Behavior Is Everything — Towards Representing Concepts with Sensorimotor Contingencies

**Nicholas Hay, Michael Stark, Alexander Schlegel, Carter Wendelken,
Dennis Park, Eric Purdy, Tom Silver, D. Scott Phoenix, Dileep George**

Vicarious AI, San Francisco, CA, USA

nick@vicarious.com

Abstract

AI has seen remarkable progress in recent years, due to a switch from hand-designed shallow representations, to learned deep representations. While these methods excel with plentiful training data, they are still far from the human ability to learn concepts from just a few examples by reusing previously learned conceptual knowledge in new contexts. We argue that this gap might come from a fundamental misalignment between human and typical AI representations: while the former are grounded in rich sensorimotor experience, the latter are typically passive and limited to a few modalities such as vision and text. We take a step towards closing this gap by proposing an interactive, behavior-based model that represents concepts using sensorimotor contingencies grounded in an agent's experience. On a novel conceptual learning and benchmark suite, we demonstrate that conceptually meaningful behaviors can be learned, given supervision via training curricula.

1 Introduction

The field of AI has seen remarkable progress in recent years, fueled in large part by advances in deep representation learning. In narrow application domains and with enough training data, deep representations deliver results that approach or exceed human performance (Krizhevsky, Sutskever, and Hinton 2012; Mnih et al. 2013; Silver et al. 2016; Lake et al. 2016).

Despite these advances, current deep representation learning-based AI still faces two main challenges:

(1) There is a clear difference between how human conceptual understanding is grounded in rich, physical, sensorimotor interaction with the environment, and the typical grounding of current deep learning systems that passively associate sensory data with textual labels. As an example, consider the human concept of *containment*, characterized by being (un-)able to move oneself or a contained object relative to the confines of a container. That is, containment is *constituted* by a particular way of interacting with a container, or more generally (in the framing of sensorimotor contingency theory): *"(concepts) are themselves techniques or means for handling what there is"* (Noë 2015).

SMC
1: cl_self-in-left-corner_vs_on-hline
2: cl_self-in-right-corner_vs_on-hline
3: ba_self-in-potential-container_from_
   left-touches-on-floor
4: ba_self-in-potential-container_from_
   right-touches-on-floor
5: ba_self-on-something_from_directly-above
6: ba_self-ontop-object_from_left-touches-
   on-floor
7: ba_self-touches-target_from_right-on-
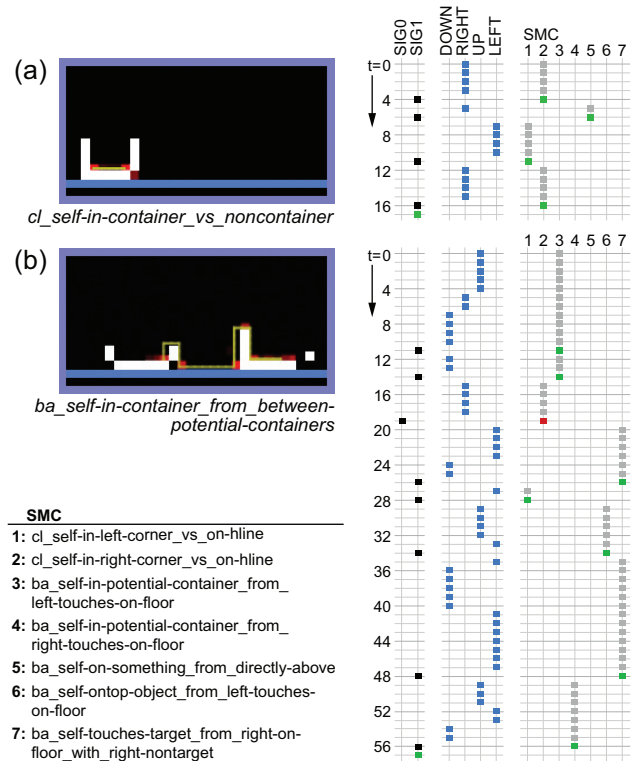   floor_with_right-nontarget

Figure 1: (Left) Example PixelWorld (PW) test environments. Red: agent (*self*), yellow: agent trajectory, white: objects, blue: floor & frame (see Fig. S1[1] for detailed annotation). (Right) Hierarchical SMC invocations rolled out over time for two concepts (see Sect. 3.3) that share the same supply of low-level SMCs from the *canonical* curriculum (see Sect. 3.4). Dots show time spent in SMCs (gray) or primitive actions: SIG_0 (red), SIG_1 (green), and direction (blue). The active 7 of 26 SMCs included in the curriculum are illustrated. Note that SMC 2 (*cl_self-in-right-corner_vs_on-hline*) is shared but used in different contexts by the two examples: it confirms the container in (a), but rules it out in (b), motivating the agent to move on to the actual container to its left (via SMC 7).

A state-of-the-art image captioning or visual question answering system (Antol et al. 2015; Karpathy and Fei-Fei 2015) would likely take note of the correlation between concave shapes and utterances of "container", "box", etc., but fail to capture the true, physical meaning of containment, limiting its ability to generalize. Such systems lack the capacity for interaction and so cannot provide an adequate general representation of concepts, such as containment, for which interaction is an essential feature. Representing such concepts is a key challenge that we seek to address.

(2) Current deep representations tend to be tailored towards specific applications. Once acquired, knowledge can hardly be reused in novel tasks, due to the lack of a common, shared representation, despite great efforts in transfer learning and domain adaptation (Pan and Yang 2010).

In this paper, we suggest a conceptual representation that directly addresses both challenges. While we demonstrate it in the context of seemingly simple simulated environments, we believe that the underlying ideas of behavior-based concept representation have broader applicability, in particular when combined with a more powerful perception system.

We address challenge *(1)* by assuming the perspective of an embodied agent that interacts with the environment through a series of actions and observations (Sutton and Barto 1998), thereby gaining grounded experience. As we show in our experiments (Sect. 5), this representation includes not only the ability to interactively test for concept presence (e.g., "is self inside a container?", Fig. 1 (a)), but also the ability to affect the environment to change whether the concept holds (e.g., "bring about containment of self", Fig. 1 (b)). We address challenge *(2)* by choosing *behavior itself* as a basic unit of representation. *Everything is a behavior* (e.g., detecting a visual feature, verifying the presence of container walls, moving into a container) that can function as a building block in conceptual knowledge representation.

In particular, our approach is based on a simplified notion of *sensorimotor contingencies* (SMCs) (O'Regan and Noë 2001; O'Regan 2011): extended behaviors that are indicative of the sensorimotor laws governing the environment. SMCs integrate perception and action into a single representational unit. Information-gathering actions, such as looking for a handle on a cup, or feeling for a hidden button, are clear examples of such integration. Similarly, many actions include elements of perception; for example, taking a step involves feedback on how that step is landed. Furthermore, many perceptual processes can benefit from close integration with action; for example, a contour may be detected by following along it with eye movements.

We approximate this notion by combining each behavior with a binary signal action: we can think of this action as the agent's own assessment of the behavior's outcome (e.g., whether a concept is present or was brought about). This representation naturally lends itself to hierarchical composition: a higher-level SMC can invoke lower-level SMCs in the same way a computer program can invoke other programs,

and is informed about their outcomes in the form of binary return values. Each higher-level SMC hence effectively abstracts away the details of lower-level SMCs' execution.

On the technical level, we draw from recent advances in reinforcement learning (RL), and formalize SMCs as hierarchical stochastic policies that are learned via policy gradient methods (Peters and Schaal 2008; Schulman et al. 2015). We use a set of related, auxiliary binary classification and bring-about-type tasks to guide exploration and compare different curricula of these tasks in our experiments (Sect. 5).

Our paper makes the following specific contributions:

(i) We introduce a representation of conceptual knowledge based on the notion of sensorimotor contingencies. It can be learned from interaction with an environment, is compositional, and encompasses both diagnostic tests for concept presence and the ability to change the state of the environment such that concept presence is achieved. *(ii)* We introduce *PixelWorld*[1] (PW), a novel, interconnected suite of conceptual learning and benchmark environments, drawn from a common distribution. In contrast to prior work emphasizing unconstrained variability (Brockman et al. 2016), PW formalizes environment generation as a family of constraint satisfaction problems, allowing for the precise characterization of both individual concepts and their relations – a vital ingredient for systematic experimentation related to interdependent concepts. *(iii)* We provide an in-depth analysis of our SMC-based representation on PW, quantifying the impact of both making the transition from passive to interactive concept representation and adding the ability to reuse behavior in hierarchical composition.

2 Related work

Our paper draws inspiration mainly from three different research directions: sensorimotor contingency theory, concept learning, and factored behavior representations in RL.

Sensorimotor contingencies. The importance of action in perception has long been acknowledged in the cognitive science and robotics communities. SMC theory (O'Regan and Noë 2001), in particular, has inspired a range of studies in which the relationships between a robotic agent's actions and sensory observations are modeled in order to learn skilled behaviors or to improve the quality of its state predictions. These studies tend to focus on narrow problem domains, including classifying objects according to their physical responses to manipulation (Hogman, Bjorkman, and Kragic 2013), segmenting objects via push-induced object movements (Bergström et al. 2011; Van Hoof, Kroemer, and Peters 2013), learning to navigate (Maye and Engel 2011; 2012; 2013), learning to manipulate objects in a generalizable manner (Sánchez-Fibla, Duff, and Verschure 2011), learning the structure of complex sensorimotor spaces such as a saccading, foveated vision system (Laflaqui 2016), and categorizing objects and their relations via programmed behaviors (Sinapov et al. 2014). (Bohg et al. 2016) provides an in depth review of robot-based sensorimotor interactions.

While this previous work operationalizes SMCs as predictive models of the effects of actions on the environ-
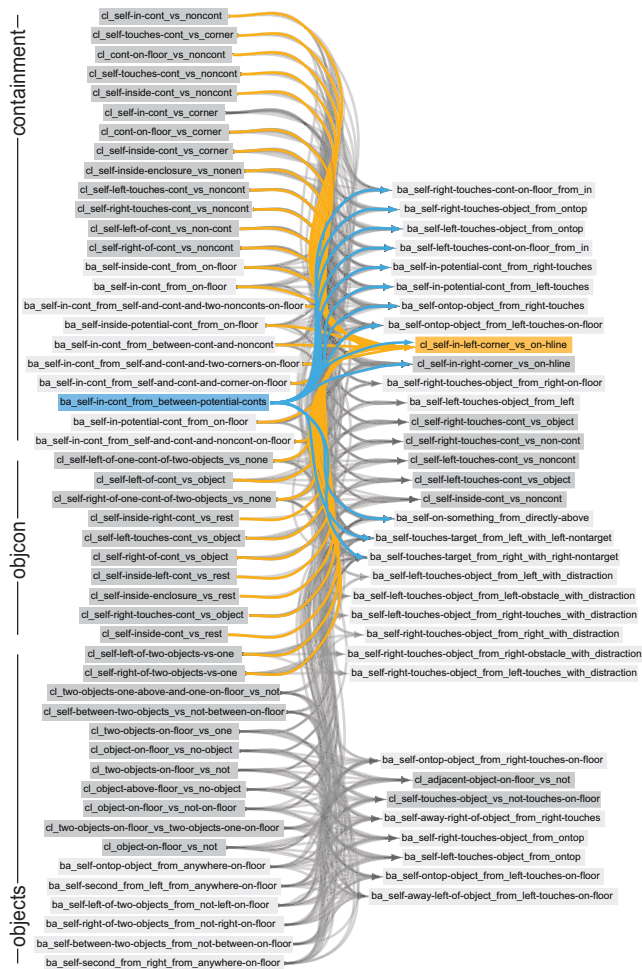

Figure 2: Concept "depends-on" relations for *canonical* curriculum, totaling 85 different concepts distributed among 2 levels. Edges demonstrate the reuse of low-level concepts (right) by high-level concepts (left). Dark nodes are classification concepts, light nodes bring-about. Blue: outgoing edges from one particular high-level concept; orange: incoming edges to one particular low-level concept.

ment state (Maye and Engel 2011; Drescher 1991), our view is distinguished by treating SMCs as sensorimotor programs that learn to characterize and affect the environment directly. Similar attempts have typically focused on isolated, low-level aspects of sensorimotor perception, such as color (Philipona and O'Regan 2006), space (Philipona, O'Regan, and Nadal 2004), or object affordances (Gibson 1977; Grabner, Gall, and Van Gool 2011). In contrast, our approach utilizes hierarchical composition of lower-level concepts to learn more complex concepts. We believe our paper to be the first to suggest a fully compositional representation of conceptual knowledge inspired by SMCs.

Concept learning. A concept is knowledge that is abstracted from one context for reuse in new contexts. Concept learning is an area of research that focuses on how conceptual knowledge is acquired. A prominent direction in concept learning develops probabilistic models of human generalization ability, applied to hypothesis spaces of various kinds, including numerical concepts (Tenenbaum 1999), relational data (Kemp et al. 2006), and visual object classes (Jia et al. 2013). While these approaches achieve remarkable results in concept learning from few examples, they are applied to passive, purely sensory data. Notable exceptions include recent work (Lake, Salakhutdinov, and Tenenbaum 2012; Ellis, Solar-Lezama, and Tenenbaum 2015) that formalizes concept learning as motor program induction, applied to character recognition and synthetic visual reasoning (SVRT) tests. In contrast, our formulation does not attempt a full, generative reconstruction of the environment, but instead derives diagnostic information from partial observations of local sensors in a partially observable Markov decision process (POMDP) setting.

Factored behavior representations. The technical implementation of our approach draws from a large body of work in RL that factors behavior into simpler building blocks. Pioneered by early work on agent architectures (Agre and Chapman 1998; Brooks 1987), factorization has been proposed in the form of hierarchical policies (Singh 1992; Barto and Singh 2004; Marthi et al. 2005; Jonsson and Barto 2005; Barto and Mahadevan 2013), spatial maps (Ring, Schaul, and Schmidhuber 2011), or more general representations that collectively implement behavior (Konidaris and Barto 2007; 2009; Simsek and Barto 2009). The novel contribution of our work lies in the duality between factored behavior representation and encoding of conceptual information. While the diagnostic tests learned by our method are related to predictive state representations and general value functions (Sutton et al. 2011; Schaul and Ring 2013; Schaul et al. 2015), we extend these notions by hierarchical reuse and compositional knowledge representation.

3 Sensorimotor contingency-based concepts

In this section, we describe the technical foundation of our SMC-based representation. We first introduce the representation of individual, flat contingencies (Sect. 3.1) and their learning (Sect. 3.2), followed by their hierarchical composition (Sect. 3.3), and training via a curriculum (Sect. 3.4).

3.1 Sensorimotor contingencies (SMCs)

We formally represent SMCs as stochastic policies in a POMDP of a particular structure. Recall that an undiscounted POMDP $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, R, \rho_0)$ consists of a finite set $\mathcal{S}$ of states, a finite set $\mathcal{A}$ of actions, a finite set $\mathcal{O}$ of observations, a function $\mathcal{P} \colon \mathcal{S} \times \mathcal{O} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ giving the probability $\mathcal{P}(s', o | s, a)$ that action $a \in \mathcal{A}$ performed in state $s \in \mathcal{S}$ results in observation $o \in \mathcal{O}$ and successor state $s' \in \mathcal{S}$, a function $R \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ giving the immediate reward $R(s, a)$ of performing action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, and a function $\rho_0 \colon \mathcal{S} \to \mathbb{R}$ giving the distribution of the initial state. A stochastic policy is a function $\pi \colon \mathcal{A} \times \mathcal{O} \to [0, 1]$ which gives the probability $\pi(a | o)$ that the policy performs
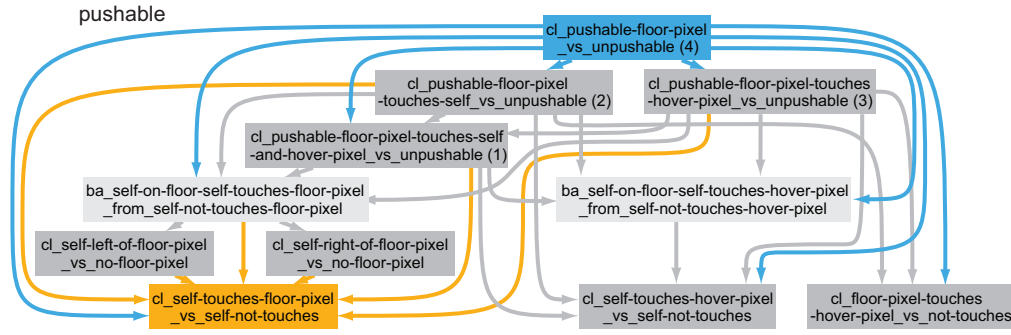
Figure 3: Concept "depends-on" relations for *pushable* curriculum, totaling 11 different concepts distributed among 6 levels and illustrating the detailed conceptual decomposition in the *pushable* curriculum leading to the concept *cl_pushable-floor-pixel_vs_unpushable*. Numerical suffixes refer to difficulty levels in Fig. 5 (c). Coloring is as in Fig. 2.

action $a \in \mathcal{A}$ given observation $o \in \mathcal{O}$. In practice, we consider parametric policies of the form $\pi_\theta(a|o)$, where $\theta \in \mathbb{R}^n$ parameterizes a neural network of a fixed structure.

Our simplified notion of sensorimotor contingencies (O'Regan and Noë 2001; O'Regan 2011) combines two basic ingredients: *a behavior* and *its outcome*, both of which are jointly represented by a stochastic policy. We further distinguish two varieties of SMC which capture two related forms of conceptual knowledge: *classification* SMCs *behave* in the environment in order to determine whether a concept is present or not ("is self inside a container?"), then signal using one of two signaling actions SIG_1 and SIG_0 (the choice of which is the SMC's *outcome*). *Bring-about* SMCs *behave* in the environment in order to bring the environment to a state in which the concept holds ("bring about containment of self"). The environment terminates either when the policy signals success with action SIG_1 (*outcome*) or when it times out after a certain number of steps.

Embodiment. In order to clearly separate the interactive aspect of concept representation from other factors, such as the impact of particular sensory features, we use a simplistic form of sensorimotor embodiment to illustrate our SMC representation. In particular, our agents perceive their two-dimensional environment through a small local observation window (3×3 pixels) centered on a "self" that can be moved through a set of actions (UP, DOWN, LEFT, RIGHT) and is represented by a vector of 9 discrete color values. They cannot directly access any other information about the environment, in particular, no global frame of reference.

Note that this embodiment requires the agent to learn even the most basic representations, such as the notion of an object, from the ground up – the presence of an object as opposed to its absence or the presence of two objects must be represented in terms of behaviors that interact with the environmental state. While this might seem like unnecessary sensory deprivation, it highlights the unique ability of our approach to compose heterogeneous behaviors into meaningful concept representations. As we show in our experiments (Sect. 5, Fig. 4 (a)), our agents learn to repeatedly apply object identification behaviors to environments with

multiple objects, which then *constitute* the representation of there being multiple objects. Furthermore, behaviors can be shared and re-used in the context of different higher-level concepts, e.g., checking for a corner to the right can be used both for identifying whether the agent is inside a container and as part of bringing about being inside a container (Fig. 1).

3.2 Learning SMCs

Since the goal of our SMC-based representation is to encode re-usable information about the environment, its success hinges on motivating the agent to learn both behaviors that are indicative of environmental states and corresponding signaling of outcomes. To that end, we introduce two kinds of reward function that encourage the formation of discriminative (classification) and environment-changing (bring-about) behaviors, resulting in corresponding SMCs.

Classification. The agent is subjected to a set of training environments with unobserved binary labels indicating concept presence or absence, and rewarded whenever it terminates an episode with the correct signal within a predetermined number of steps (assuming a fixed mapping between labels and signals). The agent receives a negative reward for terminating an episode with the incorrect signal. This reward function can be seen as a compromise between fully self-motivated exploration (Barto and Singh 2004; Hester and Stone 2012; Georgeon and Ritter 2012) and supervising full behavior.

Bring-about. The agent is tasked to bring about a change in the environment (if required) that makes the concept hold. It receives a reward when it both successfully brings about and signals that change. To speed up learning, the agent additionally receives a shaping reward (Ng, Harada, and Russell 1999) when it successfully brings about the concept. Both rewards are automatically derived during environment generation (Sect. 4), and allow for the creation of training curricula that build up more complicated conceptual representations from simpler prerequisite representations (Sect. 3.3).

concept		CSP expressions	pos. example	neg. example
cl_self-left-of-two-objects_vs_one	c	$\exists f, x, y.\ \text{floor}(f) \wedge \text{on-top}(x, f) \wedge \text{on-top}(y, f) \wedge \text{on-top}(\text{self}, f)$		
	g	$\exists f, x, y.\ \text{blue-floor}(f) \wedge \text{white}(x) \wedge \text{white}(y) \wedge \text{small-blob}(x) \wedge \text{small-blob}(y)$ $\wedge\ \text{on-top}(x, f) \wedge \text{on-top}(y, f) \wedge \text{on-top}(\text{self}, f) \wedge \neg\text{touches}(x, y)$ $\wedge\ \text{dir-right-of}(x, \text{self}) \wedge \text{dir-right-of}(y, \text{self})$		
	g	$\exists f, x.\ \text{blue-floor}(f) \wedge \text{white}(x) \wedge \text{small-blob}(x) \wedge \text{on-top}(x, f)$ $\wedge\ \text{on-top}(\text{self}, f) \wedge \text{dir-right-of}(x, \text{self})$		
ba_self-in-container_from_between-potential-containers	c	$\exists x.\ \text{container}(x) \wedge \text{inside-supported}(\text{self}, x)$		
	g	$\exists f, x, y.\ \text{blue-floor}(f) \wedge \text{white}(x) \wedge \text{white}(y) \wedge \text{on-top}(x, f) \wedge \text{on-top}(y, f)$ $\wedge\ \text{container}(x) \wedge \text{noncontainer}(y) \wedge \text{h-between}(\text{self}, x, y)$		
	g	$\exists f, x, y.\ \text{blue-floor}(f) \wedge \text{white}(x) \wedge \text{white}(y) \wedge \text{on-top}(x, f) \wedge \text{on-top}(y, f)$ $\wedge\ \text{container}(x) \wedge \text{noncontainer}(y) \wedge \text{h-between}(\text{self}, y, x)$		

Table 1: Examples of CSP-based environment generation using concept filters (*c*) applied to generator expressions (*g*) to yield environment samples (all objects, floor, and frame shown in black; positive concept region shown in green for bring-about).

3.3 Compositional hierarchies of SMCs

Having introduced SMCs as the basic building blocks of our representation (Sect. 3.1), we can now combine them into compositional hierarchies that jointly represent multiple aspects of an environment (such as the existence of a container and a distractor somewhere on the floor, Fig. 1). As an SMC consists of a behavior and an outcome, there are two ways to reuse an already learned SMC: reusing its behavior as an additional action, and reusing its outcome as an additional observation. We apply both in tandem in our experiments.

When an SMC is used as an action, the agent gives control to the SMC's policy until it signals. This allows re-use of behaviors that have already proven useful for identifying or bringing about other concepts. This composition can be nested, with SMCs called as actions calling further SMCs as actions, giving a hierarchical structure to behavior. The re-use of SMCs as actions is related to the use of options in reinforcement learning (Sutton, Precup, and Singh 1999).

Using an SMC simply as an action would discard a valuable bit of information: its outcome. To avoid this, the environment maintains a vector with a distinct entry per available SMC recording the result returned by the SMC the last time it was executed. Note that this effectively adds state to the policy, enhancing the representational power of non-recurrent policies in the POMDP setting (however, all experiments in Sect. 5 use recurrent policies).

In practice, while all SMCs are both actions and observations, the emphasis can vary. Bring-about SMCs are mainly useful for the changes they bring about in the environment, while classification SMCs are mainly useful for the information they provide. A higher-level SMC, whether classification or bring-about, typically involves a mixture of action-focused (bring-about) and observation-focused (classification) lower-level SMCs. For example, in Fig. 1, the relatively simple higher-level classification SMC *cl_self-in-container_vs_noncontainer* is composed of two lower-level classification SMCs (*cl_self-in-left-corner_vs_on-hline* and *cl_self-in-right-corner_vs_on-hline*) that test for left and right corners, and one lower-level bring-about SMC *ba_self-on-something_from_directly-above* that moves down to find a surface. Similarly, the higher-level bring-about SMC *ba_self-in-container_from_between-potential-containers* involves two lower-level classification SMCs as well as five lower-level bring-about SMCs.

3.4 Curricula

The hierarchical compositional structure lends itself to a training regime in which lower-level SMCs are trained before higher-level SMCs in a greedy fashion. In Sect. 5, we compare two ways of training. First, we train SMC hierarchies under the explicit guidance of hand-designed training curricula (i.e., defining a DAG with nodes i corresponding to training environments $\mathcal{E}_i$, such that SMCs trained on $\mathcal{E}_i$ are free to invoke SMCs trained on any environment $\mathcal{E}_j$ for which (i, j) is a graph edge, Figs. 2 & 3). Second, we explore a weaker form of supervision, by training a common set of shared SMCs for related groups of concepts in the spirit of a *canonical* curriculum. While this still involves human judgment, it is up to the agent to separate useful from potentially distracting behaviors (Fig. 1).

4 The PixelWorld concept benchmark

We introduce PixelWorld (PW), a novel learning and benchmark suite of presently 96 related sets of environments. In contrast to existing loose environment collections such as OpenAI gym (Brockman et al. 2016) or ALE (Bellemare et al. 2013), it is explicitly designed to feature re-occurring content that enables reuse of learned representations across environments. In contrast to PyVDGL (Schaul 2013), PW allows the automatic creation of entire distributions of randomized environments that exhibit certain properties, not just instances. In contrast to Omniglot (Lake, Salakhutdinov, and Tenenbaum 2016), PW is fully interactive, not a collection of static images.

Dynamics. PW instances are discrete, 2D environments (size 20×37) that are inhabited by an agent (*self*) and one or more objects of different kinds, all composed of a few *pixels* (e.g., lines, blobs, containers, and enclosures). PW inhabitants adhere to the laws of a simple physics engine, including collision detection and agent-environment interactions like pushing or grasping. PW also features multiple depth planes and resulting occlusion, although this is not explored further in the present experiments. While PW might seem simplistic at first glance, it confronts the agent with a rich repertoire of concepts that can be present, absent, or brought about in countless combinations.

4.1 Concepts and environments

Ground truth for a concept (classification or bring-about) is represented by a sample set of PW environments that is generated automatically by a rejection sampler working in tandem with a general purpose constraint satisfaction problem (CSP) (Russell and Norvig 2009) solver. Environment distributions are specified in a fragment of first-order logic, using a predefined vocabulary of unary and binary predicates that can be combined using conjunction and negation. The specification consists of two key parts: a set of generators and a concept filter. Generators are conjunctions of first-order logic expressions that specify random samples of environment specifications (up to but not including reward functions). For a given concept, to create a new environment specification, one of the generators is selected uniformly at random and then that generator is invoked. The concept filter is a first-order logic expression that filters the generated environments into those that satisfy the concept and those that do not (Table 1 gives examples).

For *classification environments*, environment specifications that satisfy the concept filter are associated with a positive classification label (rewarding $+1$ for SIG_1, -1 for SIG_0, and 0 otherwise), while those that do not are associated with a negative classification label (instead rewarding -1 for SIG_1, $+1$ for SIG_0).[2] For *bring-about environments*, there are no labels. Instead, the concept filter is evaluated at every step of execution (rewarding $+1$ iff both the expression holds and SIG_1, and 0 otherwise).

We use a short-hand notation that identifies concepts (sets of environments) by a prefix cl_ (classification), ba_ (bring-about), infix _vs_ (positive, negative generators), _from_ (starting condition), and an optional suffix _with_ describing potential distractors.

4.2 Training curricula

Figs. 2 & 3 depict two training curricula used in our experiments (Sect. 5) as directed acyclic graphs: each node is a set of environments corresponding to a binary classification or bring-about task for a particular concept, and each directed edge is a "depends-on" relation, used to train different levels of SMCs in reverse topological order. Curricula give the opportunity to reuse earlier SMCs, but don't require reuse. Fig. 5 gives experimental results for the higher-level concepts in Figs. 2 & 3.

We divide our concepts into categories depending on the contrasts they make: *containment*, contrasting containers from visually similar non-containers such as containers with holes in them; *objcon*, contrasting containers from other objects; *objects*, contrasting configurations of one or more objects; and *pushable*, contrasting immovable from pushable objects. The first three categories include large collections of first-level concepts that are included in both hand-designed and canonical curricula targeted at learning of second-level concepts. The *pushable* category includes a smaller, more focused set of concepts designed to demonstrate learning of a deeper hierarchical curriculum (up to level 6). Note that

---

[2]This filtering is balanced: equal numbers of positively and negatively classified environments are generated.

with *pushable* concepts, the agent has to assess the effects of push-related actions based entirely on changes relative to reference objects in the environment due to its limited sensors.

5 Experiments

In this section, we perform an in-depth evaluation of our approach, highlighting its ability to ground sensorimotor concepts in environmental interactions and enable effective reuse of learned representations. We give examples of learned classification and bring-about SMCs, and we quantify the impact of using SMCs in compositional hierarchies in connection with different curricula, considering classification and bring-about concepts.

Methods. We evaluate and compare the following variants of our SMC-based approach (Sect. 3):

- *smc-base*: a flat policy trained on a set of environments.

- *smc-curr*: a hierarchical policy, trained using a curriculum of multiple sets of environments (Sect. 3.4). Specifically, we manually design 1–3 intuitive curricula per concept, and report best case (maximum) performance in Fig. 5.

- *smc-canon*: a hierarchical policy, trained using a *canonical* curriculum (shown in Fig. 2) in which we choose larger sets of 11–15 SMCs for related groups of concepts.

Implementation details. We represent stochastic policies as Gated Recurrent Unit (GRU) (Chung et al. 2014) networks of a fixed size (one hidden layer with 32 units), which we learn using natural policy optimization (NPO) (Schulman et al. 2015) implemented in RLLab (Duan et al. 2016). We run NPO for 200 iterations with a batch size of 2,000 and a maximum trajectory length of 100. We initialize our networks following (Glorot and Bengio 2010), adjusting the output layer's bias to reduce the initial probability of performing a signal, increasing initial expected episode length and exploration.

Protocol. Each method is trained on a dedicated training set and tested on a held out test set (50 examples each), averaging over $r = 10$ repeated rollouts of the learned policies on each test environment, for 5 different random seeds. For each seed, test performance is evaluated on the best performing iteration (as determined on the training set), and the best test performance across seeds is reported. When hierarchical SMCs reuse lower level SMCs, they use those from the two best performing seeds (determined on the training set). Ultimately, hierarchical policies have access to both more training environments and more total training iterations than *smc-base*. While this might seem like an unfair comparison, we stress that our contribution is precisely to make this additional experience available to the agent in a way that affords re-use of previously learned representations.

(a) *cl_self-left-of-two-objects_vs_one*

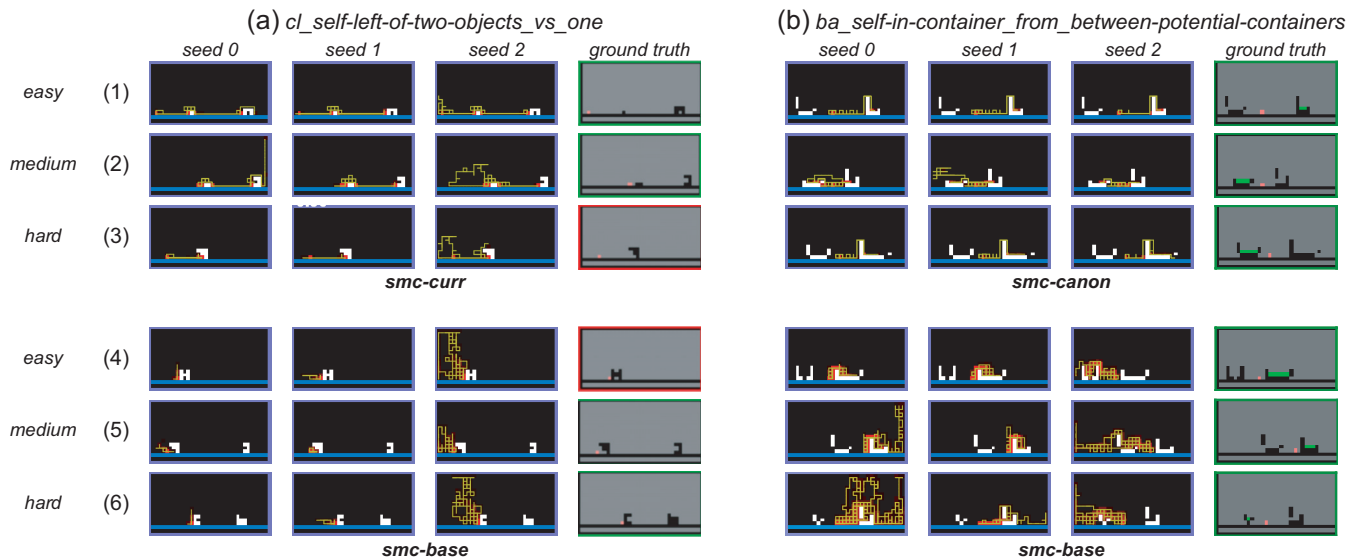**(b)** *ba_self-in-container_from_between-potential-containers*

Figure 4: Qualitative results for *smc-curr*, *smc-canon*, and *smc-base*, for two concepts: (a) *cl_self-left-of-two-objects_vs_one*, (b) *ba_self-in-container_from_between-potential-containers*. Rows correspond to test examples, showing agent trajectories (yellow) of 3 independently trained (seeded) policies, averaged over 10 rollouts. Green environment background denotes positive ground truth for classification SMCs, red negative ground truth. Per method, test examples are sorted by difficulty (average test performance across seeds), and roughly bucketed for illustrative purposes.

Evaluation. For classification concepts, we evaluate performance on a series of binary classification tasks, each corresponding to a distinct sensorimotor concept in PW (see Figs. 2 & 3). For bring-about concepts, the agent is assessed by the fraction of successful bring-about behaviors.

5.1 Examples

We begin by examining learned behavior for two specific concepts: object number classification (*cl_self-left-of-two-objects_vs_one*, Fig. 4 (a)), and bring-about containment (*ba_self-in-container_from_between-potential-containers*, Fig. 4 (b)). The former task demonstrates a large advantage of curriculum (*smc-base*: 52.6%, *smc-curr*: 90.4%, *smc-canon*: 89.0%, see Fig. 5 (b)). While *smc-base* fails to look for the second object on the right (Fig. 4 (a), rows 4,5,6), *smc-curr* reliably climbs the first object in order to verify a potential second one (rows 1,2), but sometimes gets stuck in a concavity (row 3). For the latter task, *smc-base* fails to learn a successful policy (48.4% success). In Fig. 4 (b), row 6, *smc-base* does not continue the search for a container to move into in case of a distractor on the right. In contrast, curriculum-based approaches perform well (*smc-curr*: 82.0%, *smc-canon*: 71.2%). *smc-canon* (row 2) utilizes previously learned search, climbing, and containment-checking behaviors to enter a potential container, determine that it is not a container, and move on to the real container.

5.2 Classification concepts

We add as a reference a passive CNN that (unlike the SMC methods) has access to the full, global observation of the environment, but not to any actions (2 conv + 1 FC layer $(8, 16, 8)$ with l_2 regularization)[1]. Fig. 5 (a) (ALL) gives the corresponding summary results, showing binary classification accuracy averaged across concepts for four methods: *cnn*, *smc-base*, *smc-curr*, and *smc-canon*.

Of primary interest, there is a clear advantage for curriculum (*smc-base*: 74.5%, *smc-curr*: 84.9%, *smc-canon*: 80.2%; *smc-curr* vs. *smc-base*: $t(34) = 4.9$, $p = .00002$, *smc-canon* vs. *smc-base*: $t(34) = 2.3$, $p = .025$). This advantage is most pronounced for concepts that involve multiple objects or complex sequences of behaviors to verify a given concept (e.g., *cl_self-between-two-objects_vs_not-between*, *cl_self-left-of-two-objects_vs_one*, Fig. 5 (b)). Overall SMC performance is comparable to the *cnn* (79.9%), but this overall similarity masks large differences at the level of individual concepts. Fig. 5 (a) shows classification results subdivided according to concept category (Sect. 4.2). For *containment* (containers vs. visually similar non-containers), the SMC approach clearly outperforms the *cnn*. The categories *objects* and *objcon* involve more visual distinctions that enable better performance by the *cnn* and thus less advantage for the interactive SMC approach.

For *pushable* concepts (Fig. 3, not included in the overall averages reported above), the observed advantage of the SMC approach over the *cnn* is particularly pronounced (Fig. 5 (c)), especially for *smc-curr*. Each numbered concept in Fig. 5 (c) utilizes a combination of lower-numbered concepts in its curriculum and a shared set of basic concepts (e.g., classifying whether the self is touching a particular object).

5.3 Bring-about concepts

For bring-about concepts, hierarchical SMCs outperform flat SMCs for 14 out of 16 tasks, with a large average difference
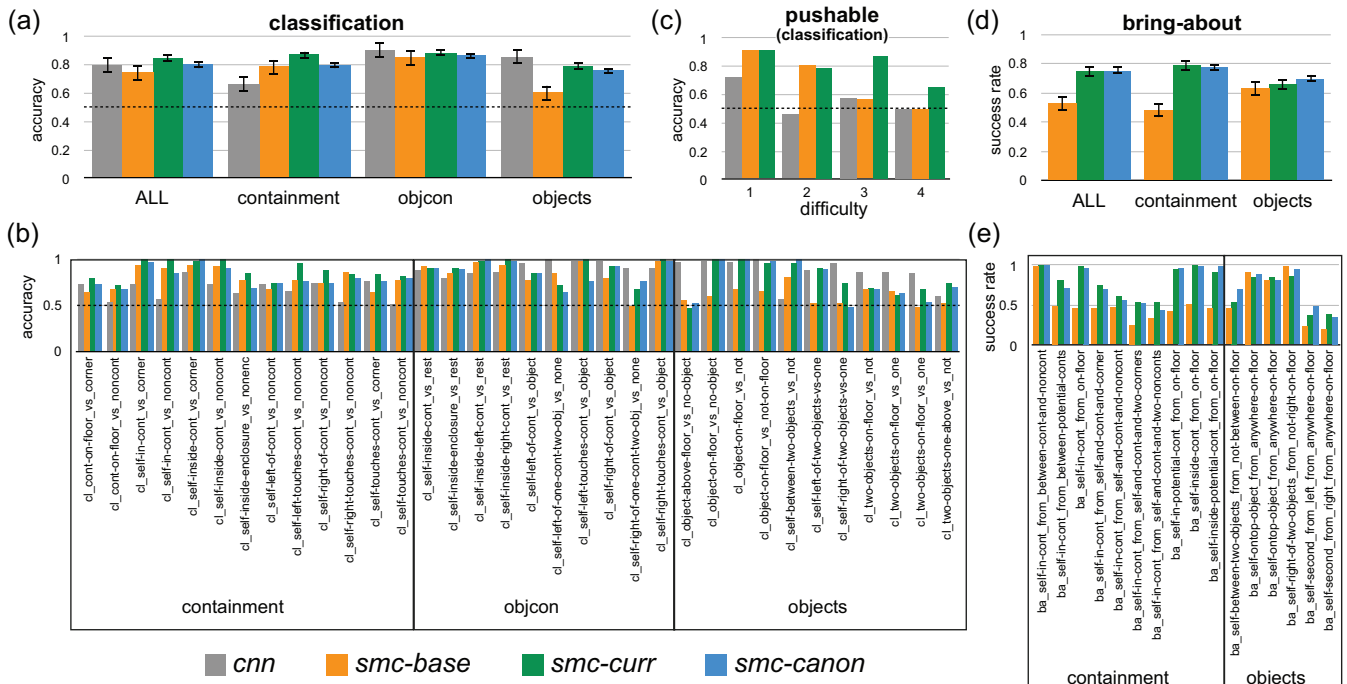
Figure 5: Quantitative results. (a & b) Performance on classification tasks, (a) averaged by category and (b) organized by category (each task is a box in the left column of Fig. 2). (c) Performance on *pushable* tasks involving classifying whether an object is pushable. Tasks are ordered according to difficulty, determined by the degree to which initial environment states allow classification without additional action. Fig. 3 shows corresponding task labels. (d & e) Performance on bring-about tasks, (d) averaged by category and (e) organized by category. Error bars are standard error.

in performance (Fig. 5 (d-e); *smc-base*: 52.7%, *smc-curr*: 74.5%, *smc-canon*: 75.0%; *smc-curr* vs. *smc-base*: $t(16) = 4.1$, $p = .0008$; *smc-canon* vs. *smc-base*: $t(16) = 4.4$, $p = .0005$). Again, the advantage is greatest for concepts that require complex interaction sequences, such as *ba_self-in-container_from_between-potential-containers* (visualized in Fig. 4 (b)) and other tasks involving containment.

Role of hierarchy. In order to verify that hierarchical reuse of SMCs is indeed taking place and responsible for improved performance, we conduct another series of experiments in which primitive actions are excluded from hierarchical SMCs. For both classification and bring-about concepts, performance is nearly identical whether including or excluding primitive actions (classification: *smc-curr*: 84.9% vs. 82.7%, *smc-canon*: 80.3% vs. 81.1%; bring-about: *smc-curr*: 74.5% vs. 71.6%, *smc-canon*: 75.1% vs. 75.1%).

6 Limitations

The approach in this paper has several limitations which present opportunities for future work:

1. The need to construct the large number of different environments that form the curriculum. Replacing these curricula at least in part by using intrinsic motivation (Barto and Singh 2004) is one natural avenue for future work.

2. In the present work, bring-about SMCs are only used in situations where it is possible to bring about the concept. An extension to cases where it cannot, allowing the SMC to signal whether it cannot achieve the concept, would increase the power of this representation.

3. PixelWorld, while capable of representing a wide variety of concepts, cannot match the range of concepts expressible in the real world. In light of this, extensions to robotic domains would be especially interesting.

7 Conclusions

We have introduced a representation of conceptual knowledge inspired by the notion of SMCs. In contrast to most prior work in concept learning, it assumes the perspective of an embodied agent interacting with the environment, and uses behavior itself as a basic unit of representation. In extensive experiments on PixelWorld, a novel conceptual learning and benchmark suite, we have demonstrated that conceptually meaningful behaviors can indeed be learned and successfully applied to test environments, given supervision in the form of specific and more canonical training curricula. This work opens a path to richer conceptual representations, with natural extensions to robotic domains with intrinsic motivation.

References

Agre, P. E., and Chapman, D. 1998. Pengi: An implementation of a theory of activity. In *AAAI*.

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Zitnick, C. L.; and Parikh, D. 2015. VQA: Visual Question Answering. In *ICCV*.

Barto, A. G., and Mahadevan, S. 2013. Recent advances in hierarchical reinforcement learning. *DEDS*.

Barto, A. G., and Singh, S. 2004. Intrinsically motivated learning of hierarchical collections of skills. In *NIPS*.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *JAIR*.

Bergström, N.; Ek, C. H.; Björkman, M.; and Kragic, D. 2011. Scene understanding through autonomous interactive perception. In *ICVS*.

Bohg, J.; Hausman, K.; Sankaran, B.; Brock, O.; Kragic, D.; Schaal, S.; and Sukhatme, G. 2016. Interactive Perception: Leveraging Action in Perception and Perception in Action. *arXiv*.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym.

Brooks, R. A. 1987. Intelligence without representation. *Artificial Intelligence*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv*.

Drescher, G. L. 1991. *Made-up minds: A constructivist approach to artificial intelligence*. MIT Press.

Duan, Y.; Chen, X.; Houthooft, R.; Schulman, J.; and Abbeel, P. 2016. Benchmarking deep reinforcement learning for continuous control. In *ICML*.

Ellis, K.; Solar-Lezama, A.; and Tenenbaum, J. B. 2015. Unsupervised learning by program synthesis. In *NIPS*.

Georgeon, O. L., and Ritter, F. E. 2012. An intrinsically-motivated schema mechanism to model and simulate emergent cognition. *Cognitive Systems Research*.

Gibson, J. J. 1977. The theory of affordances. *In Perceiving, Acting, and Knowing*.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AIstats*, volume 9, 249–256.

Grabner, H.; Gall, J.; and Van Gool, L. 2011. What makes a chair a chair? In *CVPR*.

Hester, T., and Stone, P. 2012. Intrinsically motivated model learning for a developing curious agent. In *ICDL*.

Hogman, V.; Bjorkman, M.; and Kragic, D. 2013. Interactive object classification using sensorimotor contingencies. *IROS*.

Jia, Y.; Abbott, J.; Austerweil, J.; Griffiths, T.; and Darrell, T. 2013. Visual concept learning: Combining machine vision and bayesian generalization on concept hierarchies. In *NIPS*.

Jonsson, A., and Barto, A. 2005. A causal approach to hierarchical decomposition of factored MDPs. In *ICML*.

Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

Kemp, C.; Tenenbaum, J. B.; Griffiths, T. L.; Yamada, T.; and Ueda, N. 2006. Learning systems of concepts with an infinite relational model. In *AAAI*.

Konidaris, G., and Barto, A. G. 2007. Building portable options: Skill transfer in reinforcement learning. *IJCAI*.

Konidaris, G. D., and Barto, A. G. 2009. Efficient skill learning using abstraction selection. In *IJCAI*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Laflaqui, A. 2016. Autonomous Grounding of Visual Field Experience through Sensorimotor Prediction. *arXiv*.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2016. Building machines that learn and think like people. *Arxiv*.

Lake, B. M.; Salakhutdinov, R.; and Tenenbaum, J. B. 2012. Concept learning as motor program induction: A large-scale empirical study. In *CogSci*.

Lake, B. M.; Salakhutdinov, R.; and Tenenbaum, J. B. 2016. Human-level concept learning through probabilistic program induction. *Science*.

Marthi, B.; Russell, S. J.; Latham, D.; and Guestrin, C. 2005. Concurrent hierarchical reinforcement learning. In *IJCAI*.

Maye, A., and Engel, A. K. 2011. A discrete computational model of sensorimotor contingencies for object perception and control of behavior. In *ICRA*.

Maye, A., and Engel, A. K. 2012. Time scales of sensorimotor contingencies. *BICS*.

Maye, A., and Engel, A. K. 2013. Extending sensorimotor contingency theory: prediction, planning, and action generation. *Adaptive Behavior*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv*.

Ng, A. Y.; Harada, D.; and Russell, S. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*.

Noë, A. 2015. Concept pluralism, direct perception, and the fragility of presence. *Open MIND*.

O'Regan, J., and Noë, A. 2001. A sensorimotor account of vision and visual consciousness. *BBS*.

O'Regan, J. 2011. *Why Red Doesn't Sound Like a Bell: Understanding the Feel of Consciousness*. Oxford Univ. Press.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *TKDE*.

Peters, J., and Schaal, S. 2008. Reinforcement learning of motor skills with policy gradients. *Neural networks*.

Philipona, D., and O'Regan, J. 2006. Color naming, unique hues, and hue cancellation predicted from singularities in reflection properties. *Visual Neuroscience*.

Philipona, D.; O'Regan, J.; and Nadal, J.-P. 2004. Perception of the structure of the physical world using unknown sensors and effectors. In *NIPS*.

Ring, M.; Schaul, T.; and Schmidhuber, J. 2011. The two-dimensional organization of behavior. In *ICDL'11*.

Russell, S. J., and Norvig, P. 2009. *Artificial Intelligence: A Modern Approach*. Pearson Education, 3 edition.

Sánchez-Fibla, M.; Duff, A.; and Verschure, P. F. M. J. 2011. The acquisition of intentionally indexed and object centered affordance gradients: A biomimetic controller and mobile robotics benchmark. *IROS*.

Schaul, T., and Ring, M. 2013. Better generalization with forecasts. *IJCAI*.

Schaul, T.; Horgan, D.; Gregor, K.; and Silver, D. 2015. Universal value function approximators. *JMLR*.

Schaul, T. 2013. PyVGDL: A video game description language in Python.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *ICML*.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the game of go with deep neural networks and tree search. *Nature*.

Simsek, Ö., and Barto, A. G. 2009. Skill characterization based on betweenness. In *NIPS*.

Sinapov, J.; Schenck, C.; Staley, K.; Sukhoy, V.; and Stoytchev, A. 2014. Grounding semantic categories in behavioral interactions: Experiments with 100 objects. In *RAS*.

Singh, S. P. 1992. Reinforcement learning with a hierarchy of abstract models. In *AAAI*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Sutton, R. S.; Modayil, J.; Delp, M.; Degris, T.; Pilarski, P. M.; White, A.; ; and Precup, D. 2011. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *AAMAS*.

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*.

Tenenbaum, J. B. 1999. Bayesian modeling of human concept learning. In *NIPS*.

Van Hoof, H.; Kroemer, O.; and Peters, J. 2013. Probabilistic interactive segmentation for anthropomorphic robots in cluttered environments. *Humanoids*.