Relational Graph Representation Learning for Open-Domain Question Answering

Salvatore Vivona
Autodesk AI Lab
Toronto, Canada
vivonasg@cs.toronto.edu

Kaveh Hassani
Autodesk AI Lab
Toronto, Canada
kaveh.hassani@autodesk.com

Abstract

We introduce a relational graph neural network with bi-directional attention mechanism and hierarchical representation learning for open-domain question answering task. Our model can learn contextual representation by jointly learning and updating the query, knowledge graph, and document representations. The experiments suggest that our model achieves state-of-the-art on the WebQuestionsSP benchmark.

1 Introduction

Fusing structured knowledge from knowledge graphs into deep models using Graph Neural Networks (GNN) [23, 30, 29] is shown to improve their performance on tasks such as visual question answering [12], object detection [9], natural language inference [2], neural machine translation[11], and open-domain question answering [17]. This particularly helps question answering neural models such as Memory Networks [22] and Key-Value Memory Networks [10] by providing them with well-structured knowledge on specific and open domains [28]. Most models, however, answer questions using a single information source, usually either a text corpus, or a single knowledge graph. Text corpora have high coverage but extracting information from them is challenging whereas knowledge graphs are incomplete but are easier to extract answers from [17].

In this paper, we propose a relational GNN for open-domain question answering that learns contextual knowledge graph embeddings by jointly updating the embeddings from a knowledge graph and a set of linked documents. More specifically, our contributions are as follows: (1) we use documents as contextualized relations to augment the knowledge graph and co-train the knowledge graph and document representations, (2) we introduce a bi-directional graph attention mechanism for knowledge graphs, (3) we propose a simple graph coarsening method to fuse node and cluster representations, and (4) we show that our model achieves state-of-the-art results on the WebQuestionsSP benchmark.

2 Related Works

Graph representation learning on knowledge graphs allows projecting high-level factual information into embedding spaces which can be fused with other learned structural representations to improve down-stream tasks. Pioneer works such as Trans-E [1], Complex-E [18], Hole-E[13], and DistMul [25] use unsupervised and mostly linear models to learn such pre-trained representations. A few recent works, on the other hand, use GNNs to compute the knowledge graph representation [24, 20, 27]. These high-level knowledge graph representations are particularly important for question answering task [22, 10, 17]. We use pre-trained representations to initialize the model and then update them using a relational and bi-directional GNN model.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Only a few works fuse knowledge graphs with text corpora to answer question. In [3] early fusion of knowledge graph facts and text is performed using Key-Value Memory Networks (KVMN). This model, however, ignores relational structure between the text and knowledge graph. Our model links the knowledge graph and documents through document-contextualized edges and also links entities with their positions in the corpus. This linking is used in GRAFT-Net as well which also performs question answering through fusing learned knowledge graph and linked document representations [17]. Unlike GRAFT-Net, our model uses variants of differential pooling [26] and bi-directional graph attention [19] for more powerful message passing. Our model also introduces trainable document-contextualized relation embeddings instead of exclusively relying on fixed relation representations.

3 Method

Assume a knowledge graph $G = (V, E, R)$ where V, R, and E are sets of entities, relations, and edges, respectively. Each edge $e_i = (s, r, o) \in E$ denotes an object o interacting with subject s through a relationship r. Given a textual query $q \in (w_1...w_{|q|})$ and a set of relevant documents D linked to G by an imperfect linker, the task is to predict the answer nodes: $v_{a_q} \in V$. A query can have zero or multiple answers and hence the task is reduced to binary node classification on G (i.e., binary cross entropy loss). Following [17], we first extract a subgraph $G_q \subset G$ which contains v_{a_q} with high probability. This is done by linking the query entities to G and expanding their neighborhood using Personalized PageRank (PPR) method. We then initialize G_q, q, D, and R in embedding space as follows. Each entity $v \in V$ is initialized using pre-trained TransE embeddings [1]: $h_v^{(0)} = \text{TransE}(v) \in \mathbb{R}^{d_{kb}}$. Documents are encoded as $H_D^{(l)} \in \mathbb{R}^{|D| \times n_d \times d_w}$ where n_d is the maximum number of tokens in documents, and $H_{D_{k,j}}^{(l)} \in \mathbb{R}^{d_w}$ corresponds to the jth token embedding of dimension d_w in the kth document. A bidirectional LSTM that takes in pre-trained GloVe embeddings [15] of tokens and computes a global sequence embedding is shared between query and documents to initialize their embeddings: $h_q^{(0)} = \text{BiLSTM}\left(w_1...w_{|q|}\right)$ and $H_d^{(0)} = \text{BiLSTM}\left(w_1...w_{|d|}\right)$. Each relation $r \in R$ is initialized as: $h_r^{(0)} = \text{ReLU}\left(\mathbf{W}_r^{(0)}\left[\text{TransE}(r) \parallel \mu_{glove}(w_1...w_{|r|})\right]\right)$ where $\mathbf{W}_r^{(0)} \in \mathbb{R}^{(d_r+d_w) \times d_r}$ is a trainable parameter, $\parallel$ is the concatenation operator, and $\mu_{glove} \in \mathbb{R}^{d_w}$ is the mean pool over GloVe embeddings of the relation tokens.

3.1 Documents as Relations

We use documents to explicitly augment the knowledge graph by forming additional relations between entities. Assuming two unique entities v_i and v_j co-occurring within document d_k, we compute the document relation embedding for both plausible directions between v_i and v_j. This lets the model to learn to attend to the direction that optimizes the objective. These embeddings are computed as follows:

$$h_{r(d_k)}^{(l)}\left(v_i \to v_j\right) = \text{ReLU}\left(\mathbf{W}_{dr}^{(l)}\left[h_{v_i(d_k)}^{(l)} \parallel h_{d_k}^{(l)} \parallel h_{v_j(d_k)}^{(l)}\right]\right) \tag{1a}$$

$$h_{r(d_k)}^{(l)}(v_j \to v_i) = \text{ReLU}\left(\mathbf{W}_{dr}^{(l)}\left[h_{v_j(d_k)}^{(l)} \parallel h_{d_k}^{(l)} \parallel h_{v_i(d_k)}^{(l)}\right]\right) \tag{1b}$$

where $\mathbf{W}_{dr}^{(l)} \in \mathbb{R}^{3d_w \times d_r}$ is a trainable parameter, $h_{d_k}^{(l)}$ is the learned textual embedding of the document and $h_{v(d_k)}^{(l)} = \sum_{p \in M_{d_k}(v)} H_{d_k,p}^{(l)}$ denotes the textual representation of the entity and is computed by summing up the textual embeddings of its tokens within the document (i.e., $M_{d_k}(v)$ returns the positions of entity v in document d_k). An example of this process is illustrated in Figure 1.

3.2 Bi-Directional Graph Attention

To update the node embedding $h_v^{(l)}$ we aggregate the embeddings of its connecting edges. An edge embedding is computed by aggregating its relation embedding $h_r^{(l)}$ with the neighbor node embedding $h_v^{(l)}$ connecting through that edge. Let $e_{out} = (v, r, v_i) \in E$ and $e_{in} = (v_j, r, v) \in E$ represent two facts in which the node v is either the subject or the object. If the node is an object, we will refer the edge to point inwards, and outwards if it is a subject. The edge embedding are computed as follows:

$$h_{e_{out}}^{(l)} = \text{ReLU}\left(\mathbf{W}_r^{(l)} h_r^{(l)} + \mathbf{W}_v^{(l)} h_{v_i}^{(l)}\right), \quad h_{e_{in}}^{(l)} = \text{ReLU}\left(\mathbf{W}_r^{(l)}(-h_r^{(l)}) + \mathbf{W}_v^{(l)} h_{v_j}^{(l)}\right) \tag{2}$$

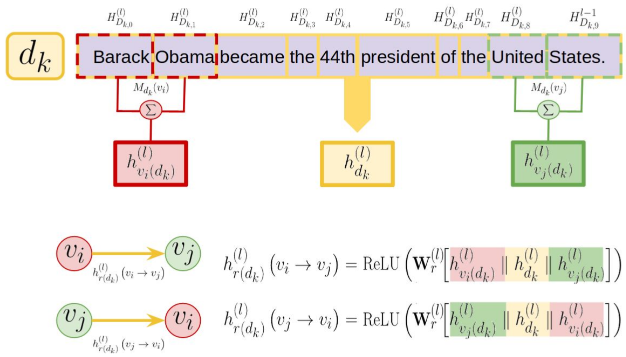

Figure 1: An example of forming relations from documents.

where $h^{(l)}_{e_{in}}$ and $h^{(l)}_{e_{out}}$ denote the embeddings of the inward and outward edges connecting to node v, and $\mathbf{W}^{(l)}_r$ and $\mathbf{W}^{(l)}_v$ are trainable parameters. To distinguish inward edges from outward edges, we negate h_r. This is distinct from previous approaches which only process incoming nodes [17].

The next step is aggregating the embeddings of the edges connecting to the node, i.e., $h^l_{e_{out}}$ and $h^l_{e_{in}}$. We apply two attention mechanisms to perform the aggregation and hence the model separately aggregates weighted sums of edge embeddings over each attention parameters: $\alpha^{(l)}_q$ and $\alpha^{(l)}_{GAT}$. The first attention parameter $\alpha^{(l)}_q$ is based on the normalized similarity between the relation embedding of an edge (i.e., $h^{(0)}_r$) and the question embedding (i.e., $h^{(0)}_q$) at layer 0: $\alpha^{(l)}_{q,e_{in}} = \mathrm{softmax}(h^{0\top}_{r_{in}} h^0_q)$ and $\alpha^{(l)}_{q,e_{out}} = \mathrm{softmax}(h^{0\top}_{r_{out}} h^0_q)$. This captures the semantic similarity between the query and the inward and outward relations separately. The second attention parameter $\alpha^{(l)}_{GAT}$ is based on the similarity between a node and its neighbors with respect to the relationship between them. Assume (v_i, r, v_j) is an edge between nodes v_i and v_j with relationship of type r. The edge score is defined as the dot product between the edge embedding of the original direction and inverted direction (v_j, r, v_i) and then normalized over all inward edges: $\alpha^{(l)}_{e_{in}} = \mathrm{softmax}(h^{(l)\top}_{\vec{e}} h^{(l)}_{\overleftarrow{e}})$ and outward edges: $\alpha^{(l)}_{e_{out}} = \mathrm{softmax}(h^{(l)\top}_{\vec{e}} h^{(l)}_{\overleftarrow{e}})$. Unlike graph attention [26] this method addresses heterogeneous directed knowledge graphs. Finally, the model updates the node embedding (Figure 2):

$$h^{(l+1)}_v = \mathrm{ReLU}\left(\mathbf{W}^{(l)}_v h^{(l)}_v + \sum_{e_{in} \in N_{in}(v)} \alpha^{(l)}_{e_{in}} \mathbf{W}^{(l)}_{e_{in}} h^{(l)}_{e_{in}} + \sum_{e_{in} \in N_{out}(v)} \alpha^{(l)}_{e_{out}} \mathbf{W}^{(l)}_{e_{out}} h^{(l)}_{e_{out}}\right) \tag{3}$$

3.3 Hierarchical Aggregation

GNN models cannot learn hierarchical representations as they do not exploit the compositionality of graphs. Inspired by [26] we define a linear layer to assign nodes into clusters and then use them to represent the nodes. This increases the receptive field by letting messages to pass among nodes belonging to the same cluster. We compute the soft cluster assignments using a linear layer $\mathbf{C}^{(l)} = \mathrm{softmax}(\mathbf{H}^{(l)}_v \mathbf{W}^{(l)}_c)$ where $H_v \in \mathbb{R}^{n_v \times d_v}$ is the node embedding matrix, $\mathbf{W}^{(l)}_c \in \mathbb{R}^{d_v \times n_c}$ is a trainable parameter, and $\mathbf{C}^{(l)} \in \mathbb{R}^{n_v \times n_c}$ is the normalized soft assignment matrix mapping n_v nodes to n_c clusters. We then compute the cluster centroids using $\mathbf{H}^{(l)}_c = \mathbf{C}^{(l)\top}\mathbf{H}^{(l)}_v$ and compute the cluster-based node representation using $\mathbf{H}^{(l)}_{vc} = \mathrm{softmax}((\mathbf{H}^{(l)}_v \mathbf{W}^{(l)}_c)^\top)\mathbf{H}^{(l)}_c$. Finally, we concatenate the node representation in the final layer with all the cluster-based node representations from previous layers (i.e., similar to DenseNet [5]). We reduce the dimensionality using a trainable parameter $\mathbf{W}_{final} \in \mathbb{R}^{d_v L \times d_v}$ followed by a sigmoid function to produce the probability score per node.

$$\mathcal{Y} = \sigma\left(\mathbf{W}_{final}\left[\left(\Big\Vert_{l=0}^{L-1} \mathbf{H}^{(l)}_{vc}\right) \parallel \mathbf{H}^{(L)}_v\right]\right) \tag{4}$$

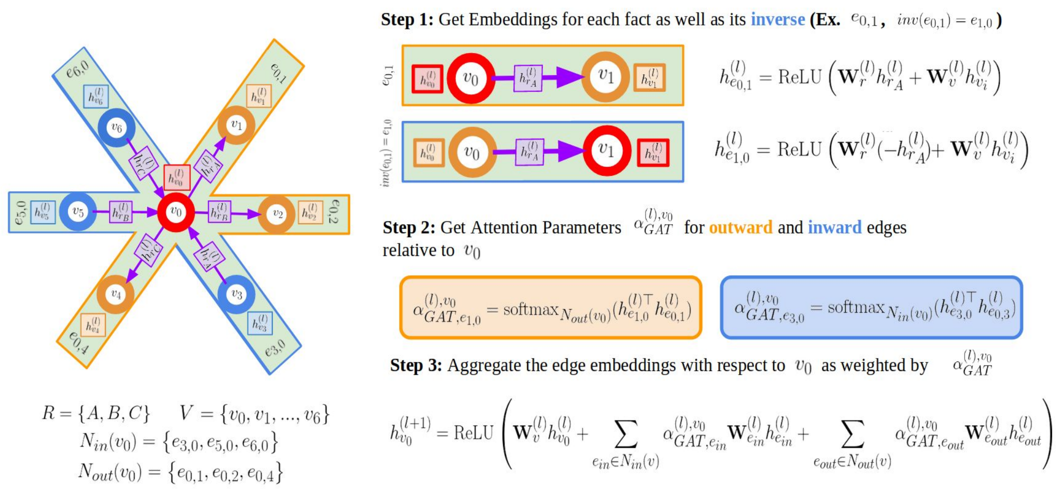

Figure 2: Message passing in proposed bi-directional graph attention mechanism.

Model	$\mathbf{F}_1^{micro}$	$\mathbf{F}_1^{macro}$	$\mathbf{F}_1^{avg}$	Hits@1
KVMN [3]	-	-	30.9	40.5
GRAFT-Net [17]	66.0	61.0	60.4	67.8
Ours	**72.6**	**62.5**	**61.7**	**68.2**

Table 1: Comparative results on the WebQuestionsSP benchmark.

4 Results

We implemented the model with Pytorch [14] on a Nvidia DGX-1 server with 8 Volta GPUs and optimize it using Adam [7] with initial learning rate of 0.001 and batch size of 10 for 100 epochs with early stopping. The learning rate is decayed by a factor of 0.8 every 10 epochs. We also apply batch-normalization [6] and dropout [16]. We evaluated our model on the **WebQuestionsSP** dataset consisting of 4,737 natural language questions (i.e., 3,098 training, 250 validation, and 1,639 test questions) posed over Freebase entities [8]. Following [17], we apply the same pre-processing and report average F_1 and Hits@1, as well as micro-average, and macro-average F_1 scores. [4] suggests that micro-averaged F_1 best represents the performance on imbalanced binary classification .

Table 1 shows the performance of our model compared to other models that also feature early fusion of the knowledge graph and text. These include Key-Value Memory Networks with (KVMN) [3] and GRAFT-Net [17]. The results suggest that our model outperforms GRAFT-Net with an absolute increase in all metrics. To investigate the effect of the proposed methods we performed an ablation study by masking each introduced component and training and evaluating the model. The results in Table 2 (Appendix) shows the effect of each component and suggest that all introduced components contribute to the performance.

5 Conclusion

We introduced a relational GNN with bi-directional attention and hierarchical representation learning for open-domain question answering that jointly learns to represent the query, knowledge graph, and documents. The experiments showed that our model achieves state-of-the-art performance on WebQuestionsSP. For future directions, we are planning to expand our model towards cross-modal question answering benchmarks such as Fact-based Visual Question Answering (FVQA) [21].

References

[1] 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.

[2] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2406–2417, 2018.

[3] Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. Question answering on knowledge bases and text using universal schema and memory networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 358–365, 2017.

[4] George Forman and Martin Scholz. Densely connected convolutional networks. In *ACM SIGKDD*, pages 49–57, 2010.

[5] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456, 2015.

[7] Diederik P Kingma and Jimmy Lei Ba. Adam: Amethod for stochastic optimization. In *International Conference on Learning Representation*, 2014.

[8] Chen Liang, Jonathan Berant, Quoc Le, Kenneth Forbus, and Ni Lao. Glove: Global vectors for word representation. pages 1532–1543, 2017.

[9] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. The more you know: Using knowledge graphs for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2673–2681, 2017.

[10] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[11] Diego Moussallem, Mihael Arčan, Axel-Cyrille Ngonga Ngomo, and Paul Buitelaar. Augmenting neural machine translation with knowledge graphs. *arXiv preprint arXiv:1902.08816*, 2019.

[12] Medhini Narasimhan, Svetlana Lazebnik, and Alexander Schwing. Out of the box: Reasoning with graph convolution nets for factual visual question answering. In *Advances in neural information processing systems*, pages 2654–2665. 2018.

[13] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Thirtieth AAAI conference on artificial intelligence*, pages 1955–1961, 2016.

[14] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[15] Jeffrey Pennington, Richard Socher, and Manning Christopher. Glove: Global vectors for word representation. *EMNLP*, 2014.

[16] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[17] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, 2018.

[18] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080, 2016.

[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Pietro Romero, Adriana Liò, and Yoshua Bengio. Graph attention networks. *International Conference on Learning Representations*, 2018.

[20] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 968–977, 2019.

[21] Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Fvqa: Fact-based visual question answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(10):2413–2427, 2018.

[22] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *International Conference on Learning Representations*, 2015.

[23] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

[24] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *Thirtieth AAAI Conference on Artificial Intelligence*, pages 2659–2665.

[25] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *International Conference on Learning Representations*, 2015.

[26] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, pages 4800–4810, 2018.

[27] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, pages 5165–5175, 2018.

[28] Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*, pages 6069–6076, 2018.

[29] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *arXiv preprint arXiv:1812.04202*, 2018.

[30] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

6 Appendix

Masked Component	$\mathbf{F}_1^{micro}$	$\mathbf{F}_1^{macro}$	$\mathbf{F}_1^{avg}$	**Hits@1**
Document Relations	69.3	61.1	59.0	69.3
Bi-Directional Attention	67.9	60.6	58.5	66.9
Graph Coarsening	67.9	60.8	58.7	67.8
No Mask	**72.6**	**62.5**	**61.7**	**68.2**

Table 2: Effect of components on the performance of the proposed model.