FINE-TUNING MULTI-HOP QUESTION ANSWERING WITH HIERARCHICAL GRAPH NETWORK

Guanming Xiong*
School of Software & Microelectronics
Peking University
gm_xiong@pku.edu.cn

ABSTRACT

In this paper, we present a two stage model for multi-hop question answering. The first stage is a hierarchical graph network, which is used to reason over multi-hop question and is capable to capture different levels of granularity using the nature structure(i.e., paragraphs, questions, sentences and entities) of documents. The reasoning process is convert to node classify task(i.e., paragraph nodes and sentences nodes). The second stage is a language model fine-tuning task. In a word, stage one use graph neural network to select and concatenate support sentences as one paragraph, and stage two find the answer span in language model fine-tuning paradigm. Evaluated on HotpotQA, the two stage model achieves competitive performance in distractor setting compared to other existing systems on the leaderboard.

1 INTRODUCTION

In one-hop question answering, also known as machine reading comprehension, answers span can be derived from a single paragraph. Numerous neural models have been proposed (Seo et al. (2017), Chen et al. (2017), Clark & Gardner (2018), Feldman & El-Yaniv (2019)) and achieved admirable performances on several different data sets, such as SQuAD(Rajpurkar et al. (2016), Rajpurkar et al. (2018)) and TriviaQA(Joshi et al. (2017)). in such task, language models have been proved to performed better than human after the release of BERT(Devlin et al. (2019)), a lot of excellent works blowout likes Retro-Reader on ALBERT(Zhang et al. (2020)), XLNet + SG-Net Verifier (Zhang et al. (2019)), or just fine-tuning pre-trained language model like ALBERT (Lan et al. (2019)).

Naturally, Extending language models' reading capacity to multi-hop question answering is a challenging problem. WikiHop (Welbl et al. (2018)), ComplexWebQuestions (Talmor & Berant (2018)) and HotpotQA(Yang et al. (2018)) are popular multi-hop reasoning data sets. These data sets require multi-hop reasoning over multiple supporting documents to find the answer. An example from HotpotQA is illustrated in 1. In order to correctly answer the question ("The director of the romantic comedy 'Big Stone Gap' is based in what New York city"), the model first needs to identify P1 as a relevant paragraph, whose title contains keywords that appear in the question ("Big Stone Gap"). S1, the first sentence of P1, is then verified as supporting facts, which leads to the next-hop paragraph P2. From P2, the span "Greenwich Village, New York City" is selected as the predicted answer.

Most existing studies solve the multi-hop task in two directions. The first direction focuses on applying or adapting previous frame work that are successful in single-hop QA tasks to multi-hop QA tasks(e.g. Dhingra et al. (2018), Nishida et al. (2019), Zhong et al. (2019)).

The other direction treats the connectivity of Graph Neural Networks (GNN) as reasoning chain so that multi-hop task is convert to path choosing(or sub-graph) problem or node classifying problem. many prominent works followed this direction (Cao et al. (2019), De Cao et al. (2019), Tu et al. (2019), Ding et al. (2019), Qiu et al. (2019), Asai et al. (2019)). obviously, Graph neural networks have demonstrated their promising potential in many recent works.

Despite of the above achieved success, there are still several limitations of the current approaches on multi-hop QA. First, the entity graph is widely used for predicting answers or extent reasoning path,

*thanks info

Question: What city is the band that recorded Renegade from?
Paragraph 1, Renegade (Styx song): "Renegade" is a 1979 hit song recorded by the American rock band Styx.
Paragraph 2, Styx (band): Styx is an American rock band from Chicago that formed in 1972 and became famous for its albums released in the late 1970s and early 1980s.
Answer: Chicago
Supporting facts: [['Renegade (Styx song)', 0], ['Styx (band)', 0]]

Figure 1: An example from HotpotQA. Under line denotes the bridge entity (unlabeled). "Supporting facts" is the original format in data set.

but is insufficient for finding supporting facts. Entities graph contains few information compared to sentences or paragraphs, relying heavily on it obviously limits the model capacity. Second, almost all existing methods directly work on all documents either by simply concatenating them, regardless of the fact that most context is not related to the question or not helpful in finding the answer. In pretrained language model fine-tuning paradigm, context length is restricted to a fixed number(e.g. 512 or 1024), but few works have been conducted to design a sentence level filter in order to remove redundant context. Motivated by Hierarchical Graph Network (HGN) (Fang et al. (2019)), we propose a two stage model to incorporate the reasoning capacity of HGN and the reading capacity of pretrained language models. the origin work proposed a multi-task learning model, the HGN part and the reading comprehension part share the same context encoding which is generated from BERT, than the model learns how to classify node classes(choose support sentence) and answer span at the same time. we decompose the model to two stage model for the reason that purely fine-tuning the pretrained language model is a better way to fully explore the LM's potential. Meanwhile, our method initialize the node in a different way, we use a simpler [CLS] tokens rather than bi-LSTM.

our model procedure is constructed intuitively. given a question and a set of paragraph (hotpotqa distractor setting):
1. identify support paragraphs and sentences.
2. concatenate all sentences as context.
3. fine-tuning language model to find a answer span in context.

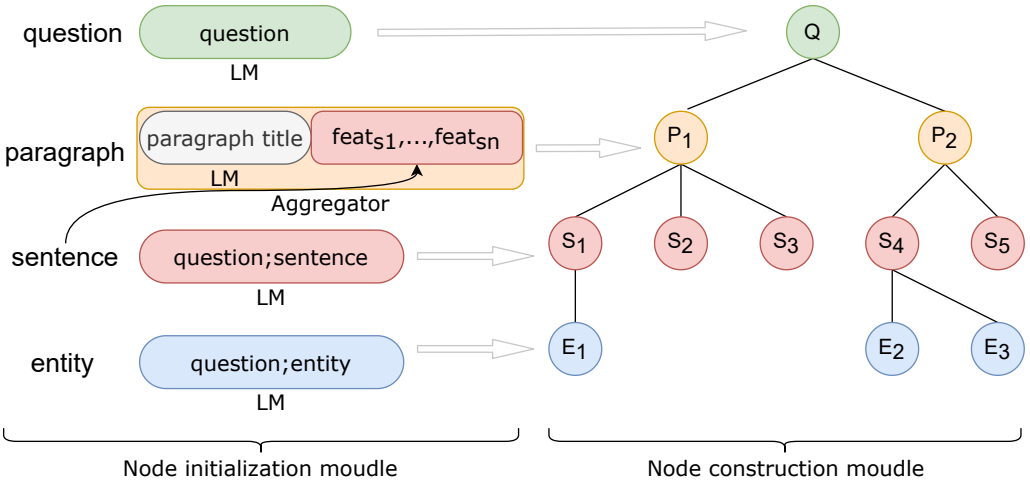

Figure 2: Model architecture of our model. $\texttt{feat}_s$ is the feature vector of a sentence.

details of Hierarchical Graph Network:
1. four type of nodes: question, paragraphs, sentences, and entities (see Figure 2)
2. initialization of nodes: follow [CLS] text1 text2 format, we initialize different type of nodes with different text pairs. see section xxx. 3. seven types of edge. see sections xxx for detail.

our two stage model has the following contributions:
1. taking advantages of Hierarchical Graph Network to select support sentences, we convert multi-hop reasoning to single-hop reading comprehension.
2. explore the potential of pretrained language model for question answering fine-tuning task.

2 RELATED WORK

language model(LM) LM have been performed better than human in machine reading comprehension task, which is a sub-task of LM, since the release of BERT (Devlin et al. (2019)). a mountain of work that attempts to improve BERT have presented explosively. roberta (Liu et al. (2019)) incorporate many training tricks and slightly modify origin loss function; transformer-XL (Dai et al. (2019)) extent the model to variable input sequence with recurrent structure; XLNET(Yang et al. (2019)), which is a upgrade of transformer-XL, creates a permutations attention mask matrix to solve the [MASK] tokens bias; A Lite BERT (ALBERT) (Lan et al. (2019)) incorporates two parameter reduction techniques to accelerate both the train and inference speed. T5 model (Raffel et al. (2019)) is a generative model by introducing a unified framework that converts every language problem into a text-to-text format. benefit by these excellent models, solving questions answering task with transfer learning paradigm is a future tendency.

Graph Neural Network for Multi-hop QA GNN is a powerful tool for reasoning via message passing between neighbourhood. recent studies on multi-hop QA focus on creating graph based on entities. MHQA-GRN (Song et al. (2018)) and Coref-GRN (Dhingra et al. (2018)) construct an entity graph based on co-reference resolution or sliding windows. Entity-GCN (De Cao et al. (2019)) connects different documents via entity mentions. BAG (Cao et al. (2019)) extents biDAF framework to learn graph representations. Cognitive Graph QA (Ding et al. (2019)) mimics human cognitive process, uses iterative generative entities graph to find the reasoning path. Dynamically Fused Graph Network (DFGN) (Qiu et al. (2019)) constructs a dynamic entity graph, where in each reasoning step irrelevant entities are softly masked out, and a fusion module is designed to improve the interaction between the entity graph and the documents.

different from entities graph, our hierarchical graph models all granularities from paragraphs to entities. different from origin HGN, which is a multi-task learning model, our two stage model demonstrate the HGN and LM's capacity separately.

3 HIERARCHICAL GRAPH NETWORK

the Hierarchical Graph Network (HGN) consists of four main components:
(i) Graph Construction Module, through which nodes were created according to the nature structure of data;
(ii) features generations Module, where initial representations of graph nodes are obtained via a pretrained language model encoder;
(iii) Graph Reasoning Module, where graph-attention-based message passing algorithm is applied to jointly update node representations;
(iv) Node Classify Module, which converts choosing support paragraphs task to predicting paragraph nodes.

The following sub-sections describe each component in detail.

3.1 GRAPH CONSTRUCTION

as we say, HGN consists of four types of nodes: questions, paragraphs, sentences, entities. according to the data set structure, one question has a set of paragraphs where one or more support labels in there. In the labeled paragraphs, one or more sentences are labeled 'support', which are necessary context for answering the question, but the answer span may lies on one of them. therefore, we can create questions, paragraphs, sentences node directly. entity nodes come from sentence. we extract all the entities in the sentence and add edges between the sentence node and these entity nodes. entities play roles like bridges, which is defined as hyperlink. we use an external tool to identify and add hyperlinks between sentences and paragraph titles if one entity appears in both of them.

seven different types of edges are defined as follows:
(i) edges between question node and paragraph nodes;
(ii) edges between question node and entity nodes that appears in the question;
(iii) edges between paragraph nodes and their sentence nodes (sentences within the paragraph);
(iv) edges between sentence nodes and their linked paragraph nodes (linked through hyperlinks);
(v) edges between sentence nodes and their corresponding entity nodes (entities appearing in the sentences);
(vi) edges between paragraph nodes;
(vii) edges between sentence nodes that appear in the same paragraph.

3.2 NODE INITIALIZATION

every node is represented by a feature vector. in order to obtain semantic information, we use LM's [CLS] tokens feature as usual does. we denote question text as Q, paragraph title text as P, sentence text as S, entity text as E. so that, features f_x denotes the features vector of node x, where $x \in \{Q, P, S, E\}$.

question node just passing the question raw text to obtain the features is reasonable and effective.

$$f_Q = \text{[CLS] in LM(“[CLS] } Q \text{ [SEP]”)}$$

sentence node it is important to mention that LM has limited max sequence length(e.g.512, 1024), For training a model, inputs dimension has to be a constant number even through using XLNET. it is a crucial limitation, but it is rare that a sentence contains more than 512 tokens. the original HGN model (multi-task learning version) extracts sentences encoding from paragraph encoding, which is made up of sentences, by sentences offsets. paragraph are much more likely to exceed this limitation of token length.

$$f_S = \text{[CLS] in LM(“[CLS] } Q \text{ [SEP] } S \text{ [SEP]”)}$$

paragraph node intuitively, a paragraph is made up of a title and a set of sentences. therefore we simply add the title features and the sentence features.

$$f_P = \text{[CLS]} + \text{sum}(F_{S_i})$$
$$\text{where [CLS] in LM(“[CLS] } P \text{ [SEP]”), } s_i \in P$$

entity node just consider the context of entity name and paragraph title.

$$f_E = \text{[CLS] in LM(“[CLS] } P \text{ [SEP] } E \text{ [SEP]”)}$$

3.3 GRAPH REASONING

after node initialization, the node features are updated via graph neural network. we use a heterogeneous version of Graph Attention Network (GAT) (Velickovic et al. (2018)) to pass message over the hierarchical graph. Specifically, GAT aggregates all neighbors' information with learn-able weights to update a node feature. Formally,

$$h_i^{'} = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W h_j)$$

where $h^{'}$ is the next step hidden states, $W \in \mathbb{R}^{d \times d}$ is a weight matrix, $\sigma(.)$ denotes an activation function, and α_{ij} is the attention coefficients, which is calculated by:

$$\alpha_{ij} = \frac{exp(\sigma(W_{e_{ij}}[h_i; h_j]))}{\sum_{k \in \mathcal{N}_i} exp(\sigma(W_{e_{ik}}[h_i; h_j]))}$$

where $W_{e_{ij}}$ is the weight matrix with respect to the edge type e_{ij} between the i-th and j-th nodes. In a summary, after graph reasoning, we obtain $H' = \{h_0^{'}, h_1^{'}, ..., h_N^{'}\} \in \mathbb{R}^{N \times d}$, which is the updated representations for each node.

3.4 NODE CLASSIFY

the data set provided labeled support paragraphs and sentences, therefore we directly use a 2-layer perceptron to reduce dimension from hidden size to two, converting a two-class choosing problem. it noteworthy that the answer type "comparison" has two option: yes or no. intuitively, answering an question needs to read all the context words, but judging a question type between "comparison" and "word span" only requires the questions text. specifically, using question node features is sufficient, therefore we add a classifier on the question node, where predicted question type will be passed to second stage model, where the final choice would be made, otherwise, the second model will find answer span in tokens sequences. formally, we define three loss terms:

$$
\begin{aligned}
L_1 &= L_{para} + L_{sent} + L_{qtype} \\
where \quad L_{para} &= \mathcal{F}(MLP(h_P), h_P^{ans}) \\
L_{sent} &= \mathcal{F}(MLP(h_S), h_S^{ans}) \\
L_{qtype} &= \mathcal{F}(MLP(h_Q^0), h_Q^{ans})
\end{aligned} \tag{1}
$$

$\mathcal{F}$ denotes a loss function, h denotes the last hidden state and subscript denotes node type, h^0 denotes initial hidden states.

4 LANGUAGE MODEL FINE-TUNING

the second stage model is a Language Model with minimal modification. follow the guidance of T5 (Raffel et al. (2019)), fine-tuning all of the model's parameters can lead to suboptimal results, particularly on low-resource tasks.in the first strategy, we only add a small classifier that was fed into sentence embeddings produced by a fixed pre-trained LM.

The second alternative fine-tuning method we consider is "gradual unfreezing" [Howard and Ruder, 2018]. In gradual unfreezing, the model's parameters are fine-tuned from top to bottom over time. in our setting, the additional header was trained for a number of fix step, then unfreeze the whole attention block gradually.

the third way we tried is adding "adapter layers". adapter layers are additional dense-ReLU dense blocks that are added after each of the preexisting feed-forward networks in each block of the Transformer. such layers have only one hyperparameter: hidden dimension. We experiment with various values for d.

formally, given a set of support sentences(set by a hyper-parameter) predicated by stage one model, the targets of model 2 are:

$$
\begin{aligned}
L_2 &= L_{span} + L_{yesorno} \\
L_{span} &= \mathcal{F}(start_{pre}, start_{ans}) + \mathcal{F}(end_{pre}, end_{ans}) \\
L_{yesorno} &= \mathcal{F}(yesno_{pre}, yesno_{ans})
\end{aligned} \tag{2}
$$

$$
\begin{aligned}
where \quad start_{pre} &= LM_{mod}(S_1, S_2, ..., S_{topN}) \\
end_{pre} &= LM_{mod}(S_1, S_2, ..., S_{topN}; start_{pre}) \\
yesno_{pre} &= MLP([CLS]\ of\ LM_{mod}(S_1, S_2, ..., S_{topN}))
\end{aligned} \tag{3}
$$

where $start_{pre}$, end_{pre}, $yesno_{pre}$ denote logits in corresponding positions, LM_{mod} denotes a kind of fine-tuning method. $topN$ means selecting the top N sentences as support evidences.

Model	Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1
Baseline Model (Yang et al. (2018))	45.60	59.02	20.32	64.49	10.83	40.16
DFGN (Qiu et al. (2019))	56.31	69.69	51.50	81.62	33.62	59.82
HGN (Fang et al. (2019))	66.07	79.36	60.33	87.33	43.57	71.03
Two stage model (ours)	2	3	4	5	6	7

Table 1: Results on the test set of HotpotQA in the Distractor setting.

sentence permutations we notice that LM sum up positional embeddings and word embeddings when sentences fed in, positional embedding is crucial for model to capture sequence information. However, the order of sentences predicted by GNN can not be promised. in this situation, sentences are likely to occur at any kind of orders. hence we permutate the set of sentences to form a set of context paragraphs as training datas.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Dataset HotpotQA is a question answering data set that requires reading multi-sentences across multi-documents to reveal the final answer span. this is constructed in the way that crowd workers are asked to provide a question with multiple documents. the data set also provided golden answers of questions, named support sentence & paragraph and answer span. There are about 90K training samples, 7.4K development samples, and 7.4K test samples. Please refer to the original paper (Yang et al. (2018)) for more details.

HotpotQA presents two tasks: answer span prediction and supporting facts prediction. results are evaluated based on Exact Match (EM) and F1 score of the two tasks. to evaluated the overall performance, Joint EM and F1 scores are used separately. we train our two stage model on the training set, and tune hyperparameters on the development set.

Implementation Details Our implementation is based on the pre-trained language models provided by Transformer library, we use RoBERTa-base for generating graph node features, tokenization, and fine-tuning the question answering model. in graph construction step, we use spacy[1] to recognize entities in sentence and question. according to the statistical data of HotpotQA, 80% questions requires 3 support sentences, thus we set this parameters as one of base line settings. in fine-tuning step, we compare three different methods advised by (t5), and set single-MLP header as base line setting.

Fine-tuning strategies we use three kinds of fine-tuning strategies. firstly, we follow the same fine-tuning procedure as (Devlin et al. (2019)), create a start vector and an end vector during fine-tuning, what is a single-layer MLP, actually. secondly, we consider a method named "gradual unfreezing" (Howard & Ruder (2018)). In gradual unfreezing, more and more of the model's parameters are fine-tuned over time. starting from the last layer (in LM, an self-attention block is considered as the minimum unit), model components are unfrozen step by step or after training for a certain number of updates. in practice, we notice that both BERT (Devlin et al. (2019)) and XLNET (Yang et al. (2019)) augment their training data with additional QA datasets, we only finetune using the provided SQuAD training data. The third method, 'adapter layers'" (Houlsby et al. (2019), Bapna & Firat (2019)), is motivated by the goal of keeping most of the original model fixed while fine-tuning. Adapter layers are additional MLP blocks (dense-active function-dense) that are added after each of the preexisting feed-forward networks in each block of the Transformer. In training procedure, only these MLP blocks are updated. therefore the only hyperparameter is the hidden dimension of the liner layer in the bottom of MLP blocks.

HGN	Sup. precision	Sup. recall	Sup. F1
base model (topN=3)	xx	xx	xx
large model (topN=3)	xx	xx	xx
base model (topN=4)	xx	xx	xx
large model (topN=4)	xx	xx	xx

Table 2: Results of model 1 on the dev set of HotpotQA.

LM/strategy	Ans. precision (EM)	Ans. recall	Ans. F1
single MLP:			
BERT-base	xx	xx	xx
RoBERTa-base	xx	xx	xx
unfreeze last 1 layer:			
BERT-base	xx	xx	xx
RoBERTa-base	xx	xx	xx
unfreeze last 2 layer:			
BERT-base	xx	xx	xx
RoBERTa-base	xx	xx	xx
adapter layer with d=16:			
BERT-base	xx	xx	xx
RoBERTa-base	xx	xx	xx
adapter layer with d=32:			
BERT-base	xx	xx	xx
RoBERTa-base	xx	xx	xx

Table 3: Results of model 2 on the dev set of HotpotQA.

5.2 RESULTS

in 5.2, we show the performance comparison among different models on leaderboard. we show that our method improves more than xx% and xx% absolutely in terms of joint EM and F1 scores over the baseline model. Compared to original HGN work, our model ...

stage 1 model in table2, we demonstrate the performance of HGN. the model reach xx precision and xx recall, it confirms that graph neural network has great potential in modeling reasoning relationship. we also set topN parameters to 4 as comparison, where 4 sentences are predicted as support sentences. in the two stage model frame, we dont have to change stage 2 as it is able to find answer span as long as it appears in the topN sentences.

stage 2 model the 2nd model's performance is shown in table 3, the base model reach xx precision and xx recall. (add more)

5.3 ABLATION STUDIES

In order to better understand how the performance is affected by different part of modules, we conduct several ablation studies on the development set of data. ablation test on LM is the same as comparison of different header of LM fine-tuning task, which has been studied in section xx. therefore in this section, we focus on model 1.

If we remove the edge type and treat all edge types equally, the accuracy and recall drop xx, xx separately. it proves that different types information is important for gnn.

if ... the acc degrades by xx...

[1] https://spacy.io

5.4 RESULTS ANALYSIS

to investigate model deeply, analysis is done based on different reasoning types in the development set. every question belongs to a category, either "bridge" or "comparison", that is provided in data set. "bridge" means answering a question requires reading multi sentences connected by at least one "bridge entity". "comparison" means answer would be inferred from comparing attributes of different entities, We calculate the joint EM and F1 in each categorization and compare ours with the baseline model and the DFGN model under these two reasoning types.

In Table4 ...

6 CONCLUSION

REFERENCES

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. Learning to retrieve reasoning paths over wikipedia graph for question answering. CoRR, abs/1911.10470, 2019. URL http://arxiv.org/abs/1911.10470.

Ankur Bapna and Orhan Firat. Simple, scalable adaptation for neural machine translation. 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), pp. 1538–1548, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1165. URL https://www.aclweb.org/anthology/D19-1165.

Yu Cao, Meng Fang, and Dacheng Tao. BAG: Bi-directional attention entity graph convolutional network for multi-hop reasoning question answering. 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), pp. 357–362, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1032. URL https://www.aclweb.org/anthology/N19-1032.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1171. URL https://www.aclweb.org/anthology/P17-1171.

Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 845–855, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1078. URL https://www.aclweb.org/anthology/P18-1078.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://www.aclweb.org/anthology/P19-1285.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Question answering by reasoning across documents with graph convolutional networks. 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), pp. 2306–2317, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1240. URL https://www.aclweb.org/anthology/N19-1240.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 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), pp. 4171–4186, Minneapolis, Minnesota, June

2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. Neural models for reasoning over multiple mentions using coreference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pp. 42–48, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2007. URL https://www.aclweb.org/anthology/N18-2007.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. Cognitive graph for multi-hop reading comprehension at scale. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2694–2703, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1259. URL https://www.aclweb.org/anthology/P19-1259.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. Hierarchical graph network for multi-hop question answering. arXiv preprint arXiv:1911.03631, 2019.

Yair Feldman and Ran El-Yaniv. Multi-hop paragraph retrieval for open-domain question answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2296–2309, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1222. URL https://www.aclweb.org/anthology/P19-1222.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzkebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of Proceedings of Machine Learning Research, pp. 2790–2799. PMLR, 2019. URL http://proceedings.mlr.press/v97/houlsby19a.html.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1031. URL https://www.aclweb.org/anthology/P18-1031.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://www.aclweb.org/anthology/P17-1147.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. Answering while summarizing: Multi-task learning for multi-hop QA with evidence extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2335–2345, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1225. URL https://www.aclweb.org/anthology/P19-1225.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. Dynamically fused graph network for multi-hop reasoning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 6140–6150, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1617. URL https://www.aclweb.org/anthology/P19-1617.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://www.aclweb.org/anthology/D16-1264.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://www.aclweb.org/anthology/P18-2124.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017. URL https://openreview.net/forum?id=HJ0UKP9ge.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks, 2018.

Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 641–651, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1059. URL https://www.aclweb.org/anthology/N18-1059.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2704–2713, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1260. URL https://www.aclweb.org/anthology/P19-1260.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. Transactions of the Association for Computational Linguistics, 6:287–302, 2018. doi: 10.1162/tacl_a_00021. URL https://www.aclweb.org/anthology/Q18-1021.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://www.aclweb.org/anthology/D18-1259.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems, pp. 5754–5764, 2019.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. Sg-net: Syntax-guided machine reading comprehension, 2019.

Zhuosheng Zhang, Junjie Yang, and Hai Zhao. Retrospective reader for machine reading comprehension, 2020.

Victor Zhong, Caiming Xiong, Nitish Shirish Keskar, and Richard Socher. Coarse-grain fine-grain coattention network for multi-evidence question answering. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019. URL https://openreview.net/forum?id=Syl7OsRqY7.

A Appendix

You may include other additional sections here.