Author version. KI 2018, LNAI 11117, pp. 182-189, 2018. The final authenticated publication is available online at https://doi.org/10.1007/978-3-030-00111-7_16

Towards Hypervector Representations for Learning and Planning with Schemas

Peer Neubert and Peter Protzel

Chemnitz University of Technology, 09126 Chemnitz, Germany
fistname.lastname@etit.tu-chemnitz.de

Abstract. The Schema Mechanism is a general learning and concept building framework initially created in the 1980s by Gary Drescher. It was inspired by the constructivist theory of early human cognitive development by Jean Piaget and shares interesting properties with human learning. Recently, Schema Networks were proposed. They combine ideas of the original Schema mechanism, Relational MDPs and planning based on Factor Graph optimization. Schema Networks demonstrated interesting properties for transfer learning, i.e. the ability of zero-shot transfer. However, there are several limitations of this approach. For example, although the Schema Network, in principle, works on an object-level, the original learning and inference algorithms use individual pixels as objects. Also, all types of entities have to share the same set of attributes and the neighborhood for each learned Schema has to be of the same size. In this paper, we discuss these and other limitations of Schema Networks and propose a novel representation based on hypervectors to address some of the limitations. Hypervectors are very high dimensional vectors (e.g. 2,048 dimensional) with useful statistical properties, including high representational capacity and robustness to noise. We present a system based on a Vector Symbolic Architecture (VSA) that uses hypervectors and carefully designed operators to create representations of arbitrary objects with varying number and type of attributes. These representations can be used to encode Schemas on this set of objects in arbitrary neighborhoods. The paper includes first results demonstrating the representational capacity and robustness to noise.

Keywords: Schema Mechanism · Hypervectors · Vector Symbolic Architectures · Transfer Learning.

1 Introduction

The idea to let machines learn like children, in contrast to manually programming all their functionalities, at least goes back to Turing 1946 [3]. Although a comprehensive picture of human learning is still missing, a lot of research has been done. A seminal work is the theory of cognitive development by Jean Piaget [21]. It describes stages and mechanisms that underly the development of children. Two basic concepts are assimilation and accommodation. The first

describes the process of fitting new information in existing schemas and the latter to adapt existing schemas or create new schemas based on novel experiences. Schemas can be though of as set of rules, mechanisms, or principles, that explain the behaviors of the world. In the 1980s, Gary Drescher developed the Schema Mechanism [6], a "general learning and concept-building mechanism intended to simulate aspects of Piagetian cognitive development during infancy" [6, p.2]. The Schema Mechanism is a set of computational algorithms to learn schemas of the form $< context, action, result >$ from observations.

Recently, Schema Networks were proposed [16]. They combine inspiration of the Schema Mechanism with concepts of Relational Markov Decision Processes and planning based on Factor Graph optimization. Schema Networks demonstrated promising results on transfer learning. In particular, to learn a set of schemas that resemble the underlying "physic" of a computer game and enable zero-shot transfer to modified versions of this game. Kansky et al. [16] demonstrated these capabilities on variations of the Arcade game Breakout. Previously, Mnih et al. [19] used end-to-end deep reinforcement learning to solve this and other Arcade games. In contrast to this subsymbolic end-to-end approach, Schema Networks operate on objects. However, the algorithms provided in the Schema Network paper require all objects to share the same set of attributes and all schemas to share neighborhoods of the same size. This restricts the application to domains with similar properties of all entities and regular neighborhoods. Thus, the experiments in [16] use again pixels as objects instead of more complex entities (like "brick" and "paddle" in the Breakout game).

In this paper, we present ongoing work on using hypervector representations and Vector Symbolic Architectures to relax the above conditions on the objects. In particular, we describe how objects can be represented as superposition of their attributes based on hypervector representations and how this can be used in a VSA to implement schemas. Similar approaches have previously been successfully applied to fast approximate inference [25] and mobile robot imitation learning [20]. We start with an introduction to the Schema Mechanism, Schema Networks and hyperdimensional computing, followed by a description of the proposed combination of these concepts and initial experimental results.

2 Introduction to the Schema Mechanism

The Schema Mechanism is a general learning and concept-building framework [6]. Schemas are constructed from observation of the world and interaction with the world. They are of the form $< context, action, result >$: Given a certain state of the world (the *context*), if a particular *action* would be performed, the probability of a certain change of the world state (the *result*) would be increased. A schema makes no predication in case of not fulfilled context. Schemas maintain auxiliary data including statistics about their reliability. According to Holmes and Isbell [12, p.1] they "are probabilistic units of cause and effect reminiscent of STRIPS operators" [7]. In the original Schema Mechanism, the state of the world is a set of binary items. Schema learning is based on *marginal attribution*, involving two steps: discovery and refinement [6]. In the discovery phase, statis-

tics on action-result combinations are used to create context-free schemas. In the refinement phase, context items are added to make the schema more reliable. An important capability of the original Schema Meachanism is to create synthetic item to model non-observable properties of the world [6].

Drescher [6] presented an implementation and results on perception and action planning of a simple simulated agent in a micro-world. Several extensions and applications of this original work have been proposed. For example, Chaput [4] proposed a neural implementation using hierarchies of Self Organizing Maps. This allows to learn schemas with a limited amount of resources. Holmes and Isbell [12] relaxed the condition of binary items and modified the original learning criteria to better handle POMDP domains. They also demonstrated the application to speech modeling. An extension to continuous domains was proposed by Guerin and Starkey [10]. Schemas provide both declarative and procedural meaning. Declarative meaning in form of expectations what happens next and procedural meaning as component in planning. The recently proposed Schema Networks [16] exploit both meanings.

3 Overview of Schema Networks

Schema Networks [16] are an approach to learn generative models from observation of sequential data and interaction with the environment. For action planning, these generative models are combined with Factor Graph optimization. Schema Networks work on entities with binary attributes. For learning, each training sample contains a set of entities with known attributes, a current action of the agent and a resulting state of the world in the next timestep (potentially including rewards). From these samples, a set of ungrounded schemas is learned using LP-relaxation. Ungrounded schemas are similar to templates in Relational MDPs [2,13]. During inference, they are instantiated to grounded schemas with the current data. For each attribute y, there is a set of ungrounded schemas W. The new value of y is computed from its neighborhood:

$$y = \overline{\overline{X}W}\mathbf{1} \tag{1}$$

W is a binary matrix. Each column is an ungrounded schema. X is a binary matrix where each row is the concatenation of attributes of entities in a local neighborhood and a binary encoding of the current action(s). The matrix multiplication in equation 1 corresponds to grounding of schemas. If any of the schemas in W is fulfilled, the attribute y is set. For action planning, a Factor Graph is constructed from the schemas. Optimization on this Factor Graph assigns values to variables for each relevant attribute of each relevant entity, the actions and the expected rewards at each timestep in the planning horizon. For more details on this simplified version of schemas, please refer to [16].

Schema Networks showed promising results on learning Arcade games and applying the learned generative model to modified game versions without retraining (zero-shot transfer). However, the description in the paper [16] is rather coarse and not self-contained. Moreover, there are also several theoretical limitations: The perception side is assumed to be solved. Schema Networks work on

entities and attributes, not on raw pixel data. In particular, the types of entities and their attributes have to be known in advance and have very large influence on the overall system. The schema learning approach can not deal with stochastic environments, i.e. contradicting (or noisy) observations are not allowed. All items have to be binary. Moreover, all entities have to share the same set of attributes and the neighborhood of all schemas has to be of the same size. This is a consequence of the matrix representation in equation 1. Section 5 presents an approach to use hypervector-based VSAs to address these latter two limitations.

4 Properties and Applications of Hypervectors and VSAs

Hypervectors are high dimensional representations (e.g. 2,048 dimensional) with large representational capacity and high robustness to noise, particularly in case of whitened encodings [14,1]. With increasing number of dimensions, the probability of sampling similar vectors by chance deceases rapidly. If the number of dimensions is high enough, randomly sampled vectors are expected to be almost orthogonal. This is exploited in a special type of algorithmic systems: Vector Symbolic Architectures (VSA) [18]. A VSA combines a high dimensional vector space $\mathbb{X}$ with (at least) two binary operators with particular properties: bind $\otimes$ and bundle $\oplus$, both are of the form: $\mathbb{X} \times \mathbb{X} \rightarrow \mathbb{X}$. bind $\otimes$ is an associative operator which is self-inverse, this is $\forall x \in \mathbb{X} : x \otimes x = I$ with I being the identity element. For example in a binary vector space, binding can be implemented by an elementwise XOR. Binding two vectors results in a vector that is not similar to both of the input vectors. However, the results of binding two vectors to the same third vector preserves their distance. In contrast, applying the second bundle $\oplus$ operator creates a result vector that is similar to both input vectors. For more details on these operations, please refer to [15,8,22].

Hypervectors and VSAs have been applied to various tasks. VSA can implement concepts like role-filler pairs [24] and model high-level cognitive concepts [9]. This has been used to model [17] and learn [20] reactive robot behaviors. Hypervectors and VSAs have also been used to model memory [5], aspects of the human neocortex [11], and approximate inference [25]. An interesting property of VSAs is that all entities (e.g. a program, a variable, a role) are of the same form, a hypervector, independent of their complexity - a property that we want to exploit for representation in schemas in the next section.

5 Combining Hypervectors and Schemas

This section describes an approach to represent context, action and result of a schema based on hypervectors and VSA operators. The goal is to provide a representation for the context that allows to combine objects with varying number and types of attributes and neighborhoods of varying size. The approach is inspired by Predication-based Semantic Indexing (PSI) [25] a VSA-based system for fast and robust approximate inference and our previous work on encoding robot behavior using hypervectors [20].

We propose to represent a schema in form of a single condition hypervector and a corresponding result hypervector. The condition hypervector encodes the

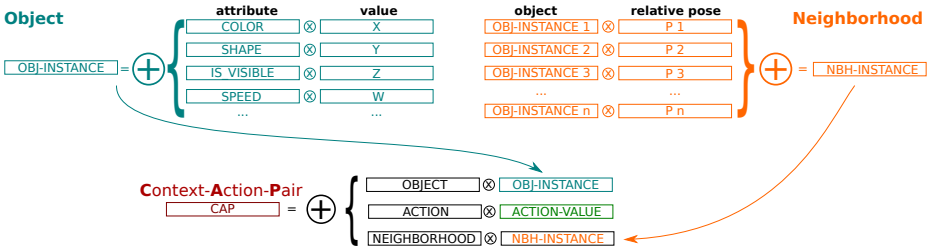

Fig. 1. Hypervector encoding of Context-Action-Pairs (all rectangles are hypervectors).

context-action-pair (CAP) of the schema. To test whether a known schema is applicable for the current context and action, the similarity of the current CAP and the schema's CAP can be used. Fig. 1 illustrates the encoding of arbitrary sets of attributes of objects and arbitrary neighborhoods in a single hypervector. We assume that hypervector encoders for basic datatypes like scalars are given (cf. [23]). Objects are encoded as "sum" of their attributes using the VSA bundle operator similar to the PSI system [25]. The more attributes two objects share, the more similar are their hypervector representations. Each attribute is encoded using a role-filler pair. One hypervector is used to represent the type (role) of the attribute and a second (the filler) to encode its value. Filler hypervectors can encode arbitrary datatypes, in particular, it can also be a hypervector representation of an object. The binding of the role and filler hypervectors results again in a hypervector of the same dimensionality. The bundle of all object properties is the hypervector representation of the object. The shape of the representation is independent of the number and complexity of the combined attributes.

Neighborhoods are encoded similarly by encoding the involved objects and binding them to their relative position to the regarded object. Let us consider the very simple example of a 3×3 neighborhood in an image. In a hypervector representation of this neighborhood, there are 8 objects surrounding a central object, each object is bound to a pose (i.e., top, top-right, ...) and the 8 resulting hypervectors are bundled to a single hypervector. In contrast to the matrix encoding in Schema Networks, the hypervector encoding allows to bundle an arbitrary number of neighbors at arbitrary poses (e.g. at the opposite side of the image). This is due to the fact that the shape of the hypervector bundle is independent of the number of bundled hypervectors (in contrast to the concatenation of the neighbors in Schema Networks) and the explicit encoding of the pose. Thus we can use an individually shaped neighborhood for each schema.

The creation of the CAP is illustrated at the bottom of Fig. 1: object-, action- and neighborhood-hypervector representations are bundled to a single CAP hypervector. Each of the representations is created by binding the filler encoding to the corresponding role (e.g. filler "OBJ-INSTANCE" to role "OBJECT").

6 Results

The initial goal to allow different attributes in objects and different neighborhoods for schemas is already fulfilled by design. In noiseless environments, recall

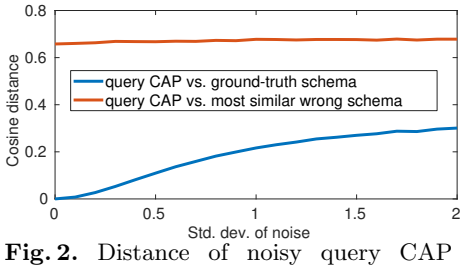

Fig. 2. Distance of noisy query CAP to schema CAPs (averaged over 1000 queries).

Vectorspace	$[-1, 1]^d$
#dimensions d	2,048
VSA bind	elementwise product
VSA bundle	elementwise sum (limited to $[-1, 1]$)
Metric	cosine similarity

Fig. 3. Hypervector and VSA parameters used for experiments. For details on the implementation refer to [20].

of a schema based on the similarity of CAP representations is inherently ensured as well (this can also be seen in the later explained Fig. 2 at noise 0). What about finding correct schemas in case of noisy object attributes? We want to demonstrate the robustness of the presented system to noise in the input data. The attributes of the objects that should toggle applicability of schemas are hidden rather deeply in the created CAPs. For application in real world scenarios, a known schema should be applicable to slightly noise-affected observations. If the derivation of the attributes is too large, the schema should become inapplicable. In the presented system, this should manifest in a equivariant relation of change in the input data and the similarity of the resulting CAP to the known schema.

For a preliminary evaluation of this property, we simulate an environment with 5,000 randomly created objects. Each object has 1-30 attributes randomly selected from a set of 100 different attribute types (e.g. color, shape, is-palpable, ...). All attribute values are chosen randomly. There are 1,000 a priori known schemas. Each is composed of one of the above objects, one out of 50 randomly chosen actions, a neighborhood of 1-20 other randomly chosen objects, and a randomly chosen result. All random distributions are uniform distributions. These are ad-hoc choices, the results are alike for a wide range of parameters. The properties of the used VSA are provided in Fig. 3. Fig. 2 shows the influence of noise on the encoding of the object's attributes on the similarity to the original schema. Noise is induced by adding random samples of a zero-mean Gaussian, drawn independently for each dimension of the hypervector encoding of the object's attribute value encodings. The standard deviation of the noise is varied as shown in Fig. 2. It can be seen that the distance of the noise-affected CAP to the ground-truth schema smoothly increases as desired, although the varied object attribute is deeply embedded in the CAP. The noisier the object attributes are, the less applicable becomes the schema. For comparison, the red curve shows the distance to the most similar wrong schema.

7 Conclusion

We presented a concept to use hypervectors and VSAs for encoding of schemas. This allows to address some limitations of the recently presented Schema Networks. We presented preliminary results on recall of schemas in noisy environments. This is work in progress, there are many open questions. The next steps towards a practical demonstration will in particular address the hypervector encoding of real data and action planning based on the hypervector schemas.

References

1. Ahmad, S., Hawkins, J.: Properties of sparse distributed representations and their application to hierarchical temporal memory. CoRR **abs/1503.07469** (2015), http://arxiv.org/abs/1503.07469
2. Boutilier, C., Reiter, R., Price, B.: Symbolic dynamic programming for first-order mdps. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 1. pp. 690–697. IJCAI'01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001), http://dl.acm.org/citation.cfm?id=1642090.1642184
3. Carpenter, B.E., Doran, R.W. (eds.): A. M. Turing's ACE Report of 1946 and Other Papers. Massachusetts Institute of Technology, Cambridge, MA, USA (1986)
4. Chaput, H.: The Constructivist Learning Architecture: A Model of Cognitive Development for Robust Autonomous Robots. Ph.D. thesis, Computer Science Department, University of Texas at Austin (2004)
5. Danihelka, I., Wayne, G., Uria, B., Kalchbrenner, N., Graves, A.: Associative long short-term memory. CoRR **abs/1602.03032** (2016), http://arxiv.org/abs/1602.03032
6. Drescher, G.: Made-up minds : a constructivist approach to artificial intelligence. Ph.D. thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science. (1989), http://hdl.handle.net/1721.1/77702
7. Fikes, R.E., Nilsson, N.J.: Strips: A new approach to the application of theorem proving to problem solving. Artificial Intelligence **2**(3), 189 – 208 (1971). https://doi.org/https://doi.org/10.1016/0004-3702(71)90010-5, http://www.sciencedirect.com/science/article/pii/0004370271900105
8. Gayler, R.W.: Multiplicative binding, representation operators, and analogy. In: Advances in analogy research: Integr. of theory and data from the cogn., comp., and neural sciences. Bulgaria (1998)
9. Gayler, R.W.: Vector symbolic architectures answer jackendoff's challenges for cognitive neuroscience. In: Proc. of ICCS/ASCS Int. Conf. on Cognitive Science. pp. 133–138. Sydney, Australia (2003)
10. Guerin, F., Starkey, A.: Applying the schema mechanism in continuous domains. In: Proceedings of the Ninth International Conference on Epigenetic Robotics. pp. 57–64. No. - in Lund University Cognitive Studies, Kognitionsforskning, Lunds universitet (2009)
11. Hawkins, J., Ahmad, S.: Why neurons have thousands of synapses, a theory of sequence memory in neocortex. Frontiers in Neural Circuits **10**, 23 (2016). https://doi.org/10.3389/fncir.2016.00023, https://www.frontiersin.org/article/10.3389/fncir.2016.00023
12. Holmes, M.P., Jr., C.L.I.: Schema learning: Experience-based construction of predictive action models. In: NIPS. pp. 585–592 (2004)
13. Joshi, S., Khardon, R., Tadepalli, P., Fern, A., Raghavan, A.: Relational Markov Decision Processes: Promise and prospects. In: AAAI Workshop: Statistical Relational Artificial Intelligence. AAAI Workshops, vol. WS-13-16. AAAI (2013)
14. Kanerva, P.: Fully distributed representation. In: Proc. of Real World Computing Symposium. pp. 358–365. Tokyo, Japan (1997)
15. Kanerva, P.: Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. Cognitive Computation **1**(2), 139–159 (2009)

16. Kansky, K., Silver, T., Mély, D.A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, D.S., George, D.: Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In: ICML. Proceedings of Machine Learning Research, vol. 70, pp. 1809–1818. PMLR (2017)
17. Levy, S.D., Bajracharya, S., Gayler, R.W.: Learning behavior hierarchies via high-dimensional sensor projection. In: Proc. of AAAI Conference on Learning Rich Representations from Low-Level Sensors. pp. 25–27. AAAIWS'13-12 (2013)
18. Levy, S.D., Gayler, R.: Vector symbolic architectures: A new building material for artificial general intelligence. In: Proc. of Conference on Artificial General Intelligence. pp. 414–418. IOS Press, Amsterdam, The Netherlands (2008)
19. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (Feb 2015), `http://dx.doi.org/10.1038/nature14236`
20. Neubert, P., Schubert, S., Protzel, P.: Learning vector symbolic architectures for reactive robot behaviours. In: Proc. of Intl. Conf. on Intelligent Robots and Systems (IROS) Workshop on Machine Learning Methods for High-Level Cognitive Capabilities in Robotics (2016)
21. Piaget, J.: The origins of intelligence in children. London: Routledge & Kegan Paul (1936), (French version published in 1936, translation by Margaret Cook published 1952)
22. Plate, T.A.: Distributed Representations and Nested Compositional Structure. Ph.D. thesis, Toronto, Ont., Canada, Canada (1994)
23. Purdy, S.: Encoding data for htm systems. CoRR **abs/1602.05925** (2016)
24. Smolensky, P.: Tensor product variable binding and the representation of symbolic structures in connectionist systems. Artif. Intell. **46**(1-2), 159–216 (Nov 1990)
25. Widdows, D., Trevor, C.: Reasoning with vectors: A continuous model for fast robust inference. Logic journal of the IGPL / Interest Group in Pure and Applied Logics (2), 141173 (2015)