Search on the Replay Buffer
Bridging Planning & RL

Ben Eysenbach\textsuperscript{12}, Ruslan Salakhutdinov\textsuperscript{1}, Sergey Levine\textsuperscript{23}

\textsuperscript{1}CMU, \textsuperscript{2}Google Brain, \textsuperscript{3}UC Berkeley
Why don't we use RL to manufacture airplanes?
Why don't we use RL to manufacture airplanes?

- Stakes are too high to learn by trial and error.
- Instead, airplanes come together through a planning process.
- Many components might be solved with RL (e.g., painting, riveting)
- How can we meld RL with planning?
Agent wants to navigate from start to goal, but will only succeed if it starts close to the goal (green region).
Search on the Replay Buffer

Consider states that we've seen before (i.e., drawn from our replay buffer).
Search on the Replay Buffer

While we still cannot reach the goal state, we can reach some of the intermediate states.
Search on the Replay Buffer

**Key Idea**: Build a graph on previously seen states, use shortest path algorithm to find subgoals, use policy to reach each subgoal.
Search on the Replay Buffer

Problems to solve:
● How to learn the policy?
● How to learn the distances?
Goal-Conditioned RL [Kaelbling 93, Schaul 15, Pong 18, Andrychowicz 17]

Define:

\[ r(s, a, s_g) \triangleq -1 \]
\[ \gamma = 1 \]
\[ \text{done}(s_t, a_t, s_g) \triangleq \delta(s_t = s_g) \]

Q values correspond to negative shortest-path distance:

\[ V(s, s_g) = -d_{sp}(s, s_g) \]
Goal-Conditioned RL [Kaelbling 93, Schaul 15, Pong 18, Andrychowicz 17]

How do you find the policy? ✔
How do you find the distances? ✔
Constructing a Graph

**Nodes:** Observations from our replay buffer.

**Edges:** All pairs of nodes

\[ \mathcal{E} = \mathcal{B} \times \mathcal{B} = \{ e_{s_1 \rightarrow s_2} \mid s_1, s_2 \in \mathcal{B} \} \]

**Weights:**

\[ \mathcal{W}(e_{s_1 \rightarrow s_2}) = \begin{cases} d_\pi(s_1, s_2) & \text{if } d_\pi(s_1, s_2) < \text{MAXDIST} \\ \infty & \text{otherwise} \end{cases} \]
SoRB: a policy that performs search internally.

```plaintext
function SEARCHPOLICY(s, s_g, B, V, \pi)
    s_{w_1}, \ldots \leftarrow \text{SHORTESTPATH}(s, s_g, B, V)
    d_{s \rightarrow w_1} \leftarrow -V(s, s_{w_1})
    d_{s \rightarrow g} \leftarrow -V(s, s_g)
    \text{if } d_{s \rightarrow w_1} < d_{s \rightarrow g} \text{ or } d_{s \rightarrow g} > \text{MAXDIST}
        a \leftarrow \pi(a, | s, s_{w_1})
    \text{else}
        a \leftarrow \pi(a, | s, s_g)

return a
```
Goal-conditioned RL produces an agent that is adept at reaching some goals.
Search on the Replay Buffer

Search on the Replay Buffer is a simple trick for improving that policy, *without retraining*. 

Goal-Conditioned RL → SoRB
Two Problems with Distance Estimates

Avoiding Wormholes  Calibration
Problem 1: Avoiding Wormholes
Solution to Wormholes: Ensembles

\[ Q_\theta(s, a, s_g) \]
Solution to Wormholes: Ensembles

\[ d(s, s_g) = \max (-V_{\theta_1}(s, s_g), -V_{\theta_2}(s, s_g), \ldots) \]

Plan using largest predicted distance (pessimistic). We tried to use weight sharing, but predictions were too correlated.
Problem 2: Uncalibrated Distances

- Correct Q-value for an unreachable goal is $\infty$.
- Causes Q-learning to diverge.
Solution to Calibration: Distributional RL

Distributional RL is simple when using sparse rewards and no discounting.

Experiments
2D Navigation
Planning in Image-Space

Environment from [Shah 18]
Planning in Image-Space

- SoRB = Search on the Replay Buffer (our method)
- SPTM = Semi-Parametric Topological Memory [Savinov 18]
- C51 = Distributional RL [Bellemare 18]
- VIN = Value Iteration Networks [Tamar 16]
- HER = Hindsight Experience Replay [Andrychowicz 17]
- Random = Random agent
Ensembles are Important
Distributional RL is Important

![Graph showing the success rate over distance for RGB and Depth data with different policies. The graph includes lines for SoRB, SoRB with distributional RL, and random.]
Just Learn Distances with Supervised Learning?

**SPTM [Savinov 18]:** Collect data from random policy, learn distances via supervised learning

**Problem:** Learns distances w.r.t. random policy, but used to predict performance of non-random policy.
Does SoRB Generalize?

Train on many houses

Test on new houses
Does SoRB Generalize?

![Graphs showing success rates over distance for RGB and Depth data. The graphs compare SoRB, RL, and random methods.](image)
Relation to Prior Work

SoRB as a Planning Algorithm [Choset 05, LaValle 06, Kavraki 96, Lau 05]

- Combining RL + Planning [Chiang 19, Faust 18, Savinov 18]
- Representations for Planning [Florensa 19, Savinov 18, Wu 18, Lenz 15, Watter 15]
- Differentiable Planners [Amos 18, Lee 18, Srinivas 18, Tamar 16]
- Hierarchical RL [Bacon 17, Frans 17, Kaelbling 93, Kulkarni 16, Nachum 18, Parr 98, Precup 2000, Sutton 99, Vezhnevets 17, Drummond 02, Fox 17, Simsek 05]
- Model-Based RL [Agrawal 17, Chua 18, Finn 17, Kurutach 18, Nagabandi 18, Oh 15, Sutton 90]
Relation to Prior Work

SoRB as a Planning Algorithm [Cho10, LaVa06, Ka96, Lau05]

- Combining RL + Planning [Chia19, Faus18, Savi18, Wu18, Len15, Watt15]
- Representations for Planning [Flore19, Savi18, Wu18, Len15, Watt15]
- Differentiable Planners [Amos18, Lee18, Sriniv18, Tamar16]
- Hierarchical RL [Baco17, Fran17, Keeb93, Kulk16, Nach18, Par98, Precu00, Sutton99, Vezhe17, Drum02, Fox17, Simse05]
- Model-Based RL [Agraw17, Chua18, Finn17, Kuruta18, Nagab18, Oh15, Sutton90]

<table>
<thead>
<tr>
<th>Model</th>
<th>Real States</th>
<th>Multi-Step</th>
<th>Prediction Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>State-Space</td>
<td>✓</td>
<td>✓</td>
<td>1000s+</td>
</tr>
<tr>
<td>Latent-Space</td>
<td>✗</td>
<td>✓</td>
<td>10s</td>
</tr>
<tr>
<td>Inverse</td>
<td>✓</td>
<td>✗</td>
<td>10s</td>
</tr>
<tr>
<td>SoRB</td>
<td>✓</td>
<td>✓</td>
<td>1</td>
</tr>
</tbody>
</table>
Takeaways & Open Problems

Takeaways

● Planning is useful for many real-world problems.
● Goal-Conditioned RL works well locally.
● Graph search is a tool to boost performance of goal-conditioned agent.
● Distributional RL + Ensembles provide robust distance estimates.

Open Problems

● How to incorporate planning into policy search?
● SoRB as an inference procedure?
● Better generalization?
● What states to use for planning?