Is Graph Structure Necessary for Multi-hop Reasoning?

Nan Shao[†], **Yiming Cui**[‡†], **Ting Liu**[‡], **Shijin Wang**[†§], **Guoping Hu**[†]

[†]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research, China
[‡]Research Center for Social Computing and Information Retrieval (SCIR),
Harbin Institute of Technology, Harbin, China
[§]iFLYTEK AI Research (Hebei), Langfang, China
[†§]{nanshao,ymcui,sjwang3,gphu}@iflytek.com
[‡]{ymcui,tliu}@ir.hit.edu.cn

Abstract

Recently, many works attempt to model texts as graph structure and introduce graph neural networks to deal with it on many NLP tasks. In this paper, we investigate whether graph structure is necessary for multi-hop reasoning tasks and what role it plays. Our analysis is centered on HotpotQA. We use the state-of-the-art published model, Dynamically Fused Graph Network (DFGN), as our baseline. By directly modifying the pre-trained model, our baseline model gains a large improvement and significantly surpass both published and unpublished works. Ablation experiments established that, with the proper use of pre-trained models, graph structure may not be necessary for multi-hop reasoning. We point out that both the graph structure and the adjacency matrix are task-related prior knowledge, and graph-attention can be considered as a special case of self-attention. Experiments demonstrate that graph-attention or the entire graph structure can be replaced by self-attention or Transformers, and achieve similar results to the previous state-of-the-art model achieved.

1 Introduction

Different from one-hop question answering, where the answer can be derived from a single sentence in a single paragraph, more and more studies focus on multi-hop reasoning across multiple documents or paragraphs (Welbl et al., 2018; Talmor and Berant, 2018; Yang et al., 2018). To solve this problem, the majority of existing studies constructed a graph structure according to co-occurrence relations of entities that scattered across multiple sentences or paragraphs. Dhingra et al. (2018) and Song et al. (2018) designed a DAG-styled recurrent layer to model the relations between entities. De Cao et al. (2019) first used GCN (Kipf and Welling, 2017) to tackle entity graph. Qiu et al. (2019) proposed a dynamic entity graph for span-based multi-hop

reasoning tasks. Tu et al. (2019b) extended the entity graph to a heterogeneous graph by introducing document nodes and query nodes.

Previous works argue that a fancy graph structure is a vital part of their models and demonstrate that by ablation experiments. However, in experiments, we find when we use the pre-trained models in the fine-tuning approach, removing entire graph structure may not hurt the final results. Therefore, in this paper, we aimed to answer the following question: **How much does graph structure contribute to multi-hop reasoning?**

To answer the question above, we choose the widely used multi-hop reasoning benchmark, HotpotQA (Yang et al., 2018), as our testbed. We fine-tune the pre-trained model in DFGN and achieve state-of-the-art performance on HotpotQA leaderboard.

In the subsequent ablation experiments, we find that the graph structure can play an important role only when the pre-trained models are used in a feature-based manner. While the pre-trained models are used in the fine-tuning approach, the graph structure may not be helpful.

To explain the results of experiments, we point out that graph-attention (Veličković et al., 2018) is a special case of self-attention. The adjacency matrix based on manually defined rules and the graph structure can be regarded as prior knowledge, which could be learned by self-attention or Transformer (Vaswani et al., 2017). We design the experiments to show when we model text as an entity graph, both graph-attention and self-attention can achieve comparable results. When we treat texts as a sequence structure, only a 2-layer Transformer could achieve similar results as DFGN.

Although our experiments are performed on a multi-hop reasoning task, the conclusions of this paper may also apply to some models with graph structure on other NLP tasks.

Model	Joint	
	EM	F1
Baseline (Yang et al., 2018)	10.83	40.16
QFE (Nishida et al., 2019)	34.63	59.61
DFGN (Qiu et al., 2019)	33.62	59.82
SAE (Tu et al., 2019a)	38.81	64.96
TAP2 (Glass et al., 2019)	39.77	69.12
EPS+BERT	42.47	70.48
HGN (Fang et al., 2019)	43.57	71.03
Our Model	**44.67**	**72.73**

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

2 The Approach

We use the open-source implementation of DFGN[1] (Qiu et al., 2019), the published state-of-the-art model. We modify the use of the pre-trained model and the retriever model.

2.1 Model Description

Retriever. We use the RoBERTa-large (Liu et al., 2019) to calculate the relevant score between the query and each document in an example. We filter the document whose relevant score is less than 0.1, and the maximum number of selected documents is 3. Selected documents are concatenated as context and fed into the encoding layer.

Encoding Layer. We concatenate the query $\mathbf{Q}$ and context $\mathbf{C}$ and feed the sequence into another RoBERTa model. The results are further fed into a bi-attention layer (Seo et al., 2016) to obtain the representations from the encoding layer.

Graph Fusion Block. Given context representations $\mathbf{C}_{t-1}$ at hop $t-1$, the tokens representations are passed into a mean-max pooling layer to get nodes representations in entity graph $\mathbf{H}_{t-1} \in \mathbb{R}^{2d \times N}$, where N is the number of entity. After that, a graph-attention layer is applied to update nodes representations in the entity graph:

$$\beta_{i,j}^{(t)} = \text{LeakyReLU}(\mathbf{W}_t^\top [\mathbf{h}_i^{(t-1)}, \mathbf{h}_j^{(t-1)}]) \quad (1)$$

$$\alpha_{i,j}^{(t)} = \frac{\exp(\beta_{i,j}^{(t)})}{\sum_{k \in \mathcal{N}_i} \exp(\beta_{i,k}^{(t)})} \quad (2)$$

$$\mathbf{h}_i^{(t)} = \text{ReLU}(\sum_{k \in \mathcal{N}_i} \alpha_{i,k}^{(t)} \mathbf{h}_k^{(t-1)}) \quad (3)$$

where $\mathcal{N}_i$ is the set of neighbors of node i. We follow the same Graph2Doc module as Qiu et al.

[1] https://github.com/woshiyyya/DFGN-pytorch

(2019) to transform the nodes representations into the tokens representations. Besides, there are several extra modules in the graph fusion block, including query-entity attention, query update mechanism, and weak supervision.

Prediction Layer. We follow the same cascade structure as Qiu et al. (2019) to predict supporting sentences, the start/end position of the answer, and the answer-type.

2.2 Model Results

In Table 1, we show the performance comparison with different models on the blind test set of HotpotQA. Our baseline model outperforms both published and unpublished works on each metric.

3 Graph Structure May Not Be Necessary

In order to analyze how much the graph structure contribute to the entire model, we perform a set of ablation experiments. We remove the whole graph fusion block, and the outputs of the pre-trained model are directly fed into the prediction layer. By the reason that the main difference between our baseline model and DFGN is that we use a large pre-trained model in the fine-tuning approach instead of the feature-based approach, we perform the experiments in two different settings.

The results are shown in Table 2. By using the fine-tuning approach, model with and without graph fusion block can reach equal results. When we fix parameters of the pre-trained model, the performance significantly degrades by 9% for EM and 10% for F1. If we further remove graph fusion block, both EM and F1 drop 4%.

Taken together, only when pre-trained models are used in the feature-based approach, graph neural networks can play an important role. Nevertheless, if pre-trained models are used as a fine-tuning approach, which is a common practice, graph structure does not contribute to the final results. In other words, the graph structure may not be necessary for multi-hop reasoning.

4 Understanding Graph Structure

Experimental results in Section 3 imply that self-attention or Transformer may have superiority in multi-hop reasoning tasks. To understand this, in this section, we will first discuss the connection between graph structure, graph-attention, and self-attention. We then evaluate the effect of replacing
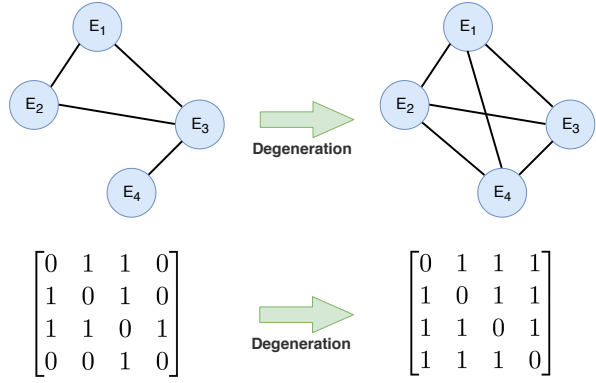
Figure 1: Entities in raw texts are modeled as an entity graph and handled by graph attention networks. When the entity graph are fully connected, a graph-attention layer will degenerate into a vanilla self-attention layer.

Setting	Joint EM	Joint F1
Baseline (Fine-tuning)	45.91	**73.93**
w/o Graph	**45.98**	73.78
Baseline (Feature-based)	36.45	63.75
w/o Graph	32.26	59.76

Table 2: Ablation of graph structure under different settings.

graph-attention or graph structure by self-attention or Transformer.

4.1 Graph Attention Versus Self Attention

The key to solve the multi-hop reasoning problems is to find the corresponding entity in the original text through the query. Then one or more reasoning paths are constructed from these start entities toward other identical or co-occurring entities. As shown in Figure 1, previous works usually extract entities from multiple paragraphs and model these entities as an entity graph. The adjacency matrix is constructed by manually defined rules, which usually the co-occurrence relationship of entities. From this point of view, both the graph structure and the adjacency matrix can be regarded as task-related prior knowledge. The entity graph structure determines that the model can only do reasoning based on entities, and the adjacency matrix assists the model to ignore non-adjacent nodes in a hop. However, it is probably that the model without any prior knowledge can still learn the entity-entity attention paradigm.

In addition, considering Eq.1-3, it is easy to find that graph-attention has a similar form as self-attention. In this paper, we consider that the graph-attention as a special case of self-attention. In for-

ward propagation, each node in the entity graph calculates attention scores with other connected nodes. As shown in Figure 1, graph-attention will degenerate into a vanilla self-attention layer when the nodes in the graph are fully connected.

4.2 Experimental Setup

According to discussion above, we aimed to evaluate whether the graph structure with an adjacency matrix is superior to self-attention.

To this end, we use the model described in Section 2 as our baseline model. The pre-trained model in the baseline model is used in the feature-based approach. Several different modules are added between the encoding layer and the prediction layer.
Model With Graph Structure. We apply graph-attention or self-attention on the entity graph and compare the difference in the final results. Each entity representation is obtained from a mean-pooling layer and fed into a self-attention layer or graph-attention layer. In order to make a fair comparison, we choose the self-attention that has the same form with graph-attention. The main difference is that the self-attention does not keep an adjacency matrix as prior knowledge and the entities in the graph are fully connected. At each time step, we use Graph2Doc module to transform entities representation into tokens representation. Moreover, we define that the density of a binary matrix is the percentage of '1' in it. We sort each example in development set by the density of its adjacency matrix and divide them by different quantiles. We evaluate how different density of the adjacency matrix affect the final results.
Model Without Graph Structure. In this experiments, we verify whether the whole graph structure

Quantile	0.2	0.4	0.6	0.8	1.0	avg
Density	18.7	23.6	29.0	36.8	100	28.8

Table 3: The density of the adjacency matrix at different quantiles.

can be replaced by Transformers. We directly feed the context representations from the encoding layer into the Transformers. Moreover, we also explored how adding an adjacency matrix as the mask matrix to the transformers affects the final results. Since the adjacency matrix cannot be directly applied in sequence structure, we construct a mask matrix to restrict the model can only attend from the tokens of one entity to the tokens of other entities connected to it.

In all experiments, the number of layers of different modules is two, and the hidden dimensions are set to 300 with an initial learning rate of 2e-4.

4.3 Experimental Results

The results of the experiments are shown in Table 4. Compared with the baseline, the model with the graph fusion block obtains a significant advantage. We add the entity graph with self-attention to the baseline model, and the final results significantly improved. Compared with self-attention, graph-attention does not show clear advantage. The density of examples at different quantile are shown in Table 3, the adjacency matrix in multi-hop reasoning task is relatively dense, which may cause that graph-attention can not make a significant difference. The results of graph-attention and self-attention in the different intervals of density are shown in Figure 2. Despite the different density of the adjacency matrix, graph-attention consistently achieves similar results as self-attention. This signifies that self-attention can learn to ignore irrelevant entities. Besides, examples with a more dense adjacency matrix are simpler for both graph-attention and self-attention, this probably because these adjacency matrices are constructed from shorter documents.

The Transformer shows a powerful reasoning ability. Only stacking two layers of the Transformer can achieve comparable results as the sophisticated DFGN. Adding the adjacency matrix as a mask matrix in the Transformer will cause the results to drop significantly. We consider the reason is that the Transformer has the capacity to learn the pattern of attention from one entity to another. Restricting

Setting	Joint EM	Joint F1
Baseline	32.26	59.76
+ Graph Fusion Block	**36.45**	63.75
+ Self Attention	35.41	61.77
+ Graph Attention	35.79	61.91
+ Transformer	36.23	**63.82**
+ Masked Transformer	35.19	62.48

Table 4: Performance comparison in terms of joint EM and F1 scores under different module settings.
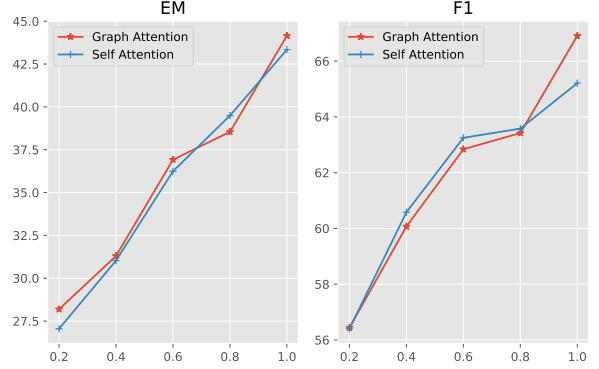

Figure 2: Results of graph-attention and self-attention on examples with different density of the adjacency matrix.

the model through the mask matrix can only attend that the positions of entities will cause the information contained in other positions of non-entities to be lost.

5 Conclusions

This study set out to investigate whether graph structure is necessary for multi-hop reasoning tasks and what role it plays. We established that with the proper use of pre-trained models, graph structure may not be necessary for multi-hop reasoning. In addition, we point out that the adjacency matrix and graph structure can be regarded as some kind of task-related prior knowledge. We find both graph-attention and graph structure can be replaced by self-attention or Transformer.

Our results suggest that the ablation experiments of future works should be done under the circumstance that parameters of the pre-trained model are trainable, or should compare directly with the results of the plain pre-trained model on the same task. Future works introducing graph structure into NLP tasks should explain the necessity and the differences from widely used modules such as self-attention or Transformer.

References

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. 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)*, pages 2306–2317, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. 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)*, pages 42–48, New Orleans, Louisiana. Association for Computational Linguistics.

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

Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, GP Bhargav, Dinesh Garg, and Avirup Sil. 2019. Span selection pretraining for question answering. *arXiv preprint arXiv:1909.04120.*

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR).*

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 preprint arXiv:1907.11692.*

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. 2019. 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*, pages 2335–2345, Florence, Italy. Association for Computational Linguistics.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150, Florence, Italy. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations (ICLR).*

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv preprint arXiv:1809.02040.*

Alon Talmor and Jonathan Berant. 2018. 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)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2019a. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. *arXiv preprint arXiv:1911.00484.*

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019b. 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*, pages 2704–2713, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

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

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.