Logic Tensor Networks (Extended Abstract)

Artur d'Avila Garcez[1] and Luciano Serafini[2]

[1] City, University of London, UK, a.garcez@city.ac.uk
[2] Fondazione Bruno Kessler, Trento, Italy, serafini@fbk.eu

Abstract. *Logic Tensor Networks* provide a well-founded integration of deductive reasoning and relational learning using first-order logic. Logical constants are interpreted as feature vectors of real numbers, logical formulas have truth-value in the interval [0,1] and semantics defined in the domain of real numbers, all implemented in deep tensor neural networks with the use of Google's TENSORFLOW™. Keywords: Knowledge Representation, Relational Learning, Tensor Networks, Neural-Symbolic Computing, Knowledge Completion.

1 Introduction

The recent availability of large-scale data combining multiple modalities, such as image, text and audio, has opened various research and commercial opportunities, underpinned by machine learning methods and techniques [1, 7]. Recent work in machine learning has sought to combine logical services, such as knowledge completion, approximate inference, and goal-directed reasoning with data-driven statistical and neural network-based approaches. We argue that there are great possibilities for improving the current state of the art in machine learning and artificial intelligence (AI) thought the principled combination of knowledge representation, reasoning and learning [9].

Logic Tensor Networks (LTN) integrate learning based on tensor networks [10] with reasoning using first-order many-valued logic [2], all implemented in TENSORFLOW™ [6]. This enables a range of knowledge-based tasks using rich symbolic knowledge representation in first-order logic (FOL) to be combined with efficient data-driven machine learning based on the manipulation of real-valued vectors. In practice, FOL reasoning including function symbols is approximated through the usual iterative deepening of clause depth. Given data available in the form of real-valued vectors, logical soft and hard constraints and relations which apply to certain subsets of the vectors can be specified compactly in FOL. With the use of *Real Logic* and *learning as approximate satisfiability*, reasoning can help improve learning, and learning from new data may revise the constraints thus modifying reasoning.

2 Real Logic

An agent has to manage knowledge about an unbounded, possibly infinite, set of objects $O = \{o_1, o_2, \dots\}$. Some of the objects are associated with a set of quantitative attributes, represented by an n-tuple of real values $\mathcal{G}(o_i) \in \mathbb{R}^n$, which we call *grounding*. For example, a person may have a grounding into a 4-tuple containing some numerical representation of the person's name, height, weight, and number of friends in some social network. The same person may have a different grounding in a different context, and two or more people may have very similar groundings within a context. It

is this similarity that can be explored by the tensor networks for learning, subject to the constraints given by the logical relations. It is also possible that some of the attributes of an object (in the above example, a person) are unknown or missing. Object tuples can participate in a set of relations $\mathcal{R} = \{R_1, \ldots, R_k\}$, with $R_i \subseteq O^{\alpha(R_i)}$, where $\alpha(R_i)$ denotes the arity of relation R_i. We presuppose the existence of a latent (unknown) relation between the above numerical properties, i.e. groundings, and partial relational structure $\mathcal{R}$ on O. Starting from this partial knowledge, an agent is required to: (i) infer new knowledge about the relational structure on the objects of O; (ii) predict the numerical properties or the class of the objects in O. Classes and relations are not normally independent. For example, it may be the case that if an object x is of class C, $C(x)$, and it is related to another object y through relation $R(x, y)$ then this other object y should be in the same class $C(y)$. In logic: $\forall x \exists y((C(x) \wedge R(x, y)) \rightarrow C(y))$. Whether or not $C(y)$ holds will depend on the application: through reasoning, one may derive $C(y)$ where otherwise there might not have been evidence of $C(y)$ from training examples only; through learning, one may need to revise such a conclusion $C(y)$ once examples to the contrary become available. The vectorial representation permits both reasoning and learning as exemplified above and detailed in the next section.

The above forms of reasoning and learning are integrated in a unifying framework, implemented within tensor networks, and exemplified in relational domains combining data and relational knowledge about the objects in [9]. It is expected that, through an adequate integration of numerical properties and relational knowledge, differently from the immediate related literature [5, 3], LTNs will be capable of combining in an effective way first-order logical inference on open domains with efficient relational multi-class learning using tensor networks.

3 Learning as approximate satisfiability

Suppose that $O = \{o_1, o_2, o_3\}$ is a set of documents defined on a finite dictionary $D = \{w_1, ..., w_n\}$ of n words. Let $\mathcal{L}$ be the language that contains the binary function symbol $concat(x, y)$ denoting the document resulting from the concatenation of documents x and y. Let $\mathcal{L}$ contain also the binary predicate Sim which is supposed to be *true* if document x is deemed to be similar to document y. An example of grounding is the one that associates to each document its bag-of-words vector [4].

To check satisfiability on a subset of all the functions on real numbers, recall that a grounding should capture a latent correlation between the quantitative attributes of an object and its relational properties. For example, whether a document is to be classified as from being in the field of Artificial Intelligence (AI) depends on its bag-of-words grounding. If the language $\mathcal{L}$ contains the unary predicate $AI(x)$ standing for "x is a paper about AI" then the grounding of $AI(x)$, which is a function from bag-of-words vectors to [0,1], should assign values close to 1 to the vectors which are close semantically to AI. Furthermore, if two vectors are similar (e.g. according to the cosine similarity measure) then their grounding should be similar.

When a grounded theory is inconsitent, that is, there is no grounding $\mathcal{G}$ that satisfies it, we are interested in finding a grounding which satisfies *as much as possible* of it. For any formula and real number interval, this is a grounding $\mathcal{G}$ that minimizes a *satisfiability error*: an error occurs when a grounding $\mathcal{G}$ assigns a value $\mathcal{G}(\phi)$ to a clause ϕ which is outside the interval $[v, w]$ prescribed by knowledge-base $\mathcal{K}$.

LTNs were implemented as a Python library called `ltn` using Google's TENSORFLOW™. To illustrate knowledge completion in `ltn`, the well-known *friends and smokers* example [8] was used in [9]. Normally, a probabilistic approach is taken to solve this problem, and one that requires instantiating all clauses to remove variables, essentially turning the problem into a propositional one; `ltn` takes a different approach.

The facts contained in the friends and smokers knowledge-base (K) should have different degrees of truth, and this is not known. Otherwise, K would be inconsistent. Our main task is to complete K, that is, find the degree of truth of the facts. Hence, we use `ltn` to find a grounding that best approximates K. Results show that more facts can be learned with the inclusion of background knowledge than through reasoning alone. Similarly, using the symmetry of the friendship relation, `ltn` derives new facts through reasoning. The axioms in the background knowledge are satisfied with a degree of satisfiability higher than 90%.

4 Conclusion and future work

We have proposed *Real Logic* and the use of approximate satisfiability as a uniform framework for the integration of learning and reasoning, whereby knowledge and data are mapped onto real-valued vectors. With such an inference-as-learning approach, relational knowledge and state-of-the-art data-driven approaches can be combined. We are in the process of making the implementation of LTN available in TENSORFLOW™ to enable comparisons with statistical relational learning, neural-symbolic computing, and (probabilistic) inductive logic programming approaches.

References

1. Y. Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.
2. M. Bergmann. *An Introduction to Many-Valued and Fuzzy Logic: Semantics, Algebras, and Derivation Systems*. Cambridge University Press, 2008.
3. T. R. Besold, A. d'Avila Garcez, G. Marcus, and R. Miikulainen (eds.). Cognitive computation: Integrating neural and symbolic approaches. Workshop at NIPS 2015, Montreal, Canada, April 2016.
4. D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.
5. A. d'Avila Garcez, M. Gori, P. Hitzler, and L. Lamb. Neural-symbolic learning and reasoning (dagstuhl seminar 14381). *Dagstuhl Reports*, 4(9):50–84, 2014.
6. M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
7. D. Kiela and L. Bottou. Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In *Proc. EMNLP 2014*, Doha, Qatar, 2014.
8. M. Richardson and P. Domingos. Markov logic networks. *Mach. Learn.*, 62(1-2):107–136, February 2006.
9. L. Serafini and A. d'Avila Garcez. Logic tensor networks: Deep learning and logical reasoning from data and knowledge. *CoRR*, abs/1606.04422, 2016.
10. R. Socher, D. Chen, C. Manning, and A. Ng. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In *NIPS 26*. 2013.