "Sequential decision making under uncertainty."

- Three necessary building blocks:
  1. **Generalization**
Reinforcement Learning

- “Sequential decision making under uncertainty.”

- Three necessary building blocks:
  1. Generalization
  2. Exploration vs Exploitation

Data & Estimation = Supervised Learning
+ partial feedback = Multi-armed Bandit
+ delayed consequences = Reinforcement Learning
Reinforcement Learning

- “Sequential decision making under uncertainty.”

- Three necessary building blocks:
  1. Generalization
  2. Exploration vs Exploitation
  3. Credit assignment
Reinforcement Learning

• “Sequential decision making under uncertainty.”

• Three necessary building blocks:
  1. Generalization
  2. Exploration vs Exploitation
  3. Credit assignment

Data & Estimation = Supervised Learning
+ partial feedback = Multi-armed Bandit
+ delayed consequences = Reinforcement Learning
Reinforcement Learning

- “Sequential decision making under uncertainty.”

- Three necessary building blocks:
  1. Generalization
  2. Exploration vs Exploitation
  3. Credit assignment

- As a field, we are pretty good at combining any 2 of these 3. ... but we need practical solutions that combine them all.
Reinforcement Learning

• “Sequential decision making under uncertainty.”

• Three necessary building blocks:
  1. Generalization
  2. Exploration vs Exploitation
  3. Credit assignment

• As a field, we are pretty good at combining any 2 of these 3. … but we need practical solutions that combine them all.

We need effective uncertainty estimates for Deep RL
Estimating uncertainty in deep RL
Estimating uncertainty in deep RL

Dropout sampling
Estimating uncertainty in deep RL

Dropout sampling

“Dropout sample \equiv posterior sample” (Gal+Gharamani 2015)
Estimating uncertainty in deep RL

- Dropout sampling
- "Dropout sample $\equiv$ posterior sample" (Gal+Gharamani 2015)
- Dropout rate does not concentrate with the data.
Estimating uncertainty in deep RL

Dropout sampling

“Dropout sample \(\approx\) posterior sample” (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even “concrete” dropout not necessarily right rate.
Estimating uncertainty in deep RL

**Dropout sampling**

“Dropout sample ≈ posterior sample” (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even “concrete” dropout not necessarily right rate.

**Variational inference**
Estimating uncertainty in deep RL

**Dropout sampling**

“Dropout sample \(\approx\) posterior sample” (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

**Variational inference**

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Even “concrete” dropout not necessarily right rate.
Estimating uncertainty in deep RL

**Dropout sampling**

"Dropout sample ≅ posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

**Variational inference**

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$

Uncertainty in $Q$ ➞ correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.
### Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Dropout sample = posterior sample&quot; (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Bellman error: $Q(s, a) = r + \gamma \max_\alpha Q(s', \alpha)$</td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td></td>
<td>Uncertainty in $Q \implies$ correlated TD loss.</td>
</tr>
<tr>
<td>Even &quot;concrete&quot; dropout not necessarily right rate.</td>
<td></td>
<td>VI on i.i.d. model does not propagate uncertainty.</td>
</tr>
</tbody>
</table>
Estimating uncertainty in deep RL

**Dropout sampling**

"Dropout sample \( \approx \) posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

**Variational inference**

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Bellman error:

\[
Q(s, a) = r + \gamma \max_{\alpha} Q(s', \alpha)
\]

Uncertainty in \( Q \) \( \rightarrow \) correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.

**Distributional RL**

Models Q-value as a distribution, rather than point estimate.
Estimating uncertainty in deep RL

- **Dropout sampling**
  - "Dropout sample is equivalent to posterior sample" (Gal+Gharamani 2015)
  - Dropout rate does not concentrate with the data.
  - Even "concrete" dropout not necessarily right rate.

- **Variational inference**
  - Apply VI to Bellman error as if it was an i.i.d. supervised loss.
  - Bellman error: $Q(s, a) = r + \gamma \max_{\alpha} Q(s', \alpha)$
  - Uncertainty in $Q \Rightarrow$ correlated TD loss.
  - VI on i.i.d. model does not propagate uncertainty.

- **Distributional RL**
  - Models Q-value as a distribution, rather than point estimate.
  - This distribution ≠ posterior uncertainty.
Estimating uncertainty in deep RL

**Dropout sampling**

"Dropout sample = posterior sample" (Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even "concrete" dropout not necessarily right rate.

**Variational inference**

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Bellman error: \( Q(s, a) = r + \gamma \max_a Q(s', a) \)

Uncertainty in \( Q \) ➔ correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.

**Distributional RL**

Models Q-value as a distribution, rather than point estimate.

This distribution ≠ posterior uncertainty.

Aleatoric vs Epistemic … it’s not the right thing for exploration.
## Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
<th>Count-based density</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Dropout sample = posterior sample” (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td></td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td>Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$</td>
<td>This distribution $\neq$ posterior uncertainty.</td>
<td></td>
</tr>
<tr>
<td>Even “concrete” dropout not necessarily right rate.</td>
<td>Uncertainty in Q → correlated TD loss.</td>
<td>Aleatoric vs Epistemic … it’s not the right thing for exploration.</td>
<td></td>
</tr>
</tbody>
</table>

Q(s, a) = r + γ \max_a Q(s', a)
### Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
<th>Count-based density</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Dropout sample ≈ posterior sample” (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td>Estimate number of “visit counts” to state, add bonus.</td>
</tr>
</tbody>
</table>

- Dropout rate does not concentrate with the data.
- Even “concrete” dropout not necessarily right rate.

- Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$
- Uncertainty in $Q$ → correlated TD loss.
- VI on i.i.d. model does not propagate uncertainty.

- This distribution ≠ posterior uncertainty.

- Aleatoric vs Epistemic… it’s not the right thing for exploration.
Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
<th>Count-based density</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Dropout sample = posterior sample” (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td>Estimate number of “visit counts” to state, add bonus.</td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td>Bellman error: $Q(s, a) = r + \gamma \max_\alpha Q(s', \alpha)$</td>
<td>This distribution != posterior uncertainty.</td>
<td>The “density model” has nothing to do with the actual task.</td>
</tr>
<tr>
<td>Even “concrete” dropout not necessarily right rate.</td>
<td>Uncertainty in Q ➔ correlated TD loss.</td>
<td>Aleatoric vs Epistemic</td>
<td>… it’s not the right thing for exploration.</td>
</tr>
<tr>
<td></td>
<td>VI on i.i.d. model does not propagate uncertainty.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dropout sampling</td>
<td>Variational inference</td>
<td>Distributional RL</td>
<td>Count-based density</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------------------</td>
<td>-------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>“Dropout sample = posterior sample”</td>
<td>Apply VI to Bellman</td>
<td>Models Q-value as</td>
<td>Estimate number of</td>
</tr>
<tr>
<td>(Gal+Gharamani 2015)</td>
<td>error as if it was an</td>
<td>a distribution,</td>
<td>“visit counts” to</td>
</tr>
<tr>
<td></td>
<td>i.i.d. supervised loss.</td>
<td>rather than point</td>
<td>state, add bonus.</td>
</tr>
<tr>
<td>Dropout rate does not concentrate</td>
<td>Bellman error:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>with the data.</td>
<td>$Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Even “concrete” dropout not</td>
<td>Uncertainty in $Q \Rightarrow$</td>
<td>This distribution $\neq$</td>
<td>The “density model”</td>
</tr>
<tr>
<td>necessarily right rate.</td>
<td>correlated TD loss.</td>
<td>posterior uncertainty.</td>
<td>has nothing to do</td>
</tr>
<tr>
<td></td>
<td>VI on i.i.d. model</td>
<td></td>
<td>with the actual task.</td>
</tr>
<tr>
<td></td>
<td>does not propagate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>uncertainty.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bellman error: $Q(s,a) = r + \gamma \max_{\alpha} Q(s', \alpha)$
Uncertainty in $Q \Rightarrow$ correlated TD loss.
VI on i.i.d. model does not propagate uncertainty.

Aleatoric vs Epistemic
... it's not the right thing for exploration.

With generalization, state “visit count” $\neq$ uncertainty.
Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
<th>Count-based density</th>
<th>Bootstrap ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Dropout sample = posterior sample&quot; (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td>Estimate number of &quot;visit counts&quot; to state, add bonus.</td>
<td></td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td>Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$</td>
<td>This distribution != posterior uncertainty.</td>
<td>The &quot;density model&quot; has nothing to do with the actual task.</td>
<td></td>
</tr>
<tr>
<td>Even &quot;concrete&quot; dropout not necessarily right rate.</td>
<td>Uncertainty in $Q$ (\Rightarrow) correlated TD loss.</td>
<td>Aleatoric vs Epistemic ... it's not the right thing for exploration.</td>
<td>With generalization, state &quot;visit count&quot; != uncertainty.</td>
<td></td>
</tr>
</tbody>
</table>
# Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th>Dropout sampling</th>
<th>Variational inference</th>
<th>Distributional RL</th>
<th>Count-based density</th>
<th>Bootstrap ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Dropout sample = posterior sample” (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td>Estimate number of “visit counts” to state, add bonus.</td>
<td>Train ensemble on noisy data - classic statistical procedure!</td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td>Bellman error: ( Q(s, a) = r + \gamma \max_a Q(s', a) )</td>
<td>This distribution != posterior uncertainty.</td>
<td>The “density model” has nothing to do with the actual task.</td>
<td></td>
</tr>
<tr>
<td>Even “concrete” dropout not necessarily right rate.</td>
<td>Uncertainty in Q ➔ correlated TD loss.</td>
<td>Aleatoric vs Epistemic ... it’s not the right thing for exploration.</td>
<td>With generalization, state “visit count” != uncertainty.</td>
<td></td>
</tr>
</tbody>
</table>
### Estimating uncertainty in deep RL

<table>
<thead>
<tr>
<th><strong>Dropout sampling</strong></th>
<th><strong>Variational inference</strong></th>
<th><strong>Distributional RL</strong></th>
<th><strong>Count-based density</strong></th>
<th><strong>Bootstrap ensemble</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>“Dropout sample = posterior sample” (Gal+Gharamani 2015)</td>
<td>Apply VI to Bellman error as if it was an i.i.d. supervised loss.</td>
<td>Models Q-value as a distribution, rather than point estimate.</td>
<td>Estimate number of “visit counts” to state, add bonus.</td>
<td>Train ensemble on noisy data - classic statistical procedure!</td>
</tr>
<tr>
<td>Dropout rate does not concentrate with the data.</td>
<td>Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$</td>
<td>This distribution ! = posterior uncertainty.</td>
<td>The “density model” has nothing to do with the actual task.</td>
<td>No explicit “prior” mechanism for “intrinsic motivation”</td>
</tr>
<tr>
<td>Even “concrete” dropout not necessarily right rate.</td>
<td>Uncertainty in Q ⇒ correlated TD loss.</td>
<td>Aleatoric vs Epistemic … it’s not the right thing for exploration.</td>
<td>With generalization, state “visit count” ! = uncertainty.</td>
<td></td>
</tr>
</tbody>
</table>

- Dropout sampling
- Variational inference
- Distributional RL
- Count-based density
- Bootstrap ensemble

**“Dropout sample = posterior sample”** (Gal+Gharamani 2015)

- Dropout rate does not concentrate with the data.
- Even “concrete” dropout not necessarily right rate.

- Bellman error: $Q(s, a) = r + \gamma \max_a Q(s', a)$
  - Uncertainty in Q ⇒ correlated TD loss.
  - VI on i.i.d. model does not propagate uncertainty.

- Models Q-value as a distribution, rather than point estimate.
- This distribution ! = posterior uncertainty.

- Estimate number of “visit counts” to state, add bonus.
- The “density model” has nothing to do with the actual task.

- Train ensemble on noisy data - classic statistical procedure!
- No explicit “prior” mechanism for “intrinsic motivation”
Estimating uncertainty in deep RL

**Dropout sampling**

“Dropout sample = posterior sample”  
(Gal+Gharamani 2015)

Dropout rate does not concentrate with the data.

Even “concrete” dropout not necessarily right rate.

**Variational inference**

Apply VI to Bellman error as if it was an i.i.d. supervised loss.

Bellman error: 
\[ Q(s, a) = r + \gamma \max_a Q(s', a) \]

Uncertainty in \( Q \) \( \Rightarrow \) correlated TD loss.

VI on i.i.d. model does not propagate uncertainty.

**Distributional RL**

Models Q-value as a distribution, rather than point estimate.

This distribution \( \neq \) posterior uncertainty.

Aleatoric vs Epistemic  
… it’s not the right thing for exploration.

**Count-based density**

Estimate number of “visit counts” to state, add bonus.

The “density model” has nothing to do with the actual task.

With generalization, state “visit count” \( \neq \) uncertainty.

**Bootstrap ensemble**

Train ensemble on noisy data - classic statistical procedure!

No explicit “prior” mechanism for “intrinsic motivation”

If you’ve never seen a reward, why would the agent explore?
Randomized prior functions
Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.
Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

\[ Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}. \]
Randomized prior functions

- **Key idea**: add a random untrainable “prior” function to each member of the ensemble.

\[
Q_\theta(x) = f_\theta(x) + p(x).
\]

- Visualize effects in 1D regression:
Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

\[ Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + \underbrace{p(x)}_{\text{prior}}. \]

- Visualize effects in 1D regression:
  - Training data \((x,y)\) **black points.**
Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

- Visualize effects in 1D regression:
  - Training data \((x,y)\) black points.
Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

  \[ Q_\theta(x) = \underbrace{f_\theta(x)}_{\text{trainable}} + p(x). \]

- Visualize effects in 1D regression:
  - Training data \((x, y)\) black points.
  - Prior function \(p(x)\) blue line.
Randomized prior functions

• **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

• **Visualize effects in 1D regression:**
  - Training data \((x,y)\) **black points**.
  - Prior function \(p(x)\) **blue line**.
Randomized prior functions

• **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

• **Visualize effects in 1D regression:**
  
  • Training data \((x, y)\) **black points**.
  
  • Prior function \(p(x)\) **blue line**.
  
  • Trainable function \(f(x)\) **dotted line**.
Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

- **Visualize effects in 1D regression:**
  - Training data \((x,y)\) black points.
  - Prior function \(p(x)\) blue line.
  - Trainable function \(f(x)\) dotted line.
Randomized prior functions

• **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

• **Visualize effects in 1D regression:**
  • Training data \((x,y)\) black points.
  • Prior function \(p(x)\) blue line.
  • Trainable function \(f(x)\) dotted line.
  • Output prediction \(Q(x)\) red line.
Randomized prior functions

- **Key idea:** add a random untrainable “prior” function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

- **Visualize effects in 1D regression:**
  - Training data \((x,y)\) **black points**.
  - Prior function \(p(x)\) **blue line**.
  - Trainable function \(f(x)\) **dotted line**.
  - Output prediction \(Q(x)\) **red line**.
Randomized prior functions

- **Key idea:** add a random untrainable "prior" function to each member of the ensemble.

\[ Q_\theta(x) = f_\theta(x) + p(x). \]

- **Visualize effects in 1D regression:**
  - Training data \((x,y)\) **black points**.
  - Prior function \(p(x)\) **blue line**.
  - Trainable function \(f(x)\) **dotted line**.
  - Output prediction \(Q(x)\) **red line**.

**Exact Bayes posterior for linear functions!**
“Deep Sea” Exploration
“Deep Sea” Exploration
“Deep Sea” Exploration

• Stylized “chain” domain testing “deep exploration”: 
"Deep Sea" Exploration

- Stylized "chain" domain testing "deep exploration":
  - State = $N \times N$ grid, observations 1-hot.
“Deep Sea” Exploration

- Stylized “chain” domain testing “deep exploration”:
  - State = $N \times N$ grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
“Deep Sea” Exploration

- **Stylized “chain” domain testing “deep exploration”:**
  - State = \(N \times N\) grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
  - Actions \(\{0,1\}\) map to left/right in each cell.
“Deep Sea” Exploration

- Stylized “chain” domain testing “deep exploration”:  
  - State = N x N grid, observations 1-hot.  
  - Start in top left cell, fall one row each step.  
  - Actions {0,1} map to left/right in each cell.  
  - “Left” has reward = 0, “right” has reward = -0.1/N
“Deep Sea” Exploration

• Stylized “chain” domain testing “deep exploration“:
  - State = N x N grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
  - Actions {0,1} map to left/right in each cell.
  - “left” has reward = 0, “right” has reward = -0.1/N
  - … but if you make it to bottom right you get +1.
“Deep Sea” Exploration

- Stylized “chain” domain testing “deep exploration”:
  - State = N x N grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
  - Actions {0,1} map to left/right in each cell.
  - “left” has reward = 0, “right” has reward = -0.1/N
  - … but if you make it to bottom right you get +1.
“Deep Sea” Exploration

• Stylized “chain” domain testing “deep exploration“:
  - State = N x N grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
  - Actions \{0,1\} map to left/right in each cell.
  - “left” has reward = 0, “right” has reward = -0.1/N
  - … but if you make it to bottom right you get +1.

• Only one policy (out of more than \(2^N\)) has positive return.
“Deep Sea” Exploration

- Stylized “chain” domain testing “deep exploration”:
  - State = N x N grid, observations 1-hot.
  - Start in top left cell, fall one row each step.
  - Actions {0,1} map to left/right in each cell.
  - “left” has reward = 0, “right” has reward = -0.1/N
  - … but if you make it to bottom right you get +1.

- Only one policy (out of more than \(2^N\)) has positive return.

- \(\varepsilon\)-greedy / Boltzmann / policy gradient / are useless.
“Deep Sea” Exploration

- Stylized “chain” domain testing “deep exploration”:  
  - State = N x N grid, observations 1-hot.  
  - Start in top left cell, fall one row each step.  
  - Actions {0,1} map to left/right in each cell.  
  - “left” has reward = 0, “right” has reward = -0.1/N  
  - … but if you make it to bottom right you get +1.

- Only one policy (out of more than 2^N) has positive return.

- ε-greedy / Boltzmann / policy gradient / are useless.

- Algorithms with deep exploration can learn fast!

Visualize BootDQN+prior exploration.
Visualize BootDQN+prior exploration.

- Compare DQN+$\varepsilon$-greedy vs BootDQN+prior.
Visualize BootDQN+prior exploration.

- Compare DQN+ε-greedy vs BootDQN+prior.

- Define ensemble average: \( \frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha) \)
Visualize BootDQN+ prior exploration.

- Compare DQN+$\varepsilon$-greedy vs BootDQN+ prior.

- Define ensemble average:
  \[
  \frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)
  \]

- Heat map shows estimated value of each state.
Visualize BootDQN+prior exploration.

- Compare DQN+$\epsilon$-greedy vs BootDQN+prior.

- Define ensemble average: $\frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)$

- Heat map shows estimated value of each state.
Visualize BootDQN+prior exploration.

- Compare DQN+\(\epsilon\)-greedy vs BootDQN+prior.
- Define ensemble average: \(\frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)\)
- Heat map shows estimated value of each state.
- Red line shows exploration path taken by agent.
Visualize BootDQN+prior exploration.

- Compare DQN+ε-greedy vs BootDQN+prior.
- Define ensemble average: $\frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)$
- Heat map shows estimated value of each state.
- Red line shows exploration path taken by agent.
Visualize BootDQN+prior exploration.

- Compare DQN+\(\varepsilon\)-greedy vs BootDQN+prior.

- Define ensemble average: 
  \[
  \frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha)
  \]

- Heat map shows estimated value of each state.

- Red line shows exploration path taken by agent.

- DQN+\(\varepsilon\)-greedy gets stuck on the left, gives up.
Visualize BootDQN+prior exploration.

• Compare DQN+ε-greedy vs BootDQN+prior.

• Define ensemble average: \( \frac{1}{K} \sum_{k=1}^{K} \max_{\alpha} Q_k(s, \alpha) \)

• Heat map shows estimated value of each state.

• Red line shows exploration path taken by agent.

• DQN+ε-greedy gets stuck on the left, gives up.

• BootDQN+prior hopes something is out there, keeps exploring potentially-rewarding states… learns fast!
Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer
Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post
bit.ly/rpf_nips
Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post
bit.ly/rpf_nips

Montezuma’s Revenge!
Come visit our poster!

Ian Osband, John Aslanides, Albin Cassirer

Blog post: bit.ly/rpf_nips

Montezuma’s Revenge!

Demo code: bit.ly/rpf_nips