Focused Skill Discovery: Using Per-Factor Empowerment to Control State Variables

Jonathan Colaço Carr *†
McGill University
Montreal, QC
jonathan.colacocarr@mail.mcgill.ca

Wendy Sun†
Massachusetts Institute of Technology
Boston, MA
wendysun@mit.edu

Cameron Allen
UC Berkeley
Berkeley, CA
camallen@berkeley.edu

Abstract

Skills are essential for unlocking higher levels of problem solving. One popular approach for discovering skills is to maximize the mutual information between the agent's choice of skill and the state of the environment, empowering the agent to control its environment. However, these approaches fail to make use of the natural state variables that exist in many reinforcement learning domains. We present a general method that allows these algorithms to discover *focused skills*, which we define as skills that control specific state variables. Applied to Variational Intrinsic Control, our method improves state space coverage by a factor of three, unlocks new learning capabilities and automatically avoids side effects on downstream tasks.

Keywords: Skill Discovery, Hierarchical RL.

Acknowledgements

The authors would like to thank Jacy Anthis, Anand Siththaranjan, Atrisha Sarker, Lucas Hansen, Siddharth Hiregowdara, Tianyi Qiu, Sandy Tanwisuth, and Stuart Russell for helpful discussions and feedback.

*Correspondence to jonathan.colacocarr@mail.mcgill.ca and camallen@berkeley.edu
†Work done while at the Center for Human-Compatible AI

1 Introduction

Skills are learned behaviours that allow an agent to decompose a challenging problem into a set of easier sub-problems. In reinforcement learning (RL), a key challenge is *skill discovery*: finding a useful collection of skills, either from the agent's experiences or from an explicit task description. The main difficulty stems from needing to determine the best way to decompose a given problem before the agent has been told which problem to solve.

Fortunately, many existing RL domains come with a problem decomposition built in, only in terms of states, rather than actions. State information is commonly *factored* into a list of distinct state variables. For example, the state of a cart-pole balancing task might be specified in terms of the positions of the cart and pole, along with their respective linear and angular velocities. The state of a board game task might be specified as a list of locations and piece types. The underlying structure in such a domain presents an opportunity for addressing the problem of skill discovery.

Useful skills empower the agent to control its environment. For this reason, it has become popular to discover skills that maximize the mutual information between the agent's choice of skill and the state of the environment. In other words, skills are thought to be useful if they can reliably reach different states. However, while such approaches are effective at generating a diverse set of behaviours, those behaviours tend not to provide the agent with much actual control over the individual state variables of its environment. The problem is that these methods treat the agent's state representation as a single monolithic entity. This leads to poor state coverage, unwanted side effects, and reduced learning efficiency on downstream tasks.

This paper introduces a general method for making empowerment-based skill discovery algorithms better at learning useful skills. Rather than treating the agent's state as a unified whole, we leverage the factored state representation to learn skills focused on changing just one state variable at a time. We apply our method to an existing skill discovery algorithm, Variational Intrinsic Control [1], improving exploration efficiency by a factor of 3. In downstream tasks, we show that these skills can solve problems that their un-focused counterparts struggle with and can automatically avoid side effects with no modification to the agent's goal.

2 Background

We consider skill learning in a Controlled Markov Process $(\mathcal{S}, \mathcal{A}, P)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, and $P : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0, 1]$ is the transition probability function. We assume states are factored into n different variables, $s = (s^1, \ldots, s^n)$, where each state variable s^i is in $\mathbb{R}^{d_i}$. A *skill* consists of a name—an element z of an index set $\mathcal{Z}$—and an associated policy $\pi_z(a|f(h))$ that specifies a particular decision rule for choosing actions based on some function f of the agent's history h. This function might count elapsed time steps, detect whether certain salient events have occurred, or simply return the current state. Each skill has a special *termination* action that ends the skill at the current state. The index set $\mathcal{Z}$ can either be finite (e.g. the integers 1 through n) or continuous (e.g. $\mathbb{R}^d$). In downstream tasks (after skill learning), we augment the controlled Markov process with a scalar reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ to form a Markov decision process, and train a meta-policy to select from among the learned skills.

Discovering Skills through Empowerment. One popular way to evaluate skills is through the lens of the agent's empowerment, defined as the maximum mutual information between skills and states. Intuitively, a useful set of skills ought to increase the agent's influence over its environment, so we can search for a set of skills that maximally increase the agent's control. The most relevant starting point for our use case is Variational Intrinsic Control [1], whose objective depends on the start state and only rewards skills once they terminate. This incentivizes the *net effect* of skills to be different. The VIC algorithm maximizes the conditional mutual information $I(Z; S_T|s_0)$, where s_0 is a starting state, Z is sampled from a distribution ν over skills and S_T is sampled from the distribution $\rho_Z(s_0)$ induced by following skill Z from start state s_0 until termination. Since this mutual information is difficult to compute, VIC maximizes the lower bound developed by [2]:

$$\begin{aligned} I(Z; S_T|s_0) &= -H(Z|S_T, s_0) + H(Z|s_0) \\ &= \mathbb{E}_{Z\sim\nu(\cdot|s_0), S_T\sim\rho_Z(s_0)}[\log p(Z|S_T, s_0)] - \mathbb{E}_{Z\sim\nu(\cdot|s_0)}[\log \nu(Z|s_0)] \\ &\geq \mathbb{E}_{Z\sim\nu(\cdot|s_0), S\sim\rho_Z(s_0)}[\log d(Z|S_T, s_0)] - \mathbb{E}_{Z\sim\nu(\cdot|s_0)}[\log \nu(Z|s_0)]. \end{aligned}$$

Here $H(\cdot)$ and $H(\cdot|\cdot)$ are the entropy and conditional entropy. The *skill discriminator* $d(Z|S_T, s_0)$ is a learned estimate of $p(Z|S_T, s_0)$, the posterior distribution over skills. The discriminator tries to predict which skill was chosen given the start state and final state. This lower bound gets tighter as d approaches the posterior distribution over skills. Therefore, the goal of information-driven skill discovery is to learn *both* the skill policies *and* a good skill discriminator. As shown in [1], the lower bound is reached by training a set of skill policies to maximize the reward

$$r_{\text{VIC}}(z, s_T, s_0) = \log(d(z|s_T, s_0)) - \log \nu(z|s_0), \tag{1}$$

while updating d to estimate the posterior distribution over skills.

3 Related Work

Empowerment quantifies how much an agent can influence its environment. Its original definition measured the maximum control of a fixed action sequence over the environment [3]. This definition was extended by Gregor et al. [1] to measure the maximum control of *skill* sequences — which can select different actions based on the environment state — leading to the first skill discovery algorithm based on empowerment: Variational Intrinsic Control. A stream of related work followed, including the "Diversity Is All You Need" algorithm [4] and the Dynamics-Aware Discovery of Skills [5]. These methods do not take advantage of the factored nature of many RL domains. Our key contribution is to show that this factorization can be used to learn significantly better skills.

When the agent's behavior has unintended consequences, we often call these consequences *side effects*. Allen et al. [6] showed that such side effects can degrade planning efficiency, and introduced a method to learn focused skills that tend to perform much better than side-effecting ones. While Allen et al.'s approach is limited to deterministic planning problems, our method learns skills that avoid side effects in stochastic reinforcement learning environments. Beyond learning and planning, there are also obvious safety benefits to avoiding side effects and a number of algorithms mitigate these effects [7, 8, 9]. We use the Distance Impact Penalties [9] approach to avoid side effects, which penalizes the differences between the agent's start and end states. The focus of these algorithms has traditionally been to minimize the side effects of a single policy using a specific utility function. Our goal is to automatically discover a collection of skill policies, each with minimal side effects, and show how this can leads to gains in both safety and problem-solving on downstream tasks.

4 Focused Skill Discovery

Focused skills ought to control specific state variables. Therefore, we assign each focused skill a *target variable*, the state variable it should control. Our goal is to achieve *per-variable empowerment*, which we define as the maximum mutual information between a skill and the values of its target variable.

Algorithm 1 Focused Variational Intrinsic Control

Assume i is the target factor
for episode $= 1, M$ **do**
  Sample s_0 from the initial state distribution μ
  Sample skill z from $\nu(\cdot|s_0, i)$
  Follow policy π_z until termination state s_T
  Update the skill discriminator d_i from (z, s_T^i, s_0^i)
  Calculate the reward $r(z, s_T, s_0)$ using Eq. (2)
  Update π_z to maximize r
  Reinforce option prior $\nu(\cdot|s_0, i)$ based on r
end for

To maximize mutual information for each target variable, we learn a separate skill discriminator $d_i(z|s_0^i, s_T^i)$ for every target variable i. This discriminator observes the initial and final values of the state variable i, and estimates the probability of *only those skills* that target variable i. Restricting the skill discriminator's estimates in this way allows skills with different target variables to be learned *in parallel*, since the skill discriminator updates no longer depend on all skills. This parallelization is not possible with previous information-driven approaches because a single discriminator learns to predict all skills. To ensure that only the target variable is modified, skills are penalized if they terminate in a state that differs from the start state on non-target variables. For a skill z which targets variable i, the *focused* VIC reward is

$$r(z, s_T, s_0) = \log(d_i(z \mid s_T^i,\ s_0^i)) - \log(\nu(z|s_0)) - \|s_T - s_0\|_{\boldsymbol{\lambda},2}, \tag{2}$$

where $\|\cdot\|_{\boldsymbol{\lambda},2}$ is a weighted ℓ_2 norm with weights $\boldsymbol{\lambda} \in \mathbb{R}^{d_1 \times \cdots \times d_n}$ controlling the penalty strength. Using a weight matrix to control the penalty strength is useful since each state variable may have a different range of values. In our experiments, we chose a *single hyperparemeter* $\lambda > 0$ and set λ_{ij} equal to λ divided by the maximum ℓ_2 norm between any two values on variable j. So, for a state variable whose values range from 0 to k, the weight was λ/k. This ensures that the strength of the side effect penalty for each variable is between 0 and λ.The values of $\boldsymbol{\lambda}$ on the target variable are set to zero so that changes on the target variable are not penalized.

There are two key differences between focused VIC reward in Equation 2 and the VIC reward in Equation 1. First, the focused reward restricts the skill discriminator estimates to a specific target variable. Second, the focused reward penalizes skills if they terminate in a state that is different from the start state on non-target variables. These modifications encourage skills to change just one state variable at a time. Algorithm 1 shows how to modify VIC with the focused VIC reward.

5 Experiments

In this section we showcase the benefits of focused skills by applying our method to Variational Intrinsic Control and comparing the focused skills with unfocused skills in two GridWorld environments. Our method produces skills that control individual state variables and improve the state-space coverage of the learned skills. In downstream tasks, focused skills unlock new learning capabilities and can minimize side effects without changing the agent's goal.

Environments. We learn skills in the FourRooms and ForageWorld environments shown in Figure 1. These environments have four "primitive" actions that move the agent in each of the cardinal directions. When an agent selects an action it moves in one of the other three directions with probability 0.1. The FourRooms environment has the same map as Sutton et al. [10], with four added tools that the agent can pick up (shown in pink). The ForageWorld environment contains resources (yellow), as well as green cells that should be avoided. While tools are always picked up if the agent moves to a tool's location, resources are stochastically generated.

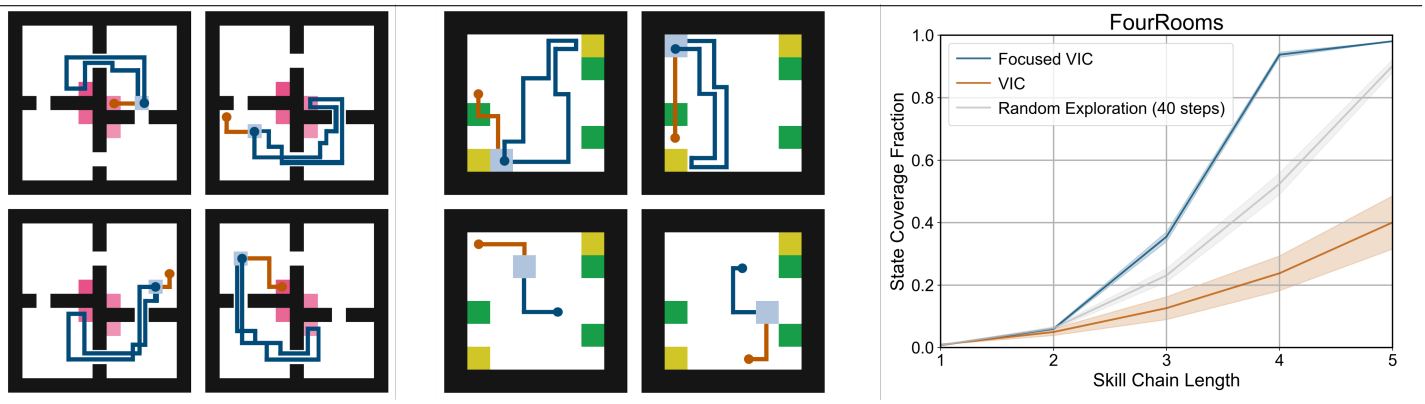

Figure 1: [Left, Middle] Visualization of Focused VIC (blue) and VIC (brown) skills in FourRooms and ForageWorld environments. Skills start in the grey square and terminate at the circles. Focused skills that collect tools (pink) and resources (yellow) return to the start state to avoid side effects. [Right] State coverage in the FourRooms environment. Focused VIC skills explore three times more efficiently than VIC skills (AUC 1.85 vs. AUC 0.62).

The agent can try collecting a resource by attempting to move to a resource square. It obtains the resource with probability 0.7 and stays in place otherwise. Green cells are destroyed if an agent moves to a position with a green cell. The states in the FourRooms and ForageWorld are composed of five and six state variables, respectively: one variable for the agent's position and one variable for each tool, resource, or green cell.

Training Details. We trained all policies using tabular Q-learning with ϵ-greedy exploration and a discount factor of 0.99. We trained 16 skills for both VIC and Focused VIC algorithms ($|\mathcal{Z}| = 16$). Following the experiments of Gregor et al. [1], the skill distribution ν is uniform over all skills and held constant during training. The skill discriminators for both algorithms made predictions using an exponentially weighted moving average of the previous samples. For all focused skills we used $\lambda = 5$ as the penalty strength hyperparameter. To select target variables for the focused skills, we assigned two skills for each tool or resource, and used the remaining skills (8 in FourRooms, 12 in ForageWorld) to target the position variable. In the downstream tasks (Section 5.2), we trained a *meta-policy* to select which skill to execute. Once selected, skills proceed until they terminate. The meta-policy had access to either the focused VIC or VIC skills. We also considered meta-policies that could choose primitive actions in addition to skills.

5.1 Analysis of Learned Skills

Focused Effects. Sample trajectories from eight focused skills are shown in the left and middle plots of Figure 1. As expected, the focused skills targeting tools in FourRooms learn to collect the tools and return to the start state, avoiding side effects on the position state variable. The ForageWorld resource skills also return to the start state, and navigate around green cells. These behaviours are very different from VIC skills, which often change a combination of the state variables.

State-space Coverage. To measure state-space coverage, we used the Area Under the Curve (AUC) of the State Coverage vs. Skill Chain Length graph, shown in Figure 1. We define the state coverage fraction for a skill chain of length l from start state s_0 as number of unique final states that can be reached after executing all possible skill combinations of length l, divided by the total number of states. We plotted the mean state coverage fraction over 10 random start states. The Focused VIC skills are three times more efficient at exploration than the VIC skills (AUC 1.85 vs. AUC 0.62). After just 4 skill executions, the focused skills can cover *93.8%* of the state space, compared to *23.8%* coverage with the normal VIC skills.

5.2 Performance on Downstream Tasks

Focused skills significantly improve performance on the two downstream tasks we considered. In the FourRooms environment, the agent's goal is to pick up all four tools and navigate to the bottom-right corner. In the ForageWorld environment, the agent must collect both resources and navigate to the bottom right corner. There is *no penalty* for destroying the green cells. In both cases, the agent starts in the top-left corner and receives a (sparse) reward of +1 for accomplishing the task. We conduct 200 independent training runs for each of the five agents we consider, plotting the mean and 95% confidence intervals of our results in Figure 2.

Improved Problem Solving. Agents with focused skills find much better solutions to their problems. Without primitive actions, the agent with VIC skills fails to accomplish its goal in FourRooms. In contrast, the agent with focused VIC skills takes fewer than 150 steps, which corresponds to *fewer than six skill executions*. Focused skills also more efficient in the ForageWorld task, taking about 13 fewer steps than the baseline VIC skills (32.4 steps vs. 45.6 steps).

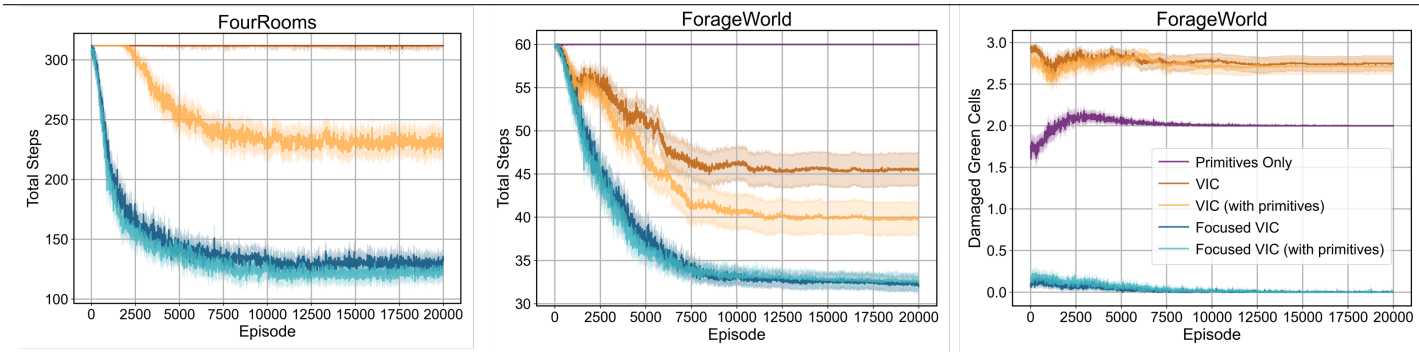

Figure 2: [Left, middle] Learning performance in FourRooms and ForageWorld. Focused VIC skills lead to faster learning on both tasks. The agent with VIC skills (and no primitive actions) fails to reach the goal in FourRooms. Agents with only primitive actions don't reach the goal in either domain. [Right] Side effects to green cells in ForageWorld are minimized with focused VIC skills.

Avoiding Side Effects. One of the best features of focused skills is that they naturally avoid side effects. As shown in the right of Figure 2, the agent with focused skills *almost never* damages green cells while learning to solve the downstream task, despite the agent never being rewarded for avoiding green cells. This is a stark improvement over both the un-focused skills and primitive agent, which typically break between two and three green cells every episode.

6 Discussion & Conclusion

We presented focused skill discovery, a method that enables empowerment-based skills to control specific state variables. Applied to Variational Intrinsic Control, we showed that our method triples exploration efficiency, unlocks new learning capabilities, and naturally avoids side effects without changing the agent's goal. In the future, we plan to apply this method to other skill discovery algorithms and experiment with larger environments. While pre-defined state variables play a key role, we hope to extend this idea to include other kinds of state abstractions.

References

[1] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv*, art. 1611.07507, 2017.

[2] David Barber and Felix Agakov. The im algorithm: a variational approach to information maximization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'03, page 201–208, Cambridge, MA, USA, 2003. MIT Press.

[3] A.S. Klyubin, D. Polani, and C.L. Nehaniv. Empowerment: a universal agent-centric measure of control. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 128–135 Vol.1, 2005. doi: 10.1109/CEC.2005.1554676.

[4] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

[5] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *International Conference on Learning Representations*, 2020.

[6] Cameron Allen, Michael Katz, Tim Klinger, George Konidaris, Matthew Riemer, and Gerald Tesauro. Efficient black-box planning using macro-actions with focused effects. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4024–4031, 2021.

[7] Alex Turner, Neale Ratzlaff, and Prasad Tadepalli. Avoiding side effects in complex environments. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21406–21415. Curran Associates, Inc., 2020.

[8] Victoria Krakovna, Laurent Orseau, Richard Ngo, Miljan Martic, and Shane Legg. Avoiding side effects by considering future tasks. *Advances in Neural Information Processing Systems*, 33:19064–19074, 2020.

[9] Charlie Griffin, Joar Max Viktor Skalse, Lewis Hammond, and Alessandro Abate. All's well that ends well: Avoiding side effects with distance-impact penalties. In *NeurIPS ML Safety Workshop*, 2022. URL https://openreview.net/forum?id=3tgegVVh2j6.

[10] Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(99)00052-1. URL https://www.sciencedirect.com/science/article/pii/S0004370299000521.