Towards Robotic Minds: Dynamic Interpretation and Schemata Recombination

Stevan Tomic[1,*,†]

[1]*Center for Applied Autonomous Sensor Systems (AASS), Örebro University, Örebro, Sweden.*

Abstract

Humans excel at interpreting the world dynamically and forming simplifying abstractions. Moreover, depending on the context, humans can re-interpret one situation in terms of another, which allows us to assign different meanings to (same) physical objects. With this ability, we can re-use existing knowledge, understand the behaviors of others, use language with metaphors, and learn and reason in novel situations. These seem to be prevalent in humans in contrast to other animals or, at least, current computational systems. Thus, a cognitive-inspired computational framework would need to consider dynamic interpretation and abstraction as the main design principle. This work presents such a framework focusing on the problem of dynamic interpretation.

Keywords

dynamic interpretation, abstraction, schema, grounding, cognition, mental simulation, categorization, fast/slow system, context, reinforcement learning

1. Introduction

Approaches in AI and robotics are often directed toward solving concrete, specialized problems. For instance, impressive results are achieved in-game playing [1] or robot control [2]. However, they are not yet applicable to solving general-purpose tasks. On the other hand, many theories in Cognitive Science offer abstract views on how (general) cognition in humans might work. Still, their applicability to concrete problems seems troublesome as the subtle low-level details get lost in abstractions. For similar reasons, many novel cognitive architectures are hybrid, i.e., they contain both abstract, symbolic parts suitable for reasoning, and sub-symbolic, good at handling low-level specific details. However, these processes are functionally separated. That is, as noted in the recent review of cognitive architectures [3], some use sub-symbolic approaches for sensory processing while the rest of the representation and reasoning is symbolic. Others relate symbolically represented modules via sub-symbolic processes (e.g., activation spreading, reinforcement learning). Overall, it seems natural to introduce abstractions to sub-symbolic learning, and make them capable of reasoning and learning on an explicit strategic level. Related to this is a review of abstraction in the area of machine learning [4]. In addition, the author states that a precondition to general-purpose AI is to automatically construct an appropriate and problem-specific abstract representation of a new problem, hence reinforcing Brooks [5] belief

AIC 2022, 8th International Workshop on Artificial Intelligence and Cognition

tomic01@gmail.com (S. Tomic)
http://aass.oru.se (S. Tomic)
0000-0002-4379-6614 (S. Tomic)

that finding the right abstraction is essential to intelligence. In Cognitive Science, some areas like Grounded Cognition [6, 7, 8] argue that (abstract) knowledge is grounded in sensory-motor patterns and simulation theories. As robots face the dichotomy between symbolic reasoning and sub-symbolic perception and learning, they make ideal candidates for testing such theories. For example, Billing et al. [9] discuss different ways of realization of simulation theories and enable a robot to simulate its actions after learning them from demonstrations. Still, a major question in this area is, How to represent complex and abstract concepts that seems detached from sensory-motor patterns?

The work presented in this paper makes an effort toward merging abstract (symbolic) and sub-symbolic approaches in a cognitive process named *dynamic interpretation*. It brings (abstract) symbol manipulation in (sub-symbolic) reinforcement learning (RL) and formalizes the notion of abstraction in a context. Depending on which components of the framework are known, and which have to be computed, it identifies different computational or cognitive processes. These processes build on each other, where one of the core processes is *interpretation*. For example, as we will see, interpretation allows learning an abstract policy, *schema* that can be re-used across domains or simulated in a physical engine. Then, depending on whether an agent is familiar with a situation (knows how to interpret it) or does not, one can naturally distinguish between (automatic) associative retrieval, or deliberative reasoning process [10]. The focus of this work is a deliberative reasoning/learning system implemented as recombination of abstract schemata. The proposed computational approach is briefly discussed through the view of Cognitive Science in Section 6. As such it seems to underlay many other cognitive phenomena and brings about a unified view of different cognitive processes under the same framework. Simulating schemata in a mental space is believed by some authors to be the thinking process (e.g., [11]). Thus, the main contribution of this paper is bringing closer together ideas from the field of AI, robotics, and cognition via a common framework.

2. Representation of Domain, Grounding and Interpretation

The framework used in this paper is based on the previous work [12, 13]. This section briefly introduces the relevant building blocks of the framework.

2.1. Domain State Matrix

A *domain* consists of *entities* such as agents, behaviors and objects. A state of a domain is described by *state variables*, each defined by an *attribute* describing entities or their relations. An attribute can be a simple feature (sensor value) or a function of other attributes (e.g., ‘at’ is a function of ’position’). For example, the Boolean state variable $\mathsf{detected}(\mathsf{agent}_1, \mathsf{object}_1)$ consists of attribute ‘detected’ describing the relation between two domain entities (or a tuple): $\langle agent_1, object_1 \rangle$. Each of the m domain elements and their tuples in relation to each other, are represented as $1 \times n$ dimensional row vector, where n is the number of available attributes. Stacking these vectors, we obtain the domain matrix $D^{m \times n}$, refereed simply as D. Matrix D represents a state at a given time t. Given a time interval I, a tensor D_τ is constructed. It captures $D(t)$ for all t in interval I, hence, representing a bounded trajectory (τ, I). In this way, a trajectory, (τ, I), is decomposed along three dimensions: entities, attributes and time

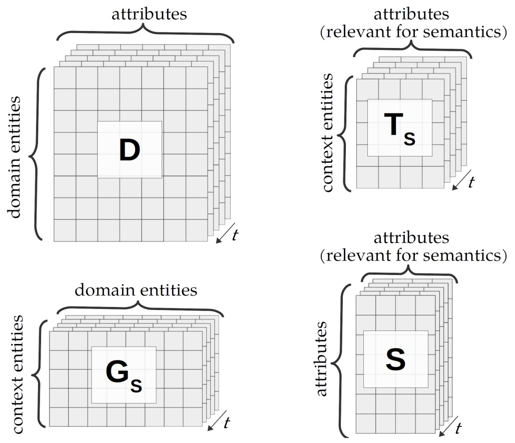

Figure 1: Matrices in the framework

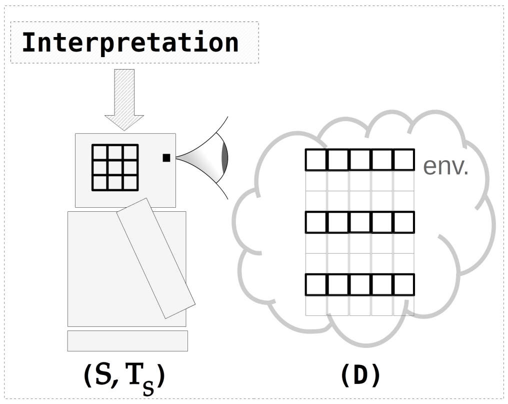

Figure 2: Illustration of interpretation

(see Figure 1). For example, the state variable $\mathsf{detected}(\mathsf{agent}_1, \mathsf{object}_1)$, would be placed in row i, representing tuple $\langle \mathsf{agent}_1, \mathsf{object}_1 \rangle$; and in column j, representing attribute detected; while the third index t will reflect the value of the variable in time $t \in I$. We assume discrete space and time.

2.2. Abstract Context

Context is an abstraction of a concrete situation that may arise in various physical domains in which robots operate. It may help to select and attend to relevant parts of a situation. In addition, context provides meaning to robotic actions, rendering them intentional, goal-oriented behavior. In the framework, abstraction is defined as a *context*, $\mathcal{C}$. It contains a set of *rules* or *relations*, relating relevant high-level categories, named *contextual entities*. Such entities are categories such as roles, actions, and artifacts (objects with special meaning in the context). For instance, in a context of a buying/selling scenario, a rule may state that a buyer should pay at a place dedicated to paying: $\mathsf{at}\,(\langle \mathsf{Buyer}, \mathsf{Pays}, \mathsf{PayingPlace} \rangle)$, where Buyer is a contextual entity, i.e., a category for agents that fits into certain behavior. Similarly, Paying is a category for behaviors associated with money transfer/exchange from one agent to another. Similarly, PayingPlace is a category for a particular position where paying takes place, e.g., the place of a cash-register or a seller.

2.3. Grounding And Semantics

Rules or relations like 'at' are understandable by humans as we already know its meaning. However, such relations are without explicit connection to the concrete physical meaning, hence they are too abstract for robots. To make them concrete, we link them to a physical domain through *grounding* and rules *semantics*.

Grounding. The grounding, $\mathcal{G}_{\mathcal{S}}$, is defined as a function that associates contextual entities in $\mathcal{C}$ with concrete entities in a domain $\mathcal{D}$. Then, for example, an execution of passing a piece of paper in a physical domain may be associated with contextual entities such as 'paying' and 'money' if it adheres to contextual constraints, that is, its relations. We represent grounding as a binary matrix $G_{\mathcal{S}}$ that captures such associations. Each entry in $g_{i,j}$, has a value 1 if high-level contextual entity (or their tuples) in a row i, is grounded to a (low-level) domain entity (or their tuples) in column j.

Semantics. Rules are defined as relations between contextual entities (e.g., $\mathsf{at}\left(\left(\mathsf{Buyer}, \mathsf{Pays}, \mathsf{PayingPlace}\right)\right)$), and in this abstract form, while specify syntax, they do not say much about concrete meaning on the level of an execution. Therefore rule's semantics is defined through a function $[\![\cdot]\!]$ that maps rules to trajectories. Then, for example, a rule $\mathsf{at}\left(\left(\mathsf{Buyer}, \mathsf{Pays}, \mathsf{PayingPlace}\right)\right)$ has semantics that specifies that a 'position' of a $Buyer$, when executing $Payment$, should be the same as the 'position' of $PayingPlace$.

$$
\begin{aligned}
&[\![\, \mathsf{at}\left(\left(Buyer, Pays, PayingPlace\right)\right)]\!] \equiv \\
&\{(I, \tau) \mid \forall t \in I, \mathsf{active}(Pays, Buyer)(\tau(t)) = \top \implies \\
&\ \mathsf{position}(Buyer)(\tau(t)) = \mathsf{position}(PayingPlace)(\tau(t))\}.
\end{aligned}
$$

Given a context $\mathcal{C}$, we denote by $\mathcal{S}_{\mathcal{C}}$ the set of all rule semantics for the rules in $\mathcal{C}$. Note that semantics is a function of grounding, hence concrete state-values can be obtained only after grounding. The context of an execution is therefore established by rules and the pair $\langle \mathcal{S}_{\mathcal{C}}, \mathcal{G}_{\mathcal{S}} \rangle$. Semantics and grounding together allow concretization of abstract rules on a level of physical execution (trajectories). The expressiveness of rules depends on two factors, namely, on the complexity of relations in norm semantics, and the number of available attributes.

Having the semantics we define a binary semantic indicator matrix S, whose rows correspond to all available attributes, and columns only to attributes used in the semantic functions (context-relevant attributes): entry $s_{i,j}$ has value 1 if attributes in the column j and the row i are identical, and 0 otherwise.

2.4. Interpretation Relation

Domain states, semantic indicators and grounding are all represented in a vector space via related matrices, D, S, and $G_{\mathcal{S}}$. What is still missing is matrix representations of a context, that is, the state of abstract contextual entities, like Buyer or Paying. This intuition is captured by the following relation:

$$G_{\mathcal{S}} D S = T_{\mathcal{S}} \tag{1}$$

Matrix $T_{\mathcal{S}}$ describes the state of context $\mathcal{C}$, and it can be understood as follows. The product between $G_{\mathcal{S}}$ and D selects domain entities (rows in D) that are grounded, which become row vectors in $T_{\mathcal{S}}$ as corresponding *contextual* entities. The product between D and S selects only attributes (columns in D) that are relevant for the context $\mathcal{C}$, that is, attributes used in semantics functions in $\mathcal{S}_{\mathcal{C}}$. In this way, $T_{\mathcal{S}}$ captures a state of context-relevant attributes of contextual entities, at time t, that is, it captures the state of a context (see Figure 2). Note also that values in $T_{\mathcal{S}}$ are selections of rows and columns already existing in D, hence another way to see this equality is as an selective attention mechanism, where the knowledge about context allows

Table 1
Relevant computation/cognitive problems.

Problems	Context	Grounding	Trajectory	Domain	Semantics
Verification	Given	Given	Given	Given	Given
Grounding	Given	?	–	Given	–
Interpretation	Given	?	Given	Given	Given
Abstract Learning/Planning	Given	Given	?	Given	Given
General Reasoning/Learning	?	?	?	Given	–

us to attend to the context-relevant part of D. In a similar way, the context-relevant part of a whole trajectory is represented as a tensor.

3. Computational (Cognitive) Problems

The goal of this section is to systematically classify problems relevant for this work and discuss their solutions. They are arranged in Table 1, where columns represent components in our framework, while rows specify problems depending on which component is given, irrelevant (marked with a dash) and which has to be computed (marked with a question-mark).

3.1. Verification, Grounding and Interpretation Problems

Verification. Since all components are known, this problem is to ensure that the given trajectory *adheres* to the given set of context's rules (or constraints), that is, it complies to all rules provided by $\mathcal{S}_\mathcal{C}$.

Grounding Problem. This is a 'simple' symbol grounding problem that does not take rules semantics into account, nor a trajectory. It could be addressed with associative or similarity based approaches for categorization as it does not require verification (see Section 6.1).

Interpretation Problem. This is another form of a grounding problem when the trajectory and semantics functions are known, hence have to be taken into consideration. For example, given an improvised football game, what is a role of a sweater on a meadow? Any grounding $G_\mathcal{S}$ for which a trajectory (τ, I) adheres (can be verified) to semantics functions $\mathcal{S}_\mathcal{C}$ is a solution to this problem, hence it has multiple solutions.

3.2. Abstract Learning and Reasoning Problems

Abstract Learning. This problem is concerned with generating a trajectory adherent to $\mathcal{S}_\mathcal{C}$. The problem can be seen as a planning problem, or as an abstract learning problem. The latter is concerned with *learning* a policy based on abstract (lifted) representation, that can generate concrete trajectories adherent to semantics specifications (for details please refer to Tomic et al [13]). As the dimensions of matrix $T_\mathcal{S}$ depend only on semantics $\mathcal{S}_\mathcal{C}$, we use the tuple

$\langle \mathcal{S}_{\mathcal{C}}, T_{\mathcal{S}} \rangle$ to learn such a policy. We use $T_{\mathcal{S}}$ to obtain a feature-state vector (input), and $\mathcal{S}_{\mathcal{C}}$ to check adherence to rules, and depending on it, generate a feedback (reward) signal. The abstract policy acts towards achieving (declarative) semantics $\mathcal{S}_{\mathcal{C}}$, with domain entities specified in grounding $\mathcal{G}_{\mathcal{S}}$, and we denote it by $\pi_{\mathcal{S}}(Acts \mid T_{\mathcal{S}})$, where $Acts$ are abstract (contextual) actions that can be grounded to concrete ones. An abstract policy can be seen as *schema* (see Section 6.3) (e.g. $at(agent, place)$ - a schema whose execution should result in moving grounded $agent$ to be at a grounded $place$).

General Reasoning/Learning Problem. This is the most general problem defined in the table. The problem assumes only the knowledge of parts of a domain D. In reinforcement learning, this problem typically assumes the presence of a final goal or a reward. Some authors argue that the generic objective of maximizing rewards is enough for learning general behaviors [14]. Still, the process of learning can benefit from learning the underlying structure of a task (context) in which rewards are obtained. Thus, in our approach to this problem, a novel (abstract) schema is learned (in a context of a goal) from local interactions of existing behaviors/schemata. We address this problem by *dynamically interpreting* schemata and recombining them towards learning a novel abstract policy that generates trajectories that achieve a given *grounded* goal (goal-adherent trajectories).

3.3. More About Interpretation Problem

The interpretation problem based only on verification from rules semantics has multiple solutions (see Section 3.1). As semantics functions describe a certain (temporal) pattern of values of attributes, the same patterns may occur in arbitrary entities/attributes that typically should not be related to a context. The question of correctness (verification) is related to the central question in the philosophy of computer science [15]: What does it mean for a physical machine to satisfy an abstract requirement. It is known that abstracting by simple mappings, comparable to our grounding, leads to pancomputationalism, a view where "every ordinary open system realizes every abstract finite automaton" [16, 17]. Different arguments to counter this view are suggested. They usually consist of adding requirements to further restrict possible interpretations, e.g., causal constraints or that a computational system must be associated with a semantic description. Maybe a view that is the most relevant to this work is the assumption that computation depends on interpretation (discussed in [18, 19]). When physical interactions are causal, consistent, and reproducible, then an observer (or interpreter) can easily (directly) map (ground) an abstract machine to a physical process or vice versa. Hence, whether a physical system implements a computation depends on how an observer interprets (maps) the system. An interpreter would already have to possess prior knowledge of the patterns it wants to recognize to verify their correctness. Hence the computational part would be already in the interpreter, which begs the question, How did that initial pattern emerge? The answer might lie in an interpreter's goals [1]. Provided a *concrete goal* the interpretation problem would be constrained by causal inference provided by a physical/simulation system since only causal sequences of actions would consistently lead to a goal state. An example that further demonstrate that

[1]The initial goal of an agent's self-preservation, naturally arise in the process of evolution.

interpretation depends on goal is as follows. When the goal is to keep a fire burning, we can interpret a newspaper as fuel (since it has a 'flammable' attribute), regardless of its typical use as an information artifact. Searching or learning interpretation (dynamic interpretation) towards achieving a concrete goal is the main topic in this work.

4. Dynamic Interpretation

This section introduces the dynamic interpretation of schemata and their simulation in which their interpretation depends on a given goal.

4.1. Schema Interpretation

In an abstract policy, semantics $\mathcal{S}_C$ define what kind of behavioral pattern is learned, while $\mathcal{G}_C$ defines involved domain entities. Then, an *interpretation* of an abstract policy (schema) is defined by a tuple $\langle \pi_{\mathcal{S}}, \mathcal{G}_{\mathcal{S}} \rangle$. For example, an interpretation of abstract policy $\langle \pi_{[\![at]\!]}, \mathcal{G}_{1[\![at]\!]} \rangle$, that acts toward achieving 'at' relation between, e.g., agent_1 and object_1 for $\mathcal{G}_{1[\![at]\!]}$, when interpreted with $\mathcal{G}_{2[\![at]\!]}$, with agent_1 and object_2, would act toward achieving the same 'at' relation but with new object_2. Thus, with different schema interpretations, an agent can achieve different concrete behaviors. This feature is used for learning interpretation and a sequence of abstract schemata so that a given goal is achieved.

4.2. Learning via Dynamic Interpretation/Grounding

Working Example. Imagine a robotic agent named Robby. He is soon to run out of battery power, hence his main goal is to obtain a pack of batteries located at a hardware store (see Figure 3). He does not know what 'store' is, nor which are the actions (or sequences) to obtain/buy an item like a battery. All that Robby knows is his grounded goal, how to verify it, and a set of low-level schemata in his procedural memory (e.g., $\pi_{[\![at]\!]}$) that he can interpret and simulate in his mental space – a physical engine.

4.2.1. Approach

In this approach, a goal is realized similarly to rules semantics (constraints or conditions over trajectories). For instance, it can specify that $\mathsf{has}(\mathsf{Buyer}, \mathsf{Item}) = \mathsf{true}$, where Buyer and Items are grounded, e.g., $G(Buyer) = Robby$, $G(Item) = Battery$. In this way, the existing verification and feedback (reward) mechanism are used [13]. Given a grounded goal, an agent learns a sequence of schemata interpretations, that is, choices of $\langle \pi_{\mathcal{S}}, \mathcal{G}_{\mathcal{S}} \rangle$ in I. Thus we can see how by choosing schemata and their interpretations (groundings) in different time intervals, we can create a higher level policy:

$$\pi_{\mathcal{S}_{\lambda+1}}(\langle \pi_{\mathcal{S}_\lambda}, \mathcal{G}_{\mathcal{S}_\lambda} \rangle \mid T_{S_\lambda}),$$

where λ in $\mathcal{S}_\lambda$ indicate the level of semantical abstraction.

$\pi_{\mathcal{S}_{\lambda+1}}$ is a policy that recombines (lower-level) schemata, however, note that it is not abstract. The groundings are learned for a specific task, and one does not have $G_{\mathcal{S}\lambda+1}$ or $S_{\lambda+1}$ needed

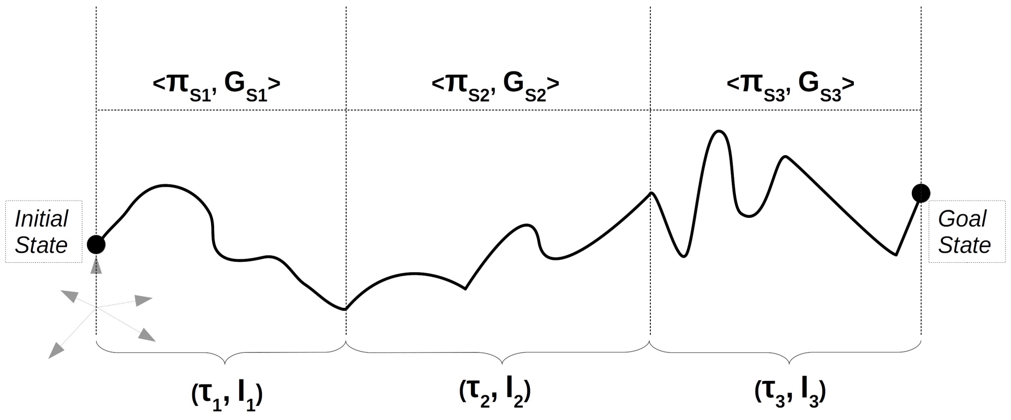

Figure 3: A sequence of interpretations of schemata leads to a goal-adherent trajectory.

for interpreting learned policy. To be abstract, its has to have a stable interpretable structure describing the overall context of a high level task, $T_{\mathcal{S}\lambda+1}$. Moreover, the size of input (feature) vector for $\pi_{\lambda+1}$ changes with different interpretations for each abstract policy which may have different dimensions ($T_{\mathcal{S}}$) for each low-level policy π_{λ}.

Making High-Level Policy Abstract. Still, this kind of learning explores the space of groundings (interpretations) to find exactly those groundings that solve the problem. As the higher-level policy, $\pi_{\lambda+1}$, uses only domain entities ground by each instantiation of lower-level policies, this means that the unified set of these groundings is $\mathcal{G}_{\mathcal{S}_{\lambda+1}}$. The matrix form $G_{\mathcal{S}\lambda+1}$ can be obtained directly by taking each learned $G_{\mathcal{S}\lambda}$ and concatenating all unique rows vertically into overall $\mathbf{G}_{\lambda+1}$. Similarly, as the approach learns exact low-level abstract policies, for which we know their S_{λ}, we then know all context-relevant attributes in the higher-level policy also. To create $S_{\lambda+1}$, then, it is enough to concatenate all unique columns horizontally from used lower-level policies S's). After these steps one would have $\mathbf{G}_{\lambda+1}$ and $\mathbf{S}_{\lambda+1}$, which can be applied to domains via equation (1), hence obtaining $\mathbf{T}_{\mathbf{S}\lambda+1}$ and providing the stable input vector for abstract policy $\pi_{\mathcal{S}_{\lambda+1}}$ that can be re-learned as abstract policy (interpratable schema).

In summary, the dynamic interpretation is a goal-driven approach that takes primitives or policies on level λ and creates representation and abstract policies on level $\lambda + 1$.

5. The Pilot Experiment

The goal of the presented experiment is to empirically prove the feasibility of the proposed approach. It focuses on learning interpretation and selection of schemata depending on a goal, as described in the previous section. The hypothesis is that learning algorithms will need fewer learning steps to learn to achieve a goal, since learning does not start from scratch, rather, it uses already abstracted previously learned policies (schemata) that execute in longer temporal intervals. The experiment is confined to learning a sequence of interpretations of

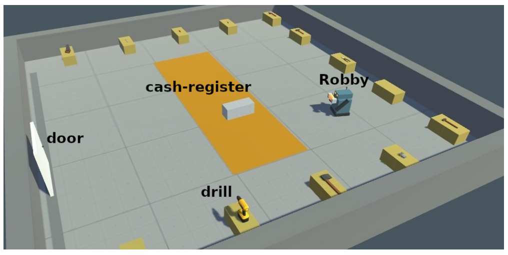

Figure 4: Learning Environment.

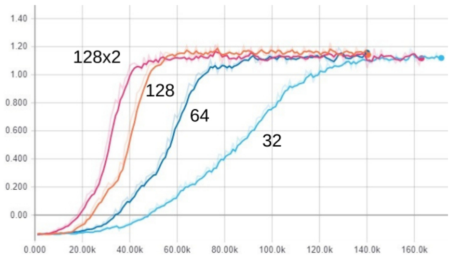

Figure 5: Results.

only one schema. This is done for simplicity, but also to stress the fact that by simply re-using (transferring) the same abstract schema, an agent can learn more complex behavior. The video of this pilot experiment is available at https://youtu.be/0C1YZ7Av2Eo.

Scenario. The scenario is the same as the working example in the previous section 4.2 (see Figure 4). The Robby's goal is specified as: $\mathsf{has}(\mathsf{buyer}, \mathsf{item}) = \mathsf{true}$, $\mathsf{executed}(\mathsf{buyer}, \mathsf{pay}) = \mathsf{true}$, $\mathsf{executed}(\mathsf{buyer}, \mathsf{exit}) = \mathsf{true}$, and grounded as follows: $\mathcal{G}(\mathsf{buyer}) = \mathsf{Robby}$, $\mathcal{G}(\mathsf{item}) = \mathsf{battery}$, $\mathcal{G}(\mathsf{pay}) = \mathsf{wtransfer}$, and $\mathcal{G}(\mathsf{exit}) = \mathsf{open}$. Robby has only one pre-trained abstract policy $\pi_{[\![execute\_at]\!]}$, which will make an agent navigate *at* grounded location and *execute* grounded action. Thus an RL algorithm has to find/learn a sequence of interpretations $\mathcal{G}$'s of one schema.

Hardware ans Software Setup. The simulations are executed on a desktop computer – Intel Core i7 4790K CPU @ 4GHz (x64), 16GB DDR3 RAM, Nvidia GTX970 (4GB GDDR5). Simulations are done in a game engine [20]. Reinforcement learning is done with PPO [21] (an artificial neural network based approach), with the help of the Machine Learning toolkit [22] (v0.6).

Trials. The simulation used a maximum of 2000 steps per learning episode, where the data is collected over 16 parallel simulations. Since abstract policies (schemata) are temporally extended actions, they need a certain time interval to execute. Thus instead of requesting the output of the RL algorithm at each simulation step, we get its output, groundings, every 200 simulation steps.

Results. Results are shown in Figure 5. It contains learning curves for models consisting of 32, 64, 128, and two layers of 128 neurons. Results are promising as they suggest that learning is one order of magnitude faster than learning the same behavior from scratch [13], which in most cases did not converge in 4M simulation steps. That confirms our hypothesis and, importantly, shows the feasibility of the dynamic interpretation approach.

6. Brief Connections to Cognitive Science

One of the fundamental questions in Cognitive Science is, How do humans organize and represent knowledge? The classical views on categorization assume a precise, rigid definition of a category as a set of necessary and sufficient features or attributes. This view has drawbacks, as humans often categorize based on typicality. For instance, a pigeon or robin seems more typical within the category of birds than an ostrich or a penguin. Cognitive science offers additional views on categorization and concepts representation. Roughly, views can be divided into, ones that categorize based on similarity comparison, and others based on rules-based representations.

6.1. Similarity Based Views

Popular categorization theories are based on *similarity* that measure how close a certain stimulus matches a category. Most popular are Prototype [23] and Exemplar [24] theories. The prototype theory suggests that a stimulus is compared with an ideal or prototypical example of a category. A prototype is a summary representation of a category, its central tendency of a category or an ideal example. Whether an entity (or experience) belongs to a category depends on how close it is to a category's prototype. The difficulty with finding an ideal/common prototype arises when category members have high variability in features. The exemplar view works by comparing the stimuli with a set of examples of previous instances (members of categories) stored in a memory. The stimuli are then classified based on the maximum number of matches it has with known members of a particular category. A 'heterogeneous hypothesis' [25] assumes that both views are used to form or classify a concept, and both approaches are implemented in cognitive architecture (Dual PECCS) [25], using conceptual spaces representation [26] where each quality of an object (attribute) is represented on a separate dimension. An entity is then represented as a point in multi-dimensional space where the distance between points is a similarity measure, hence providing a tool for computational realizations of similarity-based categorization. Other similarity-based approaches include unsupervised clustering and feed-forward artificial neural networks (e.g. classifiers). Classifiers are often based on convolution neural network architecture, and, to some extent, can also include temporal dimension (e.g, [27, 28]). In classifiers hidden layers are seen as a conceptual system [29].

Conceptual spaces are comparable to our representation of a domain state matrix (see Section 2.1). Similarity-based approaches could be used for associative retrieval to select candidate groundings and guide reasoning (dynamic interpretation). Still, categorization based on similarities does not include verification and to an extent, corresponds to a grounding problem defined in Table 1.

6.2. Theoretic-based View

Usually similarity-based approaches ignore the role of reasoning, goals, or context of an interpreter. Prinz [11], among others, states that much more knowledge is contained in concepts than prototype and exemplar theories had recognized [11]. This gives rise to a theoretic-based view on concepts [30]. In this view humans concepts are *"mini theories of the categories they represent"* [11]. Theoretic-based view does not look at categorization in isolation, rather they assume rules and relations between other categories in an overall structure. "To understand the concept of shoes, we need to understand the concept of legs and walking" [31]. Recently the 'heterogeneous hypothesis' is extended [32], providing an algorithm that decides when to use an exemplary, prototype, or theoretically based approach.

The role of theories is recognized in the presented computational framework. For example, a concept ‘store’ is defined over other categories mutually related with a set of rules (theories). Hence, a category of goods (merchandise) is hard to understand without dynamical (temporal) relations to categories of buyers, sellers, etc. In the framework, theories are represented as a context with grounded (declarative) semantics, $\langle \mathcal{S}_\mathcal{C}, \mathcal{G}_\mathcal{S} \rangle$, and the problem of categorization corresponds to grounding and interpretation problems/processes (see Section 3.1). An interesting question is, How can theories be represented in a sub-symbolic system (without declarative semantics $\mathcal{S}_\mathcal{C}$)? The answer might lie in schematic (procedural) mental simulation.

6.3. Schema

The term ‘schema’ was initially introduced in psychology by Bartlett [33]. Their work proposes that human knowledge is stored in underlying mental structures that represent a generic knowledge about the world. Since then, other terms are used to describe schema, such as: ‘frame’, or ‘script’. Schema is described as a pattern of behaviors that organize categories of information and relationships between them. Schemata are the foundation of numerous theories regarding modeling human cognition, concepts formation, language development and comprehension, culture, theories of mind, etc. Alba and Hasher [34] describe four important processes relevant to schemata: (1) *choosing incoming stimuli*, which can guide attention and focus only on relevant stimuli; (2) *abstraction*, which stores relevant patterns of interactions without the details and its original content; (3) *interpretation* of the new information by association to previously-stored knowledge; and (4) *integration* of those processes into a memory.

It can be argued that we have made a step towards achieving these processes in the presented computational framework, thus creating a schematic knowledge representation, $\langle \pi_\mathcal{S}, \mathcal{G}_\mathcal{S} \rangle$. Choosing incoming stimuli is done via interpretation or grounding, $\mathcal{G}_\mathcal{S}$. The knowledge stored in abstract policies $\pi_\mathcal{S}$ does not contain information about a particular domain, rather it stores information regarding patterns that relate to attributes of abstract categories. As such, it can be used to interpret information about new domain entities depending on previously-stored relations, by fitting them into corresponding categories. The presented framework is primarily used for social level interactions (social schemata), hence it is focused on representing normative rules [12]. The knowledge about other properties of the world is represented with other types of schemata. Some examples are self-schema, that is, knowledge about oneself based on past and grounded in present experiences; object-schema, knowledge about different categories of objects (entities), their function, structure; body schema, sensory-motor information of postures, etc. Importantly, they can still be seen as a relation between categories, hence interpretable and computable as is the case in the presented framework. Schemata are not static, they can change in time, and be simulated. Simulation is one of the main ideas behind the area of grounded cognition.

6.4. Grounded Cognition and Mental Simulations

The view in Grounded Cognition [35] states that conceptual representation is grounded in sensory-motor systems. As a category is learned from its member examples, only salient patterns of activation of sensory-motor (or introspective) features are stored in a memory.

Since stimuli can be a combination of different sensory inputs, this establishes a multi-modal representation of a category, called ‘perceptual symbol’. Depending on different factors (selective attention, existing knowledge, etc.), they can be retrieved (in working memory) and act as *simulators* by re-activating patterns of a similar set of features, a subset of those that were active during learning. Such process reenact experiences (causing mental imaginary), the process known as *mental simulation*. In this view a concept is not static, rather it is seen as a dynamic interpretation [36], or a ‘skill’ to construct representation depending on the constraints of a situation [7]. The construction of more complex concepts is done by recombining existing ones in mental simulation. As described by Prinz [11] the perceptual symbols stand as proxies for the category they represent, hence they are often called *proxytypes*. It is not clear what proxytypes exactly are (e.g., multi-modal representations, visual models, etc.). Still, it is known that a *context* determines which proxytypes are retrieved in working memory.

Views in this area seem to closely correspond to the presented approach of dynamic interpretation and recombination of schemata as a way to dynamically construct novel (action) schema. Interestingly, grounded cognition is often seen as a core to other cognitive phenomena. Some additional discussions are provided in Tomic et al. [37].

7. Conclusion

This work proposes a dynamic interpretation process that bridges the gap between high-level symbolic manipulation and low-level, sub-symbolic learning by searching in the space of groundings/interpretations and schemata. Interestingly, while the ideas on grounding, mental simulation and recombination come from cognitive science, we have reached complementary conclusions by investigating dynamic interpretation of abstract policies. This intersection of different paths toward complementary conclusions stresses the unifying role of interpretation and abstraction, and brings ideas in AI, robotics, and cognitive theories closer together in the pursuit of human-style intelligence.

8. Acknowledgments

This work has been supported by the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 761758 (Humane-AI-Net). I would like to thank my supervisors Alessandro Saffiotti and Federico Pecora for providing priceless feedback, crucial to my work.

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, Nature 550 (2017) 354.

[2] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al., Learning dexterous in-hand manipulation, arXiv preprint arXiv:1808.00177 (2018).

[3] I. Kotseruba, J. K. Tsotsos, 40 years of cognitive architectures: core cognitive abilities and practical applications, Artificial Intelligence Review 53 (2020) 17–94.
[4] G. Konidaris, On the necessity of abstraction, Current opinion in behavioral sciences 29 (2019) 1–7.
[5] R. A. Brooks, Intelligence without representation, Artificial intelligence 47 (1991) 139–159.
[6] L. W. Barsalou, Grounded cognition: Past, present, and future, Topics in cognitive science 2 (2010) 716–724.
[7] L. W. Barsalou, Abstraction in perceptual symbol systems, Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences 358 (2003) 1177–1187.
[8] G. Pezzulo, L. W. Barsalou, A. Cangelosi, M. H. Fischer, M. Spivey, K. McRae, The mechanics of embodiment: A dialog on embodiment and computational modeling, Frontiers in psychology 2 (2011) 5.
[9] E. A. Billing, H. Svensson, R. Lowe, T. Ziemke, Finding your way from the bed to the kitchen: Reenacting and recombining sensorimotor episodes learned from human demonstration, Frontiers in Robotics and AI 3 (2016) 9.
[10] D. Kahneman, Thinking, fast and slow, Macmillan, 2011.
[11] J. J. Prinz, Furnishing the mind: Concepts and their perceptual basis, MIT press, 2004.
[12] S. Tomic, F. Pecora, A. Saffiotti, Norms, institutions, and robots, arXiv preprint arXiv:1807.11456 (2018).
[13] S. Tomic, F. Pecora, A. Saffiotti, Learning normative behaviors through abstraction., in: ECAI, 2020, pp. 1547–1554.
[14] D. Silver, S. Singh, D. Precup, R. S. Sutton, Reward is enough, Artificial Intelligence (2021) 103535.
[15] N. Angius, G. Primiero, R. Turner, The Philosophy of Computer Science, in: E. N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, Spring 2021 ed., Metaphysics Research Lab, Stanford University, 2021.
[16] H. Putnam, Minds and machines, In Dimensions of mind 138 (1960).
[17] H. Putnam, Representation and reality, MIT press, 1988.
[18] J. Searle, The rediscovery of the mind. cambridge, massachusetts: Bradford, 1992.
[19] W. J. Rapaport, What is a computer? a survey, Minds and Machines 28 (2018) 385–426.
[20] U. Technologies, https://unity.com/, 2019.
[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347 (2017).
[22] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, D. Lange, Unity: A general platform for intelligent agents, arXiv preprint arXiv:1809.02627 (2018).
[23] E. H. Rosch, Natural categories, Cognitive psychology 4 (1973) 328–350.
[24] E. E. Margolis, S. E. Laurence, Concepts: Core Readings., The MIT Press, 1999.
[25] A. Lieto, D. P. Radicioni, V. Rho, Dual peccs: a cognitive system for conceptual representation and categorization, Journal of Experimental & Theoretical Artificial Intelligence 29 (2017) 433–452.
[26] P. Gardenfors, Conceptual spaces: The geometry of thought, MIT press, 2004.
[27] R. Vrskova, R. Hudec, P. Kamencay, P. Sykora, Human activity classification using the 3dcnn architecture, Applied Sciences 12 (2022) 931.
[28] M. Jung, J. Hwang, J. Tani, Self-organization of spatio-temporal hierarchy via learning of

dynamic visual image patterns on action sequences, PloS one 10 (2015) e0131214.

[29] L. Barsalou, Situated simulation in the human conceptual system, Language and cognitive processes 18 (2003) 513–562.

[30] S. A. Sloman, The empirical case for two systems of reasoning., Psychological bulletin 119 (1996) 3.

[31] W.-k. Ahn, C. C. Luhmann, Demystifying theory-based categorization, Building object categories in developmental time (2005) 277–300.

[32] A. Lieto, Heterogeneous proxytypes extended: Integrating theory-like representations and mechanisms with prototypes and exemplars, in: Biologically Inspired Cognitive Architectures Meeting, Springer, 2018, pp. 217–227.

[33] F. C. Bartlett, Remembering: A study in social and experimental psychology, 1932.

[34] J. W. Alba, L. Hasher, Is memory schematic?, Psychological Bulletin 93 (1983) 203.

[35] L. W. Barsalou, Grounded cognition, Annu. Rev. Psychol. 59 (2008) 617–645.

[36] Z. W. Pylyshyn, What the mind's eye tells the mind's brain: A critique of mental imagery., Psychological bulletin 80 (1973) 1.

[37] S. Tomic, F. Pecora, A. Saffiotti, Robby is not a robber (anymore): On the use of institutions for learning normative behavior, arXiv preprint arXiv:1908.02138 (2019).