Learning Concepts through Differentiable Logical Induction

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

We develop a framework where an algorithm learns concepts that have symbolic and subsymbolic content. The generative algorithm can learn the logical structure underlying a set of observed data while simultaneously operating on dense vector representations of concepts. We revisit the framework studied by previous work that models semantic cognition as a form of logical dimensionality reduction inducing a set of abstract rules, while simultaneously benefiting benefiting from the recent substantial progress of artificial neural networks. Our algorithm inherits some of the advantages of both perspectives by combining the simplicity of forward chaining with the power of parametrized unification. We show that these Neural Logical Reasoners can perform inductive and deductive inferences while handling ambiguity and noise, at much better speed than previous symbolic approaches.

Preamble. About Concepts

There is much to say about concepts. Crucially, the meaning of a concept depends in part of their relationship with other concepts and in part of their relation with the world. Recent theories from cognitive science research posit two aspects for concepts and their determination. 'Narrow' content is related to conceptual role, this part of the meaning of a concept comes from how it relates to other concpets. The Theory Theory of concepts (TODOREFERENCE) posits that you cannot learn and represent concepts in isolation, instead concepts acquire their meaning arrises in the context of a whole conceptual system. Thus it has been argued that children dont learn concepts like ENERGY, MOMENTUM and ACCELERATION in terms of concepts they already know. Rather, they gradually learn how to use the whole new terminology. It is worth noting this is very aligned with predicate invention in the context of Inductive Logic Programing (ILP), in those cases the meaning of a predicate like EVEN comes from how it relates to other predicates.

On the other hand, 'wide' content is related to the referents picked up out there in the world by the representations. This Dual-Factor Theory of concepts is influenced by Kripke and Putnam's subtle analysis suggesting that references cannot be entirely determined by internal relations of concepts with other concepts. That is, what we know about entities picked out by a concept cannot be entirely what determines which entities those are, that is because what we know about entities is always subject to revision and to be mistaken. Instead of conceptual role semantics, wide content is mediated by informational semantics which is more related to statistical covariation between representations and reality. To not get into a philosophical digression, we refer the reader to relevant extensive literature(TODOREFERENCE). Instead we just mention that this duality of factors can perhaps help accomodate a wide range of psychological data from the 1970's showing that people can rank concepts in terms of their typicality and let them to propose that concepts are better characterized as lists of properties of features. Thes reankings are robust and have a direct effect on the speed of categorization(TODOREFERENCE) and also provides one candidate explanation of why concepts are rather fuzzy and inexact.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

Most representaions in artificial inteligence dont have rich dual-factor components. Symbolic approaches seem intuitively more related to conceptual role, often explicitly specifying the relationships between variables. In contrast, subsymbolic approaches compute representations with little conceptual role, at the very least, the "meaning" of the representations is obtained through statistical covariation rather than loical composition. The two approaches seem strikingly complementary in their strengths and weaknesses. Just like in the phiosophical debates around concepts, both approaches have managed to appeal some of the smartest people in the field, often opposing them. Just like with the history of the theory of concepts, perhaps these suggests a combination.

1 Introduction

Until recently, most research on the problem of learning logical rules such as $grandfatherof(x, y) \leftarrow fatherof(x, z), parentof(z, y)$ from data was in the context of symbolic systems like Prolog(REFERENCE). These algorithms have a number of desirable properties. First, they are interpretable, as they learn explicit relations between the concepts. Second, they tend to be very data efficient, able to generalize well from a handful of examples. While Neural Networks tend to require a lot of training examples and struggle to generalise, ILP approaches are doomed to generalised by learning general rules involving free variables that apply for all concepts. Third, as (EVANS) argue, this models easily support transfer learning, being explicit, learned rules can be transfered to other tasks to continue further training. Relatedly, this models present a straight forward way of incorporating domain-specific knowledge in the form of explicit logical rules. In contrast, traditional ILP models are inable to handle noisy, erroneous or ambiguous data. A recent paper (REFERENCE) proposed a differentiable version of ILP. This work combined some of the advantages of ILP and of neural network-based systems which allowed them to use gradient descent and backpropagation to lear rules that were interpretable and data-efficient while robust to noisy and ambiguous data. While our work ressembles in several senses this work, it also differs in several dimensions. Fundamentally, while their approach, like previous ILP solutoins, requires to first generate all the possible rules and then learns to select the relevant rules with a mechanism akin to attention; our approach opperates at the more modular and compositional level of individual concepts, and directly learns the atoms that conform the logical rules.

In a less explored domain, previous symbolic approaches have also been embedded in hierarchical Bayesian models that are capable of performing a kind of dimensionlaity reduction for structured logical theories(REFERENCE). These probabilistic generative models learn theories by inducing a set of logical rules along with a set of core relations that form a compression of the data which can be recovered using the logical rules. As an example of how theories support compression consider an animal taxonomy like that of Figure 1. The algorithm can learn to compress all the information about salmons into the core relation IS(salmon, fish) which is sufficient to recover and infer all sorts of things about salmons (IS(salmon, animal), HAS(salmon, fins), etc.) using other core relations and a set of learned logical rules ($IS(x, y) \leftarrow IS(x, z), IS(z, y)$; $HAS(x, y) \leftarrow IS(x, z), HAS(z, y)$). Thus through the induction of logical rules, the algorithm can learn to make deductive inferences. The additional induction of core relations allows the very interesting capability to the argument of making inductive inference. When observing that salmons have fins, and gills and are animals, these probabilistic algorithms can be incentived to compress the information into the single core relation that salmons are fishes. These capability of making very rich inductive and deductive inferences from very sparse data and some general abstract knowledge is a landmark of human learning. While the Bayesian symbolic presented a very promising direction with interesting results it showed limitation in terms of speed and scalability to real datasets. It also suggest challengins in terms of connecting to lower level perception modules.

In this work we explore to what extent the substantial progress made in the last year(s) in terms of modelling logical reasoning with neural approaches can capture the indctive and deductive inferences shown in previous work. We build on(SEBASTIANS REFERENCE) and work with a new version of neural logical reasoning that combines the simplicity of forward chaining with the flexibility of parametrized unification by operating in dense vector representations. This allows to apply rules when the symbols of the atoms are not equal but similar in meaning and thus replace symbolic comparison with a graded notion of similarity. This approach can also connect seamlesly to upstream perception units and scale better due to the efficacy of SGD and backpropagation. We show that

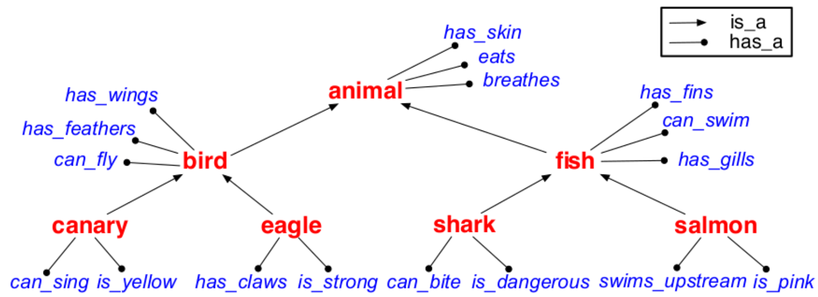

Figure 1: Animal Taxonomy. Constants in red and blue, relations indicated with lines and arrows.

neural logical reasoners can perform induction and deduction at much better speed than previous symbolic approaches.

The structure of the paper is as follows. We begin with an overview of some relevant background. Covering ILP and other recent neural models. Then in section 3 we propose our algorithm that performs a differentiable form of ILP where dense vector representations are learned through back-propagation while forming the building blocks of logical rules. In section 4 we discuss three differnt case studies that highlight different capabilities of our approach. We conclude in section 5 with a discussion about the limitations and possible future directions of the proposed framework.

2 Background

2.1 Inductive Logic Programming

A Logic Program is a set of logical if-then rules and background facts which can be used to answer queries. A rule in these systems is typically of the form:

$$h \leftarrow b_1, b_2, \ldots, b_k \tag{1}$$

Here h is called the *head* of the rule and $b_1, b_2, \ldots, b_k$ constitute the *body*. Intuitively the head of the rule is true if each of the b_i in the body are. For example a rule might be:

$$grandfather(X, Y) \leftarrow father(X, Z), parent(Z, Y) \tag{2}$$

which is read: X *is the grandfather of* Y *if* X *is the father of* Z *AND* Z *is the parent of* Y. Here X and Y are universally quantified and Z is existentially quantified. This means that the rule holds for any X and Y as long there is some individual Z for which $father(X, Z)$ and $parent(Z, Y)$ are both true facts. Given background facts such as $father(Bill, Mary)$ and $parent(Mary, Liz)$, an logic programming system can use the rule to prove a goal fact $grandfather(Bill, Liz)$. This is an example of a *forward chaining* deduction because it starts from a set of facts and matches (unifies) the body of a rule to derive the goal. It is also possible to do *backward chaining* in which we start with a goal and work backwards by unifying it with the head of a rule, recursively trying to prove the body.

But specifying all the rules is cumbersome and brittle and we would like instead to be able to learn the rules from examples. Learning the rules necessary for a logic program from a set of background facts and positive and negative examples of the goal is called *inductive logic programming* (ILP) system. In practice, the necessary rules are chosen from a human-supplied template or meta-rule which narrows the space of possibilities. ILP has been extensively developed over the last three decades for symbolic systems but only recently recast as a continuous optimization problem amenable to solution using neural networks and stochastic gradient descent [FIX: citations].

Symbolic ILP systems do very well at generalizing from just a few examples. This is because they are learning universal rules. They are however susceptible to noisy inputs and even a single bad fact can cause them to fail. On the other hand, neural systems generally are very robust to noisy input but sample inefficient and prone to overfitting on small amounts of data. Differentiable ILP systems aim for the best of both worlds. They can be made robust to noisy inputs while still retaining some of the strong generalization properties typically associated with symbolic systems. However, current systems suffer from poor memory scalability since they must create ground versions of their rules.

2.2 Differentiable ILP systems

There are two differentiable ILP approaches of which we are aware [FIX: cite, evans and rock]. Both of these approaches assume a rule template which describes the structure of any candidate rule to be induced. Both construct a differentiable function which implements a proof of the desired goal. Both require grounding the constructed rules on all the constants – effectively creating a family of ground rules to evaluate. And both offer interpretable rules after training. However they differ in several key respects.

First, [cite: rock] constructs a function representing a backward-chained proof of the goal, while [cite: evans] do a forward chained proof of the goal from the initial facts. [cite: evans] requires a representation of the truth values of all possible facts and non-facts, which may require considerable space. By contrast, [cite: rock] only requires a representation of the true facts. A more conceptual distinction arises in their parameterizations. In [cite: evans] the parameters are weights on the set of possible choices for each atom in the body of the rule – the rule structure. On the other hand in [cite:rock], the rule structure is, in effect, fixed and what is learned instead is the embeddings of the goal predicate and arguments.

In our approach we follow [cite:rock] in parameterizing with embeddings but use the forward rather than backward chaining approach so that we don't have to represent a proof tree explicitly. This greatly improves memory scalability since we do not need to represent all possible groundings of a set of symbolic rules.

2.3 Semantic Cognition

Joshs models for importance of:

- Deductive and inductive inferences.

- Core relations in theories

2.4 Neural Networks for Knowledge

-Sebastians paper

Ours better... not rule enforcement

2.5 Other Related Work

Go for a walk? Already in limitations.

This could be a substitute for the usual section of related work

3 Neural Logical Reasoner

In this section we describe the Neural Logical Reasoner. A model that induces logical rules through the learning of the vector embeddings that constitute the atoms. The algorithm is conceptually very simple: It starts with a set of known facts that constitutes its current Background Knowledge. For each forward step, the algorithm generates all the consequences implied by its known logical rules. Implication is done through the unifications of the predicates of the rules and those of the known facts. The generated consequences after a fixed number of forward steps are compared with a set of positive and negative examples (Figure 2).

3.1 Inference

The setup is similar to that in (SEBASTIANS). 1) Facts consists of triplets of the form Relation(Constant1, Constant2) (i.e Mother(Rosa, Andres)). 2) Concepts are associated with vector embeddings through a dictionary. For now only relations will have associated embeddings. Later we will consider the case where constants can also have embeddings. 3) Logical rules with a pre-specified structural form are initialized as randomly parameterized sets of vectors , for example: $[Head](X,Y) \leftarrow [Body_1](Y,Z),[Body_2](Z,Y)$ or $[Head](X,Y) \leftarrow [Body_1](Y,X)$. The content of these embedding representations is learned from the data. 4) Finally, fact are associated with a valuation $v \in [0, 1]$ which represents the belief of that fact being true.

Forward Chaining A forward step consists on taking a set of known facts and generating all the consequences implied by the rules. We implement two alternative methods.

1) When the problem is sufficiently simple and there are only a few concepts we use Method1. Method1 keeps a valuation for every potential fact (i.e $r_1(s, o)$) and updates its valuation according to:

$$v^{i+1}_{r_1(s,o)} = \max_{r_1 \leftarrow r_2, r_3} (\max_Z \min(v^i_{r_2(s,Z)}, v^i_{r_3(Z,o)}))$$

That is, maximising over all rules with the same head, maximizing over the free variable z, and minimising over the atoms in the rule (this implements logical conjunction, minimization can also be used).

2) Method2 considers only the set of known facts. It implements the same update, but instead of iterating through the space of all facts to find the pair that maximally implies it, it iterates through all the pairs of known facts and keeps the top K maximally implied facts.

Parametrized Unification To know if the relation of a particular fact is related to the atom of a particular rule. The unification score for the relevant embeddings $U(r, f)$is computed using a similarity metric (cosine_similarity was selected after some exploration). With the unification scores, the equation above for a particular rule $head \leftarrow body_1, body_2$ now becomes:

$$v^{i+1}_{head(s,o)} = U(body_1, fact_1) * U(body_2, fact_2) * (\max_Z \min(v^i_{fact_1(s,Z)}, v^i_{fact_2(Z,o)}))$$

Implicitly adding the requirement that for an implication to happen the unification of the rule and the considered facts has to be high.

3.2 Learning

As mentioned, logical rules are initialized as randomly parameterized sets of vectors. And the content of these embedding representations is learned from the data with backpropagation through all the forward chaining steps.

Vector and Conceptual role influence each other

- **Loss**
- **One-Hot Vectors**
- **General Embeddings**
  Note that intellligent sampling helps with that problem. Like humans

3.3 Learning Theories

- **Learning Background Knowledge** Same mechanism of induction. But for core facts
- **Constant Embeddings**

4 Case Studies

- **Space of tasks**
  A wide range of previous work has focused on different aspects of logical induction and knowledge base completion. Here we consider three different case studies to highlight the range of capabilities of our algorithm. We consider three case studies where our algorithm is better in some sense that previous considered approaches: Forward chaining simple Josh.. fast Grefenstette compositional, no memory, cleaner When compared against such a large range it has limitatins, discussed further down

4.1 Standard ILP Tasks

- **Evans and Grefenstette** Ambiguity. Noise?

4.2 Inferring Core Relations

- **Josh's**
- **Constant Embeddings?** The space of constants
- **Sparse Inference**

4.3 Completing a Knowledge Base

better: no enforcement + no ComplEX+ Simplicity

- **Countries and other Sebastians tasks** Contrast with Joshs models, ours is more scalable than those

5 Conclusion and Future work

We see our work as a proposition of a research direction that combines the simplicity of logical forward chaining with parametrized unification using dense embedding representations. Such a combination allows for concepts that acquire their meaning simultaneously from both their relations to other concepts through logical rules like in ILP, and from their subsymbolic embeddings like in Neural Networks. We suggest how such representations consists of a move towards representations with paralellisms to a contemporary notion of a "Concept" which has been the subject of much study in cognitive science and philosophy. The algorithm we proposed inherits many of the advantages of both approaches. From the symbolic side, it acquire interpretable representations that are learned with little data and show great generalization. At the same side, its subsymbolic nature allows it to handle ambiguity, noise and graded similarity, while being able to be learn through gradient descent, which makes it more scalable to bigger real datasets. We highlit this different advantages in three case studies. First we evaluate it in a set of traditional ILP tasks, replicating results from a recent seminal paper. Second, we show that the structure of the algorithm allows it to be deployable in realistic datasets where previous ILP approahes fail. Third and perhaps most importantly we show that neural logical reasoners can perform induction and deduction from sparse data, through the additional induction of core relations, with much better speed than previous bayesian symbolic approaches.

Limitations Our algorithm tries to address a broad range of aspects in several dimensions and presents limitations in all of them. From an algorithmic perspective, like all the considered previous work, our algorithm is provided with templates that contain information of the structural form of the rules. This would ideally be part of the learning algorithm. From a scalability perspective. While greatly improving in terms of memory, speed and size of the data relative to previous models, the model remains far from being able to reach bigger tasks of knowledge completion, that have been attacked with other purely neural approaches (REFERENCE GO FOR A WALK ???Do we want this). From a cognitive science perspective, the model is still more limited than its bayesian symbolic counterparts. Specifically, while those models provide graded measures of confidence in their inferences, the neural logical reasoners do not currently provide meaningful estimates of uncertainty, but see below.

Future Directions We signal some concrete (and straightforward in some cases) ways of addressing the above limitations. We also point to some clear directions to enrich our current framework that we would like to explore in future work. First, a straightforward way of having the algorithm learn the templates would be to encode the structural information of the atoms in the rules (arity and variable order) by adding dimensions to the embeddings and have the algorithm use independent unifications that it would then interpret in the desired ways. This would constitute only a slightly more complicated learning task but would maintain the same structure and mechanism of the problem that could be trained through gradient descent. Second, more interesting sampling procedures an the integration of forward with backward chaining could perhaps yield regimes more similar to those that humans yield with that could help cope with scalability to larger datasets. Third, we would like to investigate different ways of providing better estimates of uncertainty: from a full neural probabilistic formulation, to a heuristic metric based on the number of initializations and on the unification scores. Finally, an interesting direction to be explored relies on the insight that when a generative model has a very particular form, in this case the forward application of logical rules, the latent variables are forced to acquire a very particular form, in this case a set of core relations. We are starting to investigate the idea of Logical Autoencoders that because of their particular logical decoders, are forced to create powerful encoders that can perform things like inductive inference.

The current framework constitutes an attempt of a step in the direction of building the representations that let humans learn so much from so little.

Acknowledgments

Gracias.

References

Test of **?** [1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.