Complementing Logical Reasoning with Sub-Symbolic Commonsense

Federico Bianchi[1,2], Matteo Palmonari[1], Pascal Hitzler[2,3], and Luciano Serafini[4]

[1] University of Milan-Bicocca, Milan, Italy
[2] Wright State University, Dayton, OH, USA
[3] Kansas State University, Manhattan, KS, USA
[4] Fondazione Bruno Kessler, Trento, Italy
federico.bianchi@unimib.it, matteo.palmonari@unimib.it, pascal.hitzler@wright.edu, serafini@fbk.com

Abstract. Neuro-symbolic integration is a current field of investigation in which symbolic approaches are combined with deep learning ones. In this work we start from simple non-relational knowledge that can be extracted from text by considering the co-occurrence of entities inside textual corpora; we show that we can easily integrate this knowledge with Logic Tensor Networks (LTNs), a neuro-symbolic model. Using LTNs it is possible to integrate axioms and facts with commonsense knowledge represented in a sub-symbolic form in one single model performing well in reasoning tasks. In spite of some current limitations, we show that results are promising.

1 Introduction

Neuro-symbolic integration models [11, 12] aim at combining properties of symbolic reasoning and neural networks, to account both for data-driven learning and high-level reasoning, two tightly related aspects of human cognition. Additional advantages of this combination can be found in a higher explainability of learned knowledge and in the capability of softening some aspects of crisp logic-based reasoning approaches. This integration is also connected to the combination of sub-symbolic perception with high-level reasoning, a critical task in artificial intelligence [19].

Logic Tensor Networks (LTNs) [9, 24] are an example of a neuro-symbolic model that embeds first-order fuzzy logic in a vector space. In LTNs logic constants are represented as vectors and n-ary predicates are n-ary functions whose values are real numbers in the range $[0, 1]$. A neural network for each predicate learns both the representation of logic constants and the weights that characterize the n-ary function. Learning is based on a set of axioms.

On the other hand, computational linguistics has developed distributional models of language that have been found cognitively plausible at a large extent by psychologists [18]. We believe that these models, once adapted to be easily integrated with existing logical frameworks that combine learning and reasoning, can provide an account for commonsense knowledge and structured inferences that go beyond crisp reasoning approaches.

In this paper, we focus on the integration of two aspects of knowledge: (i) *sub-symbolic* common sense knowledge [17] that accounts for some kind of intuitive understanding of the world [7] and (ii) axiomatic knowledge has the one found in knowledge bases that accounts for structured inference. As an example of how this combination might work, imagine an agent that has access to the following set of axioms $\{species(cat), mammal(tiger), bird(penguin), \forall x(mammal(x) \rightarrow animal(x))\}$, we refer to the first three as instantiated atoms or facts and to the latter one as universally quantified formula; this axiomatic knowledge is not enough to infer $mammal(cat)$. However, if the agent knows that cats and tigers are similar to each other and both are dissimilar to penguins, she might infer that cats are mammals too (i.e., $mammal(cat)$). Once the latter instantiated atom has been inferred, the agent can make use of the axiom $\forall x(mammal(x) \rightarrow animal(x))$ to infer that cats are also animals (i.e., $animal(cat)$), bridging the gap with more complex inferences. We believe that combining these two worlds would bring great benefits in reasoning approaches since one requires the help of the other.

We present a first approach towards this direction that feeds Entity Embedding (EEs) generated using distributional semantics, i.e., vector-based representations of entities generated from text using Word2Vec [3], to a knowledge base represented in LTNs. This EEs encode the similarity between entities based on the principle that entities that share more contexts within a text corpus are more similar to each other. Distributional semantics has been found to provide representations that are strongly correlated with associative learning [18]; we thus refer to these representations as *sub-symbolic* commonsense. While in LTNs, neural representations of axioms are usually learned only from a partial (structured) knowledge base, EEs are used here as representations for the LTNs constants. In this way, LTNs will only need to learn the representation of the predicate network. Moreover, with the use of pre-trained representations, we can make inferences on entities that do not occur in the knowledge base as long as we have a sub-symbolic commonsense representation of those entities. Figure 1 shows the elements of our model that combines logical reasoning and sub-symbolic commonsense knowledge.

In once sentence, the major contribution of this work is to show that combining commonsense knowledge under the form of text-based entity embeddings with LTNs is not only simple, but it is also promising. Our experiments explore a limited part of a knowledge base but results show that the model is flexible and can be useful under different settings and use-cases.

The paper is organized as follows: in Section 2 we summarize related work and in Section 3 we outline the two main components of our model, namely the embeddings and LTNs and we show how we can combine the strengths of both; in Section 4 we develop and experiment comparing our model with some

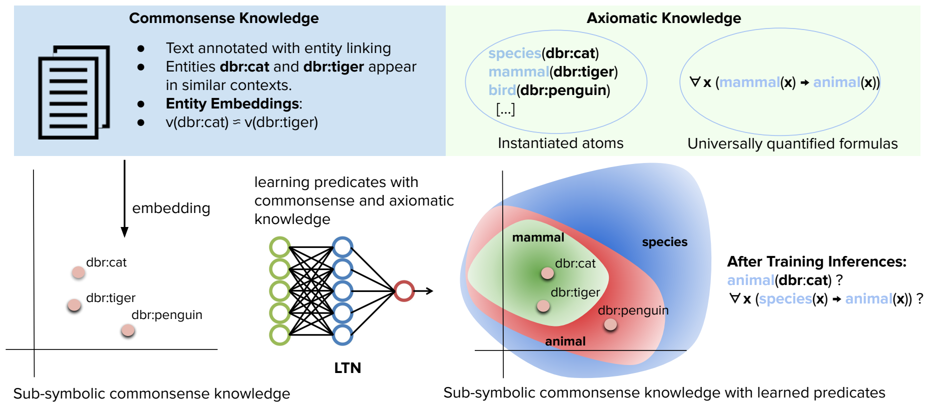
Commonsense Knowledge
Text annotated with entity linking
Entities dbr:cat and dbr:tiger appear in similar contexts.
Entity Embeddings:
v(dbr:cat) ≈ v(dbr:tiger)
Axiomatic Knowledge
species(dbr:cat)
mammal(dbr:tiger)
bird(dbr:penguin)
[...]
Instantiated atoms
∀x (mammal(x) → animal(x))
Universally quantified formulas
embedding
dbr:cat
dbr:tiger
dbr:penguin
learning predicates with commonsense and axiomatic knowledge
LTN
mammal
species
animal
After Training Inferences:
animal(dbr:cat) ?
∀x (species(x) → animal(x)) ?
Sub-symbolic commonsense knowledge
Sub-symbolic commonsense knowledge with learned predicates

Fig. 1. We learn embeddings from text and we use LTNs to learn how to represent predicates with the network. "dbr:" stands for the DBpedia Knowledge Graph namespace.

baselines; eventually we end the paper in Section 5 with conclusions and future work.

2 Related Work

While recently deep learning [13] has shown great capabilities in many different tasks its techniques are still limited when they have to take into account the same reasoning and knowledge transformation capabilities that symbolic approaches show. However, symbolic artificial intelligence is constrained by computational limits and knowledge acquisition bottlenecks. The neuro-symbolic field was introduced to address the limits of both approaches by at the same time taking advantage of the capabilities of each of them. In this section, we present recent works from both the symbolic/statistical relational learning and the neuro-symbolic fields.

Symbolic and Statistical Relational Learning Approaches. Different symbolic/statistical relational learning approaches have been devised to treat inference; Recently, ProbLog [8] has been proposed as a probabilistic logic programming language that can be used to combine probability and logical inference, allowing the user to treat both probabilistic uncertainty and classical inference. Another approach to inference is the one represented by Probabilistic Soft Logic (PSL) [1] that is a statistical relational learning model that comes from the family of Markov Logic Networks (MLNs) [20].

Neuro-symbolic Approaches. We refer to recent surveys for discussions of different neuro-symbolic approaches proposed in the literature [11, 12] while hereby we cite some examples of relevant approaches. DeepProbLog [19] is, for example, a "deep" extension of ProbLog [8], that show that it is possible to combine the power of deep nets with the expressive capabilities of logical reasoning. On the other hand, the Neural Theorem Prover (NTP) was introduced as an extension of the Prolog language that supports soft unification rules with the use of similarity between embedded representations [23]. Other approaches address reasoning with networks [25, 16] and transferability of reasoning with memory networks [10]. Different approaches have been defined to generate sub-symbolic representations of entities and relationships of a knowledge base [5, 26] and some of these integrate fuzzy logic in the process of learning these embedded representations [14].

We also report that the combination between distributional semantics and logic is not new and there is a recent line of research that is currently exploring *formal distributional semantics* [4].

We propose a method to complement logical reasoning in the vector space with sub-symbolic commonsense knowledge. We decided to focus on LTNs because the integration of sub-symbolic knowledge is straightforward and simpler to do with respect to other neuro-symbolic algorithms: LTNs gives us the advantage representing first-order logic inside a vectors space; at the same time we use entity embeddings to represent commonsense knowledge as the starting vector space on which LTNs learn to do reasoning.

3 Logical Reasoning with Sub-symbolic Commonsense

3.1 Logical Reasoning with Logic Tensor Networks

LTNs [9, 24] use first-order fuzzy logic and represent terms, functions, and predicates in a vector space. Connectives are interpreted as binary operations over real numbers in $[0, 1]$. For example, *t-norms* are used in place of the conjunction from classical logic (e.g., the t-norm can be interpreted as the min between two truth values). The action of representing elements of the logic language as elements in the vector space is referred to as grounding.

In LTNs, constants are grounded to vectors in $\mathbb{R}^n$ and predicates are grounded to neural network operations which output values in $[0, 1]$. The neural network learns to define the truthness of an atom $P(c_1, \ldots, c_n)$ as a function of the grounding of the terms $c_1, \ldots, c_n$ [24]. For a predicate of arity m and for which $\mathbf{v}_1, \ldots, \mathbf{v}_m \in \mathbb{R}^n$ are the groundings of m terms, the grounding of the predicate is defined as $\mathcal{G}(P)(\mathbf{v}) = \sigma(u_P^T(tanh(\mathbf{v}^T W_P^{[1:k]}\mathbf{v} + V_P\mathbf{v} + B_P)))$ where $\mathbf{v} = \langle \mathbf{v}_1, \ldots, \mathbf{v}_m \rangle$ represents the concatenation between the vectors $\mathbf{v}_i$, σ is the sigmoid function, W, V, B and u are parameters to be learned by the network while k is layer size of the tensor.

LTNs reduce the learning problem to a maximum satisfiability problem: the task is to find groundings for terms and predicates that maximize the satisfiability of the formulas in the knowledge base. For example, for a grounded formula like $mammal(cat)$, the network updates the representation of the predicate $mammal$ (i.e., the parameters in the tensor layer) and the representation of cat (i.e., its vector) in such a way that the degree of truth of an instantiated atom is closer to 1. Optimization also works in place of quantified formulas (e.g.,

$\forall x(mammal(x) \rightarrow animal(x))$; In fact, the universally quantified formulas are computed by using an aggregation operation [24] defined over a subset of the domain space $\mathbb{R}^n$. LTNs can be be used to do after-training reasoning over combinations of axioms on which it was not trained on (e.g., ask the truth value of queries like $\forall x(\neg mammal(x) \rightarrow species(x))$; this property allows us to explore the learned knowledge base with different combinations of predicates.

3.2 Sub-symbolic Commonsense with Entity Embeddings

Sub-symbolic representations of knowledge are popular in the deep learning community. In the NLP area, pre-trained representation of words (aka, word embeddings [21]) based on in-text co-occurrence are used frequently to enhance the performance on several tasks. In the same way, embeddings of entities and relationships that come from a knowledge base are becoming widely used in several contexts [5].

We use text-based embeddings of entities [3]. This approach is grounded in distributional semantics, originally introduced for words: similar words appear in similar context share similar meanings [15]; the same is true for entities [3], with two main advantages: (i) entities identifiers are not ambiguous and (ii) entities identifiers are interpreted as logical constants. The original work [3] presented also embeddings of ontological types, we ignore this component in our work since in this work we interpret entity types as unary predicates in LTNs. Starting from a text T, containing a sequence of words $w_1, \ldots, w_n$ we use entity linking tools [22] to find entities and to generate an annotated text that contains sequences of entity identifiers $e_1, \ldots, e_m$. The word2vec algorithm [21] is used to learn an embedding function ϕ based on the co-occurrence of entity mentions in the text $\phi(e_i) = \mathbf{e_i}$. Word2vec lets the user decide the dimension of the embedding and a window size to define the width of the context for each entity.

3.3 Combining Sub-symbolic Commonsense and Logical Reasoning

In our *commonsense* vector space, logical constants are represented by a vector and thus we can use LTNs to learn representations for the predicates over the *commonsense* vector space. Thus, we used the entity embeddings $\mathbf{e_1}, \ldots, \mathbf{e_m}$ as vectors to feed to LTNs. The truth value computed by LTNs is function not only of the parameters of the networks but also of the text-based pre-trained representations. While LTNs generally needs to learn the representation of vector from scratch in our setting are already learned (sub-symbolic commonsense) and do not need any more training.

Figure 1 shows a summary of the components of our model. We generate distributional embeddings from text and then we use axiomatic knowledge to learn the representations of predicates. After training we can use the model to reason over new axioms. A good way of understanding how LTNs work is to consider the learned predicate network as an area for which vectors have a truth value of 1 in certain locations that decreases to values close to 0 when vectors are distant from those locations.

4 Experiments

The main motivation that guides this experimental section is to show the capabilities of a model that combines logical reasoning and sub-symbolic commonsense knowledge like the one defined in the previous section.

We use 100 dimensional DBpedia entity embeddings [3]; these embeddings are generated first by using an automatic annotator (DBpedia Spotlight) and then by using word2vec (Skip-gram) [21]. LTNs were initialized with $k = 20$ and we used the fuzzy Lukasiewicz t-norm as in [24]. Code, data and architectures to replicate our experiments are available online[5].

4.1 Reference Knowledge Base

We create three small knowledge completion tasks for our experiments that are based on a common reference knowledge base introduced in this paper[6]. Our knowledge base is based on DBpedia and contains: a set of predicates P (e.g., $mammal$); a set of constants C (e.g., dbr:cat[7]); a set I of instantiated atoms, i.e., facts such as (e.g., $mammal$(dbr:cat)); a set Q of universally quantified formulas that represent the dependency in the DBpedia ontology (e.g., $\forall x\ mammal(x) \rightarrow animal(x)$); the set I^Q of formulas closed under the application of standard FOL inference to the previous set I (e.g., $animal$(dbr:cat)); a set of negated I^N instantiated atoms that are derived as follows: all the instantiated atoms built with predicates in P that are not in I^Q and I (e.g., $\neg fungus$(dbr:cat)). The reference knowledge base D is $I \cup I^Q \cup I^N$.

We first of all extract entities (C) from DBpedia and its ontology of the following classes (note that some classes are much less represented than others): Mammal (0.38%), Fungus (0.17%), Bacteria (0.03%), Plant (0.42%). We add the universally quantified formulas Q to derive inferences for predicates Animal, Eukaryote and Species for each atom, and apply this axiom to generate the set of instantiated axioms I^Q. Finally, we also generate all the negative instantiated atoms in I^N (e.g., $\neg fungus$(dbr:cat)). Considering positive and negative instantiated axioms this reference knowledge base contains 35,133 elements. We test the following three tasks by splitting the reference knowledge base D in training and testing:

[5] `https://github.com/vinid/logical_commonsense`

[6] Other knowledge base exist but some are too big to be explored [5] and others can be completed with simple axioms [6].

[7] We are aware that in some cases there is a subtle difference between what can be considered an instance and what is instead a type; *cat* can be for example the type of all the instances of cats. Since this generally depends on the granularity of the knowledge base we think that this does not affect the general applicability of the proposed experiment.

– **D1**. **Objective:** evaluate the performance of the algorithms in a task in which only positive atoms are given, not all the atoms can be influenced from the axioms. As training, we have 1,400 positive atoms and we ask the models to find all the other 7,077 atoms related to the entities found in the 1,400 atoms. For example, models have to infer atoms about the instance "dbr:cat" even if the only know that "dbr:cat" is a Species.
– **D2**. **Objective:** evaluate the performance of the algorithms in a task in which both positive and negative atoms are given; each entity in the training set appears also in the test set. As training, we have 7,026 atoms both positive and negatives and as in D1 we ask the models to find other 20,890 atoms (positive and negative).
– **D3**. **Objective:** evaluate the performance of the algorithms in a task in which both positive and negative atoms are given, but the test set will also contain atoms of entities not present in the training set: The models will need to rely on the sub-symbolic commonsense vectors. As training, we have 1,756 atoms and the models are now asked to infer the value of 33,377 atoms (positive and negative).

Domain Theory We define a set of universally quantified axioms to be used by the models that contains 22 axioms. The complete list is available online and we hereby show some of them.

– $\forall x(plant(x) \rightarrow eukaryote(x))$
– $\forall x(mammal(x) \rightarrow animal(x))$
– $\forall x(plant(x) \rightarrow \neg mammal(x))$
– $\forall x(fungus(x) \rightarrow \neg animal(x))$

Note that this set is different from the set Q: the models will not know for example that $\forall(x : animal(x) \rightarrow eukaryote(x))$.
Baseline. We will compare the LTN_{EE} model (trained over atoms and universally quantified formulas to reach 0.99 satisfiability over the input knowledge base) with the following competitors.

– Simple LTNs model not initialized with pre-trained embeddings. We use this model to show that the use of pre-trained representation is useful.
– Probabilistic Soft Logic [1], the main competitor for the symbolic field that will be trained on both atoms and universally quantified formulas. We use the tool provided by the original authors with default parameters[8].
– Deep Neural Network initialized with EEs trained to assign 0 or 1 to instantiated atoms (note that we cannot use universally quantified formulas here), we explored several architectures often obtaining similar results. The DNN referenced in the results embeds the pre-trained representations of entities and a one-hot representation of predicates in 20 dimensions, concatenate them and apply another transformation to 1 dimension plus a sigmoid as non-linearity. Validation is done on 20% of the input data. Note that DNN cannot make use of the axioms of the domain theory.

[8] `https://psl.linqs.org/`

4.2 Results

Table 1 shows the results of the different models over the different settings with the F1 measure for each predicate. In the following sections we discuss the result on each dataset.

Experiments on D1 In this setting we compare LTN_{EE} with LTN and PSL. We cannot use DNN because we are using only positive atoms. We also report that a simple rule-based model that uses axioms to complete the knowledge base would be able to infer only 45% of the axioms (with a 100% precision). The LTN_{EE} approach is the best performing one. The comparison between pure LTN and PSL suggests that while the latter performs better their difference is not high in this setting.

Experiments on D2 In this experiment each entity for which we require to find other instantiated atoms appear at least one time in the training set; this allows us to use PSL as a baseline in this setting. PSL performance is more or less similar to the one shown for the D1 dataset, but the performance between LTN_{EE} and DNN is comparable. In this settings the domain theory does not seem to provide increases in performance, but we remark that LTNs provide a model that can be queried after training.

Experiments on D3 From this experiment it is clear that LTN_{EE} generalizes slightly better than the competitor, and this could be due to both the domain theory and the fact that LTN_{EE} trains each predicate as a separate tensor layer. While the F1 score for many classes are comparable, the ones for Fungus and Bacteria reached a lower score than in the previous experiment: this might be due to the fact that the representation of elements of the class Fungus are similar to those of the class Plant while the Bacteria class as only a few instances in this experiment. Even if DNN and LTN_{EE} performances are similar (as expected, since both are neural models), we stress the fact that LTN_{EE} can be used for after-training logical inferences and this is a key aspect.

4.3 Examples and Limits

After training we can evaluate the truthfulness of axioms for which it was not specifically trained on. Table 2 reports some examples. We also explored the possibilities given by a more complex example that contains KG triples with facts *nationality(Person, Country)*, *bornIn(Person, City)* and *locatedIn(City, Country)* with 200 training examples (for which we also defined some simple axioms like $\forall x, \forall y, \forall z(bornIn(x, y) \wedge locatedIn(y, z) \rightarrow nationality(x, z))$ during training, but not the ones we show in the Table). It is interesting how LTNs can learn to reason on non-trivial axiomatic properties like the fact that being born in New York makes one American. The small experiment with KG triples is limited by the fact that the current implementation of LTNs suffers from heavy computational requirements in the presence of predicates in combination with quantifiers [2]. While the use of quantifiers extends the expressive power of the model it certainly downgrades the efficiency.

Table 1. $F1$ score per tested class.

D1	A_{F1}	F_{F1}	M_{F1}	P_{F1}	B_{F1}	E_{F1}	S_{F1}
LTN_{EE}	**0.81**	**0.74**	**0.84**	**0.66**	**0.52**	**0.97**	**1.00**
LTN	0.40	0.14	0.12	0.10	0.03	0.93	**1.00**
PSL	0.54	0.19	0.15	0.14	0.07	0.93	**1.00**
D2	A_{F1}	F_{F1}	M_{F1}	P_{F1}	B_{F1}	E_{F1}	S_{F1}
LTN_{EE}	0.91	**0.86**	0.91	0.86	**0.63**	**0.99**	**1.00**
DNN	**0.93**	0.82	**0.93**	**0.87**	0.54	**0.99**	1.00
PSL	0.56	0.20	0.20	0.17	0.10	0.88	0.98
D3	A_{F1}	F_{F1}	M_{F1}	P_{F1}	B_{F1}	E_{F1}	S_{F1}
LTN_{EE}	**0.88**	**0.80**	**0.89**	**0.82**	**0.60**	**0.99**	**1.00**
DNN	0.87	0.64	0.85	0.77	0.47	0.98	1.00

Table 2. The truth values of novel axioms.

Axiom	**Truth**
$\forall x(species(x) \rightarrow animal(x))$	0
$\forall x(eukaryote(x) \rightarrow \neg bacteria(x))$	0.73
$\exists x(eukaryote(x) \wedge \neg plant(x))$	1
$\forall x, y, z(nationality(x, y) \wedge locatedIn(y, z) \rightarrow bornIn(x, z))$	0.33
$\exists x(nationality(x, Canada) \wedge bornIn(x, Montreal))$	1
$\forall x(bornIn(x, New\ York) \rightarrow nationality(x, United\ States))$	0.88

5 Conclusions

In this paper, we have shown that the combination of sub-symoblic commonsense representations, under the form of entity embeddings generated from text, and logical reasoning in vector spaces is flexible and can be used to solve completion tasks. Since LTNs are based on Neural Networks, they reach similar results while also achieving high explainability due to the fact that they ground first-order logic. The real advantage comes from the fact that LTNs allow us to get the best of both the symbolic and connective worlds and to easily integrate additional knowledge like sub-symbolic commonsense knowledge. Despite the limitations and the simple experimental setting, the preliminary results show that the approach is promising. The key point of this paper is that with the combined model we can inject domain knowledge in a network (using LTNs) and at the same time use pre-trained representations. Our futures steps include improving LTNs training to treat bigger knowledge bases [5], introducing commmonsense knowledge within other frameworks and testing natural language inference tasks.

6 Acknowledgment

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Bach, S.H., Broecheler, M., Huang, B., Getoor, L.: Hinge-loss markov random fields and probabilistic soft logic. Journal of Machine Learning Research **18**, 1–67 (2017)
2. Bianchi, F., Hitzler, P.: On the capabilities of logic tensor networks for deductive reasoning. In: AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering (2019)
3. Bianchi, F., Palmonari, M., Nozza, D.: Towards encoding time in text-based entity embeddings. In: ISWC. pp. 56–71. Springer (2018)

4. Boleda, G., Herbelot, A.: Formal distributional semantics: Introduction to the special issue. Computational Linguistics **42**(4), 619–635 (2016)
5. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS. pp. 2787–2795 (2013)
6. Bouchard, G., Singh, S., Trouillon, T.: On approximate reasoning capabilities of low-rank vector spaces. AAAI Spring Syposium on Knowledge Representation and Reasoning (KRR): Integrating Symbolic and Neural Approaches (2015)
7. Chudnoff, E.: Intuitive knowledge. Philosophical Studies **162**(2), 359–378 (2013)
8. De Raedt, L., Kimmig, A.: Probabilistic (logic) programming concepts. Machine Learning **100**(1), 5–47 (2015)
9. Donadello, I., Serafini, L., d'Avila Garcez, A.: Logic tensor networks for semantic image interpretation. In: IJCAI. pp. 1596–1602 (2017)
10. Ebrahimi, M., Sarker, M.K., Bianchi, F., Xie, N., Doran, D., Hitzler, P.: Reasoning over RDF knowledge bases using deep learning. arXiv preprint arXiv:1811.04132 (2018)
11. Garcez, A.d., Gori, M., Lamb, L.C., Serafini, L., Spranger, M., Tran, S.N.: Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. arXiv preprint arXiv:1905.06088 (2019)
12. Garcez, A.S., Lamb, L.C., Gabbay, D.M.: Neural-symbolic cognitive reasoning. Springer Science & Business Media (2008)
13. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
14. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: EMNLP. pp. 192–202 (2016)
15. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
16. Hohenecker, P., Lukasiewicz, T.: Deep learning for ontology reasoning. arXiv preprint arXiv:1705.10342 (2017)
17. Kuipers, B.: On representing commonsense knowledge. In: Associative networks, pp. 393–408. Elsevier (1979)
18. Lenci, A.: Distributional semantics in linguistic and cognitive research. Italian journal of linguistics **20**(1), 1–31 (2008)
19. Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., De Raedt, L.: Deepproblog: Neural probabilistic logic programming. arXiv preprint arXiv:1805.10872 (2018)
20. Meza-Ruiz, I., Riedel, S.: Jointly identifying predicates, arguments and senses using markov logic. In: NAACL. pp. 155–163. ACL (2009)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS. pp. 3111–3119 (2013)
22. Rizzo, G., Troncy, R.: Nerd: a framework for unifying named entity recognition and disambiguation extraction tools. In: EACL. pp. 73–76. ACL (2012)
23. Rocktäschel, T., Riedel, S.: End-to-end differentiable proving. In: NIPS. pp. 3788–3800 (2017)
24. Serafini, L., Garcez, A.S.d.: Learning and reasoning with logic tensor networks. In: AI*IA. pp. 334–348. Springer (2016)
25. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: Advances in neural information processing systems. pp. 926–934 (2013)
26. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML. pp. 2071–2080 (2016)