A Boxology of Design Patterns for Hybrid Learning and Reasoning Systems

Frank van Harmelen and Annette ten Teije

Department of Computer Science, Vrije Universiteit Amsterdam, Netherlands
E-mail: frank.van.harmelen@vu.nl; annette.ten.teije@vu.nl

Received 06 January 2019;
Accepted 01 March 2019

Abstract

We propose a set of compositional design patterns to describe a large variety of systems that combine statistical techniques from machine learning with symbolic techniques from knowledge representation. As in other areas of computer science (knowledge engineering, software engineering, ontology engineering, process mining and others), such design patterns help to systematize the literature, clarify which combinations of techniques serve which purposes, and encourage re-use of software components. We have validated our set of compositional design patterns against a large body of recent literature.

Keywords: Hybrid systems, neurosymbolic systems, knowledge representation, machine learning, design patters.

1 Motivation

Recent years have seen a strong increase in interest in combining Machine Learning methods with Knowledge Representation methods.

The interest in this is fuelled by the complementary functionalities of both types of methods, and by their complementary strengths and weaknesses. This is witnessed by keynote addresses at all the major conferences in recent years (Hector Geffner and Josh Tenenbaum at ECAI/IJCAI 2018 [23, 33], Lise Getoor at NIPS 2017 [25], and William Cohen at ILP 2018 [55] to name just a few in the last two years), and by publications that have attracted widespread attention such as [6] and [34]. The workshop series on Neural-Symbolic Learning and Reasoning stretches back to over a decade[1] but has seen a sharp rise in interest in recent years, and many of the major conferences have dedicated workshops to the topic, such as the MAKE workshop at the AAAI Spring Symposium 2018[2], the Induce and Deduce workshop[3] and the workshop on Hybrid Reasoning and Learning[4] at KR 2018, the workshop on Relational Representation Learning[5] at NIPS 2018 and many others.

This increasing interest has resulted in an explosion of a large volume of diverse papers in a variety of venues, and from a variety of communities (of course from machine learning and knowledge representation, but also from semantic web, from natural language, from cognitive science, etc). Both this volume and this diversity of origin has created a very diffuse literature on the subject, with no consensus of which approaches are promising, using very different formalisms (ranging from graph theory to linear algebra and continuous differentiable functions), different architectures, different algorithms, often even different vocabularies to speak about the same concepts depending on the community of origin, and spread out over a large space of journals and conference, typically not surveyed by any single researcher.

This paper is an attempt to create some structure in this large, diverse and rapidly growing literature. We do not think it is possible anymore

[1]http://neural-symbolic.org/

[2]https://www.aaai-make.info/

[3]https://sites.google.com/view/r2k2018/home

[4]https://www.hybrid-reasoning.org/kr2018_ws/

[5]https://r2learning.github.io/

to provide an exhaustive overview of the entire literature. The literature is simply too voluminous, heterogeneous and spread out for such an exhaustive overview. Instead, what we aim for in this paper is to present a conceptual framework that can be used to categorize much if not all of the techniques for combining learning and reasoning. Our framework takes the form of a set of design patterns, which we will motivate in the next section. Our claim is *not* the that we cite and discuss the complete relevant literature on this topic (as stated, we think this would be impossible by now). Instead, our completeness claim is that our set of patterns covers all design variations that appear in the literature. Thus, referring to an additional paper would not by itself be an extension to this work, but would only be an extension to this work if such a paper describes a hybrid learning-and-reasoning architecture not yet covered by our set of patterns.

We have validated our set of design patterns against a set of more than 50 papers from the research literature from the last decade. Our claim is that each of the systems that we encountered in those references is captured by one of our design patterns.

In the next section we will discuss the basic distinction between learning systems and reason systems. In Section 3 we introduce our graphical notation, before the main Section 4 where we present our library of compositional patterns. Section 5 discusses future work and Section 6 concludes.

2 The Two Families of Techniques

Although not universal, there is some consensus in the literature as to the need for combining methods for learning (which are most predominantly statistical in nature) with methods for reasoning (which are predominantly discrete in nature): “Our general conclusion is that human-level AI cannot emerge solely from model-blind learning machines; it requires the symbiotic collaboration of data and models” [38]; “By pushing beyond perceptual classification and into a broader integration of inference and knowledge, artificial intelligence will advance greatly.” [34] ; “the question is not whether it is functions or models but how to profoundly integrate and fuse functions with models"[6] [15].

Other papers extensively discuss the advantages and disadvantages of both types of approaches in depth, but we briefly summarize the main points here, citing from [34] and the introductory section of [22]:

Limitations of (deep) learning systems:

- **Data hungry:** Learning systems (and in particular modern deep learning systems) need very large training sets;
- **Limited transfer:** a trained network that performs well on one task often performs very poorly on a new task, even if the new task is very similar to the one it was originally trained on;
- **Brittle:** they are susceptible to adversarial attacks, meaning that even for seemingly similar inputs for the same task, the outputs may differ significantly;
- **Opaque:** and they are opaque, meaning that is typically difficult to extract a humanly-comprehensible chain of reasons for output of the system.
- **No use of prior knowledge:** the performance of learning systems is based on the data they see during their training phase, and is not informed by general principles such as causality, or general domain knowledge.

Limitations of symbolic reasoning systems:

- **Brittle:** their foundation in discrete formalisms makes it hard to capture exceptions, and make these systems very unstable in the presence of noisy data.
- **Size:** acquiring explicit knowledge-bases (typically from experts) is error-prone and expensive, typically limiting the scope of such systems
- **Efficiency:** the logic-based reasoning methods are typically subject to combinatorial explosions that limit both the number of axioms, the number of individuals and relations described by these axioms, and the depth of reasoning that is possible.

[6]See our discussion for an explanation of this terminology.

Many authors in the literature have noted that these two sets of limitations are strongly complementary, and that furthermore the two families of techniques have seen successes in very different application scenario's, with statistical learning techniques successful in pattern recognition (e.g. image interpretation, speech recognition, natural language translation, board games and video games), while the successes of symbolic reasoning techniques are in such applications as planning (e.g. in robotics), diagnosis, design tasks and question answering (e.g. in personal assistants). This begs the question of the more precise delineation of these two families of systems.

Task-based distinction

A first characterization found in the literature is based on the *task* performed by the system:

Deduction vs. Induction: the classical distinction between reasoning and learning is the late 19th century Peirce-ian distinction between deduction and induction. Deduction derives specific conclusions from general statements, where conversely induction derives general statements from specific observations. More formally, deduction is the derivation of specific conclusions ϕ given a set of general formulae *T* and a deductive calculus $\vdash$: $T \vdash \phi$. Conversely, induction is the derivation of a set of general formulae *T* given a set of specific observations $\phi_1, \ldots, \phi_n$ such that $T \vdash \phi_i$ for all $i = 1, \ldots, n$: deduction is the problem of deriving ϕ given *T*, while induction is the problem of deriving *T* given $\phi_1, \ldots, \phi_n$.[7]

Compression vs. decompression: a more recent characterization is that of learning as compression [13]. The intuition here is that an inductive learning process "compresses" a large set of observations $\phi_1, \ldots, \phi_n$ into a more compact model *T*, using the Minimum Description Length principle [5] as a complexity measure for both data and model. Conversely, reasoning is then the process of producing (other) predictions ϕ from such a model *T*, which can be seen as a form of

[7]Confusingly, in logic *T* is called a "theory", while in machine learning, *T* would be called the "model", whereas the term "model" is used in a very different sense in logic.

decompression: after all, all the conclusions ϕ were already "implicitly" present in *T*, and the job of deduction is to simply "decompress" the general theory T into the more specific conclusions ϕ.

Both of these characterizations impose a strict dichotomy between the two modes of learning and reasoning. Other authors have instead tried to conceptualize a continuum of options that interpolate between induction and deduction (e.g. [9]), but these have not been widely adopted.

Representation-based distinction

Whereas the above dichotomies tried to capture the different *tasks* that are performed by reasoning or learning systems, another popular and somewhat orthogonal distinction in the literature is based on the *representation* that is used by different systems.

Pearl [38] uses the term "model-free" for the representations typically used in many learning systems, and Darwiche [15] described them as "function-based"[8], to emphasize that the main task performed by deep learning systems is function-fitting: fitting data by a complex function defined by a neural network architecture. Such "function-based" or "model-free" representations are in contrast to the "model-based" representations typically used in reasoning systems.

There is no consensus in the literature what precisely constitutes such a "model-based" representation, but typical properties that are ascribed to such model-based representations are that they are

- **compositional:** the meaning of model is a function of the meaning of its components,
- **referential:** the model is constructed out of symbols that refer to objects and relations in the world
- **homologous:** the structure of the model mirrors the structure of the world it is modelling,
- **interpretable:** the structure and content of a model is human-understandable and traceable,
- **symbolic:** as opposed to numeric,
- **discrete:** as opposed to real-valued, continuous and differentiable.

[8]Other names used for inferences at this layer are: "model-blind," "black-box," or "data-centric" [38]

Examples of such models are of course logical formalisms (such as propositional, first-order, modal and non-monotonic logics [26]), but also grammars, knowledge graphs [54], ontologies [47], graphical models [30], models from qualitative physics [48], etc.

Notice that the distinction of "model-based" vs. "model-free" (or: "function-based") representations is orthogonal to the previously discussed task-based dichotomy: systems such as Markov Logic Networks [41] do perform a learning task (in order to learn weights on the relations between variables from data), but do so based on an explicit (graphical) model. Similarly, Inductive Logic Programming (ILP [29]) performs a learning task, but again does so on an explicit model (and even on a model (Horn Clauses) originally intended for deductive reasoning). In the words of Darwiche [15]: *"Machine learning [...] has a wide enough span that it overlaps with the model-based approach; for example, one can learn the parameters and structure of a model but may still need non-trivial reasoning to obtain answers from the learned model"*. Nevertheless, it would be fair to say that the vast majority of modern work on machine learning (and in particular work on deep learning) uses a "model-free" (or: "function-based") representation of data.

We will use both of the above distinctions (task-based and representation based) in the design patterns that we will introduce next.

3 Design Patterns and Notation

The notion of re-usable design patterns has been successfully used in many different areas of Computer Science. Perhaps the best known of these are the Design Patterns from Software Engineering [19, 20]. These Design patterns have successfully captured general reusable solutions to commonly occurring problems in software design in the form of a template for how to solve a problem, that can be used in many different situations. These design patterns are organized in a hierarchical taxonomy, and typically expressed in a graphical notation[9]. Around

[9]sometimes described tongue-in-cheek as a *boxology*: "A representation of an organized structure as a graph of labelled nodes and connections between them", https://www. definitions.net/definition/boxology

the same time, a similar set of design patterns were developed for Knowledge Engineering in the form of the CommonKADS task library [45, 46]. In a similar vein to the patterns from Software Engineering, the CommonKADS library identified frequently occurring problem templates (called “tasks”), together with known templates for solving these problems (called “inference structures”), these templates were again organized in a hierarchical taxonomy, and expressed in a UML-like graphical notation. Other examples of Computer Science subdisciplines which have developed such design patterns are ontology design [21] and process mining [7].

Broadly recognized advantages of such design patterns are they distill previous experience in a reusable form for future design activities, they encourage re-use of code, they allow composition of such patterns into more complex systems, they provide a common language in a community, and they are a useful didactic device [1, 8]. In this paper we aim to define such a set of design patterns for capturing the wide variety of theories, proposals and systems in the literature on hybrid systems that combine learning and reasoning. Our patterns distinguish between systems both on the functionality of their components and on the representations that they deploy, using both of the distinctions discussed above. We show that our patterns are indeed compositional (complex configurations can be built by composing simple architectures), and we claim a substantial degree of completeness for our library of compositional patterns, by validating them against a body of more than 50 papers from the research literature of the past decade.

We will now introduce the informal graphical “boxology” notation that we use to express our patterns. We use ovals to denote algorithmic components (i.e. objects that perform some computation), and boxes to denote their input and output (i.e. data structures). Following the task-based dichotomy described above, we distinguish two types of algorithmic components (ovals): those that perform some form of deductive inference (labelled as the “KR” components) and those that perform some form inductive inference (the “ML” components): KR ML. Based on the representation-based distinction discussed above, we also use two kinds of input- and output-boxes: those that

contain "model-based" (symbolic, relational) structures, those that contain "model-free" data: $\boxed{\text{sym}}\boxed{\text{data}}$
The *sym-boxes* are the input and output of a classical KR reasoning system:

$$\begin{tikzcd} \boxed{\text{sym}} \arrow[r] & (\text{KR}) \arrow[r] & \boxed{\text{sym}} \end{tikzcd} \tag{1}$$

and idem the data boxes are the typical input and output boxes of an ML system:

$$\begin{tikzcd} \boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{data}} \end{tikzcd} \tag{2}$$

Based on the discussion in the previous section, the labels "inductive" and "deductive" for the algorithmic components would have been more accurate, but we use the labels "KR" and "ML" for brevity. Similarly, the labels "model-based" and "model-free" (or "model-based" and "function-based") would have been more accurate for the input- and output-boxes, but again we use the labels "sym" and "data" for brevity.

4 A library of Patterns

In this section we identify common patterns for hybrid systems that perform reasoning and learning.

Learning with symbolic input and output

Instead of applying ML techniques to model-free data such as images, text or numbers, the ML techniques can be applied to symbolic structures, also yielding symbolic output:

$$\begin{tikzcd} \boxed{\text{sym}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{sym}} \end{tikzcd} \tag{3}$$

A classical examples of this are the aforementioned approaches based on Inductive Logic Programming [29, 42], Probabilistic Soft Logic [3, 24] and Markov Logic Networks [41]. Even this simple example shows the value of these abstract patterns: even though the algorithms and representations of ILP, PSL and MLN's are completely different, the architecture patterns shows that they are all aimed at the same goal: inductive reasoning over symbolic structures.

From symbols to data and back again

A more recent class of this "graph completion" systems [35, 37, 50] also satisfies this design pattern: a machine learning algorithm takes a knowledge graph as input and uses inductive reasoning to predict addition edges which are deemed to be true based on observed patterns in the graph, even though they are missing from the original graph. However, allmost all graph completion algorithms perform this task by first translating the knowledge graph to a representation in a high-dimensional vector space (a process called "embedding"), to the following refinement of pattern (3) would be more accurate:

$$\begin{tikzcd} \boxed{\text{sym}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{sym}} \end{tikzcd} \qquad (4)$$

Learning from data with symbolic output

A variation of the above is when ML techniques are applied to model-free, but still yielding symbolic, model-based output:

$$\begin{tikzcd} \boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{sym}} \end{tikzcd} \qquad (5)$$

The typical example here is ontology learning from text [2]. Again, a large number of different approaches are captured by this single pattern: ontology learning using Inductive Logic Programming [31], using conceptual spaces [10] or text mining [52] are all described by pattern (5). A related but different instantiation of this patter is the use of text-mining not to learn full-blown ontologies, but to learn just the class/instance distinction (which is always problematic in ontology modelling), as done in [36]. As concerns the design patterns, this work only differs in the actual content of the symbolic output: a full-blown ontology, or only a class/instance label.

An entirely different application of this pattern is not to learn an ontology, but instead to learn a knowledge graph, as done in [40] by using Probabilistic Soft Logic as the learning engine. It is useful to note that many "classical" learning algorithms such as decision tree learning and rule mining are also covered by this design pattern.

Explainable learning systems

A major motivation for pattern (5) is the opaqueness problem mentioned in Section 2: the symbolic output is more amenable to crafting an explanation of the learning results [51]. A natural extension of this pattern is therefore to use the symbolic output as input for a classical reasoning system, where the reasoning systems is used to craft an intelligible explanation of the results of the machine learner.

$$\begin{tikzcd}
\boxed{\text{data}} \arrow[r] & \text{ML} \arrow[r] & \boxed{\text{sym}} \arrow[r] & \text{KR} \arrow[r] & \boxed{\text{sym}}
\end{tikzcd} \tag{6}$$

Explainable learning systems with background knowledge

An extension of this pattern describes the work in both [44] and [49], where background knowledge is used in the process of deductively reconstructing an explanation for the results of the learner:

$$\begin{tikzcd}
 & & & \boxed{\text{sym}} \arrow[d] & \\
\boxed{\text{data}} \arrow[r] & \text{ML} \arrow[r] & \boxed{\text{sym}} \arrow[r] & \text{KR} \arrow[r] & \boxed{\text{sym}}
\end{tikzcd} \tag{7}$$

Explainable learning systems through inspection

An alternative approach to explainable systems is taken in [11], where the behaviour of machine learning system (in this case: a neural net classifier trained with transfer learning) is inspected by a reasoning system (in this case: a Description Logic reasoner), which then tries to explain the behaviour of the learner (in this case: which features were succesfully used in the transfer learning process).

$$\begin{tikzcd}
\text{data} \arrow[r] \arrow[dr] & \text{ML} \arrow[r] & \text{data} \\
& \text{KR} \arrow[ur] & \\
& \text{sym} \arrow[u] &
\end{tikzcd} \tag{8}$$

Learning an intermediate abstraction for learning

In pattern (4) we have already seen a case where an intermediate representation is produced by one learning system as the input for a subsequent learning system. This turns out to be a rather generic pattern, of which also other variations are possible. One such variation is described in [22], where perceptual ("model-free") input is used to learn an intermediate symbolic ("model-based") representation of a the environment, and this symbolic spatial representation is then used in a reinforcement learning step to learn optimal behaviour:

$$\begin{tikzcd}
\text{data} \arrow[r] & \text{ML} \arrow[r] & \text{sym} \arrow[r] & \text{ML} \arrow[r] & \text{data}
\end{tikzcd} \tag{9}$$

The results in [22] show that the intermediate (and more abstract) symbolic representation gives a more robust behaviour of the system and allows for transfer learning between situations. Besides learning a spatial abstraction (as in [22]), the work in [28] uses the same architecture pattern for deriving a temporal abstraction of sequence of subtasks, which are then input to reinforcement learning agents.

Learning an intermediate abstraction for reasoning

Contrary to widespread popular belief, the Alpha Go system is not a single machine learning system. It is in fact built out of a machine learning component which learns functions for board valuation and region selection, which are subsequently used as components in a (classical) Monte Carlo search. This architecture can be described as

$$\begin{tikzcd}
\boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{sym}} \arrow[r] & (\text{KR}) \arrow[r] & \boxed{\text{sym}}
\end{tikzcd} \tag{10}$$

Deriving an intermediate abstraction for reasoning

In [32] a raw data-stream is first abstracted into a stream of symbols with the help of a symbolic ontology, and this stream of symbols is then fed into a classifier (which performs better on the symbolic data than on the original raw data).

$$\begin{tikzcd}
 & (\text{sym}) \arrow[d] & & & \\
\boxed{\text{data}} \arrow[r] & (\text{KR}) \arrow[r] & \boxed{\text{sym}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{sym}}
\end{tikzcd} \tag{11}$$

Learning with symbolic information as a prior

The following design patterns aims to resolve one of the issues mentioned in Section 2, namely how to enable machine learning systems to use prior knowledge:

$$\begin{tikzcd}
 & \boxed{\text{sym}} \arrow[d] & \\
\boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{data}}
\end{tikzcd} \tag{12}$$

An example of this are the Logic Tensor Networks in [17], where the authors show that encoding prior knowledge in symbolic form allows for better learning results on fewer training data, as well as more robustness against noise. A similar example is given in [4], where knowledge graphs are successfully used as priors in a scene description task, and in [16] where logical rules are used as background knowledge for a gradient descent learning task in a high-dimensional real-valued vector space.

The work by [53] is at first sight apparently unrelated: it investigates the use of a semantically formulated loss-function to drive the gradient descent learning process (the semantic loss function is defined as a propositional formula in conjunctive normal form, which is then made differentiable by weakening satisfiability to maximal satisfiability). But when drawing the design pattern for this work we arrive at precisely diagram (12) above. This suggests that we could look at [17] and [4] with entirely different eyes, namely that they are in essence using their background knowledge as encoding a “semantic loss” function, and on closer inspection this is in fact a rather faithful account of what these papers are doing. This analogy was not mentioned at all in these papers, but was revealed by the design pattern that describes these systems.

Learning with derived symbolic information as a prior

Of course the “inductive bias” (using the terminology from [6]) does not need to be given, but can itself be derived by a reasoner, leading to a variation of pattern (12):

$$\begin{tikzcd}
\boxed{\text{sym}} \arrow[r] & (\text{KR}) \arrow[r] & \boxed{\text{sym}} \arrow[d] & \\
& \boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{data}}
\end{tikzcd} \tag{13}$$

Meta-reasoning for control

There is a long-standing tradition in both AI [14] and in the field of cognitive architectures (e.g. [39]) to investigate so-called meta-reasoning systems, where one system reasons about (or:learns from) the behaviour of another system.

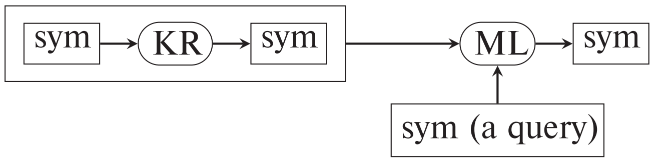
In one pattern, also known as meta-cognition, symbolic reasoning is used to control the behaviour of a learning agent, to decide what it should learn and when, when it should stop learning, and in general to decide on the hyper-parameters that control the learning process:

$$\begin{tikzcd}
& (\text{KR}) \arrow[d, leftrightarrow, "\text{sym}"] & \\
\boxed{\text{data}} \arrow[r] & (\text{ML}) \arrow[r] & \boxed{\text{data}}
\end{tikzcd} \tag{14}$$

Here the KR system has a symbolic representation of the state of the ML system, reasons about it, and effectuates its conclusions as control instructions to the ML system. In a loose cognitive analogy, this could be compared with a consciously learning student, who constantly reflects on her learning progress to adjust her learning behaviour.

Meta-reasoning for learning to reason

In a second meta-reasoning pattern, the behaviour of one system (a symbolic reasoner) is the input of a second, machine learning, system. The machine learning system observes the behaviour of the symbolic reasoner, and learns from this behaviour how to perform deductive behaviour, which it is then able to mimic on new symbolic queries:



(15)

This pattern for training a neural network to do logical reasoning captures a wide variety of approaches such as reasoning over RDF knowledge bases [18], Description Logic Reasoning [27] and logic programming [43].

Compositional systems

The first of our patterns (patterns (1), (2), (3) and (5)) are the elementary building blocks out of which the more complex patterns can all be constructed. Compositionality can also be seen between more complex patterns, as in the meta-reasoning diagrams.

For example, (4) is a sequential composition of (3) and (5); pattern (13) is a non-sequential of the two elementary patterns (1) and (2).

5 Future Work

In future work, we intend to design a set of grammar rules that generate the space of all syntactically possible combinations of our atomic patterns. This should be followed by an attempt to find examples of all of these patters in the literature, and (if no exemplars are found in the literature) to investigate if these are meaningful combinations that have not yet been explored.

A simple but interesting future extension of our notation would be to introduce a third type of processor (besides symbolic reasoner and learning system) namely that of a human agent. This would then allow our architecture patterns to be extended to human-in-the-loop systems, including the recently emerging family of hybrid intelligence systems that combine AI systems with humans in a single team.

A more fundamental and almost philosophical issue to be addressed in future work is the distinction between *data* and *symbols*. Even though there is a shared intuition about this distinction (as in the earlier cited papers of Pearl [38] and Darwiche [15], we have not been able to come up with a crisp distinction, let alone a way to capture this distinction formally.

6 Concluding Comments

What is notably different in our approach from other survey work in the literature is that we are not categorizing work based on the specific techniques that are being used inside the building blocks, but only on how the building blocks fit together. Each category abstracts from specific mathematical and algorithmic details of the specific approaches in that category, but only looks at the functional behaviour of the pattern and at the functional dependencies between the ML and KR components. This makes our categorization of systems in design patterns much more abstract and general. For example, while major battles are being fought in the literature between different forms of statistical relational learning, we abstract all of these approaches into a single design pattern (in this case pattern (5), allowing us to see that any of them could be deployed in more complex configurations such as (9) or (6).

Based on the experience with design patterns in other subdisciplines of Computer Science such as Software Engineering and Knowledge Engineering, we hope that this classification of a wide variety of systems in a small number of compositional patterns will help with a better understanding of the design space of such systems, including a better understanding of the advantages and disadvantages of the different configurations, and a better understanding which design patterns are more suited for which kinds of performance tasks. Examples of this in the above were the use of an intermediate symbolic representation of space in [22] to obtain more efficient and robust learning, the use of a symbolic representation in [44] to produce explanations of the results of a classical learner, the use in [32] of a symbolic reasoner to obtain a data abstraction which improved the performance of a subsequent learning algorithm, etc.

Finally, we are experimenting with this approach as a didactic device[10]

Although we have refrained from linking our design patterns to the design of cognitive architectures (see [12] for a survey, it is tempting to do so. System such as ACT-R, SOAR, Sigma and others distinguish components for temporal and spatial abstraction, for short and long term memory, for goal formulation and attention guidance, etc. Some of the patterns we have discussed are clearly reminiscent of some of these cognitive functions, and a study of these analogies would yield potentially interesting insights.

Further obvious next steps in this work would be to perform a deeper analysis in which to apply these patterns to a wider body of literature, to formalize and further refine the informal descriptions in this paper, and to ultimately use this approach in a prescriptive design theory of statistical-symbolic systems.

References

[1] Ellen Agerbo and Aino Cornils. “How to Preserve the Benefits of Design Patterns”. In: *Proceedings of the 1998 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA ’98), Vancouver,*

[10]and in fact, two years of teaching a research seminar on combining statistical and symbolic approaches in AI has been our main motivator for this work.

British Columbia, Canada, October 18–22, 1998. Ed. by Bjørn N. Freeman-Benson and Craig Chambers. ACM, 1998, pp. 134–143. doi: 10.1145/286936.286952. URL: https://doi.org/10.1145/286936.286952.

[2] Muhammad Nabeel Asim et al. “A survey of ontology learning techniques and applications”. In: *Database* 2018 (2018), bay101.

[3] Stephen H. Bach et al. “Hinge-Loss Markov Random Fields and Probabilistic Soft Logic”. In: *Journal of Machine Learning Research* 18 (2017), 109:1–109:67. URL: http://jmlr.org/papers/v18/15-631.html.

[4] Stephan Baier, Yunpu Ma, and Volker Tresp. “Improving Visual Relationship Detection Using Semantic Modeling of Scene Descriptions”. In: *The Semantic Web – ISWC 2017 – 16th International Semantic Web Conference, Vienna, Austria, October 21–25, 2017, Proceedings, Part I*. Ed. by Claudia d’Amato et al. Vol. 10587. Lecture Notes in Computer Science. Springer, 2017, pp. 53–68. ISBN: 978-3-319-68287-7. doi: 10.1007/978-3-319-68288-4_4. URL: https://doi.org/10.1007/ 978-3-319-68288-4%5C_4.

[5] A. Barron, J. Rissanen, and Bin Yu. “The minimum description length principle in coding and modeling”. In: *IEEE Transactions on Information Theory* 44.6 (Oct. 1998), pp. 2743–2760. ISSN: 0018-9448. doi: 10.1109/18.720554.

[6] Peter W. Battaglia et al. “Relational inductive biases, deep learning, and graph networks”. In: *CoRR* abs/1806.01261 (2018). arXiv: 1806. 01261. URL: http://arxiv.org/abs/1806.01261.

[7] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. “Abstractions in Process Mining: A Taxonomy of Patterns”. In: *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8–10, 2009. Proceedings. Ed.* by Umeshwar Dayal et al. Vol. 5701. Lecture Notes in Computer Science. Springer, 2009, pp. 159–175. doi: 10.1007/978-3-642-03848-8_12. URL: https://doi.org/10.1007/978-3-642-03848-8%5C_12.

[8] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. “Abstractions in Process Mining: A Taxonomy of Patterns”.

In: *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8–10, 2009. Proceedings. Ed.* by Umeshwar Dayal et al. Vol. 5701. Lecture Notes in Computer Science. Springer, 2009, pp. 159–175. doi: 10.1007/978-3-642-03848-8_12. URL: https://doi.org/10.1007/978-3-642-03848-8%5C_12.

[9] Léon Bottou. "From Machine Learning to Machine Reasoning". In: *CoRR* abs/1102.1808 (2011). arXiv: 1102.1808. URL: http://arxiv. org/abs/1102.1808.

[10] Zied Bouraoui, Shoaib Jameel, and Steven Schockaert. "Inductive Reasoning about Ontologies Using Conceptual Spaces". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4–9, 2017, San Francisco, California, USA*. Ed. by Satinder P. Singh and Shaul Markovitch. AAAI Press, 2017, pp. 4364–4370. URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14916.

[11] Jiaoyan Chen et al. "Knowledge-Based Transfer Learning Explanation". In: *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October – 2 November 2018*. Ed. by Michael Thielscher, Francesca Toni, and Frank Wolter. AAAI Press, 2018, pp. 349–358. URL: https://aaai.org/ocs/index.php/KR/KR18/paper/view/18054.

[12] Hui-Qing Chong, Ah-Hwee Tan, and Gee Wah Ng. "Integrated cognitive architectures: a survey". In: *Artif. Intell. Rev.* 28.2 (2007), pp. 103–130. doi: 10.1007/s10462-009-9094-9. URL: https://doi.org/10. 1007/s10462-009-9094-9.

[13] Rudi Cilibrasi and Paul M. B. Vitányi. "Clustering by compression". In: *IEEE Trans. Information Theory* 51.4 (2005), pp. 1523–1545. doi: 10.1109/TIT.2005.844059. URL: https://doi.org/10.1109/TIT. 2005.844059.

[14] Stefania Costantini. "Meta-reasoning: A Survey". In: *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*. Ed. by Antonis C. Kakas and Fariba Sadri. Vol. 2408. Lecture Notes in Computer Science.

Springer, 2002, pp. 253–288. doi: 10.1007/3-540-45632-5_11. URL: https://doi.org/10.1007/3-540-45632-5%5C_11.

[15] Adnan Darwiche. "Human-level Intelligence or Animal-like Abilities?" In: *Commun. ACM* 61.10 (Sept. 2018), pp. 56–67. ISSN: 0001-0782. doi: 10.1145/3271625. URL: http://doi.acm.org.vu-nl.idm. oclc.org/10.1145/3271625.

[16] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. "Lifted Rule Injection for Relation Embeddings". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016.* Ed. by Jian Su, Xavier Carreras, and Kevin Duh. The Association for Computational Linguistics, 2016, pp. 1389–1399. URL: http://aclweb.org/ anthology/D/D16/D16-1146.pdf.

[17] Ivan Donadello, Luciano Serafini, and Artur S. d'Avila Garcez. "Logic Tensor Networks for Semantic Image Interpretation". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017.* Ed. by Carles Sierra. ijcai.org, 2017, pp. 1596–1602. ISBN: 978-0-9992411-0-3. doi: 10.24963/ijcai.2017/221. URL: https://doi.org/10.24963/ ijcai.2017/221.

[18] Monireh Ebrahimi et al. "Reasoning over RDF Knowledge Bases using Deep Learning". In: *CoRR* abs/1811.04132 (2018). arXiv: 1811.04132. URL: http://arxiv.org/abs/1811.04132.

[19] Erich Gamma. "Design Patterns – Past, Present and Future". In: *The Future of Software Engineering.* Ed. by Sebastian Nanz. Springer, 2010, p. 72. doi: 10.1007/978-3-642-15187-3_4. URL: https://doi. org/10.1007/978-3-642-15187-3%5C_4.

[20] Erich Gamma et al. *Design Patterns: Elements of Reusable Object-Oriented Software.* 1st ed. Addison-Wesley Professional, 1994. ISBN: 0201633612.

[21] Aldo Gangemi and Valentina Presutti. "Ontology Design Patterns". In: *Handbook on Ontologies.* Ed. by Steffen Staab and Rudi Studer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 221–243. doi: 10.1007/978-3-540-92673-3_10. URL: https://doi.org/10.1007/ 978-3-540-92673-3_10.

[22] Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. “Towards Deep Symbolic Reinforcement Learning”. In: *CoRR* abs/1609.05518 (2016). arXiv: 1609.05518. URL: http://arxiv.org/abs/1609.05518.

[23] Hector Geffner. “Model-free, Model-based, and General Intelligence”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden.* Ed. by Jérôme Lang. ijcai.org, 2018, pp. 10–17. ISBN: 978-0-9992411-2-7. doi: 10.24963/ijcai.2018/2. URL: https://doi.org/ 10.24963/ijcai.2018/2.

[24] Lise Getoor. “Probabilistic Soft Logic: A Scalable Approach for Markov Random Fields over Continuous-Valued Variables – (Abstract of Keynote Talk)”. In: *Theory, Practice, and Applications of Rules on the Web – 7th International Symposium, RuleML 2013, Seattle, WA, USA, July 11–13, 2013. Proceedings.* Ed. by Leora Morgenstern et al. Vol. 8035. Lecture Notes in Computer Science. Springer, 2013, p. 1. ISBN: 978-3-642-39616-8. doi: 10.1007/978-3-642-39617-5_1. URL: https://doi.org/10.1007/978-3-642-39617-5%5C_1.

[25] Lise Getoor. “Statistical Relational Learning: Unifying AI and DB Perspectives on Structured Probabilistic Models”. In: *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14–19, 2017.* Ed. by Emanuel Sallinger, Jan Van den Bussche, and Floris Geerts. ACM, 2017, p. 183. ISBN: 978-1-4503-4198-1. doi: 10.1145/3034786. 3056450. URL: https://doi.org/10.1145/3034786.3056450.

[26] Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter, eds. *Handbook of Knowledge Representation.* Vol. 3. Foundations of Artificial Intelligence. Elsevier, 2008. ISBN: 978-0-444-52211-5. URL: http://www.sciencedirect.com/science/bookseries/15746526/3.

[27] Patrick Hohenecker and Thomas Lukasiewicz. “Deep Learning for Ontology Reasoning”. In: *CoRR* abs/1705.10342 (2017). arXiv: 1705 . 10342. URL: http://arxiv.org/abs/1705.10342.

[28] Rodrigo Toro Icarte et al. “Teaching Multiple Tasks to an RL Agent using LTL”. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10–15, 2018*. Ed. by Elisabeth André et al. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018, pp. 452–461. URL: http://dl.acm. org/citation.cfm?id=3237452.

[29] Katsumi Inoue, Hayato Ohwada, and Akihiro Yamamoto. “Special issue on inductive logic programming”. In: *Machine Learning* 106 (2017), pp. 1863–1865.

[30] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models – Principles and Techniques*. MIT Press, 2009. ISBN: 978-0-262-01319-2. URL: http://mitpress.mit.edu/catalog/item/ default.asp? ttype=2%5C&tid=11886.

[31] Stasinos Konstantopoulos and Angelos Charalambidis. “Formulating description logic learning as an Inductive Logic Programming task”. In: *FUZZ-IEEE 2010, IEEE International Conference on Fuzzy Systems, Barcelona, Spain, 18–23 July, 2010, Proceedings*. IEEE, 2010, pp. 1–7. doi: 10.1109/FUZZY.2010.5584417. URL: https://doi.org/10. 1109/FUZZY.2010.5584417.

[32] Reinier Kop et al. “Predictive modeling of colorectal cancer using a dedicated pre-processing pipeline on routine electronic medical records”. In: *Comp. in Bio. and Med.* 76 (2016), pp. 30–38.

[33] Brenden M. Lake et al. “Building Machines That Learn and Think Like People”. In: *The Behavioral and brain sciences* 40 (2017), e253.

[34] Gary Marcus. “Deep Learning: A Critical Appraisal”. In: *CoRR* abs/1801.00631 (2018). arXiv: 1801.00631. URL: http://arxiv.org/ abs/1801.00631.

[35] Maximilian Nickel et al. “A Review of Relational Machine Learning for Knowledge Graphs”. In: *Proceedings of the IEEE* 104.1 (2016), pp. 11–33. doi: 10.1109/JPROC.2015.2483592. URL: https://doi.org/10. 1109/JPROC.2015.2483592.

[36] Ankur Padia, David Martin, and Peter F. Patel-Schneider. “Automating Class/Instance Representational Choices in Knowledge Bases”. In: *Knowledge Engineering and Knowledge*

Management – 21st International Conference, EKAW 2018, Nancy, France, November 12–16, 2018, Proceedings. Ed. by Catherine Faron-Zucker et al. Vol. 11313. Lecture Notes in Computer Science. Springer, 2018, pp. 273–288. doi: 10.1007/978-3-030-03667-6_18. URL: https://doi.org/10.1007/978-3-030-03667-6%5C_18.

[37] Heiko Paulheim. "Knowledge graph refinement: A survey of approaches and evaluation methods". In: *Semantic Web* 8.3 (2017), pp. 489–508. doi: 10.3233/SW-160218. URL: https://doi.org/10.3233/SW-160218.

[38] Judea Pearl. "Theoretical Impediments to Machine Learning With Seven Sparks from the Causal Revolution". In: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018.* Ed. by Yi Chang et al. ACM, 2018, p. 3. doi: 10.1145/3159652.3176182. URL: https://doi.org/10.1145/3159652.3176182.

[39] Don Perlis et al. "The Internal Reasoning of Robots". In: *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning, COMMONSENSE 2017, London, UK, November 6–8, 2017.* Ed. by Andrew S. Gordon, Rob Miller, and György Turán. Vol. 2052. CEUR Workshop Proceedings. CEUR-WS.org, 2017. URL: http://ceur -ws.org/Vol-2052/paper16.pdf.

[40] Jay Pujara et al. "Using Semantics and Statistics to Turn Data into Knowledge". In: *AI Magazine* 36.1 (2015), pp. 65–74. URL: http://www.aaai.org/ojs/index.php/aimagazine/article/view/2568.

[41] Matthew Richardson and Pedro M. Domingos. "Markov logic networks". In: *Machine Learning* 62.1-2 (2006), pp. 107–136. doi: 10.1007/s10994-006-5833-1. URL: https://doi.org/10.1007/s10994-006-5833-1.

[42] Fabrizio Riguzzi, Elena Bellodi, and Riccardo Zese. "A History of Probabilistic Inductive Logic Programming". In: *Front. Robotics and AI* 2014 (2014).

[43] Tim Rocktäschel and Sebastian Riedel. "End-to-end Differentiable Proving". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information*

Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA. Ed. by Isabelle Guyon et al. 2017, pp. 3791–3803. URL: http://papers.nips.cc/paper/6969-end-to-end-differentiable-proving.

[44] Md. Kamruzzaman Sarker et al. "Explaining Trained Neural Networks with Semantic Web Technologies: First Steps". In: *Proceedings of the Twelfth International Workshop on Neural-Symbolic Learning and Reasoning, NeSy 2017, London, UK, July 17–18, 2017*. Ed. by Tarek R. Besold, Artur S. d'Avila Garcez, and Isaac Noble. Vol. 2003. CEUR Workshop Proceedings. CEUR-WS.org, 2017. URL: http://ceur -ws.org/Vol-2003/NeSy17%5C_paper4.pdf.

[45] Guus Schreiber et al. "CommonKADS: A Comprehensive Methodology for KBS Development". In: *IEEE Expert* 9.6 (1994), pp. 28–37. doi: 10.1109/64.363263. URL: https://doi.org/10.1109/64.363263.

[46] Guus Schreiber et al. *Knowledge Engineering and Management The CommonKADS Approach.* MIT Press, 1999.

[47] Steffen Staab and Rudi Studer, eds. *Handbook on Ontologies.* International Handbooks on Information Systems. Springer, 2009. ISBN: 978-3-540-70999-2. doi: 10.1007/978-3-540-92673-3. URL: https: //doi.org/10.1007/978-3-540-92673-3.

[48] Peter Struss. "Model-based Problem Solving". In: *Handbook of Knowledge Representation.* Ed. by Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter. Vol. 3. Foundations of Artificial Intelligence. Elsevier, 2008, pp. 395–465. doi: 10.1016/S1574-6526(07)03010-6. URL: https://doi.org/10.1016/S1574-6526(07)03010-6.

[49] Ilaria Tiddi, Mathieu d'Aquin, and Enrico Motta. "Data Patterns Explained with Linked Data". In: *Machine Learning and Knowledge Discovery in Databases – European Conference, ECML PKDD 2015, Porto, Portugal, September 7–11, 2015, Proceedings, Part III.* Ed. by Albert Bifet et al. Vol. 9286. Lecture Notes in Computer Science. Springer, 2015, pp. 271–275. doi: 10.1007/978-3-319-23461-8_28. URL: https://doi.org/10.1007/978-3-319-23461-8%5C_28.

[50] Q. Wang et al. “Knowledge Graph Embedding: A Survey of Approaches and Applications”. In: *IEEE Transactions on Knowledge and Data Engineering 29.12* (Dec. 2017), pp. 2724–2743. ISSN: 1041-4347. doi: 10. 1109/TKDE.2017.2754499.

[51] Daniel S. Weld and Gagan Bansal. “Intelligible Artificial Intelligence”. In: *CoRR* abs/1803.04263 (2018). arXiv: 1803 .04263. URL: http : //arxiv.org/abs/1803.04263.

[52] Wilson Wong, Wei Liu, and Mohammed Bennamoun. “Ontology learning from text: A look back and into the future”. In: *ACM Comput. Surv.* 44.4 (2012), 20:1–20:36. doi: 10.1145/2333 112.2333115. URL: https://doi.org/10.1145/2333112.2333115.

[53] Jingyi Xu et al. “A Semantic Loss Function for Deep Learning with Symbolic Knowledge”. In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10–15, 2018.* Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. JMLR Workshop and Conference Proceedings. JMLR.org, 2018, pp. 5498–5507. URL: http://proceedings.mlr.press/v80/ xu18h.html.

[54] Jihong Yan et al. “A retrospective of knowledge graphs”. In: *Frontiers of Computer Science* 12.1 (Feb. 2018), pp. 55–74. ISSN: 2095-2236. doi: 10.1007/s11704-016-5228-9. URL: https://doi.org/
10.1007/ s11704-016-5228-9.

[55] Fan Yang, Zhilin Yang, and William W. Cohen. “Differentiable Learning of Logical Rules for Knowledge Base Reasoning”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 2316–2325. URL: http://papers.nips.cc/paper/6 826-differentiable-learning-of-logical-rules-for-knowledge-base-reasoning.

Biographies

Frank van Harmelen is professor of Knowledge Representation and Reasoning at the VU University Amsterdam, and adjunct professor at Wuhan University and Wuhan University of Science and Technology. He played a leading role in the development of the Semantic Web, which aims to make data on the web semantically interpretable by machines through formal representations. He was a contributor to the Web Ontology Language OWL, now a standard in worldwide commercial use, and the basis for an entire research community. He is one of the architects of the semantic storage engine Sesame (now RDF4J). This work received the 10-year impact award of the Semantic Web community. He co-authored the Semantic Web Primer, the first text book on the semantic web (now translated into 5 languages), and he co-edited the standard reference work in his field (The Handbook of Knowledge Representation). He is a member of the Royal Netherlands Academy of Arts and Sciences, the Royal Holland Academy of Sciences, and Academia Europea, and fellow of the European Association for Artificial Intelligence.

Annette ten Teije is an associate professor at the Vrije Universiteit Amsterdam. Her interests are in knowledge modelling, representation and reasoning in particular in the medical domain. She earned a PhD (1997) from the University of Amsterdam (SWI) for her thesis entitled “Automated configuration of problem solving methods in diagnosis”. She was involved in a number of EU-funded projects IBROW project under the FET-O programme, Protocure-II project, concerned with formal modelling and verification of medical guidelines and protocols, WS-DIAMOND FET-Open project concerned with self- healing web-services, FP7-ICT EURECA project concerned with enabling information re-use by linking clinical research and clinical care. She was program co-chair for the 18th International Conference on Knowledge Engineering and Knowledge Management (2012), and was the general chair of the 16th conference on Artificial Intelligence in Medicine.