Relay Policy Learning: Solving Long-Horizon Tasks via Imitation and Reinforcement Learning

Presenter: Harshit Sikchi

November 4, 2021
Motivation

- Solving long horizon problems require extended exploration and current methods fail to achieve this usually failing to discover the goal/or encountering high variance policy updates.

- Imagine planning a path from Torchy Taco’s to GDC. Would you rather plan it in terms of your muscle torques/footsteps or in terms on landmarks you will encounter along the way?

Planning a path from point A to B
Motivation

- This intuition forms the basis of Hierarchical Reinforcement Learning (HRL).
- Your muscle control is your low-level policy and the landmarks are subgoals that are given by your high-level policy.
- HRL allows for better multitask generalization as we can reuse learned skills!
Motivation

“Can we use meaningful but unstructured human demonstrations to learn hierarchical policies that can be finetuned with Reinforcement Learning”
Problem Setting

Goal-Conditioned Reinforcement Learning

Goal-Conditioned Policy: \( \pi(a|s, s_g) \)  
GC-reward function \( r(s,a,s_g) \)

GC-RL objective  
\[ \mathbb{E}_{s \sim s_g} \left[ \mathbb{E}_\pi \left[ \sum \gamma^t r_t(s_t,a_t,s_g) \right] \right] \]

Goal-Conditioned Imitation Learning

Given a dataset of demonstrations trajectories in \( D \) reaching goals \( s_g^i, s_g^k, \ldots \)

GC-IL objective: Learn a policy \( \pi(a|s, s_g) \) able to achieve different goals imitating \( D \).
Limitations of Prior Work

- A number of previous HRL approaches[1,2,3] which learn both high-level and low-level policy struggle with extended exploration and optimization difficulties!

  🤔 Can we solve this issue using human demonstrations?

- A number of previous HIL approaches[4,5,6] learn segmentation/primitives from demonstration data, but these methods are not amenable to further fine-tuning?

  😞 We might need fine-tuning with RL for complex long-horizon problems where imitation alone suffers from compounding errors.
Method – Relay Policy Learning (RPL)

- A policy architecture for RPL

Low-level policy takes that subgoal and output low level actions to act in the environment.

Low-level policy runs for fixed timesteps – in this paper 30.
Stage 1: Relay Imitation Learning

- Learn from meaningful but unstructured human-demonstrations
Stage 1: Relay Imitation Learning

- Learning the **low-level** policy

Consider a trajectory from the dataset:

\[ s_1, a_1, s_2, a_2, s_3, a_3, s_4, a_4, s_5, a_5, s_6, a_6, \ldots s_T, a_T \]

Generated labels for this window:

- \[ s_1, a_1, s_2 \]
- \[ s_1, a_1, s_3 \]
- \[ s_1, a_1, s_4 \]
- \[ s_1, a_1, s_5 \]

---

**Algorithm 2** Relay data relabeling for RIL low level

- **Require:** Demonstrations \( D = \{ \tau_0, \tau_1, \ldots \tau_N \} \)

- 1: for \( n = 1 \ldots N \) do
- 2: for \( t = 1 \ldots t_n \) do
- 3: for \( w = 1 \ldots W_l \) do
- 4: Add \( (s^n_t, a^n_t, s^n_{t+w}) \) to \( D_l \)
- 5: end for
- 6: end for
- 7: end for
Stage 1: Relay Imitation Learning

- Learning the **high-level** policy

Consider a trajectory from the dataset:

\[ s_1, a_1, s_2, a_2, s_3, a_3, s_4, a_4, s_5, a_5, s_6, a_6, \ldots, s_T, a_T \]

Generated labels for this window:

\[ s_1, s_6, s_2 \]
\[ s_1, s_6, s_3 \]
\[ s_1, s_6, s_4 \]
\[ s_1, s_6, s_5 \]
\[ s_1, s_6, s_6 \]
\[ s_1, s_6, s_7 \]

---

**Algorithm 3** Relay data relabeling for RIL high level

**Require:** Demonstrations \( D = \{ \tau_0, \tau_1, \ldots, \tau_N \} \)

1: for \( n = 1 \ldots N \) do
2: \hspace{1em} for \( t = 1 \ldots t_n \) do
3: \hspace{2em} for \( w = 1 \ldots W_h \) do
4: \hspace{3em} Add \( (s_t^n, s_t^{n+\min(w,W_i)}, s_t^{n+w}) \) to \( D_h \)
5: \hspace{2em} end for
6: \hspace{1em} end for
7: end for
Stage 1: Relay Imitation Learning

- Train the policies $\pi^h_\theta, \pi^l_\theta$ by maximizing likelihood (behavior cloning)

$$\max_{\phi, \theta} \mathbb{E}_{(s,a,s'_g) \sim D_l}[\log \pi_{\phi}(a|s,s'_g)] + \mathbb{E}_{(s,s'_l,s'_h) \sim D_h}[\log \pi_{\theta}(s'_l|s,s'_h)].$$

- RIL improves upon naïve imitation learning by:

1. Generating more data by relabelling.
2. Improves generalization by training on a variety of goals.
Stage 2: Relay Reinforcement Learning

- Finetune the learned hierarchical GC policy by Reinforcement Learning.

- Method: Decoupled Optimization
  - Fix low level policy, and train high level policy.
  - Fix high level policy, and train low level policy.
Stage 2: Relay Reinforcement Learning

- Finetune the learned hierarchical GC policy by Reinforcement Learning.

Low level policy update:

\[
\nabla_\phi J_l = \mathbb{E}[\nabla_\phi \log \pi_{\phi}^l(a|s, s_g^l) \sum r_l(s_t, a_t, s_g^l)] + \lambda_l \mathbb{E}_{(s,a,s,g) \sim D_l} [\nabla_\phi \log \pi_{\phi}^l(a|s, s_g^l)]
\]

High level policy update:

\[
\nabla_\theta J_h = \mathbb{E}[\nabla_\theta \log \pi_{\theta}^h(s_g^l|s, s_g^h) \sum r_h(s_t, s_g^l, s_g^h)] + \lambda_h \mathbb{E}_{(s,s_g,s_g^h) \sim D_h} [\nabla_\theta \log \pi_{\theta}^h(s_g^l|s, s_g^h)].
\]
Stage 1: Relay Policy Learning
Variants of RPL

- Finetune the learned hierarchical GC policy by Reinforcement Learning.

**IRIL-RPL**: At each iteration of RL, relabel the collected trajectories with the states reached along the trajectory as goals and add to dataset ($D_t$ and $D_{ht}$) for behavior cloning.

  Assumes states are reached optimally within the trajectory of intermediate RL policies. Too strong assumption?!

**DAPG-RPL**: Fine tune the policy without the off-policy addition as in IRIL.

**NPG-RPL**: Fine-tune policy without off-policy dataset or the behavior cloning term.
Experimental Setup

- **Tasks:**
  1. Open microwave
  2. Four turnable over burners
  3. Move kettle
  4. Open hinged cabinet
  5. Open sliding door

- **400 Expert demonstrations are collected by VR.**

- **Each experiment consists of 4 of the tasks above.**
Baselines

- Behavior cloning (BC) [no hierarchy]
- Goal conditioned behavior cloning (GCBC) [no hierarchy]
- Behavior cloning + Finetuning (DAPG-BC) [no hierarchy]
- Goal conditioned behavior cloning + Finetuning (DAPG-GCBC) [no hierarchy]
- Oracle split: Low level policies are trained to imitate oracle segmented demonstrations. [hierarchy]
- HIRO: HRL method that learns both low level and high level policy from scratch. [hierarchy]
- PreTrain low level: Learn low level policy from demonstrations and high level from scratch. [hierarchy]
- Nearest neighbour: Executes the trajectory open loop which is nearest to commanded goal in demonstrations [no-hierarchy]
Results – Only Imitation

- RIL does not learn to solve the tasks but does better than non-hierarchical imitation learning.

<table>
<thead>
<tr>
<th></th>
<th>RIL (ours)</th>
<th>GCBC relabeling</th>
<th>GCBC no relabeling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success Rate (%)</td>
<td>21.7</td>
<td>8.8</td>
<td>7.6</td>
</tr>
<tr>
<td>Average Step Completion (of 4)</td>
<td>2.4 ± 1.13</td>
<td>2.2 ± 0.95</td>
<td>1.78 ± 1.0</td>
</tr>
</tbody>
</table>

- RIL in action:
Results

- RPL succeeds at learning to solve ~3.5/4 tasks outperforming baselines.
Results

- RPL succeeds at learning to solve ~3.5/4 tasks outperforming baselines.
Results

- Effect of window size and reward function used for finetuning.
  - Higher window size is detrimental since the low level policy takes same actions for more number of different goals.
  - Sparse reward is more successful as the exploration of RPL is sufficient and Sparse rewards prevents local optima.
Results – Failure cases

- Agent sometimes gets stuck after 1 or 2 tasks!
Discussion of Results

- Hierarchical Imitation Learning (HIL) improves upon Flat Imitation Learning
  - HIL demonstrates better multitask generalization as a result of the added structure!

- RPL presents a hierarchical policy architecture that enables easy optimization and is easy to fine-tune further with RL.

- RPL is better at learning long-horizon behavior with high success rate compared to baselines.
Critique

- The simplified policy architecture uses a fixed horizon for low level skills – Does not take into account some skills are extended and some are short.

- Requires meaningful demonstrations from Humans.

- Uses a strong assumption of optimality in IRIL-RPL which is not clarified to be correct theoretically and needs more discussion.

- Experiments rely on a fixed horizon of 4 tasks. Paper does not discuss how the method scales with task-horizon since the main claim is RPL solves long—horizon tasks.
- Experiment clarifications are lacking in appendix.
Future Work/Open Questions

1. How to learn from unstructured demonstrations rather than assuming meaningful demonstrations?

2. Learning options/skills vs Learning fixed-horizon low-level policies?

3. Efficient architectures for HRL:
   1. Planning for high-level policy and Learning for low level policy:
      a. Search on the replay buffer: Eysenbach et al
      b. Planning with goal conditioned policies: Nasiriany et al
Extended Readings

- Meta Learning shared Hierarchies – Frans et al 17
- Data efficient Hierarchical Reinforcement Learning – Nachum et al. 18
- Accelerating Reinforcement Learning with Learned Skill Priors – Pertsch et al 20
- Parrot: Data driven behavioral priors for Reinforcement Learning – Singh et al 20
Summary

- How can we learn long horizon tasks given meaningful human demonstrations?

- Long horizon tasks are hard for RL agents due to extended exploration and variance of Policy Gradient.

- Previous work either fail to incorporate demonstrations or are not amenable to finetuning with RL.

- Key insights: 1) A simple bi-level hierarchical architecture allows for improved imitation learning leveraging multi-task generalization 2) This policy is amenable to fine-tuning with RL.

- RPL achieves better performance on long-horizon kitchen manipulation tasks than baselines.