Should All Semantic Web Reasoners Include a Neural Network?

Jeff Heflin[1,*,†], Yifan Zhang[1,†]

[1]Lehigh University, Computer Science and Engineering, 113 Research Dr., Bethlehem, PA, 18015

Abstract

This position poster makes the case that deep neural nets can be incorporated into traditional symbolic reasoners to enable reasoning at Web scale. This will maintain the correctness guarantees of the reasoner while offering efficiency by exploring fruitful paths first. We suggest a first-level decomposition of the problem into representation, learning mechanism, and control strategy and describe some preliminary results.

Keywords

scalable reasoning, neurosymbolic, knowledge base, deep learning

1. Introduction

Since the early days of Semantic Web research, one of the core questions has been how to perform inference at Web scale. For example, the Scalable Semantic Web Knowledge Base Systems workshop has been held 13 times since 2005. In recent years, deep learning has been used to find solutions to problems that were previously unsolvable. This position poster asks whether deep learning could be a vital component in research into scalable reasoning.

Neural networks are capable of quickly making predictions from large data, are robust to noise (such as exceptions and contradictions) and can capture knowledge that is difficult to describe formally. However, it is difficult to explain how the trained model came to its conclusion, requires very careful curation of the training set to avoid biases, and the predictions can be very wrong when encountering the long-tail of inputs seldom seen during training.

Due to the opaqueness and unpredictability on out-of-distribution data, we do not suggest that all reasoning be performed by a deep neural net. To the contrary, we propose to build traditional symbolic systems that use machine learning models in the decision-making process. Generally, all machine reasoning is a search for a proof, and at each step, the system has many options to choose from. In classical reasoning systems, there is often nothing to distinguish the options, so the choice amounts to a random guess. If the guess turns out to be wrong, and leads to a dead-end, the reasoner will back up and try a different choice. Deep learning can be used to train a meta-reasoner that will use knowledge about the question at hand and the

Posters, Demos, and Industry Tracks at ISWC 2024, November 13–15, 2024, Baltimore, USA
*Corresponding author.
†These authors contributed equally.
heflin@cse.lehigh.edu (J. Heflin); yiz521@lehigh.edu (Y. Zhang)
http://www.cse.lehigh.edu/~heflin/ (J. Heflin)
0000-0002-7290-1495 (J. Heflin); 0009-0002-2219-0514 (Y. Zhang)

structure of the axioms, in order to make more informed choices. In essence, the meta-reasoner will function as “intuition,” guiding the reasoner to a solution with fewer false starts. Since the reasoner takes discrete steps, each of which has a justification, the result is reliable and explainable. Admittedly, this is not a new idea. However, we have two reasons to believe that success is possible where previous approaches have failed: first, our proposal for a new way to represent logical statements in vectors that capture their logical semantics has shown early promise [1, 2]. Second, new deep learning architectures are pushing the boundaries of what problems can be solved by neural networks.

There are two key research areas that are relevant to this topic: neurosymbolic AI and learning for automated theorem provers. Neurosymbolic AI seeks “to integrate neural network-based methods with symbolic knowledge-based approaches” [3]. This can include a broad range of topics from generating embeddings of knowledge graphs to training a neural network to predict whether one logical statement entails another.

In the field of automated theorem proving, there has been various work on proof guidance. Wang et al. [4] proposed to use Graph Convolutional Networks to identify which mathematical statements were relevant to a given conjecture. Jakubův and Urban’s ENIGMA is a learning-based method to train a classification model to identify “*useful* and *un-useful*” clauses for proof search [5]. Crouse et al. combine various embedding strategies, including from ENIGMA, with a deep reinforcement learning approach to proof search [6]. More recently, Google has combined large language models and symbolic reasoners to enable automated proving of several Olympiad-level geometry problems [7].

2. Challenges

We propose to augment a traditional symbolic reasoner with a learned meta-reasoning component. In order to train the meta-reasoner, there are three main problems that must be addressed : representation, learning mechanism and control strategy. Representation is the determination of how to express symbolic information in a form that can be input to a neural architecture since such architectures require inputs to be vectors of numbers. There are many possible choices, and the ability of the choice to capture the semantics of the input can have significant impact on neural reasoning. The learning mechanism is the approach by which we train the model(s) used by the reasoner to help guide decisions for future queries. It considers the type of learning, the architecture of the neural model, and for supervised learning, how the training data is produced. Control strategy determines how the knowledge is utilized. In particular, the control strategy dictates which choices the algorithm will make, and when these choices will be made.Let us consider each of these challenges in turn.

2.1. Representation

Crucial to the problem of creating an intuitive symbolic reasoner is the choice of good **representations** for the logical statements. There are many ways to translate a sequence of symbols into a vector of numbers, but not all of them are useful. The objective is to find a representation that places logically similar statements close in vector space; thus similar statements will have similar outcomes.

We have had initial success by training an embedding using triplet loss to place atoms that unify closer together [1]. Representations for more complex statements can be composed from the embeddings of their constituent atoms. For example, to create the representation of a Horn Logic rule, you can sum the embeddings of the atoms in the body, and concatenate that with the embeddings of the head. Such a representation matches our objective well: if two atoms different only by a variable renaming, their representations should be close; if two rules differ only by the order of their body, their representations will be identical.

Designing a representation for a fundamental unit like an atom is appealing. This could become the building block for representations of more complex statements, much in the same way that Word2Vec is used as an initial representation for words by some Large Language Models. Several strategies might be used to create the more complex representations: such as long short-term memory networks, graph convolutional networks, or transformers.

2.2. Learning Mechanism

After the representation question has been solved, the next question is **how to train** the intuitive component. The first choice is between supervised learning and reinforcement learning. For early explorations, supervised learning has the advantage that one can more consistently compare different alternatives, but does require the collection of suitable training data. In general, training data can be produced by using traditional reasoning algorithms to answer a suite of queries. There are different types of functions that we can try to predict: e.g., given a choice, how many steps will be required to reach a solution vs. what is the likelihood that a solution will be found? Different reasoning algorithms will require different choices. For example, in backward-chaining, at each step the algorithm has to choose which subgoal to attempt next, and then which rule to apply to that subgoal. In the presence of cycles, a poor choice can lead to needless search or even infinite loops. We can train one or more scoring models to rank the choices. We can produce the training data automatically by running traditional reasoners (with randomized choices) on random queries. We have had initial success by using the scoring model to choose the hardest goal first, and then try to prove it using the most promising rule first.

Alternatively, reinforcement learning (RL) provides a strong framework that can replace supervised learning, especially when feedback (rewards or penalties) guides decisions. In RL, the reasoner acts as an agent making decisions step-by-step, such as choosing sub-goals or rules. Unlike supervised learning, RL doesn't need pre-labeled data; the agent learns through attempts to answer queries, getting feedback on whether a solution is found and how efficiently. Each decision is treated as an action, with rewards based on minimizing steps or avoiding loops. Over time, the RL agent refines its strategy to maximize overall rewards.

2.3. Control Strategy

The final piece is the **control strategy**: how will the trained intuitive component impact the reasoning process? The specific options will depend on the reasoning algorithm, but in all cases we can imagine a spectrum of strategies, At one extreme, the valid options are determined by the reasoner and the meta-reasoner merely ranks these choices. At the other extreme, the meta-reasoner determines both the available options and the order in which to evaluate them. When the design approaches this extreme, we go from intuition that merely makes traditional reasoners more efficient to intuition that enables new capabilities such as making intuitive leaps and selectively ignoring contradictory information.

3. Preliminary Results

We conducted a preliminary study using the Lehigh University Benchmark (LUBM), designed to evaluate the performance of semantic web databases in simulated universities, to assess our approach. In 62% of queries, our proposed system reduced the search space by several orders of magnitude compared to a traditional backward-chaining reasoner. Depending on query complexity, it found an answer by exploring between 5x to 60x fewer nodes.

4. Conclusion

If the three core problems can be solved, then this approach could become an essential component to any scalable knowledge base system. Future work includes extending the approach to description logics and tableau-based reasoning, finding efficient ways to encode statements in knowledge bases that have billions of constants, how to learn models that are still useful as the knowledge base evolves, and how best to avoid outliers.

References

[1] A. Arnold, J. Heflin, Learning a more efficient backward-chaining reasoner, in: Tenth Annual Conference on Advances in Cognitive Systems (ACS-2022), Cognitive Systems Foundation, Arlington, VA, 2022.

[2] Y.-B. Jia, G. Johnson, A. Arnold, J. Heflin, An evaluation of strategies to train more efficient backward-chaining reasoners, in: Proceedings of the 12th Knowledge Capture Conference 2023, 2023, pp. 206–213.

[3] A. Sheth, K. Roy, M. Gaur, Neurosymbolic AI- why, what, and how, arXiv preprint arXiv:2305.00813 (2023).

[4] M. Wang, Y. Tang, J. Wang, J. Deng, Premise selection for theorem proving by deep graph embedding, in: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 2783–2793.

[5] J. Jakubův, J. Urban, Enigma: Efficient learning-based inference guiding machine, in: H. Geuvers, M. England, O. Hasan, F. Rabe, O. Teschke (Eds.), Intelligent Computer Mathematics, Springer International Publishing, Cham, 2017, pp. 292–302.

[6] M. Crouse, I. Abdelaziz, B. Makni, S. Whitehead, C. Cornelio, P. Kapanipathi, K. Srinivas, V. Thost, M. Witbrock, A. Fokoue, A Deep Reinforcement Learning Approach to First-Order Logic Theorem Proving, 35th AAAI Conf. on Artificial Intelligence, AAAI 2021 7 (2021) 6279–6287. arXiv:1911.02065.

[7] T. Trinh, Y. Wu, Q. Le, H. He, L. Thng, Solving olympiad geometry without human demonstrations, Nature 625 (2024) 476–482. doi:10.1038/s41586-023-06747-5.