Accelerating Reinforcement Learning with Learned Skill Priors

Karl Pertsch, Youngwoon Lee, Joseph J. Lim

Yibin Xiong

November 2021

Motivation

- Transferring prior knowledge on other tasks accelerates RL training.
- There can be too many prior skills to transfer, making it slow to explore all of them.
- We want to identify the skills that are "useful" to our downstream task.
- Once we know which skills are "useful", which is represented by a prior probability distribution over skills, then we can use it to guide downstream RL training by incorporating a regularization towards this skill prior.

Motivation

- Transferring prior knowledge on other tasks accelerates RL training.
- There can be too many prior skills to transfer, making it slow to explore all of them.
- We want to identify the skills that are "useful" to our downstream task.
- Once we know which skills are "useful", which is represented by a prior probability distribution over skills, then we can use it to guide downstream RL training by incorporating a regularization towards this skill prior.

  Pre-training: *jointly* learn a embedding space of skills and a skill prior for the downstream task

  Downstream: SAC regularized by the skill prior

Skills

- In an unsupervised setting, we leverage plenty of unannotated data. For RL, this could be trajectories without rewards or even exploration trajectories without clear task definition.
- Extract *skills* [1] from these trajectories.
  A skill is a sequence of actions the represents useful behaviors.

$$\mathbf{a} := \{a_1, \ldots, a_H\}, \text{ with a fixed horizon H}$$

- Skill *itself* as a prior knowledge:
  ◦ With proper H chosen, we impose the knowledge that some consecutive actions are meaningful (or say dependent).
  ◦ The agent does not need to figure out that such sequence of actions are meaningful through exploration and reward signal, thus accelerating RL training.
  ◦ H is too large ⇒ Takes some exploration to "counteract" the undesired behavior(s) in a skill, slow convergence

[1] easy to extract from offline data, no need for reward annotations

Embedding Space

Note that we do not have a label for what a *skill* is doing.
How should we know which are useful?

– Construct an embedding space that produces good representation (e.g. fulfilling assumptions like "disentanglement", "semi-supervised learning", etc.)

- Use VAE to build an encoder $q(z|\mathbf{a}_i)$ and a decoder $p(\mathbf{a}_i|z)$
- Assume that $p(z) \sim \mathcal{N}(0, I)$
- Maximize regularized ELBO (β-VAE, promotes disentanglement)

$$\log p(\mathbf{a}_i) \geq \mathbb{E}_q\big[\log p(\mathbf{a}_i|z)\big] - \beta\, D_{KL}\big(q(z|\mathbf{a}_i)\|p(z)\big)$$

Skill Prior

– Specify which skills are useful given the current situation

◦ Learn the skill prior $p_{\mathbf{a}}(z|\cdot)$.

- Here we condition on state s_t.
- We can also use z_{t-1}, the previous skill executed, but this imposes a weaker prior knowledge.

◦ Minimize $\mathbb{E}_{(s,\mathbf{a}_i)\sim\mathcal{D}}\left[D_{KL}\left(q(z|\mathbf{a}_i) \parallel p_{\mathbf{a}}(z|s_t)\right)\right]$

- $\mathbf{a}_i$ is the ground-truth, while s_t is a highly related feature.
- We obtain a fairly accurate $q(z|\mathbf{a}_i)$ in VAE training and make $p_{\mathbf{a}}(z|s_t)$ close to $q(z|\mathbf{a}_i)$.
- A property of KL divergence: Suppose we use $D_{KL}\left(p_{\mathbf{a}}(z|s_t)\| q(z|\mathbf{a}_i)\right)$ At places where $p_a(z|s_t)$ is close to 0 and $q(z|a_i)$ is significantly non-zero, $q(z|a_i)$'s effect is discarded.[2] This means we ignore some z and thus some skills.

[2] https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html

Training

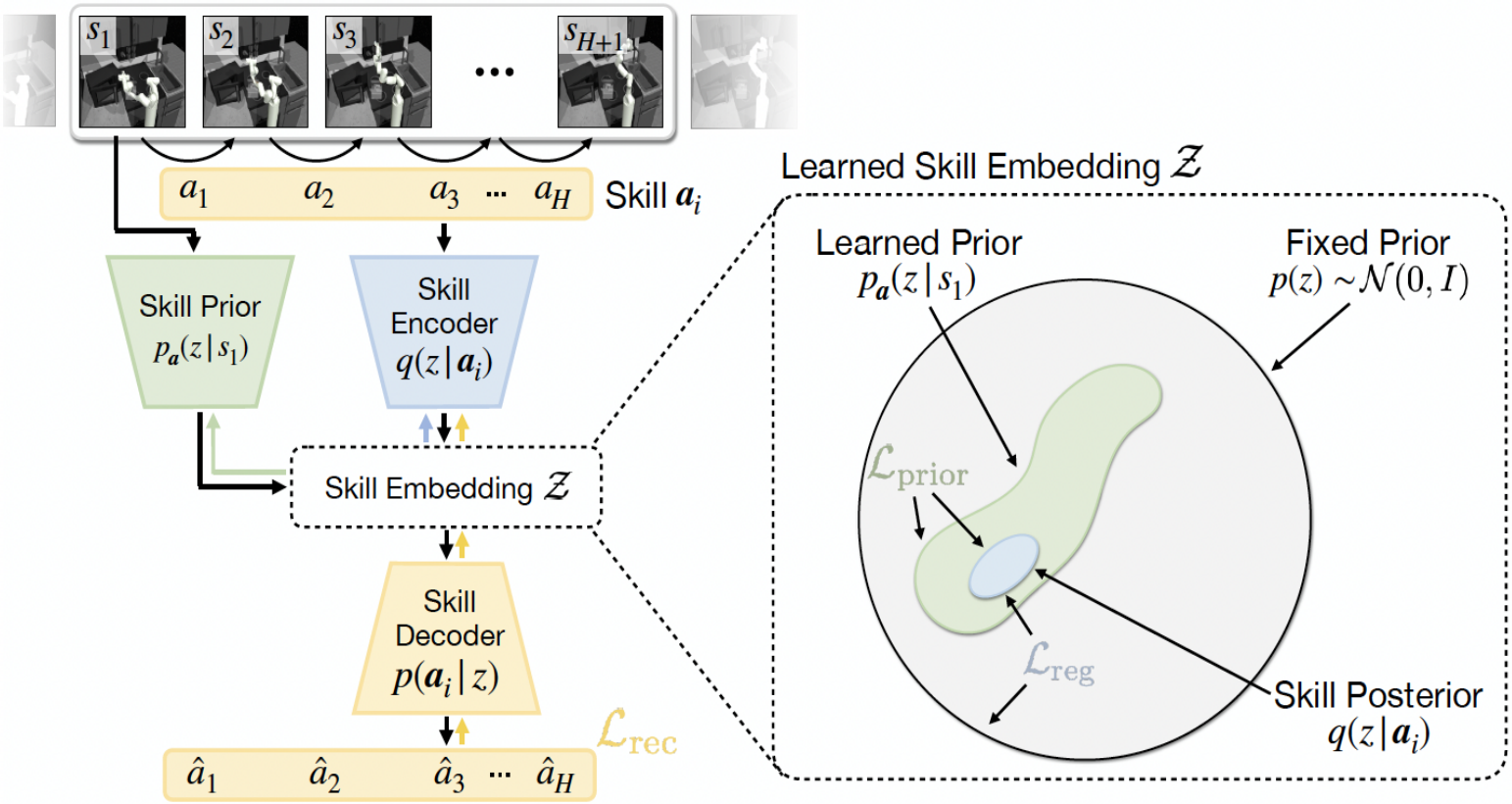

Figure 2: Deep latent variable model for joint learning of skill embedding and skill prior. Given a state-action trajectory from the dataset, the skill encoder maps the action sequence to a posterior distribution $q(z|\boldsymbol{a}_i)$ over latent skill embeddings. The action trajectory gets reconstructed by passing a sample from the posterior through the skill decoder. The skill prior maps the current environment state to a prior distribution $p_{\boldsymbol{a}}(z|s_1)$ over skill embeddings. Colorful arrows indicate the propagation of gradients from reconstruction, regularization and prior training objectives.

Regularization in Downstream RL

Idea: Use the skill prior $p_{\mathbf{a}}(z|s_t)$ to regularize RL training.

▷ Typical regularization method: Maximum Entropy

$$\max \ J(\theta) = \mathbb{E}_{\pi}\Big[\sum_{t=1}^{T} \gamma^t r(s_t, a_t) + \alpha \mathcal{H}\big(\pi(a_t|s_t)\big)\Big]$$

Note that $\mathcal{H}\big(\pi(a_t|s_t)\big) = -\mathbb{E}_{\pi}\big[\log \pi(a_t|s_t)\big] \propto -D_{KL}\big(\pi(a_t|s_t) \parallel U(a_t)\big)$, so maximum entropy is equivalent to regularizing $\pi(a_t|s_t)$ using uniform prior distribution.

▷ In our case, we want to make $\pi(a_t|s_t)$ not deviate much from the skill prior $p_{\mathbf{a}}(z|s_t)$, so we replace $U(a_t)$ by $p_{\mathbf{a}}(z|s_t)$ in the regularization term.

$$\max \ J(\theta) = \mathbb{E}_{\pi}\Big[\sum_{t=1}^{T} \gamma^t r(s_t, a_t) - \alpha D_{KL}\big(\pi(a_t|s_t) \parallel p_{\mathbf{a}}(z|s_t)\big)\Big]$$

Experiments

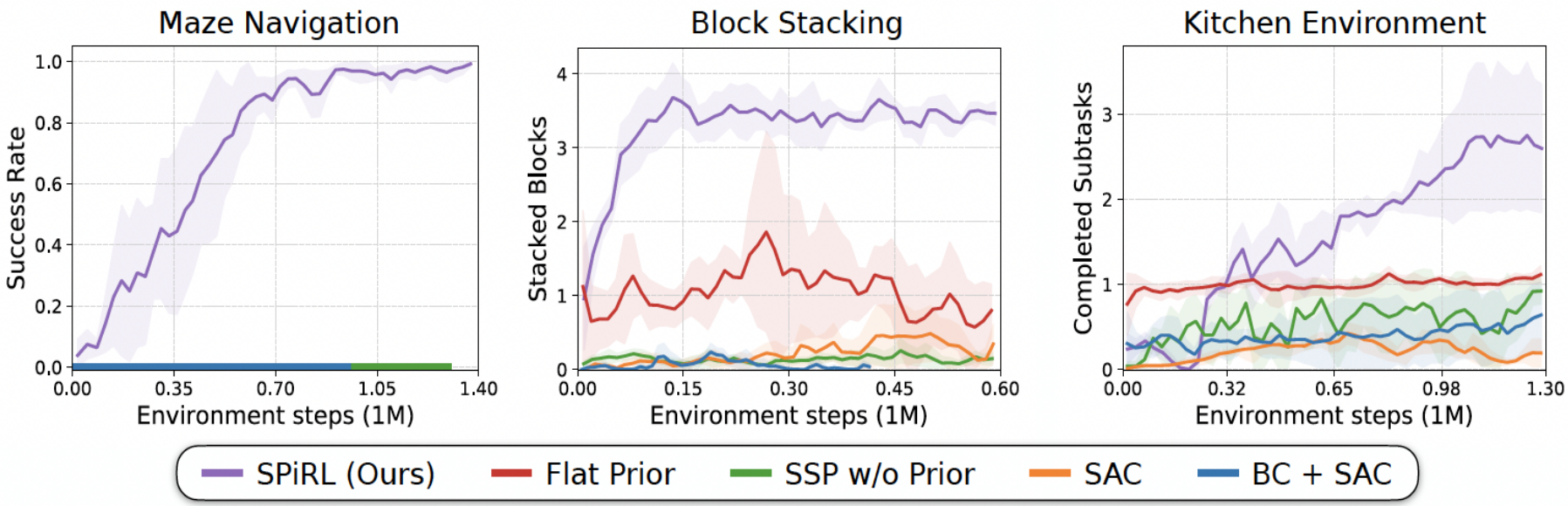

Figure 4: Downstream task learning curves for our method and all comparisons. Both, learned skill embedding and skill prior are essential for downstream task performance: single-action priors without temporal abstraction (**Flat Prior**) and learned skills without skill prior (**SSP w/o Prior**) fail to converge to good performance. Shaded areas represent standard deviation across three seeds.

Both skills (prior about temporal abstraction) and the skill prior are important. It seems that the skill prior is more beneficial.

Reference

Karl Pertsch, Youngwoon Lee, and Joseph J Lim. "Accelerating reinforcement learning with learned skill priors". In: *arXiv preprint arXiv:2010.11944* (2020)