# KNOWLEDGE GRAPH AUGMENTED MULTI-HOP QUESTION ANSWERING USING LARGE LANGUAGE MODELS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

BARIŞ DENIZ SAĞLAM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF DATA INFORMATICS

AUGUST 2024

**Knowledge graph augmented multi-hop question answering using large language models**

submitted by **BARIŞ DENIZ SAĞLAM** in partial fulfillment of the requirements for the degree of **Master of Science in Data Informatics Department, Middle East Technical University** by,

**Date: 29.08.2024**

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

**Name, Surname: Barış Deniz Sağlam**

**Signature :**

# ABSTRACT

**KNOWLEDGE GRAPH AUGMENTED MULTI-HOP QUESTION ANSWERING USING LARGE LANGUAGE MODELS**

Sağlam, Barış Deniz

M.S., Department of Data Informatics

Supervisor: Prof. Dr. Altan Koçyiğit

August 2024, 79 pages

This thesis explores the use of small to medium-sized large language models (LLMs) for multi-hop question answering. As computational resources and latency present significant constraints in real-world applications, smaller language models (LLMs) are often utilized. However, these smaller models generally lack the extensive parametric knowledge and advanced reasoning capabilities possessed by their larger counterparts, such as GPT-4. This research investigates various augmentation strategies, notably the use of knowledge graphs, which provide a structured representation of facts and relationships to compensate for these limitations. This study investigates: whether knowledge graphs improve multi-hop question-answering capabilities of LLMs, the impact of integrating entity-relation triplets with textual content, and whether adaptation methods such as supervised fine-tuning or reinforcement learning with task-specific feedback improve the joint entity-relation extraction performance. The study introduces a novel prompting technique, Connect-the-Entities (CTE), which facilitates the extraction of relevant entity-relations before answering questions, thereby improving performance on the MuSiQue-Ans dataset with reduced computational demand. Additionally, the use of a pre-built knowledge graph as an external knowledge source demonstrates comparable results to baseline systems. Overall, this thesis contributes to the field by demonstrating how smaller LLMs can achieve enhanced question-answering performance through structured knowledge integration and advanced prompting techniques.

Keywords: multi-hop question answering, large language model, knowledge graph, question decomposition, prompt engineering

# ÖZ

**BÜYÜK DİL MODELLERİ KULLANILARAK BİLGİ GRAFİĞİ DESTEKLİ ÇOK ADIMLI SORU CEVAPLAMA**

Sağlam, Barış Deniz

Yüksek Lisans, Bölümü

Tez Yöneticisi: Prof. Dr. Altan Koçyiğit

Ağustos 2024, 79 sayfa

Bu tez, küçük ve orta büyüklükteki büyük dil modellerinin çok adımlı soru yanıtlamadaki kullanımını incelemektedir. Gerçek dünya uygulamalarında hesaplama kaynakları ve gecikme önemli kısıtlamalar sunduğundan, daha küçük dil modelleri genellikle tercih edilir. Ancak, bu daha küçük modeller genellikle GPT-4 gibi daha büyük muadillerinin sahip olduğu geniş parametrik bilgi ve ileri düzeydeki akıl yürütme yeteneklerinden yoksundur. Bu araştırma, bu sınırlamaları telafi etmek için yapılandırılmış bir bilgi ve ilişkiler temsili sunan bilgi grafikleri kullanımı başta olmak üzere çeşitli artırma stratejilerini araştırmaktadır. Bu çalışma şu konuları incelemektedir: bilgi grafiklerinin büyük dil modellerinin çok adımlı soru yanıtlama yeteneklerini iyileştirip iyileştirmediği, varlık-ilişki üçlülerinin metin içeriğiyle bütünleştirilmesinin etkisi ve görevle ilgili geri bildirimle denetimli ince ayar veya pekiştirmeli öğrenme gibi uyarlama yöntemlerinin ortak varlık-ilişki çıkarım performansını iyileştirip iyileştirmediği. Çalışma, ilgili varlık-ilişkilerin soruları yanıtlamadan önce çıkarılmasını kolaylaştıran ve MuSiQue-Ans veri seti üzerinde daha az hesaplama gereksinimi ile performansı artıran, Bağlantı-Kur-Varlıklarla (CTE) adlı yeni bir yönlendirme tekniği sunmaktadır. Ek olarak, harici bir bilgi kaynağı olarak önceden oluşturulmuş bir bilgi grafiğinin kullanılması, temel sistemlere benzer sonuçlar göstermektedir. Genel olarak, bu tez, görece küçük dil modellerin yapılandırılmış bilgi entegrasyonu ve gelişmiş yönlendirme teknikleri ile geliştirilmiş soru yanıtlama performansına nasıl ulaşabileceğini göstererek alana katkıda bulunmaktadır.

Anahtar Kelimeler: çok adımlı soru cevaplama, büyük dil modeli, bilgi grafiği, soru ayrıştırma, komut mühendisliği

To the light of Science in our dark universe

# ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor, Prof. Dr. Altan Koçyiğit, for his continuous support, mentorship, and invaluable insights throughout the course of this research. His guidance was crucial in keeping me on track amidst the challenges encountered, making this work possible. Prof. Dr. Koçyiğit did everything within his power to help me overcome obstacles and ensure that the research continued to move forward.

Special thanks are owed to my family, who have provided unwavering support throughout my life and especially during the last three years of this endeavor. To my mother, Tülay Sağlam, and my father, Ayhan Sağlam, who have nurtured and empowered me to pursue my academic goals, I am eternally grateful. I also extend heartfelt thanks to my brother, Ruşen Erdem Sağlam, and my sister, Çiğdem Sena Sağlam, for their encouragement and support.

I must also thank my friend Berfin Eylül Aydemir, who was a fellow traveler through the peaks and valleys of academic journey and always knew when to send the perfect academic meme to make us laugh instead of cry over our theses.

I would like to extend my heartfelt thanks to Rana, whose emotional support in the final stretch of this journey was invaluable, offering me the strength and encouragement I needed to reach the finish line.

I would like to acknowledge my former managers, Emma Matsubara and Sven Sabas, for their support and understanding, which allowed me the flexibility to attend lectures and manage my study commitments effectively.

I want to express my appreciation to Kerem Nazlıel, who is a research assistant at our institute, for his support in setting up the infrastructure on which many experiments of this study were executed.

I would also like to express my gratitude to Mrs. Ayşe Nur Özdere Yüksel from our department's student affairs office, who kindly assisted me on administrative issues during my master's program.

The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources). I thank TUBITAK ULAKBIM for their support.

# TABLE OF CONTENTS

ABSTRACT . . . . . iv

ÖZ . . . . . v

DEDICATION . . . . . vi

ACKNOWLEDGMENTS . . . . . vii

TABLE OF CONTENTS . . . . . viii

LIST OF TABLES . . . . . xii

LIST OF FIGURES . . . . . xiii

LIST OF ABBREVIATIONS . . . . . xiv

CHAPTERS

1 INTRODUCTION . . . . . 1

1.1 Research Questions . . . . . 2

1.2 Contributions of the Study . . . . . 2

1.3 Organization of the Thesis . . . . . 2

2 BACKGROUND . . . . . 5

2.1 Open-Domain Question Answering . . . . . 5

2.2 Large Language Models . . . . . 6

2.3 Auto-regressive Text Generation . . . . . 7

2.4 Adaptation of LLMs . . . 8
2.5 Knowledge Graph . . . 9
2.6 Knowledge Graph Construction . . . 10
2.7 Enhancing LLM Accuracy and Reliability through RAG . . . 10
2.8 Reinforcement Learning in LLMs . . . 11
3 RELATED WORK . . . 15
3.1 Multi-hop question answering . . . 15
3.2 Question decomposition . . . 17
4 METHOD . . . 19
4.1 Scope . . . 19
4.2 Knowledge Graph Construction . . . 20
4.2.1 Zero-shot Prompting . . . 21
4.2.2 Few-shot Prompting . . . 21
4.2.3 Supervised Fine-tuning . . . 23
4.2.4 Reinforcement Learning based Fine-tuning . . . 23
4.3 Question Decomposition . . . 24
4.4 Information Retrieval . . . 24
4.4.1 Sparse retriever . . . 24
4.4.2 Dense retriever . . . 25
4.5 Knowledge Graph based Question Answering . . . 26
4.6 Prompting Techniques for Question Answering . . . 26
4.7 Evaluation Metrics . . . 29
4.7.1 Information Retrieval . . . 30

4.7.2 Joint Entity Relation Extraction . . . . . 31
4.7.3 Question Answering . . . . . 32
4.7.3.1 Exact Match (EM) . . . . . 32
4.7.3.2 F1 Score . . . . . 32
4.7.3.3 Examples . . . . . 33
5 EXPERIMENTS . . . . . 35
5.1 Datasets . . . . . 35
5.1.1 MuSiQue . . . . . 35
5.1.2 WebNLG Dataset . . . . . 36
5.2 Experimental Setup . . . . . 37
5.2.1 Joint Entity Relation Extraction . . . . . 37
Supervised Fine-Tuning (SFT) . . . . . 37
Reinforcement Learning from Task Feedback (RLTF) . . . . . 38
5.2.2 Question Answering . . . . . 39
5.2.3 Compared Models . . . . . 39
5.3 Results . . . . . 40
5.3.1 Knowledge Graph based Question Answering . . . . . 40
5.3.2 Question Answering with CTE prompting . . . . . 45
5.3.3 Evaluating Impact of Individual Components and Parameters . . . . . 45
5.3.3.1 Impact of Prompting Technique on Question Answering . . . . . 45
5.3.3.2 Impact of Temperature Value on Question Answering . . . . . 47
5.3.3.3 Impact of LLM Adaptation Techniques for JERX . . . . . 48
5.3.3.4 Retrieval Performances of Different Retrievers . . . . . 49

5.4 Discussion . . . . . 50

6 CONCLUSIONS . . . . . 53

6.1 Conclusion . . . . . 53

6.2 Limitations . . . . . 54

6.3 Future Works . . . . . 55

APPENDICES

A EXTRA MATERIALS FOR KNOWLEDGE GRAPHS . . . . . 67

A.1 Visualizations for knowledge graph . . . . . 67

B EXPERIMENTS . . . . . 71

B.1 Samples for validation-mini subset . . . . . 71

B.2 Experiment results for KGQA . . . . . 72

B.3 Experiment results for question answering based on paragraphs . . . . . 74

B.4 Experiment results for impact of prompting on question answering . . . . . 76

B.5 Experiment results for impact of temperature on Question Answering . . . . . 76

B.6 Experiment results for impact of LLM adaptation techniques for JERX . . . . . 78

# LIST OF TABLES


Table 1 Examples for the comparison of string equivalence criteria . . . . . 31

Table 2 Examples of Exact Match and F1 Score calculations for various predictions . . . . . 33

Table 3 Question answering performances of official baselines on MuSiQue-Ans validation-mini subset . . . . . 39

Table 4 Question answering performances of state-of-the-art models on MuSiQue-Ans test set 40

Table 5 Evaluation results for knowledge graph based question answering . . . . . 42

Table 6 Impact of retrieval on the performance of knowledge graph based question answering 43

Table 7 Evaluation Results for Paragraph based Question Answering . . . . . 46

Table 8 Evaluation results for question answering with different prompting techniques . . . . . 46

Table 9 Comparison of output token counts for CTE and CoT prompting techniques . . . . . 47

Table 10 Comparison of CTE and CoT prompting techniques on an example data point . . . . . 47

Table 11 Question answering performances with different temperature values . . . . . 48

Table 12 JERX performances on WebNLG validation set . . . . . 49

Table 13 Question answering performances with different JERX models . . . . . 49

Table 14 Comparison of retrieval algorithms by recall . . . . . 50

Table 15 All experiment results for knowledge graph based question answering . . . . . 72

Table 16 All experiment results for paragraph based question answering . . . . . 75

Table 17 All experiment results for question answering with different prompting techniques . . 77

Table 18 All experiment results for question answering with different temperature values . . . . . 77

Table 19 All experiment results for question answering with different JERX models . . . . . 79

# LIST OF FIGURES


Figure 1 Zero-shot prompt for JERX . . . . . 21

Figure 2 Few-shot prompt for JERX . . . . . 22

Figure 3 An example data point to SFT for JERX . . . . . 23

Figure 4 Few-shot prompt for question decomposition . . . . . 25

Figure 5 Architecture of knowledge graph-based question answering system . . . . . 26

Figure 6 Standard prompt for question answering . . . . . 27

Figure 7 Zero-shot Chain-of-Thought prompt for question answering . . . . . 27

Figure 8 Few-shot Chain-of-Thought prompt for question answering . . . . . 28

Figure 9 Few-shot Connect-the-Entities question-answering prompt . . . . . 29

Figure 10 An example data point from MuSiQue-Ans dataset . . . . . 36

Figure 11 An example data point from WebNLG . . . . . 36

Figure 12 Mean reward during RL training . . . . . 38

Figure 13 An example failure of KGQA due to temporal dynamics . . . . . 44

Figure 14 High-Level Network Layout of a Knowledge Graph . . . . . 68

Figure 15 Detailed View of a Knowledge Graph . . . . . 69

# LIST OF ABBREVIATIONS

| | |
|---|---|
| API | Application Programming Interface |
| BERT | Bidirectional Encoder Representations from Transformers |
| COS | Chain-of-Skills |
| CoT | Chain-of-Thought |
| CTE | Connect-the-Entities |
| DQN | Deep Q-Network |
| EM | Exact Match |
| GPT | Generative Pre-trained Transformer |
| ICL | In-context Learning |
| IR | Information Retrieval |
| JERX | Joint Entity-Relation Extraction |
| KG | Knowledge Graph |
| KGC | Knowledge Graph Construction |
| KGQA | Knowledge Graph Question Answering |
| LM | Language Model |
| LLM | Large Language Model |
| LORA | Low-Rank Adaptation |
| NLP | Natural Language Processing |
| NLG | Natural Language Generation |
| ODQA | Open-Domain Question Answering |
| POMDP | Partially Observable Markov Decision Process |
| PPO | Proximal Policy Optimization |

| | |
|---|---|
| QA | Question Answering |
| QLORA | Quantized Low-Rank Adaptation |
| RAG | Retrieval-Augmented Generation |
| RC | Reading Comprehension |
| RL | Reinforcement Learning |
| RLAIF | Reinforcement Learning from AI Feedback |
| RLHF | Reinforcement Learning from Human Feedback |
| RLTF | Reinforcement Learning from Task Feedback |
| RQ | Research Question |
| SFT | Supervised Fine-Tuning |

# CHAPTER 1

# INTRODUCTION

The emergence of large language models (LLMs) [1] has significantly advanced the field of natural language processing (NLP). Among these, GPT-4 stands out as a state-of-the-art model, which is believed to encompass over one trillion parameters [2]. Such expansive models have been shown to exhibit emergent and advanced capabilities that scale with their size. However, the benefits of increased size come at the cost of higher computational resources for both training and inference. Consequently, for reasons of latency and cost-efficiency, smaller models may be preferred in certain practical applications.

To enhance the capabilities of these smaller models, various strategies have been employed. One notable approach is the use of tools [3] and retrieval-augmented generation (RAG) [4] techniques. These methods aim to supplement the inherent limitations of the parametric knowledge of smaller models by integrating external knowledge sources [4]. Knowledge graphs (KGs) are particularly effective in this regard, providing accurate, explicit knowledge and the capability for symbolic reasoning, which leads to interpretable results, and their dynamic and domain-specific nature enhances the retrieval capabilities and overall performance of RAG frameworks [5].

Question answering (QA) tasks, particularly those focused on reading comprehension, have long been central to evaluating the capabilities of language models (LMs). Reading comprehension in QA involves answering queries based on a given textual context, which tests the model's ability to parse and interpret written information.

As language models have evolved, the complexity of QA tasks has increased to better test and stretch their capabilities. A notable advancement in this area is the introduction of multi-hop question answering. This task requires a language model to gather and synthesize information from multiple passages or data points to answer a question correctly. It represents a significant leap from single-passage reading comprehension, as it demands a deeper understanding and more sophisticated reasoning across several pieces of text.

Moreover, recent datasets such as SQUAD 2.0 [6] and MuSiQue [7] have been designed to further enhance the difficulty of these tasks by integrating irrelevant or distracting content within the context. This addition challenges the models to discern relevant from irrelevant information effectively, a critical skill for real-world applications where data is rarely clean or perfectly structured.

This thesis investigates the effectiveness of knowledge graph augmented multi-hop question answering with LLMs. It covers several techniques for knowledge graph construction with LLMs and also for prompt engineering in the context of question answering.

## 1.1 Research Questions

**RQ1:** *Do knowledge graphs enhance the multi-hop question-answering performance of LLMs?*
It is posited that the multi-hop question-answering capabilities of a language model can be enhanced through the use of knowledge graphs. The structured representation of knowledge in graphs facilitates multi-hop reasoning by enabling pathfinding mechanisms across the knowledge graph.

**RQ2:** *What is the impact of incorporating entity-relation triplets with the raw content of paragraphs on multi-hop reasoning?*
This research hypothesizes that integrating entity-relation triplets with the raw textual content of paragraphs enhances the multi-hop question-answering performance compared to a baseline system that only utilizes paragraphs.

**RQ3:** *Do fine-tuning methods such as supervised and reinforcement learning with task-specific feedback improve the joint entity-relation extraction and overall question-answering performance of the system?*
It is hypothesized that a language model, when fine-tuned with supervised or reinforcement learning approaches that incorporate QA task feedback, produces entity-relation triplets of higher quality and completeness, thereby improving the overall question-answering performance of the system.

## 1.2 Contributions of the Study

The main contribution of this study is to reveal the effectiveness of knowledge-graph based multi-hop question answering with LLMs. The study particularly focuses on small to medium size LLMs such as Meta-Llama-3-8B-Instruct and GPT-3.5 Turbo rather than larger models like GPT-4. This is because smaller models are more practical for real-world applications due to their lower computational cost, which results in lower monetary cost and latency.

The study proposes a new prompting technique, named Connect-the-Entities (CTE), for multi-hop question answering. It instructs LLM to extract relevant entity-relation-entity triplets from the context before answering the question. We show that this technique improves question answering performance on the MuSiQue-Ans dataset [7] with less computation cost.

The study also explores the use of a knowledge graph as an external knowledge source for multi-hop question answering. We show that using a knowledge graph, which is pre-built with an LLM on paragraphs of a sample, as a knowledge base for question answering yields comparable performances with baselines.

## 1.3 Organization of the Thesis

This thesis is structured to systematically explore the effectiveness of knowledge graphs and large language models (LLMs) for multi-hop question answering. Chapter 1 introduces the study, presenting the research questions, contributions, and the overall structure of the thesis. Chapter 2 provides the necessary background, including discussions on open-domain question answering, LLMs, knowledge graphs, and reinforcement learning within LLMs. Chapter 3 reviews related work, particularly focus-

ing on multi-hop question answering and question decomposition strategies. Chapter 4 describes the methodology employed in this study, covering aspects such as knowledge graph construction, information retrieval, and various prompting techniques. Chapter 5 presents the experiments conducted, including the dataset, experimental setups, a detailed analysis of results, and the discussion of the results. Finally, Chapter 6 concludes the thesis, identifies the limitations of the study, and outlines future research directions.

# CHAPTER 2

# BACKGROUND

## 2.1 Open-Domain Question Answering

Open-domain Question Answering (ODQA) is a task where models respond to fact-based questions presented in natural language. The responses are based on objective truths, which allows for straightforward evaluation of the model's performance. The term "open-domain" indicates that these questions are asked without providing any specific contextual information. If both the question and its related context are given, the challenge then shifts to Reading Comprehension (RC) [8].

A central architecture in ODQA is the *Retriever-Reader* model [9]. The *Retriever* functions as an information retrieval (IR) system, selecting documents likely to contain the answer to a given question and ranking them by relevance. The *Reader*, on the other hand, processes these documents to deduce the final answer, differentiating ODQA from simpler IR tasks by its ability to interpret and synthesize information from multiple sources [9].

Retrievers are categorized into three types: *Sparse*, *Dense*, and *Iterative*. Sparse Retrievers, such as those using TF-IDF or BM25 [10] algorithms, rely on keyword matching to find relevant documents [11, 12]. Dense Retrievers leverage deep learning to understand semantic similarities, providing substantial improvements over traditional methods by embedding questions and passages in a shared vector space for more nuanced retrieval [13]. Iterative Retrievers enhance this process by repeatedly refining their search parameters based on feedback from initial results, proving particularly effective for complex, multi-hop questions [9].

The Reader component can be implemented as either *Extractive* or *Generative*. Extractive Readers identify specific answer spans within the retrieved text, while Generative Readers construct answers in natural language, often using advanced language models such as Bidirectional Encoder Representations from Transformers (BERT) [14] and Generative Pre-trained Transformer (GPT) [15]. This distinction is crucial as it affects both the flexibility of the response and the system's ability to deal with ambiguous or poorly structured questions [9].

Recent studies highlight the significant impact of context on the performance of generative language models in ODQA. By augmenting queries with contextually relevant information, models not only improve their accuracy but also become adept at handling complex query scenarios. Research such as the Retrieval-Augmented Generation (RAG) model demonstrates that a blend of retriever and generative approaches can outperform traditional extractive methods [4, 16].

### 2.2 Large Language Models

Large language models (LLMs) have emerged as a transformative force in the field of artificial intelligence, showcasing unprecedented capabilities across diverse set of tasks [2] that have sparked both excitement and innovation across a broad spectrum of applications [17]. Among these, ChatGPT stands out as a prime example of how these models can facilitate interactive and engaging conversations, simulating a human-like understanding of language nuances.

The generalization capabilities of LLMs across diverse tasks such as question answering, machine translation, summarization, and conversational AI underscore their versatility and the breadth of their potential impact. This adaptability is largely attributable to the models' architecture and the scale of data on which they are trained. Specifically, the transformer architecture, introduced by Vaswani et al. [18], and further exemplified by BERT [14] and GPT [15], provides a foundation that supports the models' ability to learn and process the complexities of human language.

One of the most remarkable aspects of LLMs is their capacity for in-context learning (ICL) [19], allowing them to understand and respond to new information presented in their prompts without the need for explicit retraining [20]. This capability has been pivotal in enabling LLMs to perform a wide array of tasks directly from text inputs, showcasing their potential for rapid adaptation and application in real-world scenarios.

The introduction of a unified text-to-text framework for natural language processing tasks has revolutionized the versatility and effectiveness of transfer learning in the field [21]. This innovative approach, which processes all language tasks by converting inputs and outputs into a uniform text format, has greatly simplified the application of models across different challenges without necessitating task-specific architectures. The exploration into varied pre-training objectives, architectures, and transfer methods has significantly advanced the understanding of optimizing models for a comprehensive range of language understanding tasks. Such advancements have not only extended the limits of what transfer learning can achieve but also paved the way for further innovations, including models fine-tuned with human feedback to better align with user instructions. This shift towards a more unified method in handling diverse NLP challenges has been instrumental in enhancing model performance and efficiency across the board.

The integration of human feedback into the training process of language models [22] represents a significant advancement in the field of NLP. This approach, termed instruction tuning, involves fine-tuning pre-existing language models like GPT-3 with datasets curated from human feedback to align the models more closely with human intent. The process includes collecting demonstrations of desired model behavior based on prompts and subsequently refining the models through reinforcement learning derived from human evaluations of model outputs. The result, InstructGPT [22], demonstrates a notable improvement in the model's ability to follow user instructions, producing outputs that are more truthful, less toxic, and generally more aligned with what users find helpful, even when compared to outputs from significantly larger models without such fine-tuning.

As these models scale, emergent capabilities become increasingly apparent, highlighting a phenomenon where the sum of the model's abilities exceeds what might be expected from its individual components [19]. This emergence of new abilities with scale is not merely a function of model architecture but also of the unprecedented scale of unsupervised pre-training on vast corpora of text data. Such scaling has led to models that can generate coherent and contextually relevant text across various domains, a

capability that was further advanced by works like LaMDA [23] and PALM [24], which build on the transformer architecture and unsupervised learning paradigms.

Despite these advances, LLMs are not without their challenges. A notable issue is their propensity for "hallucinations," or generating information that is plausible but factually incorrect [25, 26, 27]. This phenomenon poses significant hurdles in applications where accuracy and reliability are paramount. Various mitigating techniques have been proposed and implemented to address these challenges. Few-shot prompting, chain-of-thought [28, 29], and least-to-most prompting [30] strategies have shown promise in guiding LLMs towards more accurate and reliable outputs by structuring the input prompts to elicit more thoughtful and reasoned responses.

Furthermore, Retrieval-Augmented Generation (RAG) [4] offers a compelling solution to the hallucination problem by integrating external knowledge sources directly into the generation process. RAG ensures that the model's responses are not only informed by its internal knowledge base but are also grounded in external, verifiable information, thus significantly enhancing the accuracy and relevance of its outputs [31].

The advancements in LLMs, while impressive, underscore the importance of continuous research and development in the field. As these models become more integrated into societal functions, the imperative to refine their accuracy, reduce biases, and improve their reliability becomes increasingly critical.

### 2.3 Auto-regressive Text Generation

Auto-regressive text generation is a fundamental technique employed by large language models (LLMs) for producing text sequentially. In this method, each new token is generated based on the sequence of tokens previously produced, enabling the model to maintain contextual relevance and coherence over extended passages. The foundational concepts of auto-regressive generation in neural networks were introduced by Bengio et al. [32], setting the stage for subsequent developments in language modeling.

While auto-regressive text generation is prominently featured in LLMs, which are based on transformers architecture [18], it is not exclusive to these models. It is applicable across various sequence modeling frameworks such as Recurrent Neural Networks (RNNs) [33], Long Short-Term Memory networks (LSTMs) [34], and Sequence-to-Sequence (Seq2Seq) models [35]. These models inherently depend on their ability to capture temporal dependencies, making auto-regression a natural approach for generating sequences that require contextual continuity.

A variety of sampling methods are employed in auto-regressive text generation to control the randomness and variability of the output. These methods include temperature-based sampling, top-k sampling, and top-p sampling, each playing a crucial role in the text generation process.

The temperature parameter $t$ serves as a crucial hyperparameter in temperature-based sampling, regulating the randomness inherent in the sampling process [36]. This parameter modifies the softmax function—a transformation applied to the neural network's output logits, converting them into a probability distribution where the total sums to one [37]. Specifically, the temperature $t$ influences the distribution's shape by adjusting the spread of probability mass. For $t > 1$, the distribution becomes flatter, reducing higher probabilities while enhancing lower ones, thereby increasing both entropy and perplexity. Conversely, $t < 1$ sharpens the distribution, concentrating probability mass around more

likely outcomes and reducing randomness. Typical values for $t$ range from 0 to 2, with $t = 0$ leading to greedy sampling, where the token with the highest probability is consistently selected. This parameter thus not only governs the degree of uncertainty in generated outputs but also impacts the model's creative capabilities in generative tasks.

Top-k sampling limits the model's choices to the k most probable next words, providing a balance between randomness and restriction. Top-p sampling or nucleus sampling dynamically selects a set of next-word candidates that cumulatively surpass a predefined probability threshold, enhancing the adaptability of the generation process to the complexity of the context. These sampling strategies are pivotal in managing the trade-off between diversity and coherence in generated text and are instrumental in optimizing computational efficiency during the generation process.

In conclusion, the implementation of sampling methods and the careful adjustment of hyperparameters such as the temperature parameter are critical in controlling the randomness and variance in auto-regressive text generation. These techniques enable a balance between creativity and coherence in the output text, while also optimizing the model's performance for various applications, from natural language generation to automated code completion. By adjusting these parameters, developers can effectively tailor the behavior of large language models, ensuring that the generated text is both varied in expression and contextually appropriate, thus enhancing the utility and relevance of auto-generated content.

In summary, the strategic manipulation of sampling methods and hyperparameters such as the temperature $t$ is crucial in auto-regressive text generation, particularly for applications like question answering using LLMs. For such systems, setting a lower temperature value helps reduce randomness, thereby improving the precision and relevance of responses. Importantly, maintaining a non-zero temperature ensures that there is variability in the outputs across multiple runs of the same query, which is essential for thoroughly evaluating the robustness of the models. Conversely, applications such as content creation and conversational bots may benefit from a higher temperature setting, which fosters more diverse and engaging interactions. This adaptability underscores the LLMs' capacity to balance coherence and creativity across various applications, highlighting their extensive utility in diverse text generation tasks.

### 2.4 Adaptation of LLMs

Prompt engineering has emerged as a significant field following the success of large language models, focusing on enhancing the effectiveness of these models in specific tasks or domains. The essence of prompt engineering lies in the design of prompts that may include instructions, examples, or context, all within the constraints of the model's context window size. It seeks to leverage the inherent capabilities of language models by guiding them towards desired outcomes through tailored inputs.

Recent studies have demonstrated that certain prompting techniques can significantly improve the reasoning capabilities of language models [28, 29, 30]. These techniques play a pivotal role in enabling models to process and interpret information in ways that align more closely with human reasoning patterns.

In contrast to the manual crafting of prompts in token space, recent methodologies have aimed to achieve prompt engineering outcomes within vector space. This shift represents a move away from tra-

ditional, text-based prompting towards a more nuanced approach involving the optimization of prompt vectors. By optimizing a prompt vector, as opposed to crafting text prompts, researchers can learn a task-specific prompt via supervised training while keeping the model parameters frozen, thus maintaining the integrity and knowledge encapsulated within the pre-trained model [38, 39, 40].

The advent of large language models has rendered the fine-tuning of all model parameters increasingly impractical for domain adaptation. Previously, fine-tuning pre-trained language models was a widely accepted method for adapting these models to specific domains [41]. However, the growing interest in parameter-efficient fine-tuning methods has led to the exploration of alternatives that require fewer adjustments to the model's parameters, thereby reducing computational costs and preserving model generalizability. One notable method is Low-Rank Adaptation (LoRA) [42], which introduces trainable rank decomposition matrices into each layer of the Transformer architecture. This approach significantly reduces the number of trainable parameters needed for downstream tasks without compromising model performance [42]. Building on this foundation, QLoRA further optimizes the fine-tuning process by backpropagating gradients through a quantized, frozen model, thus lowering resource requirements and enabling more efficient model adaptation [43].

### 2.5 Knowledge Graph

The concept of a knowledge graph was popularized by Google in 2012 [44] when they introduced their "Knowledge Graph" to enhance their search engine's results with information gathered from a variety of sources, displayed in an informative box beside search results. This development marked a significant shift towards more semantic search capabilities, where understanding the intent and context behind user queries became as important as the content of the queries themselves.

The utility of knowledge graphs extends beyond search engines. In the realm of artificial intelligence, they facilitate more nuanced understanding and generation of natural language, contributing to the development of more sophisticated question answering systems by providing semantically structured information interpretable by computers [45].

A knowledge graph (KG) can be defined as a structured representation that maps real-world entities and their connections, outlines potential categories and relationships, enables linking of diverse entities, and covers a wide range of subjects [46]. Formally, KGs encapsulate structured knowledge through the formulation of triples $KG = \{(s, p, o) \subseteq E \times R \times E\}$, where $E$ and $R$ respectively symbolize the sets of entities and relations [47]. However, this representation as a set of triples can sometimes be insufficient, particularly when statements about statements are needed, which requires the use of reification. Reification allows a triple itself to be treated as an entity so that additional information can be provided about it. For example, to express that "Alice believes that Bob said 'The sky is blue,'" we need to create a statement about another statement. In this case, reification would involve creating a new entity to represent the statement "Bob said 'The sky is blue'" and then linking it to Alice's belief [48, 49]. In this study, reification is not utilized because it is not necessary for answering multi-hop questions; hence, the knowledge graph is represented as a set of simple triples.

KGs become essential for their explicit representation of precise knowledge across various applications [50]. They are known for symbolic reasoning, leading to clear results [51]. Unlike LLMs, KGs can also be updated continuously with new information [52]. Moreover, domain-specific KGs offer accurate knowledge for particular fields [53].

### 2.6 Knowledge Graph Construction

Knowledge Graph Construction (KGC) relies on integrating data from multiple sources, often requiring manual input for tasks such as source identification and ontology creation. Quality metrics for KGs include correctness, freshness, comprehensiveness, and succinctness. They ensure that the graph is accurate, up-to-date, complete, and focused. Achieving high standards in these areas makes a KG a reliable and efficient resource for information retrieval and analysis.

Automated KGC from text is becoming increasingly crucial as systems relying on these graphs expand and the volume of data grows. This construction process involves several key NLP tasks: identifying named entities, extracting relationships between these entities, and resolving when different terms refer to the same entity [54].

Historically, deep neural networks have played a pivotal role in advancing knowledge graph construction, and they offer significant improvements in efficiency and accuracy over earlier methods [55]. The approaches vary, with some requiring the development of specialized models for specific tasks [56], while others adapt pre-existing language models through fine-tuning [57, 58] or by employing prompt engineering [59].

These methods fundamentally rely on language models to identify entities and their interrelations within text, laying the groundwork for the knowledge graph. While certain approaches advocate for separating the tasks of entity identification and relation extraction [58, 59], others support a joint approach, citing its simplicity and effectiveness in streamlining the construction process [57, 60].

### 2.7 Enhancing LLM Accuracy and Reliability through RAG

While LLMs have been successful in storing vast amounts of factual knowledge within their parameters, they also present challenges, such as the propensity for generating hallucinations and relying on outdated information [61]. To address these limitations, the technique of Retrieval-Augmented Generation (RAG) has emerged as a promising solution, significantly enhancing the accuracy and reliability of LLMs when supplemented with information from external data sources during inference [4]. The adoption of RAG has gained traction, particularly for applications that leverage the capabilities of LLMs, by merging their intrinsic knowledge with dynamic, external databases [31].

RAG utilizes three main types of external data sources: dense vector indices, relational databases, and knowledge graphs. Each source offers a unique approach to storing and accessing information, thereby enriching the context available to LLMs during the generation process [4] [5] [31]. The process of RAG is methodically structured into three sequential steps: retrieval, augmentation, and generation. Initially, a retriever component identifies relevant information from the chosen external data source. This information is then used to augment the context provided to the language model, which, in turn, generates outputs based on this enriched context [4].

Further exploration into the capabilities of LLMs has shown that their performance on knowledge-intensive tasks is notably improved when they utilize external tools to lookup factual information. This approach not only enhances their accuracy but also their applicability in complex reasoning tasks [3]. Specifically, in the domain of question answering over knowledge graphs (KGQA), various method-

ologies have been developed to navigate the challenges posed by the incompleteness of knowledge graphs. Traditional methods that rely solely on reasoning over sub-graphs extracted for specific questions often falter due to the inherent limitations of the graphs [62].

To mitigate these challenges, recent studies have advocated for the integration of LLMs with RAG techniques, enabling them to reason over relevant entities and relations despite the incompleteness of the underlying knowledge graphs. This hybrid approach has been shown to significantly enhance the performance of LLMs, particularly in addressing complex queries that necessitate advanced reasoning capabilities, such as count, intersection, or multi-hop reasoning operations [63] [64]. By leveraging the strengths of both LLMs and external knowledge sources, RAG stands as a robust framework that not only improves the quality of generated responses but also contributes to the ongoing evolution of language models towards more accurate, reliable, and knowledge-rich systems.

Luo et al. [61] introduce Reasoning on Graphs (RoG), a method that integrates large language models (LLMs) with knowledge graphs (KGs) to enhance reasoning fidelity and interpretability. RoG addresses limitations in current KG-based reasoning approaches by formulating reasoning as an optimization problem that maximizes the probability of deriving answers through structured relation paths. The method employs a planning-retrieval-reasoning framework, which involves joint training of LLMs for generating and optimizing these relation paths, thus improving both the accuracy and transparency of the reasoning process. The approach is evaluated using the WebQuestionSP and Complex WebQuestions datasets with Freebase as the knowledge source, showing promising results in reasoning capability and interpretability.

KAPING, a framework for Knowledge-Augmented Language Model Prompting, addresses the limitations of LLMs' parametric knowledge by utilizing external knowledge from knowledge graphs (KGs) to enhance the accuracy of zero-shot question answering tasks [63]. The method employs semantic retrieval techniques to filter out irrelevant triples based on their similarity to the input question, ensuring that only relevant information is used during the LLM prompting process. KAPING was evaluated on QA datasets using the Freebase and Wikidata knowledge graphs, demonstrating its effectiveness in overcoming the static and potentially outdated knowledge inherent in LLMs.

### 2.8 Reinforcement Learning in LLMs

Reinforcement Learning (RL) [65] is a domain of machine learning where an agent learns to make good decisions by performing proper actions in an environment to achieve a goal. The agent learns from the consequences of its actions, rather than from explicit instructions, by maximizing cumulative rewards. It is effective in situations where the decision-making process can be modeled as a sequence of actions leading to an outcome, especially in complex environments where explicit programming of all possible scenarios is impractical. RL has shown significant success in various fields such as robotics, game playing, autonomous vehicles, and personalized recommendations, where adaptability and learning from interaction are crucial [65].

Deep Reinforcement Learning (Deep RL) combines reinforcement learning with deep learning, integrating neural networks to approximate the decision-making functions of an agent. This approach enables the handling of high-dimensional, complex environments that traditional RL methods struggle with. Deep RL has been pivotal in achieving human-level performance in tasks like playing video games, navigating robots through unpredictable terrains, and optimizing energy efficiency in data cen-

ters. Its effectiveness stems from the ability of deep neural networks to extract and learn features from raw input data, thereby significantly enhancing the learning capabilities of RL agents in environments where the state and action spaces are too large for traditional methods to be feasible. Notable breakthroughs, such as the development of Deep Q-Networks (DQN) and policy gradient methods, have underscored the potential of Deep RL to solve a wide array of challenging problems across different domains [66, 67].

A significant advancement in the field has been the integration of human feedback into the learning process. This approach has been pivotal in addressing complex tasks where direct demonstration of desired behavior by humans is challenging. Christiano et al. demonstrated that deep reinforcement learning from human preferences, despite its scalability and efficiency challenges, could effectively navigate complex tasks by learning a reward function from human feedback, rather than relying on direct human feedback as a reward [68]. This methodology marks a critical shift towards utilizing human judgment in a more scalable and efficient manner, enabling the reinforcement learning models to achieve tasks previously deemed too complex for direct reinforcement learning methods.

The concept of Reinforcement Learning from Human Feedback (RLHF) further refines this approach. Ouyang et al. illustrated how RLHF significantly aligns models to follow instructions more accurately, enhancing their truthfulness and safety [69]. This method involves collecting human rankings of model-generated texts, training a reward function to predict these rankings, and finally fine-tuning the model with reinforcement learning using the learned reward function. The scalability of RLHF, particularly for language models, stems from the reduced labor in evaluating generated texts compared to creating desired texts, making it a more viable option than supervised fine-tuning for enhancing model performance.

In seeking alternatives to human feedback, which is often costly and not always scalable, the field has begun exploring Reinforcement Learning from AI Feedback (RLAIF). This approach aims to eliminate the need for human feedback in the RLHF process by employing models to evaluate the generated text on various criteria, such as harmfulness and truthfulness [70]. Additionally, Kwon et al. introduce an approach that further simplifies reward design in RL by utilizing an LLM like GPT-3 as a proxy for the reward function. By allowing users to specify desired behaviors through textual prompts, this method facilitates the alignment of RL agents with user objectives, showcasing the potential of language models in enhancing reward design efficiency and effectiveness in various tasks [71]. RLAIF represents a significant step towards automating the feedback process in reinforcement learning, potentially increasing the efficiency and scalability of training reinforcement learning models.

Moreover, Singh et al. have shown that for tasks where the correctness of model-generated text can be automatically verified, such as math problems or coding tasks, binary feedback can serve as an effective substitute for human feedback in training models with reinforcement learning [72]. This suggests a promising direction for reducing reliance on expensive and time-consuming human feedback by leveraging binary or automated feedback mechanisms in suitable domains.

In addition to the foundational advancements in leveraging human and AI feedback within reinforcement learning, several practical applications and challenges highlight the field's dynamic nature and its potential for future growth. The integration of RLHF into practical scenarios, such as improving dialogue agents and summarization tasks, underscores the method's versatility and effectiveness. For instance, the use of RLHF has led to significant improvements in language models' ability to generate human-like dialogues and produce summaries that align more closely with human preferences

[73, 74]. These successes not only validate the approach but also open avenues for its application across a broader range of language processing tasks.

Moreover, the challenges of aligning language models with human preferences have spurred innovative solutions, such as targeted human judgments and dynamic planning in open-ended dialogues. Glaese et al. and Cohen et al. explored these areas, demonstrating that targeted feedback and advanced planning techniques could further refine the interaction between humans and language models, enhancing the models' responsiveness and relevance in conversations [75, 76]. These advancements are crucial for developing dialogue systems that can understand and adapt to the nuances of human communication, marking a significant step toward more natural and efficient human-AI interactions.

# CHAPTER 3

# RELATED WORK

In this chapter, the related studies are covered for multi-hop question answering, and question decomposition.

## 3.1 Multi-hop question answering

Multi-hop question answering is a sub field of open domain question answering where the answer to a question requires multi-hop reasoning over multiple documents.

The HotpotQA dataset [77] is a multi-hop question answering dataset which aims to enhance question-answering (QA) systems by enabling them to perform complex reasoning over multiple documents and provide explanations for their decisions. It consists of a collection of 113k Wikipedia-based question-answer pairs which also include factoid comparison questions. It leverages the entire English Wikipedia corpus by constructing a directed hyperlink graph from the first paragraphs of all articles, recognizing hyperlinks as natural indicators of relationships between disambiguated entities that facilitate multi-hop reasoning. This process involves generating meaningful paragraph pairs for question-answering by identifying 'bridge entities'—key concepts linking different pieces of information, such as the connection between a band and its member in the case of "Radiohead" and "Thom Yorke"—and creating questions that necessitate reasoning over these connected entities, after manually curating suitable pages to ensure the generation of high-quality, multi-hop questions. It has two main benchmark settings: distractor and full-wiki. The distractor setting requires models to discern true supporting facts from a mix of two gold paragraphs and eight distractor paragraphs, which are selected through bigram tf-idf techniques to challenge the model's ability to filter relevant information from noise. The full wiki setting further tests models by demanding they identify and reason with relevant facts across the entire scope of Wikipedia's first paragraphs, without relying on preselected supporting paragraphs. This approach assesses the models' capabilities in both finding accurate information amidst distractions and performing comprehensive multi-document reasoning.

NarrativeQA dataset [78] is a question answering dataset based on fictional stories, including books and movie scripts. It consists of 1,572 stories, including books and movie scripts, with 46,765 human-written question-answer pairs based on human-generated abstractive summaries. The questions cannot be answered using superficial information and require understanding the underlying narrative. The dataset offers tasks of varying complexity, from answering questions based on summaries to reading and comprehending entire stories. This makes it particularly relevant to multi-hop question answering, as it requires models to perform iterative reasoning and handle long-range temporal dependencies.

The MuSiQue dataset [7] was developed in response to the challenge that existing multi-hop QA datasets could be solved without the need for connecting supporting facts, due to issues such as overly specific sub-questions, train-test leakage, and insufficient distractors. This dataset employs a bottom-up construction approach by selecting and combining single-hop questions from existing datasets such as SQuAD [79], Natural Questions [80], MLQA [81], TREx [82] and Zero Shot RE [83], ensuring that each multi-hop question requires the connection of supporting facts for accurate answers. MuSiQue introduces rigorous filters to ensure that no sub-question can be answered without addressing the preceding connected sub-questions, aiming to minimize the effectiveness of disconnected reasoning and reduce train-test leakage. It has two settings: Ans and Full. The dataset, MuSiQue-Ans, consists of approximately 25K questions that span 2-4 hops in complexity and are designed to be significantly more challenging than previous datasets, as evidenced by a threefold increase in the gap between human and machine performance and a significant reduction in disconnected reasoning scores. Additionally, a more stringent variant, MuSiQue-Full, introduces approximately 50K multihop questions, including pairs of answerable and unanswerable questions, to further challenge and evaluate the depth of multi-hop reasoning required by QA models.

Mainstream multi-hop question answering datasets contain long supporting facts for each question. This creates a challenge for transformer-based models due to their limited sequence length. BigBird [84] addresses this issue by introducing a sparse attention mechanism that allows for processing longer sequences. It achieves runner up performance on HotPotQA dataset behind Beam Retrieval [85].

EmbedKGQA [86] aims to improve multi-hop question answering over knowledge graphs by leveraging knowledge graph embeddings to handle sparse and incomplete graphs. The model operates without requiring fine-grained annotations, such as specific question types or detailed reasoning steps, making it more flexible in diverse scenarios. By learning embeddings for both the knowledge graph entities and the questions, EmbedKGQA integrates these representations to accurately predict answers, reducing the reliance on pre-specified answer neighborhoods used in prior approaches.

AISO [87] introduces an adaptive information-seeking approach for ODQA, which conceptualizes ODQA task as a Partially Observable Markov Decision Process (POMDP). The approach employs a variety of retrieval functions, including sparse and dense retrieval mechanisms, to adaptively select and reformulate queries based on the type of question, demonstrating its effectiveness by achieving high performance on HotPotQA dataset.

The Chain-of-Skills (COS) method [88] introduces a modular approach for ODQA, utilizing a transformer-based architecture to master five retrieval reasoning skills: single retrieval, expanded query retrieval, entity span proposal, entity linking, and reranking. By combining these skills in flexible configurations and employing a multi-task learning framework, COS adapts effectively to various ODQA datasets. This adaptability, coupled with a novel parameterization strategy to minimize task interference, allows COS to achieve superior zero-shot retrieval capabilities and high performance when fine-tuned on multiple question answering datasets [88].

HopRetriever [89], a technique designed to improve multi-hop question answering by leveraging both structured and unstructured knowledge within Wikipedia. By encoding entity mentions and document text using mention embeddings and BERT, respectively, HopRetriever efficiently collects evidence necessary for answering complex questions, significantly outperforming previous methods on the HotpotQA dataset.

Beam Retrieval [85] employs a beam search strategy in multi-hop question-answering by tracking multiple potential sets of relevant passages through each hop, starting with the highest scoring passages from a candidate set and iteratively refining the search based on new scores from a classification model. This method broadens the search space and increases the chances of identifying truly relevant passages while reducing early-stage retrieval errors. It achieves state-of-the-art performance on the HotPotQA and MuSiQue-Ans dataset, outperforming existing methods by a significant margin.

Kızıldağ [90] proposes a semi-automatic prompting method for multi-hop question answering that improves baseline performance. A few-shot prompt is constructed in two steps. First, the answers for sub-questions in training set are clustered by named entities. Then, the examples from training set for each named entity group are selected to ensure diversity across the examples in the few-shot prompt.

### 3.2 Question decomposition

Min et al. [91] introduce a method for multi-hop reading comprehension that breaks down complex questions into simpler sub-questions answerable by single-hop reading comprehension models. This approach, utilizing a span prediction technique for sub-question generation from a minimal set of labeled examples, along with a global rescoring strategy for final answer selection, demonstrates improved performance on the HotPotQA dataset. The system provides transparent reasoning by showcasing the generated sub-questions and their answers as part of the decision-making process.

As described in [92], Cao and Liu [93] address the challenge of multi-hop question answering by introducing two strategies: Coarse-Grained Decomposition (CGDe) and Fine-Grained Interaction (FGIn). The CGDe strategy simplifies complex questions into simpler ones, enhancing the model's focus on relevant intermediate answers through improved vector distribution. Meanwhile, the FGIn strategy refines the interaction between document words and questions by utilizing a modified bidirectional attention mechanism, which aids in the accurate extraction of necessary information by focusing on sentences critical for the inference path.

Khot et al. [94] present Decomposed Prompting, an approach that improves the use of LLMs for complex tasks by decomposing them into simpler sub-tasks. This method enables a modular and flexible framework that allows for optimizing each prompt individually, further decomposition of sub-tasks, or replacement with more effective prompts or symbolic functions. The approach demonstrates enhanced performance over previous few-shot prompting methods across multiple tasks, including symbolic reasoning and multi-hop question answering, by integrating a shared library of sub-task-specific prompts and the possibility of incorporating external systems for tasks like information retrieval.

# CHAPTER 4

# METHOD

This thesis proposes a multi-hop question answering system utilizing both knowledge graphs and large language models. Multi-hop reasoning is essential for answering complex queries that span multiple entities and relationships. It can be formulated as a path finding process on a knowledge graph, aiming to identify a sequence of triples $(e_i, \text{rel}_i, e_{i+1})$ that connect a starting entity $e_1$ to a target entity $e_{n+1}$ through a series of intermediate steps. We hypothesize that by structuring data as entity-relation-entity triples, a knowledge graph facilitates effective multi-hop reasoning and enhances the retrieval performance by isolating irrelevant sub-graphs.

The pseudo-codes for two variants of a multi-hop question answering algorithm, with and without question decomposition, are provided in Algorithms 1 and 2. The first variant takes a straightforward approach by retrieving passages relevant to the query, forming a contextual basis, and directly answering the question, with the final step being the return of the derived answer. The second variant enhances the process by incorporating question decomposition. It begins by splitting the main question into two sub-questions. Relevant passages are retrieved for the first sub-question to form a context and derive an answer. If this answer definitively indicates no further inquiry is needed, the process terminates. Otherwise, it is used to refine the second sub-question, repeating the retrieval and answering phases for this modified query, culminating in the answer to the second sub-question.

**Algorithm 1** Multi-hop question answering without question decomposition

**Require:** $passages, question$
**Ensure:** $answer$
Retrieve relevant passages for the question
Form the context from retrieved passages
Answer the question using the context
**return** The answer

## 4.1 Scope

The study evaluates the proposed method using the validation set of the MuSiQue-Ans dataset. The study is limited to two-hop question-answering problems. For the knowledge graph construction task, we explore various techniques for LLMs such as zero-shot prompting, few-shot prompting, supervised fine-tuning with an entity-relation dataset such as WebNLG [95], and reinforcement learning based fine-tuning with a small language model such as *Meta-Llama-3-8B-Instruct* [96]. For question

**Algorithm 2** Multi-hop question answering with question decomposition

**Require:** $passages, question$
**Ensure:** $answer$
  **# Question Decomposition**
  Decompose the question into two sub-questions
  **# The First Hop**
  Retrieve relevant passages for the first sub-question
  Form the context from retrieved passages
  Answer the first sub-question using the context
  Store the answer for first sub-question
  **if** the first answer is "No answer" **then**
    **return** "No answer"
  **end if**
  **# The Second Hop**
  Modify the second sub-question using the answer from the first hop
  Retrieve relevant passages for the modified second sub-question
  Form the context from retrieved passages
  Answer the modified second sub-question using the context
  **return** The answer for the second sub-question

answering task, we use a medium sized language model such as *GPT-3.5 Turbo*. It is known that the reasoning capabilities of language models advances as the model size grows. Hence, we aim to improve reasoning capabilities of medium sized language models.

### 4.2 Knowledge Graph Construction

For each sample in the dataset, a knowledge graph is constructed using a large language model by performing joint entity relation extraction (JERX) over all paragraphs of the sample. As the paragraphs do not co-reference each other, the knowledge graph can be constructed by treating each paragraph as a separate document. Hence, we extract entity relation triplets from each paragraph and build a knowledge graph from the whole set of triplets. Example visualizations for a knowledge graph constructed for a sample are provided in Fig. 14 and 15 in Appendix A.1.

Four techniques are explored for JERX with selected LLMs:

- Zero-shot prompting
- Few-shot prompting
- Supervised fine-tuning
- Reinforcement learning based fine-tuning

#### 4.2.1 Zero-shot Prompting

In zero-shot prompting setting, the language model is instructed to extract entities and relations from the paragraph without any examples provided. The system prompt is constructed using expert prompting [97] [98] and role/persona prompting [99] [100] [101] techniques, where the LLM is prompted to adopt a persona with expertise in a given domain. This has been shown to enhance its performance in that domain compared to baseline prompts [97] [102] [103] [102]. Accordingly, we used the prompt given in Fig. 1.

```
System:
You are an excellent knowledge graph construction
agent.  Extract knowledge triplets in the form of
(subject,relation,object) from user's messages.  Avoid stop
words.  Use '|' as delimiter and provide one triplet per line.
```

Figure 1: Zero-shot prompt for JERX

#### 4.2.2 Few-shot Prompting

In few-shot prompting setting, the selected LLM is prompted with two examples to guide the model for extracting entities and relations from a given input. To this end, we utilized two examples taken from MuSiQue-Ans [7] training set and manually annotated the triplets for the paragraphs in these examples. The selected model is instructed to extract entities and relations from the paragraph using the prompt given in Fig. 2.

**System:**
You are an excellent knowledge graph construction agent. Extract knowledge triplets in the form of (subject,relation,object) from user's messages. Avoid stop words. Use '|' as delimiter and provide one triplet per line.

**User:**
Glenhis Hernández (born 7 October 1990 in Havana) is a taekwondo practitioner from Cuba. She was the 2013 World Champion in middleweight.
The current mayor of Havana ("President of the People's Power Provincial Assembly") is Marta Hernández Romero, she was elected on March 5, 2011.

**Assistant:**
Glenhis Hernández (Athlete) | born on | October 7, 1990
Glenhis Hernández (Athlete) | birth place | Havana
Glenhis Hernández (Athlete) | specializes in | taekwondo
Glenhis Hernández (Athlete) | won | 2013 World Champion title (Middleweight)
Marta Hernández Romero (Politician) | serves as mayor | Havana
Marta Hernández Romero (Politician) | holds | the position of "President of the People's Power Provincial Assembly"
Marta Hernández Romero (Politician) | elected on | March 5, 2011.

**User:**
The United States Air Force, which fought in the Korean War, operates the Al Asad airbase. Among the USAF attack aircraft are the Lockheed C-130, which serves as a transport plane, the General Dynamics F-16 Fighting Falcon and the attack aircraft Lockheed AC-130. The orbital Period of 1001 Gaussia is 5.75 days and the epoch date is 27 June 2015. The Abarth company makes the Abarth 1000 GT Coupe and was founded in Bologna.

**Assistant:**
Al Asad Airbase | operating organisation | United States Air Force
United States Air Force | attack aircraft | Lockheed AC-130
United States Air Force | battle | Korean War
United States Air Force | aircraft fighter | General Dynamics F-16 Fighting Falcon
United States Air Force | transport aircraft | Lockheed C-130 Hercules
1001 Gaussia | epoch | 2015-06-27
1001 Gaussia | orbital period | 5.75 (days)
Abarth | foundation place | Bologna
Abarth 1000 GT Coupé | manufacturer | Abarth



Figure 2: Few-shot prompt for JERX

#### 4.2.3 Supervised Fine-tuning

The language model is fine-tuned on an entity-relation dataset, WebNLG[95], to enhance its joint entity-relation extraction capabilities. The dataset is pre-processed such that each sample uses the same system prompt in zero-shot and few-shot prompting settings and a few examples from the training set to form a conversation sample. Using examples from the training set of WebNLG dataset prevents data leak as the JERX performance of the fine-tune model is evaluated with the validation set of WebNLG dataset. We use few-shot prompting technique while performing supervised fine-tuning (SFT) to minimize the gap between the base model and the fine-tuned model. This improves the training stability and task performance of the model. Each sample uses a different set of examples so that the model generalizes over few-shot examples. An example data point is provided in Figure 3.

```
System:
You are an excellent knowledge graph construction
agent.  Extract knowledge triplets in the form of
(subject,relation,object) from user's messages.  Avoid stop
words.  Use '|' as delimiter and provide one triplet per line.
User:
Sausage is a main ingredient of Bacon Explosion but it also
uses bacon.  The icebreaker Aleksey Chirikov is 99.83 metres
long.  The country of Akron, Ohio, is United States, where Asian
Americans are one of the ethnic groups.
Assistant:
Akron, Ohio | country | United States
Bacon Explosion | main ingredient | Sausage
United States | ethnic group | Asian Americans
Aleksey Chirikov (icebreaker) | length | 99.83
Bacon Explosion | ingredient | Bacon
User:
Alvinegro is the nickname of Agremiação Sportiva Arapiraquense.
Assistant:
Agremiação Sportiva Arapiraquense | nickname | Alvinegro
```

Figure 3: An example data point to SFT for JERX

#### 4.2.4 Reinforcement Learning based Fine-tuning

We also explored fine-tuning the language model by reinforcement learning with task feedback, where task is question answering. This approach aims to improve the qualities and completeness of the entity relation triplets extracted by the model. The reward signal consists of two components:

- **Question Answering Accuracy:** Another language model answers the question based on the extracted triplets, and the correctness of the answer forms the basis of the reward.

- **Heuristics-Based Reward:** A heuristic reward function considers several factors such as the length of the response, the correctness of triplet formatting, the number of unique entities and relations, and the total number of triplets. This component aims to guide the language model towards extracting a balanced and diverse set of triplets.

The final reward signal is calculated by combining these components through a weighted sum, as described in Equation 1. In this equation, $\text{reward}_{\text{QA}}$ represents the reward linked to question answering accuracy, while $\text{reward}_{\text{heuristics}}$ corresponds to the reward associated with the heuristic assessment of the extracted triplets. The weights $w_{\text{QA}}$ and $w_{\text{heuristics}}$ are tuning parameters that control the influence of each reward component on the final reward signal, allowing the model to balance the trade-off between question answering accuracy and heuristic evaluation.

$$\text{reward} = w_{\text{QA}} \times \text{reward}_{\text{QA}} + w_{\text{heuristics}} \times \text{reward}_{\text{heuristics}} \tag{1}$$

### 4.3 Question Decomposition

The question decomposition module decomposes a two-hop question into two sub-questions by utilizing an LLM with the few-shot prompt provided in Fig. 4. The system prompt is carefully crafted to guide the language model in generating sub-questions suitable for a multi-hop question answering dataset. It specifically discourages open-ended questions such as "Who is..." or "How is...", which often yield broad or ambiguous answers that do not contribute effectively to the sequential information retrieval required. By instructing the model to create sub-questions that reference previous answers, continuity and relevance in the information gathering process are enhanced. This design, refined through iterative adjustments, ensures that the sub-questions are precise and contextually appropriate, facilitating accurate decomposition and answer synthesis. Additionally, the prompt specifies avoiding conversational statements to focus the language model solely on the task of question decomposition, making the output easier to parse and align with the structured format specified. The prompt also utilizes two examples taken from MuSiQue-Ans [7] training set.

### 4.4 Information Retrieval

Retrieving correct information from a large corpus is an important step for open domain question answering systems. In any sample of MuSiQue-Ans dataset, only one of the paragraphs among up to 20 paragraphs is relevant for each sub-question. Our study explores both sparse and dense retrieval algorithms to select relevant paragraphs for the question in a sample.

#### 4.4.1 Sparse retriever

The sparse retriever employs BM25 algorithm [10], which is a variation of the traditional TF-IDF (Term Frequency-Inverse Document Frequency) approach. BM25 algorithm enhances the retrieval process by accounting for both the frequency of the query terms in the documents and the inverse

**System:**
Decompose the given question into 2 sub-questions such that when sub-questions are answered, the original question can be answered correctly. The second sub-question must refer to the answer of the first question by `#1` as in the examples below. Do not create open-ended sub-questions like "Who is ..." or "How is ...".
Avoid statements like 'Here are the decomposed sub-question, ...' or 'Sure! Here are..' or anything along those lines. Only provide sub-questions as illustrated in the examples below.

**Question:** Who founded the journal that published The Review of Communication?

1. Who published The Review of Communication?
2. Who founded #1?

**Question:** When was the institute that owned The Collegian founded?

1. Which institute does own The Collegian?
2. When was #1 founded?

**Question:** *{question}*

Figure 4: Few-shot prompt for question decomposition

document frequency of the query terms across the dataset. It calculates scores for a document's relevance to a given search query based on the term's presence within the document, adjusted by the term's commonality across all documents. The higher the term frequency in the document relative to its frequency in the corpus, the higher the score, thereby promoting documents that are uniquely relevant to the query. Therefore, the paragraphs with the top-k highest scores are selected as the most relevant paragraphs to the query.

#### 4.4.2 Dense retriever

The dense retrieval algorithm employs the *semantic-transformers/all-MiniLM-L6-v2* model from the sentence-transformers Python library [104] to generate embeddings for queries and paragraphs. This model leverages a transformer-based architecture to encode text into dense vectors that capture the semantic nuances of the language. By comparing the embedding vector of a query with those of each paragraph in the dataset, the algorithm measures semantic similarities using distance metrics such as cosine similarity. It then selects paragraphs with the top-k lowest distances, indicating the highest relevance to the query. This method emphasizes the understanding of underlying semantic

relationships rather than merely matching keywords, providing a sophisticated approach to information retrieval. This implementation is based on the concepts introduced in [104], which pioneered the use of BERT-based models for efficient sentence embedding.

### 4.5 Knowledge Graph based Question Answering

In our study, the system for knowledge graph augmented question answering with LLM is structured into two stages: knowledge graph construction and question answering based on triplets retrieved from the knowledge graph, as illustrated in Fig. 5. During the KGC stage, entity-relation-entity triplets are extracted from each paragraph and stored in the knowledge graph using the Llama-3-8B model. In the QA stage, relevant triplets are retrieved from the knowledge graph using a retrieval algorithm (sparse or dense) based on the question. These triplets are then concatenated to form a context for GPT-3.5 Turbo model, which generates an answer based solely on the retrieved triplets.

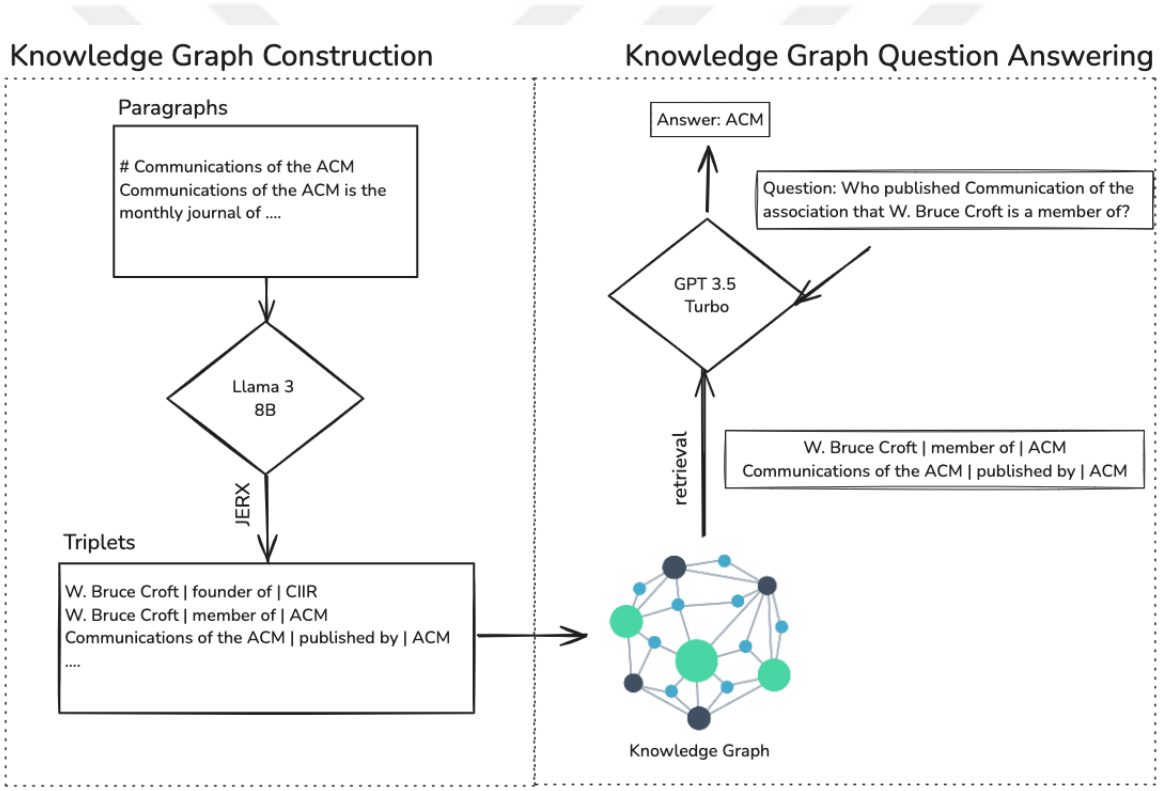


Figure 5: Architecture of knowledge graph-based question answering system

### 4.6 Prompting Techniques for Question Answering

Many prompting techniques have been developed for LLMs. In its simplest form, a prompt contains instructions for a task. For question answering based on a context, i.e. reading comprehension task, a baseline prompt is designed as shown in Fig. 6 using expert prompting [97] [98] and role/persona

prompting [99] [100] [101] techniques. This has been shown to enhance its performance in that domain compared to baseline prompts [97] [102] [103] [102]. It is templated by *{context}* and *{question}*. Wei et al. [28] demonstrated that Chain-of-Thought (CoT) prompting improves the complex reasoning capabilities of LLMs. We explore zero-shot and few-shot (with one example) variants of CoT prompting technique, whose prompts are provided in Fig. 7, and Fig. 8, respectively.

```
System:
You are an excellent question-answering system known for
providing accurate and reliable answers.  Your responses should
be solely based on the context information given, without
drawing on prior knowledge.
User:
The context information is provided below.
--------------
{context}
--------------
Given the context information and not prior knowledge, answer
the question.
{question}
```

Figure 6: Standard prompt for question answering

```
System:
You are an excellent question-answering system known for
providing accurate and reliable answers.  Your responses should
be solely based on the context information given, without
drawing on prior knowledge.  Always provide clear and logical
step-by-step reasoning in your answers.

User:
The context information is provided below.
--------------
{context}
--------------
Given the context information and not prior knowledge, answer
the question.
{question}
```

Figure 7: Zero-shot Chain-of-Thought prompt for question answering

```
System:
You are an excellent question-answering system known for
providing accurate and reliable answers.  Your responses should
be solely based on the context information given, without
drawing on prior knowledge.  Always provide clear and logical
step-by-step reasoning in your answers.

# Example

Context:
Glenhis Hernández (born 7 October 1990 in Havana) is a taekwondo
practitioner from Cuba.  She was the 2013 World Champion in
middleweight.
The current mayor of Havana ("President of the People's Power
Provincial Assembly") is Marta Hernández Romero, she was elected
on March 5, 2011.

Question:  Who is the current mayor of the city Glenhis
Hernández was born?

Reasoning:
1.  Glenhis Hernández was born in Havana, as mentioned in the
context.
2.  The current mayor of Havana mentioned in the context is
Marta Hernández Romero.
3.  Therefore, the current mayor of the city where Glenhis
Hernández was born is Marta Hernández Romero.

Answer:  Marta Hernández Romero

User:
The context information is provided below.
--------------
{context}
--------------
Given the context information and not prior knowledge, answer
the question.
{question}
```

Figure 8: Few-shot Chain-of-Thought prompt for question answering

We propose a prompting technique, named Connect-the-Entities (CTE), that instructs the LLM to list entity-relation-entity triplets from the given context before answering the question. We hypothesize that forcing LLMs to extract these triplets increases their multi-hop reasoning performance. The few-shot (with one example) prompt template for CTE prompting is provided in Fig. 9.

```
System:
You are an excellent question-answering system known for
providing accurate and reliable answers.  Your responses should
be solely based on the context information given, without
drawing on prior knowledge.
Before answering the question, first, you extract relevant
entity-relation-entity triplets from the context.  Then, you
answer the question based on the triplets.

# Example
Context:
Glenhis Hernández (born 7 October 1990 in Havana) is a taekwondo
practitioner from Cuba.  She was the 2013 World Champion in
middleweight.
The current mayor of Havana ("President of the People's Power
Provincial Assembly") is Marta Hernández Romero, she was elected
on March 5, 2011.

Question:  Who is the current mayor of the city Glenhis
Hernández was born?

Triplets:
Glenhis Hernández | birth place | Havana
Marta Hernández Romero | serves as | mayor of Havana

Answer:  Marta Hernández Romero

User:
The context information is provided below.
--------------
{context}
--------------
Given the context information and not prior knowledge, answer
the question.
{question}
```

Figure 9: Few-shot Connect-the-Entities question-answering prompt

### 4.7 Evaluation Metrics

This section covers the evaluation metrics applied to assess the effectiveness of each component in the proposed methods, tailored to the specific tasks. For information retrieval task, we discuss the use of precision, recall, and F1 score. In JERX task, the focus is on precision, recall, F1 score, and string

equivalence. Finally, for question answering task, we examine the Exact Match (EM) and F1 score metrics.

### 4.7.1 Information Retrieval

The performance of information retrieval systems is assessed using various metrics such as precision, recall, and F1 score. Among these, recall, defined on a per-sample basis, measures the proportion of relevant documents retrieved relative to the total number of relevant documents within that sample. It is calculated using the formula:

$$\text{Recall} = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Relevant Documents in the Sample}}$$

This metric evaluates the retrieval algorithm's ability to retrieve all relevant documents from a corpus. A recall score of 1 indicates that all relevant documents have been retrieved for a given sample, while a score of 0 means that no relevant documents were retrieved.

Precision is another metric for assessing the performance of information retrieval systems. It measures the proportion of retrieved documents that are relevant, providing insight into the accuracy of the retrieval process. Unlike recall, which evaluates completeness, precision assesses the quality of the retrieval by indicating how many of the retrieved documents are actually relevant. The formal definition of precision is given by the formula:

$$\text{Precision} = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Documents Retrieved}}$$

This metric is particularly important when the cost of retrieving irrelevant documents is high, as it ensures that the results returned by the retrieval process are indeed pertinent to the query. Precision is especially significant in scenarios where the response set needs to be both concise and relevant, such as in search engines where user satisfaction relies heavily on retrieving few, but highly relevant, results.

When the top-k, i.e. total number of documents retrieved, is predefined in an information retrieval setting, the metric of precision may not provide the most insightful evaluation. Precision measures the proportion of retrieved documents that are relevant within the fixed set of k results, irrespective of the total number of relevant documents that exist within the entire dataset. This fixed limitation means that precision exclusively assesses the accuracy of the retrieved subset, without considering the presence of additional relevant documents that may lie outside of the top-k selection. In contexts where it is essential to understand the extent to which all relevant documents are covered within the predefined limits of k results, precision does not offer adequate insight. Instead, recall, which quantifies the proportion of all relevant documents that are included within the top-k results, becomes a more relevant metric, as it directly addresses the completeness of the retrieval process. Hence, recall metric is chosen to evaluate the retrieval performance in our study.

### 4.7.2 Joint Entity Relation Extraction

The performance of the JERX model is commonly assessed using three standard metrics: precision, recall, and the F1 score [105]. These metrics are calculated based on the comparison of extracted triplets (entity-relation-entity) predicted by the model against a manually annotated set of reference triplets. The order of triplets does not affect the evaluation, as the focus is on the accuracy and completeness of the extracted data. Each metric is calculated for each sample and then their average over the dataset is reported.

**String Equivalence:** The evaluation employs a fuzzy matching criterion to assess the equivalence of strings within triplets. This criterion is implemented using the *SequenceMatcher* utility from the Python standard library, which is based on the Gestalt pattern matching algorithm [106]. This method introduces flexibility by tolerating minor discrepancies in spelling. By comparing the similarity between strings and ensuring it surpasses a set threshold, which can be adjusted between 0 and 1, this approach ensures that trivial differences do not significantly affect the assessment results. The setting of the threshold plays a critical role in balancing sensitivity against specificity. A lower threshold may lead to a higher occurrence of false positives, erroneously identifying dissimilar strings as equivalent. On the other hand, a higher threshold minimizes false positives by allowing less discrepancy, thereby requiring a closer match between strings. At the extreme, when the threshold reaches 1, the criterion necessitates an exact match, thereby only identifying strings as equivalent if they are identical, character-for-character. This level of precision is crucial in scenarios demanding high accuracy, though it restricts flexibility in handling minor spelling variations. In our study, the threshold is set at 0.8, optimizing the balance between flexibility and accuracy in string matching. This level permits minor spelling variations, accommodating human error and inconsistencies in data entry, while still maintaining a stringent criterion that minimally admits false positives. Such a setting ensures that only strings with a high degree of similarity are recognized as equivalent, safeguarding the precision of the matching process. For illustrative purposes, examples provided in Table 1 demonstrate the application of this threshold, showing that pairs such as "Barack Obama" and "Barak Obama" or "Center for Disease Control" and "Centre for Disease Control" are considered equivalent under the fuzzy match criterion but not under exact match. Additionally, the table highlights that significantly different strings such as "Atlantic Ocean" and "Pacific Ocean" or "1989-11-09" and "1989-09-11" are correctly identified as non-equivalent, demonstrating the effectiveness of the threshold in preventing false positives.

Table 1: Examples for the comparison of string equivalence criteria

| **String A** | **String B** | **Exact Match** | **Fuzzy Match** |
|---|---|---|---|
| Apple Inc. | Apple Inc. | True | True |
| Microsoft Corporation | Microsoft Corporation | True | True |
| Google LLC | Google LLC | True | True |
| Barack Obama | Barak Obama | False | True |
| London, United Kingdom | London / United Kingdom | False | True |
| Center for Disease Control | Centre for Disease Control | False | True |
| University of Cambridge | University of Oxford | False | False |
| Atlantic Ocean | Pacific Ocean | False | False |
| 1989-11-09 | 1989-09-11 | False | False |

**Precision:** Precision is the proportion of correct triplets among all predicted triplets, where a triplet is considered correct if it matches a reference triplet based on the fuzzy matching criterion:

$$\text{Precision} = \frac{|\{\text{Correct Predicted Triplets}\}|}{|\{\text{Predicted Triplets}\}|} \tag{2}$$

**Recall:** Recall measures the system's ability to capture all relevant triplets in the dataset, defined as the ratio of correctly predicted triplets to the total number of reference triplets:

$$\text{Recall} = \frac{|\{\text{Correct Predicted Triplets}\}|}{|\{\text{Reference Triplets}\}|} \tag{3}$$

**F1 Score:** The F1 score is the harmonic mean of Precision and Recall, providing a balanced measure of the model's performance:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

These metrics, under the fuzzy matching criterion, effectively account for minor inaccuracies in the extraction process, thereby providing a robust evaluation of the model's capability to capture meaningful relationships within the text.

#### 4.7.3 Question Answering

The question answering performance is assessed using exact match (EM) and F1 score metrics.

##### 4.7.3.1 Exact Match (EM)

The Exact Match (EM) metric assesses the accuracy of predictions in natural language processing by verifying if the predicted answer exactly matches any of the reference answers after both have been normalized. The normalization process involves converting all characters to lowercase, removing punctuation, eliminating specific articles ("a", "an", "the"), and stripping extra whitespace. The metric is binary, assigning a score of 1 for a perfect match and 0 otherwise. EM is essential in tasks requiring precise answers, such as factual question answering.

##### 4.7.3.2 F1 Score

The F1 score is a statistical measure used to test the accuracy of a model. It is the harmonic mean of precision and recall, providing a balance between these metrics by considering both the tokens present in the prediction and those in the reference. The score is calculated by first tokenizing the normalized texts into minimal units, typically words. The overlap of tokens between the prediction and the reference is used to compute precision (the proportion of predicted tokens that are correct) and recall (the proportion of reference tokens that are correctly predicted). The F1 score is defined as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

This metric is valuable in scenarios where answers can be correctly expressed in multiple forms, allowing for partial credit for non-exact matches.

#### 4.7.3.3 Examples

Below is a table showing examples of how the EM and F1 scores are calculated for different predicted and reference answers:

Table 2: Examples of Exact Match and F1 Score calculations for various predictions

| Reference Answers | Predicted Answer | Exact Match (EM) | F1 Score |
|---|---|---|---|
| University of Toronto; UT | University of Toronto | 1 | 1.00 |
| Chiang Hsiao-wu; Alex Chiang | Chiang Hsiao-wu | 1 | 1.00 |
| George Benson | George Benson | 1 | 1.00 |
| La Goulette | La Goulette, Tunisia | 0 | 0.80 |
| Dovber Schneuri | Rabbi Dovber Schneuri | 0 | 0.80 |
| Stanley | Stanley, North Dakota | 0 | 0.50 |
| Ahmad Shah Qajar | Mahmoud Mirza | 0 | 0.00 |
| in Northern Florida; Northern Florida | 45th most populated city | 0 | 0.00 |
| Het Scheur | North Sea | 0 | 0.00 |

Table 2 provides an illustrative comparison of different predictions against a reference, demonstrating the binary nature of the EM metric and the graded evaluation by the F1 score.

# CHAPTER 5

# EXPERIMENTS

This chapter covers the experiments to test the proposed method's performance, including sub-sections on Dataset, Experimental Setup, Results, and Discussion.

## 5.1 Datasets

This section covers the datasets used to evaluate the proposed methods, each selected for its relevance to specific tasks. We first describe the MuSiQue dataset, a multihop reading comprehension dataset designed to challenge models with complex, multi-step questions. The focus is on the MuSiQue-Ans variant, which is used for experiments in this study. Next, we detail the WebNLG dataset, originally designed for natural language generation tasks, but utilized here for its structured RDF triplets to enhance and evaluate entity-relation extraction models.

### 5.1.1 MuSiQue

MuSiQue [7] is a multihop reading comprehension dataset with 2-4 hop questions, specifically designed to less cheatable and more challenging than prior similar datasets. The dataset is constructed by composing multiple connected single-hop questions from five existing datasets. The dataset has two variants: MuSiQue-Ans, which contains only answerable questions, and MuSiQue-Full, which contains both answerable and unanswerable questions. In this study, MuSiQue-Ans variant is used for experiments. Each data point in MuSiQue-Ans dataset consists of a question, a context of paragraphs, decomposition of the question into sub-questions, and the answers to the question. A sample may have up to 20 paragraphs including supporting and non-supporting paragraphs. An example data point from the training set is provided in Fig. 10, where only supporting paragraphs are included for brevity.

The dataset is split into train, validation, and test sets. No question or sub-question is shared among the sets to prevent data leakage [7]. This thesis study only considers the 2-hop questions in the MuSiQue-Ans dataset, totaling 16,899 instances with 14,376 in the training set, 1,252 in the validation set, and 1,271 in the test set. 100 samples are randomly selected from validation set and used in all experiments for evaluations in this study. We name this subset as "validation-mini" and provide the IDs of the samples in Appendix B.1.

**Paragraphs:**
...
2. KNFM(92.3 FM), branded as 'Lonestar 92', is a Country music...
...
12. Midland is a city in and the county seat of Midland County...
...

**Question:** What county is the town where KNFM is licensed the capital of?

**Question Decomposition:**
1. "What town is KNFM licensed in? **Answer:** Midland
2. "#1 » capital of" **Answer:** Midland County

**Answer Aliases:** "Midland County, Texas"

Figure 10: An example data point from MuSiQue-Ans dataset

#### 5.1.2 WebNLG Dataset

The WebNLG dataset [95] is structured specifically for applications in natural language processing, originally created for challenges in natural language generation (NLG). It includes RDF (Resource Description Framework) triplets that detail relationships between entities and their corresponding natural language descriptions, aiming to facilitate both the generation of text from structured data and the extraction of structured data from text. It spans various DBpedia [107] categories, enabling extensive tests on systems' ability to both produce and interpret text accurately and contextually.

The dataset includes three main splits: the training set contains 13,211 samples, the development (dev) set comprises 1,667 samples for iterative tuning and validation, and the test set features 5,713 samples for evaluating model performance. Each set within the dataset can contain one to seven RDF triplets. An example entry is provided in Fig. 11.

**Category:** Airport
**Modified Triple Sets:**

Aarhus_Airport | cityServed | "Aarhus, Denmark"
**Lexicalizations:**

The Aarhus is the airport of Aarhus, Denmark.

Aarhus Airport serves the city of Aarhus, Denmark.

Figure 11: An example data point from WebNLG

For the purposes of this study, the focus is primarily on the reverse task of extracting RDF triples from textual descriptions, a critical capability for enhancing entity-relation extraction algorithms. The

dataset's division into "seen" and "unseen" categories in the test set allows for rigorous testing of a model's ability to generalize from trained categories to novel ones.

In this study, the WebNLG dataset is utilized to fine-tune a language model for the task of JERX, exploiting its dual utility to improve both the extraction of structured data from natural language texts and the assessment of model robustness across varied linguistic contexts. The triplets are pre-processed to convert entity and relations into their natural language forms, e.g. "Aarhus_Airport" is converted to "Aarhus Airport", and "cityServed" is converted to "city served", so that they are more familiar for LLMs.

### 5.2 Experimental Setup

The experiments were conducted during June and July 2024, utilizing *GPT-3.5 Turbo* model via the OpenAI API as question answering model and *Meta-Llama-3-8B-Instruct* [96] model, which is referenced as *Llama-3-8B* in subsequent sections, as JERX model.

The experiments for supervised fine-tuning *Llama-3-8B* were performed on a single Nvidia Tesla V100 GPU, equipped with 16 GB of VRAM, hosted on TRUBA (Turkish National Science e-Infrastructure). To accommodate the model parameters, its gradients, and mini-batch within the constraints of the GPU's memory, the model was quantized to 4 bits.

RL-based fine-tuning was conducted on the Nvidia Tesla A100, which is equipped with 40 GB of VRAM. This larger memory capacity is crucial, as the PPO algorithm requires maintaining two copies of the LLM simultaneously—one continuously updated during training and the other kept static. The KL divergence between these models is calculated to form part of the loss function, introducing a per-token KL penalty to the rewards to discourage significant deviations from the original policy. To manage these computational demands, both models were also quantized to 4 bits, similar to the supervised fine-tuning setup.

For LLM inference, the *text-generation-inference* library from HuggingFace was employed, enabling streamlined text generation capabilities compatible with OpenAI API [108].

#### 5.2.1 Joint Entity Relation Extraction

**Supervised Fine-Tuning (SFT)** The *Llama-3-8B* model was supervised fine-tuned with the WebNLG dataset. To prepare the model for varied input lengths and to simulate the longer paragraphs of the target MuSiQue-Ans dataset, samples from the WebNLG dataset were randomly concatenated. This method introduces variance in input sequence lengths, enhancing model generalization, and aligns the sequence lengths with the conditions of the target dataset.

The data was reformatted into an instruction/conversation format, suitable for the Llama-3-8B model. The fine-tuning process employed a parameter-efficient technique known as LORA [42] with 4-bit quantization [43] to accommodate the model training within the constraints of a single GPU. The model is fine-tuned on only completion tokens, i.e. the ground-truth triplets, as it is aimed to make model learn joint entity-relation extraction rather than memorize the statements in the dataset.

For prompting, we employed few-shot technique as described in Section 4.2.3. The zero-shot prompting involved a single user and assistant message. Whereas few-shot prompting includes 2-4 messages from the training set of WebNLG dataset to avoid data leakage. The fine-tuned model was evaluated using precision, recall, and F1 score metrics on the validation set of the WebNLG dataset.

**Reinforcement Learning from Task Feedback (RLTF)** Alternatively, *Llama-3-8B* model was fine-tuned using the Proximal Policy Optimization (PPO) [109] algorithm. The reward structure was dual-layered as formulated in Eq. 1: heuristic-based rewards focused on the attributes of generated triplets like length, count, and uniqueness, while task-based rewards were determined by the model's efficacy in a question-answering task. A positive reward was granted if an LLM, such as *GPT-3.5 Turbo*, could accurately answer questions using the generated triplets.

As shown in Fig. 12, the mean reward for the model fine-tuned with RL decreases during the training, indicating that the model diverges. In other words, the model's JERX capabilities do not improve with training. Hence, this model is not used as JERX model in the subsequent question answering experiments.

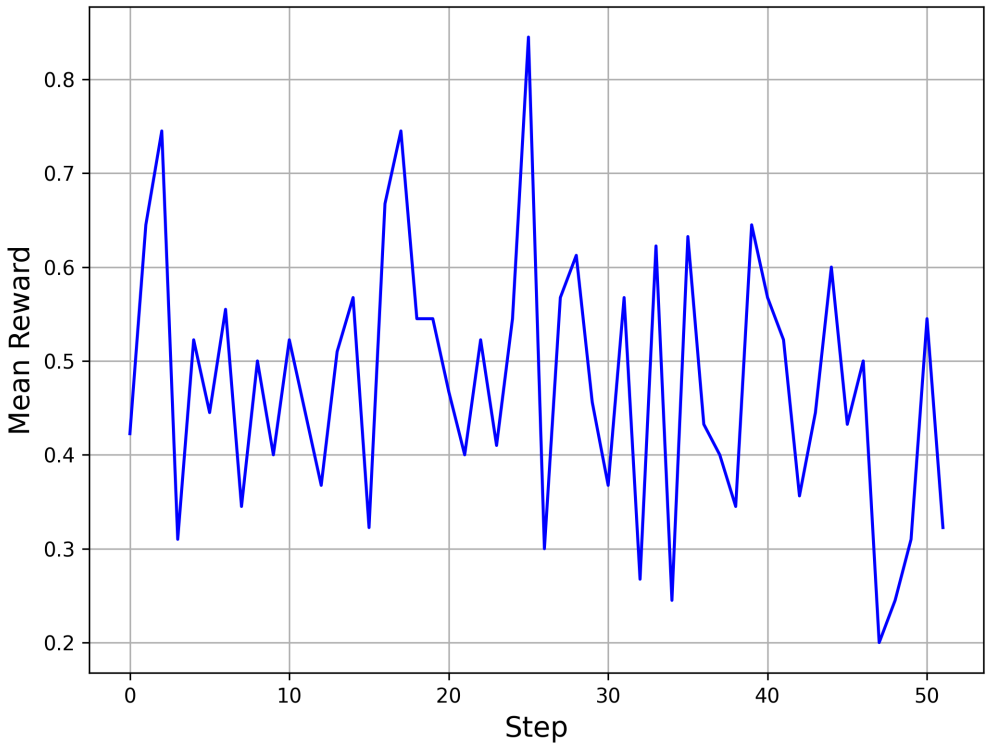


Figure 12: Mean reward during RL training

### 5.2.2 Question Answering

In all applicable experiments, *GPT-3.5 Turbo*, accessed via the OpenAI API, was utilized as question answering model with temperature parameter of 0.1 to ensure variance in generations. Other parameters such as *top_p* and *presence_penalty* are used with their default values, which are 1 and 0, respectively.

### 5.2.3 Compared Models

The performance of our proposed method was compared against the performances of two sets of models: the baseline models and the leaderboard models for the MuSiQue-Ans dataset. The baseline methods, as provided by [7], serve as standard reference points to evaluate new approaches. On the other hand, the leaderboard models represent the current state-of-the-art submissions for this dataset, as recorded on the MuSiQue-Ans leaderboard [110], which measures performance against the test split of the dataset.

The baseline models are End2End (EE) model, Select+Answer (SA) model, and two variants of Step Execution (EX) model, EX(EE) and EX(SA). The End2End (EE) model takes a question and its corresponding context as input, processes them through a transformer model, and directly outputs the answer along with relevant supporting paragraphs [7]. This model employs the Longformer-Large architecture [111], which can handle lengthy contexts, allowing it to perform strong multihop reasoning. The Select+Answer (SA) model [7] operates in two stages: first, it uses RoBERTa-large [112] to rank and select the most relevant paragraphs from the context, and then it predicts the answer based on these selected paragraphs using Longformer-Large [111]. The Step Execution (EX) model decomposes the original multi-hop question into a series of single-hop questions using a directed acyclic graph (DAG) [7]. This model has two variants: EX(EE) and EX(SA). The EX(EE) variant uses the End2End model for single-hop reasoning at each step, while the EX(SA) variant uses the Select+Answer model. Both variants execute the steps sequentially, with each step producing partial answers that contribute to the final result. The decomposition is implemented using BART-large [112], trained on gold decompositions from the dataset. The performances of baseline models on MuSiQue-Ans validation-mini subset are provided in Table 3.

Table 3: Question answering performances of official baselines on MuSiQue-Ans validation-mini subset

| Model | EM | F1 |
|---|---|---|
| Step Execution by Select+Answer (EX(SA)) | 0.620 | 0.663 |
| Step Execution by End2End (EX(EE)) | 0.530 | 0.597 |
| Select+Answer (SA) | 0.430 | 0.515 |
| End2End (EE) | 0.290 | 0.375 |

The performances of state-of-the-art models listed in the leaderboard [110] as of August 2024 are provided in Table 4. It can be noticed that the baseline models attain different scores on test set, as presented in Table 4, than the validation-mini subset, as presented in Table 3. These differences exist because the validation-mini subset is derived from validation set and only contains 2-hop questions.

Table 4: Question answering performances of state-of-the-art models on MuSiQue-Ans test set

| **Model** | **F1** |
|---|---|
| Online Finetuned Flow (OFF) | 0.693 |
| Beam Retrieval (beam size 2) | 0.692 |
| Beam Retrieval (beam size 1) | 0.669 |
| GenDec | 0.654 |
| BS1-large | 0.647 |
| neo-Z | 0.636 |
| Select+Answer (SA) | 0.523 |
| Step Execution by Select+Answer (EX(SA)) | 0.490 |
| Step Execution by End2End (EX(EE)) | 0.464 |
| End2End (EE) | 0.407 |

### 5.3 Results

This section presents the experimental results obtained from the evaluation of various components and configurations of the proposed methods. The first subsection discusses the performance of the Knowledge Graph-based Question Answering (KGQA) system, tested on the MuSiQue-Ans validation-mini subset, focusing on the impact of question decomposition, retrieval methods, and triplets as context for LLM. The second subsection explores the results of applying standard and CTE prompting techniques to paragraph-based question answering, highlighting the effectiveness the advantages of CTE prompting. Finally, the third subsection provides an in-depth analysis of individual components and parameters, including the impact of different prompting techniques, temperature values, LLM adaptation methods, and retrieval algorithms on overall system performance. To account for variability in text generation, each experiment involving an LLM was conducted three times. The results were averaged across these three repetitions for each metric, and the average values are presented in this section. Comprehensive results from all experimental runs are presented in Appendix B.

#### 5.3.1 Knowledge Graph based Question Answering

This section presents the evaluation of the knowledge graph-based question answering (KGQA) system on MuSiQue-Ans validation-mini subset. The system was tested with various configurations, focusing on the impact of question decomposition and retrieval methods. Standard prompting technique given in Fig. 6 was employed for question answering LLM (GPT-3.5 Turbo) and the context was only the triplets retrieved from knowledge graph. Performance metrics, as detailed in Table 5, provide a comprehensive insight into the system's efficiency across different settings.

The analysis revealed that question decomposition significantly enhances the system's performance. Configurations employing question decomposition consistently yielded higher EM and F1 scores compared to those without. For instance, the top-performing setup with question decomposition and dense retrieval at top-70 achieved an F1 score of 0.530. In contrast, the best configuration without question decomposition only managed an F1 score of 0.416 with sparse retrieval method.

A comparative evaluation of retrieval methods indicated dense retrieval performs slightly better than sparse retrieval. However, regardless of the retrieval type, models utilizing question decomposition outperformed those that did not.

The system's performance surpassed that of the baseline models, Select-Answer (SA) and End2End (EE), but lagged behind Step Execution by Select+Answer (EX(SA)) and Step Execution by End2End (EX(EE)). It's important to note that our system uses a pre-constructed knowledge graph as an external knowledge source while answering the questions. Hence, it is extensible to answer alternative questions about the given context, which makes it suitable for practical applications where the questions are not known in advance. Additionally, the system's design incorporates few-shot prompted general-purpose language models for joint entity recognition and question answering (JERX), enhancing its adaptability to different domains without the need for specialized models.

Table 5: Evaluation results for knowledge graph based question answering

| Q-decomp | Retrieval (Triplets) | Top-k | EM | F1 |
|---|---|---|---|---|
| False | Dense | 5 | 0.203 | 0.314 |
| | | 10 | 0.157 | 0.268 |
| | | 15 | 0.190 | 0.312 |
| | | 20 | 0.213 | 0.334 |
| | | 30 | 0.210 | 0.338 |
| | | 40 | 0.220 | 0.364 |
| | | 50 | 0.223 | 0.349 |
| | | 70 | 0.200 | 0.328 |
| | | 100 | 0.210 | 0.335 |
| | Sparse | 5 | 0.220 | 0.326 |
| | | 10 | 0.237 | 0.357 |
| | | 15 | 0.250 | 0.375 |
| | | 20 | 0.230 | 0.356 |
| | | 30 | 0.277 | 0.416 |
| | | 40 | 0.247 | 0.381 |
| | | 50 | 0.253 | 0.378 |
| | | 70 | 0.180 | 0.275 |
| | | 100 | 0.210 | 0.333 |
| True | Dense | 5 | 0.380 | 0.457 |
| | | 10 | 0.393 | 0.471 |
| | | 15 | 0.417 | 0.493 |
| | | 20 | 0.430 | 0.514 |
| | | 30 | 0.437 | 0.511 |
| | | 40 | 0.417 | 0.489 |
| | | 50 | ***0.447** | 0.529 |
| | | 70 | 0.433 | ***0.530** |
| | | 100 | 0.417 | 0.509 |
| | Sparse | 5 | 0.280 | 0.358 |
| | | 10 | 0.350 | 0.451 |
| | | 15 | 0.370 | 0.472 |
| | | 20 | 0.397 | 0.502 |
| | | 30 | 0.407 | 0.505 |
| | | 40 | 0.400 | 0.500 |
| | | 50 | 0.373 | 0.468 |
| | | 70 | 0.347 | 0.424 |
| | | 100 | 0.367 | 0.456 |

Further insights are gained from the results presented in Table 6, which explores the impact of retrieval quality. It explores a hypothetical gold retriever which retrieves all the triplets from the supporting paragraphs only and a naive retriever which retrieves all the triplets from all paragraphs. The system with gold retriever achieved an F1 score of 0.646, significantly higher than any other configuration, and also very close to the best performing baseline model, Step Execution by Select+Answer (EX(SA)),

on validation-mini dataset (see Table 3). Interestingly, under gold retrieval conditions, the advantage of question decomposition diminishes, as indicated by the closer performance metrics between configurations with and without question decomposition. This suggests that the primary benefit of question decomposition lies in its ability to enhance the relevance and precision of the retrieved information from the knowledge graph.

Further examination of the data reveals a notable performance drop when utilizing naive retrieval, as shown in Table 6. Specifically, configurations with naive retrieval exhibited a drastic reduction in both EM and F1 scores, with the best naive retrieval setup only achieving an F1 score of 0.357. This significant decline highlights the critical role of the retrieval step in the KGQA system's effectiveness. It can be inferred that the LLM's question answering capability drops in the presence of unrelated information due to poor retrieval. Effective retrieval is essential for locating relevant information within the knowledge graph, and suboptimal retrieval mechanisms, such as the naive retrieval, severely impair the system's ability to provide accurate answers. This underscores the necessity of further refining retrieval techniques to enhance the overall performance of KGQA systems.

Table 6: Impact of retrieval on the performance of knowledge graph based question answering

| **Retrieval (Triplets)** | **Q-decomp** | **EM** | **F1** |
|---|---|---|---|
| Naive | False | 0.145 | 0.224 |
| Naive | True | 0.285 | 0.357 |
| Gold | False | **0.525** | **0.646** |
| Gold | True | 0.510 | 0.624 |

Our inspection of the results of the experiment with "Gold" retriever revealed several distinct failure modes which can largely be attributed to the following causes: missing triplets, insufficient question-answering capabilities of the LLM, and challenges in handling of temporal dynamics. The most frequent cause of failure was missing triplets in the knowledge graph, accounting for 12 instances. This underscores the importance of having a comprehensive and robust knowledge graph to support accurate fact retrieval and reasoning. Even when the necessary triplets were present, 8 of the failures were due to inadequate reasoning by the LLM. Another notable failure mode, affecting 2 samples, involves the system's inability to correctly handle temporal information. An example of this issue is detailed in Fig. 13, illustrating the system's failure to distinguish between current and former roles, which led to an incorrect answer. The failure occurred because the triplets do not differentiate between the current and former chief judges. The system incorrectly identified "Rosaline Bozimo" as the chief judge due to her previous appointment, overlooking that she had been succeeded by "Honorable Justice Abiodun Smith".

**Question Decomposition:**
Who is the chief judge of Friday Osanebi's birthplace?

- Where was Friday Osanebi born in? → Delta State
- Who is the chief judge of Delta State? → Honorable Justice Abiodun Smith

**Supporting Paragraphs:**

- Friday Ossai Osanebi (born August 7, 1980) is a Nigerian and a member of the Delta State House of Assembly the Lawmaker representing Ndokwa East Local Government Constituency in the State House of Assembly.
- Rosaline Patricia Irorefe Bozimo (born 1 January 1946) is a Nigerian lawyer who was appointed Chief Justice of Delta State with effect from 23 March 2003. She retired on 1 January 2011 and was succeeded by Honorable Justice Abiodun Smith.

**Triplets (irrelevant ones are excluded):**

- Friday Osanebi | represented in | Delta State House of Assembly
- ...
- Rosaline Bozimo | position | Chief Justice of Delta State
- Rosaline Bozimo | tenure start | March 23, 2003
- Rosaline Bozimo | retirement date | January 1, 2011
- Rosaline Bozimo | successor | Honorable Justice Abiodun Smith

**Reference Answer:** Honorable Justice Abiodun Smith

**Predicted Answer:** Rosaline Bozimo

Figure 13: An example failure of KGQA due to temporal dynamics

In summary, the experimental results substantiate the effectiveness of question decomposition, and the quality of knowledge graph in KGQA performance.

### 5.3.2 Question Answering with CTE prompting

This section discusses the experimental results for the question answering with standard prompting (given in Fig. 6) and CTE prompting (given in Fig. 9) using paragraphs as context instead of triplets. The experiments evaluate different aspects of the model, including question decomposition, type of retrieval method, and prompting techniques. The results of the experiments are detailed in Table 7.

The most notable finding from the experiments is the significant impact of question decomposition on the performance of the question answering system. Models utilizing question decomposition consistently outperform those without it across different retrieval methods and prompting techniques. Specifically, the model combining question decomposition, dense retrieval with top-3, and CTE prompting emerges as the best-performing configuration, achieving an EM score of 0.547 and an F1 score of 0.654.

The analysis of retrieval methods reveals that there is no significant difference in performance between dense and sparse retrieval methods when question decomposition is applied. Both methods yield comparable results, suggesting that the choice between dense and sparse retrieval might be more dependent on specific use cases or scalability requirements rather than inherent performance benefits.

The experiments also highlight the effectiveness of the CTE prompting technique over the standard prompting method. Models employing CTE prompting generally show higher performance metrics, which is evident from the improved scores across both retrieval methods. This indicates that CTE prompting might be better suited for extracting and utilizing context from paragraphs when answering complex multi-hop questions.

Overall, these results underscore the importance of question decomposition in enhancing the performance of question answering systems and suggest that the integration of CTE prompting can further refine the effectiveness of the model in handling complex queries.

### 5.3.3 Evaluating Impact of Individual Components and Parameters

We conducted additional experiments to examine the impact of various components and parameters more thoroughly. Our investigations covered several key areas: assessing how different prompting techniques affect question answering, examining the role of temperature values in language model generation, exploring the impact of language model adaptation techniques on JERX, and evaluating retrieval performance of different retrieval methods. These studies provided insights into how each specific element contributes to the system's functionality and performance.

#### 5.3.3.1 Impact of Prompting Technique on Question Answering

Different techniques have been explored to measure the impact of prompting technique on question answering performance. We explore standard, zero-shot and few-shot variants of CoT [28] prompting techniques, and CTE prompting technique proposed by this study. Their prompts are provided, in Fig. 6, in Fig. 7, Fig. 8, and in Fig. 9, respectively.

Table 7: Evaluation Results for Paragraph based Question Answering

| Q-decomp | Prompting | Retrieval (Paragraphs) | Top-k | EM | F1 |
|---|---|---|---|---|---|
| False | CTE | Dense | 3 | 0.260 | 0.369 |
| | | | 5 | 0.310 | 0.408 |
| | | | 10 | 0.347 | 0.477 |
| | | Sparse | 3 | 0.170 | 0.278 |
| | | | 5 | 0.277 | 0.384 |
| | | | 10 | 0.347 | 0.473 |
| | Standard | Dense | 3 | 0.233 | 0.316 |
| | | | 5 | 0.180 | 0.259 |
| | | | 10 | 0.137 | 0.194 |
| | | Sparse | 3 | 0.190 | 0.283 |
| | | | 5 | 0.157 | 0.259 |
| | | | 10 | 0.123 | 0.185 |
| True | CTE | Dense | 3 | ***0.547** | ***0.654** |
| | | | 5 | 0.503 | 0.605 |
| | | | 10 | 0.537 | 0.644 |
| | | Sparse | 3 | 0.533 | 0.627 |
| | | | 5 | 0.493 | 0.608 |
| | | | 10 | 0.500 | 0.593 |
| | Standard | Dense | 3 | 0.483 | 0.572 |
| | | | 5 | 0.420 | 0.494 |
| | | | 10 | 0.370 | 0.448 |
| | | Sparse | 3 | 0.433 | 0.500 |
| | | | 5 | 0.383 | 0.448 |
| | | | 10 | 0.360 | 0.423 |

In the experiments, only supporting paragraphs were used as the context to eliminate noise, and the questions were not decomposed into sub-questions. Each experiment was repeated three times, and the scores were averaged over the runs. The *GPT-3.5 Turbo* model was used for question answering, and the temperature was set to 0.1 to introduce variance in the generations.

As shown in Table 8, few-shot CoT prompting achieves the highest performance in the EM metric, slightly surpassing CTE prompting. In terms of the F1 metric, CTE prompting yields the highest score among all prompting techniques. It can be seen that CTE prompting significantly improves multi-hop question-answering capabilities of the LLM compared to the standard prompting technique.

Table 8: Evaluation results for question answering with different prompting techniques

| Prompting Technique | EM | F1 |
|---|---|---|
| Standard | 0.530 | 0.637 |
| Zero-shot CoT | 0.527 | 0.653 |
| Few-shot (n=1) CoT | 0.530 | 0.623 |
| Few-shot (n=1) CTE | **0.573** | **0.703** |

Table 9 presents a comparison of token usage between CoT and CTE prompting techniques. Token counts were calculated by applying the GPT-3.5 Turbo tokenizer to the LLM's outputs when answering questions from the MuSiQue-Ans validation-mini subset for each prompting technique. The results indicate that CTE prompting is markedly more efficient, utilizing significantly fewer tokens than CoT prompting. Specifically, CTE prompting consumes approximately 69.7% fewer tokens on average, with a mean token count of 30.2 compared to 99.6 for CoT prompting. This efficiency is consistently reflected across all statistical measures, highlighting CTE prompting's ability to achieve better outcomes with considerably less computational overhead.

Table 9: Comparison of output token counts for CTE and CoT prompting techniques

| **Prompting Technique** | **Number of Tokens** | | | |
|---|---|---|---|---|
| | **Mean** | **Min** | **Median** | **Max** |
| Few-shot (n=1) CoT | 99.6 | 52.0 | 95.0 | 192.0 |
| Few-shot (n=1) CTE | **30.2** | **16.0** | **30.0** | **57.0** |

Table 10 compares the results for CTE and CoT prompting techniques on an example taken from MuSiQue-Ans validation set. It can be seen that CTE prompting technique reaches to correct answer in shorter generation compared to COT.

Table 10: Comparison of CTE and CoT prompting techniques on an example data point

| **Question** | In which county is the city to which KKVU is licensed to broadcast? | |
|---|---|---|
| **Reference Answer** | Berrien County | |
| **Supporting Paragraphs** | KKVU (104.5 FM, "U 104.5") is a commercial radio station licensed to Stevensville, Montana, serving the Missoula, Montana area.<br>Stevensville is a village in Berrien County in the U.S. state of Michigan. | |
| **Prompting Technique** | **CTE** | **CoT** |
| **Reasoning** | KKVU \| licensed to \| Stevensville, Montana<br>Stevensville, Michigan \| located in \| Berrien County | The radio station KKVU is licensed to broadcast in Stevensville, Montana. Stevensville, Montana is located in the state of Montana. Therefore, the county in which the city of Stevensville, Montana is located is Missoula County. |
| **Answer** | Berrien County | Missoula County |

#### 5.3.3.2 Impact of Temperature Value on Question Answering

Additional experiments were conducted to evaluate the influence of temperature parameter on the performance of LLM's question answering. This analysis utilized both standard and CTE prompting strategies, employing only supporting paragraphs as context and without question decomposition. According to the results presented in Table 11, EM and F1 scores remain relatively stable for temperature values less than 1 for each prompting technique. However, a significant decline in both metrics for

both prompting techniques is observed when the temperature exceeds 1, with a sharp drop occurring at 2.

Table 11: Question answering performances with different temperature values

| Prompting Technique | Temperature | EM | F1 |
|---|---|---|---|
| Standard | 0.0 | 0.530 | 0.636 |
| | 0.1 | 0.537 | 0.649 |
| | 0.3 | 0.503 | 0.618 |
| | 0.5 | 0.497 | 0.615 |
| | 0.7 | 0.500 | 0.614 |
| | 1.0 | 0.477 | 0.592 |
| | 1.5 | 0.480 | 0.583 |
| | 2.0 | 0.363 | 0.448 |
| CTE | 0.0 | 0.563 | 0.699 |
| | 0.1 | 0.563 | 0.699 |
| | 0.3 | 0.563 | 0.687 |
| | 0.5 | 0.563 | 0.681 |
| | 0.7 | 0.577 | 0.697 |
| | 1.0 | 0.543 | 0.667 |
| | 1.5 | 0.553 | 0.677 |
| | 2.0 | 0.147 | 0.188 |

#### 5.3.3.3 Impact of LLM Adaptation Techniques for JERX

We also explore various LLM adaptation techniques for JERX task, specifically targeting their efficacy in multi-hop question answering. Employing the *Llama-3-8B* model, we use zero-shot prompting (described in Section 4.2.1), few-shot prompting (described in Section 4.2.2), and supervised fine-tuning (SFT) (described in Section 4.2.3) to extract entity-relation-entity triplets from paragraphs. These triplets are then used for answering the question without decomposing it using standard prompt given in Fig. 6 with GPT-3.5 Turbo. To understand the influence of these adaptation techniques, we evaluate their performances based on the quality of triplets they extract. Hence, in these experiments, to ensure the focus remains solely on the triplet quality without interference from extraneous variables, only the triplets from the supporting paragraphs are used.

Table 12 presents the performance metrics for joint entity-relation extraction on the WebNLG validation set, comparing few-shot prompting and supervised fine-tuning techniques.

As it can be seen from Table 12, supervised fine-tuned model significantly outperforms few-shot model, indicating that the model adapts to WebNLG dataset.

Despite the various approaches employed in adapting the LLM for the JERX task, the performance metrics in Table 13 demonstrate only slight differences in effectiveness for the downstream question answering task. Few-shot prompted JERX model leads to the best question answering performance in both metrics and slightly surpasses the supervised fine-tuned model. This reflects that although the techniques, few-shot prompting, and SFT, differ significantly in their approach, they yield similar

Table 12: JERX performances on WebNLG validation set

| Model | Precision | Recall | F1 |
|---|---|---|---|
| Zero-shot Prompted | 0.262 | 0.250 | 0.251 |
| Few-shot (n=2) Prompted | 0.547 | 0.542 | 0.538 |
| Supervised Fine-Tuned | **0.888** | **0.906** | **0.894** |

quality of the extracted triplets for multi-hop question answering. The significant performance improvement observed with the supervised fine-tuned model on the WebNLG dataset does not translate into any performance gain for the question answering task on the MuSiQue-Ans dataset.

Table 13: Question answering performances with different JERX models

| Model | EM | F1 |
|---|---|---|
| Zero-shot Prompted | 0.500 | 0.601 |
| Few-shot (n=2) Prompted | **0.563** | **0.661** |
| Supervised Fine-Tuned | 0.557 | 0.646 |

#### 5.3.3.4 Retrieval Performances of Different Retrievers

As observed from the experimental results presented in Section 5.3, retrieving correct information from a corpus is an important step for question answering systems. Hence, we conducted an experiment to assess the performance of two retrieval types, dense and sparse, implemented as described in section 4.4.2 and 4.4.1, respectively. The performances of retrieval algorithms were assessed using the *recall* metric, which is described in 4.7.1, on MuSiQue-Ans validation set. For each sample, the metric was first calculated separately and then averaged over all samples to assess the overall performance.

For the samples with two-hop question in MuSiQue-Ans dataset, only 2 of 20 paragraphs are relevant to answer the question. In other words, for each sub-question, there is only one supporting paragraph in any sample. Hence, recall metric is calculated for each sub-question and the overall performance per sample is defined as success of retrieval for both sub-questions. In the experiments, sub-questions are used as retrieval queries, where each query builds upon the reference answer for the previous sub-question provided in the dataset. For instance, if the first sub-question asks "Who is Mahmoud Mirza's father?" with the reference answer of 'Mohammad Ali Shah Qajar' and the second sub-question asks "Who is the child of #1?", the query for the second sub-question becomes "Who is the child of Mohammad Ali Shah Qajar?". This approach enables precise evaluation of the retrieval system's capacity to manage sequential information retrieval tasks, focusing purely on the accuracy of retrieving relevant information based on the given references.

In Table 14, a comparative analysis is provided on the retrieval performance between two algorithms, dense and sparse, across different retrieval depths (top-k values of 3, 5, and 10). The results are segmented into performance metrics for first and second sub-questions, and also the overall performance considering both sub-questions. Detailed within each segment are the success rates for both algorithms, enabling direct comparison. The highest overall success rates achieved by each algorithm at various top-k thresholds are highlighted in bold, demonstrating their effectiveness under varying conditions.

It is observed that the dense retrieval algorithm generally outperforms sparse retrieval, as indicated by the higher retrieval success rates in most settings, consistent with the observations in [13].

Table 14: Comparison of retrieval algorithms by recall

| Top-k | Sub-question 1 | | Sub-question 2 | | Overall | |
|---|---|---|---|---|---|---|
| | Dense | Sparse | Dense | Sparse | Dense | Sparse |
| 3 | **0.91** | 0.81 | **0.89** | 0.85 | **0.81** | 0.69 |
| 5 | **0.95** | 0.91 | 0.94 | **0.96** | **0.89** | 0.87 |
| 10 | **0.97** | **0.97** | 0.96 | **0.97** | 0.93 | **0.94** |

### 5.4 Discussion

Several insights can be drawn from the experimental results presented in this study.

The effectiveness of question decomposition in enhancing question answering performance is also underscored by the results in Tables 5 and 7, aligning with previous research [90].

In the context of joint entity-relation extraction (JERX) on the WebNLG dataset, the fine-tuned LLM significantly outperforms the few-shot prompted LLM, as evidenced by the results in Table 12. Despite this superiority in JERX tasks, such enhancement does not translate into improved question answering performance. As outlined in Table 13, the overall question answering efficiency of systems utilizing different JERX models remains similar, with the few-shot prompted LLM yielding the best results on the MuSiQue-Ans dataset. This indicates that the additional accuracy in extracting entity-relation-entity triplets through fine-tuning does not necessarily benefit the question answering tasks. Several factors might contribute to this phenomenon: potential domain discrepancies between the WebNLG and MuSiQue datasets, a propensity for the fine-tuned model to overfit to the WebNLG dataset, and the robust baseline capabilities of the Llama-3-8B model, which suggest that further fine-tuning may not be essential for sustaining performance across different contexts.

Moreover, the results from Table 7 indicate that CTE prompting, combined with paragraph-based context, delivers the best outcomes for the MuSiQue-Ans dataset. These results validate our hypothesis stated in Chapter 4 that the entity-relation-entity triplets enhance multi-hop reasoning capabilities by providing a clear reasoning path that connects the source entity in the question to the target entity in the answer, as evidenced by the superior performance in multi-hop question answering presented in Table 7. However, utilizing triplets extracted by an external model rather than the question answering model does not achieve comparable performance as seen from Table 5. This disparity could be attributed to several reasons: the absence of direct questioning might lead the LLM to overlook vital entities and relations; paragraph contexts provide a richer informational fabric from which relevant triplets can be discerned; and the discrepancy in model capabilities, where the Llama-3-8B, which has 8 billion parameter, used for JERX tasks is less capable than the *GPT-3.5 Turbo* employed for question answering, speculated to possess around 20 billion parameters [113].

Additionally, knowledge graph-based question answering showcases performance on par with other baseline approaches as seen in Table 5. This method offers significant practical benefits such as struc-

tured data storage, easy updates to the knowledge base, and the ability of answering alternative questions. The primary challenge, however, lies in the quality of the knowledge graph, which is a pivotal factor influencing performance outcomes. Knowledge graph-based question answering falls short of 0.447 EM and 0.530 F1 score as presented in Table 5, while question answering based on paragraphs with CTE prompting achieves 0.547 EM and 0.654 F1 score as presented in Table 7. Both system utilize triplets while answering the question but yields significantly different performances. This suggests that the knowledge graphs pre-constructed with the triplets extracted by the JERX LLM potentially miss necessary information to answer the question. This highlights the importance of the quality and completeness of knowledge graph for question answering performance.

# CHAPTER 6

# CONCLUSIONS

In this chapter, we provide a summary of the key findings, discuss the limitations of the study, and outline potential directions for future research.

## 6.1 Conclusion

This thesis investigates the use of knowledge graphs and LLMs for multi-hop question answering. Through a series of experiments, we explored various settings for parameters such as question decomposition, retrieval methods, context types, and prompting techniques.

A significant contribution of this work is the proposal of a new prompting technique, named Connect-the-Entities (CTE), for multi-hop question answering. Our findings show that instructing the model to extract entity-relation-entity triplets using CTE prompting improves its question answering performance. Additionally, CTE prompting proves to be more cost-effective and faster than chain-of-thought (COT) prompting, as it requires the generation of fewer tokens.

Many experiments were conducted to understand the impact of different components: question decomposition, retrieval methods, prompting techniques, temperature values for LLM generation, and adaptation techniques for LLMs on the JERX task. The results emphasize the importance of question decomposition in multi-hop question answering, confirming findings from previous research. Furthermore, it was observed that dense retrievers outperform sparse retrievers in locating relevant paragraphs, which is consistent with prior studies. The results also indicate that the combination of CTE prompting and using paragraphs as context yields the best question answering performance for two-hop questions in the MuSiQue-Ans dataset.

The study also demonstrates that using a knowledge graph, pre-built with an LLM on paragraphs of a sample, as a knowledge base for questions yields performance comparable to baseline systems. However, it lags behind the best-performing system, which utilizes paragraphs as context and CTE prompting. These results suggest that the quality of triplets is crucial for effective knowledge graph-based question answering.

In summary, this thesis highlights the potential of combining knowledge graphs and LLMs for multi-hop question answering, particularly through innovative prompting techniques like CTE. The findings underscore the importance of optimizing various parameters and components to achieve optimal performance in complex question answering tasks. Future work could focus on improving the quality

of the triplets in knowledge graphs and exploring further enhancements in retrieval and prompting techniques.

### 6.2 Limitations

This section discusses the limitations of the proposed method for multi-hop question answering using knowledge graphs and large language models. These limitations encompass the range of question complexity the method can handle, its tested applicability across various datasets, and its ability to address temporal changes in data.

The method has currently been validated solely for two-hop questions. Although theoretically, there are no fundamental barriers to extending it to three-hop or more complex questions, such capabilities have not yet been demonstrated or empirically tested. Extending the method to handle more than two hops introduces several challenges that need to be addressed. As the number of hops increases, the complexity of decomposing the question into sub-questions using an LLM also increases, requiring more sophisticated question decomposition strategies. Additionally, with more hops, there is a higher likelihood of encountering data sparsity, where nodes in the knowledge graph may have fewer connections or missing links between entities, making it difficult to identify relevant pathways. Error propagation is another significant challenge, as errors that occur in the initial hops, whether during question answering or retrieval, can accumulate and negatively impact the accuracy of the final answer. The impact of these errors becomes more pronounced as the number of hops increases. Furthermore, the presence of ambiguity and noise in the knowledge graph tends to grow with the number of hops, complicating the process of distinguishing between relevant and irrelevant or misleading information and affecting the reliability of the answers. Future investigations are needed to confirm the adaptability and performance of the method when answering questions with more hops, taking into account these potential challenges.

In terms of dataset applicability, the method has only been evaluated using MuSiQue-Ans dataset. While this dataset is relatively new and presents a considerable challenge due to its complex and diverse question structure, there is a need to test the method's effectiveness on other datasets. Different datasets may feature unique styles of questioning and data structuring, which could impact the method's performance and generalizability. The experiments were conducted on a limited subset of data points randomly selected from validation set due to time and budget constraints. This is also a major limitation for the verification of our methods' effectiveness.

The proposed method does not currently support the effective encoding and querying of temporal dynamics within knowledge graphs as previously discussed in Section 5.3.1 and demonstrated in Fig. 13. For example, it struggles with questions like "Who was the mayor of Berlin before Angela Merkel became Chancellor?" or "What was the position of Elon Musk in Tesla before 2010?". These examples illustrate challenges in capturing and querying historical changes or sequences of events that are critical for understanding the temporal context of data.

Another possible limitation of this study could be quantization of LLMs to overcome the constraints of computational resources required for SFT and RL experiments. Although previous studies [43] demonstrate that quantized models can achieve state-of-the-art performance on various benchmarks when fine-tuned, it is yet to be explored for JERX task.

Each of these limitations points to essential areas for future research, aiming to enhance the method's robustness, scalability, and accuracy across diverse querying conditions and data sets.

### 6.3 Future Works

This research opens up several avenues for further investigation.

An immediate follow-up could be conducting the experiments on full validation set and also test set of MuSiQue-Ans dataset. This would provide us more reliable performance results while comparing our methods with other approaches.

Another follow-up could involve testing the proposed approach on additional multi-hop question answering datasets and extending the algorithm to support questions with more hops. It remains to be examined whether a knowledge graph offers advantages in information retrieval over triplets compared to retrieval over paragraphs as the number of hops increases.

Another area for exploration is the effectiveness of knowledge graph-based question answering systems when the number of documents in the corpus increases. It can be claimed that knowledge graphs, which store information in connected subgraphs, might not suffer a decrease in retrieval performance despite the addition of more documents, potentially irrelevant ones. Testing this hypothesis would involve increasing the number of irrelevant paragraphs in samples and measuring the impact on the system's question answering performance.

Further research could also include the use of rich and diverse data to supervised fine-tune an LLM for the JERX task. In this study, supervised fine-tuning an LLM on the WebNLG dataset did not significantly improve the quality of the extracted triplets for question answering. A potential direction could be to curate a larger and more diverse dataset, possibly using synthetic data generated by a state-of-the-art LLM like GPT-4, to enhance outcomes.

While supervised fine-tuning has potential, it also risks producing entity-relation-entity triplets that may not be directly useful for question answering. In this research, we attempted to fine-tune the LLM using reinforcement learning with question answering performance as the reward signal. However, this approach did not successfully enhance the model's performance, which may be attributed to suboptimal hyper-parameter settings. Future work could focus on exploring different configurations of hyper-parameters and training strategies in reinforcement learning to potentially develop a more robust model. This model would be better tuned to extract triplets that are most beneficial for answering complex multi-hop questions.

# Bibliography


[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[2] OpenAI, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[3] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[5] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, "Unifying large language models and knowledge graphs: A roadmap," *IEEE Transactions on Knowledge and Data Engineering*, p. 1–20, 2024.

[6] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics, 2018.

[7] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, " musique: Multihop questions via single-hop question composition," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 539–554, 2022.

[8] L. Weng, "How to build an open-domain question answering system?," *lilianweng.github.io*, Oct 2020. Accessed: 2024-07-28.

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

[10] S. E. Robertson, S. Walker, M. Beaulieu, M. Gatford, and A. Payne, "Okapi at trec-4," *Nist Special Publication Sp*, pp. 73–96, 1996.

[11] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," *arXiv preprint arXiv:1704.00051*, 2017.

[12] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin, "End-to-end open-domain question answering with bertserini," *arXiv preprint arXiv:1902.01718*, 2019.

[13] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2020.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," *OpenAI Blog*, 2018. Accessed: July, 2024.

[16] F. Petroni, P. Lewis, A. Piktus, T. Rocktäschel, Y. Wu, A. H. Miller, and S. Riedel, "How context affects language models' factual predictions," *ArXiv*, vol. abs/2005.04611, 2020.

[17] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, *et al.*, "Sparks of artificial general intelligence: Early experiments with gpt-4," *arXiv preprint arXiv:2303.12712*, 2023.

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

[19] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.

[20] S. An, Z. Lin, Q. Fu, B. Chen, N. Zheng, J.-G. Lou, and D. Zhang, "How do in-context examples affect compositional generalization?," *arXiv preprint arXiv:2305.04835*, 2023.

[21] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, jan 2020.

[22] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27730–27744, 2022.

[23] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, *et al.*, "Lamda: Language models for dialog applications," *arXiv preprint arXiv:2201.08239*, 2022.

[24] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[25] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, p. 1–38, Mar. 2023.

[26] H. Ye, T. Liu, A. Zhang, W. Hua, and W. Jia, "Cognitive mirage: A review of hallucinations in large language models," *arXiv preprint arXiv:2309.06794*, 2023.

[27] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, *et al.*, "A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity," *arXiv preprint arXiv:2302.04023*, 2023.

[28] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824–24837, 2022.

[29] D. Wu, J. Zhang, and X. Huang, "Chain of thought prompting elicits knowledge augmentation," in *Findings of the Association for Computational Linguistics: ACL 2023* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 6519–6534, Association for Computational Linguistics, July 2023.

[30] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. V. Le, and E. H. Chi, "Least-to-most prompting enables complex reasoning in large language models," in *The Eleventh International Conference on Learning Representations*, 2023.

[31] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, 2023.

[32] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Advances in neural information processing systems*, vol. 13, 2000.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[34] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[35] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[36] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, vol. 9, no. 1, pp. 147–169, 1985.

[37] M. Peeperkorn, T. Kouwenhoven, D. Brown, and A. Jordanous, "Is temperature the creativity parameter of large language models?," *arXiv preprint arXiv:2405.00492*, 2024.

[38] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," 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)* (C. Zong, F. Xia, W. Li, and R. Navigli, eds.), (Online), pp. 4582–4597, Association for Computational Linguistics, Aug. 2021.

[39] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Online and Punta Cana, Dominican Republic), pp. 3045–3059, Association for Computational Linguistics, Nov. 2021.

[40] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang, "Gpt understands, too," *AI Open*, 2023.

[41] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[42] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[43] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[44] A. Singhal, "Introducing the knowledge graph: Things, not strings." `https://blog.google/products/search/introducing-knowledge-graph-things-not/`, May 2012. Accessed: February, 2024.

[45] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, p. 11–33, jan 2016.

[46] P. Cimiano and H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semant. Web*, vol. 8, p. 489–508, an 2017.

[47] M. Faerber, F. Bartscherer, C. Menne, and A. Rettinger, "Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago," *Semantic Web*, vol. 9, pp. 1–53, 03 2017.

[48] World Wide Web Consortium (W3C), "Rdf 1.1 primer." `https://www.w3.org/TR/rdf11-primer/`, 2014. Last accessed: September 3, 2024.

[49] World Wide Web Consortium (W3C), "Rdf vocabulary description language 1.0: Rdf schema." `https://www.w3.org/TR/rdf-schema/#ch_reificationvocab`, 2004. Section 3.3: Reification, Last accessed: September 3, 2024.

[50] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, p. 494–514, Feb. 2022.

[51] J. Zhang, B. Chen, L. Zhang, X. Ke, and H. Ding, "Neural, symbolic and neural-symbolic reasoning on knowledge graphs," 2021.

[52] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, "Never-ending learning," *Commun. ACM*, vol. 61, p. 103–115, apr 2018.

[53] B. Abu-Salih, "Domain-specific knowledge graphs: A survey," *Journal of Network and Computer Applications*, vol. 185, p. 103076, July 2021.

[54] L. Zhong, J. Wu, Q. Li, H. Peng, and X. Wu, "A comprehensive survey on automatic knowledge graph construction," *ACM Comput. Surv.*, vol. 56, Nov. 2023.

[55] T. Nayak, N. Majumder, P. Goyal, and S. Poria, "Deep neural approaches to relation triplets extraction: a comprehensive survey," *Cognitive Computation*, vol. 13, p. 1215–1232, Aug. 2021.

[56] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Normalizing flow-based neural process for few-shot knowledge graph completion," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, ACM, July 2023.

[57] P. Dognin, I. Padhi, I. Melnyk, and P. Das, "ReGen: Reinforcement learning for text and knowledge base generation using pretrained language models," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, (Online and Punta Cana, Dominican Republic), pp. 1084–1099, Association for Computational Linguistics, Nov. 2021.

[58] I. Melnyk, P. Dognin, and P. Das, "Knowledge graph generation from text," in *Findings of the Association for Computational Linguistics: EMNLP 2022* (Y. Goldberg, Z. Kozareva, and Y. Zhang, eds.), (Abu Dhabi, United Arab Emirates), pp. 1610–1622, Association for Computational Linguistics, Dec. 2022.

[59] Y. Zhu, X. Wang, J. Chen, S. Qiao, Y. Ou, Y. Yao, S. Deng, H. Chen, and N. Zhang, "Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities," *arXiv preprint arXiv:2305.13168*, 2023.

[60] P.-L. Huguet Cabot and R. Navigli, "REBEL: Relation extraction by end-to-end language generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021* (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Punta Cana, Dominican Republic), pp. 2370–2381, Association for Computational Linguistics, Nov. 2021.

[61] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Reasoning on graphs: Faithful and interpretable large language model reasoning," *arXiv preprint arXiv:2310.01061*, 2023.

[62] W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (C. Zong and M. Strube, eds.), (Beijing, China), pp. 1321–1331, Association for Computational Linguistics, July 2015.

[63] J. Baek, A. F. Aji, and A. Saffari, "Knowledge-augmented language model prompting for zero-shot knowledge graph question answering," *ArXiv*, vol. abs/2306.04136, 2023.

[64] P. Sen, S. Mavadia, and A. Saffari, "Knowledge graph-augmented language models for complex question answering," in *ACL 2023 Workshop on Natural Language Reasoning and Structured Explanations*, 2023.

[65] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2 ed., 2018.

[66] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[67] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[68] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[69] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27730–27744, 2022.

[70] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, *et al.*, "Constitutional ai: Harmlessness from ai feedback," *arXiv preprint arXiv:2212.08073*, 2022.

[71] M. Kwon, S. M. Xie, K. Bullard, and D. Sadigh, "Reward design with language models," *arXiv preprint arXiv:2303.00001*, 2023.

[72] A. Singh, J. D. Co-Reyes, R. Agarwal, A. Anand, P. Patil, P. J. Liu, J. Harrison, J. Lee, K. Xu, A. Parisi, *et al.*, "Beyond human data: Scaling self-training for problem-solving with language models," *arXiv preprint arXiv:2312.06585*, 2023.

[73] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, "Fine-tuning language models from human preferences," *arXiv preprint arXiv:1909.08593*, 2019.

[74] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano, "Learning to summarize with human feedback," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 3008–3021, Curran Associates, Inc., 2020.

[75] A. Glaese, N. McAleese, M. Trębacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker, *et al.*, "Improving alignment of dialogue agents via targeted human judgements," *arXiv preprint arXiv:2209.14375*, 2022.

[76] D. Cohen, M. Ryu, Y. Chow, O. Keller, I. Greenberg, A. Hassidim, M. Fink, Y. Matias, I. Szpektor, C. Boutilier, *et al.*, "Dynamic planning in open-ended dialogue using reinforcement learning," *arXiv preprint arXiv:2208.02294*, 2022.

[77] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

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

[79] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.

[80] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M.-W. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov, "Natural questions: A benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 452–466, 2019.

[81] P. Lewis, B. Oguz, R. Rinott, S. Riedel, and H. Schwenk, "MLQA: Evaluating cross-lingual extractive question answering," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 7315–7330, Association for Computational Linguistics, July 2020.

[82] H. Elsahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, F. Laforest, and E. Simperl, "T-REx: A large scale alignment of natural language with knowledge base triples," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)* (N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga, eds.), (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.

[83] O. Levy, M. Seo, E. Choi, and L. Zettlemoyer, "Zero-shot relation extraction via reading comprehension," in *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)* (R. Levy and L. Specia, eds.), (Vancouver, Canada), pp. 333–342, Association for Computational Linguistics, Aug. 2017.

[84] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed, "Big bird: Transformers for longer sequences," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 17283–17297, Curran Associates, Inc., 2020.

[85] J. Zhang, H. Zhang, D. Zhang, Y. Liu, and S. Huang, "Beam retrieval: General end-to-end retrieval for multi-hop question answering," *arXiv preprint arXiv:2308.08973*, 2023.

[86] A. Saxena, A. Tripathi, and P. Talukdar, "Improving multi-hop question answering over knowledge graphs using knowledge base embeddings," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 4498–4507, Association for Computational Linguistics, July 2020.

[87] Y. Zhu, L. Pang, Y. Lan, H. Shen, and X. Cheng, "Adaptive information seeking for open-domain question answering," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Online and Punta Cana, Dominican Republic), pp. 3615–3626, Association for Computational Linguistics, Nov. 2021.

[88] K. Ma, H. Cheng, Y. Zhang, X. Liu, E. Nyberg, and J. Gao, "Chain-of-skills: A configurable model for open-domain question answering," *arXiv preprint arXiv:2305.03130*, 2023.

[89] S. Li, X. Li, L. Shang, X. Jiang, Q. Liu, C. Sun, Z. Ji, and B. Liu, "Hopretriever: Retrieve hops over wikipedia to answer complex questions," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, p. 13279–13287, May 2021.

[90] A. O. KIZILDAĞ, "Semi-automatic prompting approach with question decomposition for multi-hop question answering," Master's thesis, Middle East Technical University, 2023.

[91] S. Min, V. Zhong, L. Zettlemoyer, and H. Hajishirzi, "Multi-hop reading comprehension through question decomposition and rescoring," *arXiv preprint arXiv:1906.02916*, 2019.

[92] V. Mavi, A. Jangra, and A. Jatowt, "A survey on multi-hop question answering and generation," *arXiv preprint arXiv:2204.09140*, 2022.

[93] X. Cao and Y. Liu, "Coarse-grained decomposition and fine-grained interaction for multi-hop question answering," *Journal of Intelligent Information Systems*, vol. 58, no. 1, pp. 21–41, 2022.

[94] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal, "Decomposed prompting: A modular approach for solving complex tasks," *arXiv preprint arXiv:2210.02406*, 2022.

[95] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating training corpora for NLG micro-planners," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (R. Barzilay and M.-Y. Kan, eds.), (Vancouver, Canada), pp. 179–188, Association for Computational Linguistics, July 2017.

[96] AI@Meta, "Llama 3 model card," 2024.

[97] B. Xu, A. Yang, J. Lin, Q. Wang, C. Zhou, Y. Zhang, and Z. Mao, "Expertprompting: Instructing large language models to be distinguished experts," *arXiv preprint arXiv:2305.14688*, 2023.

[98] X. Amatriain, "Prompt design and engineering: Introduction and advanced methods," *arXiv preprint arXiv:2401.14423*, 2024.

[99] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *arXiv preprint arXiv:2302.11382*, 2023.

[100] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[101] Meta, "How-to guide: Prompting in llama," 2023. Accessed: 2024-09-08.

[102] S. M. Bsharat, A. Myrzakhan, and Z. Shen, "Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4," *arXiv preprint arXiv:2312.16171*, 2023.

[103] M. Zheng, J. Pei, and D. Jurgens, "Is" a helpful assistant" the best role for large language models? a systematic evaluation of social roles in system prompts," *arXiv preprint arXiv:2311.10054*, 2023.

[104] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 11 2019.

[105] T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, "The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020)," in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)* (T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, eds.), (Dublin, Ireland (Virtual)), pp. 55–76, Association for Computational Linguistics, 12 2020.

[106] J. W. Ratcliff and D. E. Metzener, "Pattern matching: The gestalt approach," *Dr. Dobb's Journal*, p. 46, July 1988.

[107] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: a nucleus for a web of open data," in *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, (Berlin, Heidelberg), p. 722–735, Springer-Verlag, 2007.

[108] HuggingFace, "Text-generation-inference library." https://github.com/huggingface/text-generation-inference. Accessed: 2024-08-10.

[109] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017.

[110] T. A. I. for Artificial Intelligence, "Musique question answering competition leaderboard." https://leaderboard.allenai.org/musique_ans/submissions/public. Accessed: 2024-08-10.

[111] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

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

[113] M. Singh, J. Cambronero, S. Gulwani, V. Le, C. Negreanu, and G. Verbruggen, "Codefusion: A pre-trained diffusion model for code generation," *arXiv preprint arXiv:2310.17680*, 2023.

# APPENDIX A

# EXTRA MATERIALS FOR KNOWLEDGE GRAPHS

## A.1 Visualizations for knowledge graph

To effectively illustrate the structure of a knowledge graph, Figures 14 and 15 present visualizations derived from an example data point in the MuSiQue-Ans validation set. These figures display a network constructed from entity-relation-entity triplets, which are extracted from the paragraphs of a sample. Each node in the network represents an entity, and the connecting lines, or edges, represent the relationships between these entities. These visualizations help to demonstrate how different pieces of information are interconnected within the knowledge graph.

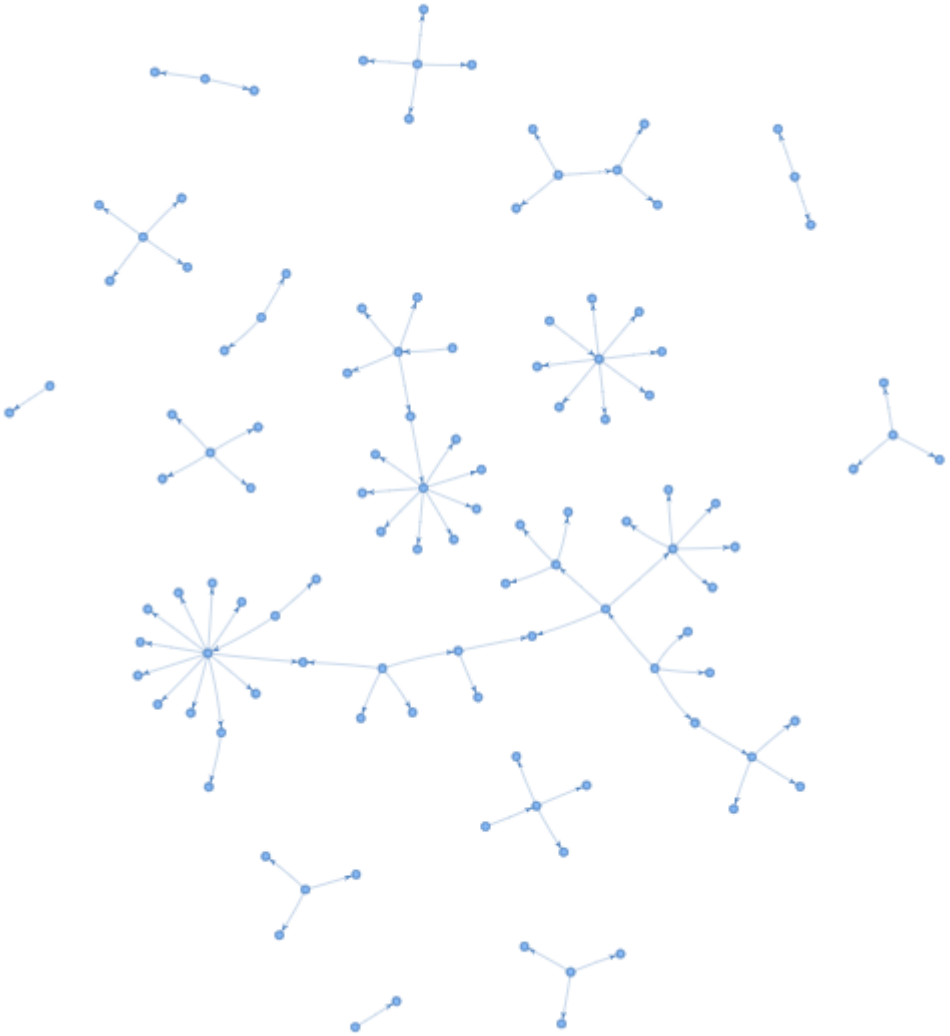

Figure 14: High-Level Network Layout of a Knowledge Graph




Figure 15: Detailed View of a Knowledge Graph

# APPENDIX B

# EXPERIMENTS

As mentioned previously, each experiment is run multiple times to mitigate the impact of randomness, and the average of target metrics across runs is reported. The raw results for all runs of various experiments are provided in this appendix. Additionally, this appendix includes the list of sample IDs for the validation-mini subset, which was used for evaluations in the study.

## B.1 Samples for validation-mini subset

The following are the IDs of the samples in the validation-mini subset used for evaluations in this study:

| | | | |
|---|---|---|---|
| 2hop__575188_342798 | 2hop__175168_110222 | 2hop__488811_30113 | 2hop__278022_71701 |
| 2hop__731584_700117 | 2hop__199513_13732 | 2hop__589264_150107 | 2hop__351045_47134 |
| 2hop__690412_526810 | 2hop__472083_7298 | 2hop__142443_768138 | 2hop__785711_63853 |
| 2hop__263638_69048 | 2hop__785711_73244 | 2hop__559273_152023 | 2hop__418941_429437 |
| 2hop__142842_68489 | 2hop__75169_92673 | 2hop__223655_463572 | 2hop__76314_177869 |
| 2hop__189094_612080 | 2hop__569582_304416 | 2hop__507722_124896 | 2hop__124412_402641 |
| 2hop__723101_150107 | 2hop__385392_756039 | 2hop__642686_7292 | 2hop__307218_161450 |
| 2hop__39954_80303 | 2hop__256778_131879 | 2hop__153532_45326 | 2hop__130984_55721 |
| 2hop__819974_129669 | 2hop__227031_144857 | 2hop__128342_375952 | 2hop__126306_396277 |
| 2hop__6736_6733 | 2hop__427249_20057 | 2hop__497191_129669 | 2hop__152229_604644 |
| 2hop__311694_129669 | 2hop__157251_556157 | 2hop__423972_540249 | 2hop__574419_270105 |
| 2hop__852657_155922 | 2hop__466700_214359 | 2hop__848156_130869 | 2hop__142621_225454 |
| 2hop__121145_561444 | 2hop__477492_240386 | 2hop__539184_119915 | 2hop__374495_68633 |
| 2hop__128420_375952 | 2hop__390947_232243 | 2hop__446324_620302 | 2hop__693444_220151 |
| 2hop__144303_483189 | 2hop__95970_456836 | 2hop__697729_576069 | 2hop__131951_643670 |
| 2hop__862994_69048 | 2hop__123166_47134 | 2hop__131818_161450 | 2hop__850984_7292 |
| 2hop__271045_68633 | 2hop__333020_69048 | 2hop__849534_122868 | 2hop__363216_787353 |
| 2hop__582045_161450 | 2hop__332453_139339 | 2hop__711946_269414 | 2hop__358582_189042 |
| 2hop__607517_161450 | 2hop__149983_108549 | 2hop__199513_801817 | 2hop__518354_67465 |
| 2hop__311931_417706 | 2hop__280451_84616 | 2hop__7855_7794 | 2hop__136067_160249 |
| 2hop__310456_846599 | 2hop__620853_22458 | 2hop__797443_120537 | 2hop__549146_223121 |
| 2hop__776377_857193 | 2hop__226204_69048 | 2hop__92385_2072 | 2hop__88906_55840 |
| 2hop__12039_11957 | 2hop__543853_124498 | 2hop__481809_184805 | 2hop__424908_500483 |

2hop__130867_54221 2hop__304852_758385 2hop__155781_155922 2hop__279729_20057
2hop__341176_711757 2hop__316374_82341 2hop__433694_20273 2hop__726717_610238

### B.2 Experiment results for KGQA

The results for all runs of experiments for multi-hop question answering with knowledge graph, as described in Section 5.3.1, are provided in Table 15.

Table 15: All experiment results for knowledge graph based question answering

| Q-decomp | Retrieval (Triplets) | Top-k | Run | EM | F1 |
|---|---|---|---|---|---|
| False | Dense | 5 | 1 | 0.210 | 0.315 |
| False | Dense | 5 | 2 | 0.200 | 0.316 |
| False | Dense | 5 | 3 | 0.200 | 0.312 |
| False | Dense | 10 | 1 | 0.160 | 0.268 |
| False | Dense | 10 | 2 | 0.150 | 0.263 |
| False | Dense | 10 | 3 | 0.160 | 0.274 |
| False | Dense | 15 | 1 | 0.180 | 0.310 |
| False | Dense | 15 | 2 | 0.200 | 0.329 |
| False | Dense | 15 | 3 | 0.190 | 0.297 |
| False | Dense | 20 | 1 | 0.210 | 0.334 |
| False | Dense | 20 | 2 | 0.220 | 0.340 |
| False | Dense | 20 | 3 | 0.210 | 0.329 |
| False | Dense | 30 | 1 | 0.210 | 0.339 |
| False | Dense | 30 | 2 | 0.210 | 0.333 |
| False | Dense | 30 | 3 | 0.210 | 0.343 |
| False | Dense | 40 | 1 | 0.210 | 0.361 |
| False | Dense | 40 | 2 | 0.220 | 0.357 |
| False | Dense | 40 | 3 | 0.230 | 0.375 |
| False | Dense | 50 | 1 | 0.220 | 0.346 |
| False | Dense | 50 | 2 | 0.240 | 0.368 |
| False | Dense | 50 | 3 | 0.210 | 0.333 |
| False | Dense | 70 | 1 | 0.190 | 0.324 |
| False | Dense | 70 | 2 | 0.210 | 0.332 |
| False | Dense | 70 | 3 | 0.200 | 0.328 |
| False | Dense | 100 | 1 | 0.220 | 0.346 |
| False | Dense | 100 | 2 | 0.190 | 0.315 |
| False | Dense | 100 | 3 | 0.220 | 0.344 |
| False | Sparse | 5 | 1 | 0.220 | 0.326 |
| False | Sparse | 5 | 2 | 0.210 | 0.320 |
| False | Sparse | 5 | 3 | 0.230 | 0.331 |
| False | Sparse | 10 | 1 | 0.240 | 0.370 |
| False | Sparse | 10 | 2 | 0.230 | 0.347 |
| False | Sparse | 10 | 3 | 0.240 | 0.355 |

*Continued on next page*

Table 15 – *Continued from previous page*

| Q-decomp | Retrieval (Triplets) | Top-k | Run | EM | F1 |
|---|---|---|---|---|---|
| False | Sparse | 15 | 1 | 0.260 | 0.392 |
| False | Sparse | 15 | 2 | 0.250 | 0.372 |
| False | Sparse | 15 | 3 | 0.240 | 0.360 |
| False | Sparse | 20 | 1 | 0.230 | 0.353 |
| False | Sparse | 20 | 2 | 0.230 | 0.353 |
| False | Sparse | 20 | 3 | 0.230 | 0.363 |
| False | Sparse | 30 | 1 | 0.270 | 0.407 |
| False | Sparse | 30 | 2 | 0.290 | 0.427 |
| False | Sparse | 30 | 3 | 0.270 | 0.413 |
| False | Sparse | 40 | 1 | 0.240 | 0.378 |
| False | Sparse | 40 | 2 | 0.250 | 0.385 |
| False | Sparse | 40 | 3 | 0.250 | 0.379 |
| False | Sparse | 50 | 1 | 0.240 | 0.365 |
| False | Sparse | 50 | 2 | 0.250 | 0.375 |
| False | Sparse | 50 | 3 | 0.270 | 0.395 |
| False | Sparse | 70 | 1 | 0.180 | 0.272 |
| False | Sparse | 70 | 2 | 0.180 | 0.276 |
| False | Sparse | 70 | 3 | 0.180 | 0.277 |
| False | Sparse | 100 | 1 | 0.200 | 0.324 |
| False | Sparse | 100 | 2 | 0.220 | 0.345 |
| False | Sparse | 100 | 3 | 0.210 | 0.330 |
| True | Dense | 5 | 1 | 0.390 | 0.464 |
| True | Dense | 5 | 2 | 0.380 | 0.454 |
| True | Dense | 5 | 3 | 0.370 | 0.452 |
| True | Dense | 10 | 1 | 0.400 | 0.474 |
| True | Dense | 10 | 2 | 0.390 | 0.469 |
| True | Dense | 10 | 3 | 0.390 | 0.470 |
| True | Dense | 15 | 1 | 0.420 | 0.491 |
| True | Dense | 15 | 2 | 0.420 | 0.500 |
| True | Dense | 15 | 3 | 0.410 | 0.486 |
| True | Dense | 20 | 1 | 0.440 | 0.531 |
| True | Dense | 20 | 2 | 0.440 | 0.511 |
| True | Dense | 20 | 3 | 0.410 | 0.499 |
| True | Dense | 30 | 1 | 0.440 | 0.505 |
| True | Dense | 30 | 2 | 0.430 | 0.515 |
| True | Dense | 30 | 3 | 0.440 | 0.513 |
| True | Dense | 40 | 1 | 0.410 | 0.475 |
| True | Dense | 40 | 2 | 0.410 | 0.493 |
| True | Dense | 40 | 3 | 0.430 | 0.500 |
| True | Dense | 50 | 1 | 0.440 | 0.516 |
| True | Dense | 50 | 2 | 0.460 | 0.553 |
| True | Dense | 50 | 3 | 0.440 | 0.519 |
| True | Dense | 70 | 1 | 0.450 | 0.544 |

*Continued on next page*

Table 15 – *Continued from previous page*

| Q-decomp | Retrieval (Triplets) | Top-k | Run | EM | F1 |
|---|---|---|---|---|---|
| True | Dense | 70 | 2 | 0.430 | 0.528 |
| True | Dense | 70 | 3 | 0.420 | 0.518 |
| True | Dense | 100 | 1 | 0.410 | 0.508 |
| True | Dense | 100 | 2 | 0.440 | 0.522 |
| True | Dense | 100 | 3 | 0.400 | 0.499 |
| True | Sparse | 5 | 1 | 0.270 | 0.351 |
| True | Sparse | 5 | 2 | 0.290 | 0.362 |
| True | Sparse | 5 | 3 | 0.280 | 0.361 |
| True | Sparse | 10 | 1 | 0.350 | 0.452 |
| True | Sparse | 10 | 2 | 0.340 | 0.448 |
| True | Sparse | 10 | 3 | 0.360 | 0.453 |
| True | Sparse | 15 | 1 | 0.380 | 0.490 |
| True | Sparse | 15 | 2 | 0.360 | 0.455 |
| True | Sparse | 15 | 3 | 0.370 | 0.470 |
| True | Sparse | 20 | 1 | 0.400 | 0.497 |
| True | Sparse | 20 | 2 | 0.390 | 0.494 |
| True | Sparse | 20 | 3 | 0.400 | 0.515 |
| True | Sparse | 30 | 1 | 0.410 | 0.514 |
| True | Sparse | 30 | 2 | 0.410 | 0.502 |
| True | Sparse | 30 | 3 | 0.400 | 0.499 |
| True | Sparse | 40 | 1 | 0.400 | 0.500 |
| True | Sparse | 40 | 2 | 0.390 | 0.498 |
| True | Sparse | 40 | 3 | 0.410 | 0.502 |
| True | Sparse | 50 | 1 | 0.390 | 0.484 |
| True | Sparse | 50 | 2 | 0.370 | 0.460 |
| True | Sparse | 50 | 3 | 0.360 | 0.459 |
| True | Sparse | 70 | 1 | 0.350 | 0.426 |
| True | Sparse | 70 | 2 | 0.340 | 0.421 |
| True | Sparse | 70 | 3 | 0.350 | 0.425 |
| True | Sparse | 100 | 1 | 0.380 | 0.462 |
| True | Sparse | 100 | 2 | 0.360 | 0.449 |
| True | Sparse | 100 | 3 | 0.360 | 0.457 |

### B.3 Experiment results for question answering based on paragraphs

The results for all runs of experiments for multi-hop question answering with different prompting techniques using paragraphs as context, as described in Section 5.3.2, are provided in Table 16.

Table 16: All experiment results for paragraph based question answering

| Q-decomp | Retrieval (Paragraphs) | Top-k | Prompting | Run | EM | F1 |
|---|---|---|---|---|---|---|
| False | Dense | 3 | CTE | 1 | 0.270 | 0.364 |
| False | Dense | 3 | CTE | 2 | 0.260 | 0.382 |
| False | Dense | 3 | CTE | 3 | 0.250 | 0.363 |
| False | Dense | 5 | CTE | 1 | 0.320 | 0.419 |
| False | Dense | 5 | CTE | 2 | 0.310 | 0.408 |
| False | Dense | 5 | CTE | 3 | 0.300 | 0.398 |
| False | Dense | 10 | CTE | 1 | 0.370 | 0.497 |
| False | Dense | 10 | CTE | 2 | 0.330 | 0.458 |
| False | Dense | 10 | CTE | 3 | 0.340 | 0.477 |
| False | Sparse | 3 | CTE | 1 | 0.180 | 0.287 |
| False | Sparse | 3 | CTE | 2 | 0.170 | 0.281 |
| False | Sparse | 3 | CTE | 3 | 0.160 | 0.267 |
| False | Sparse | 5 | CTE | 1 | 0.270 | 0.382 |
| False | Sparse | 5 | CTE | 2 | 0.290 | 0.402 |
| False | Sparse | 5 | CTE | 3 | 0.270 | 0.368 |
| False | Sparse | 10 | CTE | 1 | 0.350 | 0.481 |
| False | Sparse | 10 | CTE | 2 | 0.340 | 0.465 |
| False | Sparse | 10 | CTE | 3 | 0.350 | 0.474 |
| False | Dense | 3 | Standard | 1 | 0.240 | 0.319 |
| False | Dense | 3 | Standard | 2 | 0.240 | 0.328 |
| False | Dense | 3 | Standard | 3 | 0.220 | 0.302 |
| False | Dense | 5 | Standard | 1 | 0.180 | 0.257 |
| False | Dense | 5 | Standard | 2 | 0.190 | 0.273 |
| False | Dense | 5 | Standard | 3 | 0.170 | 0.247 |
| False | Dense | 10 | Standard | 1 | 0.120 | 0.183 |
| False | Dense | 10 | Standard | 2 | 0.140 | 0.191 |
| False | Dense | 10 | Standard | 3 | 0.150 | 0.208 |
| False | Sparse | 3 | Standard | 1 | 0.180 | 0.274 |
| False | Sparse | 3 | Standard | 2 | 0.200 | 0.298 |
| False | Sparse | 3 | Standard | 3 | 0.190 | 0.278 |
| False | Sparse | 5 | Standard | 1 | 0.160 | 0.266 |
| False | Sparse | 5 | Standard | 2 | 0.160 | 0.257 |
| False | Sparse | 5 | Standard | 3 | 0.150 | 0.253 |
| False | Sparse | 10 | Standard | 1 | 0.080 | 0.138 |
| False | Sparse | 10 | Standard | 2 | 0.140 | 0.199 |
| False | Sparse | 10 | Standard | 3 | 0.150 | 0.219 |
| True | Dense | 3 | CTE | 1 | 0.550 | 0.660 |
| True | Dense | 3 | CTE | 2 | 0.530 | 0.641 |
| True | Dense | 3 | CTE | 3 | 0.560 | 0.661 |
| True | Dense | 5 | CTE | 1 | 0.490 | 0.589 |
| True | Dense | 5 | CTE | 2 | 0.500 | 0.606 |
| True | Dense | 5 | CTE | 3 | 0.520 | 0.620 |

*Continued on next page*

Table 16 – *Continued from previous page*

| Q-decomp | Retrieval (Paragraphs) | Top-k | Prompting | Run | EM | F1 |
|---|---|---|---|---|---|---|
| True | Dense | 10 | CTE | 1 | 0.570 | 0.671 |
| True | Dense | 10 | CTE | 2 | 0.520 | 0.628 |
| True | Dense | 10 | CTE | 3 | 0.520 | 0.634 |
| True | Sparse | 3 | CTE | 1 | 0.520 | 0.628 |
| True | Sparse | 3 | CTE | 2 | 0.520 | 0.607 |
| True | Sparse | 3 | CTE | 3 | 0.560 | 0.647 |
| True | Sparse | 5 | CTE | 1 | 0.500 | 0.616 |
| True | Sparse | 5 | CTE | 2 | 0.490 | 0.599 |
| True | Sparse | 5 | CTE | 3 | 0.490 | 0.610 |
| True | Sparse | 10 | CTE | 1 | 0.510 | 0.599 |
| True | Sparse | 10 | CTE | 2 | 0.480 | 0.581 |
| True | Sparse | 10 | CTE | 3 | 0.510 | 0.600 |
| True | Dense | 3 | Standard | 1 | 0.490 | 0.574 |
| True | Dense | 3 | Standard | 2 | 0.480 | 0.564 |
| True | Dense | 3 | Standard | 3 | 0.480 | 0.578 |
| True | Dense | 5 | Standard | 1 | 0.430 | 0.504 |
| True | Dense | 5 | Standard | 2 | 0.420 | 0.498 |
| True | Dense | 5 | Standard | 3 | 0.410 | 0.480 |
| True | Dense | 10 | Standard | 1 | 0.370 | 0.442 |
| True | Dense | 10 | Standard | 2 | 0.400 | 0.472 |
| True | Dense | 10 | Standard | 3 | 0.340 | 0.429 |
| True | Sparse | 3 | Standard | 1 | 0.410 | 0.482 |
| True | Sparse | 3 | Standard | 2 | 0.430 | 0.502 |
| True | Sparse | 3 | Standard | 3 | 0.460 | 0.517 |
| True | Sparse | 5 | Standard | 1 | 0.360 | 0.426 |
| True | Sparse | 5 | Standard | 2 | 0.390 | 0.451 |
| True | Sparse | 5 | Standard | 3 | 0.400 | 0.467 |
| True | Sparse | 10 | Standard | 1 | 0.350 | 0.415 |
| True | Sparse | 10 | Standard | 2 | 0.370 | 0.436 |
| True | Sparse | 10 | Standard | 3 | 0.360 | 0.419 |

### B.4 Experiment results for impact of prompting on question answering

The results for all runs of experiments for evaluating the impact of prompting technique on multi-hop question answering described in Section 5.3.3.1 are provided in Table 17.

### B.5 Experiment results for impact of temperature on Question Answering

The results for all runs of experiments for evaluating the impact of temperature on multi-hop question answering described in Section 5.3.3.2 are provided in Table 18.

Table 17: All experiment results for question answering with different prompting techniques

| Prompting | Run | EM | F1 |
|---|---|---|---|
| Standard | 1 | 0.540 | 0.651 |
| Standard | 2 | 0.530 | 0.627 |
| Standard | 3 | 0.520 | 0.631 |
| Zero-shot CoT | 1 | 0.520 | 0.654 |
| Zero-shot CoT | 2 | 0.540 | 0.669 |
| Zero-shot CoT | 3 | 0.520 | 0.637 |
| Few-shot CoT | 1 | 0.560 | 0.664 |
| Few-shot CoT | 2 | 0.520 | 0.608 |
| Few-shot CoT | 3 | 0.510 | 0.606 |
| CTE | 1 | 0.570 | 0.700 |
| CTE | 2 | 0.590 | 0.718 |
| CTE | 3 | 0.560 | 0.692 |

Table 18: All experiment results for question answering with different temperature values

| Prompting Technique | Temperature | Run | EM | F1 |
|---|---|---|---|---|
| Standard | 0.0 | 1 | 0.540 | 0.645 |
| Standard | 0.0 | 2 | 0.530 | 0.639 |
| Standard | 0.0 | 3 | 0.520 | 0.625 |
| Standard | 0.1 | 1 | 0.540 | 0.646 |
| Standard | 0.1 | 2 | 0.540 | 0.659 |
| Standard | 0.1 | 3 | 0.530 | 0.643 |
| Standard | 0.3 | 1 | 0.480 | 0.596 |
| Standard | 0.3 | 2 | 0.520 | 0.633 |
| Standard | 0.3 | 3 | 0.510 | 0.624 |
| Standard | 0.5 | 1 | 0.480 | 0.597 |
| Standard | 0.5 | 2 | 0.510 | 0.629 |
| Standard | 0.5 | 3 | 0.500 | 0.620 |
| Standard | 0.7 | 1 | 0.530 | 0.634 |
| Standard | 0.7 | 2 | 0.480 | 0.595 |
| Standard | 0.7 | 3 | 0.490 | 0.613 |
| Standard | 1.0 | 1 | 0.490 | 0.620 |
| Standard | 1.0 | 2 | 0.450 | 0.557 |
| Standard | 1.0 | 3 | 0.490 | 0.599 |
| Standard | 1.5 | 1 | 0.480 | 0.583 |
| Standard | 1.5 | 2 | 0.480 | 0.587 |
| Standard | 1.5 | 3 | 0.480 | 0.577 |
| Standard | 2.0 | 1 | 0.340 | 0.402 |
| Standard | 2.0 | 2 | 0.370 | 0.470 |
| Standard | 2.0 | 3 | 0.380 | 0.472 |

*Continued on next page*

Table 18 – *Continued from previous page*

| Prompting Technique | Temperature | Run | EM | F1 |
|---|---|---|---|---|
| CTE | 0.0 | 1 | 0.560 | 0.691 |
| CTE | 0.0 | 2 | 0.560 | 0.695 |
| CTE | 0.0 | 3 | 0.570 | 0.712 |
| CTE | 0.1 | 1 | 0.540 | 0.686 |
| CTE | 0.1 | 2 | 0.570 | 0.698 |
| CTE | 0.1 | 3 | 0.580 | 0.712 |
| CTE | 0.3 | 1 | 0.560 | 0.685 |
| CTE | 0.3 | 2 | 0.550 | 0.676 |
| CTE | 0.3 | 3 | 0.580 | 0.701 |
| CTE | 0.5 | 1 | 0.560 | 0.684 |
| CTE | 0.5 | 2 | 0.580 | 0.691 |
| CTE | 0.5 | 3 | 0.550 | 0.669 |
| CTE | 0.7 | 1 | 0.570 | 0.703 |
| CTE | 0.7 | 2 | 0.580 | 0.704 |
| CTE | 0.7 | 3 | 0.580 | 0.684 |
| CTE | 1.0 | 1 | 0.530 | 0.646 |
| CTE | 1.0 | 2 | 0.520 | 0.666 |
| CTE | 1.0 | 3 | 0.580 | 0.690 |
| CTE | 1.5 | 1 | 0.560 | 0.654 |
| CTE | 1.5 | 2 | 0.560 | 0.684 |
| CTE | 1.5 | 3 | 0.540 | 0.692 |
| CTE | 2.0 | 1 | 0.200 | 0.226 |
| CTE | 2.0 | 2 | 0.100 | 0.150 |
| CTE | 2.0 | 3 | 0.140 | 0.188 |

### B.6 Experiment results for impact of LLM adaptation techniques for JERX

The results for all runs of experiments for evaluating the impact of LLM adaptation techniques for JERX task on question answering described in Section 5.3.3.3 are provided in Table 19.

Table 19: All experiment results for question answering with different JERX models

| Model | Run | EM | F1 |
|---|---|---|---|
| Zero-shot Prompted | 1 | 0.470 | 0.564 |
| Zero-shot Prompted | 2 | 0.540 | 0.625 |
| Zero-shot Prompted | 3 | 0.490 | 0.614 |
| Few-shot Prompted | 1 | 0.560 | 0.655 |
| Few-shot Prompted | 2 | 0.560 | 0.663 |
| Few-shot Prompted | 3 | 0.570 | 0.666 |
| Supervised Fine-Tuned | 1 | 0.550 | 0.646 |
| Supervised Fine-Tuned | 2 | 0.550 | 0.638 |
| Supervised Fine-Tuned | 3 | 0.570 | 0.654 |