Identifying relevant common sense information in knowledge graphs

Anonymous ACL submission

Abstract

Knowledge graphs are often used to store common sense information that is useful for various tasks. However, the extraction of contextually-relevant knowledge is an unsolved problem, and current approaches are relatively simple. Here we introduce a triple selection method based on a ranking model and find that it improves question answering accuracy over existing methods. We additionally investigate methods to ensure that extracted triples form a connected graph. Graph connectivity is important for model interpretability, as paths are frequently used to understand reasoning from question to answer. We make our code and data available at https://github.com/anonymous.

1 Introduction

For models to be able to reason about situations that arise in everyday life, they must have access to contextually appropriate common sense information. This information is commonly stored as a large set of facts from which the model must identify a relevant subset. One approach to structuring these facts is as a knowledge graph. Here, nodes represent high-level concepts, and relationships are expressed via typed edges joining two nodes. Each edge type represents a different kind of conceptual relationship between concepts. The contextually-relevant subgraphs ('schema graphs') that are extracted from these graphs are often encoded using neural models, which are trained for tasks including question answering or natural language inference.

Prior work has focused on different ways to encode this information, including using it as input directly to a transformer or using a graph neural network (GNNs) (Feng et al., 2020; Yasunaga et al., 2021). However, the question of how to identify useful information has been under-explored, particularly in work that uses GNN encoders. The simple retrieval methods that are used could limit performance on tasks if contextually-important information is not retrieved.

In this paper we explore methods to construct high-quality schema graphs containing contextually-relevant information. We approach this as a ranking task across triples in a knowledge graph, and select the highest scoring for the schema graph. However, simply using the most relevant triples as input for a GNN is insufficient, as the resulting subgraph is likely to have low connectivity. This is problematic for two reasons. First, it inherently limits the power of the GNN, as nodes in different graph components will not be updated with information from each other. Second, paths of reasoning through the schema graphs are often used as explanations for model behaviour (Feng et al., 2020; Yasunaga et al., 2021; Wang et al., 2020). If the graph consists of multiple separate components, this becomes impossible.

The issue of graph disconnectedness is compounded when certain nodes are required to be included. For example, in question answering, concepts mentioned in the question and in a candidate answer should be identified and included. A path starting from a question concept can then be evaluated for plausibility of reasoning as it progresses towards an answer concept. We therefore also apply a graph algorithm to ensure that the schema graph is connected, taking into account the identified edges and desired nodes, and use an embedding-based method to identify concepts mentioned.

Our contributions are summarised as follows:

- Apply a ranking model to identify common sense triples that are relevant to some context.
- Identify and thoroughly investigate methods to ensure schema graph connectivity.
- Compare existing lexical approaches to entity linking to a simple embedding-based method.

2 Background

Many prior approaches to retrieving relevant common sense triples from a knowledge graph start by identifying relevant nodes. Simple lexical overlap between a concept and the context (e.g. question text) is often used for this (Kundu et al., 2019; Khot et al., 2019). However, this entity linking approach is likely to only retrieve simple concepts (Speer et al., 2017), as the idiosyncratic phrasing of some node names in knowledge graphs like ConceptNet (Speer et al., 2017) are unlikely to show up in text. Becker et al. (2021) investigate this in detail and propose a series of pre-processing steps that allow lexically-based linking without exact phrase matches. For the same reason, the heuristics used by Lin et al. (2019) for lexical matching are employed by a series of later works (Feng et al., 2020; Yasunaga et al., 2021; Wang et al., 2020). Although lexical matching is a frequent approach with common sense knowledge graphs, in other domains embedding-based approaches are more popular (Gillick et al., 2019). These work by embedding the candidate text and finding the nearest neighbour in the space of entity embeddings.

In question answering, Lin et al. (2019) split these concepts into those identified in the question and in the answer, and iteratively find shortest paths between the two sets. This process builds a set of potentially relevant nodes until a maximum number is collected, or the path lengths exceed a threshold. The final schema graph, which is used as input to models, is constructed from this set with all valid edges added.

Some approaches work by scoring nodes and triples that have been identified. Kundu et al. (2019) score multiple paths for each question and answer and use the mean as a final scoring. Yasunaga et al. (2021) build a schema graph following Lin et al. (2019), and additionally score each node for relevance to a question using `RoBERTa` (Liu et al., 2019). Ranking is also common with prose facts, particularly when facts are concatenated to some other input in transformer-based models that take a limited number of tokens as input (Wang et al., 2021).

3 Methodology

In this section we introduce our methods for constructing a schema graph $\mathcal{G}$ for a question answering task. The graph should contain triples that are useful in distinguishing the correct answer from a set of distractors. For each instance, we represent the question text as q and the ith candidate answer as a_i, and the set of concepts extracted from each as $\mathcal{C}_q$ and $\mathcal{C}_{a_i}$ respectively.

3.1 Triple selection

We cast the task of identifying relevant triples in the knowledge graph as a ranking problem, where the highest-ranked triples are those most relevant to $[q; a_i]$. We use an existing ranking model trained to rank facts highly if they constitute part of an explanation for why a_i is the correct answer to q (Pan et al., 2021). This was developed for the TextGraphs 2021 shared task on explanation regeneration for science questions (Thayaparan et al., 2021) and achieved the highest performance. Facts that are used in an explanation are likely to be useful when choosing between answers, making the model a natural choice for identifying relevant triples.

The model consists of two parts: a fact retriever and a re-ranker. We follow the training procedure in Pan et al. (2021) and use one model based on `RoBERTa-Large` (Liu et al., 2019) for each stage. At inference time we use only the re-ranker to score each triple[1] in relation to $q; a_i$. To speed this up we pre-compute embeddings for each $q; a_i$ and each triple.

We select the top ranked triples in each instance according to limits on the total number of edges and nodes in $\mathcal{G}$. We limit possible edges to the top e ranked triples. Iterating through these in rank order, we add the triple (s, r, o) to $\mathcal{G}$ only if adding s and o does not increase the total number of nodes to above n. If $n < 2e$ then some of the top edges will be excluded; this limits the number of nodes in the graph while allowing highly-ranked edges to be present if they share nodes with other edges. We set $n = 50$ and $e = 40$ following initial experiments.

3.2 Constructing $\mathcal{G}$

The most straightforward way to construct $\mathcal{G}$ is to use only the edges identified in §3.1 and grounded nodes $\mathcal{C}_q \cup \mathcal{C}_{a_i}$ However, this method is limited in that triples are not likely to connect with $\mathcal{C}_q$ or $\mathcal{C}_{a_i}$. Indeed, there is no guarantee that the triples are connected to each other. This is problematic in cases where paths in the schema graph are to be

[1] We linearize triples using the templates from https://github.com/commonsense/conceptnet5/wiki/Relations.

used in an explanation (Feng et al., 2020; Yasunaga et al., 2021).

To rectify this, we find the minimum spanning tree (MST) that spans all nodes in $\mathcal{G}$, including nodes $\mathcal{C}_q \cup \mathcal{C}_{a_i}$ and taking into account the edges added in the previous step. This is the Steiner tree problem, which is NP-hard; we apply an approximation algorithm (Wu et al., 1986) to find solutions in a reasonable amount of time. We experiment with two variants: one where edges are uniformly weighted, and another where the triple scores are used as weights.

We further use the triple scores with the pathfinding method used in previous work (Lin et al., 2019), transforming this into a weighted shortest path search. We iteratively find the shortest path between any pair of concepts in $\mathcal{C}_q$ and $\mathcal{C}_{a_i}$, adding nodes on the paths to a set until a maximum size is reached. $\mathcal{G}$ is then formed from these nodes, as well as all valid edges between pairs from this set. We set the maximum size to be 50.

3.3 Identifying relevant concepts

It is important that $\mathcal{C}_q$ and $\mathcal{C}_{a_i}$ accurately reflect concepts mentioned in q and a, primarily to aid with explanations. An explanation path begins with a question concept and ends with an answer concept; if either is nonsensical then the explanation is invalid. The path is otherwise often used to check if the reasoning process is plausible. Additionally, the pathfinding method for schema graph construction relies on the quality of this grounding.

We use two methods for entity linking. The first is from prior work, and is based on lexical matching with heuristics (Lin et al., 2019). These include lemmatising words if an exact match is not found, and a method to avoid selecting nodes whose overlap. Despite this, this method is not able to identify relevant concepts where phrasing is substantially different; this occurs often with more specific concepts. To account for this, our second method is based on embeddings from `RoBERTa`. We embed each concept, and for each q and a_i find the 10 most similar concepts via Euclidean distance. Embeddings are constructed in each case by mean-pooling across all tokens.

3.4 Evaluation

We evaluate the quality of the extracted schema graphs by comparing accuracy on a question answering task when using them versus using baseline schema graphs. These graphs are used as input to two models, MHGRN (Feng et al., 2020) and QA-GNN (Feng et al., 2020), which are both designed for question answering with knowledge graphs. The baseline schema graph is built using the same method as both models use, which is to collect nodes along two-hop paths between question concepts and answer concepts (Lin et al., 2019).

We report accuracy on two datasets, OpenbookQA (Mihaylov et al., 2018) and CommonsenseQA (Talmor et al., 2019). OpenbookQA is a collection of science questions, and so is in-domain with respect to the data used to train the fact scorer. CommonsenseQA targets more general common sense; performance here is a reflection on how transferable the fact scorer is to other domains. This dataset has no public test set labels, so we report results on the 'in house' test split defined by Lin et al. (2019). Each model is run three times and the mean accuracy reported. Model hyperparameters are reported in appendix A.

Our base knowledge graph is ConceptNet (Speer et al., 2017). Following previous work (Lin et al., 2019), we merge similar relations and add reverse relations to the extracted graph.

4 Results

Our results on OpenbookQA are presented in table 1 and CommonsenseQA in table 2. In the majority of cases the proposed ranking systems outperform the baseline, and in some cases brings the accuracy of the older MHGRN system to the level of QA-GNN reported in Yasunaga et al. (2021). On OpenbookQA, we observe a maximum increase in accuracy of 2.7% using MHGRN and over 6% using QA-GNN; across all datasets and models we observe at least a 1% improvement over the baseline in the best case. This suggests that the ranker is able to identify facts which are relevant to the question, and that the models are able to successfully use them.

4.1 Analysis

We observe that, if models are provided with only the top-rated facts and the concepts identified by lexical linking, and no additional connectivity is added, performance remains similar to the baseline in the majority of cases. In this situation the GNN is limited in how much information it can pass between nodes due to low connectivity, although it is noteworthy that this does not cause performance

Grounding	Schema graph	MHGRN	QA-GNN
Lexical	*Baseline*	65.13	61.33
	Only top rated	65.27	65.53
Lexical	MST	65.07	65.80
	Weighted MST	59.93	62.47
	Weighted path	65.13	63.53
Embedding	MST	67.00	66.93
	Weighted MST	66.27	67.20
	Weighted path	**67.80**	**67.60**

Table 1: Accuracy on **OpenbookQA** with different schema graph construction methods.

Grounding	Schema graph	MHGRN	QA-GNN
Lexical	*Baseline*	68.44	70.53
	Only top rated	69.00	69.54
Lexical	MST	69.27	69.97
	Weighted MST	69.35	**71.53**
	Weighted path	68.71	70.34
Embedding	MST	69.44	70.05
	Weighted MST	**70.19**	69.97
	Weighted path	69.30	69.86

Table 2: Accuracy on **CommonsenseQA** with different schema graph construction methods.

to drop.

When using lexical grounding, using a weighted path has little impact on accuracy. Ensuring graph connectivity via an unweighted MST generally has minimal impact on increasing accuracy over the baseline; in the case of OpenbookQA with QA-GNN the increase is mostly realised when adding the disconnected top-rated edges. Using a weighted MST has generally positive effects, although in one case for OpenbookQA accuracy does drop. These observations are likely due not just to the choice of nodes in each method, but also the number of them. In the weighted case for both datasets, an average of 37 nodes and 83 edges are added to a schema graph because of the spanning tree, compared with 26 nodes and 71 edges in the unweighted case. The larger schema graph, coupled with the particular nodes and edges chosen, appears to benefit CommonsenseQA performance while being less useful, or harmful, for OpenbookQA.

The increase in score between lexical and embedding-based entity linking with an unweighted MST suggests that the concepts identified by the latter method are useful for question answering. Three of the best results use embedding-based grounding and a weighted graph completion method, suggesting this to be the best approach for extracting common sense information from a knowledge graph. The weighted pathfinding method is particularly successful for OpenbookQA, whereas weighted MST approaches are best for CommonsenseQA.

Similarly to with lexical grounding, the weighted MST with embedding grounding adds more nodes and edges on average (153 nodes, 217 edges) than the unweighted one (112, 172). The weighted MST also gives marginally better performance in some cases, although there is less difference between the two than with lexical grounding. There is a noteworthy increase in graph size in both cases. This is likely due to the kinds of nodes identified by entity linking – we observe that concepts which are directly related to the context are also more specific, and so are less connected within the overall graph. Conversely, concepts that are identified lexically are likely to be simpler and more general, and so better connected within the graph, meaning fewer additional nodes and edges are required to build the MST.

5 Conclusion

We present a method for extracting relevant information from a common sense knowledge graph, casting it as a ranking problem. We show that scores obtained from a ranking model can be used to select triples containing useful information for a question answering task, improving performance over a commonly-used approach. As it is undesirable for schema graphs to have low connectivity, particularly when using graphs for model interpretation, we use algorithms for calculating minimum spanning trees over a supplied set of nodes and edges to ensure the graph is connected. We find that this helps performance; in particular, the models with highest accuracy on CommonsenseQA use a weighted version of this. We additionally investigate a weighted pathfinding method and find that it gives the highest accuracy on OpenbookQA. We distribute the calculated schema graphs to facilitate future work; these drop in to existing models with no further processing required.

Future work might investigate the influence of the fact ranker, as our results suggest that it can transfer from the science to general common sense domain successfully. Further training of the ranker using higher-quality negative samples from e-QASC (Jhamtani and Clark, 2020) may yield better performance, as noted by Pan et al. (2021).

References

Maria Becker, Katharina Korfhage, and Anette Frank. 2021. COCO-EX: A tool for linking concepts from texts to ConceptNet. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1295–1309, Online. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Harsh Jhamtani and Peter Clark. 2020. Learning to Explain: Datasets and Models for Identifying Valid Reasoning Chains in Multihop Question-Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 137–150, Online. Association for Computational Linguistics.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. What's missing: A knowledge gap guided approach for multi-hop question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2814–2828, Hong Kong, China. Association for Computational Linguistics.

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. Exploiting explicit paths for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2737–2747, Florence, Italy. Association for Computational Linguistics.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China. Association for Computational Linguistics.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2021. On the Variance of the Adaptive Learning Rate and Beyond. *arXiv:1908.03265 [cs, stat]*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Chunguang Pan, Bingyan Song, and Zhipeng Luo. 2021. DeepBlueAI at TextGraphs 2021 shared task: Treating multi-hop inference explanation regeneration as a ranking problem. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 166–170, Mexico City, Mexico. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Mokanarangan Thayaparan, Marco Valentino, Peter Jansen, and Dmitry Ustalov. 2021. TextGraphs 2021 shared task on multi-hop inference for explanation regeneration. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 156–165, Mexico City, Mexico. Association for Computational Linguistics.

Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. 2021. Retrieval enhanced model for commonsense generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3056–3062, Online. Association for Computational Linguistics.

Peifeng Wang, Nanyun Peng, Filip Ilievski, Pedro Szekely, and Xiang Ren. 2020. Connecting the dots: A knowledgeable path generator for commonsense question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4129–4140, Online. Association for Computational Linguistics.

Y F Wu, P Widmayer, and C K Wong. 1986. A faster approximation algorithm for the steiner problem in graphs. *Acta Informatica*, 23(2):223–229.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

A Hyperparameters

A.1 Question answering models

We use the same hyperparameters for MHGRN and QA-GNN as used in the papers which respectively introduced them (Feng et al., 2020; Yasunaga et al., 2021). We optimise both models using RAdam (Liu et al., 2021) and a learning rate of $1\mathrm{e}-3$ for the text encoder and $1\mathrm{e}-5$ for the graph encoder. A maximum of 128 tokens are input to the text encoder, which is initialised as `RoBERTa-large`. A L2 weight decay of 0.01 is used.

For MHGRN, batch size is 32 and the text encoder is frozen for the first 3 epochs. A 1-layer 100-dimensional GNN is used with 3-hop message passing at each layer.

For QA-GNN, batch size is 128 and the text encoder is frozen for the first 4 epochs. A 5-layer 200-dimensional GNN is used.