Learning Skill Templates for Parameterized Tasks

Jan Hendrik Metzen and Alexander Fabisch
Robotics Group, University Bremen
Robert-Hooke-Str. 5, D-28359 Bremen, Germany
{*jhm,afabisch*}*@informatik.uni-bremen.de*

We consider the problem of learning *skill templates* for a parameterized reinforcement learning problem class T. That is, we assume that a task, i.e., an instance of the problem class, is defined by a task parameter vector $\tau \in T \subseteq \mathbb{R}^n$ and an associated interpretation. Likewise, a skill is considered as a parameterized policy with parameter vector $\theta \in \mathbb{R}^m$. A *parameterized skill* [1] is a mapping Θ from task vector τ to a skill vector θ_τ, i.e., $\Theta : \tau \mapsto \theta_\tau$. Let $J(\theta, \tau)$ be the expected return of the skill parametrized by θ in task τ; the goal of parameterized skill learning is to find a mapping Θ^* such that $\Theta^* = \arg\max_\Theta \int P(\tau) J(\Theta(\tau), \tau) d\tau$, where $P(\tau)$ is the task distribution. Because the parametrized skill Θ will typically not predict the optimal $\theta^*_\tau = \arg\max_\theta J(\theta, \tau)$, it is desirable to not only learn a point-estimate of θ^*_τ but also to give a measure of uncertainty of this prediction. We propose to learn a so-called skill template $\Psi = (\Theta, \Omega)$, which contains a function $\Omega : \tau \mapsto \Sigma_\tau$ with $\Sigma_\tau \in \mathbb{R}^{m \times m}$ that provides this uncertainty. Σ_τ can be interpreted as the covariance of a Gaussian distribution over the skill's parameter space. Thus, a skill template Ψ can be seen as a mapping from a task to a Gaussian distribution over the skill parameter space, with Θ predicting the distribution's mean and Ω predicting the distribution's covariance.

Skill templates are learned based on a set of skill weights that have been learned for specific task instances. Let $E = \{(\tau_i, \theta_{\tau_i}) | i = 1, \ldots, K\}$ be a training set consisting of experience collected in K tasks with $J(\theta_{\tau_i}, \tau_i) \approx J(\theta^*_{\tau_i}, \tau_i)$. Learning the parameterized skill Θ can be considered as a regression problem, trained with the pairs in E. While da Silva et al. [1] used Support Vector Regression for this regression task, we use Gaussian Process Regression (GPR) since it naturally provides an uncertainty along with each prediction. Different ways of learning Ω from E are imaginable; in this abstract, we only consider the case of diagonal Σ_τ with Σ_τ either being a multiple of the $m \times m$ identity I_m, i.e., $\Sigma_\tau = cI_m$, or with $(\Sigma_\tau)_{jj}$ being the uncertainty of the GPR's prediction for the j-th dimension of θ_τ.

We represent skills by *dynamical movement primitives* (DMPs) [2] and use the reinforcement learning method PI^2 [4] for learning in the training tasks $\tau_1, \ldots, \tau_K$ and thus generating E. PI^2 is a direct policy search method which requires to specify an initial policy (often obtained by learning-by-imitation or by setting all weights to zero) and a covariance matrix which governs exploration in weight space (often a multiple of the identity matrix). When faced with a new task τ, either the parameterized skill's prediction $\theta_\tau = \Theta(\tau)$ can be used as skill parameters or the skill parameters can be learned by means of PI^2. While the former might suffer from generalization errors, the latter might require too many trials to be practical. Skill templates provide a reasonable compromise: instead of using the standard PI^2 initialization, the skill template's prediction θ_τ can be used for the initial policy and the skill template's covariance Σ_τ as exploration matrix. This allows to explore more strongly in dimensions of the skill parameters where the GPR's prediction has larger uncertainty.

We investigate the *hypothesis* that close-to-optimal parametrized skills can be learned from a small training set E and that skill templates based on GPR uncertainty can considerably reduce the sample complexity of policy search methods like PI^2. We present preliminary results on a simple benchmark problem: in this problem, a trajectory, e.g., for an end-effector of a robotic manipulator, must be learned that goes from position $(0, 0, 0)$ to $(1, 1, 1)$ within 1 second under a minimum jerk objective and by passing close-by two viapoints $v_1 \in [0.3, 0.6]^3$ and $v_2 \in [0.4, 0.7]^3$ at time $t_1 \in [0.1, 0.3]$ and $t_2 \in [0.4, 0.6]$. The cost function that we use in this benchmark is similar to the reward function used by Kober and Peters [3]. It penalizes the distances to the viapoints at time steps t_1 and t_2, the squared acceleration at each time step, and the distance to the goal and the squared velocity at the end of the movement. Each task corresponds to a specific combination of v_1, v_2, t_1, and t_2, which are drawn uniform randomly from the respective value ranges. The skill weights θ_τ in the training set E have been learned using 500 rollouts with PI^2.

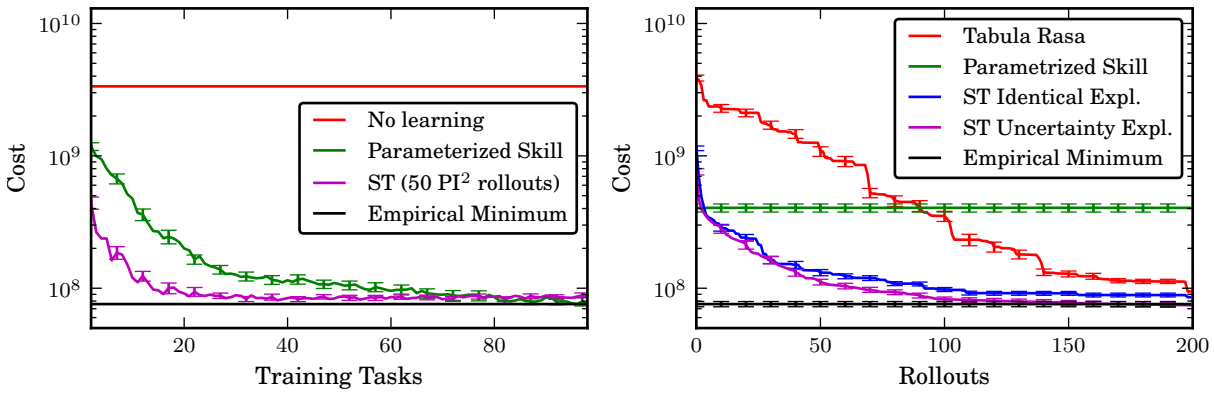

Figure 1: Left: Quality of parametrized skill and skill template for varying number of training tasks K. Right: Learning curves of PI^2 under different initializations for $K = 10$ and averaged over 100 combinations of source and target tasks. Shown are mean and standard error of mean. "Empirical minimum" shows the minimum cost obtained by PI^2 in 500 rollouts.

The left graph in Figure 1 depicts the relation of the number of training tasks K and the performance for skill weights learned with different approaches: skill weights predicted by the parameterized skill Θ and skill weights learned after 50 rollouts with PI^2 starting from Θ's prediction and using the skill template's uncertainty for exploration ("ST (50 PI^2 rollouts)"). The figure shows that (a) the parameterized skill predicts reasonable policies even for small K and (b) the skill template allows to learn considerable better policies for small K even after only 50 additional rollouts in the target task. For larger K, the difference between the performance of the parameterized skill's prediction and the performance obtained after 50 rollouts starting with the skill template becomes insignificant as both reach close-to-optimal performance. However, the left side of the plot is more relevant since in applications, the number of training tasks K will typically be small.

The right graph in Figure 1 depicts learning curves for $K = 10$. Shown are PI^2 in the tabula rasa case and PI^2 with the skill template initialization with $\Sigma_\tau = cI_m$ ("ST Identical Expl.") and with Σ_τ based on the GPR's uncertainty ("ST Uncertainty Expl."). "Parametrized Skill" shows the average cost of the parameterized skill's prediction θ_τ. One can see that it takes tabula rasa PI^2 approx. 100 rollouts to reach the parametrized skill's performance. Thus, the parameterized skill alone saves approx. 100 rollouts. Furthermore, basing exploration on the GPR's uncertainty performs better than using the standard exploration with uniform covariance. "ST Uncertainty Expl." reaches the best tabula rasa performance after approx. 150 rollouts while all other settings take considerably longer.

Future work is to evaluate other approaches for estimating the exploration covariance mapping Ω (in particular non-diagonal Σ_τ) and to evaluate the approach on more challenging and realistic benchmarks.

Acknowledgment This work was supported through a grant of the German Federal Ministry of Economics and Technology (BMWi, FKZ 50 RA 1217).

References

[1] B. C. da Silva, G. Konidaris, and A. G. Barto. Learning parameterized skills. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, 2012.

[2] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373, 2013.

[3] J. Kober and J. Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1–2):171–203, 2011.

[4] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11:3137–3181, 2010.