USING SEMANTIC STRUCTURE OF THE DATA AND KNOWLEDGE IN QUESTION ANSWERING SYSTEMS

By

Chen Zheng

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2022

ABSTRACT

Understanding and reasoning over natural language is one of the most crucial and long-standing challenges in Artificial Intelligence (AI). Question answering (QA) is the task of automatically answering questions posed by humans in a natural language form. It is an important criterion to evaluate the language understanding and reasoning capabilities of AI systems. Though machine learning systems on Question Answering (QA) have shown tremendous success in language understanding, they still suffer from a lack of interpretability and generalizability, in particular, when complex reasoning is required to answer the questions. In this dissertation, we aim to build novel QA architectures that answer complex questions using the explicit relational structure of the raw data, that is, text and image, and exploiting external knowledge. We investigate a variety of problems, including answering natural language questions when the answer can be found in multiple modalities, including, 1) Textual documents (Document-level QA), 2) Images (Cross-Modality QA), 3) Knowledge graphs (Commonsense QA) and, 4) Combination of text and knowledge graphs. **First**, for Document-level QA, we develop a new technique, Semantic Role Labeling Graph Reasoning Network (SRLGRN), via which the explicit semantic structure of multiple textual documents is used. In particular, based on semantic role labeling, we form a multi-relational graph that jointly learns to find cross-paragraph reasoning paths and answers multi-hop reasoning questions. **Second**, for the type QA that requires causal reasoning over textual documents, we propose a new technique, Relational Gating Network (RGN), that jointly learns to extract the entities and their relations to help highlight the important entity chains and find how those affect each other. **Third**, for the type of questions that require complex reasoning over language and vision modalities (Cross-Modality QA), we propose a new technique, Cross-Modality Relevance (CMR). This technique considers the relevance between textual tokens and visual objects by aligning the two modalities. **Fourth**, for answering questions based on given Knowledge Graphs (KG), we propose a new technique, Dynamic Relevance Graph Network (DRGN). This technique is based on a graph neural network and re-scales the importance of the neighbor nodes in the graph dynamically by training a relevance matrix. The new neighborhoods trained by relevance help fill in the knowledge gaps in the KG for

more effective knowledge-based reasoning. **Fifth**, for answering questions using a combination of textual documents and an external knowledge graph, we propose a new technique, Multi-hop Reasoning Network over Relevant Commonsense Subgraphs (MRRG). MRRG technique extracts the most relevant KG subgraph for each question and document and uses that subgraph combined with the textual content and question representations for answering complex questions. We improve the performance, interpretability, and generalizability of various challenging QA benchmarks based on different modalities. Our ideas have proven to be effective in multi-hop reasoning, causal reasoning, cross-modality reasoning, and knowledge based reasoning.

To my grandfather Mr. Enke Zheng and my father Mr. Wanying Zheng.

ACKNOWLEDGEMENTS

The past years at Michigan State University have been an unforgettable experience for me. Surprisingly, I have done much exciting research about helping artificial intelligence to understand and reason over natural language. Luckily, in my four-and-a-half-year Ph.D. journey, I met many outstanding people who encouraged me, helped me, and supported me, including my advisor, lab partners, intern colleagues, friends, and most importantly, family.

First and foremost, I would like to express my deepest appreciation to my advisor Dr. Parisa Kordjamshidi. Parisa is an awesome advisor who helped me in my pursuit of scientific research. I learned a lot from Parisa, not only how to conduct novel NLP research, but also how to be a high-standard researcher. I really appreciate her earnest and rigorous attitude toward scientific research. I appreciate that Parisa never limited my research direction. She always supported my ideas and motivated me to think deeply about the research challenges. I will always remember every detailed and meaningful comment from her inspiration, and I will never forget her helpful revisions on every sentence and paragraph of all my published papers on countless days and nights. In addition, I would like to thank my Ph.D. committee members, Dr. Jiayu Zhou, Dr. Kristen Johnson, and Dr. Taiquan Peng, for their valuable and insightful comments and suggestions in completing this dissertation.

Next, I would like to thank my Heterogeneous Learning and Reasoning Lab. The HLR lab has witnessed the growth of my Ph.D. career at MSU. I would like to thank my lab partners, Yue Zhang (My most trusted and reliant friend throughout my Ph.D. career), Hossein Faghihi, Roshanak Mirzaee, Guangyue Xu, Drew Hayward, Juan Castro-Garcia, Darius Nafar, Danial Kamali, Sushanta K. Pani, Hamid Karimian and Elham Barezi. Here, I want to especially thank Dr. Quan Guo, who is the Postdoc in our laboratory and also my roommate. Thank you for providing help not only for research but also for daily life.

Before pursuing my Ph.D., I had never heard of natural language processing (NLP) and deep learning until I met my big brother, Dr. Shuangfei Zhai. He helped me open the door about how to conduct the NLP research. Besides, even now, I am so impressed with the Standford CS224d online

opening course taught by Dr. Richard Socher. I can't imagine how much valuable knowledge of NLP and deep learning I have gained in this class. I want to especially thank these two talented AI scientists who helped me get started with NLP and deep learning.

I have three wonderful internships at the Baidu NLP group, the JD.com Information Retrieval group, and the Bytedance AML group. I would like to express my thankfulness to my mentors, Dr. Shengxian Wan and Yu Sun, at Baidu. This internship experience made me realize that NLP can benefit countless people's lives. Moreover, I would like to thank my mentors, Dr. Wen-yun Yang, and Songlin Wang, at JD.com. This internship experience made me understand the challenge and importance of designing a robust retrieval system. I learned a lot and put many exciting ideas into the retrieval project. Furthermore, I would like to thank my mentors, Dr. Guokun Lai and Youlong Cheng, at Bytedance. This meaningful internship experience helped me to be a research scientist, putting scientific research ideas into the Tiktok system and benefiting people's lifestyles in many ways. I also would like to thank all my colleagues for their help on my intern projects. I want to mention Jiaxiang Liu, Shuohuan Wang, Weichong Yin, Fei Yu, Xiangyang Zhou, Lu Li, Yan Zeng, Zhou Xin, Dianhai Yu (Baidu), Han Zhang, Kang Zhang, Shang Wang (JD.com), and Wumo Yan, Jimmy Kim, Wen Liang (Bytedance).

Outside of NLP research, I am fortunate to be surrounded by many friends. I would like to thank my extremely kind piano teacher, Dante Li. She made me play the piano more sweetly and pleasingly and helped me to improve my piano level rapidly. I would like to thank Qiwen Sheng and Weiyang Yang for every wonderful trip and delicious dinner. I would like to thank Qian Song for accompanying me to walk along the Manhattan River with her lovely dog during the COVID-19 situation that made people feel most depressed. I would like to thank Yijun Zheng and Yuchong Chen. I witnessed their sweet love and marriage process from two of my single roommates to a young couple. I would like to thank Weixing Ren for keeping the friendship for more than fourteen years. Moreover, I would like to mention some names that are my best friends: Yue Zhang, Dong Chen, Meng Xu, Bin Wang, Lu Lu, Haochen Liu, Yang Zheng, Yubo Wang, Zhanwang Chen, Ying Cao, and Ruolei Xia.

Finally, I would like to express my deepest appreciation to my family. I own the warmest family in the world. I always thank you for your unconditional support. My father, Wanying Zheng, usually told me that they were proud of me. As the last sentences in the acknowledgment, I would like to say: This dissertation is dedicated to my family. I am proud to be the child of my family forever.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Understanding and reasoning over natural language play a significant role in many real-world artificial intelligence applications. Question Answering (QA) is one of the most crucial problems in evaluating the understanding and reasoning over natural language text [41, 2]. Question answering is a computer science discipline within the fields of Natural Language Processing (NLP), Machine Learning, and Information Retrieval (IR), which is concerned with building systems that automatically answer questions posed by humans in a natural language. Nowadays, QA systems are now widely used in many real-world applications, such as search engines (Google, Bing, Baidu), reading comprehension, and AI conversational systems (Alexa assistant). In this dissertation, we address different types of QA problems categorized into five classes. We use a simple but straightforward example of a "crying child" story shown in Figure 1.1 to introduce these five types of QA problems.

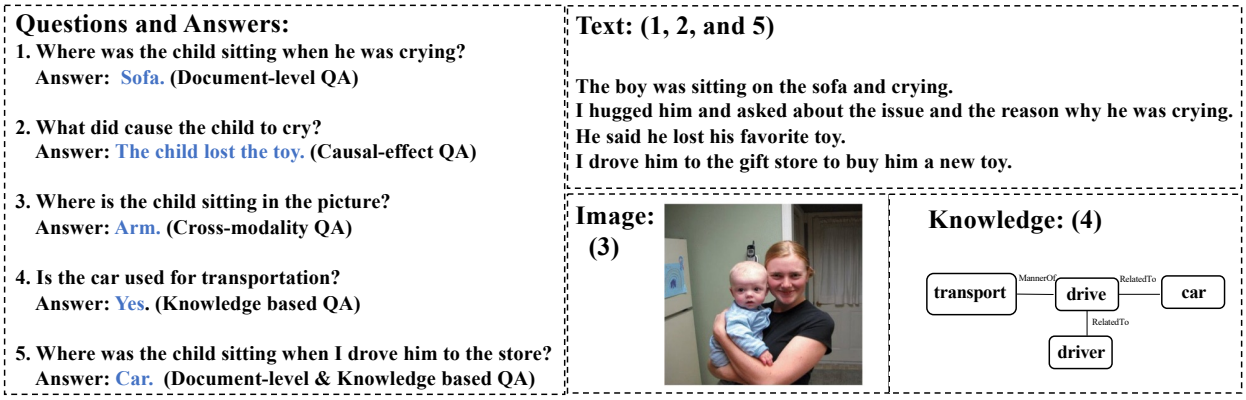

Figure 1.1 The "Crying child" example of four categories of QA tasks.

- **Document-level Question Answering.** The first question in Figure 1.1 shows a straightforward example of a Document-level QA task. Given the question "Where was the child sitting when he was crying?" and the text "The boy was sitting on the sofa and crying", the problem is to find the answer "sofa". This type of QA allows humans to ask questions based on the given

document. The QA system requires reading and understanding the text, capturing the line of reasoning from the document, and answering the question effectively [93].

- **Cause-effect Question Answering.** The second question in Figure 1.1 shows a simple example of Cause-effect QA [107]. Given the question "What did cause the child to cry?", we extract the causal events "child cry" and "the child lost the toy" in the text and answer the question. Cause-effect QA is a particular type of document-level QA, where the QA system should understand the causal and effect events and find explicit causal relationships between events.

- **Cross-modality Question Answering.** Cross-modality QA combines multidisciplinary fields, including language, vision, speech processing, etc [104]. We select Visual Question Answering (VQA) as the Cross-modality QA task in this dissertation. VQA task aims to answer a natural language question using an image. Vision-and-language reasoning requires the understanding of visual contents, language semantics, cross-modality alignments, and relationships between two modalities [118]. Let us look back to the "crying child" story and the third question "where is the child sitting in the picture?" We cannot state the answer just based on the text. However, after providing the image, we can quickly know the correct answer, "arm".

- **Knowledge based Question Answering.** Commonsense knowledge reflects the natural understanding of the world and human behavior. Structured knowledge is another modality of resources that can be fed into QA systems for answering natural language questions [61]. This type of QA task aims to answer natural language questions utilizing a knowledge base or a knowledge graph. The fourth question in Figure 1.1, "Is the car used for transportation", shows an example in which QA requires commonsense knowledge. In this case, QA systems should utilize the commonsensense like a human being about "car" $\rightarrow$ "used for" $\rightarrow$ "transportation".

- **Combine Document-level and Knowledge-based QA.** In some document-level QA scenarios, the contents included in a given text are sufficient to find the answer. However, there are

many cases in which the required knowledge is not included in the text itself [125, 107]. The fifth question in Figure 1.1, "Where was the child sitting when I drove him to the toy store?", shows an example in which QA requires commonsense knowledge. Individuals can provide the answer "car" because the human has the commonsense knowledge "drive" → "car" in their mind.

Traditionally, building QA systems have relied on natural language processing (NLP) technologies as backbones, including semantic role labeling [38], named entity recognition [57], part-of-speech tagging [70], relation extraction [95], text matching [138], etc. Intuitively, an ideal QA system should be able to understand the meanings of the text and the semantic relations between questions, documents, and answers. Over the past decade, deep learning, a particular category of machine learning, has achieved great success in multiple real-world NLP tasks, especially in Question Answering domain [87, 125, 4, 106, 103]. Specifically, the deep neural network is constructed by many neural layers. Each neural layer includes a massive number of "computational neurons" represented by scalars, tensors, and matrices. The neurons between two layers have connection edges, and the neural network propagates information via forward and backward directions. Deep learning QA architectures automatically extract the contextual and semantic features by pre-training from the large corpora and learn hundred-dimensional dense vectors to represent a word, a phrase, a sentence, or even a document. Based on the conceptually simple but empirically powerful language representations, Large-scale language models (LMs), like BERT [24] and RoBERTa [65], have achieved success in many QA benchmarks.

However, most of the current QA architectures directly utilize LMs to predict the answer but fall short of providing interpretable predictions. The semantic structures of the data and knowledge in the corpus are not explicitly stated but rather implicitly learned from a large corpus. It is thus difficult to create an explicit reasoning chain, capture high-order relations for the generalizability of reasoning, or establish the evidence used in the reasoning process. In other words, most of the existing deep learning QA works cannot track the explicit semantic relationships from various modalities, including Textual documents, Images (Cross-Modality QA), Knowledge graphs, etc.

In this dissertation, five critical challenges and contributions are addressed to make neural networks more effective for various QA tasks.

1.2 Challenges and Contributions of the Dissertation

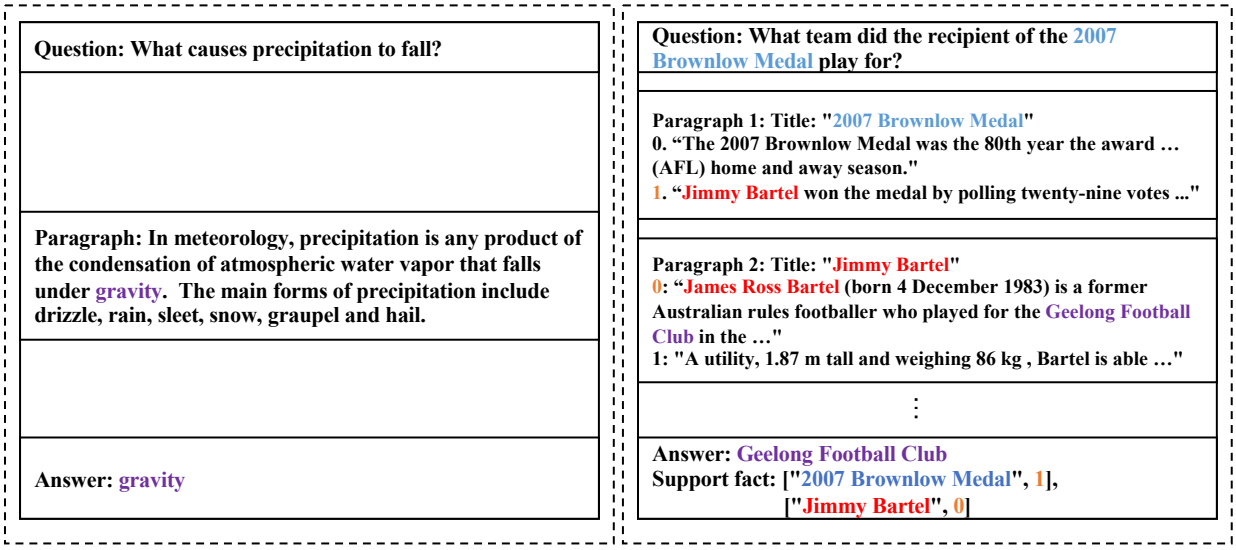

Figure 1.2 Two benchmark examples of the document-level Question Answering Task. The left side is the example of the SQuAD benchmark, while the right side is the example of the HotpotQA benchmark.

Challenge 1: Multi-hop Reasoning for Document-level QA. Answering questions over long documents usually requires the ability to understand the entities and find their connections throughout the whole document to be able to reason over them in multiple steps. The left side of Figure 1.2 shows an example of one-hop document-level QA from SQuAD benchmark [87]. Given the question "What causes precipitation to fall?" and a paragraph, we obtain the answer "gravity" after we read the sentence "Precipitation is any product of the condensation of atmospheric water vapor that falls under gravity". In contrast to one-hop question answering, where answers can be derived directly from a single paragraph [87, 86], many recent studies on question answering focus on multi-hop reasoning across multiple documents or paragraphs and aim to build multi-hop reasoning chains to capture the explicit semantic structure of the documents. In Section 2.2.1, we overview the related work about these recent studies in more detail. Even after successfully identifying a reasoning chain through multiple paragraphs, it remains a critical challenge to collect evidence from different

granularity levels (e.g., paragraphs, sentences, entities) for jointly predicting the answer and lines of reasoning. The right side of Figure 1.2 shows an example of complicated multi-hop reasoning QA task from HotpotQA benchmark [125]. Given the question "What team did the recipient of the 2007 Brownlow Medal play for?" and ten documents, we should find two sub-questions: 1) Who won the medal? 2) Where is this person playing? Then we filter out the two related documents and extract the reasoning chain: "2007 Brownlow Medal" → "Jimmy Bartel won the medal" → "Jimmy Bartel played for Geelong Football club". Finally, we obtain the answer "Geelong Football club" by the above line of reasoning.

Contribution: To solve the first challenge, we utilize the **semantic role labeling** to extract the semantic structure of the sentences. Semantic role labeling provides the semantic structure in terms of argument-predicate relationships [38], such as "who did what to whom." We innovatively construct a graph with entities and multiple relational edges from documents using semantic role labeling (SRL). We connected the SRL graphs using shared entities. This semantic structure of multiple documents can significantly improve the multi-hop reasoning capacity to find the line of reasoning to answer the questions. Then we use a graph neural model as the backbone to learning the graph node representations. We jointly train a multi-hop supporting fact prediction module that finds the cross-paragraph reasoning path, and an answer prediction module that obtains the final answer. Our experiments show that using the semantic structure of the document is effective in finding the cross-paragraph reasoning path and answering the question.

Questions and Answers:	**Document:**
1. Suppose tadpoles eat more food happens, how will it affect frogs? (A) **More** (B) Less (C) No effect 2. Suppose the weather is unusually bad happens, how will it affect the tadpoles that will need food? (A) More (B) **Less** (C) No effect	1. A frog lays eggs in the water. 2. Tadpoles develop inside of the eggs. 3. The eggs hatch. 4. The tadpoles eat and grow. 5. The tadpoles grow legs and form into frogs. 6. The frogs leave the water.

Figure 1.3 Two examples of the Cause-effect Question Answering Task.

Challenge 2: Causal Reasoning for Document-level QA. Cause-effect QA is a special type of Document-level QA. Causal reasoning requires the machine to effectively extract the explicit causal

relationships between cause and effect events (entities) over the entire document. For example, predicting a "sunny day" results in the direct effect of "sunshine" is less challenging than the indirect effect in "photosynthesis". Figure 1.3 shows an example of a cause-effect QA task from WIQA benchmark [107]. Given the procedural story and the question "Suppose tadpoles eat more food happens, how will it affect more frogs?", the following line of casual reasoning should be extracted from the text: "tadpole eat" → "tadpole grow," and "tadpole grow" → "tadpole form into frog". In Section 2.2.1, we overview the related work about finding the line of causal reasoning and limitations in more detail.

Contribution: To solve the second challenge, we aim to find relations between entities and the line of causal reasoning. Concretely, we build an *entity gating* module to extract and filter the involved entities in the question and context. Furthermore, we design a *relation gating* module with an alignment of entities to capture the higher-order chain of causal reasoning based on pairwise relations. Moreover, we propose an efficient module, called *contextual interaction module*, to incorporate cross-information from Question and Content interactions during training in an efficient way to help entities alignments.

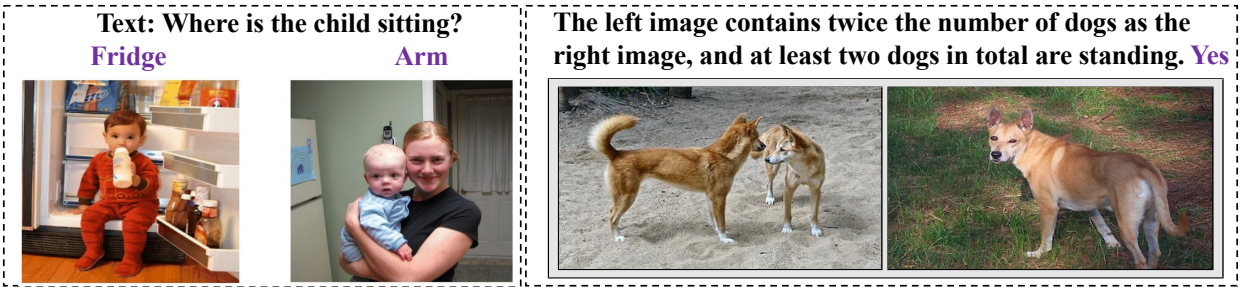

Figure 1.4 Two benchmark examples of the Cross-Modality Question Answering task. The left side is an example of the VQA benchmark, while the right side is an example of the NLVR benchmark.

Challenge 3: Relational Reasoning for Cross-Modality QA. In cross-modality QA, we require an understanding of both language and vision modalities and their connections and reason over them to be able to answer the questions. One line of research addresses this challenge by learning representations for cross-modality data and enabling reasoning for target tasks. This is done by the alignment of the representation for multiple modalities. Researchers develop models by

training features and aligning representation using Transformer architectures as the backbone [104]. However, these approaches have well-known issues for robust joint representations and reasoning for cross-modality QA [59]. In Section 2.2.3, we detailed describe the related work about these approaches. Explicit modeling of entities and relations in the neural model is one key factor that can alleviate this problem but is less explored. The right side of Figure 5.1 shows an example of a cross-modality QA task from NLVR benchmark [101]. Given two pictures and the statement, "The left image contains twice the number of dogs as the right image, and at least two dogs in total are standing," we should know the number of standing dogs in the left image and right image and reason over the twice number.

Contribution: To solve the third challenge, we aim to explicitly ground the entities as well as their relationships from language modality into vision modality. We proposed a novel cross-modality relevance (CMR) architecture that considers the relevance between textual token representations and visual object representations by explicitly aligning them in the two modalities. The relevance metric between two modalities is shown to be helpful for aligning multiple spaces of modalities in our work. We model the higher-order relational relevance for the generalizability of reasoning between entity relations in the text and object relations in the image.

Q: What is the largest island in the world?
A: Greenland.

The student practiced his guitar often, where is he always spent his free period?
A. Music room B. Toy store C. Concert

Figure 1.5 Two benchmark examples of the document-level Question Answering Task. The left side is the example of the WikiQA benchmark, while the right side is the example of the CommonsenseQA benchmark.

Challenge 4: Commonsense Reasoning for Knowledge based QA. Knowldge base QA is a task of answering questions given a structured source of knowledge, e.g. Knowledge Graph (KG). However, this task is challenging because firstly, often the given KG is very large, and secondly, the answer can not be directly retrieved, but multiple steps of reasoning over KG are needed to obtain the answer. The common approach for solving this problem is to extract a subgraph that is relevant to the question [30]. However, the challenge is that the extracted KG subgraph sometimes misses

some edges between entities, which breaks the chain of reasoning. This issue can be due to two possible scenarios. First, the KG is originally imperfect and does not include all the required edges. Second, when constructing the subgraph, some intermediate concept (entity) nodes and edges are omitted [30]. In such cases, the subgraph does not contain a complete chain of reasoning. The right side of Figure 1.5 shows an example from the CommonsenseQA benchmark. Given the question "The student practiced his guitar often, where is he always spent his free period?", the model should understand "free period" in the question and exploit the line of knowledge reasoning: "guitar" → "instrument (miss)" → "music room".

Contribution: To solve the fourth challenge, we aim to recover missing edges in the KG that were needed for finding the line of reasoning and answering the questions. We use ConceptNet [97], a general-domain knowledge graph, as the commonsense KG. ConceptNet graph has multiple semantic relational edges, e.g., HasProperty, IsA, AtLocation, etc. We extract the entities and retrieve related external knowledge from KG. Then, we construct a KG subgraph as part of the QA model to help fill the knowledge gaps and perform multi-hop reasoning. We proposed a novel Dynamic Relevance Graph Network (DRGN) that learns the node representations while exploiting the existing edges in KG and establishes new direct edges between graph nodes based on the relevance scores. As a byproduct, our model improved handling the negative questions due to deeply considering the relevance between the question node and the graph entities.

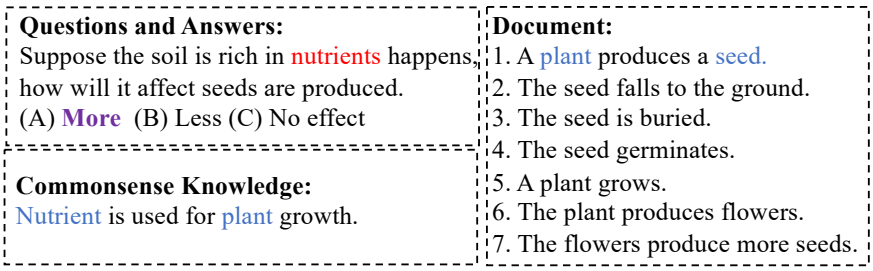

Figure 1.6 One example of Exploiting Commonsense Knowledge for Document-level QA.

Challenge 5: Exploiting Commonsense Knowledge for Document-level QA. Sometimes, answering questions over documents not only requires finding the line of the reasoning in the whole document but also exploiting the external knowledge to be able to help complete the Document-

level reasoning chain. However, the challenge is effectively extracting the most relevant external information and reducing the noise from the large KG. The irrelevant and noisy knowledge from KG will seriously mislead the QA model in predicting the answer. There are less sophisticated techniques proposed for using external knowledge explicitly in Document-level QA tasks [107, 106]. Figure 1.6 shows an example of Exploiting Commonsense Knowledge for Document-level QA. Given the question, "Suppose the soil is rich in nutrients happens, how will it affect seeds are produced", the model should understand "A plant produces a seed", and exploit the external knowledge "Nutrient is used for plant growth" to fill in the knowledge gap between the question and text and find the answer.

Contribution: To solve the fifth challenge, we aim to effectively learn to find the most relevant KG subgraph in a given large KG and combine that with the document-level information to answer the questions. We proposed a Multi-hop Reasoning network over Relevant CommonSense SubGraphs (MRRG) architecture that extracts the entities from the document and learns to retrieve the relevant external knowledge from KG using a novel KG attention neural mechanism [137]. Then, we construct a KG subgraph and use it as a part of the document-level QA model to help perform multi-hop reasoning and find the answer.

1.3 Outline of the Dissertation

The rest of this dissertation organizes as follows:

- In **Chapter 2**, following the Introduction Chapter, we describe the background and related works about document-level QA, cause-effect QA, cross-modality QA, and knowledge-based QA.

- In **Chapter 3**, we present our work on multi-hop reasoning for Document-level QA. We describe our **S**emantic **R**ole **L**abeling **G**raph **R**easoning **N**etwork (**SRLGRN**) for solving the multi-hop reasoning challenge. We clarify how it exploits the semantic structure of multiple documents based on semantic role labeling models and forms a novel multi-relational graph. We evaluate our SRLGRN architecture on the HotpotQA and SQuAD benchmark. The

experimental results and analysis indicate the effectiveness of SRLGRN on the Document-level QA task.

- In **Chapter 4**, we present our work on causal reasoning for Document-level QA. We describe an end-to-end **R**elational **G**ating **N**etwork (**RGN**) to solve the casual reasoning challenge. We clarify how it finds explicit causal relationships between entities facilitate causal reasoning over the whole document. We evaluate the model performance on the WIQA benchmark. The analysis demonstrates the effectiveness of the proposed entity gating module, relation gating module, and contextual interaction module in the RGN model.

- In **Chapter 5**, we present our work on relational reasoning for Cross-Modality QA. We describe a **C**ross-**B**odality **R**elevance (**CMR**) architecture to solve the challenges of cross-modality QA by learning and reasoning over visual and text. CMR considers the relevance between textual token representations and visual object representations by explicitly aligning them in the two modalities. We model the higher-order relational relevance for reasoning between entity relations in the text and object relations in the image. We evaluate the proposed CMR architecture on NLVR and VQA benchmarks. Moreover, we perform a detailed analysis of our CMR approach to show the effectiveness of entity relevance and relational reasoning.

- In **Chapter 6**, we present our work on commonsense reasoning for Knowledge based QA. We describe a novel **D**ynamic **R**elevance **G**raph **N**etwork (**DRGN**) to solve the commonsense reasoning challenge by exploiting the existing relations in KG and re-scaling the importance of the neighbor nodes in the graph based on training a dynamic relevance matrix. Our proposed approach shows competitive performance on two QA benchmarks, CommonsenseQA and OpenbookQA. The experiment results and analysis demonstrates that our DRGN model facilitates answering complex questions that need multiple hops of knowledge reasoning.

- In **Chapter 7**, we present our work on knowledge reasoning for document-level QA. We describe **M**ulti-hop **R**easoning Network over **R**elevant Commonsense Sub**G**raphs (**MRRG**) to solve the knowledge reasoning challenges by exploiting the external knowledge subgraph

extracted in the most relevant information from a large KG using a novel KG attention neural mechanism. We evaluate our model on the WIQA benchmark. The experimental results and analysis indicate that our MRRG model helps in filling the knowledge gaps between the question and the document and performing reasoning over the extracted knowledge.

- In **Chapter 8**, we draw the conclusion of the dissertation and several points for future direction.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we first provide a background of Transformer Architecture and Graph Neural Networks, which are the two main architectural components that we used in our neural models for QA systems. Then we introduce the related work about document-level QA, cause-effect QA, cross-modality QA, and knowledge based QA.

2.1 Background

2.1.1 Background of Transformer Architecture

Transformer Architecture is a stacked self-attention model for learning effective natural language features [111]. The Transformer has been shown to achieve extraordinary success in natural language processing not only for better performance but also for efficiency due to their parallel computations. The Transformers architecture uses a self-attention mechanism and multi-head attention as two key components to extract each token feature that helps in learning the features from all the other tokens trained in the huge natural language corpora [24, 124, 83].

Self-attention is the "soul" mechanism to learn token representations based on the Scaled Dot-Product operator. The input of Self-attention consists of Queries Q of dimension d_q, Keys K of dimension d_k, and Values V of dimension d_v. The Self-attention process is computed as follows:

$$SelfAttn(Q, K, V) = softmax(\frac{QK^T)}{\sqrt{d_k}}V, \tag{2.1}$$

where QK^T is the dot-product operation for Quires and Keys.

Multi-head attention is the "brain" module of the Transformer architecture, allowing for attending to parts of the sequence differently and running through self-attention mechanism h times in parallel. Then, all the single-head Attention outputs are combined together to obtain the integrated Attention output. The Multi-head attention is computed as follows:

$$MultiHead(Q, K, V) = Concat(Head_1, Head_2, \ldots, Head_h)W^O, \tag{2.2}$$

$$Head_i = SelfAttn(QW_i^Q, KW_i^K, VW_i^V), \tag{2.3}$$

where W^O, W^Q, W^K, W^V are learnable parameter matrices.

In recent years, Bidirectional Encoder Representations from Transformers (BERT) [24] and Robustly Optimized BERT Pretraining Approach (RoBERTa) [65] have been proposed and widely deployed in countless natural language processing tasks, especially in Question Answering [74, 139, 125]. We take BERT architecture as an example. BERT utilizes a bidirectional self-attention Transformer as the backbone to learning the pre-train deep bidirectional representations considering both left and right contexts from the large-scaled unlabeled corpus. Moreover, to better pre-train contextualized representations, BERT architecture employed two unsupervised tasks, Masked Language Model and Next Sentence Prediction. Furthermore, BERT uses an English Wikipedia corpus that contains 2, 500 million words and BooksCorpus [142] that contains 800 million words to train the architecture. However, the obstacle of BERT is the memory limitation because of millions or billions of parameters. To address this issue, ALBERT [55] utilizes two technologies, factorized embedding parameterization and cross-layer parameter sharing, to lower memory consumption and increase the training speed of BERT. Researchers also extended Transformers with both textual and visual modalities [59, 102, 104, 99, 109]. Sophisticated pre-training strategies were introduced to boost the performance [104].

2.1.2 Background of Graph Neural Networks

Graph Neural Network (GNN), which generalizes the deep learning neural network to structured graphs, has attracted increasing attention and gained valuable significance in various Natural Language Processing tasks, including Question Answering, Machine Reading Comprehension, etc. Graph Neural Networks can effectively learn robust representations from nodes, edges, and relations between the nodes in the structured graph [140, 37, 129]. Graph neural networks follow two types of approaches, which are spectral graph approaches and spatial graph approaches [23, 140, 52, 112]. **Spectral graph approaches** learn the spectral representation of the graphs. Spectral graph

methods commonly use the graph Fourier transform and graph convolution operator in the spectral domain [140]. Graph convolutional network (GCN) [52] is a classic multi-layer graph neural network and a typical spectral graph approach. For each layer of GCN, the node representations capture the information of their neighborhood nodes and edges via message passing and graph convolutional operation. R-GCN is a variation of GCN that deals with the multi-relational graph structure [90]. Adaptive Graph Convolution Network (AGCN) [60] learns the underlying relations and learns the residual graph Laplacian to improve spectral graph performance. Meanwhile, some variants of GCN replace the graph Fourier transform with other transform formats. For example, Graph Wavelet Neural Network (GWNN) [123] applies the graph wavelet transform to the graph, and achieves better performance compared to the graph Fourier transform in some tasks.

Spatial graph approaches learn the spatial graph representations based on the graph topology architecture and utilize the spatial information of the node directly [140, 7, 33]. For example, the graph attention network (GAT) [112] uses the graph attention layer and multi-head attention mechanism (like Transformer) on spatial graphs to learn the node representations efficiently. In particular, the graph attention layer consists of 3 components: (1) Linear Transformation, which is used to apply a learned parameter matrix to the feature vectors of the nodes, (2) Computation and Normalization of Attention Coefficients, which is used to determine the relative importance of neighboring features to each other, (3) Computation of Final Output Features, which is used to generate the Non-Linear Transformation node representations.

2.2 Related Work

In this dissertation, we address different types of QA that are categorized into four classes including document-level QA, cause-effect QA, cross-modality QA, and knowledge based QA. In the following subsections, we will describe the relevant benchmarks for each QA class and point to the related published QA architectures.

2.2.1 Document-level QA

Many QA tasks have been proposed to evaluate the language understanding capabilities of machines [87, 47, 28]. These tasks are single-hop QA and consider answering a question given only one single paragraph. These single-hop QA benchmarks, such as TriviaQA [47] and SearchQA [28], and MRC datasets, like SQuAD [87], rarely require complex reasoning (i.e., chain of reasoning) to obtain the answer. In these years, several multi-hop QA datasets, such as WikiHop [120] and HotpotQA [125], were proposed. They provide both multiple paragraphs and the ground-truth line of reasoning from question to answer. Those QA datasets require a multi-hop reasoning model to learn the cross-paragraph reasoning paths to predict the correct answer.

Primary studies prefer to use a neural retriever model and a neural reader model to solve the challenges of document-level multi-hop QA tasks [74, 139, 125]. First, they use a neural retriever model to find the relevant paragraphs to the question. After that, a neural reader model is applied to the selected paragraphs for answer prediction. Although these approaches obtain promising results, the performance of evaluating multi-hop reasoning capability is unsatisfactory [74]. Recently proposed multi-hop QA models [110, 122, 29] utilize the semantic structures of the data and construct a semantic graph in different ways. For example, Coref-GRN [25] utilizes co-reference resolution to build an entity graph. MHQA-GRN [96] is an updated version of Coref-GRN that adds sliding windows. Entity-GCN [13] builds the graph using entities and different types of edges called match edges and complement edges. DFGN [122] and SAE [110] construct an entity graph through named entity recognition (NER). Besides, some QA research works construct an entity graph using Spacy[1] or Stanford CoreNLP [71] and then apply a graph model to infer the entity path from question to the answer [14, 122, 16, 29].

In contrast to the models mentioned above, our SRLGRN replaces entity-based graphs with semantic role labeling graphs to take the semantic structure of the sentences into account. Semantic role labeling provides the semantic structure of the sentence in terms of argument-predicate relationships [141, 105, 72, 39, 38], such as "who did what to whom." In Chapter 3, our SRLGRN

[1] `https://spacy.io`

model utilizes graph convolutional network [52] as the backbone to learn the representations of the SRL graph, find the cross-paragraph reasoning path, and answer the question.

2.2.2 Cause-effect QA

Cause-effect Question Answering is a particular type of QA that aims to find relations between entities and the line of causal reasoning. Several new QA benchmarks were created in recent years for this purpose [20, 21]. In particular, WIQA benchmark [107] is proposed that aims solve the so called"what . . . if" kind of questions, containing multi-hop causal reasoning and commonsense reasoning, making the task more challenging.

Multiple previous works achieved impressive performance by modeling cause-and-effect entity representations on causal-effect QA [69, 44, 5, 107]. For example, REM-Net [44] architecture refines the evidence by a recursive memory mechanism and then uses a generative model to predict the answer. Logic-Guided [5] model uses logic rules, including symmetry and transitivity rules as regularization during training to impose consistency between the answers to multiple questions. However, these QA models fail to answer the questions when causal reasoning is required [27]. Therefore, we propose the Relational Gating Network (RGN) described in Chapter 4. RGN finds the line of causal reasoning and relations using entity gating and relation gating modules to solve the casual reasoning challenge.

Moreover, there are many cases in which the required knowledge for answering the question is not included in the document itself [107]. In other words, answering questions over documents not only requires finding the line of the reasoning in the whole document, but also exploiting the external knowledge to help complete the Document-level reasoning chain. EIGEN [69] builds an event influence graph based on a document and leverages LMs to create the chain of reasoning to predict the answer. However, EIGEN cannot deal with the challenge when the required knowledge is not in the given document. To address this challenge, we propose an MRRG architecture, described in Chapter 7, that captures the entities from the document and extracts external knowledge from KG.

2.2.3 Cross-Modality QA

Real-world problems often involve data from multiple modalities and resources. Learning and decision-making based on natural language and visual information have attracted the attention of many researchers due to exposing many exciting research challenges to the AI community. Solving a problem at hand usually requires reasoning about the components across all the involved modalities[62, 54, 46]. The VQA benchmark [4, 36] contains open-ended questions about images that require an understanding of and reasoning about language and visual components. In addition, the natural language visual reasoning (NLVR) [100, 101] benchmark is proposed that asks models to determine whether a sentence is matched with the image. Moreover, VQACP [1] was proposed to evaluate the capacity of language and visual understanding. Besides, several datasets contain extensive visual information such as bounding boxes, labels, etc, e.g., Flickr30k [81] and Visual Genome [54]. In addition, some visual Question Answering tasks aim to learn visual relation facts with a rich structure, such as FVQA [116], R-VQA [67], and KVQA [92]. The Video Question Answering task is a special type of visual Question Answering. Some related benchmarks were published, like PororoQA [75], Social-IQ [130], TVQA [56], and MovieQA [114], etc.

There are several challenges in learning and reasoning over cross-modality QA, including understanding visual contents, language semantics, and relationships between two modalities [45, 80, 82]. Researchers develop models by learning the joint features using Transformers architectures [59, 104]. For instance, VisualBERT [59] consists of Transformer layers that align textual and visual representation spaces with self-attention. LXMERT [104] aims to learn cross-modality encoder representations from a cross-Transformer architecture. Besides, LXMERT pre-trains the architecture with a large number of image-sentence pairs, via five diverse representative pre-training tasks. Moreover, contrastive learning positively influences learning robust joint representations for two modalities [82]. On the vision side, contrastive loss brings visual representations of two similar images closer together while distinguishing the representations of two dissimilar images [68, 50]. On the language side, contrastive loss makes two language representations closer [34, 117]. However, those approaches do not consider relational reasoning [82].

In contrast to these methods, we proposed a novel cross-modality relevance (CMR) architecture in Chapter 5 that exploits the textual and visual entities and relations and finds their relevance in the two modalities for learning joint representations. In addition, we model the higher-order relational relevance for aligning not only textual/visual entities but also the relations between them in the text and image.

2.2.4 Knowledge based QA

Using structured knowledge is another type of modality that can feed the QA systems for answering natural language questions. Some benchmarks for QA systems that provide structured sources of knowledge were published in recent years [26], such as QALD [15], WebQuestions [9], SimpleQuestion [11], and KBQA [19]. To answer the questions in these benchmarks, the structured sources of explicit knowledge are different from each other. Specifically, WebQuestions and SimpleQuestions contain questions that can be answered using Freebase [10], while QALD uses DBpedia [8] as the knowledge source. Meanwhile, CommonsenseQA [103] and OpenbookQA [73] are two benchmarks focusing on commonsense question answering that required external knowledge provided in ConceptNet [97].

However, current QA models can not effectively utilize the KG's information [30] and mostly rely on implicit knowledge stored in large language models [24, 30]. The reason is that the existing KGs are usually large and contain many nodes that are irrelevant to the question and text. Moreover, with larger KGs, the computational complexity of learning over them will increase. To deal with this issue, pruning KG nodes based on a variety of metrics has been proposed [23, 140, 112, 37, 129]. Moreover, GraphTransformer[53] and QAGNN [127] include the sentence node in the graph, while HGN [29] and SRLGRN [134] add the paragraph node and sentence node to construct a hierarchical graph structure. However, the extracted KG subgraph sometimes misses some edges between entities, which breaks the chain of reasoning [136].

To solve this challenge, in contrast to the models mentioned above, we proposed a Dynamic Relevance Graph Network (DRGN), described in Chapter 6, that learns the node representations

while exploiting the existing edges in KG and establishes new direct edges between graph nodes based on the relevance scores.

CHAPTER 3

MULTI-HOP REASONING FOR DOCUMENT-LEVEL QA

3.1 Background and Motivation

Understanding and reasoning over natural language play a significant role in Machine Reading Comprehension (MRC) and Question Answering (QA). Several types of QA tasks have been proposed in recent years to evaluate the language understanding capabilities of machines [87, 47, 28]. However, most of the current benchmarks focus on simple single-hop QA problems over a single paragraph. Many existing neural models rely on learning context and type-matching heuristics [119]. Those rarely build reasoning modules but achieve promising performance on single-hop QA tasks. The main reason is that these single-hop QA tasks lack an in-depth evaluation of the reasoning capabilities of the learning models because they do not require complex reasoning.

Figure 3.1 An example of HotpotQA data.

Recently multi-hop QA benchmarks, such as HotpotQA [125] and WikiHop [120], have been proposed to assess the multi-hop reasoning ability of the learning models. HotpotQA task provides annotations to evaluate document-level question answering and finding supporting facts. Providing supervision for supporting facts improves the explainability of the predicted answer because they clarify the cross-paragraph reasoning path. Due to the requirement of multi-hop reasoning over multiple documents with strong distractions, multi-hop QA tasks are challenging. Figure 3.1 shows an example of HotpotQA. Given a question and 10 paragraphs, only paragraph 1 and paragraph 2 are relevant. The second sentence in paragraph 1 and the first sentence in paragraph 2 are the supporting facts. The answer is "Geelong Football Club".

Primary studies in HotpotQA task prefer to use a reading comprehension neural model [74, 139, 125]. First, they use a neural retriever model to find the relevant paragraphs to the question. After that, a neural reader model is applied to the selected paragraphs for answer prediction. Although these approaches obtain promising results, the performance of evaluating multi-hop reasoning capability is unsatisfactory [74].

To solve the multi-hop reasoning problem, some previous models tried to construct an entity graph using Spacy[1] or Stanford CoreNLP [71] and then applied a graph model to infer the entity path from question to the answer [14, 122, 16, 29]. However, these models ignore the importance of the semantic structure of the sentences and the edge information and entity types in the entity graph. To take the in-depth semantic roles and semantic edges between words into account, here we use semantic role labeling (SRL) graph as the backbone of a graph convolutional network. Semantic role labeling provides the semantic structure of the sentence in terms of argument-predicate relationships [38]. The argument-predicate relationship graph can significantly improve the multi-hop reasoning results. Our experiments show that SRL is effective in finding the cross-paragraph reasoning path and answering the questions.

Our proposed Semantic Role Labeling Graph Reasoning Network (SRLGRN) jointly learns to find cross-paragraph reasoning paths and answer questions on multi-hop QA. In the SRLGRN

[1] `https://spacy.io`

model, firstly, we train a paragraph selection module to retrieve gold documents and minimize distractors. Second, we build a heterogeneous document-level graph that contains sentences as nodes (question, title, and sentences) and SRL sub-graphs, including semantic role labeling arguments as nodes and predicates as edges. Third, we train a graph encoder to obtain the graph node representations that incorporate the argument types and the semantics of the predicate edges in the learned representations. Finally, we jointly train a multi-hop supporting fact prediction module that finds the cross-paragraph reasoning path and answers prediction module that obtains the final answer. Notice that both supporting fact prediction and answer prediction are based on contextual semantics graph representations as well as token-level BERT pre-trained representations. The contributions of this work are as follows:

1) We propose the SRLGRN framework that considers the semantic structure of the sentences in building a reasoning graph network. Not only the semantics roles of nodes but also the semantics of edges are exploited in the model.

2) We evaluate and analyze the reasoning capabilities of the semantic role labeling graph compared to usual entity graphs. The fine-grained semantics of the SRL graph help in both finding the answer and the explainability of the reasoning path.

3) Our proposed model obtains competitive results on both HotpotQA (Distractor setting) and the SQuAD benchmarks.

3.2 Semantic Role Labeling Graph Reasoning Network

Our proposed SRLGRN approach is composed of Paragraph Selection, Graph Construction, Graph encoder, Supporting Fact prediction, and Answer Span prediction modules. Figure 3.2 shows the proposed architecture. In this section, we introduce our approach in detail and then explain how to train it with an efficient algorithm.

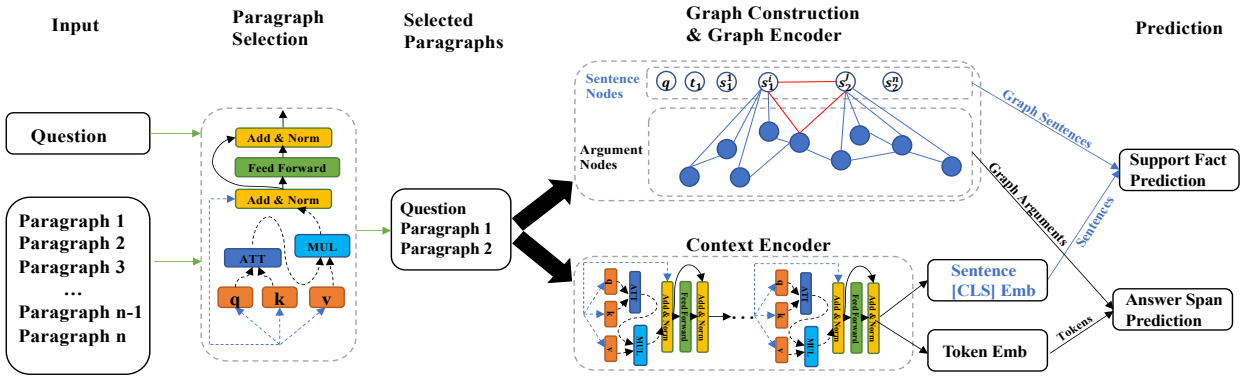

Figure 3.2 Our proposed SRLGRN model is composed of Paragraph Selection, Graph Construction, Graph Encoder, Supporting Fact prediction, and Answer Span prediction.

3.2.1 Problem Formulation

Formally, the problem is to predict supporting fact y_{SF} and answer span y_{ans} given input question q and candidate paragraphs. Each paragraph content $C = \{t, s_1, \ldots, s_n\}$ includes title t and several sentences $\{s_1, \ldots, s_n\}$.

3.2.2 Paragraph Selection

Most of the paragraphs are distractors in the HotpotQA task [125]. SRLGRN can select gold documents and minimize distractors from given N documents by a Paragraph Selection module. The Paragraph Selection is based on the pre-trained BERT model [24]. Our Paragraph Selection module has two phases explained in section 3.2.2.1 and section 3.2.2.2.

3.2.2.1 First Round Paragraph Selection

For every candidate paragraph, we take the question q and the paragraph content C to form the text input $[[CLS]; q; [SEP]; C]$, where [CLS] and [SEP] are the BERT special tokens in the tokenizer process [24]. We form the input and feed it into a pre-trained BERT encoder to obtain token representations. Then we use $BERT_{[CLS]}$ token representation as the summary representation of the paragraph. Meanwhile, we utilize a two-layer MLP to output the relevance score, y_{sel}. The paragraph which obtains the highest relevance score is selected as the first relevant context. We

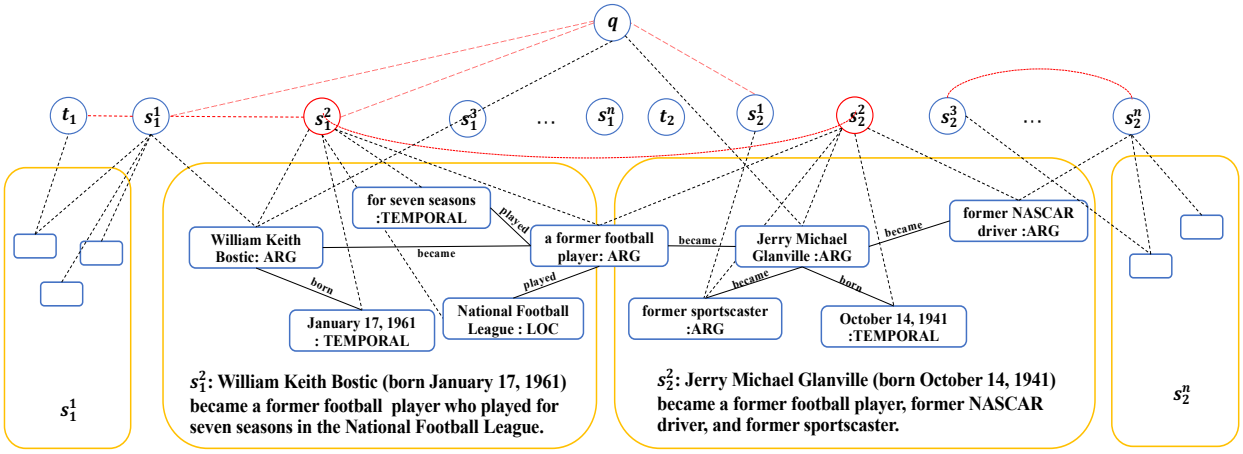

Figure 3.3 An example of Heterogeneous SRL Graph. The question is "Who is younger Keith Bostic or Jerry Glanville?" The circles show the document-level nodes, i.e., sentences. The blue squares show the argument nodes. The argument nodes include argument phrases and argument-type information. The solid black lines are semantic edges between two arguments carrying the predicate information. The black dashed lines show the edges between sentence nodes and argument nodes. The red dashed lines show the edges between two sentences if there exists a shared argument (based on an exact string match). The orange blocks are the SRL argument-predicate sub-graphs for sentences. s_i^j means the j-th sentence from the i-th paragraph.

concatenate q to the selected paragraph as q_{new} for the next round of paragraph selection.

3.2.2.2 Second Round Paragraph Selection

For the remaining $N-1$ candidate paragraphs, we use the same model as first-round paragraph selection to generate a relevance score that takes q_{new} and paragraph content as input. We call this process as second-round paragraph selection. Similar to section 3.2.2.1, one of the remaining candidate paragraphs with the highest score is selected. Afterward, we concatenate the question and the two selected paragraphs to form a new context used as the input text for the graph construction.

3.2.3 Heterogeneous SRL Graph Construction

We build a heterogeneous graph that contains document-level sub-graph $\mathcal{S}$ and argument-predicate SRL sub-graph Arg for each data instance. In the graph construction process, the document level sub-graph $\mathcal{S}$ includes question q, titles t, and sentences s from the selected paragraphs. The argument-predicate SRL sub-graphs Arg, including arguments as nodes and the predicates as

edges, are generated using AllenNLP-SRL model [94]. Each argument node is the concatenation of argument phrase and argument type, including "TEMPORAL", "LOC", etc.

Figure 3.3 describes the construction of the heterogeneous graph. The edges of the heterogeneous graph are added as follows: **1)** There will be an edge between a sentence and an argument if the argument appears in the sentence (the black dashed lines in Figure 3.3); **2)** Two sentences will have an edge if they share an argument by exact matching (the red dashed lines); **3)** Two argument nodes Arg_i and Arg_j will have an edge if they share a predicate (the black solid lines); **4)** There will be an edge between the question and a sentence if their arguments exactly matching their lexical surface (the red dashed lines). Figure 3.3 shows an example of a heterogeneous SRL graph. s_1^2 and s_2^2 are connected because of a shared argument node "a former football player: ARG". Besides, the shared argument node has several semantic edges, such as "played" and "became". In this way, the shared argument node and other connected argument nodes have argument-predicate relationships.

We create two matrices based on the constructed graph. We will describe the way we use these matrices in section 3.2.4. We build a weight matrix K to express the predicate-based semantics of the edges and a weight matrix A to express various types of edges.

The semantic edge matrix K is a matrix that stores the word index of the predicates that is shared between the two arguments. We initialize all the elements of K with empty, $\emptyset$. If two argument nodes Arg_i and Arg_j are related to the same predicate, we add that predicate word index to $K_{(Arg_i, Arg_j)}$. A is a matrix that stores different types of edge weights. We divide the edges into three types: sentence-argument edges, argument-argument edges, and sentence-sentence edges.

The weight of a sentence-sentence edge is 1 when two sentences share an argument. Meanwhile, the weight of a sentence-argument edge is 1 if there exists an edge between a sentence and an argument. If two argument nodes have an edge, the weight can be calculated by point-wise mutual information (PMI) [12]. The reason we use PMI is that it can better explain associations between nodes compared to the traditional co-occurrence count method [126].

3.2.4 Graph Encoder

Here we provide a background to Graph convolutional network that we use in our model to obtain graph embedding. We introduce the Graph Convolution Network [52] to obtain the graph embeddings. The Graph Convolution Network (GCN) is a multi-layer network that uses the graph input directly and generates embedding vectors of the graph.

GCN plays an essential role in incorporating neighborhood nodes and helps in capturing the structural graph information. The SRL graph uses the semantic structure of the sentence to form the graph nodes and semantic edges. For instance, the GCN nodes of the document level sub-graph help in finding the supporting fact path, while GCN nodes of the argument-predicate level sub-graph help in identifying the text span of the potential answers. In this work, we consider a two-layer GCN to allow message-passing operations and learn the graph embeddings. The graph embeddings are computed as follows:

$$G' = (D^{-\frac{1}{2}} A D^{-\frac{1}{2}})[X_{Arg}; X_{\mathcal{S}}]W_1, \tag{3.1}$$

$$G = (D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) f(G')W_2, \tag{3.2}$$

where G' and G are graph embedding outputs of two GCN layers that incorporate higher-order neighborhood nodes by stacking GCN layers. $f(x)$ is an activation function, D is the degree matrix of the graph [52], A is the heterogeneous edge parameters matrix, and W_1 and W_2 are the learned parameters. X represents node embeddings, including argument-predicate embedding X_{Arg} and sentence embedding $X_{\mathcal{S}}$. Given graph embedding G, we use $G_{\mathcal{S}}$ to represent document-level node embeddings, and G_{Arg} to represent argument-predicate level node embeddings.

3.2.5 Supporting-Fact Prediction

The goal of supporting fact (SF) prediction is to find the supporting fact path that is necessary to obtain the answer. Inspired by previous research [6], we utilize RNN with a beam search to find the best document-level supporting fact path. This approach turns out to be effective for selecting the

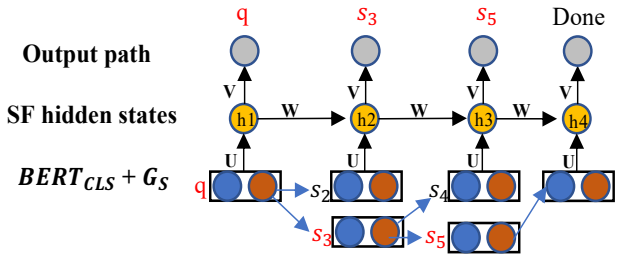

Figure 3.4 An example of Supporting Fact Prediction.

SF reasoning path. Notice that, our supporting fact prediction not only relies on BERT and RNN but also incorporates document-level graph node embeddings G_S.

Formally, we use the concatenation of the graph sentence embedding, G_S (blue circles in Figure 3.4), and BERT's [CLS] representation of the candidate sentence (orange circles) to represent the candidate supporting fact sentence X_S^{cand}. The process of selecting supporting facts is as follows:

$$h_t = \sigma(Wh_{t-1} + UX_S^{cand} + b_h), \tag{3.3}$$

$$o_t = Vh_t + b_o, \tag{3.4}$$

where h_t is the hidden state of the RNN at the t-th supporting fact selection step, σ is the activation function. W, U, V, b_h and b_o are the parameters.

Finally, we use the beam search to output SF paths, choosing the highest scored path as our final supporting fact answer y_{SF}:

$$y_{\text{SF}} = \arg\max_{1 \le t \le T} \prod o_t, \tag{3.5}$$

where T is the maximum number of reasoning hops. We penalize with the cross-entropy loss.

Figure 3.4 shows an example of the predicted SF process. Based on the constructed heterogeneous graph, two sentence nodes have an edge if they share an argument. We start from question node q as the first input sentence. Since q is a unique input, we select q as the first SF candidate. In the second step, two candidate sentence nodes, s_2 and s_3, which are neighbor nodes of q, are chosen as the input. We separately feed s_2 and s_3 to the RNN layers. The sentence s_3 that obtains a larger logit score is selected as the second SF candidate of the reasoning path. In the third step, s_4 and s_5

are neighbor nodes of the second SF, s_3. Then the model chooses s_5 as the third SF. In the end, s_1, s_3, and s_5 are the supporting facts.

3.2.6 Answer Span Prediction

The goal of the answer prediction module is to output "yes", "no", or answer span for the final answer. We first design an answer type classification based on BERT and an additional two fully connected feed-forward layers. If the highest probability of type classification is "yes" or "no", we directly output the answer. The input of type classification is $BERT_{[CLS]}$. The answer type y_{type} can be calculated as

$$y_{\text{type}} = MLP_{\text{type}}([BERT_{[CLS]}]). \tag{3.6}$$

If the answer is not "yes" or "no", we compute the logit of every token to find the start position i and end position j for the answer span. The input token representation is the concatenation of BERT token representation $BERT_{tok}$ and graph embedding G_{Arg}. The answer span y_{ans} can be computed as

$$y_{\text{ans}} = \arg\max_{i,j,\, i \le j} y^i_{start} y^j_{end}, \tag{3.7}$$

$$y^i_{\text{start}} = MLP_{\text{start}}([BERT^i_{tok}; G^i_{Arg}]), \tag{3.8}$$

$$y^i_{\text{end}} = MLP_{\text{end}}([BERT^i_{tok}; G^i_{Arg}]), \tag{3.9}$$

where y_{ans} is the index pair of (start position, end position), y^i_{start} represents the logit score of the i-th word as the start position, and y^i_{end} represents the logit score of the i-th word as the end position.

3.2.7 Objective Function

Inspired by [122] and [110], the joint objective function includes the sum of cross-entropy losses for the span prediction L_{ans}, answer type classification L_{type}, and supporting fact prediction L_{SF}. The loss function is computed as follows:

$$L_{\text{joint}} = L_{\text{ans}} + L_{\text{SF}} + L_{\text{type}}$$

$$= \lambda_1(-y_{\text{start}} \log y_{\text{start}} - y_{\text{end}} \log y_{\text{end}}) - \lambda_2 y_{\text{SF}} \log y_{\text{SF}} - \lambda_3 y_{\text{type}} \log y_{\text{type}},$$

where λ_1, λ_2, and λ_3 are hyper-parameters which are weighting factors indicating the importance of each component in the loss.

3.3 Experiments

3.3.1 Dataset Description

We use the HotpotQA dataset [125], a popular benchmark for multi-hop QA tasks, for the main evaluation of the SRLGRN. For each question in the Distractor Setting, two gold paragraphs and 8 distractor paragraphs, which are collected by a high-quality TF-IDF retriever from Wikipedia, are provided. Only gold paragraphs include ground-truth answers and supporting facts. In addition, we use Machine Reading Comprehension task, Stanford Question-Answering Dataset (SQuAD) v1.1 [87] and v2.0 [86], to demonstrate the language understanding ability of our model.

3.3.2 Implementation Details

We implemented SRLGRN using PyTorch. We use a pre-trained BERT-base language model with 12 layers, 768-dimensional hidden size, 12 self-attention heads, and around $110M$ parameters [24]. We keep 256 words as the maximum number of words for each paragraph. For the graph construction module, we utilize a semantic role labeling model [94] from AllenNLP[2] to extract the predicate-argument structure. For the graph encoder module, we use 300-dimensional GloVe [79] pre-trained word embedding. The model is optimized using Adam optimizer [51].

3.3.3 Baseline Models

In this subsection, we select three SOTA models as our main baselines. In particular, Multi-Paragraph Reading Comprehension Model [16] uses a neural retriever model and a neural reader model to find

[2] `https://demo.allennlp.org/semantic-role-labeling.`

the span answer. In addition, we select DFGN [122] and SAE [110] models that construct entity graphs through named entity recognition (NER) as the backbone to find the supporting fact path.

Multi-Paragraph Reading Comprehension Model [16] is our first strong baseline. This baseline model combines the popular technical neural modules as the components in the QA domain, including self-attention and bi-attention modules [91].

DFGN [122] is a strong baseline method for the HotpotQA task. DFGN builds an entity graph from the text. Moreover, DFGN includes a dynamic fusion layer that helps in finding relevant supporting facts.

SAE [110] is the recent SOTA model that is an effective Select, Answer and Explain system for multi-hop QA. SAE is a pipeline system that first selects the relevant paragraphs and uses the selected paragraphs to predict the answer and the supporting fact.

3.4 Experimental Results and Analysis

3.4.1 Results

Model	**Ans**(%)		**Sup**(%)		**Joint**(%)	
	EM	F1	EM	F1	EM	F1
Baseline Model [125]	45.60	59.02	20.32	64.49	10.83	40.16
KGNN [128]	50.81	65.75	38.74	76.79	22.40	52.82
QFE [76]	53.86	68.06	57.75	84.49	34.63	59.61
DecompRC [74]	55.20	69.63	-	-	-	-
DFGN [122]	56.31	69.69	51.50	81.62	33.62	59.82
TAP	58.63	71.48	46.84	82.98	32.03	61.90
SAE-base [110]	60.36	73.58	56.93	84.63	38.81	64.96
ChainEx [14]	61.20	74.11	-	-	-	-
HGN-base [29]	-	74.76	-	**86.61**	-	**66.90**
SRLGRN-base	**62.65**	**76.14**	**57.30**	85.83	**39.41**	66.37

Table 3.1 HotpotQA Result on Distractor setting. Except for the Baseline model, all models deploy BERT-base uncased as the pre-training language model to compare the performance.

Evaluation metrics. In the HotpotQA benchmark, two sub-tasks are included in this dataset: Answer prediction and Supporting facts prediction. For each sub-task, Exact Match (EM) and Partial Match (F1) are two official evaluations that were proposed in [87]. Given the precision and

recall of the answer span prediction and the supporting facts, respectively, the joint Exact Match (EM) and joint Partial Match (F1) scores are computed to evaluate the model performance on the HotpotQA Distractor Setting.

Table 3.1 shows the results of HotpotQA (Distractor setting). We can observe the SRLGRN model outperforms the previous state-of-the-art results in most of the evaluation criteria. Our model obtains a Joint Exact Matching (EM) score of 39.41% and a Joint Partial Matching (F1) score of 66.37% that combines the evaluation of answer spans and supporting facts. Our SRLGRN model has a significant improvement, about 28.58%, on Joint EM and 26.21% on F1, over the Baseline Model [125]. Compared to the current published state of the art, i.e. SAE model [110], SRLGRN improves the results for 2.29% on the Joint Exact Match and 2.56% on the joint F1. To our analysis, the main reason for the effectiveness of our model is that it uses not only token-level BERT representations but also uses graph-level SRL node representations that help in learning the line of the multi-hop reasoning.

3.4.2 Model Analysis

Our framework provides an effective way for multi-hop reasoning taking advantage of the SRL graph and the pre-trained language models. In the following section, we give a detailed analysis of the SRLGRN model.

Effect of SRL Graph. The SRL graph extracts argument-predicate relationships, including in-depth semantic roles and semantic edges. The constructed graph is the basis of reasoning as the inputs of each hop are directly selected from the SRL graph, as shown in Figure 3.4. The SRL graph provides a rich graph network, that is, providing the key semantic edges between the words to cover reasoning paths, see Figure 3.3.

Compared to the NER graph in the previous models [122], the proposed SRL graph covers the 86.5% of reasoning paths for the data samples. The NER graph of DFGN can only cover 68.7% of the reasoning paths [122]. The coverage of the semantic edges in the graph is one major reason

that the SRLGRN model has higher accuracy compared to other published models. As shown in Table 3.1, the SRLGRN improves 5.79% on joint EM and 6.55% on joint F1 over DFGN that is based on the NER graph.

Ablation	**Model**	**Ans**(%)	
		EM	F1
Graph	w/o graph	53.06	67.68
	w/o Argument type and Semantic edge	60.10	73.24
Joint	w/o joint training	58.50	71.58
Language	ALBERT-base	59.87	74.20
	BERT-base	62.65	76.14

Table 3.2 SRLGRN ablation study on HotpotQA.

To evaluate the effectiveness of the types of semantic roles and the edge types, we perform an ablation study. First, we removed the whole SRL graph. Second, we removed the predicate-based edge information from the SRL graph. Table 3.2 shows the results. The complete SRLGRN improves 8.46% on the F1 score compared to the model without the SRL graph. The model loses the connections used for multi-hop reasoning if we remove the SRL graph and only use BERT for answer prediction.

We also observe that the F1 score of answer span prediction decreases 2.9%, if we did not incorporate semantic edge information and argument types. The reason is that removing predicate edges and argument types will destroy the argument-predicate relationships in the SRL graph and breaks the chain of reasoning. For example, in Figure 3.3, the main arguments of the two supporting facts in s_1^2 and s_2^2 (William and Jerry) are connected with a predicate edge, "born", to the temporal information necessary for finding the answer. Both the "born" edge and the adjunct temporal roles are the key information in the two sentences to find the final answer to this question. The shared ARG node, "football player", also helps to connect the line of reasoning between the two sentences. These two results indicate that both semantic roles and semantic edges in the SRL graph are essential for the SRLGRN performance.

In a different experiment, we tested the influence of the joint training of the supporting facts and

answer prediction. As shown in Table 3.2, the performance will decrease by 4.56% when we do not train the model jointly.

Effect of Language Models. We use two popular and widely-used pre-trained language representation models, BERT and ALBERT [55]. The last two lines of Table 3.2 show the results. Although BERT achieves relatively better performance, ALBERT architecture has significantly fewer parameters (18x) and is faster (about 1.7x running time) than BERT. In other words, ALBERT reduces memory consumption by cross-layer parameter sharing, increases the speed, and obtains a satisfactory performance.

Effect of SRLGRN on Single-hop QA. We evaluate the SRLGRN (excluding the paragraph selection module) on SQuAD [87] to demonstrate its reading comprehension ability. We evaluate the performance on both SQuAD v1.1 and SQuAD v2.0. Table 3.3 describes the results of several baseline methods on SQuAD v1.1. Our model obtains a 1.8% improvement over BERT-large and a 1.6% improvement over BERT-large+TriviaQA [24].

Model	**Ans**(%)	
	EM	F1
Human	82.3	91.2
BERT-base	80.8	88.5
BERT-large	84.1	90.9
BERT-large+TriviaQA	84.2	91.1
BERT-large+SRLGRN	85.4	92.7

Table 3.3 SQuAD v1.1 performance.

We further test the SRLGRN on SQuAD v2.0. The main difference is that SQuAD v2.0 combines answerable questions (like SQuAD v1.1) with unanswerable questions [86]. Table 3.4 shows that our proposed approach improves the performance of the SQuAD benchmark compared to several recent strong baselines.

We recognize that our SRLGRN improves 7.1% on EM compared to the robust BERT-large model and improves 1.0% on EM compared to SemBERT [132]. The two experiments on SQuAD

Model	Ans(%)	
	EM	F1
Human	86.3	89.0
ELMo+DocQA [86]	65.1	67.6
BERT-large [24]	78.7	81.9
SemBERT [132]	84.8	87.6
BERT-large+SRLGRN	85.8	87.9

Table 3.4 SQuAD v2.0 performance.

v1.1 and SQuAD v2.0 demonstrate the significance of the SRL graph and the graph encoder.

Error Type	Model Prediction	Label
Synonyms	washington dc sars ey writer	district of columbia severe acute respiratory syndrome ernst young author
MLV	australian hessian mcdonald's, co	australia hessians mcdonalds
Month-Year	1946 25, november, 2015 10, july, 1873	1945 3, december 1, september, 1864
Number	11 fourth 2402	10 4 5922
External Knowledge	Coker	NCAA I FBS football
Other	taylor, swift film fourteenth	usher documentary 500th episode

Table 3.5 Error types on HotpotQA dev set.

3.4.3 Qualitative Analysis

Synonyms Answers is the most frequent cause of the reported errors in many cases where the predicted answer is semantically correct. As shown in the first row of Table 3.5, our predicted answer and gold label have the same meaning. For example, SRLGRN predicts "sars", while the

label is "severe acute respiratory syndrome." We know that "sars" is the abbreviation of the gold label.

Successful Cases	Comparison	Question: Which is a type of herb, Brassia or Achimenes? Supporting Facts: 1. Brassia is a genus of orchids classified in the Oncidiinae subtribe. 2. Achimenes is a genus of about 25 species of tropical and subtropical rhizomatous perennial herbs in the flowering plant family Gesneriaceae. Answer: Achimenes
	Bridge	Question: When was the University established where Laura Landweber is a professor? Supporting Facts: 1. As of 2016, she is a professor of biochemistry and molecular biophysics and of biological sciences at Columbia University. 2. Columbia University (Columbia; officially Columbia University in the City of New York), established in 1754, is a private Ivy League research university in Upper Manhattan, New York City, often cited as one of the world's most prestigious universities. Answer: 1754
Failing Cases	Wrong Paragraph Selection	Question: Luke Null is an actor who was on the program that premiered its 43rd season on which date? Wrong Paragraph Selection: 1. Luke Null 2. 43rd Battalion (Australia) Label Paragraphs Selection : 1. Luke Null 2. Saturday Night Live Supporting Facts: 1. Luke Null is an American actor, comedian, and singer, who currently works as a cast member on "Saturday Night Live", having joined the show at the start of its forty-third season. 2. The forty-third season of the NBC comedy series "Saturday Night Live" premiered on September 30, 2017 with host Ryan Gosling and musical guest Jay-Z during the 2017-2018 television season. Answer: September 30, 2017
	Comparison	Question: Who is younger, Wayne Coyne or Toshiko Koshijima? Supporting Facts: 1. Wayne Michael Coyne (born January 13, 1961) is an American musician. 2. Toshiko Koshijima (こしじま としこ , Koshijima Toshiko , born March 3, 1980 in Kanazawa, Ishikawa) is a Japanese singer. Wrong Answer: Wayne Coyne Answer: Toshiko Koshijima
	Bridge	Question: What Division was the college football team that fired their head coach on November 24, 2006? Supporting Facts: 1. The 2006 Miami Hurricanes football team represented the University of Miami during the 2006 NCAA I FBS football season. 2. Coker was fired by Miami on November 24, 2006 following his sixth loss that season. Wrong Answer: Coker Label Answer: NCAA Division I FBS football

Figure 3.5 Successful cases and Failing cases on our proposed SRLGRN framework.

Minor Lexical Variation (MLV) is another major cause of mistakes in the SRLGRN model. As shown in the second row of Table 3.5, our model's predicted answer is "Australian", while the gold label is "Australia". Many wrong predictions occur in the singular noun versus plural noun selection.

Dates and numbers are other common mistakes. By looking into the graphs, we observed that sometimes SRLGRN predicts the wrong answer when two or more arguments of the same type, in particular with "TEMPORAL" types of arguments, are connected to the same predicate. In such cases, it is hard to disambiguate the actual time that is the answer to the question.

Paragraph Selection is the cause of a small portion of errors in the SRLGRN model. As shown in Figure 3.5, the model chooses the wrong paragraph "43rd Battalion". The reason is that "43rd" is a distractor since the "43rd season" appears in the question. The paragraph "Saturday Night Live" is the correct relevant paragraph that includes both "forty-third season" and the true answer. To

resolve this issue in the future, we will try to combine our model with a robust retrieval module designed for multi-hop QA similar to the Multi-step entity-centric model proposed in [35].

Comparison questions seem to be a source of error when answering the questions in HotpotQA. For example, as is shown in Figure 3.5, the question is " Who is younger, Wayne Coyne or Toshiko Koshijima?" To correctly answer the "comparison" type of question, the model requires the ability to compare two entities that existed in the question. In the failing case of Figure 3.5, we predict the wrong answer "Wayne Coyne". The model keeps answering "Wayne Coyne" even after replacing the word "younger" with "older", which happens to be the correct answer this time.

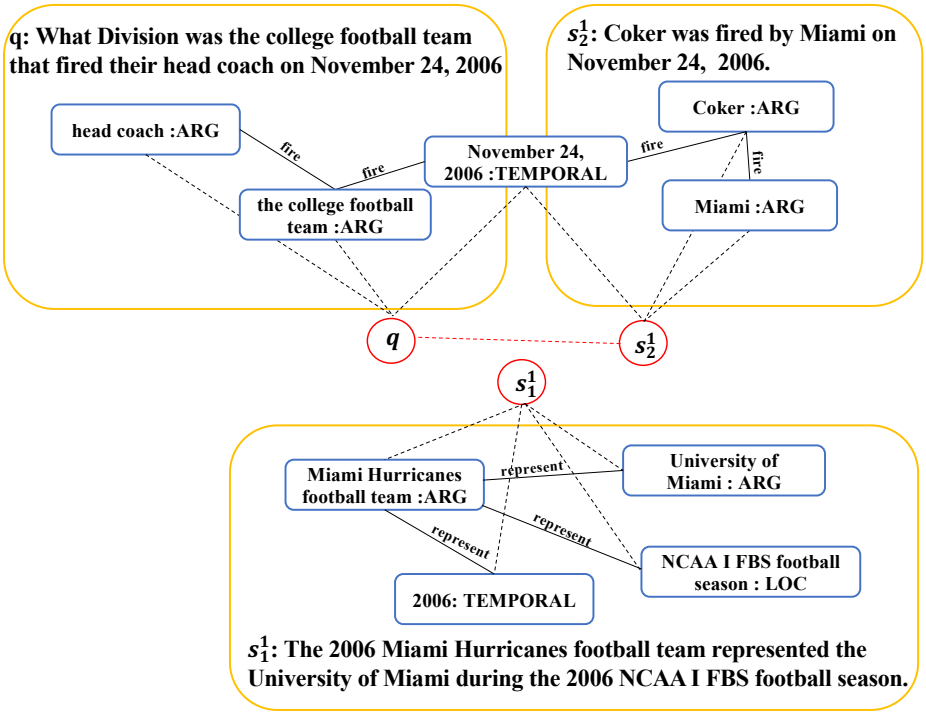

Figure 3.6 The "Disconnection" failing case that SRL fails to lead to the correct answer. The meaning of different lines and node colors are the same as Figure 3.3.

Disconnection is another type of error. While the SRL graph helps if finding the chain of reasoning, in some cases, the line of reasoning was broken. By looking into the errors, we realized in most cases, obtaining the answer needed multiple hops of reasoning while external knowledge was required to form the connections. As is shown in "Disconnection" failing cases of Figure 3.5,

the selected paragraphs do not show the relation between “Coker” and “Miami Hurricanes football team”. Figure 3.6 describes the SRL construction for this failing case. The second supporting fact and the question have the same temporal argument node “November 24, 2006”. However, there is no chain between the first supporting fact and the second supporting fact due to the lack of external knowledge that can connect “Coker”, “coach” and “Miami Hurricanes football team”. Therefore, the isolated reasoning chain leads to a wrong answer.

3.5 Summary

We proposed a novel semantic role labeling graph reasoning network (SRLGRN) to deal with multi-hop QA. SRLGRN has a graph convolutional network (GCN) as the backbone, which is created based on the semantic structure of the multiple documents. We innovatively construct a graph with entities and multiple relational edges from documents using semantic role labeling (SRL). This semantic structure of multiple documents can significantly improve the multi-hop reasoning capacity to find the line of reasoning to answer the questions. We jointly train a supporting fact prediction module that finds the cross-paragraph reasoning path, and an answer prediction module that obtains the final answer. SRLGRN exceeds most of the SOTA results on the HotpotQA benchmark. Moreover, we evaluate the model (excluding the paragraph selection module) on other reading comprehension benchmarks. Our approach achieves competitive performance on SQuAD v1.1 and v2.0.

CHAPTER 4

CAUSAL REASONING FOR DOCUMENT-LEVEL QA

4.1 Background and Motivation

Cause-effect QA is a specific type of question answering over a given document in which the questions ask about the causal impact of entities or events on each other. The recent research on reasoning over cause-effect QA has achieved promising results [87, 86, 40, 20, 106]. Specific to this problem, the WIQA benchmark [107] was proposed for the evaluation of causal reasoning capabilities of learning models on a procedural text by introducing "what . . . if" reasoning. The "what . . . if" reasoning task is a type of cause-effect QA that relates to reading comprehension, multi-hop reasoning, and commonsense reasoning. This task is rich in containing various challenging linguistic and semantic phenomena. The "what . . . if" reasoning is built based on linguistics and generating possible cause-effect relationships expressed in the context of a paragraph. Its goal is to predict what would happen if a process was perturbed in some way. It requires understanding and tracing the changes in events and entities through a paragraph. Figure 4.1 shows some examples of the WIQA benchmark. There are two types of questions in the dataset, including in-paragraph, where the answer to the question is in the procedure itself, and out-of-paragraph, where the answer does not exist in the text and needs external knowledge [107].

There are several challenges in the "what . . . if" cause-effect QA. The first challenge is reasoning over the comparative expressions for describing the effect of the entities on each other in the text that can convey a positive or negative effect (promoting or demoting each other). For example, comparative expressions such as (larger, smaller), (more, less), (higher, lower). This task requires extracting the important entities through the procedural text and understanding their influences. BERT is used as a strong baseline in [107] to predict answers by implicit representations. However, they ignore explicit comparative expressions between entities and the way they affect each other.

The second challenge is causal reasoning over relations between pairs of entities. Although recent pre-trained language models (LM) achieve promising performance on QA, there is still a

Procedural Text:
1. A frog lays eggs in the water.
2. Tadpoles develop inside of the eggs.
3. The eggs hatch.
4. The tadpoles eat and grow.
5. The tadpoles grow legs and form into frogs.
6. The frogs leave the water.

Questions and Answers:
1. Suppose tadpoles eat more food happens, how will it affect more frogs?
   (A) **More** (B) less (C) No effect
2. Suppose the weather is unusually bad happens, how will it affect the tadpoles will need more food?
   (A) More (B) **less** (C) No effect

Figure 4.1 WIQA task contains procedural paragraphs and a large collection of "what . . . if" questions. The bold font candidate answers are the gold answers.

gap between LM and human performance due to the lack of causal reasoning over entities [30]. For example, given the question "suppose more animals that hunt frogs happen, how will it affect more tadpoles lose", the LM has difficulty to consider the relation "hunt" between the entity pair ("animals", "frogs"). This recent research work [5] uses a Transformer model with regularization to produce consistent answers to multiple related questions. The model obtains a good result with augmented data following logical constraints. However, these constraints ignore the importance of causal reasoning, and can not capture the higher-order chain of causal reasoning.

The third challenge is the lexical variability in expressing the same concept, which makes entity alignment hard. For example, the same entities and events are referred to by different terms, like (insect, bee), (become, form). Entity alignment requires the alignment between question and paragraph entities, and the alignment between the entities appearing in the different paragraphs themselves. Unfortunately, all current works ignore the importance of entity alignment for tracing the entities and finding the relation between different entities in the question and paragraph.

Therefore, we propose a novel end-to-end Relational Gating Network (RGN) for causal reasoning over cause-effect QA. The RGN framework answers the "what . . . if" questions and solves challenges

of comparative expressions, causal reasoning, and entity alignment. RGN jointly learns to extract the key entities through our entity gating mechanism, finds the line of reasoning and relations between the key entities through the relation gating mechanism, and captures the entity alignment through contextual entity interaction. The main motivation of the two gating mechanisms is to learn the line of causal reasoning. Concretely, we build an *entity gating* module to extract and filter the key entities in the question and context and highlight the entities that are compared qualitatively. Furthermore, we design a *relation gating* module with an alignment of entities to capture the higher-order chain of causal reasoning based on pairwise relations. Moreover, we propose an efficient module, called *contextual interaction module*, to incorporate cross-information from Question and Content interactions during training in an efficient way to help entities alignments.

The contributions of this chapter are as follows: 1) We propose a Relational Gating Network (RGN) that captures the most important entities and relationships involved in comparative expressions and causal reasoning. 2) We propose a contextual interaction module to effectively and efficiently align the question and paragraph entities. 3) We evaluate the methods and analyze the results on the "what . . . if" question answering using the WIQA benchmark. We improve the recent state-of-the-art results and show the significance of the entity gating module and relation gating module on causal reasoning over text.

4.2 Relational Gating Network

Relational Gating Network (RGN) aims to establish an end-to-end architecture for reasoning over cause-effect QA. RGN model uses an entity gating module to extract and filter the critical entities in question and paragraph content. We enable a higher-order chain of causal reasoning based on the pairwise relationships between key entities through the relation gating mechanism. We propose a contextual interaction module to improve entity alignment in an efficient way. Figure 4.2 shows the proposed architecture. This section introduces our network and the training approach in detail.

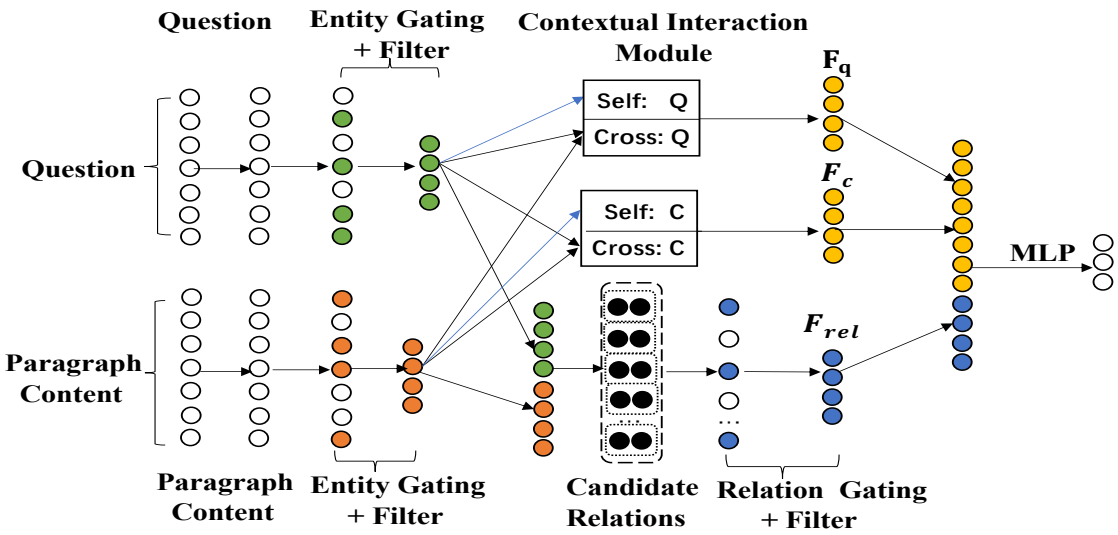

Figure 4.2 Relational Gating Network (RGN) is composed of pre-training contextual representation, entity gating module, relation gating module, and contextual interaction module followed by a task-specific classifier.

4.2.1 Problem Formulation

Formally, the task is to select one of the candidate's answers a, (A) More; (B) Less; (C) No effect, given a question q and the paragraph content C. The paragraph content includes several sentences $C = \{s_1, s_2, \ldots, s_n\}$. For each data sample, the data format is a triplet of (q, C, a).

4.2.2 Entity Representations

For each data sample, we form the input L by concatenating the question q and the paragraph content C as follows:

$$L = [[CLS]; q; [SEP]; C], \tag{4.1}$$

where [CLS] and [SEP] are the special tokens used in Language Models (LMs) [65]. We feed input L to a pre-trained LM to obtain all question and content token representations. Meanwhile, we use $E_{[CLS]}$ representation as the summary representation of the paragraph. After that, we obtain the RoBERTa token representations, $E_{[CLS]}$, E_q, and E_C, which are shown as follows:

$$E_q = [E_q^{w_1}, E_q^{w_2}, \ldots, E_q^{w_m}] \in \mathbb{R}^{m \times d}, \tag{4.2}$$

$$E_C = [E_C^{w_1}, E_C^{w_2}, \ldots, E_C^{w_n}] \in \mathbb{R}^{n \times d}, \tag{4.3}$$

where E_q represents the list of question representations, E_C represents the list of paragraph content representations, d is the learned representation dimension for tokens, m represents the max length of the question, and n represents the max length of the paragraph content.

4.2.3 Entity Gating

The intuition behind the entity gating module is to filter several key entity representations from both question, E_q, and paragraph content, E_C. We call this process entity gating which is shown in Figure 4.2. Given the question E_q, for each entity $E_q^{w_i} \in E_q$, we use a multi-layer perceptron and a softmax layer to obtain an entity importance score $U_q^{w_i}$:

$$U_q^{w_i} = \frac{\exp\left(MLP(E_q^{w_i})\right)}{\sum_{j=1}^{m} \exp\left(MLP(E_q^{w_j})\right)}, \tag{4.4}$$

$$E'_q = U_q E_q \in \mathbb{R}^{m \times d}. \tag{4.5}$$

We compute the new entity representations E'_q by multiplying the entity representations and their scores in U_q. Then we choose the most important entities with top-k scores. We denote the set of filtered key entities after gating the question as $V_q = [V_q^1, V_q^2, \ldots, V_q^k] \in \mathbb{R}^{k \times d}$. V_q is the list of question gated entity representations, k is the number of filtered entities and V_q^i is d-dimensional embedding for the i-th filtered entities.

Notice that the process of computing paragraph entity gating $V_C \in \mathbb{R}^{k \times d}$ is the same as the question entity gating V_q. Using the entity gating mechanism improves the generalizability of our deep model as we can explicitly see the selected entities and comparative expressions. The detailed analysis of entity gating is shown in Section 4.4.2.

4.2.4 Relation Gating

We consider the representations beyond entities by using a Relation Gating module. This extension allows RGN to capture the higher-order chain of causal reasoning based on pairwise relations, which

is the main contribution in this chapter. The pairs of entities enable the model to understand the relationships between words and find the line of causal reasoning. Moreover, relation gating aims to pair un-directed relations between entities for capturing the crucial relations, like "tadpole (losses) tail" and "less severe", as well as the pairs of entities that help to understand the line of reasoning. We call this process relation gating module, which is shown in Figure 4.2.

In this module, first, we concatenate V_q and V_C, which are obtained from Section 4.2.3 and form candidate set $V = \{V_q; V_C\}$. Then we pair every two gated entities and form $V_{rel}^{i,j} = [V^i; V^j] \in \mathbb{R}^{1\times 2d}$. Furthermore, the candidate relational representation, V_{rel}, is a non-linear mapping $\mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$ modeled by fully connected layers from candidate relation.

$$V_{rel} = [V_{rel}^1, V_{rel}^2, \ldots, V_{rel}^r] \in \mathbb{R}^{r\times 2d},$$

where r is the size of total relation candidate pairs, that is, $r = \frac{2k\times(2k-1)}{2}$. Given each candidate relation V_{rel}^i, we compute a multi-layer perceptron and a softmax layer to obtain a relational importance score, T^i:

$$T^i = \frac{\exp\left(MLP(V_{rel}^i)\right)}{\sum_{j=1}^{p}\exp\left(MLP(V_j^{rel})\right)}, \tag{4.6}$$

$$V'_{rel} = TV_{rel} \in \mathbb{R}^{k\times 2d}. \tag{4.7}$$

We compute the new relation representation V'_{rel} by multiplying the relation representations and their scores in T. We select the relations with top-k scores because using all the scores increases the number of parameters and the computational cost significantly. Moreover, the redundant entities make learning harder and consequently less accurate. We denote the set of filtered key relations after gating relations as $F_{rel} \in R^{k\times 2d}$ to the gated relation representation.

4.2.5 Contextual Interaction Module

Entity alignment is one of the challenges in Document-level QA. Although we propose entity and relation gating in the above sections separately, aligning questions with the paragraph is still

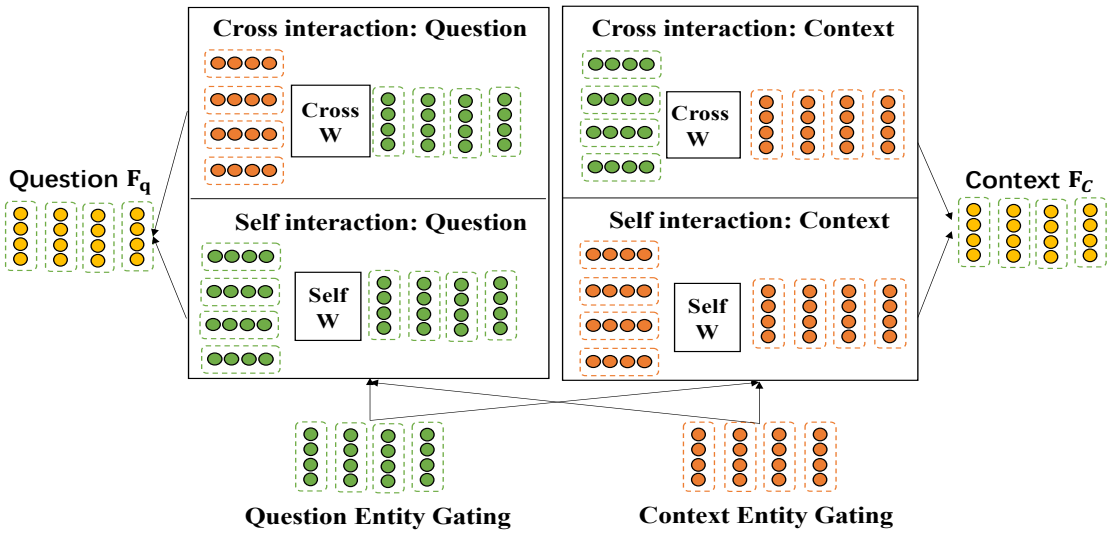

Figure 4.3 Contextual Interaction Module comprises self-interactions and cross-interactions. The inputs are the question's and paragraph's filtered entities representations, and the outputs are question and paragraph contextual representations.

important. We found that a simple concatenation of gated entity representations from the question V_q and the paragraph content V_C shows a good performance. However, concatenated representations and multi-layer perceptrons have a limited capacity for modeling the interactions.

As shown in Figure 4.3, we have developed a novel and fast encoding model, namely Contextual Interaction Module. The model needs to incorporate information from Question-Content interactions and, meanwhile, avoid expensive architectures such as Multi-Head attentions [111] because those are infeasible for large-scale datasets. Thus, we developed a model that uses only linear projections and inner products of both sides, i.e., question and context, and we apply a mechanism like simplified self-attention to model the interactions as described below.

Given the V_q, we compute the self-interaction of the question's gated entities, F_q^{self},

$$F_q^{self} = V_q^T W^{self} V_q \quad \in \mathbb{R}^{k \times d}, \tag{4.8}$$

where $W^{self} \in \mathbb{R}^{d \times k}$ is a projection matrix. The cross-interactions between gated entities and paragraph entities can be calculated as

$$F_q^{cross} = V_C^T W^{cross} V_q \quad \in \mathbb{R}^{k \times d}, \tag{4.9}$$

where $W^{cross} \in \mathbb{R}^{d \times k}$ is also a projection matrix. Then we concatenate the two matrices F_q^{self} and F_q^{cross}. Finally, we obtain the question contextual representation F_q as follows:

$$F_q = [F_q^{self}; F_q^{cross}] \in \mathbb{R}^{k \times 2d} \tag{4.10}$$

Notice that the process of paragraph contextual representation $F_C \in R^{k \times 2d}$ is the same as the question contextual representation F_q. Therefore, the output includes two representations. One is the paragraph contextual representation containing information from the question, and the question contextual representation contains information from the paragraph.

4.2.6 Output Prediction

After acquiring all the contextual entity representations and gated relations representations, we concatenate them and use the result as the final representation, F. The process is described as follows:

$$F = [F_q; F_c; F_{rel}] \in \mathbb{R}^{3k \times 2d} \tag{4.11}$$

Finally, a task-specific classifier MLP (F) predicts the output.

4.3 Experiments

Data		Train	Dev	Test V1	Test V2	Total
Questions		29808	6894	3993	3003	43698
Question type	in-para	7303	1655	935	530	10423
	out-of-para	12567	2941	1598	1218	18326
	no-effect	9936	2298	1460	1255	14949
	Total	29808	6894	3993	3003	43698
Number of hops	#hops=0	9936	2298	1460	1255	14949
	#hops=1	6754	1510	835	245	9254
	#hops=2	8969	2145	1153	1027	13294
	#hops=3	4149	941	545	476	6111
	Total	29808	6894	3993	3003	43698

Table 4.1 WIQA Dataset Statistics.

4.3.1 Dataset Description

WIQA [107] benchmark contains procedural paragraphs and a large collection of "what . . . if" questions. The task is to answer the questions given paragraph contents and a list of the candidate's answers. Table 4.1 shows the detailed data statistics and data distribution of the WIQA dataset.

4.3.2 Implementation Details

We implemented RGN using PyTorch. We used RoBERTa-Base Language Model as the backbone to train our model. All of the representations are 768−dimensions. For each data sample, we keep 128 tokens as the max length for the question, and 256 tokens as the max length for paragraph contents. Notice that both gated entity representations for question and paragraph use $k = 10$ for selecting top-k entities in our experiments. The value of this hyper-parameter was selected after experimenting with various values in $\{3, 5, 7, 10, 15, 20\}$ using the development dataset. For the Gated relation representations, top-10 ranked pairs are used to reduce the computational cost and reduce the unnecessary relations. In the relation gating process, we use two hidden layers for multi-layer perceptrons. The task-specific output classifier contains two MLP layers. The model is optimized using the Adam optimizer. The training batch size is 4. During training, we freeze the parameters of RoBERTa in the first two epochs, and we stop the training after no performance improvements are observed on the development dataset, which happens after 8 epochs.

4.4 Results and Discussion

4.4.1 Result Comparison

We show the model performance on the WIQA benchmark compared to various strong baselines in Table 4.2 and Table 4.3. We observe that, in general, Transformer-based models outperform other models, like Deomp-Attn [78]. This promising performance demonstrates the effectiveness of Transformers [111] and large-scale pre-trained Language Models [24, 65]. Moreover, our RGN achieves state-of-the-art results compared to all baseline models. Especially, RGN outperforms [107]

Models	in-para	out-of-para	no-effect	Test V1 Acc
Majority	45.46	49.47	55.0	30.66
Adaboost [24]	49.41	36.61	48.42	43.93
Decomp-Attn [78]	56.31	48.56	73.42	59.48
BERT (no para) [24]	60.32	43.74	84.18	62.41
BERT [107]	79.68	56.13	89.38	73.80
RoBERTa [107]	74.55	61.29	89.47	74.77
EIGEN [69]	73.58	64.04	90.84	76.92
REM-Net [44]	75.67	67.98	87.65	77.56
Logic-Guided [5]	-	-	-	78.50
RGN	**80.32**	**68.63**	**91.06**	**80.18**
Human	-	-	-	96.33

Table 4.2 Model Comparisons on WIQA test V1 dataset. WIQA test data has four categories, including in-paragraph accuracy, out-of-paragraph accuracy, no-effect accuracy, and overall test accuracy.

Models	in	out	no-eff	Test V2
Random	33.33	33.33	33.33	33.33
Majority	00.00	00.00	100.0	41.80
RoBERTa	70.69	60.20	91.11	75.34
REM-Net	70.94	63.22	91.24	76.29
REM-Net (RoBERTa-large)	76.23	69.13	92.35	80.09
QUARTET [85]	74.49	65.65	95.30	82.07
RGN (RoBERTa-base)	75.91	66.15	92.12	79.95
RGN (RoBERTa-large)	**78.40**	68.83	93.01	**82.46**
Human	-	-	-	96.30

Table 4.3 Model Comparisons on WIQA test V2. "In" represents in-paragraph accuracy, "out" represents out-of-paragraph accuracy, and "no-eff" represents no effect accuracy, .

by 6.38% and outperforms current state-of-the-art model on test V1, logic-guided [5], by around 1.6%. Moreover, our RGN model achieves the SOTA on WIQA test V2. The improved performance demonstrates that entity gating, relation gating, and contextual interaction module are effective for "what . . . if" causal reasoning. We provide a detailed analysis of the advantage of RGN from different perspectives.

Model	# hops = 1	# hops = 2	# hops = 3
BERT(no para)	58.1%	47.3%	42.8%
BERT	71.6 %	62.5%	59.5%
RoBERTa	73.5 %	63.9%	61.1%
EIGEN	78.78 %	63.49%	68.28 %
RGN	**80.5%**	**71.2%**	**70.0%**

Table 4.4 The accuracy when the number of hops increases.

4.4.2 Model Analysis

Effects on Causal Reasoning and Multi-Hops: In-para and out-of-para question categories require multiple hops of causal reasoning to answer the questions. As shown in Table 4.4, we found that the accuracy improved 7.0% for 1 hop, 7.3% for 2 hops, and 8.9% for 3 hops compared to RoBERTa which does not have the two gating mechanisms and Contextual Interaction Module. As we expect, the RGN framework has made tremendous progress in causal reasoning with multiple hops, and the improvement in the performance of the baselines is more when the number of hops increases. For qualitative analysis, we show successful cases from our RGN in Figure 4.4. We observe that RGN is capable of bridging question and paragraph content by extracting key entities. In the successful cases, which is shown in Figure 4.4, RGN helps in constructing the chain of "water droplets are in clouds → droplets combine to form bigger drops in the clouds" through key entities "water", "clouds", and "droplets". Moreover, we observe that the key entities "water", "clouds", and "droplets" obtain high gating entity scores.

Effects of Entity Gating: As shown in Table 4.5, in the first ablation study, we remove the entity gating and relation gating modules. Notice that the contextual interaction module uses the whole question entities and paragraph entities when RGN does not use these two modules. Using whole entities significantly increases the computational cost. Moreover, Table 4.5 shows that the accuracy is lower by about 5.3% compared to full RGN when applied to the development dataset. This experiment demonstrates that using all the entities without a gating mechanism has a negative influence on the contextual interaction module and drops the performance.

Effect of Relation Gating: The goal of relation gating is to capture the higher-order chain of causal

Ablations	in	out	no-eff	dev acc
RGN (w/o gating ent & rel)	76.2	61.1	89.2	75.3
RGN (w/o gating rel)	78.4	63.6	89.9	77.4
RGN	81.7	69.2	91.3	80.6
RGN (w/o CIM)	80.2	68.4	90.5	79.7
RGN (- CIM + Multi-Head)	81.3	68.9	91.7	80.3
RGN (add regularization)	82.0	69.1	91.6	80.8

Table 4.5 Ablation Study. CIM: Contextual Interaction Module.

reasoning based on pairwise relations. The relation gating module extracts the important candidate relations by pairing up entities after gating entities. More importantly, the relation gating module helps in understanding the connections between entities and finding the line of causal reasoning. Our model captures the important pairs of influencing entities "tadpole (losses) tail" and "animal (hunts) frog".

When we keep the entity gating module and remove the relation gating module, we observe that the accuracy of WIQA decreases 3.3% compared to the full RGN architecture. Moreover, the model without the relation gating module can not capture the key relations. The results show that the performance on the out-of-para questions decreases 5.6% compared to the full RGN model. Section 4.4.3 shows more examples and analysis.

Effects of Contextual Interaction Module (CIM): WIQA research work [107] shows that around 15% of the influence changes have difficulties handling the entity alignment part due to language variability. In other words, paragraph entities use different terms, such as ("removes", "expels") to express the same semantics. Especially, the problem of language variability becomes more severe for the multi-hop cases that require aligning the question with several sentences in the paragraph. Without the Contextual Interaction Module, the development accuracy decreases more than 1%. As shown in Table 4.4, the accuracy improves significantly in the direct effect (1 hop) and indirect effects (2 hops or 3 hops) compared to all strong baselines. This demonstrates the effectiveness of the interaction module. In an additional experiment, we replaced the CIM with the Multi-Head attention that uses an encoder of the Multi-Head attention composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first layer is multi-head self-attention, and the second is

Successful Cases	Question: suppose water is absorbed into the clouds and grow happens, how will it affect clouds are filled with rain droplets? Content: ['Water evaporates from the ground up to the sky', 'Water droplets are in clouds', 'Droplets combine to form bigger drops in the clouds', 'The drops get heavy', 'Gravity makes the drops fall.'] Gold Answer: More
	Question: suppose more activity of the heart happens, how will it affect less waste being removed from the body. Content: ['Blood is full of different waste', 'Blood travels through the body', 'The blood enters the kidneys', 'The kidneys filter the blood', 'The waste is seperated', 'The urine contains the waste', 'The urine is expelled from the body.'] Gold Answer: Less
Failing Cases	Question: suppose more fruit is produced happens, how will it affect MORE plants. Content: ['The seed germinates', 'The plant grows', 'The plant flowers', 'Produces fruit', 'The fruit releases seeds', 'The plant dies.'] Gold Answer: More
	Question: suppose the climate changes happens, how will it affect there are fewer clouds? Content: ['Water evaporates because of the sun', 'Water vapor rises into the air as it evaporates', 'Water vapor forms clouds as it mixes with dust and impurities', 'Clouds become heavier and larger over time', 'Clouds eventually become too heavy to stay in the sky', 'Some water vapor exits clouds as rain.'] Gold Answer: Less

Figure 4.4 Successful and failing cases of RGN network.

a fully connected network [111]. The computational time was 936 (ms/batch) for our contextual interaction module, while it is 3002 (ms/batch) for the Transformer while the accuracy is fairly similar.

4.4.3 Qualitative Analysis

For a better understanding of how our proposed model performs qualitatively, we show successful cases and failing cases from our RGN framework in Figure 4.4. We can observe that RGN is surprisingly capable of bridging the question and content in the in-para category.

Although the RGN framework has achieved state-of-the-art performance, the framework cannot always capture the line of causal reasoning. The bottom part of Figure 4.4 shows some failing cases. In the first failing case, RGN gives a wrong prediction because the content sentence "the plant dies" is captured as a strong negative influence by our model. Although our model bridges the relation between "fruit" and "plant", the critical term "dies" obtains a high gating score and misleads our final prediction.

Commonsense reasoning is the other type of error made by RGN model. There are two types of questions in the dataset, including in-paragraph where the answer to the question is in the text itself, and **out-of-paragraph**, where the answer does not exist in the text and the source of external knowledge is required [107]. For example, in the second failing case of Figure 4.4, the question

contains “climate change,” and the paragraph does not contain the cause of the “climate change”. This needs external knowledge between “climate change,” and “water evaporating”. Since answering the question requires external knowledge, it is hard to build a casual relationship for this example. However, the improvement in the out-of-paragraph is due to observing multiple examples in the dataset that use the same type of commonsense. Because relational gating helps to find the line of reasoning, our RGN model captures those from observing the relationships frequently and learns shortcuts. For example, in the second successful case of figure 4.4, the relational gating module captures the pairwise relation between “heart body” and “blood body” due to multiple occurrences in the data –filling the information gap for reasoning.

4.5 Summary

In this chapter, we propose an end-to-end Relational Gating Network (RGN) to help “what . . . if” causal reasoning over text for answering cause-effect questions. Particularly, we propose an entity gating module, relation gating module, and contextual interaction module to find the answer. We demonstrate that the proposed approach can effectively solve the challenges in the “what . . . if” reasoning, including causal reasoning, comparative expressions, and entity alignment. We evaluate our RGN on the WIQA benchmark and achieve state-of-the-art performance. Our gating mechanism and contextual interaction module can be easily used in solving various QA tasks that need to reason over entities and their relationships and follow a procedure. The gating mechanism can be extended to work at various levels of granularity, such as sentence and paragraph levels, to filter important pieces of information and to find the line of reasoning for answering the questions.

CHAPTER 5

RELATIONAL REASONING FOR CROSS-MODALITY QA

5.1 Background and Motivation

In many real-world situations, the answers to natural language questions can be found in different types of modalities. One important modality that can convey information is the visual one. The problem of answering natural language questions based on a given image is called visual question answering (VQA). VQA requires the understanding of visual contents, language semantics, cross-modality alignments, and relationships between two modalities [118, 4, 36, 100, 101]. Recently, there have been many efforts to build such multi-modal QA benchmarks [62, 54, 46, 4, 100, 36, 101]. Inspired by the effectiveness of deep learning [24], researchers develop deep architectures on multi-modal QA by learning representations for each modality, combining two representations, and predicting answers [59, 104]. For instance, VisualBERT [59] consists of Transformer layers that separately learn textual and visual representation with the self-attention module. LXMERT [104] learns entity representations by concatenating textual tokens and visual objects and using cross-modality Transformer architecture. However, the current performance of these models is unsatisfactory because the conventional deep learning models have difficulties in learning a robust joint representation and relational reasoning cross-modalities.

Our hypothesis is that exploiting the structure of the entities and their relationships in the two modalities and explicitly aligning them is one key factor that can facilitate solving the challenges of multi-modal QA, but this is less explored. In our proposed model, we learn robust joint representations by directly modeling the relations between different modality components based on the relevance scores inspired by the ideas from information retrieval litrature [77, 113, 24, 111].

Following the above-mentioned hypothesis, in this chapter, we propose a novel cross-modality relevance (CMR) architecture that considers the relevance between textual token representations and visual object representations for explicitly aligning them. We first encode data from each modality with single-modality Transformers and combine two encoding representations and pass it into a

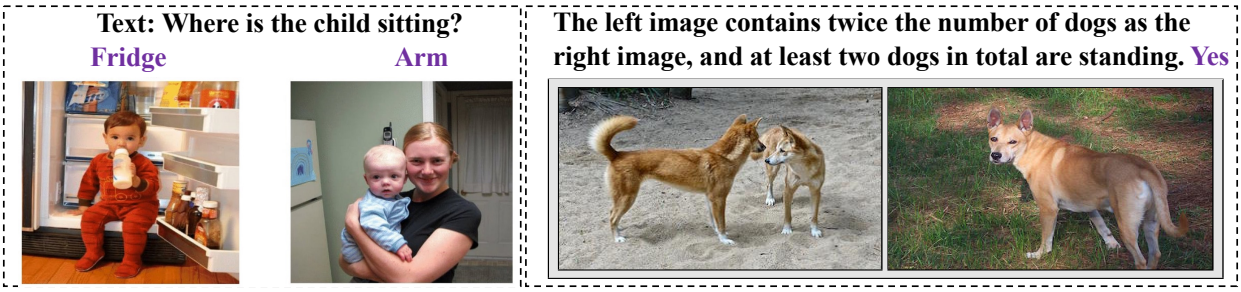

Figure 5.1 Two benchmark examples of the Cross-Modality Question Answering task. The left side is an example of the VQA benchmark, while the right side is an example of the NLVR benchmark.

cross-modality Transformer. We consistently refer to the words in text and objects in images(*i.e.* bounding boxes in images) as "entities" and their representations as "Entity Representations". We use the relevance between the components of the two modalities to model the alignment between them. We measure the relevance between their entities called "Entity Relevance", and high-order relevance between their relations called "Relational Relevance". We learn representations from the affinity matrix of the relevance scores by convolutional layers and fully-connected layers. Finally, we predict the answer based on the relevance representations.

The contributions of this chapter are as follows: **1)** We propose a cross-modality relevance (CMR) architecture that considers entity relevance and high-order relational relevance for aligning the two modalities. **2)** We evaluate the method and analyze the results on both VQA and NLVR tasks using VQA v2.0 and NLVR^2 benchmarks, respectively. We improve state-of-the-art on both tasks' published results. Our analysis shows the significance of exploiting relevance for relational reasoning for cross-modality QA.

5.2 Cross-Modality Relevance

Figure 5.2 shows our proposed Cross-Modality Relevance (CMR) architecture. As an end-to-end model, it encodes the relevance between the components of input modalities under task-specific supervision. We further add a high-order relevance between relations that occur in each modality. This architecture can help to solve tasks that need reasoning on two modalities based on their relevance. In this section, we first formulate the problem. Then we explain each component of the

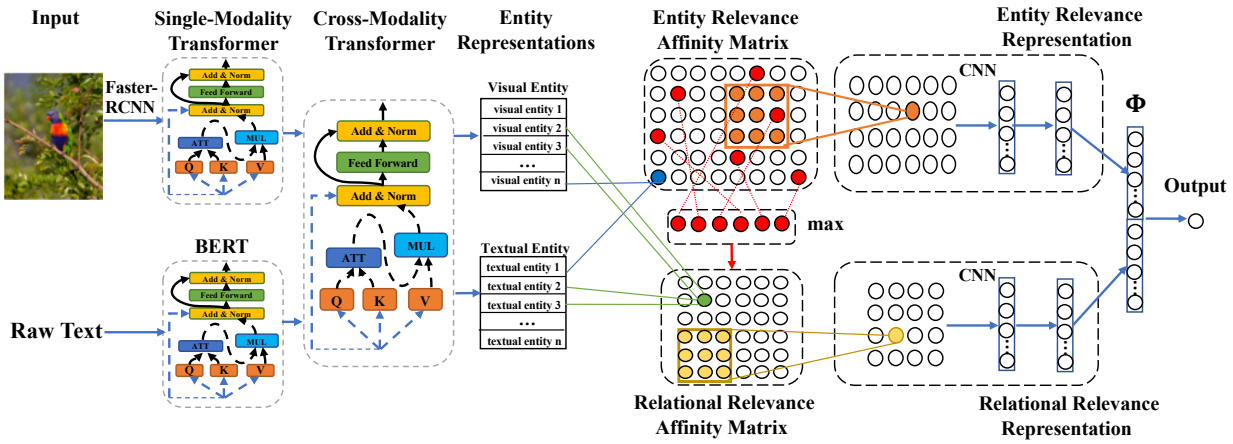

Figure 5.2 Cross-Modality Relevance model is composed of the single-modality transformer, cross-modality transformer, entity relevance, and high-order relational relevance, followed by a task-specific classifier.

CMR model, loss function, and training procedure of CMR in detail.

5.2.1 Problem Formulation

Formally, the problem is to model a mapping from a cross-modality data sample $\mathcal{D} = \{\mathcal{D}_\mu\}$ to an output y in a target task, where μ denotes the type of modality. And $\mathcal{D}_\mu = \{d_1^\mu, \cdots, d_{N^\mu}^\mu\}$ is a set of entities in the modality μ. In visual question answering, VQA, the task is to predict an answer given two modalities, that is, a textual question ($\mathcal{D}_t$) and a visual image ($\mathcal{D}_v$). In NLVR, given a textual statement ($\mathcal{D}_t$) and an image ($\mathcal{D}_v$), the task is to determine the correctness of the textual statement.

5.2.2 Representation Alignment

Single Modality Representations. For the textual modality $\mathcal{D}_t$, we utilize BERT [24] as shown in the bottom-left part of Figure 5.2, which is a multi-layer Transformer [111] with three different inputs: token embeddings [121], segment embeddings, and position embeddings. We refer to all the words as the entities in the textual modality and use the BERT representations for textual single-modality representations $\{s_1^t, \cdots, s_{N^t}^t\}$. We assume to have N^t words as textual entities.

For visual modality $\mathcal{D}_v$, as shown in the top-left part of Figure 5.2, Faster-RCNN [88] is used to generate regions of interest (ROIs), extract dense encoding representations of the ROIs, and

predict the class of each ROI. We refer to the ROIs on images as the visual entities $\left\{d_1^v, \cdots, d_{N^v}^v\right\}$. We consider a fixed number, N^v, of visual entities with the highest probabilities predicted by Faster-RCNN each time. The dense representation of each ROI is a local latent representation of a 2048-dimensional vector [88]. To enrich the visual entity representation with the visual context, we further project the vectors with feed-forward layers and encode them by a single-modality Transformer, as shown in the second column in Figure 5.2. The visual Transformer takes the dense representation, segment embedding, and bounding box positional embedding [104] as input and generates the single-modality representation $\left\{s_1^v, \cdots, s_{N^v}^v\right\}$. In case there are multiple images, for example, NLVR data (NLVR^2) has two images in each example, each image is encoded by the same procedure, and we keep N^v visual entities per image. We restrict all the single-modality representations to vectors of the same dimension d. However, these single-modality representations should be aligned.

Cross-Modality Alignment. To align the single-modality representations in a uniformed representation space, we introduce a cross-modality Transformer as shown in the third column of Figure 5.2. All the entities are treated uniformly in the cross-modality Transformer. Given the set of entity representations from all modalities, we define the matrix with all the elements in the set $S = \left[s_1^t, \cdots, s_{N^t}^t, s_1^v, \cdots, s_{N^v}^v\right] \in \mathbf{R}^{d \times (N^t + N^v)}$. Each cross-modality self-attention calculation is computed as follows [111][1],

$$\text{Attention}\,(K, Q, V) = \text{softmax}\left(\frac{K^\top Q}{\sqrt{d}}\right) V, \tag{5.1}$$

where in our case the key K, query Q, and value V, all are the same size of tensor S. A cross-modality Transformer layer consists of a cross-modality self-attention representation followed by residual connection with normalization from the input representation, a feed-forward layer, and another residual connection normalization. We stack several cross-modality Transformer layers to get a uniform representation over all modalities. We refer to the resulting uniformed representations

[1]Please note here we keep the usual notation of the attention mechanism for this equation. The notations might have been overloaded in other parts of this chapter.

as the entity representation and denote the set of the entity representations of all the entities as $\left\{s_1^{'t}, \cdots, s_{N^t}^{'v}, s_1^{'v}, \cdots, s_{N^v}^{'v}\right\}$. Although the representations are still organized by their original modalities per entity, they carry the information from the interactions with the other modality and are aligned in uniform representation space. The entity representations, as the fourth column in Figure 5.2, alleviate the gap between representations from different modalities, as we will show in the ablation studies, and allow them to be matched in the following steps.

5.2.3 Entity Relevance

Exploiting relevance, independent of the input representation, plays a critical role in reasoning ability, which is required in many tasks such as information retrieval, visual question answering, etc. To consider the entity relevance between two modalities $\mathcal{D}_\mu$ and $\mathcal{D}_\nu$, the entity relevance representation is calculated as shown in Figure 5.2. Given entity representation matrices $S^{'\mu} = \left[s_1^{'\mu}, \cdots, s_{N^\mu}^{'\mu}\right] \in \mathbf{R}^{d \times N^\mu}$ and $S^{'\nu} = \left[s_1^{'\nu}, \cdots, s_{N^\nu}^{'\nu}\right] \in \mathbf{R}^{d \times N^\nu}$, the relevance representation is calculated by

$$A^{\mu,\nu} = \left(S^{'\mu}\right)^\top S^{'\nu}, \tag{5.2a}$$

$$\mathrm{M}\left(\mathcal{D}_\mu, \mathcal{D}_\nu\right) = \mathrm{CNN}_{\mathcal{D}_\mu, \mathcal{D}_\nu}\left(A^{\mu,\nu}\right), \tag{5.2b}$$

where $A^{\mu,\nu}$ is the affinity matrix of the two modalities as shown in the right side of Figure 5.2. $A_{ij}^{\mu,\nu}$ is the relevance score of ith entity in $\mathcal{D}_\mu$ and jth entity in $\mathcal{D}_\nu$. $\mathrm{CNN}_{\mu,\nu}(\cdot)$ is a Convolutional Neural Network, corresponding to the sixth column of Figure 5.2, which contains several convolutional layers and fully connected layers. Each convolutional layer is followed by a max-pooling layer. Fully connected layers finally map the flattened feature maps to a d-dimensional vector. We refer to $\Phi_{\mathcal{D}_\mu, \mathcal{D}_\nu} = \mathrm{M}\left(\mathcal{D}_\mu, \mathcal{D}_\nu\right)$ as the entity relevance representation between μ and ν.

We compute the relevance between different modalities. For the modalities considered in this chapter, when there are multiple images in the visual modality, we calculate the relevance representation between them too. In particular, for the VQA benchmark, the above setting results in one entity relevance representation: a textual-visual entity relevance $\Phi_{\mathcal{D}_t, \mathcal{D}_v}$. For NLVR^2 benchmark, there are three entity relevance representations: two textual-visual entity relevance

$\Phi_{\mathcal{D}_t,\mathcal{D}_{v_1}}$ and $\Phi_{\mathcal{D}_t,\mathcal{D}_{v_2}}$, and a visual-visual entity relevance $\Phi_{\mathcal{D}_{v_1},\mathcal{D}_{v_2}}$ between two images. Entity relevance representations will be flattened and joined with other features in the next layer of the network.

5.2.4 Relational Relevance

We also consider the relevance beyond entities, that is, the relational relevance of the entities' relations. This extension allows our CMR to capture higher-order relational relevance between different modalities. We consider pair-wise relations between entities in each modality and calculate the relevance of the relations across modalities. The procedure is similar to entity relevance as shown in Figure 5.2. We denote the relational representation as a non-linear mapping $\mathbf{R}^{2d} \rightarrow \mathbf{R}^{d}$ modeled by fully-connected layers from the concatenation of representations of the entities in the relation:

$$r^{\mu}_{(i,j)} = \mathrm{MLP}_{\mu,1}\left(\left[s^{'\mu}_{i}, s^{'\mu}_{j}\right]\right) \in \mathbf{R}^{d}. \tag{5.3}$$

Relational relevance affinity matrix can be calculated by matching the relational representation, $\left\{r^{\mu}_{(i,j)}, \forall i \neq j\right\}$, from different modalities. However, there will be $C^{2}_{N_\mu}$ possible pairs in each modality $\mathcal{D}_\mu$, most of which are irrelevant. The relational relevance representations will be sparse because of the irrelevant pairs on both sides. Computing the relevance score of all possible pairs will introduce a large number of unnecessary parameters which makes the training more difficult.

We propose to rank the relation candidates (i.e. pairs) by the intra-modality relevance score and the inter-modality importance. Then we compare the top-K ranked relation candidates between two modalities as shown in Figure 5.3. For the intra-modality relevance score, shown in the bottom left part of the figure, we estimate a normalized score based on the relational representation by a softmax layer.

$$U^{\mu}_{(i,j)} = \frac{\exp\left(\mathrm{MLP}_{\mu,2}\left(r^{\mu}_{(i,j)}\right)\right)}{\sum_{k\neq l}\exp\left(\mathrm{MLP}_{\mu,2}\left(r^{\mu}_{(k,l)}\right)\right)}. \tag{5.4}$$

To evaluate the inter-modality importance of a relation candidate, which is a pair of entities in the same modality, we first compute the relevance of each entity in text with respect to the visual

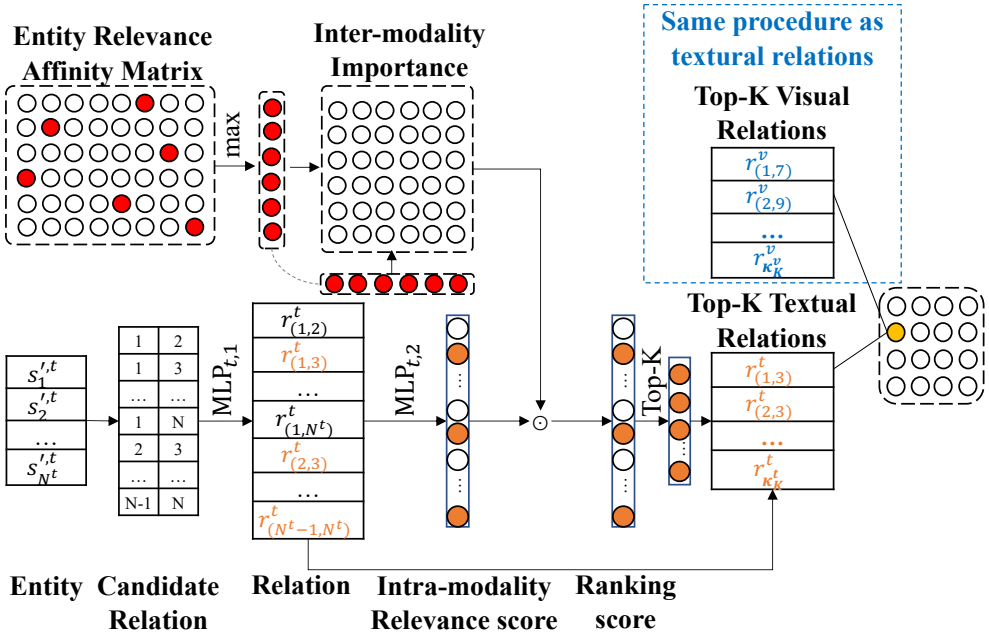

Figure 5.3 Relational Relevance is the relevance of top-K relations in terms of intra-modality relevance score and inter-modality importance.

objects. As shown in Figure 5.3, we project a vector that includes the most relevant visual object for each word, denoted this importance vector as v^t. This helps to focus on words that are grounded in the visual modality. We use the same procedure to compute the most relevant words to each visual object.

Then we calculate the relation candidates importance matrix V^μ by an outer product, $\otimes$, of the importance vectors as follows,

$$v_i^\mu = \max_j A_{ij}^{\mu,v}, \tag{5.5a}$$

$$V^\mu = v^\mu \otimes v^\mu, \tag{5.5b}$$

where v_i^μ is the ith scalar element in v^μ that corresponds to the ith entity, and $A^{\mu,v}$ is the affinity matrix calculated by Equation 5.2a.

Notice that the inter-modality importance V^μ is symmetric. The upper triangular part of V^μ, excluding the diagonal, indicates the importance of the corresponding elements with the same index in intra-modality relevance scores U^μ. The ranking score for the candidates is the combination (here the product) of the two scores $W_{(i,j)}^\mu = U_{(i,j)}^\mu \times V_{ij}^\mu$. We select the set of top-K ranked

candidate relations $\mathcal{K}_\mu = \{\kappa_1, \kappa_2, \cdots, \kappa_K\}$. We reorganize the representation of the top-K relations as $R^\mu = \left[r^\mu_{\kappa_1}, \cdots r^\mu_{\kappa_K}\right] \in \mathbf{R}^{d\times K}$. The relational relevance representation between $\mathcal{K}_\mu$ and $\mathcal{K}_\nu$ can be calculated similarly to the entity relevance representations as shown in Figure 5.2.

$$\mathrm{M}\left(\mathcal{K}_\mu, \mathcal{K}_\nu\right) = \mathrm{CNN}_{\mathcal{K}_\mu,\mathcal{K}_\nu}\left(\left(R^\mu\right)^\top R^\nu\right). \tag{5.6}$$

$\mathrm{M}\left(\mathcal{K}_\mu, \mathcal{K}_\nu\right)$ has its own parameters which results in a d-dimensional feature $\Phi_{\mathcal{K}_\mu,\mathcal{K}_\nu}$.

In particular, for the VQA task, the above setting results in one relational relevance representation: a textual-visual relevance $\mathrm{M}\left(\mathcal{K}_t, \mathcal{K}_v\right)$. For the NLVR task, there are three entity relevance representations: two textual-visual relational relevance $\mathrm{M}\left(\mathcal{K}_t, \mathcal{K}_{v_1}\right)$ and $\mathrm{M}\left(\mathcal{K}_t, \mathcal{K}_{v_2}\right)$, and a visual-visual relational relevance $\mathrm{M}\left(\mathcal{K}_{v_1}, \mathcal{K}_{v_2}\right)$ between two images. Relational relevance representations will be flattened and joined with other features in the next layers of the network.

After acquiring all the entity and relational relevance representations, namely $\Phi_{\mathcal{D}_\mu,\mathcal{D}_\nu}$ and $\Phi_{\mathcal{K}_\mu,\mathcal{K}_\nu}$, we concatenate them and use the result as the final feature $\Phi = \left[\Phi_{\mathcal{D}_\mu,\mathcal{D}_\nu}, \cdots, \Phi_{\mathcal{K}_\mu,\mathcal{K}_\nu}, \cdots\right]$. A task-specific classifier $\mathrm{MLP}_\Phi\left(\Phi\right)$ predicts the output of the target task as shown in the right-most column in Figure 5.2.

5.2.5 Training

In CMR architecture, we predict the output y from a specific task with the final feature Φ with a classification function. The gradient of the loss function is back-propagated to all the components in CMR to penalize the prediction and adjust the parameters. We freeze the parameters of BERT for textual modality and Faster-RCNN for visual modality. The parameters of the following parts will be updated by gradient descent: single modality Transformers, the cross-modality Transformers, $\mathrm{CNN}_{\mathcal{D}_\mu,\mathcal{D}_\nu}\left(\cdot\right)$, $\mathrm{CNN}_{\mathcal{K}_\mu,\mathcal{K}_\nu}\left(\cdot\right)$, $\mathrm{MLP}_{\mu,1}\left(\cdot\right)$, $\mathrm{MLP}_{\mu,2}\left(\cdot\right)$ for all modalities and modality pairs, and the task-specific classifier $\mathrm{MLP}_\Phi\left(\Phi\right)$.

The VQA task can be formulated as a multi-class classification that chooses a word to answer the question. We apply a softmax classifier on Φ and penalize it with the cross-entropy loss. For the NLVR^2 dataset, the task is the binary classification that determines whether the statement is correct

regarding the images. We apply a binary classification on Φ and penalize it with the cross-entropy loss.

5.3 Experiments

In this section, We introduce the datasets, experiment settings, and results compared to state-of-the-art published works.

5.3.1 Dataset Description

NLVR^2 [101] is a dataset that aims to joint reasoning about natural language descriptions and related images. Given a textual statement and a pair of images, the task is to indicate whether the statement correctly describes the two images. NLVR^2 contains $107,292$ examples of sentences paired with visual images and designed to emphasize semantic diversity, compositionality, and visual reasoning challenges.

VQA v2.0 [36] is an extended version of the VQA dataset. It contains $204,721$ images from the MS COCO [62], paired with $1,105,904$ free-form, open-ended natural language questions and answers. These questions are divided into four categories: Yes/No, Number, and Other.

5.3.2 Implementation Details

We implemented CMR using Pytorch. We consider the 768-dimension single-modality representations. For textural modality, the pre-trained BERT base model [24] is used to generate the single-modality representation. For visual modality, we use Faster-RCNN pre-trained by BUTD [3], followed by a five-layers Transformer. Parameters in BERT and Faster-RCNN are fixed. For each example, we keep 20 words as textual entities and 36 ROIs per image as visual entities. For relational relevance, top-10 ranked pairs are used. For each relevance-CNN, $\text{CNN}_{\mathcal{D}_\mu,\mathcal{D}_\nu}(\cdot)$ and $\text{CNN}_{\mathcal{K}_\mu,\mathcal{K}_\nu}(\cdot)$, we use two convolutional layers, each of which is followed by a max-pooling, and fully connected layers. For the relational representations and their intra-modality relevance

score, $\text{MLP}_{\mu,1}(\cdot)$ and $\text{MLP}_{\mu,2}(\cdot)$, we use one hidden layer for each. The task-specific classifier $\text{MLP}_{\Phi}(\Phi)$ contains three hidden layers. The model is optimized using the Adam optimizer with $\alpha = 10^{-4}, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-6}$. The model is trained with a weight decay of 0.01, a max gradient normalization clip of 1.0, and a batch size of 32.

5.3.3 Baseline Description

We briefly describe the recent four SOTA baselines. The first two baselines use non-Transformer neural models as the backbone, while the other two baselines use Transformer-based architectures. We describe these baselines as follows.

Compositional Attention Network (MAC) [45] is a fully differentiable neural network that aims to facilitate machine reasoning. The model designs explicit and structured reasoning by a new recurrent Memory, Attention, and Composition cell.

Feature-wise Linear Modulation (FiLM) [80] is a strong baseline on visual reasoning tasks. In the FiLM model, each layer influences neural network computation via a feature-wise affine transformation based on conditioning information.

VisualBERT [59] is an End-to-End model for language and vision tasks, consisting of Transformer layers that align textual and visual representation with self-attention. VisualBERT and CMR have a similar cross-modality alignment approach. However, VisualBERT only uses the Transformer representations, while CMR uses the relevance representations.

LXMERT [104] aims to learn cross-modality encoder representations from Transformers. It pre-trains the model with a set of tasks and fine-tunes another set of specific tasks. LXMERT is the currently published state-of-the-art on both NLVR^2 and VQA v2.0.

Models	Dev%	Test%
N2NMN	51.0	51.1
MAC-Network	50.8	51.4
FiLM	51.0	52.1
CNN+RNN	53.4	52.4
VisualBERT	67.4	67.0
LXMERT	74.9	74.5
CMR	**75.4**	**75.3**

Table 5.1 Accuracy on NLVR^2.

Model	Dev%	Test Standard%			
	Overall	Y/N	Num	Other	Overall
BUTD	65.32	81.82	44.21	56.05	65.67
ReGAT	70.27	86.08	54.42	60.33	70.58
ViLBERT	70.55	-	-	-	70.92
VisualBERT	70.80	-	-	-	71.00
BAN	71.4	87.22	54.37	62.45	71.84
VL-BERT	71.79	87.94	54.75	62.54	72.22
LXMERT	72.5	87.97	**54.94**	63.13	72.54
CMR	72.58	**88.14**	54.71	**63.16**	**72.60**

Table 5.2 Accuracy on VQA v2.0.

5.4 Results and Discussion

5.4.1 Result Comparison

NLVR^2 The results of NLVR task are listed in Table 5.1. Transformer based models (VisualBERT, LXMERT, and CMR) outperform other models (N2NMN [42], MAC [45], and FiLM [80]) by a large margin. This is due to the strong pre-trained single-modality representations and the Transformers' ability to learn the representations. Furthermore, CMR shows the best performance compared to all Transformer-based baseline methods and achieves state-of-the-art. VisualBERT and CMR have similar cross-modality alignment approaches. CMR outperforms VisualBERT by 12.4%. The gain mainly comes from entity relevance and relational relevance that model the relations.

Textural	Visual	Cross	Dev%	Test%
12	3	3	74.1	74.4
12	4	4	74.9	74.7
12	5	5	75.4	**75.3**
12	6	6	75.5	75.1

Table 5.3 Accuracy on NLVR^2 of CMR with various Transformer sizes. The numbers in the left part of the table indicate the number of self-attention layers.

Models	Dev%	Test%
CMR	**75.4**	**75.3**
without Single-Modality Transformer	68.2	68.5
without Cross-Modality Transformer	59.7	59.1
without Entity Relevance	70.6	71.2
without Relational Relevance	73.0	73.4

Table 5.4 Test accuracy of different variations of CMR on NLVR^2.

VQA v2.0: In Table 5.2, we show the comparison with published models, excluding the ensemble ones. Most competitive models are based on Transformers (ViLBERT [66], VisualBERT [59], VL-BERT [99], LXMERT [104], and CMR). BUTD [3, 108], ReGAT [58], and BAN [49] also employ an attention mechanism for a relation-aware model. The proposed CMR achieves the best test accuracy on Y/N questions and Other questions. However, CMR does not achieve the best performance on *Number* questions. This is because Number questions require the ability to count numbers in one modality, while CMR focuses on modeling relations between modalities. Performance on counting might be improved by explicit modeling of quantity representations. CMR also achieves the best overall accuracy. In particular, we can see a 2.3% improvement over VisualBERT [59], as in the above-mentioned NLVR^2 results. This shows the significance of the entity and relational relevance.

5.4.2 Model Analysis

To better understand the influence of each part in CMR, we perform the ablation study. Table 7.3 shows the performances of four variations on NLVR^2.

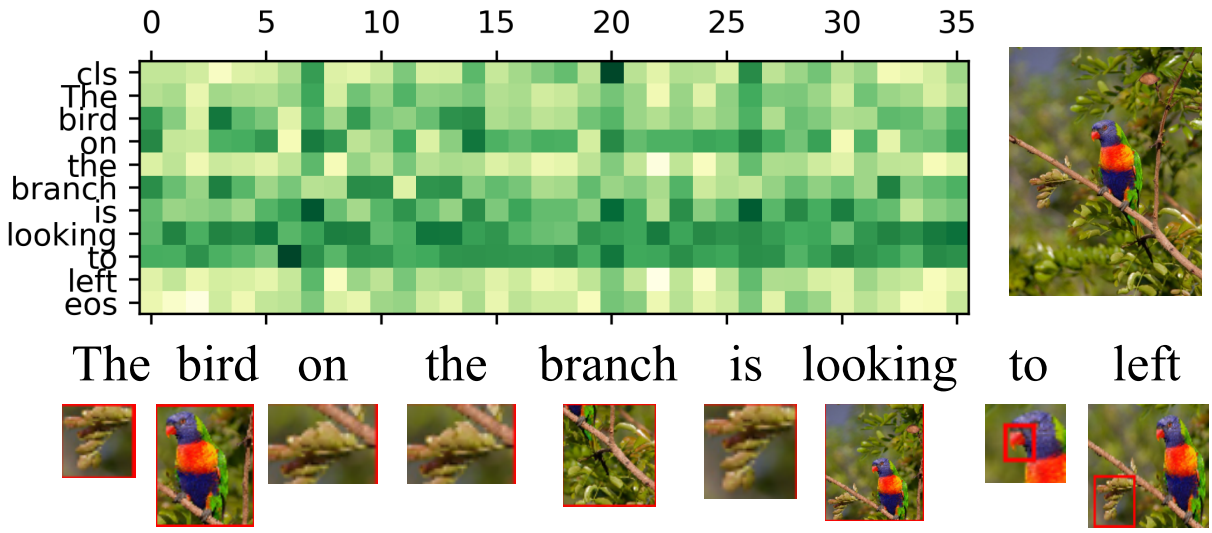

Figure 5.4 The entity affinity matrix between textual (rows) and visual (columns) modalities. The darker color indicates a higher relevance score. The ROIs with a maximum relevance score for each word are shown paired with the words.

Effect of Single Modality Transformer. In the first ablation study, we remove both textual and visual single-modality Transformers and map the raw input with a linear transformation to d-dimensional space instead. Notice that the raw input of textual modality is the WordPieces [121] embeddings, segment embeddings, and the position embeddings of each word, while that of visual modality is the 2048-dimension dense representation of each ROI extracted by Faster-RCNN. It turns out that removing single-modality Transformers decreases the accuracy by 9.0%. Single modality Transformers play a critical role in producing a strong contextualized representation for each modality.

Effect of Cross-Modality Transformer. We remove the cross-modality Transformer and use single-modality representations as entity representations. As shown in Table 7.3, the model degenerates dramatically, and the accuracy decreases by 16.2%. The huge accuracy gap demonstrates the unparalleled contribution of the cross-modality Transformer to aligning representations from input modalities.

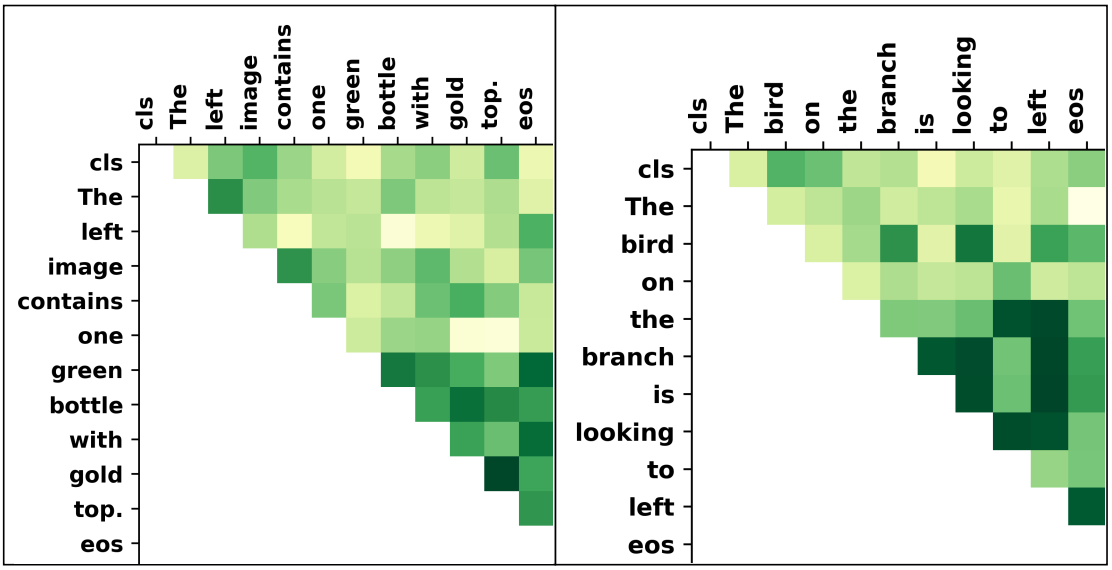

Figure 5.5 The relation ranking score of two example sentences. The darker color indicates a higher ranking score.

Effect of Entity Relevance. We remove the entity relevance representation $\Phi_{\mathcal{D}_\mu,\mathcal{D}_\nu}$ from the final feature Φ. As shown in Table 7.3, the test accuracy is reduced by 5.4%. This is a significant difference in performance among Transformer based models [59, 66, 104]. To highlight the significance of entity relevance, we visualize an example affinity matrix in Figure 5.4. The two major entities, "bird" and "branch", are matched perfectly. More interestingly, the three ROIs which are matching the phrase "looking to left" capture an indicator (the beak), a direction (left), and the semantics of the whole phrase.

Effect of Relational Relevance. We remove the entity relevance representation $\Phi_{\mathcal{K}_\mu,\mathcal{K}_\nu}$ from the final feature Φ. A 2.5% decrease in test accuracy is observed in Table 7.3. We argue that CMR models high-order relations, which are not captured in entity relevance, by modeling relational relevance. We present two examples of textual relation ranking scores in Figure 5.5. The learned ranking score highlights the important pairs, for example "gold - top", and "looking - left", which describe the important relations in textual modality.

5.4.3 Qualitative Analysis

To investigate the influence of model sizes, we empirically evaluated CMR on NLVR^2 with various sets of Transformers sizes which contain the most parameters of the model. All other details are kept the same as descriptions in Section 5.3.2. Textual Transformer remains 12 layers because it is the pre-trained BERT. Our model contains $285M$ parameters. Among these parameters, around $230M$ parameters belong to pre-trained BERT and Transformer. Table 5.3 shows the results. As we increase the number of layers in the visual Transformer and the cross-modality Transformer, it tends to improve accuracy. However, the performance becomes stable when there are more than five layers. We choose five layers of visual Transformer and cross-modality Transformer in other experiments.

5.5 Summary

In this chapter, we propose a novel cross-modality relevance (CMR) for language and vision reasoning. Particularly, we claim the significance of relevance between the components of the two modalities of reasoning, which include entity relevance and relational relevance. We propose an end-to-end Cross-Modality Relevance (CMR) architecture that is tailored for language and vision reasoning. We evaluate the proposed CMR on NLVR and VQA tasks. Our approach exceeds the state-of-the-art on NLVR^2 and VQA v2.0 datasets. The experiments and the empirical analysis demonstrate CMR's capability of modeling relational relevance. This result indicates the significance of exploiting relevance. Our proposed architectural component for exploiting relevance can be used independently from the full CMR architecture and is potentially applicable for other multi-modality tasks.

CHAPTER 6

COMMONSENSE REASONING FOR KNOWLEDGE BASED QA

6.1 Background and Motivation

Large-scale pre-trained language models (LMs) are shown to cover large amounts of world-knowledge and common sense and have achieved success in many QA benchmarks [87, 86, 74, 125]. However, the current research shows LMs have difficulties in answering questions merely based on their implicit knowledge [127, 30].

Therefore, using the external sources of knowledge explicitly in the form of knowledge graphs (KGs) is a recent trend in Question Answering [61, 30]. Figure 6.1, taken from the CommonsenseQA benchmark, shows an example for which answering the question requires commonsense reasoning. In this example, the external KG provides the required background information to obtain the reasoning chain from question to answer. We highlight two challenges in this type of QA task: (a) the extracted KG subgraph sometimes misses some edges between entities, which breaks the chain of reasoning (b) the semantic context of the question and connection to the answer is not used properly, for example, reasoning when negative terms exist in the question, such as no and not, is problematic.

The challenge (a) is caused by the following reasons. First, the knowledge graph is originally imperfect and does not include the required edges. Second, since the size of knowledge graphs is tremendously large, many models use a subgraph of KG for each example [61, 31, 127]. However, to reduce the size of the graph, most of the models select the entities that appear in two-hop paths [30]. Consequently, some intermediate concept (entity) nodes and edges are missed in the extracted KG subgraph. In such cases, the subgraph does not contain a complete chain of reasoning. Third, the current models often cannot reason over paths when there is no direct connection between the involved concepts. While finding the chain of reasoning in QA is challenging in general [135], this problem is more critical when the KG is the only source of knowledge, and there are missing edges. Looking back at Figure 6.1, the KG subgraph misses the direct connection between *guitar*

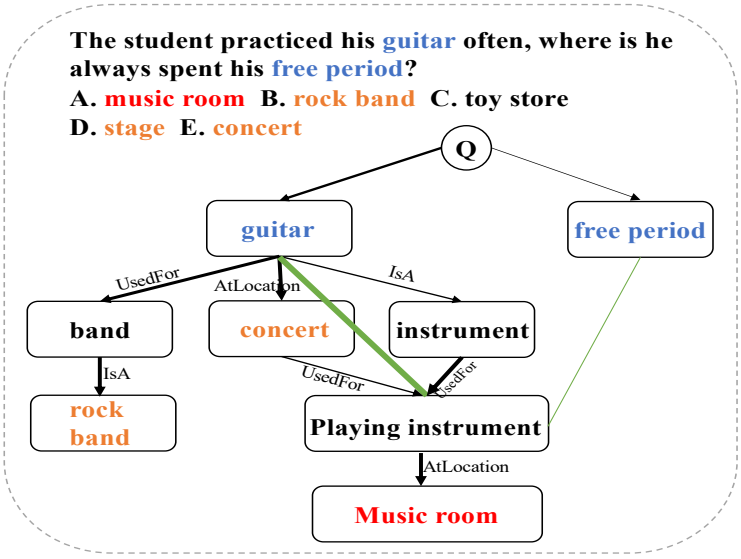

Figure 6.1 An example of the CommonsenseQA benchmark. Given the question node Q, question entity nodes (blue boxes), correct answer entity node (red box), and wrong answer entity nodes (orange boxes), we predict the answer by reasoning over the question and the extracted KG subgraph.

and *playing instrument* (green arrow). For challenge (b) about considering question semantics, as KagNET [61] points out, previous models are not sensitive to the negation words and consequently predict opposite answers. QA-GNN [127] model is the first work to deal with the negative questions. QA-GNN improves the reasoning under negation, to some extent, by adding the QA global node to the graph. However, the challenge still exists.

To solve the above challenges, we propose a novel architecture, called Dynamic Relevance Graph Network (DRGN). The motivation of our proposed DRGN is to recover the missing edges and establish direct connections between relevant concepts to facilitate multi-hop reasoning. In particular, the DRGN model uses a relational graph network module while considering the importance of the neighbor nodes using an additional relevance matrix. It can potentially recover the missing edges by establishing direct connections based on the relevancy of the node representations in the KG during the training. The module can potentially capture the connections between distant nodes while benefiting from the existing KG edges. Our proposed model learns representations directly based on the relevance scores between subgraph entity pairs that are computed by the Inner Product operation. At each convolutional layer of the graph neural network, we compute the inner product of the nodes

based on their current layer's node representations dynamically and build the neighborhoods based on this relevance measure and form a relevance matrix accordingly. This can be seen as a way to learn new edges as the training goes forward in each layer while influencing the weights of the neighbors dynamically based on their relevance. As shown in Figure 6.1, the relevance score between *guitar* and *playing instrument* is stronger than other nodes in the subgraph. Moreover, since the graph includes the question node, the relevance between the question node and entity nodes is computed at every layer, making use of the contextual information more effectively. It becomes more evident that the student will spend the *free period* in the *music room* rather than the *concert.*

In summary, the contributions of this work are as follows: (1) The Proposed DRGN architecture exploits the existing edges in the KG subgraph while explicitly using the relevance between the nodes to establish direct connections and recover the possibly missing edges dynamically. This technique helps in capturing the reasoning path in the KG for answering the question. (2) Our model exploits the relevance between the question and the graph entities, which helps consider the semantics of the question explicitly in the graph reasoning and boost the performance. In particular, it improves dealing with the negation. (3) Our proposed model obtains competitive results on both CommonsenseQA and OpenbookQA benchmarks. Our analysis demonstrates the significance and effectiveness of the DRGN model.

6.2 Dynamic Relevance Graph Network

In this section, we first define the problem, formally. Then we explain each component of our proposed model, loss function, and training procedure in detail.

6.2.1 Problem Formulation

The task of QA over pure knowledge is to choose a correct answer a_{ans} from a set of N candidate answers $\{a_1, a_2, ..., a_n\}$ given input question q and an external knowledge graph (KG). Since the knowledge graphs are often huge, as a part of the solution, we only consider a subgraph of KG as an input for each example. A subgraph is selected for each example based on a previously proposed

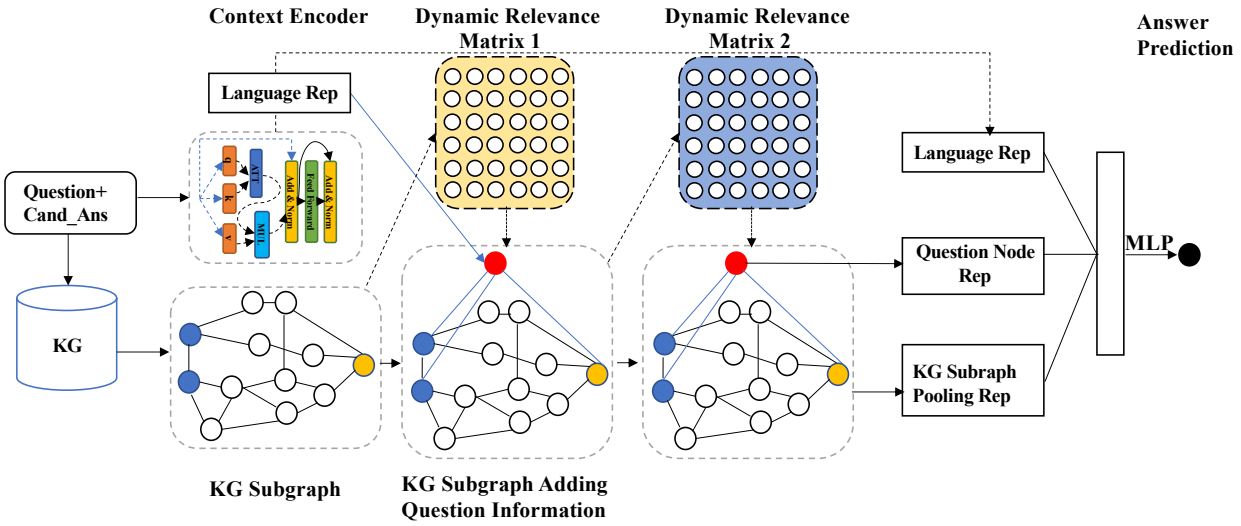

Figure 6.2 Our proposed DRGN model is composed of the Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. The blue color entity nodes represent the entity mentioned in the question. The yellow color node represents the answer node. The red color node is the question node. We use different colors to draw the dynamic relevance matrix 1 and 2 because the relevance matrix changes dynamically in each graph neural layer.

approach [31]. The approach is to construct a subgraph from KG that contains the entities mentioned in the question and answer choices.

6.2.2 Model Description

Figure 7.2 shows the proposed Dynamic Relevance Graph Network (DRGN) architecture. Our DRGN includes four modules: Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. In this section, we describe the details of our approach and the way we train our model efficiently.

6.2.3 Language Context Encoder

For the given question q and each candidate answer a_i, we concatenate them to form the Language Context L:

$$L = [[CLS]; q; [SEP]; a_i], \tag{6.1}$$

where [CLS] and [SEP] are the special tokens used by large-scale pre-trained Language Models (LMs). We feed input L to a pre-trained LMs encoder to obtain token represetnations, denoted as $h_L \in \mathbb{R}^{|L|d}$, where $|L|$ represents the length of the sequence. Then we use the [CLS] representation, denoted as $h_{[CLS]} \in \mathbb{R}^d$, as the representation of L.

6.2.4 KG Subgraph Construction

We use ConceptNet [97], a general-domain knowledge graph, as the commonsense KG. ConceptNet graph has multiple semantic relational edges, e.g., HasProperty, IsA, AtLocation, etc. We follow MHGRN [30] research work to construct the subgraphs from KG for each example. The subgraph entities are selected with the exact match between n-gram tokens and ConceptNet concepts using some normalization rules. Then another set of entities is added to the subgraph by following the KG paths of two hops of reasoning based on the current entities in the subgraph.

Furthermore, we add the semantic context of the question as a separate node to the subgraph. This node provides an additional question context to the KG subgraph, G_{sub}, as suggested by QAGNN [127]. We link the question node to entity nodes mentioned in the question. The semantic context of the question node Q is initialized by the [CLS] representation described in Section 6.2.3. The initial representation of the other entities is derived from applying RoBERTa and pooling over their contained tokens [30].

6.2.5 Graph Neural Network Module

The basis of our learning representation is Multi-relational Graph Convolutional Network (R-GCN) [90]. R-GCN is an extension of GCN that operates on a graph with multi-relational edges between nodes. In our case, the relation types between entities are taken from the 17 semantic relations from ConceptNet. Meanwhile, an additional type is added to represent the relationship between the question node and question entities, making the graph structure different from previous works. We denote the set of relations as R.

Our dynamic relevance graph network (DRGN) architecture is a variation of the R-GCN model. To establish the direct connection between the graph nodes and re-scale the importance of the neighbors, we compute the relevance score between the nodes dynamically at each graph layer based on their current learned representations. Then we build the neighborhoods based on this relevance measure and form a relevance matrix, M_{rel}, accordingly. This can be seen as a way to learn new edges based on the relevance of the nodes as the training goes forward in each graph layer. We use the inner product to compute the relevance matrix:

$$M_{rel}^{(l)} = h^{(l)\top} h^{(l)} \in \mathbb{R}^{(|V|+1)(|V|+1)}, \tag{6.2}$$

where $|V|$ is the graph entity node sizes, and 1 is added due to using the question node in the graph. The relevance matrix re-scales the weights and influences the way the neighborhood nodes' representations are aggregated in the R-GCN model. M_{rel} is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. In our proposed relational graph, the forward-pass message passing updates of the nodes, denoted by h_i, is calculated as follows:

$$h_i^{(l+1)} = \sigma\Big(\sum_{r\in R}\sum_{j\in\mathbb{N}_i^r} \frac{1}{d_{i,r}} W_r^{(l)} \cdot (M_{rel_{i,j}}^{(l)} h_j^{(l)}) + W_0^{(l)} \cdot (M_{rel_{i,i}}^{(l)} h_i^{(l)})\Big) \in \mathbb{R}^d, \tag{6.3}$$

where $\mathbb{N}_i^r$ represents the neighbor nodes of node i under relation r, $r \in R$. σ is the activation function, W_r denotes the learnable parameters. Besides, we calculate the updated question node representation as follows,

$$h_Q^{(l+1)} = \sigma\Big(\sum_{j\in\mathbb{N}^Q} W_Q^{(l)} \cdot F_c([h_Q^{(l)}; (M_{rel_{Q,j}}^{(l)} h_j^{(l)})]) + W_0^{(l)} \cdot (M_{rel_{Q,Q}}^{(l)} h_Q^{(l)})\Big) \in \mathbb{R}^d, \tag{6.4}$$

where F_c is a two-layer MLP, h_Q is the question node representation. Finally, we stack the node representations to form $h'^{(l+1)}$:

$$h'^{(l+1)} = [h_0^{(l+1)}; h_1^{(l+1)}; \cdots; h_{|V|}^{(l+1)}; h_Q^{(l+1)}] \in \mathbb{R}^{(|V|+1)d}. \tag{6.5}$$

We then compute the $(l+1)$ layer's dynamic relevance matrix $M_{rel}^{(l+1)}$ that shows the relevance scores of node representations. Finally, we use the $M_{rel}^{(l+1)}$ to multiply the node representation matrix $h'^{(l+1)}$

that helps the node representation to learn the weights of the edges based on the learned relevance and specifically to include the additional relevance edges between the nodes during the massage passing as follows:

$$h^{(l+1)} = \sigma\left(M_{rel}^{(l+1)} \cdot h'^{(l+1)} \cdot W_g\right) \in \mathbb{R}^{(|V|+1)d}, \tag{6.6}$$

where W_g is the learnable parameters.

6.2.6 Answer Prediction

Given the Language Context L and KG subgraph, we use the information from both the language representation $h_{[CLS]}$, question node representation h_Q learned from the KG subgraph, and the KG subgraph representation pooled from the last graph layer, $pool(h_{G_{sub}})$, to calculate the scores of the candidate answers as follows:

$$p(a|L, G_{sub}) = f_{out}([h_{[CLS]}; h_Q; pool(h_{G_{sub}})]), \tag{6.7}$$

where f_{out} is a two-layer MLP. Finally, we choose the highest-scored answer from N candidate answers as the prediction output. We use the cross entropy loss to optimize the end-to-end model.

6.3 Experiments

6.3.1 Dataset Description

We evaluate our model on two different QA benchmarks, CommonsenseQA [103] and OpenbookQA [73]. Both benchmarks come with an external knowledge graph. We apply ConceptNet to the external knowledge graph on these two benchmarks.

CommonsenseQA is a QA dataset that requires human commonsense reasoning capacity to answer the questions. Each question in CommonsenseQA has five candidate answers without any extra information. The dataset consists of 12, 102 questions.

OpenbookQA It is a multiple-choice QA dataset that requires reasoning with commonsense knowledge. The OpenbookQA benchmark is a well-defined subset of science QA [17] that requires

finding the chain of commonsense reasoning to answer a question. Each data sample includes the question, scientific facts, and candidate answers. In our experimental setting, scientific facts are added to the question part. This makes the problem formulation consistent with the CommonsenseQA setting.

6.3.2 Implementation Details

We implemented our DRGN architecture using PyTorch. We use the pre-trained RoBERTa-large [65] to encode the question. We use cross-entropy loss and RAdam optimizer [63] to train our end-to-end architecture. The batch size is set to 16, and the maximum text input sequence length is set to 128. Our model uses an early stopping strategy during the training. We use a 3-layer graph neural module in our experiments. Section 6.4.2 describes the effect of the different number of layers. The learning rate for the LMs is $1e-5$, while the learning rate for the graph module is $1e-3$.

6.3.3 Baseline Description

We select three SOTA models as our main baselines. One model is KagNET [61] that finds the line of reasoning without using a graph neural module. We use two more models, MHGRN [30] and QAGNN [127] that use graph neural module as the backbone to find the line of reasoning over knowledge graph.

KagNET [61] is a path-based model that models the multi-hop relations by extracting relational paths from Knowledge Graph and then encoding paths with an LSTM sequence model.

MHGRN [30]: Multi-hop Graph Relation Network (MHGRN) is a strong baseline. MHGRN model applies LMs to the question and answer context encoder, uses the GNN encoder to learn graph representations, and chooses the candidate answers by these two encoders.

QA-GNN [127] is the recent SOTA model that uses a working graph to train language and KG subgraph. The model jointly reasons over the question and KG and jointly updates the representations. QA-GNN uses GAT as the backbone to do message passing on the graph. To learn the semantic edge information, QA-GNN directly adds the edge representation to the local node representation

Models	Dev ACC%	Test ACC%
RoBERTa-no KG	69.6%	67.8%
R-GCN	72.6%	68.4%
GconAttn	72.6%	68.5%
KagNet	73.3%	69.2%
RN	73.6%	69.5%
MHGRN	74.4%	71.1%
QA-GNN	76.5%	73.4%
DRGN	**78.2%**	**74.0%**

Table 6.1 Dev accuracy and Test accuracy (In-House split) of various models on the CommonsenseQA benchmark, following by [61].

and cannot learn the global structure of the edges, which is inefficient. However, our model uses the global multi-relational adjacency matrices to learn the edge information.

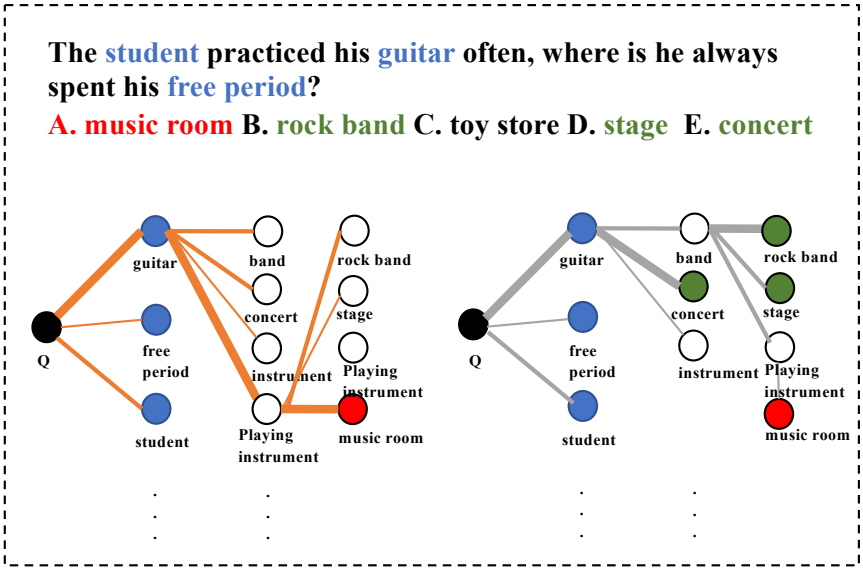

Figure 6.3 The complete reasoning chain from the question node to the candidate answer node. The blue nodes are question entity nodes, and the red and green nodes are the candidate answer nodes. The thicker edges indicate a higher relevance score to the neighborhood node, while the thinner edges indicate a lower score. The left side is the reasoning chain selected from our model (orange edges), while the right side is selected from the baseline models (grey edges).

Models	Dev	Test
RoBERTa-large	66.7%	64.8%
R-GCN	65.0%	62.4%
GconAttn	64.5%	61.9%
RN	66.8%	65.2%
MHGRN	68.1 %	66.8%
QA-GNN	68.9 %	67.8%
DRGN	**70.1%**	**69.6%**
AristoRoBERTaV7	79.2%	77.8%
T5(3 Billion Parameters)	-	83.2%
UnifiedQA(11 Billion Parameters)	-	**87.2%**
AristoRoBERTaV7+MHGRN	78.6%	80.6%
AristoRoBERTaV7+QA-GNN	80.4%	82.8%
AristoRoBERTaV7+DRGN	**81.8%**	**84.1%**

Table 6.2 Development and Test accuracy of various model performances on the OpenbookQA benchmark.

6.4 Results and Discussion

6.4.1 Result Comparison

Table 6.1 shows the performance of different models on the CommonsenseQA benchmark. KagNet and MHGRN are two strong baselines. Our model outperforms the KagNet by 4.8% and MHGRN by 2.9% on the CommonsenseQA benchmark. This result shows the effectiveness of our DRGN architecture. Table 6.2 shows the performance on the OpenbookQA benchmark. There are a few recent papers that exploit larger LMs, such as T5 [84] that contains 3 billion parameters (10x larger than our model,) and UnifiedQA [48] (32x larger). For a fair comparison, we use the same RoBERTa setting for the input representation when we evaluate OpenbookQA. Our model performance, potentially, will be improved after using these larger LMs. To demonstrate this point, we did additional experiments using AristoRoBERTaV7 [18] as a backbone to train our model. Our model achieves better performance when using the larger LMs compared to other baseline models. The performance shows that the more implicit information learned from pre-trained language models, the more effective relevance information established between graph nodes. We should note that GREASELM [131] and GSC [115] are the two most recent models that are developed in parallel with

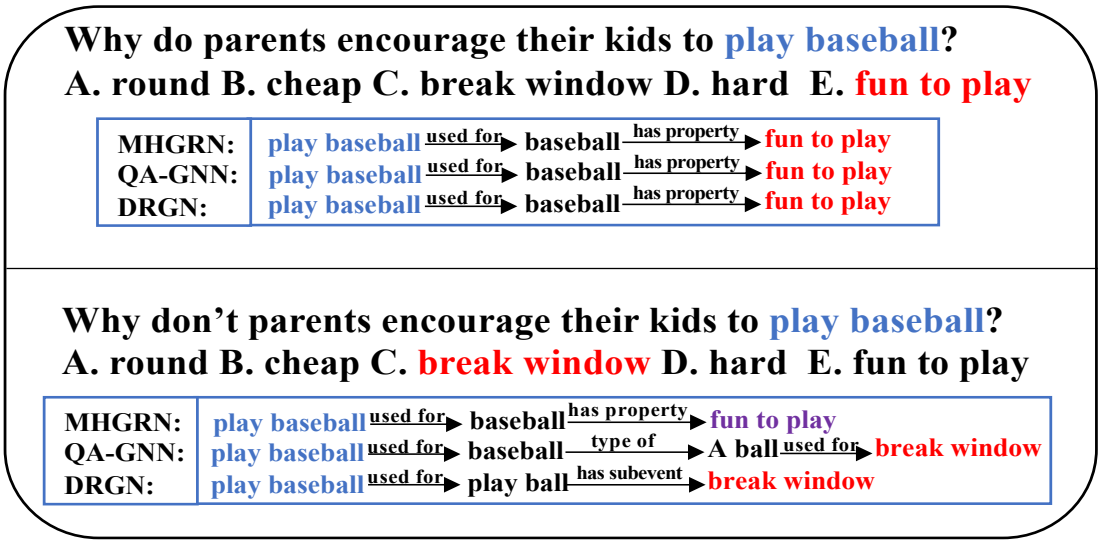

Figure 6.4 The case study of the negation examples. The question in the bottom box includes the negation words. The red colored text represents the gold answer, and the purple colored represents the wrong answer. In the blue box, each line represents the commonsense reasoning chain of each model.

our DRGN. GREASELM aims to ground language context in a commonsense knowledge graph by fusing token representations from pretrained LMs and GNN over *Modality Interaction* layers [131]. GSC designs a *Graph Soft Counter* layer [115] to enhance the graph reasoning capacity. Our results are competitive with the reported ones in those parallel works, while each work emphasizes different contributions.

6.4.2 Model Analysis

In this section, we analyze the effectiveness of our DRGN model that helps in recovering the missing edges and establishing direct connections based on the relevancy of the node representations in the KG.

Effects on Finding the Line of Reasoning As we described in Section 7.2.3, to keep the graph size small, most of the models construct the KG subgraph by selecting the entities that appear in two-hop paths. Therefore, some intermediate concept nodes and edges are missed in the extracted KG subgraph, and the complete reasoning chain from the question entity node to the candidate

Models	Test ACC % (Overall)	Test ACC% question w/ negative
RoBERTa-large	68.7 %	54.2%
KagNet	69.2 %	54.2 %
MHGRN	71.1 %	54.8%
QA-GNN	73.4 %	58.8%
DRGN	**75.0%**	**60.1%**

Table 6.3 Performance on questions with negation in In-house split test CommonsenseQA.

answer node can not be found.

For example, as shown in Figure 6.3, the question is "The student practiced his guitar often, where is he always spent his free period?" and the answer is "music room". The reasoning chain includes 2 hops, that is, "guitar → playing instrument → music room". Since the constructed graph misses the direct edge between "guitar" and "playing instrument", MHGRN and QA-GNN baselines select the wrong intermediate node and predict the wrong answer "concert" and "rock band" by the grey edges described in the Figure 6.3. In contrast, our DRGN model makes a correct prediction by computing the relevance score of the nodes based on their learned representations and forming new edges accordingly. As we describe in Section 7.2.3, our model initializes the entity node representation by large-scale pre-trained language models (LMs). The implicit representations of LMs are learned from the huge corpora, and the knowledge is implicitly learned. Therefore, these two entities, "guitar" and "playing instrument", start with an implicit connection. By looking at the relevance changes, after several layers of graph encoding, the relevance score between "guitar" and "playing instrument" becomes stronger. In contrast, the relevance score between "guitar" and "concert" becomes weaker because of the contextual information "free period". This is the primary reason why our DRGN model obtains the correct reasoning chain.

Effects on Semantic Context While the graph has a broad coverage of knowledge, the semantic context of the question and connection to the answer is not used properly. For example, dealing with negation can not perform well [127]. Since our dynamic relevance matrix includes the semantic context of the question, the relevance between the question and graph entities is computed at every

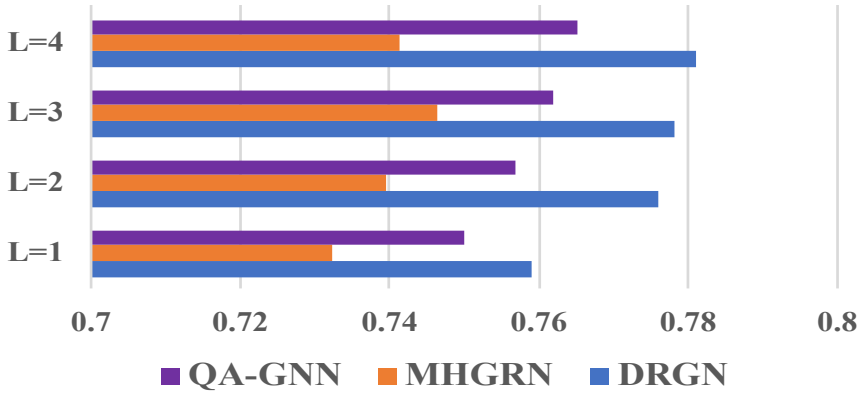

Figure 6.5 The Effect of number of layers in QA-GNN, MHGRN, and DRGN models on CommonsenseQA.

graph neural layer while considering the negation in the node representations. Intuitively, this should improve handling the negative question in our model.

To analyze this hypothesis for DRGN architecture, we compare the performance of various models on questions containing negative words (e.g., no, not, nothing, unlikely) from CommonsenseQA following recent research [127]. The result is shown in Table 6.3. We observe that the baseline models of KagNet and MHGRN provide limited improvements over RoBERTa on questions with negation words (+0.4%). However, our DRGN model exhibits a huge boost (+5.9%). Moreover, the DRGN model gains a larger improvement in accuracy compared to the QA-GNN model, demonstrating the effectiveness of considering relevance between question semantics and graph entity that experimentally confirms our hypothesis. An additional ablation study in Table 7.3 confirms this idea further. When removing the question information from DRGN, we observe that the performance on negation becomes close to the MHGRN.

Figure 6.4 shows qualitative examples of the positive and negative questions. For the positive question, all the models obtain the same reasoning chain "play baseball-(used for)→ baseball-(has property)→ fun to play", including MHGRN, QA-GNN, and our architecture. However, when adding the negative words, MHGRN obtains the same reasoning chain as the positive situation, while QA-GNN and DRGN find the correct reasoning chain. One interesting finding is that DRGN can detect the direct connection using fewer hops to establish the reasoning chain.

Models	Time	Space
l-layer KagNet	$O(\|R\|^l \|V\|^{l+1} l)$	$O(\|R\|^l \|V\|^{l+1} l)$
l-layer MHGRN	$O(\|R\|^2 \|V\|^2 l)$	$O(\|R\|\|V\|l)$
l-layer QA-GNN	$O(\|V\|^2 l)$	$O(\|R\|\|V\|l)$
l-layer DRGN	$O(\|R\|^2 \|V\|^2 l)$	$O(\|R\|\|V\|^2 l)$

Table 6.4 The time complexity and space complexity comparison between DRGN and baseline models.

Effects of Number of Graph Layers The number of graph layers is an influencing factor for DRGN architecture because our relevance matrix is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. We evaluate the effects of multiple layers l for the baseline models and our DRGN by evaluating its performance on the CommonsenseQA. As shown in Figure 6.5, the increase of l continues to bring benefits until $l = 4$ for DRGN. We compare the performance after adding each layer for MHGRN, QA-GNN, and our DRGN. We observe that DRGN consistently achieves the best performance with the same number of layers as the baselines.

Table 6.4 shows the time complexity and the space complexity comparison between the DRGN model and the baseline model. We compare the computational complexity based on the number of layers l, the number of nodes V, and the number of relations R. Our model and MHGRN have the same time complexity because both models use the R-GCN model as the backbone. Besides, QA-GNN directly adds the edge representation to the local node representation during the graph pre-processing step and learns the graph node representation without the global semantic relational adjacency matrices. After adding the dynamic relevance matrix at each graph layer, our DRGN model achieves better performance compared to other baseline architectures. For the space complexity, our model's space complexity is slightly larger than MHGRN because DRGN introduces the extra dynamic relevance matrix. However, this cost depends on the size of the subgraph, which is usually small, and it leads to a huge improvement.

Models	Dev ACC
DRGN w/o KG subgraph	69.6%
+ KG subgraph	72.6%
+ relational edges in graph	73.7%
+ question node in graph	74.9%
+ dynamic relevance matrix	78.2%

Table 6.5 Ablation Study on CommonsenseQA dataset.

6.4.3 Qualitative Analysis

To evaluate the effectiveness of various components of DRGN, we perform an ablation study on the CommonsenseQA development benchmark. Table 7.3 shows the results of the ablation study. First, we remove the whole commonsense subgraph. Our model without the subgraph obtains 69.6% on the CommonsenseQA. This shows how the implicit language model can answer the questions without the external KG, which is not high-performing but yet impressive. After adding the KG subgraph, the accuracy improves to 72.6% on the CommonsenseQA benchmark. Second, we keep the KG subgraph and add multiple relational edge information from the subgraph (described in section 6.2.5). Without the relational edges, the accuracy becomes 73.7%. This result shows that the multiple relational edges help in learning better graph node representations and obtaining higher performance. Third, we keep the multi-relational subgraph and add the question node. In other words, we incorporate the semantic relationship between the question node and the graph entities. The accuracy of the model improves to 74.9%. It demonstrates the importance of the relevance mechanism between the question information and the KG subgraph. Finally, we add the most important component, the dynamic relevance matrix, to each graph layer. The large improvement demonstrates the importance of the dynamic relevance matrix and the effectiveness of DRGN architecture.

6.5 Summary

In this paper, we propose a novel Dynamic Relevance Graph Network (DRGN) architecture for commonsense question answering given an external source of knowledge in the form of a Knowledge

Graph. Our model learns the graph node representation while a) exploits the existing relations in KG, b) re-scales the importance of the neighbor nodes in the graph based on training a dynamic relevance matrix, c) establishes direct connections between graph nodes based on measuring the relevance scores of the nodes dynamically during training. The dynamic relevance edges help in finding the chain of reasoning when there are missing edges in the original KG. Our quantitative and qualitative analysis shows that the proposed approach facilitates answering complex questions that need multiple hops of reasoning. Furthermore, since DRGN uses the relevance between the question node and graph entities, it exploits the richer semantic context of the question in graph reasoning, which leads to improvements in the performance on the negative questions. Our proposed approach shows competitive performance on two QA benchmarks, including CommonsenseQA and OpenbookQA.

CHAPTER 7

EXPLOITING COMMONSENSE KNOWLEDGE FOR DOCUMENT-LEVEL QA

7.1 Background and Motivation

Solving Question Answering (QA) problems usually requires both understanding and reasoning over natural language. In recent years, large-scale pre-trained Language Models (LMs) have made breakthrough progress and demonstrated effectiveness on language understanding in many Question Answering tasks [107, 85]. There is a large amount of world knowledge that is stored implicitly in language models that can be directly encoded and, sometimes, help in Document-level QA [24]. For example, as shown in the question 1 of Figure 7.1, "suppose plants will produce more seeds happens, how will it affect plants", the knowledge contained in a given text, (A plant produces seed, the seed germinates, the plant grows), is sufficient to predict the answer. However, there are many cases in which the required knowledge is not included in the text itself. For example, for the question 2 in Figure 7.1, the information about the "nutrient" on the seeds does not exist in the text. Therefore, an external source of knowledge is required to answer the question.

Procedural Text:
1. A plant produces a seed.
2. The seed falls to the ground.
3. The seed is buried.
4. The seed germinates.
5. A plant grows.
6. The plant produces flowers.
7. The flowers produce more seeds

Questions and Answers:
1. suppose plants will produce more seeds happens, how will it affect less plants.
(A) More (B) **Less** (C) No effect
2. suppose the soil is rich in nutrients happens, how will it affect more seeds are produced.
(A) **More** (B) Less (C) No effect
3. suppose The sun comes out happens, how will it affect less plants.
(A) More (B) Less (C) **No effect**

Figure 7.1 WIQA contains procedural text and different types of questions. The bold choices are the answers.

There are several existing resources that contain world knowledge and commonsense. Examples are knowledge graphs (KGs) like ConceptNet [97] and ATOMIC [89]. Looking back at question 2 in Figure 7.1, we observe that an explicit line of reasoning can be generated after providing the external knowledge triplets (nutrient, related to, soil) and (soil, related to, seed) derived from ConceptNet.

Two challenges exist in procedural text reasoning and using external KGs. The first challenge is effectively extracting the most relevant parts of external knowledge and reducing the irrelevant information from the KG. The second challenge is reasoning over the extracted knowledge. The irrelevant knowledge from KG will mislead the QA model in predicting the answer. Moreover, there are less sophisticated techniques proposed for using external knowledge explicitly (i.e. not through LMs) in document-level QA tasks. REM-Net [44] uses commonsense for WIQA and uses a memory network to extract the relevant triplets from the knowledge graph and solve the first challenge. However, this work has no specific mechanism for reasoning over the extracted knowledge. It just uses a simple multi-head attention operator, which combines the knowledge triplets and documents as input, to predict the answer. DFGN [122] and SAE [110] construct entity graphs using named entity recognition (NER) as the backbone to do multi-hop reasoning given the text itself. However, these models cannot deal with the challenge when the required knowledge is not in the given document.

To solve these two challenges, we propose a **M**ulti-hop **R**easoning network over **R**elevant CommonSense Sub**G**raphs (MRRG) that deals with the challenge of document-level QA when the answer requires a combination of modalities that is both document and external KG. Our motivation is to effectively and efficiently extract the most relevant information from a large KG to help procedural reasoning. First, we extract the entities, and retrieve related external triplets from KG, by learning to extract the most relevant triplets to a given text. In particular, we propose the KG Attention module to extract the most relevant triplets from large KG given the text and question and reduce the irrelevant concepts from candidate triplets. Then, we construct a commonsense subgraph based on the extracted KG triplets in a pipeline. We use the extracted subgraphs as a part of the end-to-end QA model to help in filling the knowledge gaps in the text and perform multi-hop reasoning. Our

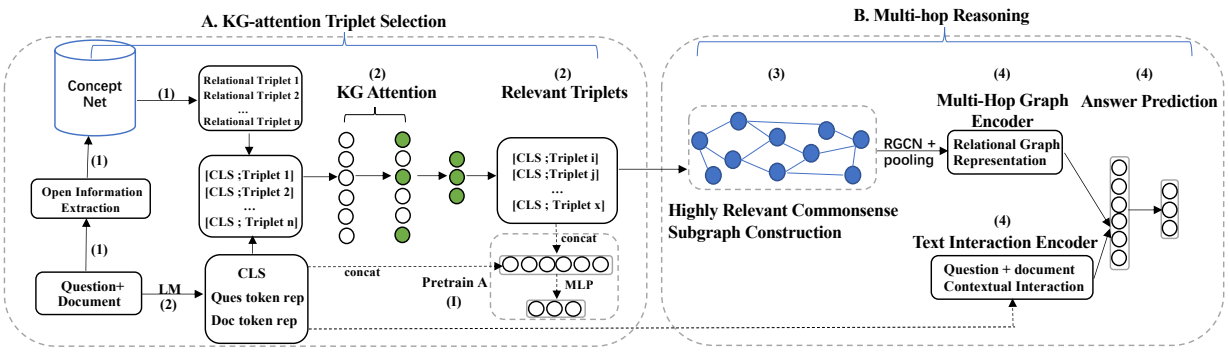

Figure 7.2 MRRG Model is composed of Candidate Triplet Extraction, KG Attention, Commonsense Subgraph Construction, Text encoder with contextual interaction, Graph Reasoning, and Answer prediction modules.

model predicts the answer by reasoning over the contextual interaction representations over the text and learning graph representations over the KG subgraphs. We evaluate our MRRG on the WIQA benchmark. MRRG model achieves SOTA and brings significant improvements compared to the existing baselines.

The contributions of our work are: **1)** We train a separate module that extracts the relevant parts of the KB given the procedure and question and reduces the noisy and inefficient usage of the information in large KBs. 2) Our end-to-end model uses the extracted QA-dependent KG subgraph to guide the reasoning over the procedural text. 3) Our MRRG achieves SOTA on the WIQA benchmark.

7.2 Model Description

Figure 7.2 shows the proposed architecture. We have numbered the parts in the figure, and here we point to the functionality of each part. (1) We extract the entities from the question and context in a preprocessing step and use them to retrieve the set of **candidate triples** from the ConceptNet. (2) We propose a novel **KG Attention** module to extract the most relevant triplets and reduce the noisy concepts from candidate triplets. (3) We augment the **commonsense subgraph** based on the relevant triplets. (4) We train a model that uses the commonsense subgraph as a relational graph network and a text encoder including question and document to do **procedural reasoning**. Below, we describe the details of each module.

7.2.1 Candidate Triplet Extraction from KG

Given the input q and C, we extract the contextual entities (concepts) using an off-the-shelf open Information Extraction (OpenIE) model [98]. For each extracted entity t_{in}, we retrieve the relational triplets $t = (t_{in}, r, t_{out})$ from KG, where t_{out} is the concept taken from ConceptNet and r is a semantic relation type. We then apply a pre-trained Language Model, RoBERTa, to obtain the representation, E^t, of each triplet:

$$E^t = f_{LM}([t_{in}, r, t_{out}]) \in \mathbb{R}^{3\times d}, \tag{7.1}$$

where f_{LM} denotes the language model operation, and the triplets are given as a sequence of concepts and relations to the LM.

7.2.2 KG Attention

The KG attention module is shown in Figures 7.3. We concatenate q and C to form Q:

$$Q = [[CLS]; q; [SEP]; C], \tag{7.2}$$

where [CLS] and [SEP] are special tokens in the LMs tokenizer process [65]. We use RoBERTa to obtain the list of token representations $E_{[CLS]}$, E_q, and E_C. $E_{[CLS]}$ is the summary representation of the question and paragraph, E_q is the list of the question tokens embeddings, and E_C is the list of the paragraph tokens embeddings output of RoBERTa.

Given triplet E^t that is generated based on the triplet extraction described in Section 7.2.1, we build a context-triplet pair E^t_z as follows:

$$E^t_z = [E_{[CLS]}; E^t_{in}; E^t_r; E^t_{out}], \tag{7.3}$$

where E^t_{in} is the representation of the head entity from text, E^t_{out} is the representation of the tail entity from KG, and E^t_r is the representation of the relation. Afterward, we compute context-triplet pair attention and a softmax layer to output the **C**ontext-**T**riplet pairwise importance **S**core CTS. The process is computed as follows:

$$CTS_t = \frac{\exp\left(MLP(E^t_z)\right)}{\sum_{j=1}^{m} \exp\left(MLP(E^t_z)\right)}. \tag{7.4}$$

Then we choose the top-k relevant triplets with the top CTS scores and then use the relevant triplets to construct the subgraph. For each selected triplet, we obtain the triplet representation, E'^t, as follows:

$$E'^t = [E'^t_{in}, E^t_r, E'^t_{out}] \in \mathbb{R}^{3\times d}, \tag{7.5}$$

$$E'^t_{in} = f_{in}([CTS_t \cdot E^t_{in}; CTS_t \cdot E^t_r]), \tag{7.6}$$

$$E'^t_{out} = f_{out}([CTS_t \cdot E^t_{out}; CTS_t \cdot E^t_r]), \tag{7.7}$$

where f_{in} and f_{out} are MLP layers, $[;]$ is the concatenation, and $[\cdot]$ is the scalar product.

7.2.3 Commonsense Subgraph Construction

We construct the commonsense subgraph G_s based on the relevant triplets from KG attention for each question and answer pair. We add more edges to the subgraph as follows: Two entities in the triplets will have an edge if a relation r in the KG exists between them. We use E'^t_{in} and E'^t_{out} for the KG subgraph initial node representation $h^{(0)}$ which is used in RGCN formulation in Section 7.2.4.

7.2.4 Reasoning over Document-level QA

To facilitate finding the answer, our MRRG architecture composes of two modules: the Graph Reasoning Encoder module and the Text Contextual Interaction Encoder module.

Graph Reasoning Encoder: this module is shown in Figure 7.2-B. Given the subgraph G_s, we use RGCN [90] to learn the representations of the relational graph. RGCN learns graph representations by aggregating messages from its direct neighbors and relational semantic edges. The $(l+1)$-th layer node representation $h^{(l+1)}_i$ is updated based on the neighborhood node representations h^l_j from the l-layer multiplied by the relational matrices $W^{(l)}_{r_1}, \ldots, W^{(l)}_{r_{|R|}}$. The representation $h^{(l+1)}_i$ is computed as follows:

$$h^{(l+1)}_i = \sigma\Big(\sum_{r\in\mathcal{R}} \sum_{j\in N^r_i} \frac{1}{|N^r_i|} W^{(l)}_r h^{(l)}_j + W^{(l)}_0 h^{(l)}_i\Big), \tag{7.8}$$

where σ denotes a non-linear activation function, N^r_i represents a set that includes neighbor indices of node i under semantic relation r. Finally, we obtain the E_{G_s} after several hops of message passing.

Text Contextual Interaction Encoder: We have obtained the contextual token representations $E_{[CLS]}$, E_q, and E_C in the KG attention module that is described in Section 7.2.2. Followed by BI-DAF research work [91], we utilize contextual interaction module to feed E_q and E_C to Context-to-Question Attention:

$$E_{C\to q} = softmax(sim(E_q^T, E_C))E_q, \tag{7.9}$$

and Question-to-Context Attention $E_{q\to C}$ to obtain the contextual interaction between question and context. Then we use LSTM to obtain the hidden state representations: $F_{q\to C} = LSTM(E_{q\to C})$, and $F_{C\to q} = LSTM(E_{C\to q})$.

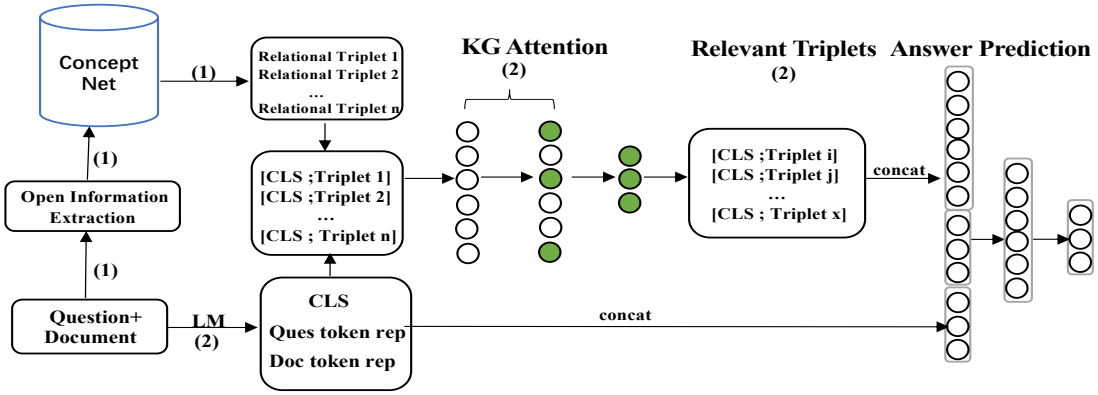

Figure 7.3 The architecture of training the KG Attention module.

7.2.5 Answer Prediction

We concatenate $E_{[CLS]}$, $F_{q\to C}$, $F_{C\to q}$, and the compact subgraph representation E'_{G_s} obtained from attentive pooling, and use it as the final representation:

$$F = [E_{[CLS]}; F_{q\to C}; F_{C\to q}; E'_{G_s}]. \tag{7.10}$$

Then we utilize a classifier MLP (F) to predict the answer.

7.2.6 Training Strategy

Training KG Attention for Triplet Selection: Figure 7.3 and the left block of Figure 7.2 show the same triplet selection model. The same KG attention module, shown in Section 7.2.2 is taken, and 3

extra MLP layers are added to the module for training as shown in Figure 7.3. The MLP is applied on the concatenation of the concatenation of $[E_{[CLS]}; E_q; E_C; E'^t_1; \ldots; E'^t_k]$ to predict the answer. We use the cross-entropy as the loss function to train the model.

Training End-to-End MRRG: After pre-training the KG attention module, we keep the learned parameters and extract the most relevant concepts and construct the multi-relational commonsense subgraph G_s. We combine subgraph representation and text interaction representation as input to train the answer prediction module by cross-entropy loss.

7.3 Experiments

7.3.1 Dataset Description

WIQA benchmark [107] is a large collection of Document-level QA examples. WIQA contains two types of questions: 1) the questions can be directly answered based on the text, called in-paragraph questions. 2) the questions require external knowledge to be answered, called out-of-paragraph questions. WIQA contains 29808 training samples, 6894 development samples, 3993 test samples (test V1), and 3003 test samples (test V2).

7.3.2 Implementation Details

We implemented our MRRG framework using PyTorch. We use a pre-trained RoBERTa [65] to encode the contextual information in the input. The maximum number of triplets is 50, and the maximum number of nodes in the graph is 100. Further details of hyper-parameters of the graph are shown in Table 7.3. The maximum number of words for the paragraph context is 256. For the graph construction module, we utilize *open Information Extraction* model [98] from AllenNLP[1] to extract the entities. The maximum number of hops for the graph module is 3. The learning rate is $1e-5$. The model is optimized using Adam optimizer [51].

[1] `https://demo.allennlp.org/open-information-extraction.`

7.3.3 Baseline Description

We briefly describe the recent SOTA baselines that use the Transformer-based language model as the backbone. The descriptions of each strong baseline are shown below:

EIGEN [69] is a baseline that builds an event influence graph based on a document, and leverages LMs to create the chain of reasoning to predict the answer. However, EIGEN does not use any external knowledge to solve the problem.

Logic-Guided [5] uses logic rules, including symmetry and transitivity rules to augment the training data. Moreover, Logic-Guided uses the rules as a regularization term during training to impose consistency between the answers to multiple questions.

RGN [135] is the recent SOTA baseline that utilizes a gating network [133] to jointly learns to extract the key entities through an entity gating module, finds the line of reasoning and relations between the key entities through a relation gating module, and captures the entity alignment through contextual entity module.

REM-Net [44] proposes a recursive erasure memory network to find out the line of reasoning. Specifically, REM-Net refines the evidence by a recursive memory mechanism and then uses a generative model to predict the answer. REM-Net is the only work that uses external knowledge for WIQA. REM-Net uses external knowledge by training an attention module that encodes the KG triplet representations for finding the answer. It does not explicitly select the most relevant triplets as we do, and the graph reasoning is not exploited for finding the chain of reasoning.

7.4 Results and Discussion

7.4.1 Result Comparison

Table 7.1 and Table 7.2 show the performance of MRRG on the WIQA task compared to other baselines on two different test sets V1 and V2. First, Both tables show that our proposed KG Attention triplet selection model outperforms the RoBERTa and has a 3.3% improvement on the out-of-para category. Second, our MRRG achieves SOTA results compared to all baseline models.

Models	in-para	out-of-para	no-effect	Test V1 Acc
Majority	45.46	49.47	55.0	30.66
Polarity	76.31	53.59	27.0	39.43
Adaboost [32]	49.41	36.61	48.42	43.93
emphDecomp-Attn [78]	56.31	48.56	73.42	59.48
BERT (no para) [24]	60.32	43.74	84.18	62.41
BERT [107]	79.68	56.13	89.38	73.80
EIGEN [69]	73.58	64.04	90.84	76.92
REM-Net [44]	75.67	67.98	87.65	77.56
Logic-Guided [5]	-	-	-	78.50
RoBERTa+KG-attention Triplet Selection	72.21	64.60	89.13	75.22
***MRRG* (RoBERTa-base)**	**79.85**	**69.93**	**91.02**	**80.06**
Human	-	-	-	96.33

Table 7.1 Model Comparisons on WIQA test V1 dataset. WIQA has four evaluation metrics, including in-paragraph, out-of-paragraph, no effect, and overall test accuracy.

MRRG achieves the SOTA on both in-para, out-of-para, and no-effect questions in WIQA V1 and V2.

Models	in-para	out-of-para	no-effect	Test v2 Acc
Random	33.33	33.33	33.33	33.33
Majority	00.00	00.00	100.0	41.80
BERT	70.57	58.54	91.08	74.26
REM-Net	70.94	63.22	91.24	76.29
REM-Net (RoBERTa-large)	76.23	69.13	92.35	80.09
QUARTET (RoBERTa-large) [85]	74.49	65.65	95.30	82.07
RGN [135]	75.91	66.15	92.12	79.95
RoBERTa+KG Attention Triplet Selection	70.02	62.30	91.23	75.86
***MRRG* (RoBERTa-base)**	76.80	67.83	92.28	80.39
MRRG (RoBERTa-large)	**78.82**	**71.10**	93.53	**82.95**
Human	-	-	-	96.30

Table 7.2 Model Comparisons on WIQA test V2 dataset.

7.4.2 Model Analysis

Effects of Using External Knowledge In the WIQA, all the baseline models achieve significantly lower accuracy in the out-of-para than in-para and no-effect categories. MRRG achieves SOTA in

the out-of-para category because of using highly relevant commonsense subgraphs. As is shown in table 7.2, the advantage of the MRRG model is reflected in out-of-para questions. MRRG improves 4.61% over REM-Net. Notice that REM-Net is the only model that utilizes external knowledge on WIQA. Figure 7.4 shows a case in which the "soil" and "nutrient" only appear in the question and do not exist in the text. The baseline models fail to answer this out-of-para question due to missing external knowledge. However, our model predicts the correct answer by explicitly incorporating the (nutrient, relatedto, soil), (soil, relatedto, seed) that connects the critical information between the question and the document.

Effect of Combine Knowledge Reasoning and Multi-hop Reasoning Both in-para and out-of-para types of questions require multiple hops of reasoning to find the answer in the WIQA benchmark. MRRG made a sharp improvement in reasoning with multiple hops due to the effectiveness of the extracted commonsense subgraph. In Particular, the MRRG model accuracy improved 2% for 1 hop, 8% for 2 hops, and 2% for 3 hops compared to EIGEN. We study some cases to analyze the multi-hop reasoning and the reasoning chains. In the third case in Figure 7.4, the extracted relevant triplets (land, relatedto, surface), (surface, relatedto, igneous rock) construct a two-hop reasoning chain "land→surface→igneous rock" that helps MRRG to find the correct answer.

7.4.3 Qualitative Analysis

Table 7.3 shows the ablation study results of MRRG in the WIQA benchmark. Firstly, we remove the commonsense subgraph and graph network. The accuracy decreases 3.4% compared to MRRG. It demonstrated the effectiveness of using external knowledge graphs on Document-level QA. Second, we report results about the impact of changing the dimensionality of the node representations in the model. we try the different dimensions of graph representation. The best performance achieved by the dimension of graph representation is 100. In an additional experiment, we use the KG attention triplet selection module to directly predict the answer without the pipeline of constructing the subgraph and using the graph reasoning module. We show the result as KG Attention Triplet

Question and Document Content	RoBERTa	+Interaction	Incorporating Triplets	+KG Attention	+Graph
Question: suppose more fruit is produced happens, how will it affect MORE plants? Content: ["The seed germinates.", "The plant grows.", "The plant flowers.", "Produces fruit.", "The fruit releases seeds." **Gold Answer: More**	X	√	(fruit, createdby, plant)	√	√
Question: suppose the soil is rich in nutrients happens, how will it affect more seeds are produced. Content: ["A plant produces a seed", "The seed falls to the ground", "The seed is buried", "The seed germinates", "A plant grows", "The plant produces flowers", "The flowers produce more seeds."] **Gold Answer: More**	X	X	(nutrient, relatedto, soil) (soil, relatedto, seed)	√	√
Question: suppose more land available happens, how will it affect less igneous rock forming. Content: ["Different kinds of rocks melt into magma", "Magma cools in the crust", "Magma goes to the surface and becomes lava", "Lava cools", "Cooled magma and lava become igneous rock."] **Gold Answer: Less**	X	X	(igneous rock, isa, rock) (land, relatedto, rock) (land, relatedto, surface) (surface, relatedto, igneous rock)	X	√

Figure 7.4 Case study of the MRRG Framework. "+interaction" means adding the contextual interaction module. "KG ATTN" means adding the KG Attention Triplet Selection module. 'X' indicates the model failed to predict the correct answer, and "✓" means the prediction was successful with the included module.

Selection in Table 7.3. The result shows that removing the triplet selection module decreases the accuracy by 1.8%. It demonstrates that the KG attention neural mechanism itself helps in extracting the most relevant information from a large KG and filling the knowledge gaps in the document.

Ablation	Model	Dev Acc
Text only	RoBERTa-base	75.51%
Text only	KG Attention Triplet Selection	77.39%
Text+Graph	GNN dim=50	79.18%
	GNN dim=100	80.30%
	GNN dim=200	79.88%

Table 7.3 Ablation and hyper-para. choices on WIQA. "GNN dim" is the dimension of graph representation.

7.5 Summary

We propose the MRRG model for using external knowledge graphs in reasoning over procedural text. Our model extracts a relevant subgraph for each question from the KG and uses that knowledge subgraph to answer the question. The extracted subgraph includes the reasoning path for answering the question and helps in filling the knowledge gap between the question and text. We evaluate

MRRG on the WIQA and achieve SOTA performance.

CHAPTER 8

CONCLUSION AND FUTURE DIRECTIONS

In this chapter, we summarize our work presented in this dissertation and highlight the contributions. Meanwhile, we discuss several potential directions for future work.

8.1 Summary of Contributions

This dissertation proposes new techniques for exploiting external knowledge and the semantic structure of data in different modalities in QA systems. My study covers a broad range of QA problems where the answer to a natural language question can be found in multiple modalities, including, Textual documents (Document-level QA), Images (Cross-Modality QA), Knowledge graphs (Commonsense QA), and combination of text and knowledge graphs.

In Chapter 3 of this dissertation, we addressed the challenges of Document-level QA. In particular, we focused on answering questions that need multiple hops of reasoning that expand over multiple documents. We exploited the semantic structure of multiple documents to find the line of reasoning to answer the questions. We extracted a graph with entities and multiple relational edges from documents using semantic role labeling (SRL). We connected the SRL graphs using shared entities. We proposed a Semantic Role Labeling Graph Reasoning Network (SRLGRN) that utilizes LM and GNN as the backbone to find the cross-paragraph reasoning paths while answering the questions. Exploiting the semantic structure of the documents makes the line of reasoning more explicit and explainable. Our proposed model obtains competitive results on both multi-hop document-level QA and single-hop document-level QA benchmarks, including HotpotQA and SQuAD.

In Chapter 4 of this dissertation, we addressed the challenges of cause-effect QA, a special type of Document-level QA. In contrast to relying on the implicit representation of the pre-trained language models, finding explicit causal relationships between entities facilitate causal reasoning over the whole document. We proposed a Relational Gating Network (RGN) that jointly extracts the most important entities and models their relations explicitly. The RGN contains an entity gating module, relation gating module, and contextual interaction module. These modules help solve

different aspects of cause-effect QA challenges, including multiple-hop causal reasoning and entity alignment. We demonstrated that modeling pairwise relationships help to capture higher-order relations. Our proposed approach achieves state-of-the-art results on the cause-effect QA benchmark, WIQA.

In Chapter 5 of this dissertation, we addressed the challenges of Visual Question Answering, a classic type of cross-modality QA. Our main contribution was to explicitly ground the entities as well as their relationships from language modality into vision modality. We proposed a novel cross-modality relevance (CMR) architecture that is an end-to-end framework that considers the relevance between textual token representations and visual object representations by explicitly aligning them in the two modalities. We model the higher-order relational relevance for the generalizability of reasoning between entity relations in the text and object relations in the image. Our proposed CMR approach shows competitive performance on two different language and vision benchmarks, including NLVR and VQA. The proposed architecture improves robustness and effectiveness compared to the previous state-of-the-art models.

In Chapter 6 of this dissertation, we addressed the challenge of knowledge-based QA given an external source of knowledge in the form of a Knowledge Graph. The main contribution has been recovering missing edges in the KG that were needed for finding the line of reasoning and answering the questions. We proposed a novel Dynamic Relevance Graph Network (DRGN) that learns the node representations while a) exploits the existing edges in KG, b) establishes direct edges between graph nodes based on the relevance scores, c) re-scales the importance of the neighbor nodes in the graph based on training a dynamic relevance matrix. As a byproduct, our model improved handling the negative questions due to deeply considering the relevance between the question node and the graph entities. Our proposed approach showed competitive performance on two QA benchmarks, CommonsenseQA and OpenbookQA, compared to the state-of-the-art published architectures.

In Chapter 7 of this dissertation, we deal with the challenge of document-level QA when the answer needs a combination of modalities that is both document and external KG. The main contribution is to effectively extract the most relevant external information from a given large KG

and combine that with the document-level information to answer the questions. We proposed a novel architecture called MRRG that extracts the entities from the document and learns to retrieve the relevant external knowledge from KG using a novel neural KG attention mechanism. Then, we constructed a KG subgraph as part of the document-level QA model to help fill in the knowledge gaps and facilitate multi-hop reasoning. We evaluated our model on the commonly used WIQA benchmark for this task. The proposed model achieves SOTA and brings significant improvements.

8.2 Future Directions

Beyond the topics covered in this dissertation, there are many new exciting directions related to the Question Answering problem, including Prompt Learning for Question Answering and Integration of Domain-Knowledge into Question Answering. In the following subsections, we point to some QA future directions.

8.2.1 Prompt Learning for Question Answering

Recent Research on large-scale pre-trained LMs demonstrates that a unified paradigm [48] could potentially apply to solve various existing NLP tasks. Developing a unified framework for QA based on prompt learning becomes a new trend for solving various QA tasks.

The QA architectures usually are based on a supervised learning paradigm. In general, these QA architectures take in an input x (question, context, image, knowledge, etc.) and predict an output y (yes/no, span answer, multiple choices of candidate answer, etc.) as $P(y|x;\theta)$ in a "pre-train, fine-tune" architecture, where θ represents the learned parameters in the model. However, prompt-based QA architectures reformulate the original input x to a prompt, $T(x)$, where T is a prompting transformation function. In general, the generated prompt $T(x)$ has several empty slots like cloze that require filling in. The empty slots are the outputs y in a "pre-train, prompt, and predict" architecture. From the application point of view, UnifiedQA [48] is a pioneer research work that reformulates various QA tasks as a unified text generation prompting problem. UnifiedQA model first generates the prompts from the questions and the corresponding context, then utilizes a

pre-trained Sequence-to-Sequence LM, T5 model, to predict the answer directly. As mentioned in [64], prompt learning models use the "pre-train, prompt, and predict" architecture to achieve SOTA on many QA tasks, including Document-level QA and Knowledge based QA.

Following this new trend of prompt-based tuning, we can point to two possible future research. The first is how to develop prompt learning for QA in different *structured* modalities, such as relational knowledge graphs, and SQL tables. The second is how to design prompts that can learn the required type of reasoning that is needed for generating output. For example to learn different types of reasoning, including spatial, temporal, compositional, etc, to enhance transferability and generalizability among different types of QA.

8.2.2 Integration of Domain-Knowledge into Question Answering

Integration of explicit domain knowledge can alleviate deep learning QA challenges [22], including inconsistent decisions, and low performance on tasks with complex reasoning. The domain knowledge can be represented through explicit constraints such as logical rules, context-free grammar, or probabilistic relations. While there are many recent research efforts on the integration of knowledge graphs based on neural representations, using knowledge in symbolic form and with explicit reasoning in neural models is less explored. Given the challenges that we faced on complex QA reasoning problems with long hops of reasoning, we think the neuro-symbolic direction is key for better generalizability of the models. This is very important in cross-modality QA, where multiple modalities need to be understood and grounded in each other.

There are very recent neuro-symbolic solutions to solve visual question answering with external knowledge such as in VQAR task [43]. In VQAR, given a query, "Identify the tall animal on the left.", we require external knowledge and commonsense reasoning ("what is the tall animal in the real world"), and spatial reasoning ("which animal is on the left of the image"), to answer the question. While these solutions are very futuristic and interesting, the main issue in dealing with this task is their scalability and efficiency to make them practical for real scenarios. Exploiting symbolic reasoning over commonsense in VQA will raise efficiency problems. On the image side,

the obtained visual features (e.g., object, attribute, and relation) are associated with the deep learning model. With the increase of the extensive visual information, such as bounding boxes, detected from the image, the time complexity of computing the deep learning model will be extremely high. Moreover, although integrating commonsense knowledge into VQA can possibly offer good interpretability, the models are hardly scalable because the number of knowledge facts using in each data example is huge.

BIBLIOGRAPHY

[1] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980, 2018.

[2] Ali Mohamed Nabil Allam and Mohamed Hassan Haggag. The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, 2(3), 2012.

[3] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[5] Akari Asai and Hannaneh Hajishirzi. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5642–5650. Association for Computational Linguistics, July 2020.

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

[7] J. Atwood and D. Towsley. Diffusion-convolutional neural networks. In *NIPS*, 2016.

[8] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

[9] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544, 2013.

[10] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008.

[11] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *ArXiv*, abs/1506.02075, 2015.

[12] Gerlof Bouma. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40, 2009.

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

[14] Jifan Chen, Shih-Ting Lin, and Greg Durrett. Multi-hop question answering via reasoning chains. *ArXiv*, abs/1910.02610, 2019.

[15] Philipp Cimiano, Vanessa Lopez, Christina Unger, Elena Cabrio, Axel-Cyrille Ngonga Ngomo, and Sebastian Walter. Multilingual question answering over linked data (qald-3): Lab overview. In *International conference of the cross-language evaluation forum for european languages*, pages 321–332. Springer, 2013.

[16] Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. In *ACL*, 2018.

[17] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[18] Peter Clark, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. From'f'to'a'on the ny regents science exams: An overview of the aristo project. *arXiv preprint arXiv:1909.01958*, 2019.

[19] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. Kbqa: learning question answering over qa corpora and knowledge bases. *arXiv preprint arXiv:1903.02419*, 2019.

[20] Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen tau Yih, and P. Clark. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *NAACL-HLT*, 2018.

[21] Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, and Peter Clark. Everything happens for a reason: Discovering the purpose of actions in procedural text. In *EMNLP*, 2019.

[22] Tirtharaj Dash, Sharad Chitlangia, Aditya Ahuja, and Ashwin Srinivasan. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Scientific Reports*, 12(1):1–15, 2022.

[23] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.

[24] 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

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

[26] Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information systems*, 55(3):529–569, 2018.

[27] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *NAACL-HLT*, 2019.

[28] Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. Searchqa: A new q&a dataset augmented with context from a search engine. *ArXiv*, abs/1704.05179, 2017.

[29] Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. Hierarchical graph network for multi-hop question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8823–8838, Online, November 2020. Association for Computational Linguistics.

[30] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *EMNLP*, 2020.

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

[32] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT*, 1995.

[33] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[34] Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[35] Ameya Godbole, D. Kavarthapu, R. Das, Zhiyu Gong, A. Singhal, Hamed Zamani, Mo Yu, Tian Gao, Xiaoxiao Guo, M. Zaheer, and A. McCallum. Multi-step entity-centric information retrieval for multi-hop question answering. In *MRQA@EMNLP*, 2019.

[36] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6325–6334, July 2017.

[37] William L. Hamilton, Zhitao Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.

[38] Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018.

[39] Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and what's next. In *ACL*, 2017.

[40] Mikael Henaff, J. Weston, Arthur Szlam, Antoine Bordes, and Y. LeCun. Tracking the world state with recurrent entity networks. In *ICLR*, 2017.

[41] Lynette Hirschman and Robert Gaizauskas. Natural language question answering: the view from here. *natural language engineering*, 7(4):275–300, 2001.

[42] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[43] Jiani Huang, Ziyang Li, Binghong Chen, Karan Samel, Mayur Naik, Le Song, and Xujie Si. Scallop: From probabilistic deductive databases to scalable differentiable reasoning. *Advances in Neural Information Processing Systems*, 34:25134–25145, 2021.

[44] Yinya Huang, Meng Fang, Xunlin Zhan, Qingxing Cao, Xiaodan Liang, and Liang Lin. Rem-net: Recursive erasure memory network for commonsense evidence refinement. In *AAAI*, 2021.

[45] Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations (ICLR)*, 2018.

[46] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[47] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017.

[48] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online, November 2020. Association for Computational Linguistics.

[49] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems*, pages 1564–1574, 2018.

[50] Seonhoon Kim, Seohyeong Jeong, Eunbyul Kim, Inho Kang, and Nojun Kwak. Self-supervised pre-training and contrastive representation learning for multiple-choice video qa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13171–13179, 2021.

[51] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

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

[53] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. In *NAACL*, 2019.

[54] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

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

[56] Jie Lei, Licheng Yu, Tamara L. Berg, and Mohit Bansal. Tvqa+: Spatio-temporal grounding for video question answering. In *ACL*, 2020.

[57] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70, 2020.

[58] Linjie Li, Zhe Gan, Yu Cheng, and Jingjing Liu. Relation-aware graph attention network for visual question answering. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10312–10321, 2019.

[59] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[60] Ruoyu Li, S. Wang, Feiyun Zhu, and J. Huang. Adaptive graph convolutional neural networks. In *AAAI*, 2018.

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

[62] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[63] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *ICLR*, 2020.

[64] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

[65] 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.

[66] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23, 2019.

[67] Pan Lu, Lei Ji, Wei Zhang, Nan Duan, Ming Zhou, and Jianyong Wang. R-vqa: learning visual relation facts with semantic attention for visual question answering. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1880–1889, 2018.

[68] Jianjie Luo, Yehao Li, Yingwei Pan, Ting Yao, Hongyang Chao, and Tao Mei. Coco-bert: Improving video-language pre-training with contrastive cross-modal matching and denoising. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5600–5608, 2021.

[69] Aman Madaan, Dheeraj Rajagopal, Yiming Yang, Abhilasha Ravichander, Eduard Hovy, and Shrimai Prabhumoye. Eigen: Event influence generation using pre-trained language models. *arXiv preprint arXiv:2010.11764*, 2020.

[70] Christopher D Manning. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *International conference on intelligent text processing and computational linguistics*, pages 171–189. Springer, 2011.

[71] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL*, 2014.

[72] Diego Marcheggiani, Anton Frolov, and Ivan Titov. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *CoNLL*, 2017.

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

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

[75] Kyung min Kim, Min-Oh Heo, Seongho Choi, and Byoung-Tak Zhang. Deepstory: Video story qa by deep embedded memory networks. In *IJCAI*, 2017.

[76] 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 *ACL*, 2019.

[77] Liang Pang, Yanyan Lan, J. Guo, Jun Xu, Shengxian Wan, and X. Cheng. Text matching as image recognition. In *AAAI*, 2016.

[78] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas, November 2016. Association for Computational Linguistics.

[79] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics, October 2014.

[80] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[81] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

[82] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[83] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[84] 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. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[85] Dheeraj Rajagopal, Niket Tandon, Peter Clark, Bhavana Dalvi, and Eduard Hovy. What-if I ask you to explain: Explaining the effects of perturbations in procedural text. In *Findings of EMNLP 2020*, 2020.

[86] 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)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[87] 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*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[88] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[89] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035, 2019.

[90] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

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

[92] Sanket Shah, Anand Mishra, Naganand Yadati, and Partha Pratim Talukdar. Kvqa: Knowledge-aware visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8876–8884, 2019.

[93] Yashvardhan Sharma and Sahil Gupta. Deep learning approaches for question answering system. *Procedia computer science*, 132:785–794, 2018.

[94] Peng Shi and Jimmy Lin. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*, 2019.

[95] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*, 2019.

[96] 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. *ArXiv*, abs/1809.02040, 2018.

[97] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[98] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. Supervised open information extraction. 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 885–895, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[99] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*, 2020.

[100] Alane Suhr, Mike Lewis, James Yeh, and Yoav Artzi. A corpus of natural language for visual reasoning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 217–223, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[101] Alane Suhr, Stephanie Zhou, Iris D. Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. In *ACL*, 2018.

[102] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7463–7472, 2019.

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

[104] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

[105] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. Deep semantic role labeling with self-attention. In *AAAI*, 2018.

[106] Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[107] Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085, Hong Kong, China, November 2019. Association for Computational Linguistics.

[108] Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4223–4232, 2018.

[109] Yao-Hung Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences. In *ACL*, 2019.

[110] Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bufang Zhou. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*, 2019.

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

[112] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[113] Shengxian Wan, Yanyan Lan, J. Guo, Jun Xu, Liang Pang, and X. Cheng. A deep architecture for semantic matching with multiple positional sentence representations. In *AAAI*, 2016.

[114] Bo Wang, Youjiang Xu, Yahong Han, and Richang Hong. Movie question answering: Remembering the textual cues for layered visual contents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[115] Kuan Wang, Yuyu Zhang, Diyi Yang, Le Song, and Tao Qin. Gnn is a counter? revisiting gnn for question answering. In *ICLR*, 2022.

[116] Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Fvqa: Fact-based visual question answering. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2413–2427, 2017.

[117] Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. Logic-driven context extension and data augmentation for logical reasoning of text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1619–1629, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[118] Zixu Wang, Yishu Miao, and Lucia Specia. Cross-modal generative augmentation for visual question answering. *arXiv preprint arXiv:2105.04780*, 2021.

[119] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Making neural qa as simple as possible but not simpler. In *CoNLL*, 2017.

[120] 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.

[121] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

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

[123] Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. Graph wavelet neural network. In *ICLR*, 2019.

[124] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.

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

[126] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *AAAI*, 2019.

[127] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *NAACL*, 2021.

[128] D. Ye, Yankai Lin, Deming Ye, Zhenghao Liu, Z. Liu, and Maosong Sun. Multi-paragraph reasoning with knowledge-enhanced graph neural network. *ArXiv*, abs/1911.02170, 2019.

[129] Rex Ying, Ruining He, K. Chen, Pong Eksombatchai, William L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[130] Amir Zadeh, Michael Chan, Paul Pu Liang, Edmund Tong, and Louis-Philippe Morency. Social-iq: A question answering benchmark for artificial social intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8807–8817, 2019.

[131] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models. In *ICLR*, 2022.

[132] Zhuosheng Zhang, Yu-Wei Wu, Zhao Hai, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiaodong Zhou. Semantics-aware bert for language understanding. In *AAAI*, 2019.

[133] Chen Zheng, Quan Guo, and Parisa Kordjamshidi. Cross-modality relevance for reasoning on language and vision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7642–7651. Association for Computational Linguistics, July 2020.

[134] Chen Zheng and Parisa Kordjamshidi. SRLGRN: Semantic role labeling graph reasoning network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8881–8891, Online, November 2020. Association for Computational Linguistics.

[135] Chen Zheng and Parisa Kordjamshidi. Relational gating for "what if" reasoning. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4015–4022. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.

[136] Chen Zheng and Parisa Kordjamshidi. Dynamic relevance graph network for knowledge-aware question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1357–1366, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics.

[137] Chen Zheng and Parisa Kordjamshidi. Relevant CommonSense subgraphs for "what if..." procedural reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1927–1933, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[138] Chen Zheng, Yu Sun, Shengxian Wan, and Dianhai Yu. Rltm: An efficient neural ir framework for long documents. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5457–5463. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

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

[140] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and M. Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[141] Jie Zhou and Wei Xu. End-to-end learning of semantic role labeling using recurrent neural networks. In *ACL*, 2015.

[142] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.