
RelNet: End-to-end Modeling of Entities & Relations

Trapit Bansal, Arvind Neelakantan, Andrew McCallum
 University of Massachusetts Amherst
 {tbansal, arvind, mccallum}@cs.umass.edu

Abstract

We introduce RelNet: a new model for relational reasoning. RelNet is a memory augmented neural network which models entities as abstract memory slots and is equipped with an additional *relational memory* which models relations between all memory pairs. The model thus builds an abstract knowledge graph on the entities and relations present in a document which can then be used to answer questions about the document. It is trained end-to-end: only supervision to the model is in the form of correct answers to the questions. We test the model on the 20 bAbI question-answering tasks with 10k examples per task and find that it solves all the tasks with a mean error of 0.3%, achieving 0% error on 11 of the 20 tasks.

1 Introduction

Reasoning about entities and their relations is an important problem for achieving general artificial intelligence. Often such problems are formulated as reasoning over graph-structured representation of knowledge. Knowledge graphs, for example, consist of entities and relations between them [1, 2, 3, 4]. Representation learning [5, 6, 7, 8] and reasoning [9, 10, 11, 12] with such structured representations is an important and active area of research.

Most previous work on knowledge representation and reasoning relies on a pipeline of natural language processing systems, often consisting of named entity extraction [13], entity resolution and coreference [14], relationship extraction [5], and knowledge graph inference [15]. While this cascaded approach of using NLP systems can be effective at reasoning with knowledge bases at scale, it also leads to a problem of compounding of the error from each component sub-system. The importance of each of these sub-component on a particular downstream application is also not very clear.

For the task of question-answering, we instead make an attempt at an end-to-end approach which directly models the entities and relations in the text as memory slots. While incorporating existing knowledge (from curated knowledge bases) for the purpose of question-answering [12, 9, 16] is an important area of research, we consider the simpler setting where all the information is contained within the text itself – which is the approach taken by many recent memory based neural network models [17, 18, 19, 20].

Recently, Henaff et. al. [18] proposed a dynamic memory based neural network for implicitly modeling the state of entities present in the text for question answering. However, this model lacks any module for relational reasoning. In response, we propose RelNet, which extends memory-augmented neural networks with a relational memory to reason about relationships between multiple entities present within the text. Our end-to-end method reads text, and writes to both memory slots and edges between them. Intuitively, the memory slots correspond to entities and the edges correspond to relationships between entities, each represented as a vector. The only supervision signal for our method comes from answering questions on the text.

We demonstrate the utility of the model through experiments on the bAbI tasks [19] and find that the model achieves smaller mean error across the tasks than the best previously published result [18] in the 10k examples regime and achieves 0% error on 11 of the 20 tasks.

2 RelNet Model

We describe the RelNet model in this section. The model is sequential in nature, consisting of the following steps: read text, process it into a dynamic relational memory and then attention conditioned on the question generates the answer. We model the dynamic memory in a fashion similar to Recurrent Entity Networks [18] and then equip it with an additional relational memory.

There are three main components to the model: 1) input encoder 2) dynamic memory, and 3) output module. We will describe these three modules in details. The input encoder and output module implementations are similar to the Entity Network [18] and main novelty lies in the dynamic memory. We describe the operations executed by the network for a single example consisting of a document with T sentences, where each sentence consists of a sequence of words represented with K -dimensional word embeddings $\{e_1, \dots, e_N\}$, a question on the document represented as another sequence of words and an answer to the question.

Input Encoder: The input at each time point is a sentence from the document which can be encoded into a fixed vector representation using some encoding mechanism, such as a recurrent neural network. We use a simple encoder with a learned multiplicative mask [18, 17]: $s_t = \sum_i m_i \odot e_i$.

Dynamic Relational Memory This is the main component of an end-to-end reasoning pipeline, where we need to process the information contained in the text such that it can be used to reason about the entities, their properties and the relationships among them. The memory consists of two parts: entity memory and relational memory. The entity memory is organized as a key-value memory network [12], where the keys are global embeddings updated during training time but not during inference, and the value memory slot is a dynamic memory for each example (document, question) whose values are updated while reading the document. The memory thus consists of D memory slots $\{m_1, \dots, m_D\}$ (each is a vector of dimension K) and associated keys $\{k_1, \dots, k_D\}$ (again vectors of dimension K). At time t , after reading the sentence t into a vector representation s_t , a gating mechanism decides the set of memories to be updated ($\langle \cdot, \cdot \rangle$ denotes inner product):

$$g_i^m \leftarrow \sigma(\langle s_t, m_i + k_i \rangle) \quad (1)$$

Intuitively the memory slots can be thought of as entities. Indeed, Henaff et. al. [18] found that if they tie the key vectors to entities in the text then the memories contain information about the state of those entities. The update in (1) essentially does a soft selection of memory slots based on cosine distance in the embedding space. Note that there can be multiple entities in a sentence hence a sigmoid operation is more suitable, and it is also more scalable [18]. After selecting the set of memories, there is an update step which stores information in the corresponding memory slots:

$$\begin{aligned} \tilde{m}_j &\leftarrow \text{PReLU}(U m_j + V k_j + W s_t) \\ m_j &\leftarrow m_j + g_j^m \odot \tilde{m}_j \end{aligned} \quad (2)$$

where PReLU is a parametric Rectified linear unit [21], and U , V and W are $k \times k$ parameter matrices.

Now we augment the model with additional *relational memory cells*. Intuitively, the entity memory allows modeling of entities and information about the entities in isolation. This can be insufficient in scenarios where a particular entity participates in many relations with other entities across the document. Thus, in order to succeed at relational reasoning the model needs to be able to compare each pair of the entity memories. The relational memories will allow modeling of these relations and provide an inherent inductive bias towards a more structured representation of the participating entities in the text, in the form a latent knowledge graph. The relational memories are D^2 memory slots $\{r_{ij}\}$ indexed by the entity memory slots $i, j \in D$.

The relational memories are updated as follows. First, a gating mechanism decides the set of active relational memories:

$$g_{ij}^r \leftarrow g_i^m g_j^m \sigma(\langle s_t, r_{ij} \rangle) \quad (3)$$

where g_i^m, g_j^m select the relational memory slot based on the active entity slots and the last sigmoid gate decides whether the corresponding relational memory needs to be updated based on the current input sentence. After selecting the set of active relational memory, we update the contents of the relational memory:

$$\begin{aligned}\tilde{r}_{ij} &\leftarrow PReLU(Ar_{ij} + Bs_t) \\ r_{ij} &\leftarrow r_{ij} + g_{ij}^r \odot \tilde{r}_{ij}\end{aligned}\tag{4}$$

where again A, B are $k \times k$ parameter matrices. Note that for updates (3)–(4) we use a different encoding mask to obtain the sentence representation for relations.

Similar to [18], we normalize the memories after each update step (that is after reading each sentence). This acts as a forget step and does not cause the memory to explode.

The full memory consists of the entity memory slots $\{h_j\}$ and the relational memory slots $\{r_{ij}\}$.

Output Module This is a standard attention module used in memory networks [17, 18]. The question is encoded as a K dimensional vector q using the same encoding mechanism as the sentences (though with a separate learned mask). We first concatenate the relational memory vectors with the corresponding entity vectors, and project the resulting memory vector to k dimension. Then attention on these projected memories, conditioned on the vector q , yields the final answer:

$$\begin{aligned}m_{ij}^f &= C[m_i; m_j; r_{ij}] \\ p_{ij} &= Softmax(\langle q, m_{ij}^f \rangle) \\ u &= \sum_{ij} p_{ij} m_{ij}^f \\ o &= PReLU(q + Hu) \\ y &= Zo\end{aligned}$$

where y is the predicted answer, and C, H, Z are parameter matrices.

3 Related Work

There is a long line of work in textual question-answering systems [22, 23]. Recent successful approaches use memory based neural networks for question answering [24, 19, 25, 20, 18]. Our model is also a memory network based model and is also related to the neural turing machine [26]. As described previously, the model is closely related to the Recurrent Entity Networks model [18] which describes an end-to-end approach to model entities in text but does not directly model relations. Other approaches to question answering use external knowledge, for instance external knowledge bases [27, 12, 28, 29, 10] or external text like Wikipedia [30, 31].

Very recently, and in parallel to this work, a method for relational reasoning called relation networks [32] was proposed. They demonstrated that simple neural network modules are not as effective at relational reasoning and their proposed module is similar to our model. However, relation network is not a memory-based model and there is no mechanism to read and write relevant information for each pair. Moreover, while their approach scales as the square of the number of sentences, our approach scales as the square of the number of memory slots used per QA pair. The output module in our model can be seen as a type of relation network.

Representation learning and reasoning over graph structured data is also relevant to this work. Graph based neural network models [33, 34, 35] have been proposed which take graph data as an input. The relational memory however does not rely on a specified graph structure and such models can potentially be used for multi-hop reasoning over the relational memory.

4 Experiments

We evaluate the model’s performance on the bAbI tasks [19], a collection of 20 question answering tasks which have become a benchmark for evaluating memory-augmented neural networks. We

Task	EntNet [18]	RelNet
1: 1 supporting fact	0	0
2: 2 supporting facts	0.1	0.7
3: 3 supporting facts	4.1	3.4
4: 2 argument relations	0	0
5: 3 argument relations	0.3	0.6
6: yes/no questions	0.2	0
7: counting	0	0.1
8: lists/sets	0.5	0
9: simple negation	0.1	0
10: indefinite knowledge	0.6	0.1
11: basic coreference	0.3	0.1
12: conjunction	0	0
13: compound coreference	1.3	0
14: time reasoning	0	0.2
15: basic deduction	0	0
16: basic induction	0.2	0.1
17: positional reasoning	0.5	0
18: size reasoning	0.3	0.4
19: path finding	2.3	0
20: agents motivation	0	0
Tasks with 0 % error	7	11
Mean % Error	0.5	0.3

Table 1: Mean % Error on the 20 Babi tasks.

compare the performance with the Recurrent Entity Networks model (EntNet) [18]. Performance is measured in terms of mean percentage error on the tasks.

Training Details: We used Adam and did a grid search for the learning rate in $\{0.01, 0.005, 0.001\}$ and choose a fixed learning rate of 0.005 based on performance on the validation set, and clip the gradient norm at 2. We keep all other details similar to [18] for a fair comparison. embedding dimensions were fixed to be 100, models were trained for a maximum of 250 epochs with mini-batches size of 32 for all tasks except 3 for which the batch size was 16. The document sizes were limited to most recent 70 sentences for all tasks, except for task 3 for which it was limited to 130. The RelNet models were run for 5 times with random seed on each task and the model with best validation performance was chosen as the final model. The baseline EntNet model was run for 10 times for each task [18].

The results are shown in Table 1. The RelNet model achieves a mean error of **0.285%** across tasks which is better than the results of the EntNet model [18]. The RelNet model is able to achieve 0% test error on 11 of the tasks, whereas the EntNet model achieves 0% error on 7 of the tasks.

5 Conclusion

We demonstrated an end-to-end trained neural network augmented with a structured memory representation which can reason about entities and relations for question answering. Future work will investigate the performance of these models on more real world datasets, interpreting what the models learn, and scaling these models to answer questions about entities and relations from reading massive text corpora.

References

- [1] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61, 2013.
- [2] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of*

- the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008.
- [3] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.
 - [4] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. *The semantic web*, pages 722–735, 2007.
 - [5] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL-HLT*, pages 74–84, 2013.
 - [6] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.
 - [7] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.
 - [8] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.
 - [9] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. Incorporating selectional preferences in multi-hop relation extraction. *Proceedings of AKBC*, pages 18–23, 2016.
 - [10] Arvind Neelakantan, Quoc V Le, and Ilya Sutskever. Neural programmer: Inducing latent programs with gradient descent. *arXiv preprint arXiv:1511.04834*, 2015.
 - [11] Arvind Neelakantan, Quoc V Le, Martin Abadi, Andrew McCallum, and Dario Amodei. Learning a natural language interface with neural programmer. *arXiv preprint arXiv:1611.08945*, 2016.
 - [12] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*, 2016.
 - [13] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.
 - [14] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics, 2010.
 - [15] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. Chains of reasoning over entities, relations, and text using recurrent neural networks. *arXiv preprint arXiv:1607.01426*, 2016.
 - [16] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1156–1165. ACM, 2014.
 - [17] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.
 - [18] Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. *International Conference on Learning Representations*, 2017.
 - [19] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.
 - [20] Tsendsuren Munkhdalai and Hong Yu. Reasoning with memory augmented neural networks for language comprehension. *arXiv preprint arXiv:1610.06454*, 2016.

- [21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.
- [22] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1223–1233. Association for Computational Linguistics, 2010.
- [23] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, volume 2, page 6, 2013.
- [24] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *ICLR*, 2014.
- [25] Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. *arXiv*, 1603, 2016.
- [26] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.
- [27] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.
- [28] Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. Question answering on knowledge bases and text using universal schema and memory networks. *arXiv preprint arXiv:1704.08384*, 2017.
- [29] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Learning to compose neural networks for question answering. In *NAACL*, 2016.
- [30] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*, pages 2013–2018, 2015.
- [31] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.
- [32] Adam Santoro, David Raposo, David GT Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. *arXiv preprint arXiv:1706.01427*, 2017.
- [33] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.
- [34] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.
- [35] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.