Multi-hop Question Answering

Suggested Citation: Vaibhav Mavi, Anubhav Jangra and Adam Jatowt (2023), "Multi-hop Question Answering", : Vol. xx, No. xx, pp 1–75. DOI: 10.1561/XXXXXXXXX.

Vaibhav Mavi
New York University, United States of America
vaibhavg152@gmail.com

Anubhav Jangra
Indian Institute of Technology Patna, India
anubhav0603@gmail.com

Adam Jatowt
University of Innsbruck, Austria
jatowt@acm.org

now
the essence of knowledge
Boston — Delft

Contents

Multi-hop Question Answering

Vaibhav Mavi[1], Anubhav Jangra[2] and Adam Jatowt[3]

[1]*New York University, United States of America; vaibhavg152@gmail.com*
[2]*Indian Institute of Technology Patna, India; anubhav0603@gmail.com*
[3]*University of Innsbruck, Austria; jatowt@acm.org*

ABSTRACT

The task of Question Answering (QA) has attracted significant research interest for long. Its relevance to language understanding and knowledge retrieval tasks, along with the simple setting makes the task of QA crucial for strong AI systems. Recent success on simple QA tasks has shifted the focus to more complex settings. Among these, Multi-Hop QA (MHQA) is one of the most researched tasks over the recent years. In broad terms, MHQA is the task of answering natural language questions that involve extracting and combining multiple pieces of information and doing multiple steps of reasoning. An example of a multi-hop question would be "The Argentine PGA Championship record holder has won how many tournaments worldwide?". Answering the question would need two pieces of information: "Who is the record holder for Argentine PGA Championship tournaments?" and "How many tournaments did [Answer of Sub Q1] win?". The ability to answer multi-hop questions and perform multi step reasoning can significantly improve the utility of NLP systems. Consequently, the field has seen a surge with high quality datasets, models and evaluation strategies. The notion of 'multiple hops' is somewhat abstract which results in a large variety of tasks that require

Vaibhav Mavi, Anubhav Jangra and Adam Jatowt (2023), "Multi-hop Question Answering", : Vol. xx, No. xx, pp 1–75. DOI: 10.1561/XXXXXXXXX.

multi-hop reasoning. This leads to different datasets and models that differ significantly from each other and makes the field challenging to generalize and survey. We aim to provide a general and formal definition of the MHQA task, and organize and summarize existing MHQA frameworks. We also outline some best practices for building MHQA datasets. This book provides a systematic and thorough introduction as well as the structuring of the existing attempts to this highly interesting, yet quite challenging task.

1

Introduction

1.1 Question Answering

An eventual goal of artificial intelligence (AI) is to impart the ability to reason over natural language to machines. In order to achieve this, several natural language understanding and generation tasks have been proposed that require an agent to do some reasoning to get to the goal. One such example is the task of Question Answering (QA) where given a question and some relevant context, the goal is to predict the correct answer. The question answering task provides a quantifiable way to evaluate a system's capability of language understanding and reasoning (Qiu *et al.*, 2019; Rajpurkar *et al.*, 2016a; Hermann *et al.*, 2015). It is a critical problem in the fields of natural language processing (NLP) and information retrieval (IR), and a long-standing AI milestone.

Abundance of readily-available, high-quality information on the internet facilitates the need of automated QA systems that help probe this rich content based on individual needs. Due to recent advancements in Deep Learning techniques (Lan *et al.*, 2019), the machines have become able to successfully beat human performance on datasets like SQUAD 2.0 (Rajpurkar *et al.*, 2016b). However, we have only scratched the surface of what these modern systems are capable of achieving.

Depending on the user requirements, the complexity of QA tasks may vary. Some questions can be answered in brief (*e.g.*, "*Which color do you get when you mix red and yellow paints?*") - such questions are called *objective questions* or *factoid questions*. On the other hand, there exist *subjective questions* that demand detailed explanations to meet user requirements (*e.g.*, "*Why does mixing red, green and blue paints give black color paint, but projecting red, green, and blue light on a white surface return white light?*"). A question can also be considered complex, if it requires a very niche domain expertise to answer the question (*e.g.*, "*What symptoms help diagnose chickenpox?*").

1.2 What is Multi-hop Question Answering (MHQA)?

For questions mentioned above, there might exist a single document or a single passage (formally referred as a '*context*') that can provide a justifiable answer. However, there exists certain questions that cannot be answered using a single *context* (*e.g.*, "*What is the national bird of the nation that has a negative carbon footprint?*"). The task of answering such questions is called multi-hop question answering (MHQA). The goal of MHQA is to predict the correct answer to a question that requires multiple reasoning 'hops' across given contexts (text, table, knowledge graph etc). We look at a more detailed definition of the task in Chapter 2.

The success in simple QA systems (also referred as *single hop QA*) does not necessarily entail success of MHQA systems. Min *et al.* (2018) and Qiu *et al.* (2019) observe that most questions in existing single-hop QA datasets are answerable without much reasoning, by retrieving a small set of sentences. Moreover, multi-step reasoning is required by the models to answer complex questions (refer to Table 1.1). Humans can easily perform these multi-step reasoning in their everyday tasks, yet this is still a difficult task for machines. An agent can be said to perform multi-step reasoning if it reaches one or more intermediate conclusions before deriving the final answer and each of the intermediate conclusions serves as a necessary premise for some other conclusion. This sequence of intermediate conclusions, including the final answer, is called a *reasoning chain* and each step from one conclusion to the next

can be referred to as a *hop*.

Table 1.1: Examples of various types of multi-hop questions.

Type of question	Question	Answer
Bridge Entity-based (temporal entity)	Who was the president of United States in the year in which Mike Tyson declared his retirement?	George W. Bush
Bridge Entity-based (geographical entity)	What is the national bird of the nation that has a negative carbon footprint?	The Raven
Bridge Entity-based (named entity)	What is the birth place of the tennis player who has won the most grand slams?	Belgrade, Serbia
Intersection	Who is the only person to win an olympic medal and a Nobel prize?	Philip John Noel-Baker
Comparison	Which country has won more soccer world cups - Argentina or Brazil?	Brazil
Commonsense Reasoning	If A prefers fruits over meat, when given an option of apple and chicken sandwich, what will A prefer?	Apple

It is important to note that the inability of AI systems to perform multiple steps of reasoning can be severely limiting, significantly reducing their usability. One such instance can be as shown in Fig. 1.1. Say a user is interested in knowing more about 'the daughter of A' and the only relevant information available in this context is 'B's father is C and her mother is A'. In this case, the AI system has to first infer that B is female and her mother is A. The system will then have to use common sense reasoning to conclude that B is the entity of interest and then retrieve the required information (refer to Fig. 1.1 for visual aid). Something like this seems trivial to humans but it may fatally confuse many existing AI systems. Therefore, we argue that multi-step reasoning is a crucial challenge and solving it can be a giant leap towards the goals of AI.

1.3 Applications of MHQA

As discussed above, MHQA serves as an appropriate benchmark task for evaluating an agent's ability to perform multi-step reasoning. Along with this scientific significance, the task of MHQA has various practical applications. Queries given to current web search systems can often require multi-hop reasoning to reach the relevant documents. User satisfaction when using such systems can be greatly improved by utilizing multi-hop reasoning models. Furthermore, conversations

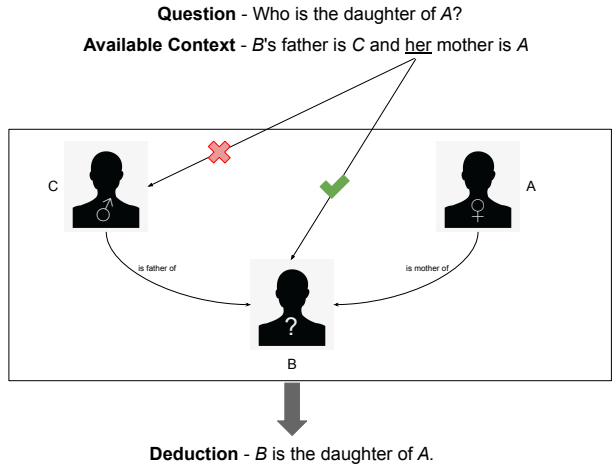


Figure 1.1: An example of multi-hop reasoning

between humans and agents can be smoother and more informative if the latter can handle complex questions. Answering a multi-hop question requires systems to aggregate information over multiple contexts. Therefore, techniques that are successful for MHQA can inspire progress in tasks such as sentence fusion (Weiss *et al.*, 2021; Geva *et al.*, 2019b) and abstractive summarization (Nayeem *et al.*, 2018; Lebanoff *et al.*, 2019), event occurrence time prediction (Wang *et al.*, 2021c), as well as multi-document summarization (Ma *et al.*, 2020; Goldstein *et al.*, 2000; Haghighi and Vanderwende, 2009; Barzilay *et al.*, 1999) or timeline summarization (Yan *et al.*, 2011; Ghalandari and Ifrim, 2020; Steen and Markert, 2019; Yu *et al.*, 2021) that require information aggregation over multiple documents. Additionally, most applications of QA such as information extraction (IE) and entailment, can be immensely benefited by multi-hop reasoning abilities (Boros *et al.*, 2021).

Kumar *et al.* (2019) argue that MHQA is a challenging task to an extent that they quantify the difficulty of a question as the number of inference steps (or hops) required to answer the question. This illustrates the direct utility of MHQA for the task of Difficulty controllable Question Generation (DQG) (Gao *et al.*, 2018) that has various applications including curriculum-learning based methods for QA systems (Kurdi *et*

al., 2019) and designing school exams of certain difficulty levels (Sachan and Xing, 2016).

Another problem closely related to MHQA consists of generating clarifying questions for conversational QA (chatbots) (Sun *et al.*, 2021; Zaib *et al.*, 2021). In this setting, the original question/query can be ambiguous and hence more information is needed to disambiguate it. The model is supposed to generate a clarifying question in natural language, asking the user for the missing information. This can be considered as another task involving multi-step reasoning and can be greatly helped by improvements in MHQA.

1.4 Overview

Recently, a variety of datasets and techniques have been proposed for MHQA, including ones designed for MHQA over Knowledge Bases and Knowledge Graphs as well as those designed for QA over tables and text. A substantial number of recent works have focused on the task of MHQA and contributed to significant advancements. High quality datasets (Yang *et al.*, 2018; Welbl *et al.*, 2018; Kočiskỳ *et al.*, 2018; Mihaylov *et al.*, 2018; Khashabi *et al.*, 2018; Chen *et al.*, 2020b; Khot *et al.*, 2020) have encouraged better models to be proposed which in turn have achieved impressive accuracy on these benchmarks. There has been a significant research in the recent years to solve the task. A variety of methods model the task as performing inference over static or dynamic graphs to find the reasoning paths (Ding *et al.*, 2019; Fang *et al.*, 2020; Zhang *et al.*, 2021; Cao *et al.*, 2019; Thayaparan *et al.*, 2019; De Cao *et al.*, 2019; Zhang *et al.*, 2020; Qiu *et al.*, 2019; Huang and Yang, 2021; Shao *et al.*, 2020; Cao and Liu, 2021). A number of works have also attempted to decompose the multi-hop questions into single hop questions or generate follow-up questions based on the retrieved information (Min *et al.*, 2019b; Cao and Liu, 2021; Sun *et al.*, 2021; Zhang *et al.*, 2021; Malon and Bai, 2020). The recent success of large language models (LLMs) has significantly influenced MHQA as well, with multiple attempts of using LLMs' strong natural understanding and emergent abilities for answering complex multi-hop questions (Zhao *et al.*, 2023b; Patel *et al.*, 2022; Balepur *et al.*, 2023; Wang *et al.*, 2023;

Rahgouy *et al.*, 2023; Xu *et al.*, 2021). We discuss all these methods in a detailed and organized manner in Chapters 4, 5 and 6.

Due to the surge in the attention received by the task over the last decade, we believe that the community would benefit from an extensive survey encompassing recent advancements in MHQA. In this work, we closely cover ∼75 works from top venues including but not limited to EMNLP, ACL, NAACL, TACL, AAAI, EACL, SIGIR, ICLR, COLING, CoRR etc. published from 2016 to 2024. The research community has already several surveys in the field of question-answering, such as for single-hop QA (Allam and Haggag, 2012; Bouziane *et al.*, 2015; Mishra and Jain, 2016; Höffner *et al.*, 2017; Soares and Parreiras, 2020; Dimitrakis *et al.*, 2020), open-domain QA (Roy and Anand, 2021; Etezadi and Shamsfard, 2023; Zhu *et al.*, 2021), medical QA (Lin *et al.*, 2021; Jin *et al.*, 2022), visual QA (Srivastava *et al.*, 2020; Wu *et al.*, 2017), etc. The surveys that are most relevant to MHQA are the ones focused on QA over knowledge bases (Fu *et al.*, 2020; Lan *et al.*, 2021; Diefenbach *et al.*, 2018; Roy and Anand, 2021) and visual QA (Srivastava *et al.*, 2020; Lin *et al.*, 2021; Wu *et al.*, 2017). However, these can be considered as sub-domains of the more general formulation of the MHQA field that this book aims to survey. Since the existing works go a long way in summarizing their intended domains, we choose to exclude Visual MHQA and MHQA over Knowledge Bases and Knowledge Graphs from the scope of this work.

We observe that despite the impressive accuracy of recent models on MHQA benchmarks, significant concerns have been raised regarding whether the models are actually able to perform multi-step reasoning in order to answer the multi-hop questions. Several works (Jansen, 2018; Wang *et al.*, 2019; Chen and Durrett, 2019; Min *et al.*, 2019a; Trivedi *et al.*, 2020; Jhamtani and Clark, 2020; Inoue *et al.*, 2020; Tang *et al.*, 2021; Tu *et al.*, 2020) conduct experiments and demonstrate that a significant portion of the accuracy can be ascribed to pattern matching and single step reasoning (also termed as *shortcut reasoning*). This points to new challenges and future directions for research in MHQA. Above all, it is fair to say that despite the inspiring progress made so far, the task of MHQA is still a long way from being solved.

A promising direction for solving some of these challenges is the task

of explainable MHQA, a particular setting of MHQA that requires the model to output the correct reasoning chain (or equivalently, some kind of representation of the reasoning chain) along with the correct answer. This increases the model's accountability and interpretability to the end user since the model now has to also explain how it reached the answer. Interpretability of the AI systems is crucial for their wide adoption for most high-stake applications such as finance, law and healthcare (Samek *et al.*, 2017; Alvarez-Melis and Jaakkola, 2017; Arras *et al.*, 2016; Biran and Cotton, 2017; Gilpin *et al.*, 2018). Consequently, more recent works (Feng *et al.*, 2020; Chen *et al.*, 2019; Yang *et al.*, 2018; Inoue *et al.*, 2020; Jhamtani and Clark, 2020) have focused on this setting. Yang *et al.* (2018) have also argued that training the model to output reasoning chain can further help in training to predict the correct answer as it serves as a useful auxiliary task. Tu *et al.* (2020) also find that using the reasoning chain as a supervision signal during training improves the performance on adversarial examples as well.

The remainder of this book is structured as follows: Chapter 2 aims to formalize the task of MHQA in a way that encompasses most existing variants. Chapter 3 describes existing MHQA datasets, their creation techniques, critiques and challenges[1]. Chapter 4 discusses traditional pre-LLM models in-depth in a structured way that leads to a taxonomy for existing methods in Chapter 6. Chapter 5 is dedicated to recent LLM based methods for MHQA, challenges of incorporating LLMs and their proposed solutions. Chapter 7 discusses the standard evaluation metrics along with evaluation methods specifically designed for evaluating multi-step reasoning/retrieval. Chapter 8 touches upon the multi-hop question generation problem. Chapter 9 then summarizes the insights of the book and critiques of the existing methods and datasets, to propose promising directions for future research in MHQA.

[1]We discuss the datasets before methods as doing so provides on overview of the existing variants of the tasks which would be helpful to understand the intuition behind the proposed architectures.

2

Formulating the Multi-Hop Question Answering Task

Before diving into the advancements, we try to provide a formal and descriptive definition of the task. Although many attempts at formally defining the task have been made, we observe that they tend to focus on specific cases of the broader task. However, we aim to cover a large variety of tasks that can be considered as variants of the MHQA task. Therefore, we try to propose a broader definition that encompasses many variants of the task that have been tackled. By doing so, we also aim to clearly define the scope of what concerns Multi-Hop Question Answering for the rest of this book.

Formally defining the task of multi-hop question answering is not straightforward, since the definition of a *hop* is ambiguous in itself. For instance, in the context of open-domain QA on text documents, a hop could signify reasoning across different documents (Yang *et al.*, 2018) whereas for QA over long documents, reasoning across different sections or paragraphs is a hop (Sun *et al.*, 2021). To the best of our knowledge, existing works do not provide a general definition of the task that encompasses its different variants. We argue that in order to systematically tackle the problem, and to obtain a good understanding of the progress in MHQA, it is crucial to first have a general definition.

We attribute the generality of the MHQA by keeping the notion of a context abstract. Depending on the task, a context can be any single independent piece of information: a sentence, a document, an image or an entity in a knowledge graph. Keeping this in mind, we formally define the task of multi-hop question answering as:

Let $\mathbb{C}$ denote the set of all contexts, $\mathcal{S}$ denote the set of all questions and $\mathcal{A}$ denote the set of all possible answers. Given a question $q \in \mathcal{S}$ and a set of related contexts, $C \subseteq \mathbb{C}$, the task is to approximate a function $f : \mathcal{S} \times \mathbb{C}^n \mapsto \mathcal{A} \cup \{\Phi\}$, that satisfies:

$$f(q, C) = \begin{cases} a \in \mathcal{A} & \exists\ P_q = \{p_1, \cdots, p_k\} \subseteq C, k > 1 \\ & \quad \&\ \ P_q \models (a\ answers\ q) \\ \Phi & otherwise \end{cases} \tag{2.1}$$

where $\models$ represents entailment, and Φ is the output when q is unanswerable using $\mathbb{C}$[1]. Given a question q and a set of contexts C, f returns an answer a that answers q by using a subset of 'gold' supporting contexts P from C. The number of gold supporting contexts k is restricted to be more than 1 to ensure that the question is not solvable using a single hop ($k = 1$ reduces the task to traditional QA).

This definition captures the commonly adopted breakdown of the task to two sub-problems: Information Retrieval (IR) and Reading Comprehension (RC). Typically, f can be decomposed into (IR) $g : \mathcal{S} \times \mathbb{C}^n \mapsto \mathbb{C}^k$ and (RC) $h : \mathcal{S} \times \mathbb{C}^k \mapsto \mathcal{A} \cup \{\Phi\}$, for some $k \in \mathbb{N},\ k > 1$ such that:

$$g(q, C) = P_q \tag{2.2}$$

where $P_q \subseteq C$ is the set of contexts relevant to q.

$$h(q, P_q) = \begin{cases} a & P_q \models (a\ answers\ q) \\ \Phi & otherwise \end{cases} \quad and \tag{2.3}$$

$$f(q, C) = h(q, g(q, C)) \tag{2.4}$$

[1] In case of multiple correct answers to q, Eq. 2.1 allows f to output any a that answers q, which is the case for datasets like SQuAD (Rajpurkar *et al.*, 2016a; Rajpurkar *et al.*, 2016b). However, some applications may require f to output all correct answers (Min *et al.*, 2020) when multiple correct answers exist.

Reasoning chain: A reasoning chain for a question $P'_q = \{p'_{q,i}\}_{i=1}^{k}$ is defined as an ordered permutation of the set P_q defined above, such that:

$$\forall\ j,\ 1 \leq j < k,\ p'_{q,j} \rightarrow p'_{q,j+1} \text{ represents the } j^{th} \text{ reasoning step and}$$
$$p'_{q,k} \rightarrow a \text{ is the } k^{th} \text{ reasoning step.}$$

It is important to note that the granularity of what constitutes a reasoning chain can also be smaller than that of the contexts. For example, when the granularity of a context is passage, the reasoning chain may consist of particular sentences or particular entities belonging to those passages.

Hop: Each reasoning step of the reasoning chain can be termed as a hop. Furthermore, some commonsense knowledge might additionally be required to perform a reasoning step from one context (i.e., a document, table, etc.) to another. In that case, the commonsense reasoning can also be considered as a hop. The definition provided here does not consider reasoning hops over external/commonsense knowledge although it can be accommodated by allowing $P_q \subseteq C \cup Q$, where Q is the external/commonsense knowledge base. However, we omit this for simplicity.

As mentioned in the introduction, many recent works focus on **explainable MHQA** to ensure the accountability and interpretability of the models while answering multi-hop questions. Formally, explainable MHQA is the setting of MHQA that requires f to output the reasoning chain P'_q (as an explanation for the answer) along with the answer, a. The set of 'facts' in the reasoning chain are often referred to as supporting facts (Yang *et al.*, 2018).

The given definition is generic and can be extended to accommodate multiple variations of MHQA, some of which are listed below. Note that the given list is not exhaustive and the proposed definition of MHQA may be extended with new variations.

- **MHQA via fact composition:** C_i represents independent facts.
- **MHQA over long documents:** QA over long documents can be considered multi-hop if the question requires the model to aggregate information across different sections, passages or sentences

of the same document (Khashabi *et al.*, 2018; Sun *et al.*, 2021). Here, each C_i is a section/passage in the same document.

- **MHQA over multiple text documents:** Each C_i is an independent document.
- **Multiple choice MHQA:** For each question, there is a small set of possible answers given beforehand. Thus, $\mathcal{A}$ in Eq. 2.1 is dependent on q, $\mathcal{A} = \mathcal{A}(q)$.
- **Open Domain vs Closed Domain MHQA:** In the open domain setting, the set of contexts relevant to the question, C spans the entire corpus i.e., $C = \mathbb{C}$ whereas in the closed domain setting, C, the input to the model along with the question q, is a small subset of $\mathbb{C}$ and may be different for each question i.e., $C = C_q \subset \mathbb{C}$. The closed domain setting guarantees that C_q is sufficient to answer q and might also contain noisy irrelevant passages. It is important to note that the distinction between open-domain and closed-domain is regardless of the sizes of $C, \mathbb{C}$, but is determined by the input of the task - whether each question is provided with a specific subset of sufficient contexts or not. As we will see in Chapter 4, the open-domain setting can be reduced to closed-domain by performing a preliminary retrieval over $\mathbb{C}$.
- **MHQA over Knowledge Bases/Knowledge Graphs:**[2] $\mathbb{C}$ is a Knowledge Base (KB) or a Knowledge Graph (KG) with C_i representing a triplet or a graph node.
- **Visual MHQA:**[3] $\mathbb{C}$ is a set of images and/or videos (or equivalently, sequence of images).
- **Conversational QA with clarifying questions:** Conversational QA can be regarded as an MHQA problem if answering the question requires the model to ask follow-up questions to the

[2]For the scope of this book, we do not consider QA over KB or KG since these have been extensively covered by Fu *et al.* (2020), Lan *et al.* (2021), and Roy and Anand (2021).

[3]For the scope of this book, we only consider text based QA.

user. In this task, the user asks a question and the model tries to answer it based on some context. If the model cannot find the information necessary to answer the question, it generates a follow-up question for the user and enquires the missing information. Here, each follow-up question can be considered as a retrieval hop and figuring out the missing information can be regarded as a reasoning hop (Sun *et al.*, 2021).

- **Temporal MHQA:** C_i represents here diverse temporal contexts which could be defined in several different ways. These could be simply documents published over different time points. However, on a more general level, contexts could be in the form of relevant time periods. Such time periods could mark bursts in the temporal distribution of relevant documents returned for an input question, or they could be formed by the focus time (Jatowt *et al.*, 2013) of relevant documents estimated either based on the embedded temporal expressions in the documents (Wang *et al.*, 2021a; Wang *et al.*, 2020), or inferred from past events and entities mentioned in text (Wang *et al.*, 2021c).

We conclude this chapter by acknowledging some of the limitations of the proposed definition. Firstly, there may be multiple ways of mathematically denoting the same task. We do not claim the presented notation to be better than other possible notations in any way. We use the definition and notation that works best for us and allows us to clearly distinguish the scope of this book. We also prefer this notation as it allows us to naturally derive the commonly used concepts in MHQA such as reasoning chains, contexts and hops. It also allows us to capture the two-step process of retrieval and reasoning that is commonly adopted in question answering. We welcome the community to progress towards a more accurate and widely acceptable definition.

3

A Comprehensive Study of Datasets: Analysis and Guidelines

The previous chapter highlighted the different forms of the task which implies the existence of multiple datasets that are unique and equally significant, and are suited to different variants of the problem. In this chapter we aim to briefly summarize existing datasets and provide their comparison. This chapter details on the process of dataset creation, statistics, comparison among the existing datasets, and is followed by our critique of these datasets. We start the discussion with existing datasets as this will provide necessary context and intuition of the data for the following chapters devoted to methods which operate on top of these datasets.

We split our discussion on datasets into three broad sections. In the first section, we dive into details of how the datasets are created, followed by a section devoted to some statistics and comparisons among the existing datasets. We end the discussion with a section explaining some critiques and observed shortcomings of the current datasets.

3.1 Dataset Creation

We begin by describing how MHQA datasets are generated. In this section, we include major challenges for creating these datasets, followed

by different steps involved in the process.

3.1.1 Challenges

Creating a dataset for multi-hop question answering is more challenging than for a traditional (i.e., single hop) QA setup. The major challenges include:

- Contexts used for questions should form a valid and unambiguous reasoning chain. This implies that a context used for a particular question should have some information overlap or some kind of entailment relation with at least one of the other contexts present in the expected reasoning chain (Yang *et al.*, 2018).
- Since the questions should require reasoning over multiple contexts, the process of question generation itself needs to somehow encapsulate information across those particular contexts.
- The creation process needs to ensure that the question is not answerable by using any single context. For instance, the question "*Who was the president of United States in the year in which World War II began?*" requires two contexts containing "*World War II began in 1939.*" and "*Franklin D. Roosevelt was the president of United States in 1939.*" However, there may be a separate context containing "*Franklin D. Roosevelt, president of United States at the start of World War II, was unwilling to...*". This challenge is more prevalent in the setting of Open Domain QA because of the incomplete prior knowledge about the possible contexts.

Therefore, it becomes crucial to look at the methods adopted during creation of existing datasets, along with their advantages and drawbacks and possible ways to mitigate these. In general, the dataset creation task can involve three major steps, which we discuss next.

1. Generating reasoning chains
2. Question generation by crowd-sourcing
3. Automatic/manual filtering

3.1.2 Reasoning Chain Candidates

Methods for coming up with reasoning chains are highly specific to the task and domain and thus, differ significantly for each dataset. Thus, we look at a few datasets individually.

1. **HotpotQA (Yang *et al.*, 2018):** HotpotQA contains only 2-hop questions formed using the first passages of documents from the English Wikipedia dump[1]. Two passages are chosen as a reasoning chain (termed *candidate passage pair*) if they satisfy either of the two conditions:

    - There exists a hyperlink from the first document to the second. The entity which forms the hyperlink is termed as the *bridge entity* and the questions are termed as *bridge* questions.
    - The entities for those passages belong to the same category (e.g. Michael Jordan and Kobe Bryant). These are specifically sampled from 42 manually created lists. Such pairs are used for creating *comparison* questions.

2. **MultiHop-RAG (Joshi *et al.*, 2023a):** MultiHop-RAG uses a set of diverse news articles and prompts GPT-4 to extract factual sentences from each article. These articles are again passed to GPT-4 for paraphrasing the factual sentence into a natural language claim. A topic and an entity are also generated in this step which are used as bridge entities for the reasoning chains.

3. **HybridQA (Chen *et al.*, 2020b):** In HybridQA, the definition of a context can be either a passage or a table. It is argued that using a table as context avoids the ambiguity of the questions that could arise when using texts. To ensure at least two hops, the questions are restricted to have reasoning chains containing at least one table and one text document. The tables are filtered from the set of tables released in WikiTables (Bhagavatula *et al.*, 2013), and the Wikipedia hyperlinks present in the cells of the

[1] https://en.wikipedia.org/

table are used to retrieve relevant passages. At most 12 sentences of the first passage of each Wikipedia page are treated as the passages.

4. **NarrativeQA (Kočiskỳ *et al.*, 2018):** The dataset is for answering multi-hop questions on long stories. To ensure that the questions are indeed multi-hop and answering requires non localized reasoning, the annotators are asked to form questions using human-generated summaries of the stories. The stories are collected from books from Project Gutenberg[2], while movie scripts are scraped from the web.

5. **OpenBookQA (Mihaylov *et al.*, 2018):** OpenBookQA requires question answering on the scientific domain using a *book* (in simpler words, a collection) of scientific facts along-with a *broad common knowledge* (large open-domain scientific sentences). The *book* and *common knowledge* are the two contexts required to answer the questions in OpenBookQA. *Book* is collected by filtering a subset of the WorldTree corpus identified by Jansen *et al.* (2018) where the *common knowledge* is collected from a collection of 14M scientific facts across Wikipedia, ConceptNet and other scientific corpora.

6. **QASC (Chen *et al.*, 2020b):** QASC is also a dataset for two-hop QA using scientific facts. The process of generating reasoning chains is very similar to that of OpenBookQA. The two contexts of the reasoning chain are chosen from a set of good quality seed facts and a large corpus of auxiliary facts, respectively.

7. **MultiRC (Khashabi *et al.*, 2018):** The purpose of this dataset is to create multi-domain multi-hop questions. Documents across various domains are selected from multiple corpora. Here, the multiple contexts are part of the same passage and the task of candidate reasoning chain generation is left to the annotators; each document is given to the annotators and they are asked to form valid multi-hop reasoning questions.

[2] https://www.gutenberg.org/

3.1.3 Generating Questions

Generating multi-hop QA pairs requires accumulation of information across different contexts which itself is an unsolved problem (Min *et al.*, 2019b; Jansen, 2018; Khot *et al.*, 2020; Kumar *et al.*, 2019; Su *et al.*, 2020; Gupta *et al.*, 2020; Sachan *et al.*, 2020; Pan *et al.*, 2021; Yu *et al.*, 2020).

Automatic generation: Welbl *et al.* (2018) automatically generate questions using existing KBs where WikiData and DrugBank (Wishart *et al.*, 2008) are used as the knowledge bases, and Wikipedia and MEDLINE[3] are used as the document corpus. However, the question and answer types in the two datasets are highly constrained, owing to the creation technique. For instance, the only question type in the MedHop dataset is of the kind $(drug_1, interacts\_with, ?)$, where in the WikiHop dataset we have $(item_1, property, ?)$. Therefore, automatic generation using KBs is argued to result in datasets limited by the incompleteness of entity relations and schema of the KB used (Zhang *et al.*, 2020; Yang *et al.*, 2018). Furthermore, automating the generation of free-form text questions is known to be a challenging task and requires substantial training data (Gatt and Krahmer, 2018; Novikova *et al.*, 2017; Min *et al.*, 2019b). However, with the latest research in LLMs, automatic question generation followed by semi-automatic verification is commonly adopted for creating small scale datasets (Joshi *et al.*, 2023a).

Crowd sourcing: Consequently, traditional datasets propose that the question creation step should be done using human-intelligence. Since the task is not very straight-forward even for humans, existing works have added comprehensive guidelines for creating questions. These annotation guidelines follow a common pattern, with slight nuances, irrespective of the task. We aim to summarize the annotation guidelines used by various datasets and hope that this could serve as a reference for the relevant future endeavours. Based on the techniques adopted by the existing datasets, the common annotation instructions and best-practices include:

[3] https://www.nlm.nih.gov/medline/medline_overview.html

1. To ensure that the questions **require multi-hop reasoning**:
    - Give examples of both positive and negative instances of what the task requires.
    - Break down the question generation task into simpler steps to prevent mistakes.
    - Use rule-based techniques to provide in-the-loop friendly hints and prevent the annotators from submitting trivial incorrect samples.
    - Use simple single-step IR or PLM (Pre-trained Language Models) based techniques to check if the question submitted is answerable using a single context only.
    - Ask annotators to submit the reasoning chain along with the question and answer.
    - Ask a different set of annotators to answer the questions using only a single context.
    - Ask a different set of annotators if the question requires multiple contexts to be answered.
2. To ensure that the questions **are answerable**:
    - Ask the annotator to also provide the answer to the question.
    - Prohibit the use of negation words in answer choices that can trivially fool baselines.
    - Ask for questions with very specific answers.
    - Ask a different set of annotators to answer the question and discard questions with significant error rates.
3. To ensure that the questions and answers **are formed well**:
    - Restrict the annotators to experts/native speakers of the intended language of the dataset.
    - Ask a different set of annotators to verify if the questions are grammatically well-formed.
    - Place a limit on the length of the answer.

- For multiple choice questions, randomly shuffling answer choices can avoid annotator biases, such as choice A being the correct answer more often than choice D.

4. To ensure that the questions **are challenging to answer**:

   - Prohibit the annotator from copying spans of text from the input contexts.
   - Ask the annotators to make the questions challenging.
   - Ask the annotators to create confusing and irrelevant incorrect answer choices.
   - Ask the annotators to consider high-level relations between entities and events rather than localized relations.

However, crowd-sourced datasets also have severe limitations. Dua *et al.* (2021) argue that these datasets usually only present a partial picture of the underlying data distribution and are marred by numerous biases, such as annotator bias (Geva *et al.*, 2019a), label bias (Dua *et al.*, 2020; Gururangan *et al.*, 2018), survivorship bias (Min *et al.*, 2019a; Tu *et al.*, 2020), and ascertainment bias (Jia and Liang, 2017). Furthermore, the absence of adversarial contexts during training time allows the models to learn shortcuts in reasoning and perform the task without doing multi-hop reasoning.

Post-processing: The above-mentioned practices and instructions might not preclude all human errors. Thus, a further manual or rule-based automatic screening is generally done to get rid of these errors. For instance, Chen *et al.* (2020b) remove a question 1) if the answer cannot be found from either table or passage, 2) if the answer is longer than 20 tokens, or 3) if the answer passage is easily retrieved using a single hop of TF-IDF retrieval. Mihaylov *et al.* (2018) and Khot *et al.* (2020) remove questions that the annotators are unable to answer correctly. Khashabi *et al.* (2018) remove a question if the annotators are able to answer it using only a single context. Welbl *et al.* (2018) aim to resolve the candidate frequency imbalance by sub-sampling the dataset to ensure that questions having any particular answer candidate

constitute not more than 0.1% of the dataset, and also by omitting articles that are about the United States. They tackle the issue of document-answer correlation by discarding the questions having any commonly occurring pair of a document and candidate answer. Finally, Joshi *et al.* (2023a) use GPT-4 to evaluate examples in the dataset against the following criteria: 1) The generated query must utilize all provided evidence in formulating the response; 2) The query should be answerable solely based on the provided evidence; 3) The response to the generated query should be either a single word or a specific entity; 4) The query must conform to its designated query type.

Table 3.1: Comparison of MHQA datasets. $\mathbb{C}$ represents the set of all contexts present in the dataset whereas C represents the set of contexts for a single question. In OD (Open Domain setting), $C = \mathbb{C}$.

	Context granularity	$\lvert\mathbb{C}\rvert$	Size	#hops	$\lvert C \rvert$	Question Source	Context source	Domain	Ans. type
HotpotQA	Passage	|Wikipedia|	112,779	1/2/3	10 / OD[a]	Wikipedia	Wikipedia	Generic	Span
HybridQA	Table, Passage	Tables: 13k Passages: 293k	69611	2/3	1 table passages	Wikitables, Wikipedia	Wikitables, Wikipedia	Generic	Span
NarrativeQA	Sentence	Books: 783 Movies: 789	46765	-	1 story	Multiple[b]	Multiple	Fiction	Generative
MultiRC	Sentence	871	9872	2.37	1 passage	Multiple[c]	Multiple	Generic	MCQ $\lvert\mathcal{A}\rvert$: 5.44
Medhop	Passage	Medline	2508	-	OD	Drugbank	Medline	Medicine	MCQ $\lvert\mathcal{A}\rvert$: 8.9
Wikihop	Passage	|Wikipedia|	51318	-	OD	Wikidata	Wikipedia	Generic	MCQ $\lvert\mathcal{A}\rvert$:19.8
QASC	Sentence	Core: 928 Aux: 7672	9980	2	OD	Core: WorldTree Aux: (Becker *et al.*, 2020)	WorldTree	Science	MCQ $\lvert\mathcal{A}\rvert$: 8
OpenBookQA	Sentence	Core: 1326 Aux: 6000	5947	2	OD	WorldTree	WorldTree	Science	MCQ $\lvert\mathcal{A}\rvert$: 4

[a] 10 for distractor setting and OD for Full-wiki setting

[b] Summary: Wikipedia; Stories: Gutenberg; Movies: http://www.imsdb.com/, http://www.dailyscript.com/, http://www.awesomefilm.com/

[c] News: CNN, WSJ, NYT; Wikipedia; Articles on society, law and justice; Articles on history and anthropology; Elementary school science books; Stories: Gutenberg project, MCTest; Movies: CMU movie summary corpus

3.2 Existing Datasets: Statistics, Comparisons and Examples

A comparison of the popular datasets used for MHQA is shown in Table 3.1. Some of these datasets have multiple settings, and different statistics for different settings. We describe these settings along with some additional details for three such datasets below:

1. **HotpotQA** test set has two settings i) *Full-wiki setting* which is open domain with the set of contexts being the first passages of all Wikipedia pages, and ii) *Distractor setting* which is closed domain with a set of 10 passages (2 gold + 8 distractors) provided with each question. The 8 distractors are collected by using a TF-IDF retriever with the question as the query. The dataset also evaluates an auxiliary task of predicting *supporting facts*. These are the sentences in the gold passages which were used by the annotator to create the question. Since many of the *comparison* questions in the dataset are of yes/no type, many models also use the answer type prediction as an auxiliary task. This involves a 3-way classification of the answer being 'yes'/'no'/the extracted span.
   The train set of HotpotQA is split into easy/medium/hard. The easy subset predominantly consists of single-hop answerable questions (over 70%) whereas distinction between medium and hard questions is determined by training multiple baselines and testing the answerability of the questions. Although the dev and test set contain only the hard questions, authors show that the easy questions are also useful while training the model.

2. **QAngaroo** datasets **(MedHop and WikiHop)** contain a masked version along with the original unmasked dataset for avoiding the candidate frequency imbalance. For instance, in the MedHop dataset, some drugs (such as Aspirin) interact with more drugs than others (such as Isotretinoin), which can lead to candidate frequency imbalance. To mitigate this, any candidate expression is randomly replaced by a unique placeholder token (e.g. "Mumbai is the most populous city in <MASK7>"). It is argued that doing this removes the answer frequency cues and also removes statis-

tical correlations between frequent answer strings and relevant contexts.

3. **NarrativeQA** question-answer pairs are created using the summaries of movies or books whereas the model is asked to answer the question based on the original story (referred to as the *story version*). Another somewhat less challenging task of answering questions by directly using the summaries, referred to as the *summary version* is also provided.

3.3 Critiques and Challenges

Even with carefully designed datasets it remains to be doubtful whether the datasets in fact require the model to perform multi-step reasoning to conclude the answer. Min *et al.* (2019a) show that the questions formed by compositions of two contexts (as used by many datasets) are not the same as they do not generalize well to multi-hop questions generated in typical use cases. Consequently, there have been multiple insightful methodologies proposed to test this aspect.

Chen and Durrett (2019) design two baseline models to predict the sentence containing the answer and constrain them to score each sentence from each context independently without looking at other sentences. Thus, the models should ideally perform poorly when trying to answer multi-hop questions. However, the performance of these baselines is overwhelmingly closer to the state-of-the-art than to a random classifier. This is observed more so for multiple choice MHQA datasets (WikiHop and a multi-choice modification of HotpotQA) than for a span-based MHQA dataset (HotpotQA). Furthermore, a no-context baseline that predicts the answer only by looking at answer candidates performs almost as good as the baselines defined above even when the number of candidates is increased significantly (refer to Figure 3 in Chen and Durrett (2019)). Thus, it is argued that multiple choice questions are easier to hack while testing and less helpful while training when compared to the span based questions.

Min *et al.* (2019a) train a single-paragraph BERT model (Kenton and Toutanova, 2019) that achieves comparable performance to the

state-of-the-art on the distractor setting, while it lags behind on the open-domain setting indicating that the open-domain setting is more challenging for a single-hop model and is worthy of future study. To further verify the hypothesis that the distractor setting is majorly single-hop solvable, human evaluation is performed. Humans achieve an accuracy of 87.37 F1 when using all the ten input paragraphs and that of 82.06 when one of the gold paragraphs is missing. In order to improve the quality of these distractors, an adversarial set of distractors is collected by training a single-hop BERT model and picking the top scoring incorrect paragraphs. Although, the performance of the single-hop model drops significantly, it is recovered after fine-tuning the model on these distractors.

Another possible way to improve the quality of distractors is to add a large number of such distractors. However, it is observed that even with 500 distractors, the single hop model performance is significant (Min *et al.*, 2019a). For the open domain setting, the performance goes down significantly indicating the challenging nature of open-domain MHQA.

On manual analysis of single-hop solvable questions, it is found that 35% of the questions in the HotpotQA dataset are solvable only by matching the entity type. These questions would require multi-hop reasoning in the open-domain setting. However, other 26% of the questions are single-hop solvable even in the open-domain setting. These are questions whose answers can be derived by finding an entity that uniquely satisfies two properties (intersection-type questions) but a unique answer can also be found by using only one of these properties. Another 8% of the questions are non-compositional single-hop and only the remaining 27% require multi-hop reasoning.

Similarly, Min *et al.* (2019b) split the HotpotQA dev set into single-hop solvable (3426) and single-hop non-solvable (3979) questions.

Das *et al.* (2019) make a similar conclusion by observing that 1184 (20%) of the questions have the answer span mentioned in both of the supporting passages and the answer can be extracted by only considering one of them.

Xiong *et al.* (2019) conduct an experiment that compares a QA model that has access to all the supporting passages in HotpotQA,

to another model that has access only to the answer (or the second hop) passage and observe that the model with full access only gives a marginal improvement from 49.43 EM to 50.96 EM.

Trivedi *et al.* (2019) find that more than 60% of the dev questions in MultiRC have at least one adjacent relevant sentence pair making the multi-hop reasoning very easy.

In summary, there seems to be enough evidence that the existing datasets have some flaws and a model should be able to exploit these flaws and show an impressive performance without being actually good at multi-hop reasoning. At the same time, these experiments point out the potential issues and challenges relating to MHQA datasets, and motivate development of higher quality datasets.

4

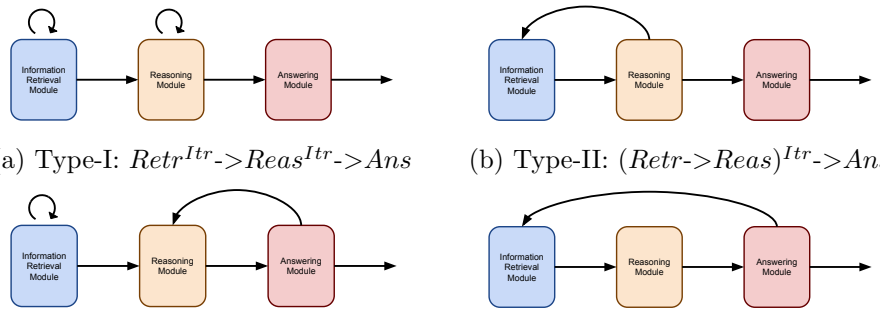
Existing Approaches for MHQA

In this chapter, we will be taking a look at the existing approaches used for solving multi-hop question answering. Since there is a very large number of works we aim to cover, we try to categorize them on many levels into categories and sub-categories. We hope this leads to a more structured study culminating into a taxonomy proposed in the Chapter 6. For each level of categorization, we briefly describe the category/sub-category followed by a deep dive into some of the representative methods for the category and finally, contrast the sub-categories and list some known pros and cons for each. To avoid over technicality, the readers can skip the deep dive into particular methods.

As discussed in Chapter 2, a large number of works divide the task of MHQA into two steps, a retrieval (IR) step that extracts all the relevant contexts from the corpus and a reading comprehension (MRC) step that reads the resulting contexts to find the answer. In general, the existing works have three basic units - Retriever, Reasoner (or Reader) and Answer Predictor[1]. These units are often employed iteratively to be able to perform multiple hops. A coarse level classification of methods

[1]Although initial works have a single module for reasoning as well as answer prediction, more recent works recommend further segregating these two as different modules. Therefore, we consider these as separate modules in our discussion.

can be how these units interact with each other. One way is to complete all the iterations of one unit before moving to the next one, and the other is to perform two or three of the tasks in a single iteration and repeat for the corresponding hops. Thus, there are four possibilities that arise by considering the multi-step nature of the retrieval and reasoning modules. These four types of models are shown in Figure 4.1.



(a) Type-I: $Retr^{Itr}$->$Reas^{Itr}$->Ans (b) Type-II: $(Retr$->$Reas)^{Itr}$->Ans

(c) Type-III: $Retr^{Itr}$->$(Reas$->$Ans)^{Itr}$ (d) Type-IV: $(Retr$->$Reas$->$Ans)^{Itr}$

Figure 4.1: The four types of architectures used for MHQA. A self loop at a module indicates that that module is independent of the succeeding modules whereas an incoming connection from a succeeding module represents some kind of feedback from that block. a) The IR and reasoning modules perform multiple hops independent of each other as well as of the answering module. b) At each hop, the IR module sends its output to the reasoning module which then gives feedback to the IR module. Answering module then predicts the answer in a single step. c) IR module first iteratively retrieves all the relevant documents for all the hops. The reasoning module performs a hop and sends the output to the answering module. Answering module either answers the question or sends feedback to the reasoning module asking for another hop of reasoning. d) At each hop, the IR module sends its output to the reasoning module which further sends its output to the answering module. The answering module either predicts a final answer or provides feedback to the IR module indicating that some required information is missing.

A few models retrieve important sentences or entities from the contexts as an intermediate step for reasoning. Since the granularity of the context is sentences for some datasets like MultiRC and OpenBookQA, it can be difficult to determine what constitutes reasoning and what constitutes retrieving. To avoid such confusion, we refer back to our definition of the task in Section 2. If the granularity of the output of a step is the same as the granularity of the context, we refer to that step

as part of the retrieval process. If the granularity is lower, we take it as part of the reasoning. We begin by describing the techniques used for these three units followed by some auxiliary tasks in practice.

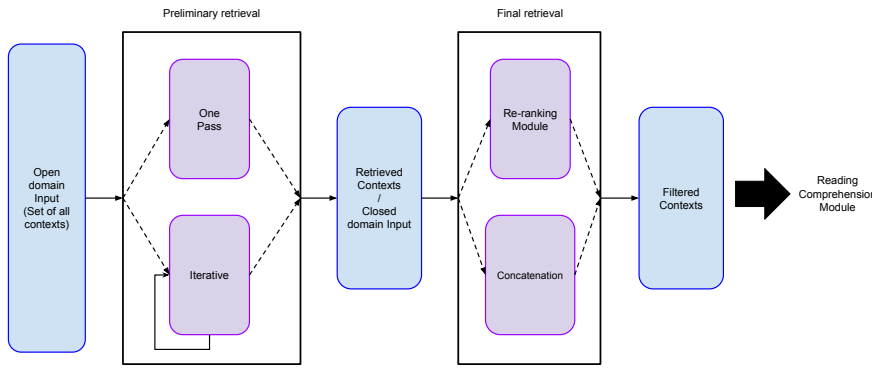


Figure 4.2: Retrieval Module. The process of retrieval can be decomposed into two steps: a) Preliminary retrieval which is a quick and high recall step to retrieve all the relevant context from the set of all contexts in the open domain setting. It can be performed either as a single pass or iteratively. b) Final retrieval which is a more thorough step to filter out all the irrelevant contexts. Because of a smaller number of input contexts, more complex re-ranking models can be used at this step. Some models directly pass the concatenation of the input documents to the reasoning module. Note that the closed domain setting of MHQA eliminates the need for preliminary retrieval.

4.1 Retrieval

The retrieval step can be a bottleneck when the set of available contexts for a question is large and becomes a particularly challenging task in case of Open Domain QA. Intuitively, multi-hop retrieval is significantly more challenging than a single-step retrieval since any retrieval error in a hop is accumulated with each subsequent hop of retrieval and leads to a well known problem of semantic drift (Yadav *et al.*, 2020). Das *et al.* (2019) verify the hypothesis by running a simple BM25 on 'easy' and 'hard' subsets of HotpotQA (which contain predominantly single-hop and multi-hop questions respectively) to find that the accuracy drops from 53.7% on the easy subset to 25.9% on the hard subset. Similarly, Feldman and El-Yaniv (2019) run a TF-IDF retriever and find that

although it succeeds in retrieving at least one of the gold passages among the top 32 passages for more than 90% questions, it often fails to retrieve both the gold passages.

The existing methods for retrieval can be broadly classified into two categories as shown in Figure 4.2: open domain retrieval where the model has to search through the entire context set $\mathcal{C}$, and closed domain retrieval where the model is provided with a smaller set of relevant (and noisy) contexts $C \subset \mathcal{C}$ along with the question q. Many of the techniques working in the open domain setting use a two step strategy: a) the first step (referred in this text as *preliminary retrieval*) is a fast and high recall filtering to get an initial set of relevant contexts. b) the second step (referred in this text as *final retrieval*) is then reduced to the closed domain setting where more complex techniques can be used to further remove the noisy contexts. This two step approach combines the best of both worlds, getting the good time efficiency of coarse retrievers and the good performance of fine-grained retrievers.

4.1.1 Preliminary Retrieval

The first step of the open domain retrieval (referred to preliminary retrieval from hereon) can be a single shot retrieval as done in single hop question answering i.e., the model attempts to extract all the relevant contexts in a single retrieval step. We call this strategy *single-pass*. Another solution would be to retrieve the passages iteratively, where each iteration of retrieval can correspond to each hop in the reasoning. We describe in detail some of these approaches below.

Single-pass approach As mentioned above, the single-pass approach performs the retrieval in one step. We list some of the representative methods of this approach below.

- Qi *et al.* (2019) and Chen *et al.* (2020b) follow a simple setting of the single-pass retrieval approach, calling the retrieval only once for each query.

- For the multi-choice questions setting, Yadav *et al.* (2019b) append each candidate answer with the question to get multiple queries

and BM25 is used on these queries to extract top $n(= 20)$ contexts (sentences in case of the MultiRC dataset).

- For the text and table hybrid setting, Chen *et al.* (2020b) parse the passages linked to each cell in the given table and retrieve all the cells relevant to the question in one go.

Critique: This strategy works well for a few-hop case where the number of contexts to be retrieved is limited and the candidate answers are provided for the multi-choice scenario. However, for more than two hops, the retrieval or the ranking algorithm could fail to capture the terms relevant to the intermediate hops. For instance, in a three-hop question, the information in the second hop could be unrelated to both the question and the candidate answers. Furthermore, if the candidate answers are not provided, this strategy could also fail to capture the second hop well (Ding *et al.*, 2019). To validate this hypothesis, Ding *et al.* (2019) and Feldman and El-Yaniv (2019) use a simple BM25 with the original question as the query to evaluate the second hop paragraph recall on HotpotQA and find the results to be indeed underwhelming.

Iterative approach Many of the existing techniques, therefore, propose a multi-step retrieval. Two main classes of iterative retrieval are query reformulation and entity-linking, described below.

Query reformulation: Deriving a query for the current hop (q_t) from the previous hop query (q_{t-1}) is commonly known as query reformulation (sometimes also referred to as pseudo-relevance feedback). Based on the strategy used for query reformulation, these methods can be further classified into two subcategories: text space reformulation and hidden space reformulation.

Hidden space query reformulation: In this case, the query is reformulated in the embedding space. As a representative example for hidden space query reformulation, we take a detailed look at Feldman and El-Yaniv (2019). They first encode the question using a Bi-GRU layer on contextualized ELMo (Peters *et al.*, 2018) embeddings and retrieve all the relevant passages using MIPS. To reduce the search space for MIPS, a TF-IDF based retriever (Chen *et al.*, 2017a) is employed to get top n_i passages. A supervised re-ranker scores the paragraphs and

each of the top k paragraphs is used to modify the question hidden representation to give k new search vectors for the next step. The reformulation module uses a bi-directional attention (Seo *et al.*, 2016) on the paragraph and question encodings followed by a linear layer with ReLU activation. A residual connection is added with this output being passed to a Bi-GRU layer followed by another linear layer with ReLU activation. Max pooling is applied to the residual output to give the updated query vector. At every step, top k paragraphs are selected by the re-ranker. The paragraph encoding being independent of the question allows the encodings to be pre-computed for an efficient retrieval during training and inference.

Text space query reformulation: Some methods reformulate the question by adding to, modifying or re-weighting the text of the question. Changing the questions in the text space allows the intermediate questions to be interpretable. We dive a little deeper into some of the methods following this approach:

- Yadav *et al.* (2020) concatenate the question with each answer candidate to obtain initial queries. Justification sentences are retrieved using the unsupervised alignment method proposed by Yadav *et al.* (2019a) which uses GloVe embeddings (Pennington *et al.*, 2014). They compute a matrix storing the cosine similarity of embeddings of the query tokens with the sentence tokens. Max pooling across sentence tokens is applied to get the most similar tokens for each query token. Dot product between this vector and a vector containing the IDF values of the query tokens is calculated to produce the overall alignment score. For MultiRC, these are selected from the sentences in all the relevant paragraphs. For QASC, relevant sentences are retrieved using heuristic IR approaches. The reformulation process only keeps the uncovered tokens and if the number of such tokens is $< T(= 2-4)$, new tokens are added from the previously retrieved sentences. The process is repeated until either a) no new query tokens are retrieved or b) all tokens are discovered. 10.7% improvement is observed when the tokens are identified using soft matching over GloVe embeddings.

- Zhang *et al.* (2021) use TF-IDF with the question to get the

first hop paragraphs. For each subsequent hop, the ALBERT based reader module is used as a span extractor on the previously retrieved documents to extract the text relevant to the question. The extracted span is concatenated with the query of the previous step for performing the next hop of retrieval. This is repeated until the reader module finds an answer or a maximum number of hops is reached.

- Qi *et al.* (2019) follow a similar approach to Zhang *et al.* (2021) with DrQA's Document Reader model used as the span extractor. During training, heuristics are used to find the oracle query.

- Yadav *et al.* (2021) retrieve k justification sentences using an alignment technique similar to Yadav *et al.* (2020). The question Q, is concatenated with each retrieved justification q_k, and the token weights are assigned as: For each token t in original question, if q_k contains t, weight for t is 1, else it is 2. All terms in q_k have a weight of 1. This is expected to result a in higher coverage. The queries are used for second hop retrieval to get a final set of N contexts. Then, $\binom{N}{p}$ evidence chains are generated and each evidence set is ranked by how many query terms are included (coverage) and top n are picked for the supervised reranking.

- Malon and Bai (2020) propose generating text based free-form follow-up questions for performing iterative retrieval. An IR model (BM25) retrieves the set of relevant contexts using the question. A BERT based three-way controller module predicts whether each context contains the final answer, contains some intermediate information or is irrelevant. A QG model is used for generating a follow up question for each passage that contains intermediate information. BERT is used for getting the answers from contexts containing the final answer. Irrelevant passages are ignored. The QG model by Zhao *et al.* (2018a) is trained on the reverse SQuAD. The controller model is trained with cross-entropy loss for ternary classification on the ground truth triplets.

Critique: Das *et al.* (2019) argue that query reformulation methods do not necessarily use the information about entities present in the

evidence as they might not be the most frequent/ salient terms in it. Therefore, many works have proposed using entity mentions in the retrieved contexts for performing the second hop retrieval.

Using entity links/hyperlinks: In this approach, entity mentions in the first hop retrieval are used to find Wikipedia passages with titles containing any of these entities. This is an efficient technique since the entities can be extracted as part of the pre-processing step. It can be noted that this technique is particularly effective for HotpotQA since the creation process of the HotpotQA uses the Wikipedia paragraph for a bridge entity as the second hop context. However, this also indicates a bias of dataset design when building models (Fang *et al.*, 2020). Trying to imitate (or benefit from the knowledge of) the creation process of some dataset while building the models for this dataset can improve accuracy on that dataset but is not guaranteed to perform well on the other datasets. On similar lines, Das *et al.* (2019) even suggest not to use off-the-shelf entity linker as they are usually trained on Wikipedia and can lead to data leakage.

Below, we list some of the representative methods for this class of methods.

- Das *et al.* (2019) use BM25 for the first hop retrieval and then retrieve Wikipedia paragraphs for each mentioned entity. Since HotpotQA is known to contain both single-hop and multi-hop questions, a self-link is added for each entity retrieved in the first hop to deal with single hop questions.

- Xiong *et al.* (2019) use a Hybrid TF-IDF + BM25 for the first hop retrieval to get 10 documents. Span prediction and external entity linking is used to get the bridge entities from the first hop passages. A supervised re-ranker gives top 10 entities and Wikipedia passages for these entities are used as the second hop passages. A span loss is added for predicting answer passage title entities as an auxiliary task.

- Ding *et al.* (2019) and Fang *et al.* (2020) retrieve Wikipedia passages whose titles contain an entity mentioned in the question as the first hop passages. Hyperlinks from these passages are used

to get the second hop passages.

- Shao *et al.* (2021) extract key words and match them with the paragraphs title and select top N_1 paragraph with best TF-IDF scores. Apart from these, top N_2 paragraphs with best TF-IDF scores are added. For the second hop, add all the paragraphs having hyperlinks from and hyperlinks to the first hop paragraphs.

Critique: Sidiropoulos *et al.* (2021a) evaluate the performance of the two sub-categories of iterative retrieval approaches by evaluating the performance on each of the two hops on HotpotQA dataset. It is argued that entity based retrieval is limited by the availability of such hyperlinks and passages whereas the reformulation approach is limited by the performance of lexical term-based retrieval. They observe that while BM25 followed by BERT based reranking performs well on the first hop, it fails on the second hop retrieval (evaluated as the ability to retrieve second passage given the first passage). Similarly, although the model by Xiong *et al.* (2020) has the best overall performance, it has some room for improvement when evaluated for the second hop retrieval. Hence, a hybrid technique is proposed where the re-rank model is used for the first hop and a single-hop dense passage retrieval model (Sidiropoulos *et al.*, 2021b) is used for the second hop. Results show that the hybrid technique outperforms the existing techniques which calls for further study in this direction.

4.1.2 Final Retrieval

After getting an initial pool of retrieved contexts, many works suggest using a more fine-grained and more sophisticated retrieval on the retrieved contexts to get a better quality of contexts. Based on how the initial pool of retrieval output is processed, we can categorize the approaches into concatenation, supervised re-ranking and unsupervised re-ranking.

Concatenating/union Some of the methods do not filter the input contexts and directly pass a union or concatenation of the contexts to the reasoning model. For example:

- Yadav *et al.* (2020) maintain N-parallel reasoning chains in the preliminary retrieval step and a union of these chains is passed to the answer prediction module.
- Ding *et al.* (2019) and Yadav *et al.* (2019b) do not filter the passages and directly perform reasoning at a lower granularity.

Supervised re-ranking Majority of the methods use a supervised module to score and rank the input contexts, and then use some criterion to filter the less relevant documents before passing to the answer prediction module:

- Feldman and El-Yaniv (2019) use a linear layer with sigmoid activation to get a relevance score of each sentence in a paragraph with the question. Max pooling across all sentences is applied to give a relevance score and top-k most relevant paragraphs are picked. Max pooling allows only one of the sentences in a paragraph to be relevant to the question with a high score.
- Das *et al.* (2019) use BERT to compute query-aware embeddings for every pair of the first hop and second hop paragraph. The two paragraphs are concatenated and fed to 2-layer Neural Network to get a score. Top k pairs are passed to the reader module.
- Xiong *et al.* (2019) use a bi-LSTM layer (Hochreiter and Schmidhuber, 1997a) to predict relevance of each second hop passage with the question.
- Fang *et al.* (2020) use a RoBERTa encoder (Liu *et al.*, 2019) followed by a fine tuning layer to get the top N paragraphs.
- Zhang *et al.* (2021) pass the node (document) representations to a binary classifier for predicting their relevance and keep the k highest scoring documents.
- Yadav *et al.* (2021) use RoBERTa (Liu *et al.*, 2019) for reranking trained to predict the F-1 score of evidence chain.
- Zhang *et al.* (2020) and Huang and Yang (2021) concatenate paragraphs with the question and feed it to a BERT followed by

a binary classifier and keep N(=3) paragraphs with the highest scores. Qiu *et al.* (2019) follows a similar approach but set a threshold to retrieve a variable number of paragraphs.

- Shao *et al.* (2021) use the gated memory flow network inspired by the Neural Turing Machine (Graves *et al.*, 2014) to model the probability of a paragraph being the next context in the reasoning chain, conditioned on the question and the previous paragraphs in the reasoning chain. At every time step t BERT is used to compute the question aware embeddings of the paragraphs, x_t. A KVMemNN architecture models the memory as a set of key-value pairs. The model passes the key vectors and x_t to linear layers and applies Softmax to the output of $W_x x_t \cdot W_k k_i$. The output after Softmax multiplied with another matrix W_v is then used as weights while summing the value vectors v_i to give the readout vector o_t. The memory reading process is similar to computing self-attention (Vaswani *et al.*, 2017a) and the authors concatenate the outputs after computing o_t for the h attention heads. o_t and x_t are passed to another linear layer with tanh and sigmoid activation to give the relevance score s_t. x_t is written to the memory if $s_t > gate$ where $gate$ is a hyper-parameter. Hard negative examples are generated by training a BERT based model to predict the relevance score s_t and choosing the top-8 of the non-evidence paragraphs.

- Dua *et al.* (2021) propose generative context selection that learns to predict how the question would have been formed. Mathematically, the model tries to learn $p(a, q|C)$ instead of $p(a|q, C)$ (as in discriminative models). This probability is modeled as

$$p(a, q|C) = \sum_{c_{ij}} p(a|q, c_{ij}) \cdot p(q|c_{ij}) \cdot p(c_{ij}|C) \tag{4.1}$$

where $p(c_{ij}|C)$ is the *prior* that computes compatibility between any two contexts, $p(q|c_{ij})$ is the question generation model that predicts the probability of q being formed from the given contexts, and $p(a|q, c_{ij})$ is the standard answering model. During inference, the model retrieves the contexts as $c^*_{ij} = argmax_{c_{ij}} p(q|c_{ij}) \cdot p(c_{ij}|C)$. To model these probabilities, a pre-trained T5 is used

for obtaining the contextual embeddings. The prior and generative models are trained together. Concatenation of every pair of contexts is passed to an encoder and a Pointer Generator Network (PGN) (See *et al.*, 2017a) decoder is used to predict the question. The training objective is to increase the likelihood of the question for gold context pairs and the unlikelihood (Welleck *et al.*, 2019) for a sample set of negative context pairs. T5 is used for answering using the best pair of contexts. The advantage of using a generative model is that it avoids the annotator biases in the dataset. This hypothesis is tested by running the model on the adversarial set of questions formed by Tu *et al.* (2020). A multi-label sentence classifier $p(s|q, C)$ that selects relevant sentences is found to have a better performance but at the same time, is more biased.

- Chen *et al.* (2020b) feed each cell along with its neighboring cells to the cell encoder to obtain their representations. The representations are aggregated and further fed to a feed-forward neural network to obtain a score.

- Sun *et al.* (2021) perform the retrieval in two steps where the first step retrieves a paragraph, and the second step retrieves a sentence from these paragraphs. A weighted sum of paragraph and sentence retrieval scores is computed to find the best retrieved sentence. The retrieval score of the paragraph retrieved from the first hop is broadcast to the sentences in the paragraphs. The best retrieved sentence fed to BERT-large for extractive QA.

Unsupervised re-ranking Some of the methods for supervised re-ranking include:

- Yadav *et al.* (2019b) consider all $\binom{n}{k}(k \in [2, 5])$ reasoning chains formed by the n retrieved passages and use a simple formula to compute the scores. They define **R**elevance as the average BM25 scores of the chain, **O**verlap as the word overlap between each sentence pair in the chain and **C**overage as the product of word overlap of sentence with the question and with the answer. The final score is given by $\frac{R \cdot C}{O}$. By experiments, $k = 3, 4$ are found to

be the best values with the justification that getting two correct retrieved passages in a chain of size two is tough while a chain of size 5 will suffer from noise.

- Chen *et al.* (2020b) retrieve cells from the table in the HybridQA dataset in an unsupervised manner. A cell is selected if its value is either mentioned in the question, or is min/max of the corresponding column. A cell is also added if it hyperlinks to one of the passages that are retrieved by a TF-IDF retriever.

Critique: Supervised re-ranking can lead to a model that is more suited to the task at hand but requires some training signals which may not be available for certain tasks. On the other hand, unsupervised methods are more flexible but might not be optimized for a given task. In particular, Yadav *et al.* (2019b) and Yadav *et al.* (2020) experimentally verify that using a supervised re-ranking method has an implied shortcoming of resulting in a domain dependent performance. The performance for each domain depends on the portion of training data belonging to that domain. Thus, it is attempted to propose **unsupervised re-ranking** techniques that can lead to similar level of quality as supervised re-ranking.

Another parallel classification of the re-ranking approaches is based on whether the module scores each retrieved context independently or together as constituents of a possible reasoning chain. We call the two categories of methods **Context re-ranking** and **Chain re-ranking** respectively. Table 6 lists the methods belonging to the two categories. Yadav *et al.* (2019b) show that evaluating the reasoning chain as a whole leads to better performance than evaluating each passage independently.

4.2 Reading Comprehension

A reading comprehension module is responsible for reading the final set of retrieved contexts, combining the information across contexts and performing the reasoning steps or hops. The output of this module can either be an answer or some representation of the reasoning performed (for example, a reasoning chain, a semantic graph or some latent representation). A vast majority of approaches have used graphs or question

decomposition to perform the reasoning, therefore we classify the reading comprehension techniques into graph-based, question decomposition based and miscellaneous categories.

4.2.1 Graph-based Techniques

The general flow of the graph based methods can be summarized in a three-step process: 1) A graph of one or more types of nodes (entity, sentence, paragraph, document nodes, etc.) is constructed and edges are added based on some lexical heuristics. 2) Contextual encodings of the nodes are passed to one or more layers of a Graph Neural Network (Kipf and Welling, 2016a) (or a Graph Attention network (Veličković *et al.*, 2018) or a Graph Convolutional Network (Kipf and Welling, 2016b)). These layers update the representation of each node using the representation of each of its neighbouring nodes. In this way, after n such layers, nodes separated by a path of $\leq n$ edges have shared information with each other. Therefore, each layer is expected to perform one hop of the reasoning process. 3) The updated embeddings are passed to the answering module.

As can be expected, some methods slightly deviate from the proposed general flow. We describe these differences below along with some details on graph building and reasoning steps.

- Ding *et al.* (2019) build an entity graph by starting with the entities extracted during the retrieval process. For each node, a BERT model takes its representation along with the question and a retrieved passage and extracts entities from the paragraph to be added to the graph. The process is repeated until either a) no new nodes can be added or b) a maximum number of nodes are extracted. Sentences containing the extracted entities are considered as 'clues' for the respective entities. Clues are then used for updating the node representations using a Graph Neural Network (GNN). Clues are also used later for the auxiliary task of supporting facts prediction.

- Fang *et al.* (2020) propose constructing a hierarchical graph network with four types of nodes that represent question text, relevant

paragraphs, sentences in these paragraphs and entities in these sentences. Bidirectional edges are introduced between the first hop source sentences and second hop target paragraphs. Edges are also introduced between paragraph and its sentences; sentence and the entities it contains; question and the entities it contains; between all paragraphs; between each sentence and its previous and next sentences. RoBERTa is used for encoding and a bi-attention and a Bi-LSTM layer are used for getting initial contextualized representations of the nodes. Graph Attention Network (GAT) (Veličković *et al.*, 2018) is applied over the hierarchical graph and the updated representations are merged with the original contextual representations using a gated attention mechanism. The merged representations are passed to the answer prediction module. The proposed gated merging is supposed to be effective for dealing with smaller number of hops.

- Xu *et al.* (2021) build a use abstract meaning representations (AMR) (Banarescu *et al.*, 2013) to build a semantic graph from a passage. Graphs across passages are connected by adding edges between the same concepts in different graphs. A non-parametrized method is employed for generating reasoning chains. A depth first search (DFS) is used to find all paths from the question nodes to the answer nodes and all the edges on the path are used to recover the facts from the context that form the reasoning chain. The reader reads these reasoning chains and the question to predict the answer. A reinforcement learning based chain-aware loss is proposed to boost the performance of the system.

- Similar to Xu *et al.* (2021) ll13 and Li and Du (2023) build a semantic graph. However, the entities and relations are extracted using GPT-3.5.

- Zhang *et al.* (2021) construct a graph of documents and add edges if the documents have a shared entity. Question aware embeddings of the document are generated by Albert and used as initial node representations. GAT is used to update the representations of each shared entity. To update the representations of non-entity tokens,

they are fed to a transformer along with the updated representations for multi-document fusion. The updated representations are passed to the answer prediction module.

- Cao *et al.* (2019) make entity graph of all entity mentions of the answer candidate in supporting documents. Cross-document edges are added between the mentions about the same entity and within-document edges are added between every node pair belonging to the same document. For initial representations, GloVe and ELMo are used for token-level and contextualized embeddings respectively. Contextualized embeddings are crucial since graph nodes only contain entity tokens. To deal with entities containing multiple words, average is taken over their embeddings. The two embeddings are passed to a 1 layer NN (replaced by Bi-LSTM for encoding the question) for getting the initial node representations which are concatenated with the NER and POS embeddings before feeding to a GCN layer.

- Thayaparan *et al.* (2019) compute similarity between GloVe embeddings of each word of the sentence with each word of the query and take the mean among m(=5) closest sentence words to get a sentence score and select top k(=25 or 30) sentences. A graph containing sentence nodes and document nodes is created with only the selected sentences. Edges are added between documents if they have a shared entity and between a paragraph and all its sentences. Adding edges among sentences adds complexity without much improvement. Initial representation of a node is computed by passing the matrix of GloVe embeddings of the entities contained in the sentence or paragraph to a bi-linear attention (Kim *et al.*, 2018) layer on nodes and query. To compress the representations into fixed size, self-attention is used. T(=3) layers of Gated Graph Neural Network is used to update the representations.

- De Cao *et al.* (2019) create an entity graph similar to Cao *et al.* (2019) but have some additional edge types: co-reference edges: between co-reference mentions of the same entity (separate type since these are less reliable) and complement edges which signify

no connection between the two nodes. Co-reference edges is a separate edge type since these are less reliable owing to the error in the co-reference system. These are not required for the masked version. The query representations are formed by passing ELMo (Peters *et al.*, 2018) to a Bidirectional RNN (Rumelhart *et al.*, 1985). Initial query dependent node representations are formed by passing the ELMo contextual embeddings to a feed forward network. L layers of a gated relational-GCN (Schlichtkrull *et al.*, 2017) R-GCN, a version of GCN are applied to get the final representations which are passed to the prediction module. Relational-GCN is able to accommodate different edge types by using different weight matrices for the neighbouring nodes connected by different edge types. An ablation experiment verifies that, although useful, co-reference edges have the least contribution to performance. Another experiment tries to predict the edge type by training a model but the performance is poor.

- Chen *et al.* (2020b) solve the text-table hybrid MHQA by using the retrieved cells from the previous stage to then decide which neighboring cell to hop to.

- Qiu *et al.* (2019) construct an entity graph and add edges between entities with the same mention text, between entities belonging to the same sentence, and between each entity in a paragraph and each entity in its title. Question and contexts are concatenated and passed to a BERT model followed by a bi-attention layer to get the contextual node representations. The following four steps are performed iteratively: 1) Mean-max pooling over tokens in an entity mention to update the entity embeddings. 2) Attention between query and entity is used to compute the mask weights which are multiplied to the entity representations before feeding to a GAT. 3) Updated entity representations are concatenated to entity tokens and representations of all tokens are updated using an LSTM layer. 4) Bi-directional attention between query and entity representations is used to update the query representation. Updating embeddings of each token is necessary since the answer might not be an entity.

- Zhang *et al.* (2020) create two different graphs: a) sentence graph with edges between sentences belonging to the same paragraph and between sentences that share an entity. b) entity graph with edges between entities appearing in the same sentence, between different mentions of same entities and between each entity in the paragraph to each entity in its title. The intuition is that humans first focus on question-related paragraphs, then sentences, and then the important words. BERT word embeddings are passed to a bidirectional attention layer to get the contextual embedding. They also propose a novel similarity matrix for the layer. Self attention is applied to the query and query aware node embeddings are passed to a GAT. Self attention is applied on the sentence node representations and the output is appended by the representation of each word in the sentence. An LSTM is applied to fuse the output of 2 graphs.

- Huang and Yang (2021) form a sentence graph and add an edge between sentences s_i and s_j with the weight w_{ij} given by:

$$w_{ij} = \begin{cases} \frac{1}{1+e^{-n+K_1}} & s_i \text{ and } s_j \text{ have } n > 0 \text{ shared entities} \\ \frac{1}{1+e^{d+K_2}} & s_i \text{ and } s_j \text{ belong to the same paragraph} \\ & \text{separated by } d \text{ sentences.} \end{cases} \tag{4.2}$$

  These weights allow the model to deal with the edge types in a novel way. The concatenated paragraphs along with the question are fed to a BERT followed by a bi-attention layer. Sentence representations are obtained by extracting token level embeddings from the paragraph encoding and a weighted addition of token embeddings where the weights are calculated using a two layer MLP (Multi Layer Perceptron). Most methods using GNN perform message passing for each node in parallel. This requires the representation of nodes to be updated exactly L times where L needs to be specified as a hyper-parameter. If L is large, it leads to over smoothing. If it is too small, it inhibits long-path reasoning. Moreover, this algorithm performs unnecessary updates leading to inefficiency. Thus, a novel message passing algorithm is proposed

which performs a BFS starting from the question and passing the messages through every edge that is visited in the process.

Critique: Thayaparan *et al.* (2019) argue that the advantage of using graph-structured representations lies in reducing the inference steps necessary to combine multiple pieces of information related in a path-like manner. The fact that the required graphs (or some parts of it) can be created offline gives it a computational advantage. However, graph structure also has certain limitation (Shao *et al.*, 2021), particularly for comparison type questions in HotpotQA where the evidence paragraphs about the two entities are independent. Many of the techniques assume that the relation between nodes is directional which may not always be the case. For gated GNNs, computational efficiency as well as the learning ability of the model degrades with the increasing number of nodes and edge types in the graph used.

While the graph structure is prevalent for multi-hop reasoning, Shao *et al.* (2020) argue that it is not necessary and that graph attention can be considered as a special case of self-attention. Treating Qiu *et al.* (2019) as the baseline, results are compared after removing the graph fusion module. It is observed that the performance gained by using the graph structure can be easily compensated for by fine-tuning the BERT based encoder. While the graph structure enables the model to focus only on adjacent nodes, a model without any prior knowledge can still learn this behaviour. This is verified by further experiments. When the graph fusion module is replaced by self-attention layers the results are very similar whereas replacing it by a transformer leads to a significant improvement. Graph attention reduces to self-attention for a fully connected graph making it a special case of self-attention. Attention patterns are visualized and observed similar to Kovaleva *et al.* (2019). It is found that the pre-trained transformers are able to capture several types of attention patterns: a) between entities, b) between co-referenced entities, c) between an entity and its attribute, and d) between an entity and a sentence. Depending on the structure of the graph, these patterns may or may not be covered by graph attention. Therefore, self-attention can be said to be more general and flexible than graph attention.

4.2.2 Question Decomposition Techniques

The key idea here is to decompose the multi-hop question into multiple single hop questions and use a single-hop QA model to answer each of the questions. This approach is particularly effective for questions like the bridge questions in HotpotQA that are constructed using a bridge entity. These questions can be easily decomposed into two sub-questions by finding the bridge entity. Patel *et al.* (2022) manually label the decomposition for questions across multiple datasets and use GPT-3 and RoBERTa to answer the questions and observe that question decomposition can help fix 60% of the errors made on the original multi-hop questions. However, this technique exploits the knowledge about the structure of question and thus, is difficult to adapt when the structure of the questions can be flexible. Below we list some representatives methods for this approach:

- Min *et al.* (2019b) use the hypothesis that each sub-question can be formed from a multi-hop question by copying and lightly editing a key span from it. Questions in HotpotQA are categorized into four categories: bridge (47%), comparison (22%), intersection (23%)(questions asking for an entity that satisfy multiple properties) and others (8%). Editing methods are proposed to be dependent on the question type for the first three categories. Question text is segmented into several spans by training a pointer network that predicts p_{ij}, the probability of the i^{th} word to be the j^{th} index to split the question. 400 annotations are manually generated for training. For each question, three such indices for bridge questions, two indices for intersection and four indices for comparison questions are predicted by maximizing the joint probability of the decomposition. These spans are modified slightly depending on the reasoning type. Any single-hop QA model can be used for answering each single hop question. Concatenation of the question, the reasoning type, the answer, and the evidence is encoded using BERT and scored using 1 layer feed forward NN with sigmoid activation. The reasoning type is decided as the one resulting in the maximum score. An alternative way where the reasoning type is predicted before decomposing and answering is

tried and found to be giving poor results. To verify the original hypothesis, same technique is tested with span-based questions replaced by human written questions. There is a little difference in model performance indicating that the span-based sub-questions are as effective as free-form sub-questions.

- Cao and Liu (2021) break down the task of MHQA into two components: Coarse grained decomposition and Fine grained interaction. The decomposition module is responsible for making the high dimensional vector distribution of entity nouns or pronouns more inclined to the intermediate answer to the question. The fine grained interaction module is a modified bidirectional attention module. The resulting context representations are passed to a self-attention layer and then used for supporting facts prediction by passing to a Bi-GRU layer.

- Sun *et al.* (2021) propose MHQA methods for four different datasets: HybridQA, QASPER, HotpotQA-Long and ShARC-Long. QASPER is originally proposed as single hop QA over long documents whereas ShARC is proposed as a conversational QA task. The paper utilizes the fact that long documents are often structured into sections and subsections which can be used as separate contexts with limited dependence among them. A pre-trained ETC model (Ainslie *et al.*, 2020) is used as the question encoder and the context encoder. ETC is a pre-trained Mask Language Model that employs a global-local attention mechanism. ETC assigns to each sentence a special global token that only attends to local tokens in the sentence, and its embedding is trained to summarize the information of local tokens in the sentence. ETC additionally adopts Contrastive Predictive Coding (CPC) (Van den Oord *et al.*, 2018) to train the embedding of global tokens to make them aware of other sentences in the context. It takes in a sequence of sentences and outputs the contextualized representations of each sentence. For conversational QA, the question is formed by concatenating the original question with a sentence formed by each pair of follow-up question and answer. ETC is again used to encode the question paragraph. For the

2-hop questions (HotpotQA, HybridQA), a null sentence is passed to ETC along with the question to get two question encodings. The paragraph embeddings are computed by a weighted sum of sentence embeddings, where weights are the attention score of the query vector to that sentence. At every step, each sentence and every paragraph in the document is attended to the query and the output is used to update the query representations. The updated query representation are then combined with the embedding of next query sentence to give the query vector for the next hop.

Critique: Min *et al.* (2019b) also find that some of the questions are not composed of two single-hop questions but require implicit multi-hop reasoning, hence cannot be decomposed. Secondly, for some questions, answer for each sub-question does not exist explicitly in the text and has to be inferred with commonsense reasoning.

4.2.3 Miscellaneous Techniques

In this section, we list some interesting approaches that do not fall into the above defined categories.

- **Question generation:** Li *et al.* (2023) propose multi-task training of a model to perform both multi-hop question answering and sub-questions generation. The motivation is that a model capable of asking good sub-questions corresponding to a complex question should perform well at MHQA. The system consists of a GNN based QA component and an LM based QG component that are trained together with a shared encoder.

- **Entailment based MHQA:** Equation 2.1 presents MHQA roughly as an entailment task, where the premise consists of multiple contexts and the hypothesis is "a answers q" (referred as H_{aq} henceforth). This indicates the utility of entailment models for the task. However, modelling MHQA as entailment faces three major challenges: a) A larger set of possible answers makes this method difficult to scale. b) MHQA requires aggregation of multiple contexts and single sentence based entailment models can not be used directly. c) Entailment models are usually not

trained to filter irrelevant information. Trivedi *et al.* (2019) aim to tackle the two latter challenges on Multiple choice datasets, OpenBookQA and MultiRC. Two simple baselines are proposed: *Concatenate* that concatenates all sentences as the input to an entailment task, and *Max of local decisions* that uses entailment on each sentence independently and then aggregate the results with a max operation. The two baselines perform poorly validating the challenges mentioned above. Therefore, a novel method 'Multee' comprising of a sentence relevance module and a multi-level aggregation is proposed. The sentence relevance module uses a pre-trained entailment model to produce hypothesis aware representation of each sentence which are passed to a Bi-LSTM layer to give contextual representation of each sentence which is fed to a feed forward layer to get the relevance scores α_i. The multi-level aggregation module pass each sentence along with the hypothesis to k ESIM (Chen *et al.*, 2017b) entailment stacks to generate k paragraph level vectors which are concatenated and passed to a feed forward layer to predict the entailment. Each entailment stack has m layers, l of these layers process each sentence independently and the outputs are aggregated by using α_i's. Rest of the layers process this aggregated output to result in paragraph level embeddings. The aggregation is also done in the cross attention layer to produce a cross attention matrix containing attention between each hypothesis token and each paragraph token. The sentence relevance weights are used here as well. Each entailment stack used is pre-trained on SNLI (Bowman *et al.*, 2015) and MultiNLI (Williams *et al.*, 2018).

- **Commonsense knowledge for MHQA:** Bauer *et al.* (2018) argue that a pre-trained model may not be able to capture every grounded common-sense knowledge even with large corpora and propose using ConceptNet (Speer *et al.*, 2016) for extracting the same and using it, form reasoning paths leading to the answer. A tree of various reasoning paths, all starting from the question concepts ($c_1's$) is formed by following four steps: a) Selecting relations $r_1's$ from ConceptNet that link c_1 to another concept

c_2 from the context. b) selecting relations $r_2's$ that link c_2 to another concept c_3 from the context. c) Selecting all neighbouring concepts c_4 of c_3 connected by r_3. d) Selecting relations r_4 that link c_4 to another concept c_5 from the context. This results in a large number of reasoning paths that are scored in two steps: a) $n - score$ is computed by term frequency of each of c_2, c_3, c_5 in the context C. c_4 is scored with its Point-wise Mutual Information (PMI) (Church and Hanks, 1990) with c_{1-3}. Each node's score is normalized across all its siblings in the tree using Softmax. b) $c - score$ The node scores are accumulated starting from the leaf node and updating each non-leaf node recursively.

$$c\_score(c_i) = \begin{cases} n\_score(c_i) & c_i \text{ is a leaf node} \\ n\_score(c_i) + \frac{c\_score(c'_{i+1}) + c\_score(c''_{i+1})}{2} & otherwise \end{cases} \tag{4.3}$$

where c'_{i+1} and c''_{i+1} are the top two scoring children of c_i. At each level of the tree, for every node, only the two top scoring children are maintained and the rest are pruned resulting in a total of 2^4 reasoning paths. Context representation is attended to the commonsense representations formed by embedding the tokens in each of the reasoning paths. Updated context representation is combined with the original one using a sigmoid gate. The gate incorporates the fact that the commonsense might be optional. Hence, the unit is called NOIC (Necessary and Optional Information Cell).

- **Reasoning over tables:** Chen *et al.* (2020b) use BERT to encode a cell in a table where a cell is represented by its value, position, hyperlinks etc. Encoding of a cell along with the encodings of its neighbouring cells is fed to a feed forward model and a Softmax layer to find the cell for the next hop. The model can also hop to the same cell. The cell value is prepended to the hyperlinked passage and passed to the answer prediction module.

- **Pointer network for reasoning chain prediction:** Chen *et al.* (2019) encode the question and obtain query dependent paragraph encoding by using BERT. Sentence encodings are extracted from

the paragraph encoding similar to Huang and Yang (2021). Alternative baselines are a) BERT-Sent that obtains query aware encoding of each sentence by using BERT and b) BiDAF-Para that uses BiDAF (Seo *et al.*, 2016) to encode paragraphs. Results show that BERT-para performs the best. An LSTM Pointer Network is trained to predict probability of each sentence being the $t-th$ sentence in the reasoning chain i.e., $P(s_i = rc_t)$. The ground truth reasoning chain is determined using a chain extractor model discussed in Section 4.4. The model is trained using both Negative Log Likelihood (NLL) and Reinforcement Learning (RL). However, RL does not improve the performance significantly. During test time, a beam of possible chains is maintained since the best chain may not be evident until it is entirely retrieved.

- **Bi-directional attention baseline:** Yang *et al.* (2018) propose a baseline by modifying the architecture of Song *et al.* (2018). Question and the concatenation of paragraphs is passed to an RNN for combining the character and word level embeddings. Bi-directional attention is applied across the question and context embeddings to get query aware context representations. A residual connection is added to the output of another RNN on these representations before passing to a self-attention layer.

4.3 Answer Prediction Module

After the reasoning module outputs a reasoning chain or some latent space representations of the reasoning, an answer prediction module predicts the final answer. This module can be directly categorized based on the type of answer required by the task at hand:

4.3.1 Candidate Answering

This category corresponds to the multi-choice setting of the MHQA task. Methods dealing with candidate answer prediction usually start the retrieval/reasoning process by using a candidate-aware embedding of the question, and output the representations for each candidate independently. We look at some of these methods below:

- A Bi-directional attention layer on query and context (Cao *et al.*, 2019), or a self-attention layer on the embeddings may also be employed.
- Some of the methods use these representations to perform an independent binary classification for each candidate (for multiple-correct type questions such as MultiRC) (Yadav *et al.*, 2019b; Yadav *et al.*, 2020; Yadav *et al.*, 2021; Trivedi *et al.*, 2019).
- On the other hand, some methods perform a multi-way classification for all candidates (for single-correct type questions such as OpenBookQA, WikiHop) (Yadav *et al.*, 2020; Yadav *et al.*, 2021; Cao *et al.*, 2019; De Cao *et al.*, 2019; Chen *et al.*, 2019).
- De Cao *et al.* (2019) use an ensemble of five models, each trained with a different weight initialization.
- Yadav *et al.* (2020) use both the answering methods for QASC and find that the multi-way classification results in an improvement of 5% accuracy.

4.3.2 Span Answering

For tasks requiring the answer as a text span from the contexts, the following methods have been proposed:

- Yang *et al.* (2018) suggest a span answering approach where the output of the reasoning module is passed to another RNN for supporting facts prediction. Another RNN is used to predict the start and end token of the answer span. Finally, a three-way classifier layer is used to predict the answer type among 'yes', 'no' and the extracted span. Many methods focusing on the first two units use this baseline as the answer prediction unit (Feldman and El-Yaniv, 2019; Das *et al.*, 2019; Shao *et al.*, 2021; Zhang *et al.*, 2020; Cao and Liu, 2021; Qiu *et al.*, 2019; Shao *et al.*, 2020).
- Qi *et al.* (2019) make two changes to the above baseline: a) Concatenating passages before encoding makes the representations depend on the order while concatenating. Thus, a shared RNN

encoder first encodes each passage independently and then concatenates the representations. b) All the attention layers are replaced by self-attention on the concatenated question and context representation.

- Ding *et al.* (2019) and Fang *et al.* (2020) replace the RNNs in the above baseline by MLPs and directly feed the node representations for answering. Xiong *et al.* (2019) deal only with bridge question and hence use a single MLP for span prediction on entities. Thayaparan *et al.* (2019) only deal with supporting facts prediction and use a single MLP for the same.

- Zhang *et al.* (2021) use an AlBERT model to predict the answer span among the K identified passages. The model then predicts if the answer was found in the given passage. If not, the predicted span is used for iterative retrieval for the next hop.

- Min *et al.* (2019b) use an off-the-shelf single-hop answering module to answer the sub-questions formed by the reasoner.

- Malon and Bai (2020) use a simple BERT model to answer the question from the paragraph identified as the answer paragraph. Dua *et al.* (2021) do the same from the concatenation of the two identified passages.

- Chen *et al.* (2019) concatenate question and all sentences of all the retrieved reasoning chains and feed to BERT to perform the 4 tasks.

- Huang and Yang (2021) propose to use the output of the GNN model. Sentence scores are computed using MLPs on the sentence node representations. Paragraph scores are computed using MLPs on max pooling over the sentence representations. The two scores are combined with the span extraction score to predict the final answer. Separate MLPs are used for predicting the answer type and the supporting facts.

- Sun *et al.* (2021) Concatenate the embeddings at all retrieved steps $\{k^0, \cdots, k^*, \cdots, k^0, \cdots, k^*\}$ and perform a weighted sum

to get $\mathcal{K}$ that is used to make the final prediction. The Softmax weight $\mathcal{Y}_j$ is computed across the retrieved sentences from all steps.

4.3.3 Generative Answering

Previously, generative answering models had a low prevalence because of the unsatisfactory performance of models to generate relevant texts as well as the lack of reliable evaluation techniques. As a representative of the earlier approaches, Bauer *et al.* (2018) use a self-attention layer followed by a PGN decoder to get the final answer.

However, with the recent advancements relating to large language models (LLMs), generative answering approaches have become a lot more popular. We describe these methods in detail in Chapter 5 designated to LLMs for MHQA.

4.4 Auxiliary Tasks

A lot of existing approaches suggest using an auxiliary training task that can help the training of the model by providing an extra signal. Here are two of the commonly used auxiliary tasks:

4.4.1 Reasoning Chain Prediction

Reasoning chains are an integral part of explainable MHQA. HotpotQA contains the supporting facts that are required to answer the question, however these are not ordered. Several works have aimed to predict the order among these supporting facts by using simple lexical heuristics (Jhamtani and Clark, 2020; Trivedi *et al.*, 2020; Wang *et al.*, 2019) so that the reasoning chain formed can be used to further train or evaluate a model. However, it is also crucial for the models to be able to predict the reasoning chains only by using the question and contexts.

- Feng *et al.* (2020) propose a semi supervised reinforcement learning for two modules to recover the reasoning chain in a *cooperative-game* approach. Apart from predicting the reasoning chain, the model is also able to predict the relations among the sentences.

The two modules are: a) Ranker module that, given k passages and a question, selects a reasoning chain. Question and passages are encoded by bi-GRU (Cho *et al.*, 2014) and a Match-LSTM (Wang and Jiang, 2016) model is used to get a probability for each passage containing a part of the reasoning chain. A passage is then sampled and used to update the question by passing to an MLP. The process is repeated with the updated question. The ranker module is rewarded for selecting the correct passage at the correct reasoning step. b) Reasoner module that predicts the entity from the current passage to the next passage. Given the first passage (called head) selected by the trained Ranker, the Reasoner uses a Math-LSTM model to predict the probability of each entity appearing in the second passage (called tail). While the ordering of the supporting facts is generated similar to Yang *et al.* (2018) for HotpotQA, the reasoning chains need to be manually annotated for MedHop. The manual annotation is done by extracting all valid paths such that the first sentence contains an entity in the second sentence and the second contains the answer. A chain is manually labelled as positive if the corresponding passage describes the drug-protein interaction. Reasoning chains extracted in this way may not be unique.

- Chen *et al.* (2019) derive pseudo gold chains using NER and co-reference resolution. A sentence (only) graph is constructed with edges between sentences that belong to the same paragraph and between those that have a shared entity. Reasoning chains are collected by starting from the question node and finding all possible paths in the graph that lead to an answer containing sentence. These chains are then ranked by two heuristics: a) shorter chains are given higher scores. b) chains whose sentences have a high F-1 ROUGE overlap with the question are given higher scores. Experimental results indicate that b) is a good criteria for scoring chains. Human evaluation shows that the reasoning chains produced are of similar quality compared to the supporting facts present in the HotpotQA. A pointer network is then trained to predict each sentence in the reasoning chains.

4.4.2 QA with Partial Knowledge

Khot *et al.* (2019) propose a new sub task of MHQA where the model has to figure out and fill the knowledge gap for question answering. It is assumed that the first hop retrieval has been performed ideally and the task is to use the retrieved context to answer the question. A modified version of OpenBookQA is released where the core fact is part of the input and several relations have been modified.

4.5 Conclusion

In this chapter, we covered a large number of pre-LLM machine learning methods proposed to solve MHQA. While doing so, we provided some structural patters that many methods follow. We also provided various levels of classifications of the methods, with technical details for some representative methods pf each category or subcategory. In chapter 6, we build on this classification to propose a taxonomy covering most of the works discussed in this chapter. In the next chapter, we provide details for how LLMs have been incorporated for different stages of the task along with some challenges and their proposed solutions.

5

LLMs for MHQA

Large Language Models (LLMs), including the multiple variants of GPT, BERT and T5 models, have achieved remarkable success in various natural language tasks. We present a comprehensive background of LLMs and various prompting techniques in the Appendix A.5. We also recommend Zhao *et al.* (2023b) as an additional reading for a comprehensive survey on LLMs.

Since language models like T5, BERT have been used for multiple tasks in most methods we have discussed so far, we use the following distinction for LLM-based methods: A method is LLM-based if it makes use of the emergent abilities of the LLMs (Zhao *et al.*, 2023b) which include in-context learning and instruction following[1]. We categorize LLM based methods based on the sub-task the LLM performs.

5.1 Retrieval

Nair *et al.* (2023) explore using LLMs for retrieving relevant evidences from long documents. Major challenge in using LLMs for retrieval is their limited context window size. The performance decreases drastically

[1]This distinction is made for the sole purpose of better structuring the discussion.

as the input context becomes larger than the context window size. To deal with this challenge, each section in the document is independently summarized by an LLM. These summaries are concatenated together in another LLM prompt, where the instruction is to get a list of relevant sections. Once a set of relevant sections is obtained, the next step is to retrieve the relevant paragraphs from each of these sections. For this, each paragraph is summarized and passed to the LLM to get the set of relevant paragraphs. A bart-large (Lewis *et al.*, 2019) trained over the CNN/Daily-Mail Corpus (Nallapati *et al.*, 2016) is used for summarization, and GPT-3.5 is used for retrieval and answering.

5.2 Reasoning Chain Generation

Existing works have extensively explored GPT prompting for various QA tasks. Although few-shot prompting on LLMs for reasoning showed limited success (Han *et al.*, 2022), chain of thought prompting (CoT) has shown better reasoning abilities. CoT is particularly useful for explainable MHQA since the generated chain of thought can be used as the reasoning chain (Zelikman *et al.*, 2022; Wang, 2021). Saparov and He, 2023 use synthetic data to evaluate GPT-3's reasoning ability and measure the coherence of generated reasoning chains. Results indicate that GPT-3 can perform multiple steps of reasoning but may rely on background knowledge rather than explicit reasoning over the given context.

Below, we list a few works and describe in detail how the LLM was used for generating reasoning chains.

- Rahgouy *et al.*, 2023 evaluate T5 and Flan-T5 (Chung *et al.*, 2022) models with and without fine-tuning on chain-of-thought explanations and in zero-shot, few-shot and CoT prompting settings on complex multi-hop queries. The results indicate the need for improvement in the existing methods.

- Zelikman *et al.*, 2022; Wang, 2021 get promising results by directly using the chain-of-though explanations as reasoning chains.

- PathFiD Yavuz *et al.*, 2022 was one of the first to use a large language model for modeling the task as a sequence generation

problem. However, the task was not modelled as a free-text natural language generation. A pre-trained T5 model was fine-tuned to generate a 'reasoning path'. A reasoning path is defined as a sequence of sentences from the context passage along with the corresponding passage title, followed by the answer to the question. The encoder independently encodes the 'blocks' (sentence and corresponding paragraph title) in the retrieved context and the decoder decodes the reasoning path by selecting the relevant blocks and deriving the answer.

- Haji *et al.* (2023) extend the idea of PathFiD by proposing the LLM to generate parallel exploratory inference chains (EICs). They follow a multi-step generation process, prompting an LLM multiple times with different prompts in the process. During the first step, a LLM is prompted to generate some keywords from the question and the paragraph titles. These keywords are then processed independently in parallel. A keyword is matched lexically with available paragraph titles to retrieve the paragraphs for the next step, and the LLM is prompted with the retrieved passage to predict some structured facts about the keyword. The object in these generated fact is used as a keyword for the next reasoning step. Two independent LLMs are prompted to aggregate the facts for each keyword and to aggregate facts across keywords to generate a final answer. Impressively, the resulting system is able to significantly outperform a single chain-of-thought prompt.

- Trivedi *et al.*, 2023 propose IRCoT which uses interleaving-retrieval with CoT reasoning to iteratively perform retrieval and reasoning together. For each iteration, 1) the CoT-guided retrieval step ("Retrieve") uses the last generated CoT sentence as a query to retrieve more paragraphs and adds them to the accumulating set of the collected paragraphs, and 2)the retrieval-guided reasoning step generates the next sentence of the CoT using the previously generated sentences along with the retrieved contexts. Finally, another LLM derives the answer using the generated CoT and the retrieved paragraphs.

5.3 Hybrid Text-table Reasoning

The following methods for utilizing LLMs for solving hybrid MHQA have been proposed:

- Lei *et al.* (2023) were one of the first to use LLMs for solving table-text hybrid MHQA. They train a sequence-to-sequence LLM similar to Yavuz *et al.* (2022), to encode the retrieved contexts and question. The decoder, however, performs auto-regressive generation to predict the answer sentence. They also compare a prompting based approach where the retrieved contexts are provided to GPT-3.5 along with few-shot and CoT prompts. The results showed promising performance of CoT but still the fine-tuned model outperformed all the prompting techniques.
- Chen *et al.* (2021) propose converting a table into text using template based transformation and then use a RoBERTa to retrieve sentences from the combined set of original text and converted text.
- Mavi *et al.* (2023) test the GPT-3.5 and Llama2's ability to parse tabular data by representing a table in text by using separators such as | and || to separate cells in a row and multiple rows respectively and observe promising performance on tabular reasoning.
- Shi *et al.* (2024) explore using program generation and execution based framework for hybrid MHQA and get promising results.

5.4 Question Decomposition

Since GPT has shown remarkable language understanding, motivating its use as a decomposition module for decomposition based MHQA (Khot *et al.*, 2023; Zhou *et al.*, 2023).

- Zhou *et al.* (2022) explore large-scale intermediate pre-training of a T5 model to perform question decomposition. For constructing a large enough dataset for pre-training, distant supervision from comparable texts is used. In particular, parallel news articles describing the same events are used. The hypothesis is that seeing

multiple descriptions of the same facts will help the model make better educated guesses. The proposed model, called DecompEntail, first generates explicit question decomposition, then makes factual corrections on the decomposed statements with GPT-3. As a final step, an entailment model is used to derive the final answer with the generated decomposition as the premise and the question and candidate answer as the hypothesis.

- Deng *et al.* (2022) perform question decomposition in the semantic space by using an AMR graph. They build an AMR graph from the question and add an additional 'amr-unknown' node denoting a concept that represents the answer. The AMR is segmented into sub-graphs using some heuristics and each sub-graph is passed to a pre-trained BART model that converts it into a sub-question. Finally, a single-hop QA model is used for answering the sub-questions.

- Wu *et al.* (2024) fine-tune a sequence-to-sequence LLMs to generate sub-questions using the given multi-hop question. A set of retrieved context is also given as input during this step. The sub-questions are used for performing a second round of retrieval using a DeBERTa model (He *et al.*, 2021). Another DeBERTa model is used on the concatenation of the retrieved paragraphs and sub-questions to generate the final answer.

5.5 Graph Construction

Li and Du (2023) prompt an LLM with a Wikipedia passage to build a semantic graph. This is done by prompting the LLM to extract all the entities in the document and then prompt it again to get all the relations among these entities. Finally, each passage and the corresponding semantic graphs are concatenated into a prompt for generating reasoning chain and the final answer.

5.6 Multi-hop Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is the term given to the general framework where a generative model (usually a LLM) performs generation by using a set of facts extracted by a retriever. In RAG, an external corpus containing multiple documents serves as the knowledge base. Each document within this corpus is segmented into a set of chunks. These chunks are then transformed into vector representations using an embedding model and stored in an embedding database. Given a user query, the system typically retrieves the top-K chunks that best match the query. The retrieved chunks, combined with the query and an optional prompt, are then fed into an LLM to generate a final answer.

Joshi *et al.*, 2023a build a novel dataset for the development and benchmarking of multi-hop RAG models. The dataset contains a knowledge base for retrieval, a set of multi-hop queries along with their ground truth answers and corresponding supporting facts. The knowledge base is built from a set of news articles. GPT-4 is prompted to extract factual sentences from the news articles

5.7 Critique and Limitations

Despite the impressive linguistic and reasoning abilities of LLMs, they face certain limitations. In this section, we list these limitations and some attempts at resolving these.

5.7.1 Hallucinations

The biggest challenge that the LLMs currently face is that they are known to hallucinate at times, i.e., generating text that is factually incorrect or not derivable from the given information. This poses a threat to accountability of the resulting MHQA system and can produce incorrect results. A large number of works have tried to propose sophisticated and clever ways of overcoming this challenge. We describe few such interesting works in detail below:

- **Self-consistency:** Self-consistency (Wang *et al.*, 2023) is proposed as an alternative for the naive greedy decoding used in chain-of-

thought prompting. Instead of only taking the greedy decoded reasoning path, a diverse set of reasoning paths are sampled, and then the most consistent answer across the reasoning paths is predicted by marginalizing out the sampled reasoning paths. Self-consistency is motivated by the intuition that a complex reasoning problem typically admits multiple different ways of thinking leading to its unique correct answer. Assuming that the hallucinations in the LLM output are random, self-consistency is more likely to lead to the correct answer since it is unlikely that majority of the reasoning paths with hallucinations lead to a common answer. Therefore, self-consistency has been adopted to mitigate hallucinations.

- **Self-refine:** In self-refine (Madaan *et al.*, 2023), the LLM generates an initial response to the prompt, and then iteratively provides feedback for the output and refines it. The same LLM serves as the generator, feedback provider as well as the refiner. During self-refine, the LLM can also evaluate the response on factual correctness, thus fixing the hallucinations.

- Joshi *et al.*, 2023b extend self-consistency and self-refine by fine-tuning external LLMs for providing feedback on the generated reasoning chains. The authors collect labelled data for correctness, error types and descriptions, and the required corrections for each reasoning chain and answer pairs for 2361 examples. They use a weighted self-consistency approach where a fine-tuned Llama2 predicts a score for whether each answer and its corresponding reasoning path are correct or not. This score is used as the weight while voting for the correct answer. They also explore extending self-refine by using a fine-tuned Llama2 model to identify the errors in a reasoning path and refining the generated output.

- Zhao *et al.*, 2023a propose 'Verify and Edit' that combines self-consistency and self-refine. It follows a three-step process: finding uncertain predictions using self-consistency, editing their rationales by searching for supporting facts, and using the edited rationales to generate final answers. An answer is considered as uncertain, if

none of the generated answers gets the majority agreement. For each of the predictions corresponding to the uncertain answer, each sentence in the reasoning chain is verified by generating corresponding questions. These questions are answered after retrieval and the answer is used to update the particular sentence in the original reasoning chain. Finally, the updated reasoning chain is used to generate the final corrected answer.

- Balepur *et al.*, 2023 propose a claim decomposition based technique for self-evaluation of the LLM generated answers. The complex question is broken down into a set of claims that a correct answer to the question must satisfy. The LLM then identifies which claims mention the same entities and include extra tags for them. For evaluating a generated answer, the answer is replaced in each of the claims and the LLM is prompted to answer true/false based on whether the answer satisfies the claim or not. The answer is then scored on the ratio of claims satisfied by it. Multiple answers are predicted by sampling based generation of the reasoning chains and the one with the highest score is predicted.

5.7.2 Knowledge Gaps

Feng *et al.*, 2024b argue that the challenge of hallucinations is difficult to overcome since the LLMs might always suffer from knowledge gaps. A knowledge gap here refers to some missing or outdated information. Since LLMs heavily rely on knowledge gained during the pre-training phase, outdated or incomplete information in the pre-training corpus can confuse the model, leading to generating factually incorrect text. Further, pre-training LLMs is extremely expensive in terms of time and compute. It is therefore ideal that the LLM is able to detect a knowledge gap and refrain from generating information in such a case. For identifying knowledge gaps, the following approaches can be followed:

- **Calibration-based:** getting a probability score for an answer by either using token probabilities and temperature during generation (Radford *et al.*, 2019; Liang *et al.*, 2023) or directly asking the

model to generate its confidence in its output (Tian *et al.*, 2023).

- **Training based:** training an extra layer (Slobodkin *et al.*, 2023), or an external module (Cobbe *et al.*, 2021) to predict whether the answer produced has a high enough confidence. Another interesting way is to use identifying knowledge gaps as one of the instruction tuning tasks (Ouyang *et al.*, 2022).
- **Prompting based:** techniques such as self-reflect (Kadavath *et al.*, 2022), self-consistency (Wang *et al.*, 2023) predicting none-of-the-above (Kadavath *et al.*, 2022) or more information needed (Feng *et al.*, 2024a) instead of predicting the answer.
- **Multi-LLM collaboration:** using two independently trained LLMs for either providing feedback to each other, or complementing each other to fill the knowledge gaps in both (Feng *et al.*, 2024b).

6

MHQA Taxonomy

This chapter builds on the distinctions and classifications drawn in Chapter 4 and our taxonomy that covers the existing methods in a structured way. Creating a taxonomy of existing methods for multi-hop QA is key for sorting through research and making comparisons. It also helps spotting where the community needs to pay more attention. We summarize these in Table 6. The titles and acronyms used in the table are described below, and illustrated in Figure 6.1.

- **General:**
  - **Datasets used**[1]**:** The datasets used for MHQA along with their acronyms in Table 6 are: HotpotQA (Yang *et al.*, 2018) in the full-wiki (**HP-F**) and distractor (**HP-D**) settings. Sun *et al.* (2021) provide the modified versions of the ShARC (Saeidi *et al.*, 2018) and the HotpotQA datasets as ShARC-Long (**Sh-L**) and HotpotQA-Long (**HP-L**). Other datasets are referenced as: MultiRC (Khashabi *et al.*, 2018) (**MRC**),

[1]Note that we do not include the datasets used in the proposed Taxonomy, but describe it in this section for the sake of completionstill add it in the Table 6 for the sake of completeness.

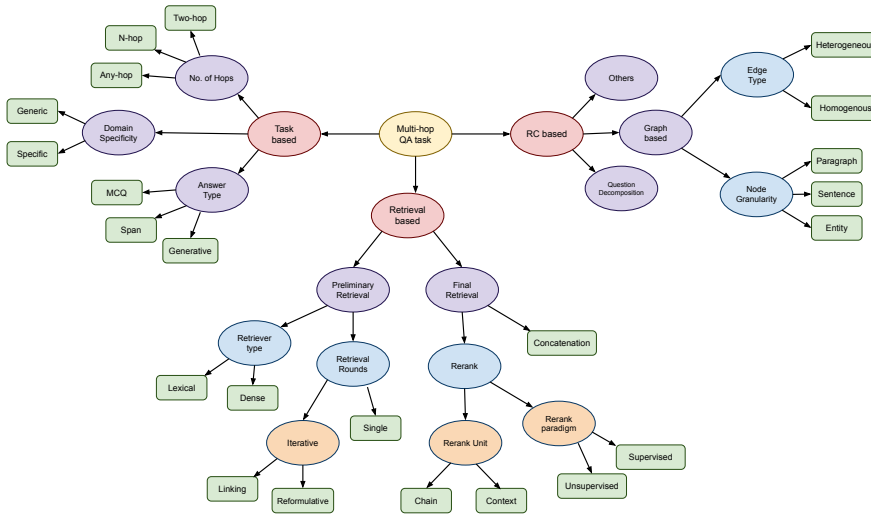


Figure 6.1: Overview of Our Taxonomy.

ARC (Clark *et al.*, 2018) (**ARC**), QAngaroo datasets (Welbl *et al.*, 2018) WikiHop (**WH**) and MedHop (**MH**), HybridQA (Chen *et al.*, 2020b) (**Hy**), QASPER (Dasigi *et al.*, 2021) (**QSP**), OpenBookQA (Mihaylov *et al.*, 2018) (**OB**), QASC(Khot *et al.*, 2020) (**QSC**) and **Nr** (Kočiskỳ *et al.*, 2018).

- **Hop Constraints:** Some of the methods require the type of questions to be exactly two-hop questions (**Two**) or require the number of hops in the question to be provided as an input (**N**). Other methods are flexible for answering any-hop questions without any additional input (**Any**).
- **Answer types:** Whether the methods proposed work for MCQ type questions (**MCQ**), Span based questions (**Span**) and generative answers (**Gen**). Note that some methods focus on more than one of these answer types.
- **Domain Specificity:** Whether the works focus on domain specific (**Specific**) or domain generic techniques (**Generic**).

- **Retrieval:**
  - **Retriever Type:** Whether the preliminary retrieval step

uses a dense (**Dense**) or a lexical (**Lexical**) retriever.

- **No. of Retrieval Passes:** Whether the preliminary step is single pass (**Single**) or iterative (**Iter**). In case it is iterative, whether it uses query reformulation (**Iter-QR**) or Entity-links/Hyperlinks (**Iter-EL/H**).
- **Final Retrieval Strategy** Whether the final retrieval step uses a simple concatenation/union (**Concat**) of the contexts or it is uses a re-ranking approach (**RR**). In case, it uses re-ranking, whether the re-ranking is supervised (**RR-S-?**) or unsupervised (**RR-U-?**), and whether the individual contexts (**RR-?-X**) or the entire chain candidates (**RR-?-C**) are scored during the re-ranking.

- **RC based:**
  - **Node Granularity:** In case the reasoner is a graph based model, what kinds of nodes it has: entity nodes (**Ent**), sentence nodes (**Snt**) or passage nodes (**Psg**). Note that the graph may have more than one type of nodes. (**-**) in case the reasoner is not graph based.
  - **Relational Edge:** Whether the graph is a relational graph (**Yes**) (i.e., has multiple types of edges) or not (**No**) (i.e., has single edge type). (-) in case the reasoner is not graph based.
  - **Question Decomposition:** Whether the reasoning module uses question decomposition (**QuesD**) or not (**Other**). The rows where both graph-based and decomposition-based columns are (**-**) are for the models that use other techniques like (Seo *et al.*, 2016).

By proposing the taxonomy, we hope to help the readers dive into these techniques, see what suits best to their needs or what needs more attention in the existing setup. This should help the research community keep pushing the boundaries for solving MHQA.

Table 6.1: Comprehensive study of existing work using the proposed taxonomy.

	General				Retrieval			Reasoning		
	Datasets	Hop Constraints	Answer type	Domain Specificity	Preliminary		Final	Graph based		Ques Decomp
					Type	Technique	Technique	Node types	Relational graph?	
Ding *et al.* (2019)	HP-F	N	Span	Generic	Lexical	Iter-EL/H	Concat	Ent	Yes	Other
Feldman and El-Yaniv (2019)	HP-F	N	Span	Generic	Dense	Iter-QR	RR-S-X	-	-	Other
Das *et al.* (2019)	HP-F	Two	MCQ	Generic	Lexical	Iter-EL/H	RR-S-C	-	-	Other
Yadav *et al.* (2019b)	ARC, MRC	Any	MCQ	Generic	Lexical	Single	RR-U-C	-	-	Other
Xiong *et al.* (2019)	HP-F	Two	Span	Specific	Lexical	Iter-EL/H	RR-S-X	-	-	Other
Fang *et al.* (2020)	HP-F	Two	MCQ	Generic	Lexical	Iter-EL/H	RR-S-C	Ent-Snt-Psg	No	Other
Yadav *et al.* (2020)	MRC, QSC	Any	MCQ	Specific	Lexical	Iter-QR	Concat	-	-	Other
Zhang *et al.* (2021)	HP-F	Any	Span	Generic	Lexical	Iter-QR	RR-S-X	Psg	Yes	Other
Yadav *et al.* (2021)	MRC, QSC	Two	MCQ	Generic	Dense	Iter-QR	RR-S-C	-	-	Other
Shao *et al.* (2021)	HP-F	Two	Span	Generic	Lexical	Iter-EL/H	RR-S-X	-	-	Other
Sidiropoulos *et al.* (2021a)	HP-F	Two	Span	Generic	Dense	Single	RR-S-X	-	-	Other
Cao *et al.* (2019)	WH	N	Span	Generic	-	-	Concat	Ent-Snt-Psg	No	Other
Thayaparan *et al.* (2019)	HP-D	N	Span	Generic	-	-	Concat	Snt-Psg	No	Other
Min *et al.* (2019b)	HP-D	Two	Span	Generic	-	-	Concat	-	-	QuesD
De Cao *et al.* (2019)	WH	N	Span	Generic	-	-	Concat	Ent	No	Other
Zhang *et al.* (2020)	HP-D	N	Span	Generic	-	-	RR-S-X	Ent-Snt	No	Other
Malon and Bai (2020)	HP-D	Any	Span	Generic	Lexical	Iter-QR	RR-S-X	-	-	QuesD
Feng *et al.* (2020)	MH, HP-D	N	Span	Specific	Dense	Single	RR-S-C	-	-	Other
Chen *et al.* (2019)	WH, HP-D	N	Span	Generic	-	-	Concat	Snt	No	Other
Huang and Yang (2021)	HP-D	Any	Span	Generic	-	-	Concat	Snt	No	Other
Cao and Liu (2021)	HP-D	Any	Span	Generic	-	-	Concat	-	-	QuesD
Sun *et al.* (2021)	HP-L, Sh-L QSP, Hy	N	Span	Specific	-	-	Concat	-	-	QuesD
Dua *et al.* (2021)	WH, HP-D	Two	Span	Generic	-	-	RR-S-C	-	-	Other
Qi *et al.* (2019)	HP-D	N	Span	Generic	Lexical	Iter-QR	Concat	-	-	Other
Qiu *et al.* (2019)	HP-D	N	Span	Generic	-	-	RR-S-X	Ent	No	Other
Shao *et al.* (2020)	HP-D	N	Span	Generic	-	-	RR-S-X	-	-	Other
Bauer *et al.* (2018)	Nr	N	Gen	Specific	-	-	RR-S-X	Ent	Yes	Other
Trivedi *et al.* (2019)	MRC, OB	Any	MCQ	Generic	-	-	Concat	-	-	Other
Khot *et al.* (2019)	OB	-	MCQ	Specific	Lexical	Single	Concat	-	-	Other

7

How to Evaluate MHQA Systems?

Having evaluation metrics suited to a task is crucial for grasping the task's nuances and gauging the performance of the existing or proposed methods accurately. It ensures researchers assess systems effectively and compare results meaningfully. Future research should benefit clear metrics guide development, revealing where improvements are needed and facilitating advancements in techniques tailored to the intricacies of the task. In this chapter, we look at some commonly used evaluation metrics and potential limitations of using them. We follow our discussion with various sophisticated and clever experiments conducted by the community which shed further light on the intricacies of how the existing systems perform and suggest some directions where further research and development is required.

7.1 Evaluation Metrics

Diversity in multi-hop QA tasks and datasets engenders the need for different evaluation metrics. The general trend in multi-hop QA methods is to split the task into a retrieval (IR) component that finds the relevant contexts, and a reading (RC) component that produces the answer. Therefore, it makes sense to evaluate the two together as well

as separately. We describe the methods used for the two evaluations below:

7.1.1 Retrieval Evaluation

As mentioned in Chapter 4, retrieval is often broken down into two steps. Different metrics can be used to evaluate the two steps:

Preliminary retrieval: Ye *et al.* (2019) propose three evaluation metrics: **P EM** (Paragraph exact match) that measures the ability of the retriever to retrieve all the gold paragraphs in the reasoning chain; **PR** (Paragraph recall) that computes the recall of gold paragraphs among the retrieved paragraphs; and **AR** (answer recall) that checks if any of the retrieved paragraph contains the answer. Das *et al.* (2019) propose **acc@**k which measures the fraction of cases where the model was able to retrieve *all* the supporting facts within top k retrieved documents. Sidiropoulos *et al.* (2021a) define the per-hop retrieval evaluation that treats each hop of retrieval independently. First hop retrieval performance is measured by fraction of cases where the first gold context is retrieved in the first hop of retrieval. For the second hop, the gold paragraph is added to the set of contexts retrieved in the first hop and the ability to retrieve the second hop gold paragraph is evaluated. Since the paper only deals with 2-hop questions, the definition is originally limited to 2 hops. However, we note that this might be extended to n hops where the n^{th} retrieval step is evaluated by the ability to retrieve the n^{th} gold paragraph p_n given $\cup_{i=1}^{n-1}(\{p_i\} \cup R_i)$ where R_i is the set of contexts retrieved during i^{th} hop.

Re-ranking: Jhamtani and Clark (2020) require the models to classify or rank several reasoning chains as valid explanations of the answer. Therefore, they use **AUC-ROC** (Melo, 2013) (area under the Receiver Operating Characteristics (ROC) curve) and **F1** scores for classification, and **P@1** and **Normalized Discounted Cumulative Gain (NDCG)** (Järvelin and Kekäläinen, 2002) for ranking. P@1 measures the fraction of cases where the top ranked chain is valid, whereas NDCG is a commonly used metric for evaluating rankings. **MRR** (Järvelin and

Kekäläinen, 2002) (Mean Reciprocal Rank) is another ranking based metric used for evaluating ranking of reasoning chains (Jansen, 2018; Kočiskỳ *et al.*, 2018). Das *et al.* (2019) also use Mean Average Precision (MAP) (Liu and Özsu, 2009) that considers the relative position of the relevant document in the ranked list (Kadlec *et al.*, 2017).

7.1.2 Answer Evaluation

Based on the type of the answer, the following evaluation metrics have been used for large scale evaluation for the task:

Multi-choice questions: MCQ questions are straight-forward to evaluate when there is a single correct answer, and classification accuracy over the answer choices is used. To deal with multiple correct answer candidates, Khashabi *et al.* (2018) propose **F1a** and **F1m**. Precision and Recall are computed by evaluating each predicted answer candidate. Macro harmonic mean of average precision and average recall is the F1m score. F1a uses micro harmonic mean of precision and recall values for all answer candidates.

Span based answers: The most commonly adopted metrics are **Exact Match (EM)** and **F1** scores on the predicted string tokens. Tang *et al.* (2021) argue that EM can often be too strict and propose an alternative **Partial Match (PM)** where a predicted answer a_p is said to be a partial match with the ground truth answer a_g if either (a) $F1(a_p, a_g) > 0.8$ or (b) $F1(a_p, a_g) > 0.6$ and one of a_p, a_g is a substring of the other. These evaluation metrics are known to work well when the answer spans are small (< 10 tokens).

Auxiliary task evaluation: supporting facts: Yang *et al.* (2018) propose to evaluate the reasoning chains by reporting EM and F1 on the supporting facts (note that this is different from reasoning chain as supporting facts are sentences and the contexts are passages). They also propose **Joint-EM** and **Joint-F1** where the precision is defined as $P_{joint} = P_{ans} \cdot P_{sup}$ and the recall as $R_{joint} = R_{ans} \cdot R_{sup}$. Qiu *et al.* (2019) compute the score of a reasoning path in the entity graph by

multiplying the corresponding soft masks and attention scores along the path and selecting the top-k scoring paths. If any entity in a supporting fact is reached by any of the k paths, that fact is said to be a hit. Entity-level Supporting fact Prediction (ESP) scores are reported as Exact Match (EM) and Recall values over these supporting facts.

Generative answers: For longer sequences of texts, directly matching strings to give a binary score fails to tell which answers are closer to the gold answers. Thus, natural language generation (NLG) evaluation metrics are required. Kočiskỳ *et al.* (2018) propose using **Bleu-1, Bleu-4, Meteor** (Papineni *et al.*, 2002; Banerjee and Lavie, 2005) and **ROUGE-L** (Lin, 2004) to evaluate predictions on their dataset. Bauer *et al.* (2018) also use **CIDer** (Vedantam *et al.*, 2014) for evaluating long answers on NarrativeQA which emphasizes on annotator consensus.

The above discussed evaluation metrics serve the purpose of providing representative scores for large scale evaluation of methods and facilitating comparison among different techniques. However, multi-hop QA is a complex task and further evaluation experiments designed particularly for the task might be needed to capture nuances of the model. We discuss these in the next section.

7.2 Adversarial Evaluation

Adversarial evaluation is a commonly used evaluation technique for various types of models. It involves testing the models against intentionally crafted difficult examples to assess their robustness and expose potential weaknesses or vulnerabilities.

As indicated in Section 3.1, the particular choice of available contexts (C) for a question in the dataset can lead to 'reasoning shortcuts' where a model can correctly answer the question by using only a single context. To avoid such shortcuts in the HotpotQA's distractor setting, the authors used TF-IDF for retrieving confusing contexts. Min *et al.* (2019b) collect a different set of distractor paragraphs for the HotpotQA dataset, to evaluate if the models are robust to this change. Same strategy as Yang *et al.* (2018) is used while making sure that there is no overlapping distractor paragraph with the original set. An adversarial

set of comparison questions is also created by altering the original question so that the correct answer is inverted (for instance, replacing 'which is higher' by 'which is lower').

Tu *et al.* (2020) use a clever technique to add fake distractors that can fool a model which uses single hop reasoning shortcuts to answer the questions. A word in the final answer is replaced by another word having a similar GloVe embedding to create a fake answer. For instance, 'Mumbai' is replaced by 'Delhi'. All occurrences of the word are replaced in the answer passage to get a confusing distractor passage. Since the bridge entity is mentioned in the title of the answer passage, all mentions of a word from the title are also replaced with a similar entity. This is done to break the connection between the fake distractor and the first gold context. This ensures that there is only a single reasoning chain and that a model cannot answer the question by only looking at the answer's paragraph. Evaluation using the adversarial distractors shows a significant drop in the accuracy of a baseline model. Moreover, training using adversarial distractors leads to better performance on the original distractors as well. Therefore, more confusing distractors would lead to better training as well as testing of the MHQA models.

7.3 Verifying the Extent of Multi-Hop Reasoning

Despite the improved scores of models indicated by multiple evaluation metrics on various datasets, it remains doubtful whether the models are actually performing the multi-hop reasoning and following the expected reasoning path for reaching the correct answer. Therefore, different evaluation techniques and modifications of existing datasets are proposed as benchmarks for testing the multi-hop reasoning capabilities of a model. In this section, we list various experiments and benchmarks proposed for the same along with their outcomes and conclusions:

- In order to evaluate the interpretability of a model, Ding *et al.* (2019) define the *Logical rigor* of a model as Joint EM/Ans EM. Intuitively, it tries to measure among the questions that were answered correctly, what fraction also had correct supporting facts prediction. Surprisingly, baselines have scores of only 30.3%

and 7.9%.

- Wang *et al.* (2019) modify and further annotate HotpotQA to provide three settings, where the models are provided (1) only the passage containing the answer, (2) both supporting passages in random order and (3) both supporting passages in the order of their occurrence in the reasoning chain, with the intuition that a model that employs multi-step reasoning to answer multi-hop questions should benefit from the supporting passages whereas a model that tries to guess the answer directly would instead be confused by the extra information given. Two common techniques, BERT and HotpotReader were tested after employing both query-reformulating and co-matching approaches (see Section 6 on taxonomy). It was observed that the models could gain very little performance ($\sim 1\%$ and 4% accuracy with query reformulation and co-matching respectively) by using the reasoning chains provided. This highlights the inability of the existing techniques to incorporate multi-hop reasoning to perform MHQA. Further, it is found that BERT and co-matching show slightly higher improvements than their respective counter-parts.

- Tang *et al.* (2021) use BERT and DecompRC (Min *et al.*, 2019b) to generate single hop sub-questions comprising the 2-hop questions in the HotpotQA dataset and the answers to these questions. The claim is that if a model employs multi-hop reasoning to answer a question, it should trivially be able to answer the individual sub-questions. Surprisingly, for $\sim 50 - 60\%$ of the questions correctly answered, at least 1 of their corresponding sub-questions could not be answered correctly. Further, of the questions where both the sub-questions were answered correctly, $\sim 10\%$ were incorrectly answered. This indicates that the models tend to jump directly to the answer instead of breaking down the questions into simpler questions.

- Jhamtani and Clark (2020) propose three modifications of the QASC dataset that require the model to explicitly predict the reasoning chains along with the final answers (explainable MHQA).

i) eQASC: For each question in QASC, up to 10 candidate reasoning chains are automatically generated and each candidate chain is annotated to be valid (if the chain can imply the answer) or invalid (otherwise). ii) eQASC-perturbed: In the candidate chains of QASC, one word/phrase that is likely to be a bridge entity among two facts, is replaced by a similar meaning word ensuring that the chain remains to be valid. This is done by crowd sourcing where workers were asked to replace one occurrence of the word that appears in different sentences of a candidate chain. iii) eOBQA: a small number of questions in OpenBookQA are used for generating candidate reasoning chains using sentences present in QASC and are annotated by crowd-sourcing. This is done to test the generalization of the model on an unseen dataset.

- Trivedi *et al.* (2020) use the term Disconnected Reasoning (DiRe) for when the model is able to arrive at the correct answer using (possibly multiple independent) incomplete reasoning chains. To measure disconnected reasoning, a DiRe probe is created that checks if the output of $h(q, C \setminus \{p_1\})$ and $h(q, C \setminus \{p_2\})$ (refer to Chapter 2 for notations) can be trivially combined to answer q, C (where '\' denotes set difference). To discourage disconnected reasoning, the dataset is modified to include negative samples where the given C is not sufficient to answer the question i.e., $C \cap P_q \neq \phi$ and the model is required to identify these questions as unanswerable. When running the DiRe probe, disconnected reasoning is found to be reduced significantly after training using this modification.

- Inoue *et al.* (2020) argue that requiring the model to only output the supporting facts might not be enough to ensure the explainability of the model and the model should be required to also output the derivation steps. A derivation step is formalized as a triplet of the form $\langle d^h, d^r, d^t \rangle$, where d^h, d^t are entities (noun phrases), and d^r is a verb phrase representing a relationship between the two entities. A small subset of the HotpotQA dataset is annotated by crowd-sourcing and released. Evaluating models on this set indicates the scope of improvement on this benchmark.

Results of these works are significant as they suggest that improved accuracy on existing datasets may not correlate well with the models' ability to perform multi-hop reasoning. Furthermore, they highlight the inefficacy of existing models to perform multi-hop reasoning as well as the inefficacy of the datasets to evaluate the same. This implies a need for more carefully created datasets and challenging benchmarks that do not allow the models to score well without accurately following the required reasoning paths. Additionally, it is encouraged to formulate better and more such tests/probes that check and prevent the models from using loop-holes instead of doing multi-hop reasoning. Above all, it is fair to say that the task of MHQA is far from solved.

8

Multi-Hop Question Generation

The task of Multi-Hop Question Generation (MHQG) is very closely related to MHQA and shares some of the required reasoning and natural language understanding abilities. The task has many applications and has also received a growing attention in the recent years. Therefore, we add a brief discussion of MHQG in this chapter.

The goal is to generate a multi-hop question given a set of contexts and optionally, an answer. In chapter 3, we already discussed the challenges of question generation while creating MHQA datasets. A lot of those challenges directly apply to the task of MHQG as well. MHQG has widespread applications in multiple domains including education, where generating questions that require multiple steps of reasoning can be very useful for inspiring critical thinking in students (Lindberg *et al.*, 2013). QG also has a direct application for chat-bots e.g., in initiating conversations, asking and providing detailed information to the user by considering multiple sources of information. MHQG will enhance the ability of these chat-bots to ask useful questions (Yao *et al.*, 2018). It can also combine with question answering (QA) models as dual tasks to boost QA systems with reasoning ability (Tang *et al.*, 2017).

The task of traditional question generation (QG) has gained a lot

of interest recently (Du *et al.*, 2017; Zhao *et al.*, 2018b; Scialom *et al.*, 2019). However, MHQG is a more challenging task than simple QG. It requires the model to first identify scattered pieces of information that can be aptly combined to form a valid reasoning path from the answer to the question, and then reason over these pieces of information to generate a factual and coherent question.

8.1 Datasets

The datasets for MHQA can also be used to train and evaluate MHQG models by modifying the input and the required output of the model (Su *et al.*, 2020; Gupta *et al.*, 2020). Since HotpotQA has annotated supporting facts, it is able to provide stronger training supervision and hence, it is the most commonly used dataset for MHQG. Kumar *et al.* (2019) use the DecompRC model (Min *et al.*, 2019b) to decompose each question in HotpotQA into two sub questions and fine tune a GPT2-small model to rewrite the first question into the second. Yu *et al.* (2020) use HotpotQA as the labelled dataset and ComplexWebQuestions (Talmor and Berant, 2018) and DROP (Dua *et al.*, 2019) as large corpora for multi-hop questions.

8.2 Evaluation

Language generation metrics such as BLEU (BLEU1-4), ROUGE-L, METEOR are usually adopted for MHQG (Kumar *et al.*, 2019; Su *et al.*, 2020; Gupta *et al.*, 2020; Sachan *et al.*, 2020; Yu *et al.*, 2020). QBLEU4 (Nema and Khapra, 2018), a QG metric which was shown to correlate significantly better with human judgements, is also used for evaluating MHQG (Kumar *et al.*, 2019; Su *et al.*, 2020). The task also often requires human evaluation of fluency, semantics, answerability etc of the generated questions. Sachan *et al.* (2020) also use GLEU (Wu *et al.*, 2016) for their experiments. Another method of evaluating MHQG is to measure the gain in performance of SOTA MHQA models when trained with data augmentation using the generated questions (Kumar *et al.*, 2019; Pan *et al.*, 2021).

8.3 Methods

Kumar *et al.* (2019) tackle the problem of difficulty controllable question generation (DQG)(Gao *et al.*, 2018) by generating questions which require a particular number of reasoning hops. The assumption is that the difficulty of a question directly correlates to the number of inference steps required to answer it. The first step of the proposed algorithm builds a context graph in the same way as (Fan *et al.*, 2019). All sentences from the context are converted into triplets of the form $\{s(subject), r(relation), o(object)\}$ and a relational edge of type r is added from s to o. Co-reference resolution is used to merge the nodes referring to the same entity. Next, a node N_0 is sampled as the final answer and with N_0 as the root, a maximum spanning tree is extracted. For generating a question with difficulty (same as the number of hops) $= d$, the tree is pruned to have $d + 1$ nodes. A GPT2-small model is fine-tuned on HotpotQA and used to generate an initial question q_0 using N_0, N_1 and the context sentence connecting the two nodes S_1. Another GPT-2 model is used to rewrite the question iteratively and successively increase the difficulty. In simpler words, for generating a question q_d with difficulty d, the re-writer model is run on q_0 for d iterations. While the model performs well for up-to 3 hops, the input to the re-writer model becomes too large for subsequent hops and results in questions with poor quality.

Su *et al.* (2020) leverage a graph constructed similar to Qiu *et al.* (2019) to generate multi-hop questions. Pre-trained GloVe embeddings and answer tagging embeddings (Lindberg *et al.*, 2013) are passed to two Bi-LSTM layers to get the initial contextual representations. An attention layer and another Bi-LSTM layer is used to get the answer aware context embeddings. An answer aware sub-graph is computed by masking the entities that are irrelevant to the answer and Graph Attention Network is applied to this sub-graph. The context encodings across the hops are combined via a gated fusion module. The Answer embeddings are updated using bi-attention and The Maxout Pointer (Zhao *et al.*, 2018b) framework is used on top of a Uni-directional LSTM decoder to generate the question. An additional BFS loss proposed by (Qiu *et al.*, 2019) is found to improve the performance.

Gupta *et al.* (2020) exploit the presence of supporting facts in HotpotQA while training by adopting an RL reward of the auxiliary task of supporting facts prediction (SFP). Similar to Su *et al.* (2020), answer tagging features are concatenated with the document word embeddings and fed to a Bi-LSTM encoder. The output of the encoder is shared by the MHQG and supporting fact prediction (SFP) models. The SFP model is a binary classifier trained on HotpotQA to output the probability of each sentence being a supporting fact. The F1 score between the predicted and ground truth supporting facts is added as a reward. The REINFORCE algorithm (Williams, 1992) is used with the self-critical sequence training (Rennie *et al.*, 2017) framework to avoid the high variance. In order to make the training more stable, a weight history similar to Rennie *et al.* (2017) is added. The output probabilities of the SFP model are also used to update the answer aware encodings by another Bi-LSTM model. For generating the question, a LSTM decoder with global attention mechanism (Luong *et al.*, 2015) is used along with the copy mechanism (See *et al.*, 2017a; Gulcehre *et al.*, 2016).

Sachan *et al.* (2020) argue that using standard transformers instead of Graph networks should be enough to reason about the relations between entities for forming multi-hop questions. A transformer is extended with sentence id embeddings and answer token indicator embeddings and trained with an additional contrastive loss as regularization. The contrastive learning setup assumes supporting fact sentences as positive samples and others as negative samples and a binary classifier consisting of a MLP. A significant mismatch in the distribution of question length over train and dev set of HotpotQA is observed and mitigated by filtering out all the questions that are more than 30 words long. Most of the pruned questions are from the train-easy subset. Both data filtering and contrastive training are found to boost the performance significantly.

A novel graph-augmented transformer encoder (GATE) is proposed which has two additional layers than the standard transformer encoder (TE): a) Graph-attention sub-layer computes the similarity scores for attention using only the nodes that are connected in a dynamically created graph. The graph is a multi-relational graph with three types

of nodes: named-entity mentions, co-referent-entities and sentence-ids. b) Fused attention sub-layer which uses a MLP with ReLU activation to aggregate the graph-attention embeddings and the TE embeddings. Experiments show that the added layers alone do not improve the performance significantly whereas an ensemble of TE and GATE does.

Pan *et al.* (2021) propose a question generation technique for the task of unsupervised MHQA. Following HotpotQA, their method generates two types of questions:

Bridge questions: Given two contexts as inputs, all entities common to the two contexts are considered as bridge entities. A Google T5 model (Raffel *et al.*, 2019) is fine-tuned on SQuAD to generate two single hop questions using the answer entity and the bridge entity respectively. The latter question is converted to a declarative form, s following Demszky *et al.* (2018). The bridge entity in the bridge entity question is replaced with "The [MASK] that {s}" and BERT-Large is used to fill the [MASK].

Comparison questions: Entities with NER types Nationality, Location, DateTime and Number are treated as potential comparative properties. Two single hop questions are generated on two entities of the same NER type and a pre-defined template is used to combine these into a multi-hop comparison question.

For generating questions using a table, a GPT-TabGen model (Chen *et al.*, 2020a) is used to generate sentences that describe a given entity using information from the given table. These sentences are then used for generating either *bridge* or *comparison* type questions. We refer the reader to Figure 4 of the original paper (Pan *et al.*, 2021) for a better understanding of the different types of generated questions. A pre-trained GPT-2 model is used to filter questions that are unnatural or dis-fluent. A BART model (Lewis *et al.*, 2019) is also used to paraphrase the generated questions. The generated questions are then used to train the model resulting in a zero shot algorithm.

Yu *et al.* (2020) aim to tackle MHQG in a low resource setting. Specifically, the model uses a small amount of labelled data D_L, in the form of $(context, answer, question)$ triplets, and a large set of multi-hop questions D_U. The idea is to first learn the semantics of multi-hop questions by training a neural hidden semi-Markov model (Dai *et al.*, 2017) on the unlabelled data D_U. The model uses two latent variables

for parameterizing the similar segments in questions from D_U: a) a state variable z_t, indicating which segment the t^{th} term belongs to, and b) a length variable l_t, specifying the length of the current segment. The term probabilities are computed using a GRU decoder followed by an attention layer.

The patterns learned in the first step are used as priors for the QG model for regularization. Prior is estimated by sampling a sequence of states z_t having length l_t. The reasoning chain extraction is similar to as described in Su *et al.* (2020). For encoding the textual input, BERT embeddings are passed to a bi-GRU followed by an attention layer. The decoder is another GRU with a copy mechanism (Gu *et al.*, 2016) which is regularized to fit the prior pattern. The training loss is the weighted sum of the cross entropy loss and RL policy gradient (Li *et al.*, 2017). The reward function evaluates a) fluency (following Zhang and Lapata (2017)), b) Answerability (using QBLEU4), and c) Semantics (using WMD).

Conclusion: The task of multi-hop question generation has gained some attention of the community and has advanced at a rapid pace. Solving MHQA would go a long way in solving MHQA since we can use MHQG models to generate high quality large scale datasets for the development of powerful MHQA models. Further, by dynamically constructing intricate questions that traverse multiple pieces of information, we pave the way for more nuanced understanding and exploration of complex knowledge domains, ultimately driving the evolution of AI-driven information retrieval and comprehension.

9

Future of MHQA

Multi-hop QA has been researched quite extensively in the recent years with multiple diverse models proposed that aim to model the multi-step retrieval-reasoning process and achieve promising improvements on existing datasets and benchmarks. Such systems capable of performing multi-step reasoning have a variety of applications ranging from chat-bot assistants that are capable of interactive conversations, to search engines that are capable to retrieve results that may be relevant but not reachable directly from the query text. At the same time the task of MHQA is significantly more challenging than its single hop counterpart. Since paragraphs multiple hops away from the question could share few common words and little semantic relation with the question (Ding *et al.*, 2019), the task to retrieve such contexts is challenging and suffers from semantic drift. The ability of pre-LLM models to combine multiple contexts for reasoning is also limited. Further challenges for solving MHQA is the difficult process of creating datasets that require the models to perform multi-hop reasoning, as well as the task of evaluating the models' abilities to do so without any hacks. Some challenging benchmarks and evaluation methods have been recently proposed that bring out some surprising and interesting observations. These results

point out to several limitations of existing systems and call for further research.

Below, we list and discuss some of these directions for future research (including ones originating from the currently recognized shortcomings) which we believe could be promising to be explored.

9.1 Flexible Any-Hop Models

As discussed in Section 6, majority of the existing methods for MHQA are either limited to two hops or require the number of hops to be a hyper-parameter. Since natural questions can require any number of reasoning hops, this attribute of existing models is artificially limiting. QA systems should be flexible and robust to the number of hops in a question in order to be practically usable. In order to do so, methods following the type III and IV in Figure 4.1 should be explored with a greater interest since feedback from the answering module can serve as a useful stopping criteria.

9.2 Explainable Multi-Hop QA

Despite the impressive gains in performance on various multi-hop datasets, it is not evident whether the models are performing the multi-step reasoning or just guessing the answers. Therefore, following the release of HotpotQA, a large number of works have focused on explainable MHQA. Apart from the standard evaluation metrics, various evaluation methods and dataset benchmarks have been proposed to test the explainability of the models which have revealed several significant results. Further such benchmarks and evaluation strategies are encouraged to measure and reflect the true progress of models in performing multi-hop reasoning. While LLM prompting strategies such as CoT prompting make the task naturally explainable, LLMs suffer from hallucinations which lowers their trust for such applications.

9.3 Better Datasets

Many works have highlighted the limitations of existing MHQA datasets. Chen and Durrett (2019) show experimentally that multi-choice questions are easier to hack by models, regardless of number of answer candidates being small or very large. Similarly, Min *et al.* (2019b) show that the distractor setting, even with as many as 500 distractors, is easier to hack for single-hop QA models. Yang *et al.* (2018) and Zhang *et al.* (2020) argue that datasets that are created using KBs suffer from lack of diversity in question and answer types. Following these observations, it is encouraged that the future datasets are in the open-domain setting, have questions with either span-based or generative answers, and do not rely completely on the structure of existing KBs.

Das *et al.* (2019), Xiong *et al.* (2019), Min *et al.* (2019b) and Min *et al.* (2019a) find that a significant portion of questions in the existing datasets are single-hop solvable due to a variety of reasons. One of these reasons is that the source of the questions is same as the set of contexts. Therefore datasets like (Welbl *et al.*, 2018; Kočiskỳ *et al.*, 2018; Wang *et al.*, 2021b; Trischler *et al.*, 2016; Choi *et al.*, 2018) that use separate sources for generating and answering questions are encouraged. However, attention needs to be given to the cases where there is a discrepancy between what is mentioned in the two sources (Fang *et al.*, 2020).

Most of the existing works have focused on datasets with MCQ or questions with span answers and more focus on the more challenging problem of generative MHQA is desirable.

9.4 Better Evaluation Metrics

As discussed in Section 7, a variety of evaluation metrics have been used for evaluating MHQA models. However, existing metrics face some challenges and might not be sufficient for evaluating MHQA. Since MHQA is a more complex task compared to single-hop QA, more metrics specific to MHQA are encouraged. One promising direction is to perform per-hop evaluation and accumulate the per-hop scores to get a final score. This kind of evaluation would require the models to be explainable as well as interpretable.

Some challenges are common to both single-hop and multi-hop QA evaluation metrics. For instance, while evaluating span based answers, metrics based on lexical matching would mark *U.S.* as incorrect when the gold answer is *United States* (Fang *et al.*, 2020). Therefore, evaluation metrics should be able to deal with synonyms when matching the answers. Another issue might be a metric giving a score of 0 to the answer *U.K.* when the gold answer is *London*. These cases are frequent in datasets like WikiHop where the sources of question and the contexts are different (De Cao *et al.*, 2019) and might have answers mentioned with different granularity. Therefore, it might be useful to give some partial score for the answer being geographically close or for the answer having a coarser granularity. Similarly, answering *January, 1989* should earn some score if the gold answer is *December, 1988* due to the predicted answer being temporally close to the gold answer. Some partial score can also be rewarded to answers having a coarser granularity (*December, 1988* vs *1988*). On similar lines, using hypernym relations from ConceptNet or WordNet (Miller, 1995) for evaluating the answer can be a possible direction.

Following a similar reasoning, the evaluation should match the semantics of the answer rather than the lexical overlap. Therefore, evaluation metrics like word mover similarity (Kusner *et al.*, 2015) or sentence mover similarity (Clark *et al.*, 2019) that perform soft matching over embeddings might be a promising direction. Since the evaluation of language generation tasks are widely known to have a scope of improvement, evaluation of generative MHQA is also an open problem and new evaluation techniques are encouraged.

9.5 Methods to Incorporate Commonsense

A hop in multi-hop reasoning can be performed using some retrieved context, where the context may either be retrieved from the corpus or from the commonsense knowledge. Fang *et al.* (2020) find that 16% of the failures of their model were caused by missing commonsense background knowledge. Bauer *et al.* (2018) propose a novel method for incorporating commonsense for MHQA that shows impressive results. More techniques for exploiting the rich commonsense knowledge bases

to perform multi-step reasoning can be a promising direction to explore.

9.6 Arithmetic Questions

The inability of QA systems to perform arithmetic operations is well known (Patel *et al.*, 2021; Hendrycks *et al.*, 2021; Schubotz *et al.*, 2018) and this inability is exacerbated in the multi-hop setting. Min *et al.* (2019a) observe that 45% of the comparison questions in HotpotQA are numerical questions. Qiu *et al.* (2019), Fang *et al.* (2020), and Min *et al.* (2019b) find that their model is unable to give a correct answer when the query is a comparison between two dates "February 20, 1959" and "February 10, 1967". Arithmetic calculation may also be required for non-comparison type questions. For example, answering "Who was the president of USA in 1994" from the context "Bill Clinton: 1993-2001" requires some arithmetic computation. Wang *et al.* (2021a) approached this kind of temporal problems by computing the overlap of content time expressions that occur in text with the computed question's time scope using kernel density estimate. Another example is (Wang *et al.*, 2021c) that answers "when" type questions by predicting the event dates based on the analysis of multi-variate time series derived from underlying news collection. Even powerful LLMs have been shown to perform poorly with arithmetic reasoning (Gao *et al.*, 2023; Maltoni and Ferrara, 2024) and multiple methods are being proposed for the same (Imani *et al.*, 2023; Guo, 2023; Maltoni and Ferrara, 2024; Gao *et al.*, 2023)

Multi-hop QA systems that are capable of solving arithmetic comparisons and computations would greatly enhance accuracy of MHQA.

Similarly, it is observed that some particular types of questions (temporal, geographical, count) are more challenging than others (Ding *et al.*, 2019; Zhang *et al.*, 2021; Min *et al.*, 2019b; De Cao *et al.*, 2019). Figure 3 in Zhang *et al.* (2021) shows the complexity and model performance on different types of questions in HotpotQA. Targeting the more challenging questions specifically could also lead to better MHQA systems.

9.7 Better Incorporation of Powerful LLMs

LLMs have been widely adopted due to their performance exceeding expectations for most tasks. Existing works employ LLMs for some components of the task to boost performance. The MHQA community would benefit from keeping up with the rapid developments of LLMs, incorporating the advancements in the models' abilities and efficiency. Consequently, developing more robust and powerful LLMs suited to multi-step reasoning would greatly help the development of MHQA systems

9.8 Conclusion

As we conclude, acknowledging both the strengths and weaknesses of existing data, models, and evaluation methods in multi-hop QA provides a solid foundation for charting future research paths. By leveraging insights gained from these limitations, we can propel advancements towards more robust, adaptable, and comprehensive question answering systems, shaping the landscape of AI-driven knowledge exploration for years to come.

Appendices

A

Background

A.1 BM25

BM25 is a ranking function used to retrieve documents given a search query. BM25 stands stands for *Best Match 25*[1]. It uses a bag-of-words mechanism to score proximity between the search query and the documents. Given a query $Q = q_1, q_2, ..., q_n$, where q_i denotes a keyword in the query Q, the BM25 score of the document D is defined as follows -

$$BM25(D,Q) = \sum_{i=1}^{n} IDF(q_i).\frac{freq(q_i,D).(k_1+1)}{freq(q_i,D) + k_1.(1-b+b.\frac{|D|}{avg.doc.len.})} \tag{A.1}$$

where $freq(q_i, D)$ is the number of times q_i occurs in D, $|.|$ denotes the number of words in D, $avg.doc.len.$ denotes the average number of words in the document, k_1 and b are free parameters[2], and $IDF(q_i)$ denotes the inverse document frequency weight of query term usually

[1]BM25 is also known as Okapi BM25, which was used first by the Okapi information retrieval system implemented by London's City University (https://en.wikipedia.org/wiki/Okapi_BM25).

[2]Typically $k_1 \in [1.2, 2.0]$ and $b = 0.75$.

computed as follows -

$$IDF(q_i) = ln(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}) + 1 \tag{A.2}$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i.

Even though the technique was devised in 1970s-80s, BM25 and its variations are still widely adopted for document retrieval, especially when the document corpus is very large and using *dense retrievers*[3] has a big computational overhead.

A.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of artificial neural networks that have loop connections that allow information propagation across time through the same neurons. Prior to transformer networks (Vaswani *et al.*, 2017b), RNNs were the most popular framework class to process sequential information, and are still widely adopted in real-world systems. Most practical RNN-based architectures have additional stored states that allow the vanilla RNN architecture to overcome its shortcoming of short-term memory loss. Gated recurrent units (GRU) cells (Cho *et al.*, 2014) and long short term memory (LSTM) cells (Hochreiter and Schmidhuber, 1997b) are two of the most popular stateful RNN cells that use gated mechanism to handle long term memory. (See *et al.*, 2017b) proposed a pointer generator network to overcome the over-repetition of RNN generated output using coverage loss. We point the readers to the comprehensive survey of recurrent neural networks by (Lipton *et al.*, 2015) for extensive explanation on the topic.

A.3 Transformers for Language Modeling

Even the advanced RNN models like LSTMs and GRUs have a tough time dealing with long sequences. Luong *et al.* (2015) introduced the

[3] *Dense retriever* is a general umbrella term used to refer to the neural network based retrieval systems.

attention mechanism which allows the model to focus on certain parts of the input when predicting a particular output token. Doing so significantly helps with tasks like machine translation where certain words of the input sequence are directly related to a word in the output sequence. Many forms of attention have since been used effectively for various tasks.

Vaswani *et al.* (2017a) extended the idea of attention by removing the recurrent component of the model altogether and proposed the transformer model where both the encoder and the decoder consist of several self-attention and feed forward layers. The transformer model also introduced the multi-head attention. These components allows for very large models which can have a lot more parameters without comprising on the performance. Transformers are also proved to be very versatile, having great success in a large number of natural language applications.

While the original transformers model was trained using the next-token prediction task implying the unidirectionality of the encoder model, BERT (Kenton and Toutanova, 2019) was a bidirectional encoder based transformer which was trained using the masked language modeling task. BERT has proved to be a versatile model and the word representations learned using BERT have been used as embeddings for almost all natural language tasks.

Success of transformer models including BERT led to their use as large pre-training models and several models like AlBERT (Lan *et al.*, 2019), RoBERTa (Liu *et al.*, 2019) and GPT were proposed. AlBERT uses parameter reduction techniques which allow for smaller and faster training of the BERT models while achieving a similar level of accuracy as BERT. RoBERTa is a much more robustly optimized version of BERT, trained with optimized design and hyperparameters choices, which could significantly outperform the originally trained BERT model.

Pre-training of large language models (LLMs) has become increasingly popular leading to larger and larger models trained on huge corpora of natural language. The different versions of the model follow the same principle, with GPT-1 having 117 million parameters and GPT-4 having about a 100 trillion parameters. GPTs are trained on huge corpora using the next token prediction task. An extensively detailed explanation of

different architectures and training techniques for transformer based models is neither feasible nor in the scope for this work. Therefore, we point the readers to the comprehensive survey of transformers by (Lin *et al.*, 2022) for further details on the topic.

A.4 Graph Neural Networks

Graphs are a very simple and versatile method of representing data and its inherent structure. Neural Networks could be adapted to incorporate this structure leading to Graph Neural Networks (GNNs). GNNs can be adopted for various different types of data and tasks, leading to several improvements increasing their capabilities. The integral part of all these models is the message passing algorithm briefly explained below.

Given a graph $G = (V, E)$ having $n = |V|$ nodes, the representation of each node is updated following the given steps:

- **Initialization:** The representation of every node v is initialized as $h_v^0 = X_v$, where X_v is the feature vector.
- **Update:** For each layer i, the representations of each node v is updated as:

$$h_v^i = \sigma_{u \epsilon N(v)}(W_i \Sigma \frac{h_u^{i-1}}{N(v)} + U_i h_v^{i-1}) \tag{A.3}$$

  where σ is the activation function, W_i and U_i are the weight matrices corresponding to the layer i and $N(v)$ is the set of neighbouring nodes of the node v.
- **Prediction:** The representations after layer K are passed to a linear network for the eventual prediction task.

At every layer, the representation of node v is updated with an activation applied to the weighted average of representations of the nodes directly connected to v. Therefore, after k layers, the node v is supposed to receive the 'message' from all nodes having a path to v of length $\leq k$. The weighted average also ensures that the nodes that are closer to v in the graph end up affecting its representation more.

A layer of a Graph Convolutional Network (GCN) (Kipf and Welling, 2016b) consists of a GNN layer followed by a Linear layer. Relation GCN (R-GCN) (Schlichtkrull *et al.*, 2017) allow for different kinds of edges by having different weight matrices for nodes connected to v via different kind of edges. Graph Attention Networks (GAN) (Veličković *et al.*, 2018) incorporate self attention into GNNs by using the attention weights while performing the message passing algorithm. Several other modifications of GNNs are proposed for different tasks.

We point the readers to the comprehensive survey of graph neural networks by (Wu *et al.*, 2020) for further reading on the topic.

A.5 Large Language models

Language models refer to a class of self-supervised NLP models that are trained on large unlabeled datasets to learn to predict the likelihood of a word or sequence of words occurring based on the context provided by the preceding words. This ability to estimate the probability of a word given its context forms the foundation of language modeling. These models undergo training on various tasks, such as next-word prediction (Brown *et al.*, 2020), masked language modeling (the task of predicting randomly missing tokens), and next-sentence prediction (Kenton and Toutanova, 2019), without the need for labeled data. Due to their reliance on extensive training data, language models develop a strong grasp of underlying language patterns and concepts. Generally, language models are not designed for specific tasks and can be fine-tuned with minimal data for various downstream applications. Extensive research has shown that utilizing large language models (LLMs) pre-trained on vast amounts of data yields impressive results in language understanding and generation tasks (Tan *et al.*, 2023; Wang *et al.*, 2021d; Hendy *et al.*, 2023; Blair-Stanek *et al.*, 2023). The advent of transformer models has made it possible to train such highly advanced language models, resulting in popular models like BERT, T5, and GPT-3 (Kenton and Toutanova, 2019; Raffel *et al.*, 2019; Brown *et al.*, 2020).

A.5.1 Generative Pre-trained Transformer (GPT)

GPT, a series of generative pre-trained large language models (Brown *et al.*, 2020), is characterized by its decoder-only transformer architecture. Unlike other transformer models that have both encoder and decoder blocks, GPT models consist solely of decoder blocks, eliminating the encoder-decoder cross-attention layer from each block. The different versions of GPT, namely GPT, GPT-2, GPT-3, and GPT-4, vary in terms of model size and training data. For example, GPT-3 has 175 billion model parameters and is trained on a massive corpus of 499 billion tokens, while GPT-2 has 1.5 billion parameters and is trained on a dataset of 10 billion tokens.

A.5.2 Prompting GPT-3

GPT-3 has achieved remarkable success in various downstream natural language tasks, including question answering (Tan *et al.*, 2023), Machine Translation (Hendy *et al.*, 2023) and Entailment prediction (Wang *et al.*, 2021d), with minimal supervision required. During a typical run of the model, an incomplete piece of text is provided as a 'prompt', and the model iteratively generates the most likely tokens to complete the text. This prompting technique has demonstrated impressive performance in the zero-shot setting, where the model is not provided with any in-context examples and is expected to predict the correct output for the given question in the prompt (Figure A.1).

On the other hand, few-shot prompting (Fei-Fei *et al.*, 2006) involves including a small number of sample input-output pairs within the prompt as references for the model (Figure A.1). The inclusion of a few reference examples provides valuable guidance to the model, allowing it to generate more accurate and relevant responses.

In their work, Wei *et al.* (2023) introduced the concept of chain-of-thought (CoT) prompting, which goes a step beyond simply providing input-sample output pairs. CoT prompting includes a coherent sequence of reasoning steps that gradually build up to the correct answer. By presenting the model with a step-by-step thought process, CoT prompting offers explicit examples of how to arrive at the correct answer based on the given input facts. This method is particularly valuable for tackling

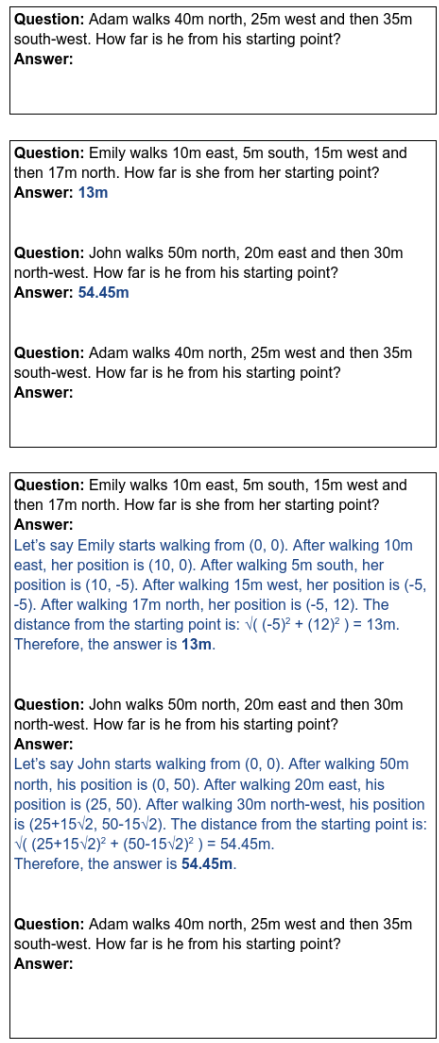


Figure A.1: Example of zero-shot (top), few-shot (middle), and CoT (bottom) prompting for the same question.

complex tasks that demand multiple layers of reasoning including the task that this study focuses on. Figure A.1 shows examples of zero-shot, few-shot, and CoT prompts for an arithmetic question. Here, the prompt consists of 2 in-context examples is 2.

For further background and details, we refer the readers to the comprehensive survey on LLMs by Zhao *et al.* (2023b)

References

Ainslie, J., S. Ontanon, C. Alberti, V. Cvicek, Z. Fisher, P. Pham, A. Ravula, S. Sanghai, Q. Wang, and L. Yang. (2020). "ETC: Encoding Long and Structured Inputs in Transformers". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics. 268–284. DOI: 10.18653/v1/2020.emnlp-main.19. URL: https://aclanthology.org/2020.emnlp-main.19.

Allam, A. M. N. and M. H. Haggag. (2012). "The question answering systems: A survey". *International Journal of Research and Reviews in Information Sciences (IJRRIS)*. 2(3).

Alvarez-Melis, D. and T. Jaakkola. (2017). "A causal framework for explaining the predictions of black-box sequence-to-sequence models". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics. 412–421. DOI: 10.18653/v1/D17-1042. URL: https://aclanthology.org/D17-1042.

Arras, L., F. Horn, G. Montavon, K.-R. Müller, and W. Samek. (2016). "What is Relevant in a Text Document?": An Interpretable Machine Learning Approach. CoRR abs/1612.07843 (2016)". *arXiv preprint arXiv:1612.07843*.

Balepur, N., J. Huang, S. Moorjani, H. Sundaram, and K. C.-C. Chang. (2023). "Mastering the ABCDs of Complex Questions: Answer-Based Claim Decomposition for Fine-grained Self-Evaluation". arXiv: 2305.14750 [cs.CL].

Banarescu, L., C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. (2013). "Abstract Meaning Representation for Sembanking". In: *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Ed. by A. Pareja-Lora, M. Liakata, and S. Dipper. Sofia, Bulgaria: Association for Computational Linguistics. 178–186. URL: https://aclanthology.org/W13-2322.

Banerjee, S. and A. Lavie. (2005). "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments". In: *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan: Association for Computational Linguistics. 65–72. URL: https://aclanthology.org/W05-0909.

Barzilay, R., K. McKeown, and M. Elhadad. (1999). "Information fusion in the context of multi-document summarization". In: *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*. 550–557.

Bauer, L., Y. Wang, and M. Bansal. (2018). "Commonsense for Generative Multi-Hop Question Answering Tasks". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4220–4230.

Becker, R., F. Corò, G. D'Angelo, and H. Gilbert. (2020). "Balancing Spreads of Influence in a Social Network". *Proceedings of the AAAI Conference on Artificial Intelligence*. 34(Apr.): 3–10. DOI: 10.1609/aaai.v34i01.5327. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5327.

Bhagavatula, C. S., T. Noraset, and D. Downey. (2013). "Methods for exploring and mining tables on wikipedia". In: *Proceedings of the ACM SIGKDD workshop on interactive data exploration and analytics*. 18–26.

Biran, O. and C. Cotton. (2017). "Explanation and justification in machine learning: A survey". In: *IJCAI-17 workshop on explainable AI (XAI)*. Vol. 8. No. 1. 8–13.

Blair-Stanek, A., N. Holzenberger, and B. V. Durme. (2023). "Can GPT-3 Perform Statutory Reasoning?" arXiv: 2302.06100 [cs.CL].

Boros, E., J. G. Moreno, and A. Doucet. (2021). "Event Detection as Question Answering with Entity Information". *CoRR*. abs/2104.06969. arXiv: 2104.06969. URL: https://arxiv.org/abs/2104.06969.

Bouziane, A., D. Bouchiha, N. Doumi, and M. Malki. (2015). "Question answering systems: survey and trends". *Procedia Computer Science*. 73: 366–375.

Bowman, S. R., G. Angeli, C. Potts, and C. D. Manning. (2015). "A large annotated corpus for learning natural language inference". In: *Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*. Association for Computational Linguistics (ACL). 632–642.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. (2020). "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Cao, X. and Y. Liu. (2021). "Coarse-grained decomposition and fine-grained interaction for multi-hop question answering". *Journal of Intelligent Information Systems*: 1–21.

Cao, Y., M. Fang, and D. Tao. (2019). "BAG: Bi-directional Attention Entity Graph Convolutional Network for Multi-hop Reasoning Question Answering". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 357–362.

Chen, D., A. Fisch, J. Weston, and A. Bordes. (2017a). "Reading Wikipedia to Answer Open-Domain Questions". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics. 1870–1879. DOI: 10.18653/v1/P17-1171. URL: https://aclanthology.org/P17-1171.

Chen, J. and G. Durrett. (2019). "Understanding Dataset Design Choices for Multi-hop Reasoning". 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)*. Minneapolis, Minnesota: Association for Computational Linguistics. 4026–4032. DOI: 10.18653/v1/N19-1405. URL: https://aclanthology.org/N19-1405.

Chen, J., S.-t. Lin, and G. Durrett. (2019). "Multi-hop question answering via reasoning chains". *arXiv preprint arXiv:1910.02610*.

Chen, Q., X. Zhu, Z.-H. Ling, S. Wei, H. Jiang, and D. Inkpen. (2017b). "Enhanced LSTM for Natural Language Inference". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics. 1657–1668. DOI: 10.18653/v1/P17-1152. URL: https://aclanthology.org/P17-1152.

Chen, W., J. Chen, Y. Su, Z. Chen, and W. Y. Wang. (2020a). "Logical Natural Language Generation from Open-Domain Tables". *CoRR*. abs/2004.10404. arXiv: 2004.10404. URL: https://arxiv.org/abs/2004.10404.

Chen, W., H. Zha, Z. Chen, W. Xiong, H. Wang, and W. Y. Wang. (2020b). "HybridQA: A Dataset of Multi-Hop Question Answering over Tabular and Textual Data". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 1026–1036.

Chen, Z., W. Chen, C. Smiley, S. Shah, I. Borova, D. Langdon, R. Moussa, M. Beane, T.-H. Huang, B. Routledge, and W. Y. Wang. (2021). "FinQA: A Dataset of Numerical Reasoning over Financial Data". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. 3697–3711. DOI: 10.18653/v1/2021.emnlp-main.300. URL: https://aclanthology.org/2021.emnlp-main.300.

Cho, K., B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*.

Choi, E., H. He, M. Iyyer, M. Yatskar, W. Yih, Y. Choi, P. Liang, and L. Zettlemoyer. (2018). "QuAC : Question Answering in Context". *CoRR*. abs/1808.07036. arXiv: 1808.07036. URL: http://arxiv.org/abs/1808.07036.

Chung, H. W., L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. (2022). "Scaling Instruction-Finetuned Language Models". DOI: 10.48550/ARXIV.2210.11416. URL: https://arxiv.org/abs/2210.11416.

Church, K. W. and P. Hanks. (1990). "Word Association Norms, Mutual Information, and Lexicography". *Computational Linguistics*. 16(1): 22–29. URL: https://aclanthology.org/J90-1003.

Clark, E., A. Celikyilmaz, and N. A. Smith. (2019). "Sentence mover's similarity: Automatic evaluation for multi-sentence texts". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2748–2760.

Clark, P., I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. (2018). "Think you have solved question answering? try arc, the ai2 reasoning challenge". *arXiv preprint arXiv:1803.05457*.

Cobbe, K., V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. (2021). "Training Verifiers to Solve Math Word Problems". arXiv: 2110.14168 [cs.LG].

Dai, H., B. Dai, Y. Zhang, S. Li, and L. Song. (2017). "Recurrent Hidden Semi-Markov Model". In: *ICLR*.

Das, R., A. Godbole, D. Kavarthapu, Z. Gong, A. Singhal, M. Yu, X. Guo, T. Gao, H. Zamani, M. Zaheer, *et al.* (2019). "Multi-step entity-centric information retrieval for multi-hop question answering". In: *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. 113–118.

Dasigi, P., K. Lo, I. Beltagy, A. Cohan, N. A. Smith, and M. Gardner. (2021). "A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics. 4599–4610. DOI: 10.18653/v1/2021.naacl-main.365. URL: https://aclanthology.org/2021.naacl-main.365.

De Cao, N., W. Aziz, and I. Titov. (2019). "Question Answering by Reasoning Across Documents with Graph Convolutional Networks". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2306–2317.

Demszky, D., K. Guu, and P. Liang. (2018). "Transforming Question Answering Datasets Into Natural Language Inference Datasets". *CoRR*. abs/1809.02922. arXiv: 1809.02922. URL: http://arxiv.org/abs/1809.02922.

Deng, Z., Y. Zhu, Y. Chen, M. Witbrock, and P. Riddle. (2022). "Interpretable AMR-Based Question Decomposition for Multi-hop Question Answering". In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. Ed. by L. D. Raedt. International Joint Conferences on Artificial Intelligence Organization. 4093–4099. DOI: 10.24963/ijcai.2022/568. URL: https://doi.org/10.24963/ijcai.2022/568.

Diefenbach, D., V. Lopez, K. Singh, and P. Maret. (2018). "Core techniques of question answering systems over knowledge bases: a survey". *Knowledge and Information systems*. 55(3): 529–569.

Dimitrakis, E., K. Sgontzos, and Y. Tzitzikas. (2020). "A survey on question answering systems over linked data and documents". *Journal of intelligent information systems*. 55(2): 233–259.

Ding, M., C. Zhou, Q. Chen, H. Yang, and J. Tang. (2019). "Cognitive Graph for Multi-Hop Reading Comprehension at Scale". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 2694–2703. DOI: 10.18653/v1/P19-1259. URL: https://aclanthology.org/P19-1259.

Du, X., J. Shao, and C. Cardie. (2017). "Learning to Ask: Neural Question Generation for Reading Comprehension". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics. 1342–1352. DOI: 10.18653/v1/P17-1123. URL: https://aclanthology.org/P17-1123.

Dua, D., C. dos Santos, P. Ng, B. Athiwaratkun, B. Xiang, M. Gardner, and S. Singh. (2021). "Generative Context Pair Selection for Multihop Question Answering". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 7009–7015.

Dua, D., S. Singh, and M. Gardner. (2020). "Benefits of Intermediate Annotations in Reading Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics. 5627–5634. DOI: 10.18653/v1/2020.acl-main.497. URL: https://aclanthology.org/2020.acl-main.497.

Dua, D., Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner. (2019). "DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs". 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)*. Minneapolis, Minnesota: Association for Computational Linguistics. 2368–2378. DOI: 10.18653/v1/N19-1246. URL: https://aclanthology.org/N19-1246.

Etezadi, R. and M. Shamsfard. (2023). "The state of the art in open domain complex question answering: a survey". *Applied Intelligence*. 53(4): 4124–4144.

Fan, A., C. Gardent, C. Braud, and A. Bordes. (2019). "Using Local Knowledge Graph Construction to Scale Seq2Seq Models to Multi-Document Inputs". 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)*. Hong Kong, China: Association for Computational Linguistics. 4186–4196. DOI: 10.18653/v1/D19-1428. URL: https://aclanthology.org/D19-1428.

Fang, Y., S. Sun, Z. Gan, R. Pillai, S. Wang, and J. Liu. (2020). "Hierarchical Graph Network for Multi-hop Question Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 8823–8838.

Fei-Fei, L., R. Fergus, and P. Perona. (2006). "One-Shot Learning of Object Categories". *IEEE transactions on pattern analysis and machine intelligence*. 28(May): 594–611. DOI: 10.1109/TPAMI.2006.79.

Feldman, Y. and R. El-Yaniv. (2019). "Multi-Hop Paragraph Retrieval for Open-Domain Question Answering". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 2296–2309. DOI: 10.18653/v1/P19-1222. URL: https://aclanthology.org/P19-1222.

Feng, S., W. Shi, Y. Bai, V. Balachandran, T. He, and Y. Tsvetkov. (2024a). "Knowledge Card: Filling LLMs' Knowledge Gaps with Plug-in Specialized Language Models". arXiv: 2305.09955 [cs.CL].

Feng, S., W. Shi, Y. Wang, W. Ding, V. Balachandran, and Y. Tsvetkov. (2024b). "Don't Hallucinate, Abstain: Identifying LLM Knowledge Gaps via Multi-LLM Collaboration". arXiv: 2402.00367 [cs.CL].

Feng, Y., M. Yu, W. Xiong, X. Guo, J. Huang, S. Chang, M. Campbell, M. Greenspan, and X. Zhu. (2020). "Learning to recover reasoning chains for multi-hop question answering via cooperative games". *arXiv preprint arXiv:2004.02393*.

Fu, B., Y. Qiu, C. Tang, Y. Li, H. Yu, and J. Sun. (2020). "A survey on complex question answering over knowledge base: Recent advances and challenges". *arXiv preprint arXiv:2007.13069*.

Gao, L., A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. (2023). "PAL: Program-aided Language Models". arXiv: 2211.10435 [cs.CL].

Gao, Y., J. Wang, L. Bing, I. King, and M. R. Lyu. (2018). "Difficulty Controllable Question Generation for Reading Comprehension". *CoRR*. abs/1807.03586. arXiv: 1807.03586. URL: http://arxiv.org/abs/1807.03586.

Gatt, A. and E. Krahmer. (2018). "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation". *Journal of Artificial Intelligence Research*. 61: 65–170.

Geva, M., Y. Goldberg, and J. Berant. (2019a). "Are We Modeling the Task or the Annotator? An Investigation of Annotator Bias in Natural Language Understanding Datasets". 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)*. Hong Kong, China: Association for Computational Linguistics. 1161–1166. DOI: 10.18653/v1/D19-1107. URL: https://aclanthology.org/D19-1107.

Geva, M., E. Malmi, I. Szpektor, and J. Berant. (2019b). "DiscoFuse: A Large-Scale Dataset for Discourse-Based Sentence Fusion". 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)*. 3443–3455.

Ghalandari, D. G. and G. Ifrim. (2020). "Examining the state-of-the-art in news timeline summarization". *arXiv preprint arXiv:2005.10107*.

Gilpin, L. H., D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. (2018). "Explaining explanations: An overview of interpretability of machine learning". In: *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. IEEE. 80–89.

Goldstein, J., V. O. Mittal, J. G. Carbonell, and M. Kantrowitz. (2000). "Multi-document summarization by sentence extraction". In: *NAACL-ANLP 2000 workshop: automatic summarization*.

Graves, A., G. Wayne, and I. Danihelka. (2014). "Neural turing machines". *arXiv preprint arXiv:1410.5401*.

Gu, J., Z. Lu, H. Li, and V. O. Li. (2016). "Incorporating Copying Mechanism in Sequence-to-Sequence Learning". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics. 1631–1640. DOI: 10.18653/v1/P16-1154. URL: https://aclanthology.org/P16-1154.

Gulcehre, C., S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio. (2016). "Pointing the Unknown Words". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics. 140–149. DOI: 10.18653/v1/P16-1014. URL: https://aclanthology.org/P16-1014.

Guo, Y. (2023). "ArthModel: Enhance Arithmetic Skills to Large Language Model". arXiv: 2311.18609 [cs.CL].

Gupta, D., H. Chauhan, R. T. Akella, A. Ekbal, and P. Bhattacharyya. (2020). "Reinforced Multi-task Approach for Multi-hop Question Generation". In: *Proceedings of the 28th International Conference on Computational Linguistics*. 2760–2775.

Gururangan, S., S. Swayamdipta, O. Levy, R. Schwartz, S. Bowman, and N. A. Smith. (2018). "Annotation Artifacts in Natural Language Inference Data". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics. 107–112. DOI: 10.18653/v1/N18-2017. URL: https://aclanthology.org/N18-2017.

Haghighi, A. and L. Vanderwende. (2009). "Exploring content models for multi-document summarization". In: *Proceedings of human language technologies: The 2009 annual conference of the North American Chapter of the Association for Computational Linguistics*. 362–370.

Haji, S., K. Suekane, H. Sano, and T. Takagi. (2023). "Exploratory Inference Chain: Exploratorily Chaining Multi-hop Inferences with Large Language Models for Question-Answering". In: *2023 IEEE 17th International Conference on Semantic Computing (ICSC)*. 175–182. DOI: 10.1109/ICSC56153.2023.00036.

Han, S., H. Schoelkopf, Y. Zhao, Z. Qi, M. Riddell, L. Benson, L. Sun, E. Zubova, Y. Qiao, M. Burtell, D. Peng, J. Fan, Y. Liu, B. Wong, M. Sailor, A. Ni, L. Nan, J. Kasai, T. Yu, R. Zhang, S. Joty, A. R. Fabbri, W. Kryscinski, X. V. Lin, C. Xiong, and D. Radev. (2022). "FOLIO: Natural Language Reasoning with First-Order Logic". arXiv: 2209.00840 [cs.CL].

He, P., X. Liu, J. Gao, and W. Chen. (2021). "DeBERTa: Decoding-enhanced BERT with Disentangled Attention". arXiv: 2006.03654 [cs.CL].

Hendrycks, D., C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. (2021). "Measuring mathematical problem solving with the math dataset". *arXiv preprint arXiv:2103.03874*.

Hendy, A., M. Abdelrehim, A. Sharaf, V. Raunak, M. Gabr, H. Matsushita, Y. J. Kim, M. Afify, and H. H. Awadalla. (2023). "How Good Are GPT Models at Machine Translation? A Comprehensive Evaluation". arXiv: 2302.09210 [cs.CL].

Hermann, K. M., T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. (2015). "Teaching machines to read and comprehend". *Advances in neural information processing systems*. 28.

Hochreiter, S. and J. Schmidhuber. (1997a). "Long short-term memory". *Neural computation*. 9(8): 1735–1780.

Hochreiter, S. and J. Schmidhuber. (1997b). "Long short-term memory". *Neural computation*. 9(8): 1735–1780.

Höffner, K., S. Walter, E. Marx, R. Usbeck, J. Lehmann, and A.-C. Ngonga Ngomo. (2017). "Survey on challenges of question answering in the semantic web". *Semantic Web*. 8(6): 895–920.

Huang, Y. and M. Yang. (2021). "Breadth First Reasoning Graph for Multi-hop Question Answering". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics. 5810–5821. DOI: 10.18653/v1/2021.naacl-main.464. URL: https://aclanthology.org/2021.naacl-main.464.

Imani, S., L. Du, and H. Shrivastava. (2023). "MathPrompter: Mathematical Reasoning using Large Language Models". arXiv: 2303.05398 [cs.CL].

Inoue, N., P. Stenetorp, and K. Inui. (2020). "R4C: A Benchmark for Evaluating RC Systems to Get the Right Answer for the Right Reason". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics. 6740–6750. DOI: 10.18653/v1/2020.acl-main.602. URL: https://aclanthology.org/2020.acl-main.602.

Jansen, P. (2018). "Multi-hop Inference for Sentence-level TextGraphs: How Challenging is Meaningfully Combining Information for Science Question Answering?" In: *Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12)*. 12–17.

Jansen, P., E. Wainwright, S. Marmorstein, and C. Morrison. (2018). "WorldTree: A Corpus of Explanation Graphs for Elementary Science Questions supporting Multi-hop Inference". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA). URL: https://aclanthology.org/L18-1433.

Järvelin, K. and J. Kekäläinen. (2002). "Cumulated Gain-Based Evaluation of IR Techniques". *ACM Trans. Inf. Syst.* 20(4): 422–446. ISSN: 1046-8188. DOI: 10.1145/582415.582418. URL: https://doi.org/10.1145/582415.582418.

Jatowt, A., C. A. Yeung, and K. Tanaka. (2013). "Estimating document focus time". In: *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*. Ed. by Q. He, A. Iyengar, W. Nejdl, J. Pei, and R. Rastogi. ACM. 2273–2278. DOI: 10.1145/2505515.2505655. URL: https://doi.org/10.1145/2505515.2505655.

Jhamtani, H. and P. Clark. (2020). "Learning to Explain: Datasets and Models for Identifying Valid Reasoning Chains in Multihop Question-Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics. 137–150. DOI: 10.18653/v1/2020.emnlp-main.10. URL: https://aclanthology.org/2020.emnlp-main.10.

Jia, R. and P. Liang. (2017). "Adversarial Examples for Evaluating Reading Comprehension Systems". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2021–2031.

Jin, Q., Z. Yuan, G. Xiong, Q. Yu, H. Ying, C. Tan, M. Chen, S. Huang, X. Liu, and S. Yu. (2022). "Biomedical Question Answering: A Survey of Approaches and Challenges". *ACM Computing Surveys (CSUR)*. 55(2): 1–36.

Joshi, N., H. Zhang, K. Kalyanaraman, Z. Hu, K. Chellapilla, H. He, and L. E. Li. (2023a). "Improving Multi-Hop Reasoning in LLMs by Learning from Rich Human Feedback". In: *Neuro-Symbolic Learning and Reasoning in the era of Large Language Models*. URL: https://openreview.net/forum?id=wxfqhp9bNR.

Joshi, N., H. Zhang, K. Kalyanaraman, Z. Hu, K. Chellapilla, H. He, and L. E. Li. (2023b). "Improving Multi-Hop Reasoning in LLMs by Learning from Rich Human Feedback". In: *Neuro-Symbolic Learning and Reasoning in the era of Large Language Models*. URL: https://openreview.net/forum?id=wxfqhp9bNR.

Kadavath, S., T. Conerly, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schiefer, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson, S. Johnston, S. El-Showk, A. Jones, N. Elhage, T. Hume, A. Chen, Y. Bai, S. Bowman, S. Fort, D. Ganguli, D. Hernandez, J. Jacobson, J. Kernion, S. Kravec, L. Lovitt, K. Ndousse, C. Olsson, S. Ringer, D. Amodei, T. Brown, J. Clark, N. Joseph, B. Mann, S. McCandlish, C. Olah, and J. Kaplan. (2022). "Language Models (Mostly) Know What They Know". arXiv: 2207.05221 [cs.CL].

Kadlec, R., O. Bajgar, and J. Kleindienst. (2017). "Knowledge Base Completion: Baselines Strike Back". In: *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada: Association for Computational Linguistics. 69–74. DOI: 10.18653/v1/W17-2609. URL: https://aclanthology.org/W17-2609.

Kenton, J. D. M.-W. C. and L. K. Toutanova. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of NAACL-HLT*. 4171–4186.

Khashabi, D., S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth. (2018). "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 252–262.

Khot, T., P. Clark, M. Guerquin, P. Jansen, and A. Sabharwal. (2020). "Qasc: A dataset for question answering via sentence composition". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. No. 05. 8082–8090.

Khot, T., A. Sabharwal, and P. Clark. (2019). "What's Missing: A Knowledge Gap Guided Approach for Multi-hop Question Answering". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2814–2828.

Khot, T., H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal. (2023). "Decomposed Prompting: A Modular Approach for Solving Complex Tasks". arXiv: 2210.02406 [cs.CL].

Kim, J.-H., J. Jun, and B.-T. Zhang. (2018). "Bilinear attention networks". *Advances in Neural Information Processing Systems*. 31.

Kipf, T. N. and M. Welling. (2016a). "Semi-Supervised Classification with Graph Convolutional Networks". *CoRR*. abs/1609.02907. arXiv: 1609.02907. URL: http://arxiv.org/abs/1609.02907.

Kipf, T. N. and M. Welling. (2016b). "Semi-Supervised Classification with Graph Convolutional Networks". *CoRR*. abs/1609.02907. arXiv: 1609.02907. URL: http://arxiv.org/abs/1609.02907.

Kočiskỳ, T., J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. (2018). "The narrativeqa reading comprehension challenge". *Transactions of the Association for Computational Linguistics*. 6: 317–328.

Kovaleva, O., A. Romanov, A. Rogers, and A. Rumshisky. (2019). "Revealing the Dark Secrets of BERT". 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)*. Hong Kong, China: Association for Computational Linguistics. 4365–4374. DOI: 10.18653/v1/D19-1445. URL: https://aclanthology.org/D19-1445.

Kumar, V., Y. Hua, G. Ramakrishnan, G. Qi, L. Gao, and Y.-F. Li. (2019). "Difficulty-controllable multi-hop question generation from knowledge graphs". In: *International Semantic Web Conference*. Springer. 382–398.

Kurdi, G., J. Leo, B. Parsia, U. Sattler, and S. Al-Emari. (2019). "A Systematic Review of Automatic Question Generation for Educational Purposes". *International Journal of Artificial Intelligence in Education*. 30(Nov.). DOI: 10.1007/s40593-019-00186-y.

Kusner, M., Y. Sun, N. Kolkin, and K. Weinberger. (2015). "From word embeddings to document distances". In: *International conference on machine learning*. PMLR. 957–966.

Lan, Y., G. He, J. JIANG, J. JIANG, W. X. ZHAO, and J.-R. WEN. (2021). "A survey on complex knowledge base question answering: Methods, challenges and solutions". In: IJCAI.

Lan, Z., M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. (2019). "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *International Conference on Learning Representations*.

Lebanoff, L., J. Muchovej, F. Dernoncourt, D. S. Kim, S. Kim, W. Chang, and F. Liu. (2019). "Analyzing Sentence Fusion in Abstractive Summarization". In: *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. 104–110.

Lei, F., X. Li, Y. Wei, S. He, Y. Huang, J. Zhao, and K. Liu. (2023). "S^{3}HQA: A Three-Stage Approach for Multi-hop Text-Table Hybrid Question Answering". arXiv: 2305.11725 [cs.CL].

Lewis, M., Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. (2019). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". *CoRR*. abs/1910.13461. arXiv: 1910.13461. URL: http://arxiv.org/abs/1910.13461.

Li, J., M. Ren, Y. Gao, and Y. Yang. (2023). "Ask to Understand: Question Generation for Multi-hop Question Answering". In: *Chinese Computational Linguistics*. Ed. by M. Sun, B. Qin, X. Qiu, J. Jing, X. Han, G. Rao, and Y. Chen. Singapore: Springer Nature Singapore. 19–36. ISBN: 978-981-99-6207-5.

Li, J., W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky. (2017). "Adversarial Learning for Neural Dialogue Generation". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics. 2157–2169. DOI: 10.18653/v1/D17-1230. URL: https://aclanthology.org/D17-1230.

Li, R. and X. Du. (2023). "Leveraging Structured Information for Explainable Multi-hop Question Answering and Reasoning". arXiv: 2311.03734 [cs.CL].

Liang, P., R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, B. Newman, B. Yuan, B. Yan, C. Zhang, C. Cosgrove, C. D. Manning, C. Ré, D. Acosta-Navas, D. A. Hudson, E. Zelikman, E. Durmus, F. Ladhak, F. Rong, H. Ren, H. Yao, J. Wang, K. Santhanam, L. Orr, L. Zheng, M. Yuksekgonul, M. Suzgun, N. Kim, N. Guha, N. Chatterji, O. Khattab, P. Henderson, Q. Huang, R. Chi, S. M. Xie, S. Santurkar, S. Ganguli, T. Hashimoto, T. Icard, T. Zhang, V. Chaudhary, W. Wang, X. Li, Y. Mai, Y. Zhang, and Y. Koreeda. (2023). "Holistic Evaluation of Language Models". arXiv: 2211.09110 [cs.CL].

Lin, C.-Y. (2004). "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics. 74–81. URL: https://aclanthology.org/W04-1013.

Lin, T., Y. Wang, X. Liu, and X. Qiu. (2022). "A survey of transformers". *AI Open*. 3: 111–132. ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2022.10.001. URL: https://www.sciencedirect.com/science/article/pii/S2666651022000146.

Lin, Z., D. Zhang, Q. Tac, D. Shi, G. Haffari, Q. Wu, M. He, and Z. Ge. (2021). "Medical Visual Question Answering: A Survey". *arXiv preprint arXiv:2111.10056*.

Lindberg, D., F. Popowich, J. Nesbit, and P. Winne. (2013). "Generating Natural Language Questions to Support Learning On-Line". In: *Proceedings of the 14th European Workshop on Natural Language Generation*. Sofia, Bulgaria: Association for Computational Linguistics. 105–114. URL: https://aclanthology.org/W13-2114.

Lipton, Z. C., J. Berkowitz, and C. Elkan. (2015). "A critical review of recurrent neural networks for sequence learning". *arXiv preprint arXiv:1506.00019*.

Liu, L. and M. T. Özsu. (2009). "Mean average precision". *Encyclopedia of Database Systems2009*. 1703.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. (2019). "Roberta: A robustly optimized bert pretraining approach". *arXiv preprint arXiv:1907.11692*.

Luong, T., H. Pham, and C. D. Manning. (2015). "Effective Approaches to Attention-based Neural Machine Translation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics. 1412–1421. DOI: 10.18653/v1/D15-1166. URL: https://aclanthology.org/D15-1166.

Ma, C., W. E. Zhang, M. Guo, H. Wang, and Q. Z. Sheng. (2020). "Multi-document summarization via deep learning techniques: A survey". *arXiv preprint arXiv:2011.04843*.

Madaan, A., N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark. (2023). "Self-Refine: Iterative Refinement with Self-Feedback". arXiv: 2303.17651 [cs.CL].

Malon, C. and B. Bai. (2020). "Generating followup questions for interpretable multi-hop question answering". *arXiv preprint arXiv:2002.12344*.

Maltoni, D. and M. Ferrara. (2024). "Arithmetic with Language Models: from Memorization to Computation". arXiv: 2308.01154 [cs.AI].

Mavi, V., A. Saparov, and C. Zhao. (2023). "Retrieval-Augmented Chain-of-Thought in Semi-structured Domains". In: *Proceedings of the Natural Legal Language Processing Workshop 2023*. Ed. by D. Preoțiuc-Pietro, C. Goanta, I. Chalkidis, L. Barrett, G. (Spanakis, and N. Aletras. Singapore: Association for Computational Linguistics. 178–191. DOI: 10.18653/v1/2023.nllp-1.18. URL: https://aclanthology.org/2023.nllp-1.18.

Melo, F. (2013). "Area under the ROC Curve". In: *Encyclopedia of Systems Biology*. Ed. by W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota. New York, NY: Springer New York. 38–39. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_209. URL: https://doi.org/10.1007/978-1-4419-9863-7_209.

Mihaylov, T., P. Clark, T. Khot, and A. Sabharwal. (2018). "Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2381–2391.

Miller, G. A. (1995). "WordNet: A Lexical Database for English". *Commun. ACM.* 38(11): 39–41. ISSN: 0001-0782. DOI: 10.1145/219717.219748. URL: https://doi.org/10.1145/219717.219748.

Min, S., J. Michael, H. Hajishirzi, and L. Zettlemoyer. (2020). "AmbigQA: Answering ambiguous open-domain questions". *arXiv preprint arXiv:2004.10645*.

Min, S., E. Wallace, S. Singh, M. Gardner, H. Hajishirzi, and L. Zettlemoyer. (2019a). "Compositional Questions Do Not Necessitate Multi-hop Reasoning". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 4249–4257. DOI: 10.18653/v1/P19-1416. URL: https://aclanthology.org/P19-1416.

Min, S., V. Zhong, R. Socher, and C. Xiong. (2018). "Efficient and Robust Question Answering from Minimal Context over Documents". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics. 1725–1735. DOI: 10.18653/v1/P18-1160. URL: https://aclanthology.org/P18-1160.

Min, S., V. Zhong, L. Zettlemoyer, and H. Hajishirzi. (2019b). "Multi-hop Reading Comprehension through Question Decomposition and Rescoring". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 6097–6109. DOI: 10.18653/v1/P19-1613. URL: https://aclanthology.org/P19-1613.

Mishra, A. and S. K. Jain. (2016). "A survey on question answering systems with classification". *Journal of King Saud University-Computer and Information Sciences*. 28(3): 345–361.

Nair, I., S. Somasundaram, A. Saxena, and K. Goswami. (2023). "Drilling Down into the Discourse Structure with LLMs for Long Document Question Answering". arXiv: 2311.13565 [cs.CL].

Nallapati, R., B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang. (2016). "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Ed. by S. Riezler and Y. Goldberg. Berlin, Germany: Association for Computational Linguistics. 280–290. DOI: 10.18653/v1/K16-1028. URL: https://aclanthology.org/K16-1028.

Nayeem, M. T., T. A. Fuad, and Y. Chali. (2018). "Abstractive unsupervised multi-document summarization using paraphrastic sentence fusion". In: *Proceedings of the 27th International Conference on Computational Linguistics*. 1191–1204.

Nema, P. and M. M. Khapra. (2018). "Towards a Better Metric for Evaluating Question Generation Systems". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics. 3950–3959. DOI: 10.18653/v1/D18-1429. URL: https://aclanthology.org/D18-1429.

Novikova, J., O. Dušek, A. Cercas Curry, and V. Rieser. (2017). "Why We Need New Evaluation Metrics for NLG". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics. 2241–2252. DOI: 10.18653/v1/D17-1238. URL: https://aclanthology.org/D17-1238.

Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe. (2022). "Training language models to follow instructions with human feedback". arXiv: 2203.02155 [cs.CL].

Pan, L., W. Chen, W. Xiong, M.-Y. Kan, and W. Y. Wang. (2021). "Unsupervised Multi-hop Question Answering by Question Generation". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 5866–5880.

Papineni, K., S. Roukos, T. Ward, and W.-J. Zhu. (2002). "Bleu: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

Patel, A., S. Bhattamishra, and N. Goyal. (2021). "Are NLP Models really able to Solve Simple Math Word Problems?" In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2080–2094.

Patel, P., S. Mishra, M. Parmar, and C. Baral. (2022). "Is a Question Decomposition Unit All We Need?" arXiv: 2205.12538 [cs.CL].

Pennington, J., R. Socher, and C. D. Manning. (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. (2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: https://aclanthology.org/N18-1202.

Qi, P., X. Lin, L. Mehr, Z. Wang, and C. D. Manning. (2019). "Answering Complex Open-domain Questions Through Iterative Query Generation". 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)*. Hong Kong, China: Association for Computational Linguistics. 2590–2602. DOI: 10.18653/v1/D19-1261. URL: https://aclanthology.org/D19-1261.

Qiu, L., Y. Xiao, Y. Qu, H. Zhou, L. Li, W. Zhang, and Y. Yu. (2019). "Dynamically Fused Graph Network for Multi-hop Reasoning". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 6140–6150. DOI: 10.18653/v1/P19-1617. URL: https://aclanthology.org/P19-1617.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.* (2019). "Language models are unsupervised multitask learners". *OpenAI blog*. 1(8): 9.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. (2019). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". *CoRR*. abs/1910.10683. arXiv: 1910.10683. URL: http://arxiv.org/abs/1910.10683.

Rahgouy, M., H. B. Giglou, D. Feng, T. Rahgooy, G. Dozier, and C. D. Seals. (2023). "Navigating the Fermi Multiverse: Assessing LLMs for Complex Multi-hop Queries". In:

Rajpurkar, P., J. Zhang, K. Lopyrev, and P. Liang. (2016a). "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics. 2383–2392. DOI: 10.18653/v1/D16-1264. URL: https://aclanthology.org/D16-1264.

Rajpurkar, P., J. Zhang, K. Lopyrev, and P. Liang. (2016b). "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2383–2392.

Rennie, S. J., E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. (2017). "Self-Critical Sequence Training for Image Captioning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Roy, R. S. and A. Anand. (2021). *Question Answering for the Curated Web: Tasks and Methods in QA over Knowledge Bases and Text Collections. Synthesis Lectures on Information Concepts, Retrieval, and Services.* Morgan & Claypool Publishers. ISBN: 978-3-031-79511-4. DOI: 10.2200/S0113ED1V01Y202109ICR076. URL: https://doi.org/10.2200/S0113ED1V01Y202109ICR076.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. (1985). "Learning internal representations by error propagation". *Tech. rep.* California Univ San Diego La Jolla Inst for Cognitive Science.

Sachan, D. S., L. Wu, M. Sachan, and W. Hamilton. (2020). "Stronger Transformers for Neural Multi-Hop Question Generation". *arXiv preprint arXiv:2010.11374*.

Sachan, M. and E. Xing. (2016). "Easy Questions First? A Case Study on Curriculum Learning for Question Answering". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics. 453–463. DOI: 10.18653/v1/P16-1043. URL: https://aclanthology.org/P16-1043.

Saeidi, M., M. Bartolo, P. Lewis, S. Singh, T. Rocktäschel, M. Sheldon, G. Bouchard, and S. Riedel. (2018). "Interpretation of Natural Language Rules in Conversational Machine Reading". In: *EMNLP*.

Samek, W., T. Wiegand, and K.-R. Müller. (2017). "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models". *arXiv preprint arXiv:1708.08296*.

Saparov, A. and H. He. (2023). "Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought". In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=qFVVBzXxR2V.

Schlichtkrull, M., T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling. (2017). "Modeling Relational Data with Graph Convolutional Networks (2017)". *Preprint*.

Schubotz, M., P. Scharpf, K. Dudhat, Y. Nagar, F. Hamborg, and B. Gipp. (2018). "Introducing mathqa: a math-aware question answering system". *Information Discovery and Delivery*.

Scialom, T., B. Piwowarski, and J. Staiano. (2019). "Self-Attention Architectures for Answer-Agnostic Neural Question Generation". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics. 6027–6032. DOI: 10.18653/v1/P19-1604. URL: https://aclanthology.org/P19-1604.

See, A., P. J. Liu, and C. D. Manning. (2017a). "Get To The Point: Summarization with Pointer-Generator Networks". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1073–1083.

See, A., P. J. Liu, and C. D. Manning. (2017b). "Get to the point: Summarization with pointer-generator networks". *arXiv preprint arXiv:1704.04368*.

Seo, M. J., A. Kembhavi, A. Farhadi, and H. Hajishirzi. (2016). "Bidirectional Attention Flow for Machine Comprehension". *CoRR*. abs/1611.01603. arXiv: 1611.01603. URL: http://arxiv.org/abs/1611.01603.

Shao, N., Y. Cui, T. Liu, S. Wang, and G. Hu. (2020). "Is Graph Structure Necessary for Multi-hop Question Answering?" In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 7187–7192.

Shao, N., Y. Cui, T. Liu, S. Wang, and G. Hu. (2021). "Memory augmented sequential paragraph retrieval for multi-hop question answering". *arXiv preprint arXiv:2102.03741*.

Shi, Q., H. Cui, H. Wang, Q. Zhu, W. Che, and T. Liu. (2024). "Exploring Hybrid Question Answering via Program-based Prompting". arXiv: 2402.10812 [cs.CL].

Sidiropoulos, G., N. Voskarides, S. Vakulenko, and E. Kanoulas. (2021a). "Combining Lexical and Dense Retrieval for Computationally Efficient Multi-hop Question Answering". In: *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*. 58–63.

Sidiropoulos, G., N. Voskarides, S. Vakulenko, and E. Kanoulas. (2021b). "Combining Lexical and Dense Retrieval for Computationally Efficient Multi-hop Question Answering". In: *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*. Virtual: Association for Computational Linguistics. 58–63. DOI: 10.18653/v1/2021.sustainlp-1.7. URL: https://aclanthology.org/2021.sustainlp-1.7.

Slobodkin, A., O. Goldman, A. Caciularu, I. Dagan, and S. Ravfogel. (2023). "The Curious Case of Hallucinatory (Un)answerability: Finding Truths in the Hidden States of Over-Confident Large Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by H. Bouamor, J. Pino, and K. Bali. Singapore: Association for Computational Linguistics. 3607–3625. DOI: 10.18653/v1/2023.emnlp-main.220. URL: https://aclanthology.org/2023.emnlp-main.220.

Soares, M. A. C. and F. S. Parreiras. (2020). "A literature review on question answering techniques, paradigms and systems". *Journal of King Saud University-Computer and Information Sciences*. 32(6): 635–646.

Song, L., Z. Wang, M. Yu, Y. Zhang, R. Florian, and D. Gildea. (2018). "Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks". *arXiv preprint arXiv:1809.02040*.

Speer, R., J. Chin, and C. Havasi. (2016). "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. Singh 2002 (2016)". *arXiv preprint arxiv:1612.03975*.

Srivastava, Y., V. Murali, S. R. Dubey, and S. Mukherjee. (2020). "Visual question answering using deep learning: A survey and performance analysis". In: *International Conference on Computer Vision and Image Processing*. Springer. 75–86.

Steen, J. and K. Markert. (2019). "Abstractive timeline summarization". In: *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. 21–31.

Su, D., Y. Xu, W. Dai, Z. Ji, T. Yu, and P. Fung. (2020). "Multi-hop Question Generation with Graph Convolutional Network". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 4636–4647.

Sun, H., W. W. Cohen, and R. Salakhutdinov. (2021). "Iterative Hierarchical Attention for Answering Complex Questions over Long Documents". *arXiv preprint arXiv:2106.00200*.

Talmor, A. and J. Berant. (2018). "The Web as a Knowledge-Base for Answering Complex Questions". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics. 641–651. DOI: 10.18653/v1/N18-1059. URL: https://aclanthology.org/N18-1059.

Tan, Y., D. Min, Y. Li, W. Li, N. Hu, Y. Chen, and G. Qi. (2023). "Evaluation of ChatGPT as a Question Answering System for Answering Complex Questions". arXiv: 2303.07992 [cs.CL].

Tang, D., N. Duan, T. Qin, and M. Zhou. (2017). "Question Answering and Question Generation as Dual Tasks". *CoRR*. abs/1706.02027. arXiv: 1706.02027. URL: http://arxiv.org/abs/1706.02027.

Tang, Y., H. T. Ng, and A. Tung. (2021). "Do Multi-Hop Question Answering Systems Know How to Answer the Single-Hop Sub-Questions?" In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics. 3244–3249. DOI: 10.18653/v1/2021.eacl-main.283. URL: https://aclanthology.org/2021.eacl-main.283.

Thayaparan, M., M. Valentino, V. Schlegel, and A. Freitas. (2019). "Identifying Supporting Facts for Multi-hop Question Answering with Document Graph Networks". In: *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*. 42–51.

Tian, K., E. Mitchell, A. Zhou, A. Sharma, R. Rafailov, H. Yao, C. Finn, and C. D. Manning. (2023). "Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback". arXiv: 2305.14975 [cs.CL].

Trischler, A., T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman. (2016). "NewsQA: A Machine Comprehension Dataset". *CoRR*. abs/1611.09830. arXiv: 1611.09830. URL: http://arxiv.org/abs/1611.09830.

Trivedi, H., N. Balasubramanian, T. Khot, and A. Sabharwal. (2020). "Is Multihop QA in DiRe Condition? Measuring and Reducing Disconnected Reasoning". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics. 8846–8863. DOI: 10.18653/v1/2020.emnlp-main.712. URL: https://aclanthology.org/2020.emnlp-main.712.

Trivedi, H., N. Balasubramanian, T. Khot, and A. Sabharwal. (2023). "Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions". arXiv: 2212.10509 [cs.CL].

Trivedi, H., H. Kwon, T. Khot, A. Sabharwal, and N. Balasubramanian. (2019). "Repurposing Entailment for Multi-Hop Question Answering Tasks". 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)*. 2948–2958.

Tu, M., K. Huang, G. Wang, J. Huang, X. He, and B. Zhou. (2020). "Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. No. 05. 9073–9080.

Van den Oord, A., Y. Li, and O. Vinyals. (2018). "Representation learning with contrastive predictive coding". *arXiv e-prints*: arXiv–1807.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. (2017a). "Attention is all you need". *Advances in neural information processing systems*. 30.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. (2017b). "Attention is all you need". *Advances in neural information processing systems*. 30.

Vedantam, R., C. L. Zitnick, and D. Parikh. (2014). "Cider: consensus-based image description evaluation. CoRR". *arXiv preprint arXiv:1411.5726*.

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. (2018). "Graph Attention Networks". In: *International Conference on Learning Representations*.

Wang, B. (2021). "Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX". https://github.com/kingoflolz/mesh-transformer-jax.

Wang, H., M. Yu, X. Guo, R. Das, W. Xiong, and T. Gao. (2019). "Do Multi-hop Readers Dream of Reasoning Chains?" In: *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. 91–97.

Wang, J., A. Jatowt, M. Färber, and M. Yoshikawa. (2020). "Answering Event-Related Questions over Long-Term News Article Archives". In: *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I*. Vol. 12035. *Lecture Notes in Computer Science*. Springer. 774–789.

Wang, J., A. Jatowt, M. Färber, and M. Yoshikawa. (2021a). "Improving question answering for event-focused questions in temporal collections of news articles". *Inf. Retr. J.* 24(1): 29–54.

Wang, J., A. Jatowt, and M. Yoshikawa. (2021b). "ArchivalQA: A Large-scale Benchmark Dataset for Open Domain Question Answering over Archival News Collections". *CoRR*. abs/2109.03438. arXiv: 2109.03438. URL: https://arxiv.org/abs/2109.03438.

Wang, J., A. Jatowt, and M. Yoshikawa. (2021c). "Event Occurrence Date Estimation based on Multivariate Time Series Analysis over Temporal Document Collections". In: *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM. 398–407.

Wang, S. and J. Jiang. (2016). "Machine comprehension using match-lstm and answer pointer". *arXiv preprint arXiv:1608.07905*.

Wang, S., H. Fang, M. Khabsa, H. Mao, and H. Ma. (2021d). "Entailment as Few-Shot Learner". arXiv: 2104.14690 [cs.CL].

Wang, X., J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. (2023). "Self-Consistency Improves Chain of Thought Reasoning in Language Models". arXiv: 2203.11171 [cs.CL].

Wei, J., X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. (2023). "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models". arXiv: 2201.11903 [cs.CL].

Weiss, D. B., P. Roit, O. Ernst, and I. Dagan. (2021). "Extending Multi-Text Sentence Fusion Resources via Pyramid Annotations". *arXiv preprint arXiv:2110.04517*.

Welbl, J., P. Stenetorp, and S. Riedel. (2018). "Constructing datasets for multi-hop reading comprehension across documents". *Transactions of the Association for Computational Linguistics*. 6: 287–302.

Welleck, S., I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston. (2019). "Neural Text Generation With Unlikelihood Training". In: *International Conference on Learning Representations*.

Williams, A., N. Nangia, and S. Bowman. (2018). "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics. 1112–1122. DOI: 10.18653/v1/N18-1101. URL: https://aclanthology.org/N18-1101.

Williams, R. J. (1992). "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". *Mach. Learn.* 8(3–4): 229–256. ISSN: 0885-6125. DOI: 10.1007/BF00992696. URL: https://doi.org/10.1007/BF00992696.

Wishart, D. S., C. Knox, A. C. Guo, D. Cheng, S. Shrivastava, D. Tzur, B. Gautam, and M. Hassanali. (2008). "DrugBank: a knowledgebase for drugs, drug actions and drug targets". *Nucleic acids research*. 36(suppl_1): D901–D906.

Wu, J., L. Yang, Y. Ji, W. Huang, B. F. Karlsson, and M. Okumura. (2024). "GenDec: A robust generative Question-decomposition method for Multi-hop reasoning". arXiv: 2402.11166 [cs.CL].

Wu, Q., D. Teney, P. Wang, C. Shen, A. Dick, and A. van den Hengel. (2017). "Visual question answering: A survey of methods and datasets". *Computer Vision and Image Understanding*. 163: 21–40.

Wu, Y., M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, D. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. (2016). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". *CoRR*. abs/1609.08144. arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.

Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. (2020). "A comprehensive survey on graph neural networks". *IEEE transactions on neural networks and learning systems*. 32(1): 4–24.

Xiong, W., X. L. Li, S. Iyer, J. Du, P. S. H. Lewis, W. Y. Wang, Y. Mehdad, W. Yih, S. Riedel, D. Kiela, and B. Oguz. (2020). "Answering Complex Open-Domain Questions with Multi-Hop Dense Retrieval". *CoRR*. abs/2009.12756. arXiv: 2009.12756. URL: https://arxiv.org/abs/2009.12756.

Xiong, W., M. Yu, X. Guo, H. Wang, S. Chang, M. Campbell, and W. Y. Wang. (2019). "Simple yet Effective Bridge Reasoning for Open-Domain Multi-Hop Question Answering". In: *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. 48–52.

Xu, W., Y. Deng, H. Zhang, D. Cai, and W. Lam. (2021). "Exploiting Reasoning Chains for Multi-hop Science Question Answering". arXiv: 2109.02905 [cs.CL].

Yadav, V., S. Bethard, and M. Surdeanu. (2019a). "Alignment over Heterogeneous Embeddings for Question Answering". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics. 2681–2691. DOI: 10.18653/v1/N19-1274. URL: https://aclanthology.org/N19-1274.

Yadav, V., S. Bethard, and M. Surdeanu. (2019b). "Quick and (not so) Dirty: Unsupervised Selection of Justification Sentences for Multi-hop Question Answering". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics. 2578–2589. DOI: 10.18653/v1/D19-1260. URL: https://aclanthology.org/D19-1260.

Yadav, V., S. Bethard, and M. Surdeanu. (2020). "Unsupervised Alignment-based Iterative Evidence Retrieval for Multi-hop Question Answering". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 4514–4525.

Yadav, V., S. Bethard, and M. Surdeanu. (2021). "If You Want to Go Far Go Together: Unsupervised Joint Candidate Evidence Retrieval for Multi-hop Question Answering". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4571–4581.

Yan, R., X. Wan, J. Otterbacher, L. Kong, X. Li, and Y. Zhang. (2011). "Evolutionary timeline summarization: a balanced optimization framework via iterative substitution". In: *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. 745–754.

Yang, Z., P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning. (2018). "HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2369–2380.

Yao, K., L. Zhang, T. Luo, L. Tao, and Y. Wu. (2018). "Teaching Machines to Ask Questions". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization. 4546–4552. DOI: 10.24963/ijcai.2018/632. URL: https://doi.org/10.24963/ijcai.2018/632.

Yavuz, S., K. Hashimoto, Y. Zhou, N. S. Keskar, and C. Xiong. (2022). "Modeling Multi-hop Question Answering as Single Sequence Prediction". arXiv: 2205.09226 [cs.CL].

Ye, D., Y. Lin, Z. Liu, Z. Liu, and M. Sun. (2019). "Multi-paragraph reasoning with knowledge-enhanced graph neural network". *arXiv preprint arXiv:1911.02170*.

Yu, J., W. Liu, S. Qiu, Q. Su, K. Wang, X. Quan, and J. Yin. (2020). "Low-Resource Generation of Multi-hop Reasoning Questions". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics. 6729–6739. DOI: 10.18653/v1/2020.acl-main.601. URL: https://aclanthology.org/2020.acl-main.601.

Yu, Y., A. Jatowt, A. Doucet, K. Sugiyama, and M. Yoshikawa. (2021). "Multi-TimeLine Summarization (MTLS): Improving Timeline Summarization by Generating Multiple Summaries". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics. 377–387.

Zaib, M., W. E. Zhang, Q. Z. Sheng, A. Mahmood, and Y. Zhang. (2021). "Conversational question answering: A survey". *arXiv preprint arXiv:2106.00874*.

Zelikman, E., Y. Wu, J. Mu, and N. D. Goodman. (2022). "STaR: Bootstrapping Reasoning With Reasoning". arXiv: 2203.14465 [cs.LG].

Zhang, M., F. Li, Y. Wang, Z. Zhang, Y. Zhou, and X. Li. (2020). "Coarse and Fine Granularity Graph Reasoning for Interpretable Multi-Hop Question Answering". *IEEE Access*. 8: 56755–56765. DOI: 10.1109/ACCESS.2020.2981134.

Zhang, X. and M. Lapata. (2017). "Sentence Simplification with Deep Reinforcement Learning". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics. 584–594. DOI: 10.18653/v1/D17-1062. URL: https://aclanthology.org/D17-1062.

Zhang, Y., P. Nie, A. Ramamurthy, and L. Song. (2021). "Answering Any-hop Open-domain Questions with Iterative Document Reranking". In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 481–490.

Zhao, R., X. Li, S. Joty, C. Qin, and L. Bing. (2023a). "Verify-and-Edit: A Knowledge-Enhanced Chain-of-Thought Framework". arXiv: 2305.03268 [cs.CL].

Zhao, W. X., K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen. (2023b). "A Survey of Large Language Models". arXiv: 2303.18223 [cs.CL].

Zhao, Y., X. Ni, Y. Ding, and Q. Ke. (2018a). "Paragraph-level Neural Question Generation with Maxout Pointer and Gated Self-attention Networks". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics. 3901–3910. DOI: 10.18653/v1/D18-1424. URL: https://aclanthology.org/D18-1424.

Zhao, Y., X. Ni, Y. Ding, and Q. Ke. (2018b). "Paragraph-level Neural Question Generation with Maxout Pointer and Gated Self-attention Networks". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics. 3901–3910. DOI: 10.18653/v1/D18-1424. URL: https://aclanthology.org/D18-1424.

Zhou, B., K. Richardson, X. Yu, and D. Roth. (2022). "Learning to Decompose: Hypothetical Question Decomposition Based on Comparable Texts". arXiv: 2210.16865 [cs.CL].

Zhou, D., N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. V. Le, and E. H. Chi. (2023). "Least-to-Most Prompting Enables Complex Reasoning in Large Language Models". In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=WZH7099tgfM.

Zhu, F., W. Lei, C. Wang, J. Zheng, S. Poria, and T.-S. Chua. (2021). "Retrieving and reading: A comprehensive survey on open-domain question answering". *arXiv preprint arXiv:2101.00774*.