

LEARNING AN EMBEDDING SPACE **FOR TRANSFERABLE ROBOT SKILLS** - (ICLR 2018)

Karol Hausman (Department of Computer Science, University of Southern California),

Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, Martin Riedmiller (DeepMind).

RATIONALE AND KEY IDEAS

The authors presented a reinforcement learning method suitable for *closely related skill* (think of them as robot movement capabilities) parameterized via a skill embedding space.

Latent variables plays an important role, as far as variational calculus, used to approximate intractable probabilities. Learning happens through entropy regularized policy gradients.

REQUIRED PROPERTIES OF THE LEARNED SKILLS

- i **Generality:** in the skill embedding space, solutions to different, potentially orthogonal, tasks can be represented; i.e. tasks such as *lifting a block* or *pushing it through an obstacle course* should both be jointly learnable.
- ii **Versatility:** different embedding vectors that are “close” to each other in the embedding space correspond to distinct solutions to the same task.
- iii **Identifiability:** Given the state and action trace (trajectory) of an executed skill, it should be possible to *identify the embedding vector that gave rise to the solution*. Hence new tasks would be solvable by picking a new sequence of embedding vectors.



Bayesian “Influenced” RL

Latent variable and variational inference are brilliantly connected to the policy learning scenario.



Skills as embedding

Continuously parametrized internal representation to help solving the control problem, also reusing what already learnt.



Entropy regularization

Latent variable to represent several alternative task-solutions, entropy based regularization term allows exploration of such alternatives, more than the reward objective would do..



Off-policy learning

Q learning is exploited to make the approach data efficient and feasible for real world robotic systems.

PROBLEM FORMALIZATION

The context is reinforcement learning in Markov decision processes (MDP).

$s \in \mathbb{R}^S$	←	Continuous state of the agent.
$a \in \mathbb{R}^A$	←	Continuous action vector.
$p(s_{t+1} s_t, a_t)$	←	Probability of transition to state s_{t+1} .
$\pi_\theta(a s)$	←	Agent policy: action distribution.

$\mathcal{T} = [1, \dots, T]$	←	Set of initial tasks.
$t \in \mathcal{T}$	←	Task id (awareness during training).
$r(s_t, a_t)$	←	Scalar reward at a certain step.
$\mathbb{E}_{\tau_\pi} [\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$	←	Discounted reward (to maximize).



Overcoming the end-to-end training

At each training episode, pick random task id t and execute the agent policy $\pi(a|s, t)$.

How can the model actually learn a “cluster of solutions” and not just T separate solution (one per tasks) ? → **Numerically represented skills !**

Use a task conditioned latent variable z (skill embedding), so that the policy is able to represent a distribution over skills for each task and share these across tasks.

The state-task policy can be re-defined as: $\pi(a | s, t) = \int \pi(a | z, s, t) p(z | t) dz$ (assuming z to be resampled at each timestamp).

ENTROPY REGULARIZED POLICY - 1

The usage of latent variable facilitates the representation of several alternative solutions, but usage of the “only” discounted reward function does not guarantee that such alternatives will be learned.

$$R(a, s, t) = \mathbb{E}_{\pi} \left[\sum_{i=0}^{\infty} \gamma^i r_t(s_i, a_i) \mid s_0 = s, a_i \sim \pi(\cdot | s, t) \right]$$

To encourage this desired behaviour, the objective is formulated as an entropy regularized RL problem, similarly to many “classical” policy gradient schemas [1, 2], with the difference of taking into account also the future actions.

Anyway, presence of latent variables z may cause intractability issues, requiring the construction of a lower bound as a solution.

$$\begin{aligned} \mathcal{H}[\pi(a \mid s, t)] &= \mathbb{E}_{\pi}[-\log \pi(a \mid s, t)] \\ &\geq \mathbb{E}_{\pi(a, z | s, t)} \left[\log \left(\frac{q(z | a, s, t)}{\pi(a, z | s, t)} \right) \right] \end{aligned}$$

$$\max_{\pi} \mathbb{E}_{\pi, p_0, t \in \mathcal{T}} \left[\sum_{i=0}^{\infty} \gamma^i (r_t(s_i, a_i) + \underbrace{\alpha \mathcal{H}[\pi(a_i | s_i, t)]}_{\text{weighting term}}) \mid a_i \sim \pi(\cdot | s, t), s_{i+1} \sim p(s_{i+1} | a_i, s_i) \right]$$

Diagram illustrating the components of the objective function:

- $p_0(s_0)$: Initial state distribution
- α : weighting term
- $\mathcal{H}[\pi(a_i | s_i, t)]$: Intractable due to presence of latent variable (indicated by a warning icon)
- Lower bound exists!



Variational Inference solution

According to Barber and Agakov we can freely choose a $q(z | a, s, t)$ distribution to construct a lower bound.

ENTROPY REGULARIZED POLICY - 2

Lower bound is further reformulated in a three terms sum, so that it would meet the desired properties and behaviours:

- In the first term, variational bound q is chosen complying **identifiability** (avoiding conditioning on the task id t , to ensure that a given trajectory alone would allow to identify its embedding).
 - Intractability of $p(z|a, s, t)$ is solved through a sampling based evaluation of the *CE (cross entropy)* term.
- Third and second terms encourages exploration of the embedding space ("*skill clusters*") ensuring **generality** and **versatility**.

Three neural networks are used to set respectively: the variational distribution $q_\psi(z|a, s)$ the embedding distribution $p_\phi(z|t)$ and the policy distribution $\pi_\theta(a|s, z)$. For trajectories (states segments) the notation of the first NN is updated into $q_\psi(z | a, s_i^H)$ with $s_i^H = [s_{i-H}, \dots, s_i]$.

Using the *entropy summation* below as a regularization term in our the original reward function, we would obtain the objective: $L(\theta, \phi, \psi)$.

$$\underbrace{\mathbb{E}_{\pi(a, z|s, t)} \left[\log \left(\frac{q(z|a, s, t)}{\pi(a, z|s, t)} \right) \right]}_{\text{Initially defined lower bound}} = \underbrace{-\mathbb{E}_{\pi_\theta(a|s, t)} [\mathcal{CE} [p(z | a, s, t) || q_\psi(z | a, s)]]}_{\text{"Preserve" identifiability}} + \underbrace{\mathcal{H} [p_\phi(z | t)]}_{\text{Entropy of embedding given the task t}} + \underbrace{\mathbb{E}_{p_\phi(z|t)} [\mathcal{H} [\pi_\theta(a | s, z)]]}_{\text{Entropy of policy conditioned on the embedding}}$$

LEARNING (ON POLICY AND OFF POLICY)

The new entropy regularized objective now looks like below... Note that the original \mathcal{Q} has been splitted into three weight terms: one for each entropy addendum, emphasizing *different aspects and desired behaviours*.

$$L(\theta, \phi, \psi) = \mathbb{E}_{\pi_{\theta}(a, z | s, t)} \left[\sum_{i=0}^{\infty} \gamma^i \hat{r}(s_i, a_i, z, t) \mid s_{i+1} \sim p(s_{i+1} \mid a_i, s_i) \right] + \alpha_1 \mathbb{E}_{t \in \mathcal{T}} [\mathcal{H}[p_{\phi}(z \mid t)]] ,$$

where $\hat{r}(s_i, a_i, z, t) = [r_t(s_i, a_i) + \alpha_2 \log q_{\psi}(z \mid a_i, s_i^H) + \alpha_3 \mathcal{H}[\pi_{\theta}(a \mid s_i, z)]]$

At this point, a direct optimization, in an *on-policy* setting is possible, authors will try to make set up *off-policy* learning procedures, in order to obtain a data efficient algorithm, effectively feasible in a real robotic system.



Q-Learning is the way

Instead of effectively estimate the terms of the discounted reward through environment interaction, using previously gathered data, this values can be efficiently estimated with a Q-value function yielding an off-policy algorithm. From a variational inference perspective, this can be viewed as a "amortized inference network estimating a log likelihood term".

$$Q^{\pi}(s_i, a_i; z, t) = \hat{r}(s_i, a_i, z, t) + \gamma \mathbb{E}_{p(s_{i+1} | a_i, s_i)} [Q^{\pi}(s_{i+1}, a_{i+1}; z, t)]$$

Authors combine two recent research results to learn a parametric representation of Q_{φ}^T : the Retrace algorithm from Munos et al. [3], and a target Q-network with parameters φ' from Mnih et al. [4].

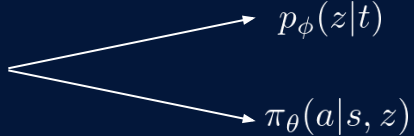
$$\min_{\varphi} \mathbb{E}_{\mathcal{B}} \left[(Q_{\varphi}^{\pi}(s_i, a_i; z, t) - Q^{\text{ret}})^2 \right] \quad \vdots \quad c_k = \min \left(1, \frac{\pi(a_k | z, s_k, t) p(z | t)}{b(a_k | z, s_k, t) b(z | t)} \right)$$

$$Q^{\text{ret}} = \sum_{j=i}^{\infty} \left(\gamma^{j-i} \prod_{k=i}^j c_k \right) [\hat{r}(s_j, a_j, z, t) + \mathbb{E}_{\pi(a | z, s, t)} [Q_{\varphi'}^{\pi}(s_i, \cdot; z, t)] - Q_{\varphi'}^{\pi}(s_j, a_j; z, t)]$$

MULTI TASK SETUP

Agent was assumed to be provided with a reply buffer \mathcal{B} containing full trajectory execution traces (including states, actions, task ids and rewards) *incrementally filled during training*. In conjunction with these traces, also the probabilities of selected states and actions were stored, denoting with $b(a|z, s, t)$ what concerns the behaviour policy probabilities, and with $b(z|t)$ the embeddings (both used in the previous slide to compute c_k).

Once the Q_φ^T computation has been all setted up, to apply gradient descent, its derivative is handled using reparametrization trick proposed in [5,6] and already applied for off-policy RL [7]. Auxiliary networks objectives can now be redefined and optimized without additional *external interactions*.

$$\hat{L}(\theta, \phi) = \mathbb{E}_{\substack{\pi_\theta(a|z,s) \\ p_\phi(z|t) \\ s,t \in \mathcal{B}}} [Q_\varphi^\pi(s, a, z)] + \mathbb{E}_{t \in \mathcal{T}} [\mathcal{H} [p_\phi(z | t)]]$$


Off-policy objectives to update policy and embedding networks parameters (above), and for variational inference network (below).

$$\hat{L}(\psi) = \mathbb{E}_{\pi_\theta(a,z|s,t)} \left[\sum_{i=0}^{\infty} \gamma^i \log q_\psi(z | a, s^H) \mid s_{i+1} \sim p_\pi(s_{i+1} | a_i, s_i) \right] \longrightarrow q_\psi(z | a, s_i^H)$$



Reusing previous observations

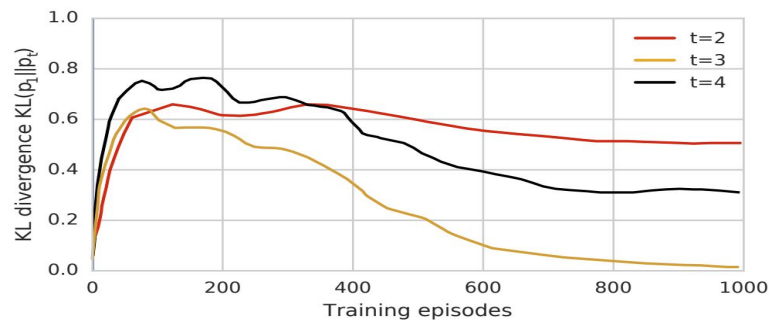
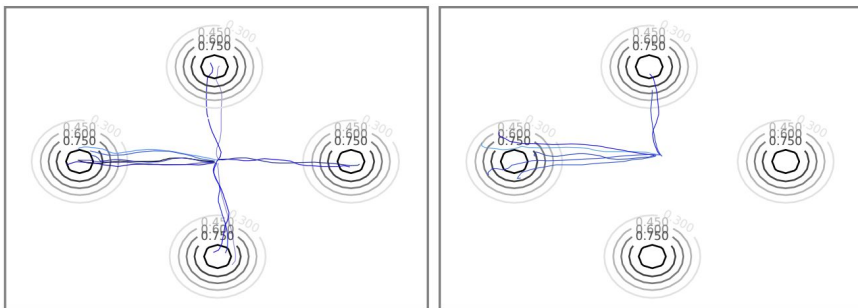
Environment interaction used to train the “main” loss $L(\theta, \phi, \psi)$ are stored in the reply buffer, used in the Q computational process, and so been recycled in order to update parameters of all the involved networks in a sample efficient way.

RESULTS: DIDACTIC TASKS

Once the skill-embedding is learned using the described multi-task setup, there would be possibilities to employ them in different scenarios, including *fine-tuning the entire policy* or **learning only a new mapping to the embedding space**, which is the aspect authors decided to focus on: the network was allowed only to learn how to modulate and interpolate between the already-learned skills, never changing the underlying policies.

The implementation consisted in 16 asynchronous workers interacting with the environment and, synchronous updates utilizing the replay buffer data. First bunch of experiments consisted in mass point tasks (aka bringing a block into a rewarded target region).

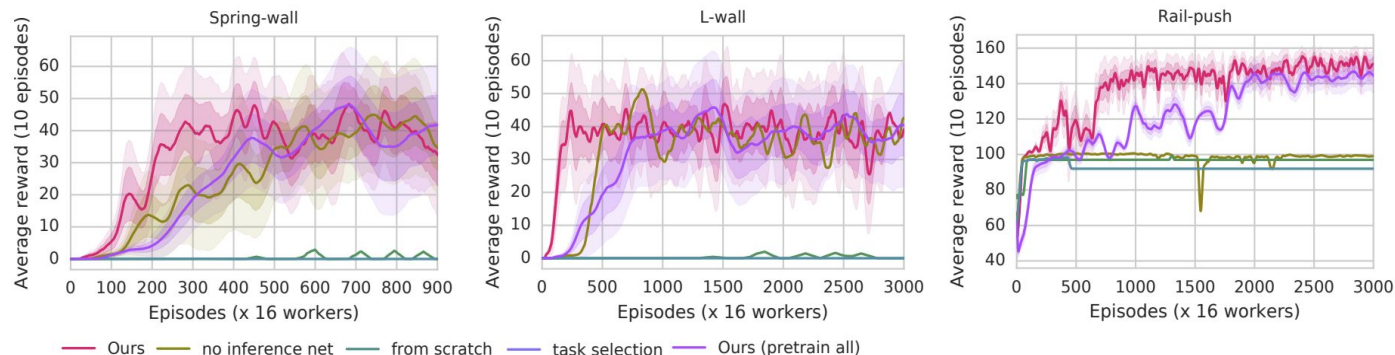
Two initial embedding distributions were tested, with a Gaussian discovering *two* solutions and a Bernoulli discovering *all four*.



KL-divergence between learned embedding distributions was evaluated over training iterations, for a fixed task ($t=1$) and three others ($t1 == t3$).

RESULTS: COMPLEX TASKS

A second bunch of simulations was carried on three complex tasks, to (successfully) confirm the model capacity of reuse skill embedding spaces previously learned on two simpler tasks. A video was published by the authors.



All the three tasks were also accomplished by the same identical model pre-trained on all the six simpler tasks.

So, having access to better exploration policies (that were encoded in the skill embedding space) resulted to be an advantage for the task accomplishment, covering a wider solution space.

Spring-wall

Model learnt to move to a spring-attached block to target region hidden by a wall. Capability of **interpolate skills** embeddings was proven, being previously trained only without wall and without spring.

L-Wall

Model learnt to overcome an L shaped wall pushing and lifting a block. Beneficial effect of learning **intuitively unrelated skills** embedding was proven.

Rail-push

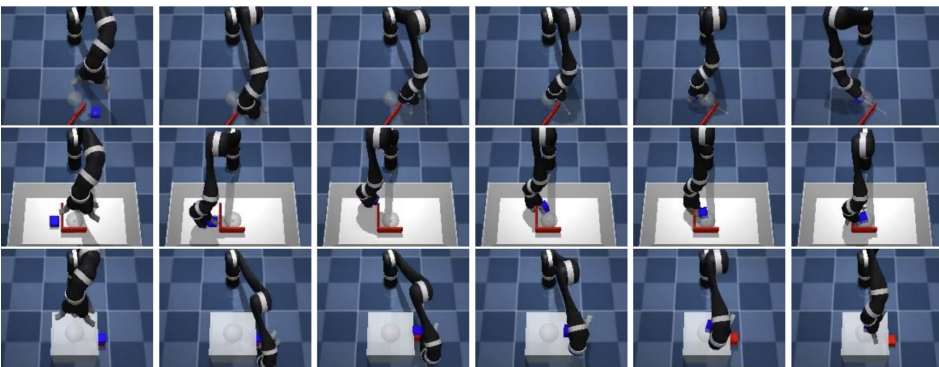
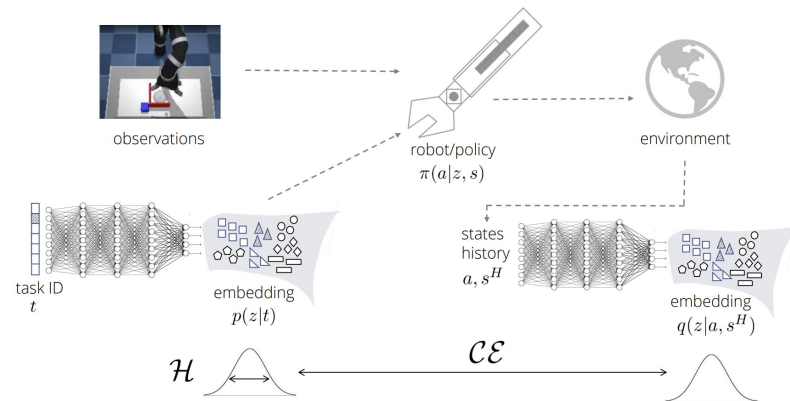
Robot arm was required to lift a block up to the side of a table and then push it to its centre. Model was pre-trained on "vertical rail push" and push to the center of the table. Task aims to demonstrate the ability to **sequence two different skills** to accomplish a new task.

TAKE HOME MESSAGES

Authors proposed a manipulation skills learning method, exploiting latent variables and solving the robot control problem in the embedding space rather than in the action one.

Many **Bayesian learning** inspired ideas (met in the course) were recognized.

Entropy regularization allowed the exploration of different possible solution, while the overall setup confer higher order capabilities like skill *sequencing* or *interpolation*. Data efficiency was also take into account, with the buffered architecture to share training experiences within all the involved NNs. Finally Q-learning and variational inference were brillianti used to solve intractability problems, greatly lowering the sampling effort.



ABOUT THE ASSIGNMENT



Intersection with Robotics

RL theoretical concepts applied in a domani of (personal) interest in the end returned a better overall vision.



AI top conference paper

The midterm allowed to experience the study of high level paper; both difficulties and motivations emerged.

BIBLIOGRAPHY - 1

1

Ronald J Williams. **Simple statistical gradient-following algorithms for connectionist reinforcement learning**. Machine learning, 8(3-4):229–256, 1992.

2

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. **Asynchronous methods for deep reinforcement learning**. In International Conference on Machine Learning (ICML), 2016.

3

Remi Munos, Tom Stepleton, Anna Harutyunyan, and Marc G. Bellemare. **Safe and efficient off-policy reinforcement learning**. In Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, pp. 1046–1054, 2016. URL

4

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. **Human-level control through deep reinforcement learning**. Nature, 518(7540):529–533, 2015.

5

Diederik P Kingma and Max Welling. **Auto-encoding variational bayes**. arXiv preprint arXiv:1312.6114, 2013.

6

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. **Stochastic backpropagation and approximate inference in deep generative models**. In Proceedings of the 31st International Conference on Machine Learning (ICML), 2014.

7

Nicolas Heess, Gregory Wayne, David Silver, Tim Lillicrap, Tom Erez, and Yuval Tassa. **Learning continuous control policies by stochastic value gradients**. In Advances in Neural Information Processing Systems, pp. 2944–2952, 2015.