Hierarchical Graph Network for Multi-hop Question Answering

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, Jingjing Liu
Microsoft Dynamics 365 AI Research
{yuwfan,siqi.sun,zhe.gan,rohit.pillai,shuohang.wang,jingjl}@microsoft.com

Abstract

In this paper, we present Hierarchical Graph Network (HGN) for multi-hop question answering. To aggregate clues from scattered texts across multiple paragraphs, a hierarchical graph is created by constructing nodes on different levels of granularity (questions, paragraphs, sentences, entities), the representations of which are initialized with pre-trained contextual encoders. Given this hierarchical graph, the initial node representations are updated through graph propagation, and multi-hop reasoning is performed via traversing through the graph edges for each subsequent sub-task (e.g., paragraph selection, supporting facts extraction, answer prediction). By weaving heterogeneous nodes into an integral unified graph, this hierarchical differentiation of node granularity enables HGN to support different question answering sub-tasks simultaneously. Experiments on the HotpotQA benchmark demonstrate that the proposed model achieves new state of the art, outperforming existing multi-hop QA approaches.[1]

1 Introduction

In contrast to one-hop question answering (Rajpurkar et al., 2016; Trischler et al., 2016; Lai et al., 2017) where answers can be derived from a single paragraph (Wang and Jiang, 2017; Seo et al., 2017; Liu et al., 2018; Devlin et al., 2019), many recent studies on question answering focus on multi-hop reasoning across multiple documents or paragraphs. Popular tasks include WikiHop (Welbl et al., 2018), ComplexWebQuestions (Talmor and Berant, 2018), and HotpotQA (Yang et al., 2018).

An example from HotpotQA is illustrated in Figure 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 is required to first identify *P1* as a relevant paragraph, whose title contains the keywords that appear in the question ("*Big Stone Gap*"). *S1*, the first sentence of *P1*, is then chosen by the model as a supporting fact that leads to the next-hop paragraph *P2*. Lastly, from *P2*, the span "*Greenwich Village, New York City*" is selected as the predicted answer.

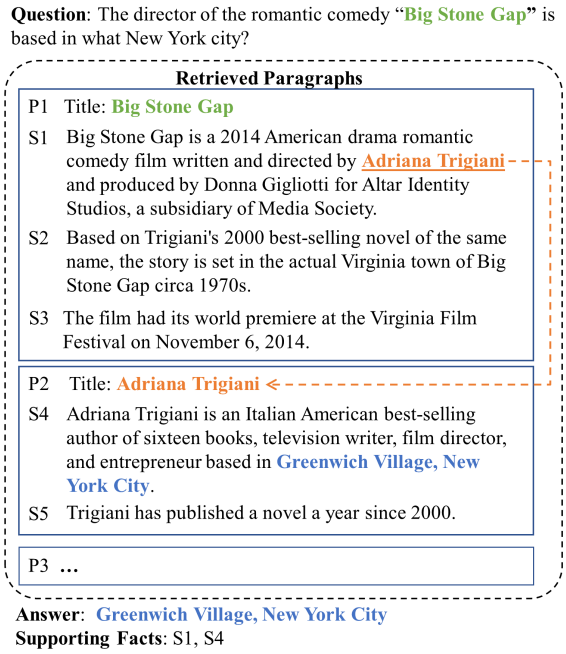

Figure 1: An example of multi-hop question answering from HotpotQA. The model needs to identify relevant paragraphs, determine supporting facts, and then predict the answer correctly.

Most existing studies use a retriever to find paragraphs that contain the right answer to the question (*P1* and *P2* in this case). A Machine Reading Comprehension (MRC) model is then applied to the selected paragraphs for answer prediction (Nishida et al., 2019; Min et al., 2019b). However, even after successfully identifying a reasoning chain through multiple paragraphs, it still remains a critical challenge how to aggregate evidence from scattered

[1] Code will be released at https://github.com/yuwfan/HGN.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 8823–8838,
November 16–20, 2020.

sources on different granularity levels (e.g., paragraphs, sentences, entities) for joint answer and supporting facts prediction.

To better leverage fine-grained evidences, some studies apply entity graphs through query-guided multi-hop reasoning. Depending on the characteristics of the dataset, answers can be selected either from entities in the constructed entity graph (Song et al., 2018; Dhingra et al., 2018; De Cao et al., 2019; Tu et al., 2019; Ding et al., 2019), or from spans in documents by fusing entity representations back into token-level document representation (Xiao et al., 2019). However, the constructed graph is mostly used for answer prediction only, while insufficient for finding supporting facts. Also, reasoning through a simple entity graph (Ding et al., 2019) or paragraph-entity hybrid graph (Tu et al., 2019) lacks the ability to support complicated questions that require multi-hop reasoning.

Intuitively, given a question that requires multiple hops through a set of documents to reach the right answer, a model needs to: (i) identify paragraphs relevant to the question; (ii) determine strong supporting evidence in those paragraphs; and (iii) pinpoint the right answer following the garnered evidence. To this end, Graph Neural Network with its inherent message passing mechanism that can pass on multi-hop information through graph propagation, has great potential of effectively predicting both supporting facts and answer simultaneously for complex multi-hop questions.

Motivated by this, we propose a **H**ierarchical **G**raph **N**etwork (HGN) for multi-hop question answering, which empowers joint answer/evidence prediction via multi-level fine-grained graphs in a hierarchical framework. Instead of only using entities as nodes, for each question we construct a hierarchical graph to capture clues from sources with different levels of granularity. Specifically, four types of graph node are introduced: *questions*, *paragraphs*, *sentences* and *entities* (see Figure 2). To obtain contextualized representations for these hierarchical nodes, large-scale pre-trained language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) are used for contextual encoding. These initial representations are then passed through a Graph Neural Network for graph propagation. The updated node representations are then exploited for different sub-tasks (e.g., paragraph selection, supporting facts prediction, entity prediction). Since answers may not be entities in the graph, a span prediction module is also introduced for final answer prediction.

The main contributions of this paper are threefold: (i) We propose a Hierarchical Graph Network (HGN) for multi-hop question answering, where heterogeneous nodes are woven into an integral hierarchical graph. (ii) Nodes from different granularity levels mutually enhance each other for different sub-tasks, providing effective supervision signals for both supporting facts extraction and answer prediction. (iii) On the HotpotQA benchmark, the proposed model achieves new state of the art in both Distractor and Fullwiki settings.

2 Related Work

Multi-Hop QA Multi-hop question answering requires a model to aggregate scattered pieces of evidence across multiple documents to predict the right answer. WikiHop (Welbl et al., 2018) and HotpotQA (Yang et al., 2018) are two recent datasets designed for this purpose. Existing work on HotpotQA Distractor setting focuses on converting the multi-hop reasoning task into single-hop subproblems. Specifically, QFE (Nishida et al., 2019) regards evidence extraction as a query-focused summarization task, and reformulates the query in each hop. DecompRC (Min et al., 2019b) decomposes a compositional question into simpler sub-questions and leverages single-hop MRC models to answer the sub-questions. A neural modular network is also proposed in Jiang and Bansal (2019b), where neural modules are dynamically assembled for more interpretable multi-hop reasoning. Recent studies (Chen and Durrett, 2019; Min et al., 2019a; Jiang and Bansal, 2019a) have also studied the multi-hop reasoning behaviors that models have learned in the task.

Graph Neural Network Recent studies on multi-hop QA also build graphs based on entities and reasoning over the constructed graph using graph neural networks (Kipf and Welling, 2017; Veličković et al., 2018). 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) considers three different types of edges that connect different entities in the entity graph. HDE-Graph (Tu et al., 2019) enriches information in the entity graph by adding document nodes and creating interactions among documents, entities and answer candidates. Cognitive Graph QA (Ding

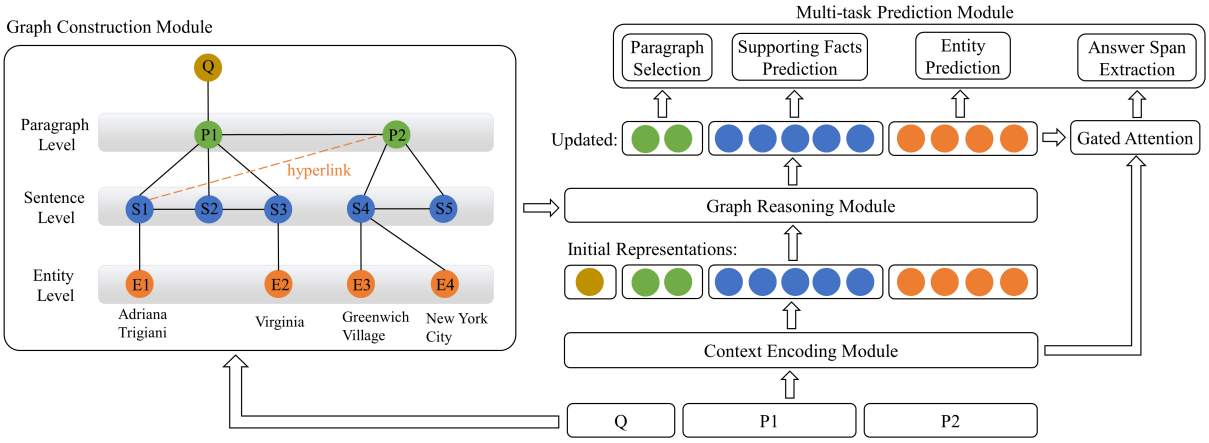

Figure 2: Model architecture of Hierarchical Graph Network. The constructed graph corresponds to the example in Figure 1. Green, blue, orange, and brown colors represent paragraph (P), sentence (S), entity (E), and question (Q) nodes, respectively. Some entities and hyperlinks are omitted for simplicity.

et al., 2019) employs an MRC model to predict answer spans and possible next-hop spans, and then organizes them into a cognitive graph. DFGN (Xiao 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 documents.

More recently, SAE (Tu et al., 2020) defines three types of edge in the sentence graph based on the named entities and noun phrases appearing in the question and sentences. C2F Reader (Shao et al., 2020) uses graph attention or self-attention on entity graph, and argues that this graph may not be necessary for multi-hop reasoning. Asai et al. (2020) proposes a new graph-based recurrent method to find evidence documents as reasoning paths, which is more focused on information retrieval. Different from the above methods, our proposed model constructs a hierarchical graph, effectively exploring relations on different granularities and employing different nodes to perform different tasks.

Hierarchical Coarse-to-Fine Modeling Previous work on hierarchical modeling for question answering is mainly based on a coarse-to-fine framework. Choi et al. (2017) proposes to use reinforcement learning to first select relevant sentences and then produce answers from those sentences. Min et al. (2018) investigates the minimal context required to answer a question, and observes that most questions can be answered with a small set of sentences. Swayamdipta et al. (2018) constructs lightweight models and combines them into a cascade structure to extract the answer. Zhong et al. (2019) proposes to use hierarchies of co-attention and self-attention to combine information from evidence across multiple documents. Different from the above methods, our proposed model organizes different granularities in a hierarchical manner and leverages graph neural network to obtain the representations for different downstream tasks.

3 Hierarchical Graph Network

As illustrated in Figure 2, the proposed Hierarchical Graph Network (HGN) consists of four main components: (i) Graph Construction Module (Sec. 3.1), through which a hierarchical graph is constructed to connect clues from different sources; (ii) Context Encoding Module (Sec. 3.2), where initial representations of graph nodes are obtained via a RoBERTa-based encoder; (iii) Graph Reasoning Module (Sec. 3.3), where graph-attention-based message passing algorithm is applied to jointly update node representations; and (iv) Multi-task Prediction Module (Sec. 3.4), where multiple subtasks, including paragraph selection, supporting facts prediction, entity prediction, and answer span extraction, are performed simultaneously.

3.1 Graph Construction

The hierarchical graph is constructed in two steps: (i) identifying relevant multi-hop paragraphs; and (ii) adding edges representing connections between sentences/entities within the selected paragraphs.

Paragraph Selection We first retrieve paragraphs whose titles match any phrases in the question (title matching). In addition, we train a para-

graph ranker based on a pre-trained RoBERTa encoder, followed by a binary classification layer, to rank the probabilities of whether the input paragraphs contain the ground-truth supporting facts. If multiple paragraphs are found by title matching, only two paragraphs with the highest ranking scores are selected. If title matching returns no results, we further search for paragraphs that contain entities appearing in the question. If this also fails, the paragraph ranker will select the paragraph with the highest ranking score. The number of selected paragraphs in the first-hop is at most 2.

Once the first-hop paragraphs are identified, the next step is to find facts and entities within the paragraphs that can lead to other relevant paragraphs (*i.e,*, the second hop). Instead of relying on entity linking, which could be noisy, we use hyperlinks (provided by Wikipedia) in the first-hop paragraphs to discover second-hop paragraphs. Once the links are selected, we add edges between the sentences containing these links (source) and the paragraphs that the hyperlinks refer to (target), as illustrated by the dashed orange line in Figure 2. In order to allow information flow from both directions, the edges are considered as bidirectional.

Through this two-hop selection process, we are able to obtain several candidate paragraphs. In order to reduce introduced noise during inference, we use the paragraph ranker to select paragraphs with top-N ranking scores in each step.

Nodes and Edges Paragraphs are comprised of sentences, and each sentence contains multiple entities. This graph is naturally encoded in a hierarchical structure, and also motivates how we construct the hierarchical graph. For each paragraph node, we add edges between the node and all the sentences in the paragraph. For each sentence node, we extract all the entities in the sentence and add edges between the sentence node and these entity nodes. Optionally, edges between paragraphs and edges between sentences can also be included in the final graph.

Each type of these nodes captures semantics from different information sources. Thus, the hierarchical graph effectively exploits the structural information across all different granularity levels to learn fine-grained representations, which can locate supporting facts and answers more accurately than simpler graphs with homogeneous nodes.

An example hierarchical graph is illustrated in Figure 2. We define different types of edges as follows: (i) edges between question node and paragraph nodes; (ii) edges between question node and its corresponding entity nodes (entities appearing in the question, not shown for simplicity); (iii) edges between paragraph nodes and their corresponding 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; and (vii) edges between sentence nodes that appear in the same paragraph. Note that a sentence is only connected to its previous and next neighboring sentence. The final graph consists of these seven types of edges as well as four types of nodes, which link the question to paragraphs, sentences, and entities in a hierarchical way.

3.2 Context Encoding

Given the constructed hierarchical graph, the next step is to obtain the initial representations of all the graph nodes. To this end, we first combine all the selected paragraphs into context C, which is concatenated with the question Q and fed into pre-trained Transformer RoBERTa, followed by a bi-attention layer (Seo et al., 2017). We denote the encoded question representation as $\mathbf{Q} = \{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_{m-1}\} \in \mathbb{R}^{m\times d}$, and the encoded context representation as $\mathbf{C} = \{\mathbf{c}_0, \mathbf{c}_1, ..., \mathbf{c}_{n-1}\} \in \mathbb{R}^{n\times d}$, where m, n are the length of the question and the context, respectively. Each $\mathbf{q}_i$ and $\mathbf{c}_j \in \mathbb{R}^d$.

A shared BiLSTM is applied on top of the context representation $\mathbf{C}$, and the representations of different nodes are extracted from the output of the BiLSTM, denoted as $\mathbf{M} \in \mathbb{R}^{n\times 2d}$. For entity/sentence/paragraph nodes, which are spans of the context, the representation is calculated from: (i) the hidden state of the backward LSTM at the start position, and (ii) the hidden state of the forward LSTM at the end position. For the question node, a max-pooling layer is used to obtain its representation. Specifically,

$$\begin{aligned}
\mathbf{p}_i &= \text{MLP}_1\left(\left[\mathbf{M}[P^{(i)}_{start}][d:]; \mathbf{M}[P^{(i)}_{end}][:d]\right]\right)\\
\mathbf{s}_i &= \text{MLP}_2\left(\left[\mathbf{M}[S^{(i)}_{start}][d:]; \mathbf{M}[S^{(i)}_{end}][:d]\right]\right)\\
\mathbf{e}_i &= \text{MLP}_3\left(\left[\mathbf{M}[E^{(i)}_{start}][d:]; \mathbf{M}[E^{(i)}_{end}][:d]\right]\right)\\
\mathbf{q} &= \text{max-pooling}(\mathbf{Q})\,, \qquad (1)
\end{aligned}$$

where $P^{(i)}_{start}$, $S^{(i)}_{start}$, and $E^{(i)}_{start}$ denote the start

position of the i-th paragraph/sentence/entity node. Similarly, $P^{(i)}_{end}$, $S^{(i)}_{end}$, and $E^{(i)}_{end}$ denote the corresponding end positions. MLP$(\cdot)$ denotes an MLP layer, and $[;]$ denotes the concatenation of two vectors. As a summary, after context encoding, each $\mathbf{p}_i$, $\mathbf{s}_i$, and $\mathbf{e}_i \in \mathbb{R}^d$, serves as the representation of the i-th paragraph/sentence/entity node. The question node is represented as $\mathbf{q} \in \mathbb{R}^d$.

3.3 Graph Reasoning

After context encoding, HGN performs reasoning over the hierarchical graph, where the contextualized representations of all the graph nodes are transformed into higher-level features via a graph neural network. Specifically, let $\mathbf{P} = \{\mathbf{p}_i\}_{i=1}^{n_p}$, $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^{n_s}$, and $\mathbf{E} = \{\mathbf{e}_i\}_{i=1}^{n_e}$, where n_p, n_s and n_e denote the number of paragraph/sentence/entity nodes in a graph. In experiments, we set $n_p = 4$, $n_s = 40$ and $n_e = 60$ (padded where necessary), and denote $\mathbf{H} = \{\mathbf{q}, \mathbf{P}, \mathbf{S}, \mathbf{E}\} \in \mathbb{R}^{g \times d}$, where $g = n_p + n_s + n_e + 1$, and d is the feature dimension of each node.

For graph propagation, we use Graph Attention Network (GAT) (Veličković et al., 2018) to perform message passing over the hierarchical graph. Specifically, GAT takes all the nodes as input, and updates node feature $\mathbf{h}'_i$ through its neighbors $\mathcal{N}_i$ in the graph. Formally,

$$\mathbf{h}'_i = \text{LeakyRelu}\Big(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{h}_j \mathbf{W} \Big), \quad (2)$$

where $\mathbf{h}_j$ is the j^{th} vector from $\mathbf{H}$, $\mathbf{W} \in \mathbb{R}^{d \times d}$ is a weight matrix[2] to be learned, and α_{ij} is the attention coefficients, which can be calculated by:

$$\alpha_{ij} = \frac{\exp(f([\mathbf{h}_i; \mathbf{h}_j]\mathbf{w}_{e_{ij}}))}{\sum_{k \in \mathcal{N}_i} \exp(f([\mathbf{h}_i; \mathbf{h}_k]\mathbf{w}_{e_{ik}}))}, \quad (3)$$

where $\mathbf{w}_{e_{ij}} \in \mathbb{R}^{2d}$ is the weight vector corresponding to the edge type e_{ij} between the i-th and j-th nodes, and $f(\cdot)$ denotes the LeakyRelu activation function. In a summary, after graph reasoning, we obtain $\mathbf{H}' = \{\mathbf{h}'_0, \mathbf{h}'_1, \ldots, \mathbf{h}'_g\} \in \mathbb{R}^{g \times d}$, from which the updated representations for each type of node can be obtained, *i.e.*, $\mathbf{P}' \in \mathbb{R}^{n_p \times d}$, $\mathbf{S}' \in \mathbb{R}^{n_s \times d}$, $\mathbf{E}' \in \mathbb{R}^{n_e \times d}$, and $\mathbf{q}' \in \mathbb{R}^d$.

Gated Attention The graph information will further contribute to the context information for answer span extraction. We merge the context representation $\mathbf{M}$ and the graph representation $\mathbf{H}'$ via a gated attention mechanism:

$$\begin{aligned} \mathbf{C} &= \text{Relu}(\mathbf{M}\mathbf{W}_m) \cdot \text{Relu}(\mathbf{H}'\mathbf{W}'_m)^{\text{T}} \\ \bar{\mathbf{H}} &= \text{Softmax}(\mathbf{C}) \cdot \mathbf{H}' \\ \mathbf{G} &= \sigma([\mathbf{M}; \bar{\mathbf{H}}]\mathbf{W}_s) \cdot \text{Tanh}([\mathbf{M}; \bar{\mathbf{H}}]\mathbf{W}_t), \end{aligned} \quad (4)$$

where $\mathbf{W}_m \in \mathbb{R}^{2d \times 2d}$, $\mathbf{W}'_m \in \mathbb{R}^{2d \times 2d}$, $\mathbf{W}_s \in \mathbb{R}^{4d \times 4d}$, $\mathbf{W}_t \in \mathbb{R}^{4d \times 4d}$ are weight matrices to learn. $\mathbf{G} \in \mathbb{R}^{n \times 4d}$ is the gated representation which will be used for answer span extraction.

3.4 Multi-task Prediction

After graph reasoning, the updated node representations are used for different sub-tasks: (i) paragraph selection based on *paragraph* nodes; (ii) supporting facts prediction based on *sentence* nodes; and (iii) answer prediction based on *entity* nodes and context representation $\mathbf{G}$. Since the answers may not reside in entity nodes, the loss for entity node only serves as a regularization term.

In our HGN model, all three tasks are jointly performed through multi-task learning. The final objective is defined as:

$$\begin{aligned} \mathcal{L}_{joint} = \mathcal{L}_{start} + \mathcal{L}_{end} + \lambda_1 \mathcal{L}_{para} + \lambda_2 \mathcal{L}_{sent} \\ + \lambda_3 \mathcal{L}_{entity} + \lambda_4 \mathcal{L}_{type}, \end{aligned} \quad (5)$$

where λ_1, λ_2, λ_3, and λ_4 are hyper-parameters, and each loss function is a cross-entropy loss, calculated over the logits (described below).

For both paragraph selection ($\mathcal{L}_{para}$) and supporting facts prediction ($\mathcal{L}_{sent}$), we use a two-layer MLP as the binary classifier:

$$\mathbf{o}_{sent} = \text{MLP}_4(\mathbf{S}'), \ \mathbf{o}_{para} = \text{MLP}_5(\mathbf{P}'), \quad (6)$$

where $\mathbf{o}_{sent} \in \mathbb{R}^{n_s}$ represents whether a sentence is selected as supporting facts, and $\mathbf{o}_{para} \in \mathbb{R}^{n_p}$ represents whether a paragraph contains the ground-truth supporting facts.

We treat entity prediction ($\mathcal{L}_{entity}$) as a multi-class classification problem. Candidate entities include all entities in the question and those that match the titles in the context. If the ground-truth answer does not exist among the entity nodes, the entity loss is zero. Specifically,

$$\mathbf{o}_{entity} = \text{MLP}_6(\mathbf{E}'). \quad (7)$$

The entity loss will only serve as a regularization term, and the final answer prediction will only rely on the answer span extraction module as follows.

[2]Note that we omit the bias term for all the weight matrices in the paper to save space.

Model	Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1
DecompRC (Min et al., 2019b)	55.20	69.63	-	-	-	-
ChainEx (Chen et al., 2019)	61.20	74.11	-	-	-	-
Baseline Model (Yang et al., 2018)	45.60	59.02	20.32	64.49	10.83	40.16
QFE (Nishida et al., 2019)	53.86	68.06	57.75	84.49	34.63	59.61
DFGN (Xiao et al., 2019)	56.31	69.69	51.50	81.62	33.62	59.82
LQR-Net (Grail et al., 2020)	60.20	73.78	56.21	84.09	36.56	63.68
P-BERT†	61.18	74.16	51.38	82.76	35.42	63.79
TAP2 (Glass et al., 2019)	64.99	78.59	55.47	85.57	39.77	69.12
EPS+BERT†	65.79	79.05	58.50	86.26	42.47	70.48
SAE-large (Tu et al., 2020)	66.92	79.62	61.53	86.86	45.36	71.45
C2F Reader(Shao et al., 2020)	67.98	81.24	60.81	87.63	44.67	72.73
Longformer⋆ (Beltagy et al., 2020)	68.00	81.25	63.09	88.34	45.91	73.16
ETC-large⋆ (Zaheer et al., 2020)	68.12	81.18	**63.25**	**89.09**	46.40	73.62
HGN (ours)	**69.22**	**82.19**	62.76	88.47	**47.11**	**74.21**

Table 1: Results on the test set of HotpotQA in the Distractor setting. HGN achieves state-of-the-art results at the time of submission (Dec. 1, 2019). (†) and (⋆) indicates unpublished and concurrent work. RoBERTa-large (Liu et al., 2019) is used for context encoding.

The logits of every position being the start and end of the ground-truth span are computed by a two-layer MLP on top of $\mathbf{G}$ in Eqn.(4):

$$\mathbf{o}_{start} = \text{MLP}_7(\mathbf{G}),\ \mathbf{o}_{end} = \text{MLP}_8(\mathbf{G})\,. \quad (8)$$

Following previous work (Xiao et al., 2019), we also need to identify the answer type, which includes the types of span, entity, yes and no. We use a 3-way two-layer MLP for answer-type classification based on the first hidden representation of $\mathbf{G}$:

$$\mathbf{o}_{type} = \text{MLP}_9(\mathbf{G}[0])\,. \quad (9)$$

During decoding, we first use this to determine the answer type. If it is "yes" or "no", we directly return it as the answer. Overall, the final cross-entropy loss ($\mathcal{L}_{joint}$) used for training is defined over all the aforementioned logits: $\mathbf{o}_{sent}, \mathbf{o}_{para}, \mathbf{o}_{entity}, \mathbf{o}_{start}, \mathbf{o}_{end}, \mathbf{o}_{type}$.

4 Experiments

In this section, we describe experiments comparing HGN with state-of-the-art approaches and provide detailed analysis on the model and results.

4.1 Dataset

We use HotpotQA dataset (Yang et al., 2018) for evaluation, a popular benchmark for multi-hop QA. Specifically, two sub-tasks are included in this dataset: (i) Answer prediction; and (ii) Supporting facts prediction. For each sub-task, exact match (EM) and partial match (F1) are used to evaluate model performance, and a joint EM and F1 score is used to measure the final performance, which encourages the model to take both answer and evidence prediction into consideration.

There are two settings in HotpotQA: *Distractor* and *Fullwiki* setting. In the Distractor setting, for each question, two gold paragraphs with ground-truth answers and supporting facts are provided, along with 8 'distractor' paragraphs that were collected via a bi-gram TF-IDF retriever (Chen et al., 2017). The Fullwiki setting is more challenging, which contains the same training questions as in the Distractor setting, but does not provide relevant paragraphs for test set. To obtain the right answer and supporting facts, the entire Wikipedia can be used to find relevant documents. Implementation details can be found in Appendix B.

4.2 Experimental Results

Results on Test Set Table 1 and 2 summarize results on the hidden test set of HotpotQA. In Distractor setting, HGN outperforms both published and unpublished work on every metric by a significant margin, achieving a Joint EM/F1 score of 47.11/74.21 with an absolute improvement of 2.44/1.48 over previous state of the art. In Fullwiki setting, HGN achieves state-of-the-art results on

Model	Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1
TPReasoner (Xiong et al., 2019)	36.04	47.43	-	-	-	-
Baseline Model (Yang et al., 2018)	23.95	32.89	3.86	37.71	1.85	16.15
QFE (Nishida et al., 2019)	28.66	38.06	14.20	44.35	8.69	23.10
MUPPET (Feldman and El-Yaniv, 2019)	30.61	40.26	16.65	47.33	10.85	27.01
Cognitive Graph (Ding et al., 2019)	37.12	48.87	22.82	57.69	12.42	34.92
PR-BERT†	43.33	53.79	21.90	59.63	14.50	39.11
Golden Retriever (Qi et al., 2019)	37.92	48.58	30.69	64.24	18.04	39.13
Entity-centric BERT (Godbole et al., 2019)	41.82	53.09	26.26	57.29	17.01	39.18
SemanticRetrievalMRS (Yixin Nie, 2019)	45.32	57.34	38.67	70.83	25.14	47.60
Transformer-XH (Zhao et al., 2020)	48.95	60.75	41.66	70.01	27.13	49.57
MIR+EPS+BERT†	52.86	64.79	42.75	72.00	31.19	54.75
Graph Recur. Retriever (Asai et al., 2020)	**60.04**	**72.96**	49.08	76.41	35.35	61.18
HGN (RoBERTa-large)	57.85	69.93	51.01	76.82	37.17	60.74
HGN (ALBERT-xxlarge-v2)	59.74	71.41	**51.03**	**77.37**	**37.92**	**62.26**

Table 2: Results on the test set of HotpotQA in the Fullwiki setting. HGN achieves state-of-the-art results at the time of submission (Feb. 11, 2020). (†) indicates unpublished work. RoBERTa-large (Liu et al., 2019) and ALBERT-xxlarge-v2 (Lan et al., 2020) are used for context encoding, and SemanticRetrievalMRS is used for retrieval. Leaderboard: https://hotpotqa.github.io/.

Joint EM/F1 with 2.57/1.08 improvement, despite using an inferior retriever; when using the same retriever as in SemanticRetrievalMRS (Yixin Nie, 2019), our method outperforms by a significant margin, demonstrating the effectiveness of our multi-hop reasoning approach. In the following sub-sections, we provide a detailed analysis on the sources of performance gain on the dev set. Additional ablation study on paragraph selection is provided in Appendix D.

Effectiveness of Hierarchical Graph As described in Section 3.1, we construct our graph with four types of nodes and seven types of edges. For ablation study, we build the graph step by step. First, we only consider edges from question to paragraphs, and from paragraphs to sentences, *i.e.*, only edge type (i), (iii) and (iv) are considered. We call this the PS Graph. Based on this, entity nodes and edges related to each entity node (corresponding to edge type (ii) and (v)) are added. We call this the PSE Graph. Lastly, edge types (vi) and (vii) are added, resulting in the final hierarchical graph.

As shown in Table 4, the use of PS Graph improves the joint F1 score over the plain RoBERTa model by 2.81 points. By further adding entity nodes, the Joint F1 increases by 0.30 points. This indicates that the addition of entity nodes is helpful, but may also bring in noise, thus only leading to limited performance improvement. By including edges among sentences and paragraphs, our final hierarchical graph provides an additional improvement of 0.24 points. We hypothesize that this is due to the explicit connection between sentences that leads to better representations.

Effectiveness of Pre-trained Language Model To verify the effects of pre-trained language models, we compare HGN with prior state-of-the-art methods using the same pre-trained language models. Results in Table 5 show that our HGN variants outperform DFGN, EPS and SAE, indicating the performance gain comes from better model design.

4.3 Analysis

In this section, we provide an in-depth error analysis on the proposed model. HotpotQA provides two reasoning types: "bridge" and "comparison". "Bridge" questions require the identification of a bridge entity that leads to the answer, while "comparison" questions compare two entities to infer the answer, which could be *yes*, *no* or *a span of text*. For analysis, we further split "comparison" questions into "comp-yn" and "comp-span". Table 6 indicates that "comp-yn" questions are the easiest, on which our model achieves 88.5 joint F1 score. HGN performs similarly on "bridge" and "comp-span" with 74 joint F1 score, indicating that there is still room for further improvement.

To provide a more in-depth understanding of our

Category	Question	Answer	Prediction	Pct (%)
Annotation	Were the films Tonka and 101 Dalmatians released in the same decade?	1958 Walt Disney Western adventure film	No	9
Multiple Answers	Michael J. Hunter replaced the lawyer who became the administrator of which agency?	EPA	Environmental Protection Agency	24
Discrete Reasoning	Between two bands, Mastodon and Hole, which one has more members?	Mastodon	Hole	15
Commonsense & External Knowledge	What is the name of second extended play by the artists of the mini-album Code#01?	Code#02 Pretty Pretty	Code#01 Bad Girl	16
Multi-hop	Who directed the film based on the rock opera 5:15 appeared in?	Franc Roddam	Ken Russell	16
MRC	How was Ada Lovelace, the first computer programmer, related to Lord Byron in Childe Byron?	his daughter	strained relationship	20

Table 3: Error analysis of HGN model. For 'Multi-hop' errors, the model jumps to the wrong film ("Tommy (1975 film)") instead of the correct one ("Quadrophenia (film)") from the starting entity "rock opera 5:15". The supporting fact for the 'MRC' example is "*Childe Byron is a 1977 play by Romulus Linney about the* ***strained relationship*** *between the poet, Lord Byron, and* ***his daughter****, Ada Lovelace*".

Model	Ans F1	Sup F1	Joint F1
w/o Graph	80.58	85.83	71.02
PS Graph	81.68	88.44	73.83
PSE Graph	82.10	88.40	74.13
Hier. Graph	**82.22**	**88.58**	**74.37**

Table 4: Ablation study on the effectiveness of the hierarchical graph on the dev set in the Distractor setting. RoBERTa-large is used for context encoding.

Model	Ans F1	Sup F1	Joint F1
DFGN (BERT-base)	69.38	82.23	59.89
EPS (BERT-wwm)†	79.05	86.26	70.48
SAE (RoBERTa)	80.75	87.38	72.75
HGN (BERT-base)	74.76	86.61	66.90
HGN (BERT-wwm)	80.51	88.14	72.77
HGN (RoBERTa)	82.22	88.58	74.37
HGN (ALBERT-xxlarge-v2)	**83.46**	**89.2**	**75.79**

Table 5: Results with different pre-trained language models on the dev set in the Distractor setting. (†) is unpublished work with results on the test set, using BERT whole word masking (wwm).

Question	Ans F1	Sup F1	Joint F1	Pct (%)
comp-yn	93.45	94.22	88.50	6.19
comp-span	79.06	91.72	74.17	13.90
bridge	81.90	87.60	73.31	79.91

Table 6: Results of HGN for different reasoning types. 'Pct' is short for 'Percentage'.

model's weaknesses (and provide insights for future work), we randomly sample 100 examples in the dev set with the answer F1 as 0. After carefully analyzing each example, we observe that these errors can be roughly grouped into six categories: (i) *Annotation*: the annotation provided in the dataset is not correct; (ii) *Multiple Answers*: questions may have multiple correct answers, but only one answer is provided in the dataset; (iii) *Discrete Reasoning*: this type of error often appears in "comparison" questions, where discrete reasoning is required to answer the question correctly; (iv) *Commonsense & External Knowledge*: to answer this type of question, commonsense or external knowledge is required; (v) *Multi-hop*: the model fails to perform multi-hop reasoning, and finds the final answer from wrong paragraphs; (vi) *MRC*: model correctly finds the supporting paragraphs and sentences, but predicts the wrong answer span.

Note that these error types are not mutually exclusive, but we aim to classify each example into only one type, in the order presented above. For

example, if an error is classified as 'Commonsense & External Knowledge' type, it cannot be classified as 'Multi-hop' or 'MRC' error. Table 3 shows examples from each category (the corresponding paragraphs are omitted due to space limit).

We observed that a lot of errors are due to the fact that some questions have multiple answers with the same meaning, such as "*a body of water* vs. *creek*", "*EPA* vs. *Environmental Protection Agency*", and "*American-born* vs. *U.S. born*". In these examples, the former is the ground-truth answer, and the latter is our model's prediction. Secondly, for questions that require commonsense or discrete reasoning (*e.g.*, "*second*" means "*Code#02*"[3], "*which band has more members*", or "*who was born earlier*"), our model just randomly picks an entity as answer, as it is incapable of performing this type of reasoning. The majority of the errors are from either multi-hop reasoning or MRC model's span selection, which indicates that there is still room for further improvement. Additional examples are provided in Appendix F.

4.4 Generalizability Discussion

The hierarchical graph can be applied to different multi-hop QA datasets, though in this paper mainly tailored for HotpotQA. Here we use Wikipedia hyperlinks to connect sentences and paragraphs. An alternative way is to use an entity linking system to make it more generalizable. For each sentence node, if its entities exist in a paragraph, an edge can be added to connect the sentence and paragraph nodes. In our experiments, we restrict the number of multi-hops to two for the HotpotQA task, which can be increased to accommodate other datasets. The maximum number of paragraphs is set to four for HotpotQA, as we observe that using more documents within a maximum sequence length does not help much (see Table 9 in the Appendix). To generalize to other datasets that need to consume longer documents, we can either: (i) use sliding-window-based method to chunk a long sequence into short ones; or (ii) replace the BERT-based backbone with other transformer-based models that are capable of dealing with long sequences (Beltagy et al., 2020; Zaheer et al., 2020; Wang et al., 2020).

5 Conclusion

In this paper, we propose a new approach, Hierarchical Graph Network (HGN), for multi-hop question answering. To capture clues from different granularity levels, our HGN model weaves heterogeneous nodes into a single unified graph. Experiments with detailed analysis demonstrate the effectiveness of our proposed model, which achieves state-of-the-art performances on the HotpotQA benchmark. Currently, in the Fullwiki setting, an off-the-shelf paragraph retriever is adopted for selecting relevant context from large corpus of text. Future work includes investigating the interaction and joint training between HGN and paragraph retriever for performance improvement.

[3]Please refer to Row 4 in Table 3 for more context.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *ICLR*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *ACL*.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *NAACL*.

Jifan Chen, Shih-ting Lin, and Greg Durrett. 2019. Multi-hop question answering via reasoning chains. *arXiv preprint arXiv:1910.02610*.

Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In *ACL*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *NAACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *NAACL*.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *ACL*.

Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. *arXiv preprint arXiv:1906.06606*.

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

Ameya Godbole, Dilip Kavarthapu, Rajarshi Das, Zhiyu Gong, Abhishek Singhal, Hamed Zamani, Mo Yu, Tian Gao, Xiaoxiao Guo, Manzil Zaheer, et al. 2019. Multi-step entity-centric information retrieval for multi-hop question answering. *arXiv preprint arXiv:1909.07598*.

Quentin Grail, Julien Perez, and Eric Gaussier. 2020. Latent question reformulation and information accumulation for multi-hop machine reading. *https://openreview.net/forum?id=S1x63TEYvr*.

Yichen Jiang and Mohit Bansal. 2019a. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop qa. In *ACL*.

Yichen Jiang and Mohit Bansal. 2019b. Self-assembling modular networks for interpretable multi-hop reasoning. In *EMNLP*.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *EMNLP*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *ACL*.

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*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *ACL*.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *ACL*.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*.

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 *ACL*.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. In *EMNLP*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Nan Shao, Yiming Cui, Ting Liu, Wang, and Guoping Hu Hu. 2020. Is graph structure necessary for multi-hop reasoningt. *arXiv preprint arXiv:2004.03096*.

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*.

Swabha Swayamdipta, Ankur P. Parikh, and Tom Kwiatkowski. 2018. Multi-mention learning for reading comprehension with neural cascades. In *ICLR*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *ACL*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *ICLR*.

Shuohang Wang, Luowei Zhou, Zhe Gan, Yen-Chun Chen, Yuwei Fang, Siqi Sun, Yu Cheng, and Jingjing Liu. 2020. Cluster-former: Clustering-based sparse transformer for long-range dependency encoding. *arXiv preprint arXiv:2009.06097*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *TACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yunxuan Xiao, Yanru Qu, Lin Qiu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *ACL*.

Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Hong Wang, Shiyu Chang, Murray Campbell, and William Yang Wang. 2019. Simple yet effective bridge reasoning for open-domain multi-hop question answering. *arXiv preprint arXiv:1909.07597*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Mohit Bansal Yixin Nie, Songhe Wang. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *EMNLP*.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *ICLR*.

Victor Zhong, Caiming Xiong, Nitish Keskar, and Richard Socher. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. In *ICLR*.

A Datasets

There are two benchmark settings in HotpotQA: *Distractor* and *Fullwiki* setting. They both have 90k training samples and 7.4k development samples. In the Distractor setting, there are 2 gold paragraphs and 8 distractors. However, 2 gold paragraphs may not be available in the Fullwiki Setting. Therefore, the Fullwiki setting is more challenge which requires to search the entire Wikipedia to find relevant documents. For both settings, there are 90K hidden test samples. More details about the dataset can be found in Yang et al. (2018).

B Implementation Details

Our implementation is based on the Transformer library (Wolf et al., 2019). To construct the proposed hierarchical graph, we use spacy[4] to extract entities from both questions and sentences. The numbers of entities, sentences and paragraphs in one graph are limited to 60, 40 and 4, respectively. Since HotpotQA only requires two-hop reasoning, up to two paragraphs are connected to each question. Our paragraph ranking model is a binary classifier based on the RoBERTa-large model. For the Fullwiki setting, we leverage the retrieved paragraphs and the paragraph ranker provided by Yixin Nie (2019). We finetune on the training set for 8 epochs, with batch size as 8, learning rate as 1e-5, λ_1 as 1, λ_2 as 5, λ_3 as 1, λ_4 as 1, LSTM dropout rate as 0.3 and GNN dropout rate as 0.3. We search hyperparameters for learning rate from {1e-5, 2e-5, 3e-5} , λ_2 from {1, 3, 5} and dropout rate from {0.1, 0.3, 0.5}.

C Computing Resources

We conduct experiments on 4 Quadro RTX 8000 GPUs. The parameters of each component in HGN are summarized in Table 7. The computation bottleneck is mainly from RoBERTa. The best model of HGN took around 12 hours for training, which is almost the same as the RoBERTa-large baseline.

Components	#Parameters
RoBERTa	355M
Bi-Attention	0.62M
BiLSTM	1.44M
GNN	29M
Multi-task Layer	0.55M

Table 7: Number of parameters for each component in HGN.

D Effectiveness of Paragraph Selection

The proposed HGN relies on effective paragraph selection to find relevant multi-hop paragraphs. Table 8 shows the performance of paragraph selection on the dev set of HotpotQA. In DFGN, paragraphs are selected based on a threshold to maintain high recall (98.27%), leading to a low precision (60.28%). Compared to both threshold-based and pure Top-N-based paragraph selection, our two-step para-

[4] https://spacy.io

Method	Precision	Recall	#Para.
Threshold-based	60.28	98.27	3.26
Top 2 from ranker	93.43	93.43	2
Top 4 from ranker	49.39	98.48	4
1st hop	96.10	59.74	1.24
2 paragraphs (ours)	**94.53**	94.53	2
4 paragraphs (ours)	49.45	**98.74**	4

Table 8: Performance of paragraph selection on the dev set of HotpotQA based on BERT-base.

graph selection process is more accurate, achieving 94.53% precision and 94.53% recall. Besides these two top-ranked paragraphs, we also include two other paragraphs with the next highest ranking scores, to obtain a higher coverage on potential answers. Table 9 summarizes the results on the dev set in the Distractor setting, using our paragraph selection approach for both DFGN and the plain BERT-base model. Note that the original DFGN does not finetune BERT, leading to much worse performance. In order to provide a fair comparison, we modify their released code to allow finetuning of BERT. Results show that our paragraph selection method outperforms the threshold-based one in both models.

Model	Ans F1	Sup F1	Joint F1
DFGN (paper)	69.38	82.23	59.89
DFGN			
+ threshold-based	71.90	83.57	63.04
+ 2 para. (ours)	72.53	83.57	63.87
+ 4 para. (ours)	**72.67**	83.34	63.63
BERT-base			
+ threshold-based	71.95	82.79	62.43
+ 2 para. (ours)	72.42	83.64	63.94
+ 4 para. (ours)	**72.67**	**84.86**	**64.24**

Table 9: Results with selected paragraphs on the dev set in the Distractor setting.

E Case Study

We provide two example questions for case study. To answer the question in Figure 3 (left), Q needs to be linked with $P1$. Subsequently, the sentence $S4$ within $P1$ is connected to $P2$ through the hyperlink (*"John Surtees"*) in $S4$. A plain BERT model without using the constructed graph missed $S7$ as additional supporting facts, while our HGN discovers and utilizes both pieces of evidence as the connections among $S4$, $P2$ and $S7$ are explicitly encoded in our hierarchical graph.

For the question in Figure 3 (right), the inference chain is $Q \rightarrow P1 \rightarrow S1 \rightarrow S2 \rightarrow P2 \rightarrow S3$. The plain BERT model infers the evidence sentences $S2$ and $S3$ correctly. However, it fails to predict $S1$ as the supporting facts, while HGN succeeds, potentially due to the explicit connections between sentences in the constructed graph.

F Additional Examples for Error Analysis

Below, we provide additional examples for error analysis, where "Q" denotes question, "A" denotes answer provided with dataset and "P" denotes the prediction of proposed model. A full list of all the 100 examples is provided in Table 10 and 11.

Category: Annotation
ID: 5ae2e0fd55429928c4239524
Q: What actor was also a president that Richard Darman worked with when they were in office?
A: George H. W. Bush
P: Ronald Reagan

ID: 5ab43b755542991779162c21
Q: What sports club based in Hamburg Germany had a Persian born football player who played for eight seasons?
A: Mehdi Mahdavikia
P: Hamburger SV

ID: 5a72e28f5542992359bc31ba
Q: Which technique did the director at Pzena Investment Management outline?
A: outlined by Joel Greenblatt
P: Magic formula investing

ID: 5a7e71ab55429949594199bc
Q: Perfect Imperfection is a 2016 Chinese romantic drama film starring a south Korean actor best known for his roles in what 2016 television drama?
A: Reunited Worlds
P: Cinderella and Four Knights

ID: 5a7a18b05542990783324e53
Q: What year was the independent regional brewery founded that currently operates in Hasting's oldest pub?
A: since 1864
P: 1698

Category: Multiple Answers

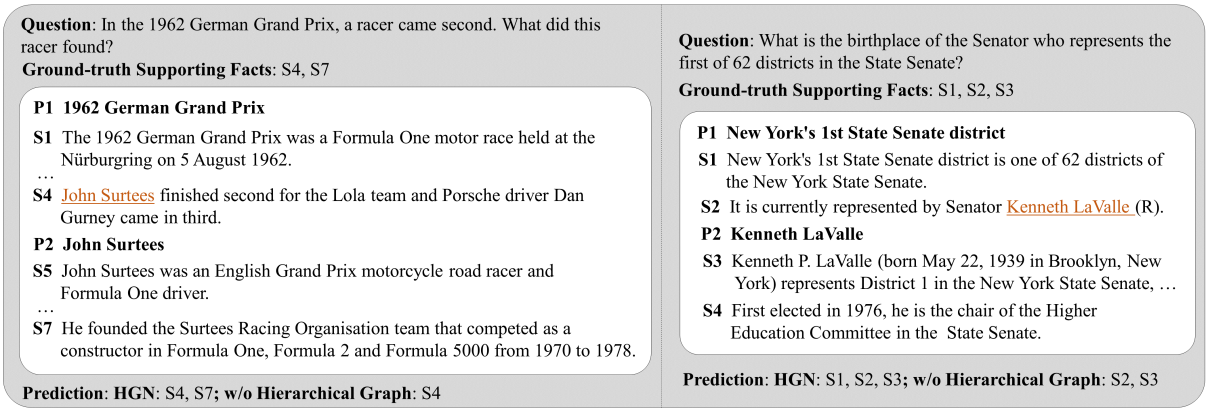

Figure 3: Examples of supporting facts prediction in the HotpotQA Distractor setting.

Category	Sample IDs
Annotation	6, 23, 33, 38, 47, 59, 75, 81, 93
Multiple Answers	1, 4, 8, 10, 11, 16, 19, 24, 26, 28, 29, 32, 39, 40, 42, 50, 53, 56, 60, 63, 67, 68, 71, 72
Discrete Reasoning	0, 2, 9, 21, 22, 35, 37, 45, 58, 64, 77, 82, 86, 88, 95
Commonsense & External Knowledge	7, 15, 20, 36, 69, 70, 73, 76, 78, 83, 84, 85, 87, 91, 92, 96
Multi-hop	3, 17, 25, 27, 30, 41, 43, 46, 54, 57, 62, 74, 79, 90, 97, 99
MRC	5, 12, 13, 14, 18, 31, 34, 44, 48, 49, 51, 52, 55, 61, 65, 66, 80, 89, 94, 98

Table 10: The categories and sample IDs for the 100 examples selected for error analysis. The sample IDs are mapped to the ground-truth IDs in Table 11.

ID: 5a8c9641554299585d9e36f5
Q: Which season of Alias does the English actor, who was born 25 June 1961, appear?
A: three
P: third season

ID: 5ae6179b5542992663a4f25b
Q: Which Hong Kong actor born on 19 August 1946 starred in The Sentimental Swordsman
A: Tommy Tam Fu-Wing
P: Ti Lung[5]

ID: 5abec66b5542997ec76fd360
Q: What do Josef Veltjens and Hermann Goering have in common?
A: A veteran World War I fighter pilot ace
P: German

ID: 5a85d6d95542996432c570fb
Q: What is one element of House dance where the dancer ripples his or her torso back and forth?
A: the jack
P: Jacking

ID: 5a79c9395542994bb94570a2
Q: Which two occupations does Ronnie Dunn and Annie Lennox have in common?
A: singer, songwriter
P: singer-songwriter

Category: Discrete Reasoning
ID: 5a8ec3205542995a26add506
Q: Does Dashboard Confessional have more members than World Party?
A: yes
P: no

ID: 5abfd83f5542997ec76fd45c
Q: Which genus has more species, Quesnelia or Honeysuckle?
A: Honeysuckle
P: Honeysuckles

[5] Alias of the true answer, Tommy Tam Fu-Wing

ID: 5ac44b47554299194317396c
Q: Which became a Cathedral first St Chad's Cathedral, Birmingham or Chelmsford Cathedral?
A: Metropolitan Cathedral Church and Basilica of Saint Chad
P: St Chad's

ID: 5ac2455e55429951e9e68512
Q: Were both Life magazine and Strictly Slots magazine published monthly in 1998?
A: yes
P: no

ID: 5a7d26bd554299452d57bb28
Q: Who was born earlier, Johnny Lujack or Jim Kelly?
A: Jim Kelly
P: John Christopher Lujack

Category: Commonsense & External Knowledge
ID: 5ac275e755429921a00aaf81
Q: From what nation is the football player who was named Man of the Match at the 2001 Intercontinental Cup?
A: Ghana
P: Ghanaian

ID: 5ac02d345542992a796decc0
Q: Where are Abbey Clancy and Peter Crouch from?
A: England
P: English

ID: 5ab2beba5542991669774 08f
Q: Who is the father of the Prince in which William Joseph Weaver is most famous for painting a full length portrait of?
A: George III
P: Queen Victoria

ID: 5a8dab16554299068b959d89
Q: What type of elevation does Aldgate railway station, Adelaide and Aldgate, South Australia have in common?
A: Hills
P: kilometres

ID: 5a82edae55429966c78a6a9f
Q: Swiss music duo Double released their best known single "The Captain of Her Heart" in what year?
A: 1986
P: 1985

Category: Multi-hop
ID: 5a7a46605542994f819ef1ad
Q: What year did Roy Rogers and his third wife star in a film directed by Frank McDonald?
A: 1945
P: 1946

ID: 5a84f7255542991dd0999e33
Q: Which country borders the Central African Republic and is south of Libya and east of Niger?
A: Republic of Chad
P: Sudan

ID: 5a77152355429966f1a36c2e
Q: What was the Roud Folk Song Index of the nursery rhyme inspiring What Are Little Girls Made Of?
A: 821
P: 326

ID: 5a7e7c725542991319bc94be
Q: In what year did Farda Amiga win a race at the Saratoga Race course?
A: (foaled February 1, 1999)
P: 1872

ID: 5ae21ef35542994d89d5b35d
Q: What college teamdid the point guard that led the way for Philedlphia 76ers in the 2017-18 season play basketball in?
A: Washington Huskies
P: University of Kansas

Category: MRC
ID: 5ae5cf625542996de7b71a22
Q: What sports team included both of the brothers Case McCoy and Colt McCoy during different years?
A: University of Texas Longhorns
P: Washington Redskins

ID: 5a8fa4a5554299458435d6a3
Q: What is name of the business unit led by Tina Sharkey at a web portal which is originally known as America Online?
A: Sesame Street
P: community programming

ID: 5a8135cc55429903bc27b943
Q: In the USA, gun powder is used in conjunction with this to start the Boomershot.
A: Anvil firing
P: an explosive fireball

ID: 5a84bb825542991dd0999dbe
Q: Who beacme a star as a comic book character created by Gerry Conway and Bob Oksner?
A: Megalyn Echikunwoke
P: Stephen Amell

ID: 5a75f1a755429976ec32bcb1
Q: Which actress played a character that dated Mark Brendanawicz?
A: Rashida Jones
P: Amy Poehler

ID		ID	
0	5ac2455e55429951e9e68512	1	5a8c9641554299585d9e36f5
2	5a8ec3205542995a26add506	3	5a7a46605542994f819ef1ad
4	5ae6179b5542992663a4f25b	5	5ac3c08a5542995ef918c217
6	5ae2e0fd55429928c4239524	7	5ac275e755429921a00aaf81
8	5a7ca98f55429935c91b5288	9	5a747a9a55429929fddd8444
10	5a88696b554299206df2b25b	11	5abec66b5542997ec76fd360
12	5ae5cf625542996de7b71a22	13	5abb729b5542993f40c73af4
14	5a85cead5542991dd0999ea9	15	5ac02d345542992a796decc0
16	5a7a88e455429941d65f268c	17	5a84f7255542991dd0999e33
18	5a8fa4a5554299458435d6a3	19	5ae7793c554299540e5a55c2
20	5a7755c65542993569682d54	21	5abfd83f5542997ec76fd45c
22	5adeb95d5542992fa25da827	23	5ab43b755542991779162c21
24	5a85d6d95542996432c570fb	25	5a8463945542992ef85e23d9
26	5ae7d0675542994a481bbdf2	27	5a82a55955429966c78a6a70
28	5ae7313c5542991e8301cbbc	29	5ac44629554299194317395d
30	5a89d36e554299515336132a	31	5ac2e97d554299657fa290c0
32	5a8a764555429930ff3c0de1	33	5a886211554299206df2b24a
34	5a8f05b1554299458435d517	35	5a840e8a5542992ef85e239e
36	5a7354e35542994cef4bc55b	37	5abc36cc55429959677d6a50
38	5a7a18b05542990783324e53	39	5ab5d27a554299494045f073
40	5ac19f405542991316484b5b	41	5a82ebb855429966c78a6a9c
42	5a72c9e85542991f9a20c595	43	5ae7739c5542997b22f6a775
44	5a84bda45542992a431d1a96	45	5a7d26bd554299452d57bb28
46	5ae21ef35542994d89d5b35d	47	5a753c8c55429916b01642ab
48	5ac24d725542996366519966	49	5ae0ec48554299422ee9955a
50	5a8febb555429916514e73e4	51	5a7c9d2e55429935c91b5261
52	5a8769475542993e715abf2b	53	5abbf519554299114383a0ad
54	5a735bae55429901807dafef	55	5a7299465542992359bc3131
56	5a8b2f2b5542995d1e6f12fa	57	5a77152355429966f1a36c2e
58	5a87954f5542996e4f308856	59	5a7e71ab55429949594199bc
60	5ac531ea5542994611c8b419	61	5ab72c7d55429928e1fe3830
62	5a7e7c725542991319bc94be	63	5ae54c085542992663a4f1c4
64	5adc7dbf5542994d58a2f618	65	5a8fb0be5542997ba9cb32ed
66	5a8135cc55429903bc27b943	67	5abcf17655429959677d6b5c
68	5ab925fd554299131ca42281	69	5ab2beba5542991669774 08f
70	5a8dab16554299068b959d89	71	5ac38ce255429939154137c2
72	5a79c9395542994bb94570a2	73	5ab946d7554299743d22eaaf
74	5a73d33e5542992d56e7e3a9	75	5a72e28f5542992359bc31ba
76	5ab1d983554299340b52540a	77	5a7cb9b95542990527d55515
78	5a773d8955429966f1a36cc4	79	5a7780e855429949eeb29e9f
80	5a84bb825542991dd0999dbe	81	5a7698c2554299373536010d
82	5ae1847e55429920d52343ee	83	5a7199725542994082a3e88f
84	5abf11d45542997719eab660	85	5ae52cb955429908b6326540
86	5ac44b47554299194317396c	87	5a7d61775542991319bc93b9
88	5ae0536755429924de1b70a6	89	5a75f1a755429976ec32bcb1
90	5adbc8e25542996e68525230	91	5a72ac8a5542992359bc3164
92	5adc6ded55429947ff17395d	93	5a7b971255429927d897bff3
94	5ae34a225542992e3233c370	95	5ac2cdaa554299657fa29070
96	5a82edae55429966c78a6a9f	97	5a8a3a355542996c9b8d5e5e
98	5adcc90c5542990d50227d1b	99	5a79c9c05542994bb94570a5

Table 11: The full index list of the 100 samples selected for error analysis.