On the Generalization Capability of Memory Networks for Reasoning

Monireh Ebrahimi [1] **Md Kamruzzaman Sarker** [1] **Federico Bianchi** [1 2] **Ning Xie** [1] **Aaron Eberhart** [1]
Derek Doran [1] **Pascal Hitzler** [1]

Abstract

A significant and recent development in neural-symbolic learning are deep neural networks that can reason over symbolic knowledge bases (KBs) and perform scalable reasoning tasks. Initial neural-symbolic systems that can deduce the entailment of a KB have been presented, but they are theoretically and practically limited: current systems learn fact relations and entailment patterns specific to a particular KB and hence do not truly learn to reason, and must be retrained for each KB they are tasked with entailing. To address this generalization limitation, we propose a differentiable end-to-end deep memory network that learns over abstract, generic symbols to discover entailment patterns common to any reasoning task. A key component of the system is a simple but highly effective normalization process for continuous representation learning of KB entities within memory networks. Our results show how the model, trained over a set of KBs, can effectively entail facts from test KBs, even when the domain of test KBs is completely different from the training KBs.

1. Background

With the recent revival of interest in artificial neural networks, they have been applied vastly for the completion of KBs. These methods (Chang et al., 2014; Nickel et al., 2012; Riedel et al., 2013; Socher et al., 2013; Toutanova et al., 2015; Trouillon et al., 2016; Yang et al., 2014) heavily rely on the subsymbolic representation of entities and relations learned through maximization of a scoring objective function over valid factual triples. Thus, the current success of such models hinges primarily on the power of those subsymbolic continuous real-valued representations in encoding the similarity/relatedness of entities and relations. Recent attempts have focused on neural multi-hop reasoners (Das et al., 2016; Neelakantan et al., 2015; Peng et al., 2015; Shen et al., 2017; Weissenborn, 2016) to equip the model to deal with more complex reasoning. More recently, a Neural Theorem Prover (Rocktäschel & Riedel, 2017) has been proposed in an attempt to take advantage of both symbolic and sub-symbolic reasoning.

Despite their success, the main restriction common to neural reasoners is that they are unable to generalize to new domains. This inherent limitation follows from both the representation functions used and the learning process. The major issue comes from the mere reliance of these models on the representation of entities learned during the training or in the pre-training phase stored in a lookup table. Consequently, these models have difficulty to deal with out-of-vocabulary(OOV) entities. Although the small-scale OOV problem has been addressed in part in the natural language processing (NLP) domain by taking advantage of character-level embedding (Ling et al., 2015), learning embeddings on the fly by leveraging text descriptions or spelling (Bahdanau et al., 2017), copy mechanism (Eric & Manning, 2017) or pointer networks (Raghu et al., 2018), still these solutions are insufficient for transferring purposes. (Talman & Chatzikyriakidis, 2018) shows the success of natural language inference (NLI) methods is heavily benchmark specific. An even greater source of concern is that reasoning in most of the above sub-symbolic approaches hinges more on the notion of similarity and geometric-based proximity of real-valued vectors (induction) as opposed to performing transitive reasoning (deduction) over them. In short, to the best of our knowledge, to date, there is no sub-symbolic reasoning work, which is able to transfer the learning capability from one KB to unseen one. In fact, since previous works have focused to conduct reasoning on the unseen part of the same KB, they have tried to gain generalization ability through induction and robustness to missing edges(Guu et al., 2015) as opposed to deduction. Likewise, recent years have seen some progress in zero-shot relation learning in sub-symbolic reasoning domain(Neelakantan et al., 2015; Rocktäschel et al., 2015; Xiong et al., 2017). Zero-shot learning refers to the ability of the model to infer new relations where that relation has not been seen before in

[1] Department of Computer Science & Engineering, Wright State University [2] University of Milan - Bicocca. Correspondence to: Monireh Ebrahimi <ebrahimi.2@wright.edu>.

ICML 2019 Workshop on Understanding and Improving Generalization in Deep Learning, Long Beach, California, 2019. Copyright 2019 by the author(s).

training set(Bordes et al., 2011). This generalization capability is still quite limited and fundamentally different from our work in terms of both methodology and purpose.

Inspired by these observations, we take a different approach by investigating the emulation of deductive symbolic reasoning using memory networks. Memory networks (Weston et al., 2014) are a class of learning models capable of conducting multiple computational steps over an explicit memory component before returning an answer. They have been recently applied successfully to a range of NLP tasks such as question answering (Hill et al., 2015; Sukhbaatar et al., 2015), language modeling (Sukhbaatar et al., 2015), and dialogue tasks (Bordes et al., 2016; Dodge et al., 2015). End-to-end memory networks (MemN2N) (Sukhbaatar et al., 2015) are a less-supervised, more general version of these networks, applicable to the settings where labeled supporting memories are not available. We have selected such networks since we believe that they are a primary candidate to perform well for deductive logical entailment. Their sequential nature corresponds, conceptually, to the sequential process underlying some deductive reasoning algorithms. The attention modeling corresponds to pulling only relevant information (logical axioms) necessary for the next reasoning step. And their success in NLI is also promising: while NLI does not follow a formal logical semantics, logical deductive entailment is nevertheless akin to some aspects of natural language reasoning. Besides, as attention can be traced over the run of a memory network, we will furthermore get insights into the ”reasoning” underlying the network output, as we will be able to see which pieces of the input KB are taken into account at each step.

The main contribution of this paper, however, is a recipe involving a simple but effective KB triple normalization before learning their representation within a MemN2N. To perform logical inference in more abstract level, and thereby facilitating the transfer of reasoning expertise from one KB to another, the normalization maps entities and predicates in a knowledge to a generic vocabulary. Facts in additional KBs are normalized using the same vocabulary, so that the network does not learn to overfit its learning to entity and predicate names in a specific KB. This emulates symbolic reasoning by neural embeddings as the actual names (as strings) of entities from the underlying logic such as variables, constants, functions, and predicates are insubstantial for logical entailment in the sense that a consistent renaming across a theory does not change the set of entailed formulas (under the same renaming). Thanks to the term-agnostic feature of our representation, we are able to create a reasoning system capable of performing reasoning over an unseen set of vocabularies in the test phase.

2. Problem Formulation

We wish to train a neural model that will learn to reason over one set of theories, and can then transfer that learning to new theories over the same logic. One of the key obstacles we face with our task is to understand how to represent training and test data. To use standard neural approaches, symbols will have to be represented over the real coordinate space R as vectors (points), matrices or tensors. Many embeddings for KBs have been proposed (Bordes et al., 2013; Lin et al., 2015; Trouillon et al., 2016; Wang et al., 2014), but we are not aware of an existing embedding that captures what seems important for the deductive reasoning scenario. Indeed, the prominent use case explored for KB embeddings is not deductive in nature; rather, it concerns the problem of the discovery or suggestion of additional links or edges in the graph, together with appropriate edge labels. In this link discovery setting, the actual labels for nodes or edges in the graph, and as such their commonsense meanings, are likely important, and most existing embeddings reflect this. However, for deductive reasoning the names of entities are insubstantial and should not be captured by an embedding. Another inherent problem in the use of such representations across KBs is the OOV problem. While a word lookup table can be initialized with vectors in an unsupervised task or during training of the reasoner, it still cannot generate vector representations for unseen terms. It is further impractical to store the vectors of all words when vocabulary size is huge (Ling et al., 2015). Similarly, memory networks usually rely on word-level embedding lookup tables, i.e., learned with the underlying rationale that words that occur in similar supervised scenarios should be represented by similar vectors. That is why they are known to have difficulties dealing with OOV, as a word lookup table cannot provide a representation for the unseen, and thus has difficulty to do NLI over new words (Bahdanau et al., 2017), and for us this would pose a challenge in the transfer to new KBs.

We thus need representations that are agnostic to the terms used as primitives in the KB. To build such a representation, we use syntactic normalization: a renaming of primitives from the logical symbols to a set of predefined entity names that are used across different normalized theories. By randomly assigning the mapping for the renaming, the network's learning will be based on the structural information within the theories, and not on the actual names of the primitives. Note that this normalization not only plays the role of “forgetting” irrelevant label names, but also makes it possible to transfer learning from one KB to the other. Indeed, the network can be trained with many KBs, and then subsequently tested on completely new ones.

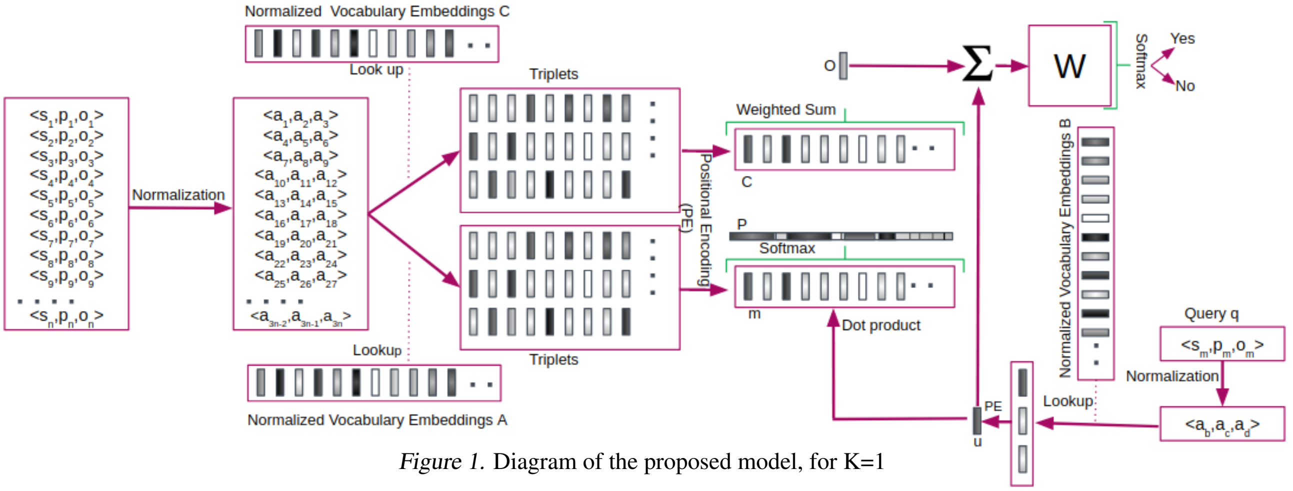

Figure 1. Diagram of the proposed model, for K=1

3. Model Architecture

We consider a model architecture that adapts the MemN2N with fundamental alterations necessary for abstract reasoning. A high-level view of our model is shown in Figure 1. It takes a discrete set G of normalized RDFS statements (called *triples*) $t_1, ..., t_n$ that are stored in memory, a query q, and outputs a "yes" or "no" answer to determine if q is entailed by G. Each of the normalized t_i and q contains symbols coming from a general dictionary with V normalized words shared among all of the normalized RDFS theories in both training and test sets. The model writes all triples to the memory and then calculates a continuous embedding for G and q. Through multiple hop attention over those continuous representations, the model then classifies the query. The model is trained by back-propagation of error from output to the input through multiple memory accesses. More Specifically, the model is augmented with an external memory that stores the embeddings of the normalized triples in our KB. This memory is defined as an $n \times d$ tensor where n denotes the number of triples in the KB and d is the dimensionality of the embeddings. The KB is stored in the memory vectors from two continuous representations of m_i and c_i obtained from two input and output embedding matrices of A and C with size $d \times V$ where V is the size of vocabulary. Similarly, the query q is embedded via a matrix B to obtain an internal state u. In each reasoning step, those memory slots useful for finding the correct answers should have their contents retrieved. To enable this, we use an attention mechanism for q over memory input representations by taking an internal product followed by a softmax:

$$p_i = \text{Softmax}(u^T(m_i)) \tag{1}$$

Equation (1)calculates a probability vector p over the memory inputs, the output vector o is computed as the weighted sum of the transformed memory contents c_i with respect to their corresponding probabilities p_i by $o = \sum_i p_i c_i$. This describes the computation within a single hop. The internal state of the query vector updates for the next hop as $u^{k+1} = u^k + o^k$. The process repeats K times where K is the number of computational hops. The output of the K^{th} hop is used to predict the label $\hat{a}$ by passing o^K and u^K through a weight matrix of size $V \times d$ and a softmax:

$$\hat{a} = \text{Softmax}(W(u^{K+1})) = \text{Softmax}(W(u^k + o^k)).$$

Figure 1 shows the model for $K = 1$. The parameters to be learned by backpropagation are A, B, C, and W matrices.

Memory Content An RDFS KB is a collection of statements stored as triples $(e1, r, e2)$ where $e1$ and $e2$ are called *subject* and *object*, respectively, while r is a relation binding $e1$ and $e2$ together. Every entity in an RDFS KB is represented by a unique Universal Resource Identifier (URI). We normalize these triples by systematically renaming all URIs which are not in the RDF or RDFS namespaces as discussed previously. Each such URI is mapped to a set of arbitrary strings in a predefined set $\mathcal{A} = \{a_1, ..., a_n\}$, where n is taken as a training hyper-parameter giving an upper bound for the largest number of entities in a KB the system will be able to handle. Note that URIs in the RDF/RDFS namespaces are not renamed, as they are important for the deductive reasoning according to the RDFS model-theoretic semantics. Consequently, each normalized RDFS KB will be a collection of facts stored as triples $\{(a_i, a_j, a_k)\}$.

It is important to note that each symbol is mapped into an element of $\mathcal{A}$ regardless of its position in the triple. Yet the position of an element within a triple is an important feature to consider. Thus we employ a positional encoding (PE) (Sukhbaatar et al., 2015) to encode the position of each element within the triple. Each memory slot thus represents the positional-weighted summation of each triplet. The PE ensures that the order of the elements now affects the encoding of each memory slot m_i.

Training Dataset	Test Dataset	Accuracy	
		Our model	Baseline
OWL-Centric	Linked Data	**96**	43
OWL-Centric(90%)	OWL-Centric (10%)	**90**	82
OWL-Centric	OWL-Centric Test Set *	**69**	61
OWL-Centric	Synthetic Data	**52**	48

* Completely Different Domain.

Table 1. Experimental results of the proposed model

4. Evaluation

Dataset We have collected RDFS datasets from the Linked Data Cloud[1] and the Data Hub[2].Our training set ("OWL-centric") is comprised of a set of RDFS KBs each of size 1,000 triples, sampled from populating around 20 OWL ontologies with different data. In order to test our model's ability to generalize to completely different datasets, we have collected another dataset called the OWL-Centric Test Set. To assure our evaluation represents real-world RDFS data completely independent of the training data, we have used almost all RDFS KBs listed in (Sam et al., 2018); we call this the Linked Data test set. Furthermore, to test the limitations of our model on artificially difficult data, we have created a small synthetic dataset which requires long reasoning chains if done with a symbolic reasoner. For each KB we have created the finite set of inferred triples using the Apache Jena[3] API. These inferred triples comprise our positive class instances. We generated non-inferred triples by random permutation of triple entities and removing those triples which were entailed.

Results Trainings were done over 10 epochs using the Adam optimizer with a learning rate of $\eta = 0.005$, a learning rate decay of $\eta/2$, and a batch size of 100 over triples. All embeddings are vectors of size 20. We have used $K = 10$. Adjacent weight sharing was used where the output embedding of one layer is the input embedding of the next one. All the weights are initialized by a Gaussian distribution with $\mu = 0$ and $\sigma = 0.1$. Here we report the average accuracy over all the KBs in the test set, obtained for both valid and invalid sets of triples. We have considered the non-normalized embedding version of our memory network as a baseline. Our technique shows a significant advantage over the baseline as shown in Table 1. A further even more important benefit of using our normalization model is its training time. In fact, this considerable time complexity difference is the result of the remarkable size difference of embedding matrices in the original and normalized cases. For instance, the size of embedding matrices to be learned by our algorithm for the normalized OWL-Centric dataset is $3,033 \times 20$ as opposed to $811,261 \times 20$ for the non-normalized one (and $1,974,062 \times 20$ for Linked Data which is prohibitively big). That has caused a remarkably high decrease in training time and space complexity. In case of the OWL-Centric dataset, for instance, the space required for saving the normalized model is 80 times less than the intact model. Likewise, the normalized model is almost 40 times faster to train than the non-normalized one for this dataset. Hence, the importance of using normalization cannot be emphasized enough.

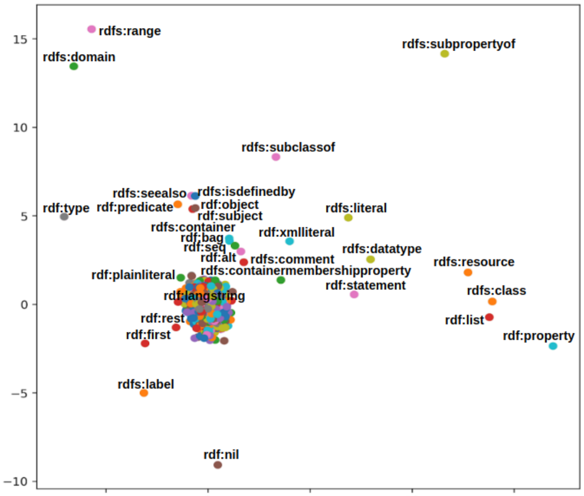

Figure 2. PCA projection of embeddings for the whole vocabulary

General Embeddings Visualization We have plotted a Principal Component Analysis (PCA) two-dimensional vector visualization of embeddings computed for the RDF(S) terms and all normalized words in the KBs, in Figure 2. The embeddings were fetched from the matrix B (embedding query lookup table) in the hop 1 of our model trained over the OWL-Centric dataset. Words are positioned in the plot based on the similarity of their embedding vectors. As anticipated, all the normalized words tend to form one cluster as opposed to multiple ones. The PCA projection illustrates the ability of our model to automatically organize RDF(S) concepts and learn implicitly the relationships between them. For instance, rdfs:domain and rdfs:range have been located very close together and far from normalized entities. rdf:subject, rdf:predicate and rdf:object vectors are very similar, and the same for rdf:seesAlso and rdf:isDefinedBy. Likewise, rdfs:container, rdf:bag, rdf:seq, and rdf:alt are in the vicinity of each other.

5. Conclusions and Future Work

We have demonstrated that a deep learning architecture based on memory networks and pre-embedding normalization is capable of learning how to perform deductive reasoning over previously unseen RDFS KBs with high accuracy. We believe that we have thus provided the first deep learning approach that is capable of high accuracy RDFS deductive reasoning over previously unseen KBs. Normalization appears to be a critical component for high performance of our system. This obviates the need for supervised retraining over the task of interest or unsupervised pretraining over the external source of data for learning the representations when encountered with a new KB. It also provides insights into representation learning for rare or OOV words, transfer learning, zero-shot learning, and domain adaptation in the reasoning domain. We plan to properly investigate scalability of our approach and to adapt it to other, more complex, logics.

[1] https://lod-cloud.net/

[2] https://datahub.io/

[3] https://jena.apache.org/

References

Bahdanau, D., Bosc, T., Jastrzebski, S., Grefenstette, E., Vincent, P., and Bengio, Y. Learning to compute word embeddings on the fly. *arXiv preprint arXiv:1706.00286*, 2017.

Bordes, A., Weston, J., Collobert, R., Bengio, Y., et al. Learning structured embeddings of knowledge bases. In *AAAI*, volume 6, pp. 6, 2011.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pp. 2787–2795, 2013.

Bordes, A., Boureau, Y.-L., and Weston, J. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*, 2016.

Chang, K.-W., Yih, S. W.-t., Yang, B., and Meek, C. Typed tensor decomposition of knowledge bases for relation extraction. 2014.

Das, R., Neelakantan, A., Belanger, D., and McCallum, A. Chains of reasoning over entities, relations, and text using recurrent neural networks. *arXiv preprint arXiv:1607.01426*, 2016.

Dodge, J., Gane, A., Zhang, X., Bordes, A., Chopra, S., Miller, A., Szlam, A., and Weston, J. Evaluating prerequisite qualities for learning end-to-end dialog systems. *arXiv preprint arXiv:1511.06931*, 2015.

Eric, M. and Manning, C. D. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *arXiv preprint arXiv:1701.04024*, 2017.

Guu, K., Miller, J., and Liang, P. Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094*, 2015.

Hill, F., Bordes, A., Chopra, S., and Weston, J. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301*, 2015.

Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, volume 15, pp. 2181–2187, 2015.

Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., Black, A. W., and Trancoso, I. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096*, 2015.

Neelakantan, A., Roth, B., and McCallum, A. Compositional vector space models for knowledge base completion. *arXiv preprint arXiv:1504.06662*, 2015.

Nickel, M., Tresp, V., and Kriegel, H.-P. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pp. 271–280. ACM, 2012.

Peng, B., Lu, Z., Li, H., and Wong, K.-F. Towards neural network-based reasoning. *arXiv preprint arXiv:1508.05508*, 2015.

Raghu, D., Gupta, N., et al. Hierarchical pointer memory network for task oriented dialogue. *arXiv preprint arXiv:1805.01216*, 2018.

Riedel, S., Yao, L., McCallum, A., and Marlin, B. M. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 74–84, 2013.

Rocktäschel, T. and Riedel, S. End-to-end differentiable proving. In *Advances in Neural Information Processing Systems*, pp. 3788–3800, 2017.

Rocktäschel, T., Singh, S., and Riedel, S. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1119–1129, 2015.

Sam, S., Hitzler, P., and Janowicz, K. On the quality of vocabularies for linked dataset papers published in the semantic web journal. *Semantic Web*, 9(2):207–220, 2018.

Shen, Y., Huang, P.-S., Gao, J., and Chen, W. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1047–1055. ACM, 2017.

Socher, R., Chen, D., Manning, C. D., and Ng, A. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pp. 926–934, 2013.

Sukhbaatar, S., Weston, J., Fergus, R., et al. End-to-end memory networks. In *Advances in neural information processing systems*, pp. 2440–2448, 2015.

Talman, A. and Chatzikyriakidis, S. Testing the generalization power of neural network models across nli benchmarks. *arXiv preprint arXiv:1810.09774*, 2018.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1499–1509, 2015.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pp. 2071–2080, 2016.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 14, pp. 1112–1119, 2014.

Weissenborn, D. Separating answers from queries for neural reading comprehension. *arXiv preprint arXiv:1607.03316*, 2016.

Weston, J., Chopra, S., and Bordes, A. Memory networks. corr abs/1410.3916, 2014.

Xiong, W., Hoang, T., and Wang, W. Y. Deeppath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690*, 2017.

Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.