Improve Pretrained Large Language Model's Faithfulness with Knowledge Graph

Xinyu Bao
Electrical Engineering and Computer Science
University of Michigan
`xinyubao@umich.edu`

Haohong Shang
Electrical Engineering and Computer Science
University of Michigan
`haohong@umich.edu`

Ziyang Xiong
Electrical Engineering and Computer Science
University of Michigan
`xziyang@umich.edu`

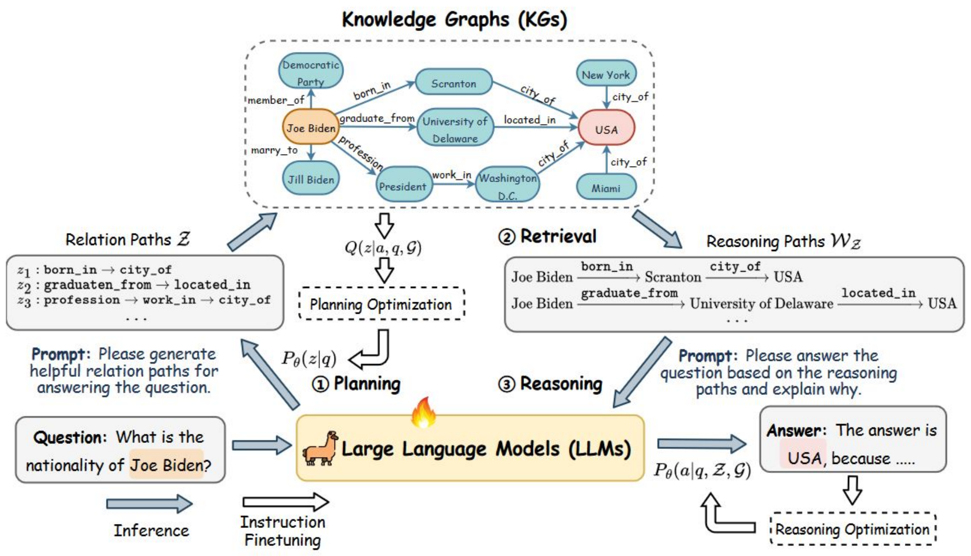
Abstract

Knowledge graph (KG) has shown great power in assisting large language models (LLMs) for knowledge retrieval and reasoning tasks. Recent works in this field, especially the "Retrieval-Augmented Generation" (RAG), "Reasoning on Graphs" (RoG) model, and the "Think-on-Graph" (ToG) model, significantly improve the performance in KG-based Q&A (KGQA) tasks. In this project, we implement the path-planning algorithm of the RoG model, demonstrating its performance in KGQA tasks using LLaMA2 as the backbone LLM, and further improve it by combining with the idea in RAG and ToG models. The model achieves 28.4% in accuracy and 42.2% in Hit10 evaluated on the benchmark KGQA dataset WebQuestionSP.

1 Introduction

1.1 Motivation

Large Language Models (LLMs), such as GPT4 [Achiam et al., 2023] and LLaMA2 [Touvron et al., 2023], have shown their ability to solve NLP tasks [Brown et al., 2020]. However, for the knowledge they do not know, the models generate unreliable answers [Liu et al., 2024]. To solve this problem, Lewis et al. [2021] proposed a method called "Retrieval-Augmented Generation" (RAG) that integrates retrieval of relevant information with the generation of content, enhancing natural language processing systems by allowing them to access and incorporate external knowledge during the generation process. Nevertheless, RAG cannot retrieve knowledge graphs (KG) properly when the query is too complex. Luo et al. [2023] introduces a method called "Reasoning on Graphs" (RoG) that solves the above problem by planning potential relation paths in advance before retrieving the KG to break down complex queries, reaching the state-of-the-art performance on KG reasoning tasks under two benchmark KGQA datasets. However, RoG is inefficient in generating relationships from KG without finetuning. Inspired by the "Think-on-Graph" (ToG) model Sun et al. [2023] that also achieved similar performance using the idea of pruning, we plan to improve RoG performance by involving pruning algorithms to fasten the selection and generation of relation path and make our model more accurate.



Knowledge Graphs (KGs)
Democratic Party
member_of
Joe Biden
born_in
Scranton
city_of
New York
city_of
USA
graduate_from
University of Delaware
located_in
marry_to
Jill Biden
profession
President
work_in
Washington D.C.
city_of
Miami
city_of
Relation Paths $\mathcal{Z}$
z_1 : born_in → city_of
z_2 : graduaten_from → located_in
z_3 : profession → work_in → city_of
...
$Q(z|a,q,\mathcal{G})$
Planning Optimization
$P_\theta(z|q)$
① Planning
② Retrieval
Reasoning Paths $\mathcal{W}_\mathcal{Z}$
Joe Biden $\xrightarrow{\text{born\_in}}$ Scranton $\xrightarrow{\text{city\_of}}$ USA
Joe Biden $\xrightarrow{\text{graduate\_from}}$ University of Delaware $\xrightarrow{\text{located\_in}}$ USA
...
③ Reasoning
Prompt: Please generate helpful relation paths for answering the question.
Prompt: Please answer the question based on the reasoning paths and explain why.
Question: What is the nationality of Joe Biden?
Large Language Models (LLMs)
Answer: The answer is USA, because .....
$P_\theta(a|q,\mathcal{Z},\mathcal{G})$
Reasoning Optimization
Inference
Instruction Finetuning

Figure 1: The overall workflow of RoG by [Luo et al., 2023].

1.2 Related works

1.2.1 LLM Reasoning Prompt

Many methods have been proposed to improve the reasoning ability of LLMs and handle complex tasks through prompting [Wang et al., 2022]. He et al. [2021] prompts LLMs to decompose the reasoning task into a series of sub-tasks and solve them step by step. Plan-and-solve [Wang et al., 2023] prompts LLMs to generate a plan and conduct reasoning based on it. However, the problem of hallucinations and lack of knowledge affect the faithfulness of LLMs' reasoning. The ReACT [Yao et al., 2023] treats LLMs as agents, which interact with the environment to get the latest knowledge for reasoning. To explore faithful reasoning, FAME [Hong et al., 2023] introduces the Monte-Carlo planning to generate faithful reasoning steps. RR [He et al., 2022] further retrieve relevant knowledge from KGs to produce faithful reasoning plans for LLMs.

1.2.2 Incorporating external knowledge into LLMs

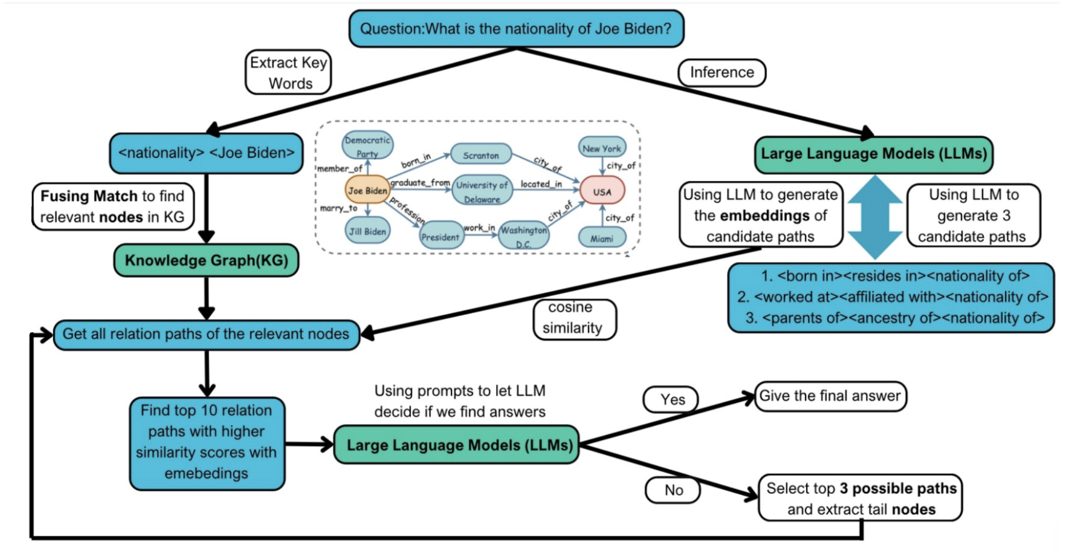
Significant effort has been devoted to leveraging external knowledge to improve the reasoning ability of LLMs. Previous work has incorporated external knowledge sources such as ConceptNet Speer et al. [2018] to enhance LMs for tabular reasoning tasks. Explicit rules have also been added to inputs to improve reasoning ability over implicit knowledge Talmor et al. [2020]. In addition, explicit knowledge from Wikidata Vrandečić and Krötzsch [2014] and implicit knowledge in LLMs have been integrated into a Transformer for visual question answering Gui et al. [2022]. Nye et al. [2021] instead introduces a symbolic reasoning module to improve coherence and consistency in LLMs.

2 Methods

2.1 Original Method

Figure 1 shows the entire workflow of the RoG model. When the user gives out a text question to the LLM, it will first plan to generate a prompt to get helpful relation paths Z for answering the question. Then, for each relation path $z_i \in Z$, it will be converted to a KG query to retrieve the corresponding knowledge from KG, and its output will be a reasoning path W_{z_i}. Given all the retrieved reasoning paths W_z and the user's input question, LLMs will reason from all the necessary information and generate the answer to the question.

This paper contributes optimizations mainly in 3 stages: planning, retrieval, and reasoning. It also tries several different LLMs, including Alpaca [Taori et al., 2023], ChatGPT [Achiam et al., 2023],



Question:What is the nationality of Joe Biden?
Extract Key Words
Inference
<nationality> <Joe Biden>
Large Language Models (LLMs)
Fusing Match to find relevant nodes in KG
Using LLM to generate the embeddings of candidate paths
Using LLM to generate 3 candidate paths
Knowledge Graph(KG)
1. <born in><resides in><nationality of>
2. <worked at><affiliated with><nationality of>
3. <parents of><ancestry of><nationality of>
cosine similarity
Get all relation paths of the relevant nodes
Find top 10 relation paths with higher similarity scores with emebedings
Using prompts to let LLM decide if we find answers
Large Language Models (LLMs)
Yes
Give the final answer
No
Select top 3 possible paths and extract tail nodes
Democratic Party
Joe Biden
Jill Biden
Scranton
University of Delaware
President
Washington D.C.
New York
USA
Miami
member_of
born_in
graduate_from
marry_to
profession
work_in
city_of
located_in

Figure 2: The overall workflow of our model.

Flan-T5 [Chung et al., 2022], and LLaMA2 [Touvron et al., 2023]. In this project, we mainly focus on optimizing the planning stage with LLaMA2 as the core LLM considering the low cost of local deployment for faster testing.

2.2 Our Method

Figure 2 shows the entire workflow of our model, which could be structured into four distinct phases:

Step 1: Path Generation and Embedding by LLM. Initially, the LLM is utilized to freely generate candidate relation paths given only the questions. Then, we select the most promising relation path, concatenate it with the query, and compute embeddings to facilitate subsequent similarity assessments. Our experiment shows that concatenating promising relation paths with the original question gives out better embeddings for similarity-based filtering in Step 3.

Step 2: Keyword Extraction and Node Identification with KG. Initially, key terms (such as verbs, nouns, and adjectives) are extracted from the input query, discarding non-essential words like adverbs and auxiliary verbs. A fuzzy matching technique is then employed to identify corresponding nodes within the KG. Once the key node is identified, the model retrieves and extracts all relevant relation paths emanating from this node in the KG as promising starting paths to answer the question.

Step 3: Similarity Calculation and Path Selection. The cosine similarity between the embeddings from Step 1 and the relation paths identified in Step 2 is computed after they are both encoded by LLaMa2. The top ten paths with the highest similarity scores are selected for further processing.

Step 4: Answer Determination by LLM. The LLM is used to determine if a satisfactory answer can be derived from the existing paths. If no answer is found, the model selects the three most promising paths, retrieves their tail nodes, and prompts them back into Step 2 to refine the search. If an answer is obtained or if the process has cycled through three iterations, the model outputs the answer.

All the interactions with the LLMs are done according to our optimized prompts with few-shot learning. Detailed implementation can be found in our submitted code files.

The innovation of our path optimization lies in Step 3, where we compute the cosine similarity between the embeddings obtained from the initial paths and those identified as relevant in the KG. Rather than searching out the question-related information through vectorized indexes as is in RAG, fuzzy matching can pinpoint the key node entity more accurately, giving a good starting point for finding the full relation paths. Further, instead of processing a cumbersome set of paths, out pruning strategy by selecting only the top ten paths based on their similarity scores not only streamlines the process but also significantly enhances the efficiency and accuracy of the model. By focusing on the

Method	**hit@1**	**hit@10**
Original LLama	-	-
LLama with our method	28.4%	42.2%

Table 1: Model Performance Metrics

most promising paths, we ensure that the computational resources are allocated to processing the most potential leads, thereby optimizing the path selection process.

Besides, in Step 4, we have a dynamic answer-determination setting. If the initial attempts do not yield an answer, the model dynamically adjusts by selecting the three most promising paths from the pruned list and recycles these back into Step 2 for further refinement. This iterative process, combined with the strategic pruning in Step 3, enhances the model's ability to converge on the correct answer efficiently, showcasing a significant optimization over traditional static query answering systems.

2.3 Extension and Benifits

Although motivated by the RAG and ROG methods, we found that they have many limitations in efficiency and scalability, both of which are successfully optimized by our method. First of all, while the use of KGs to fine-tune LLMs for Knowledge Graph Question Answering (KGQA) can enhance performance, this approach is typically limited to specific datasets, losing much flexibility in switching between different knowledge fields. Moreover, fine-tuning LLMs can be prohibitively costly and demand extensive computing resources.

In contrast, our methodology has some significant advances upon the foundation set by ROG, offering distinct advantages in terms of scalability, adaptability, and efficiency:

Enhanced Scalability and Adaptability Unlike the ROG model which is primarily finetuned on fixed datasets, our approach exhibits unparalleled flexibility in adapting to diverse and evolving datasets without the need for retraining. This adaptability extends to various domains and applications, ensuring that our model remains effective even as the scope of the knowledge base expands.

Reduced Computational Overhead By eliminating the need for continuous fine-tuning typically associated with ROG models, our method substantially lowers computational costs and expedites the retrieval process.

Enhanced Query Handling and Knowledge Exploration Our method combines the structured organization of KGs with the adaptive learning potential of LLMs, enabling it to effectively handle new and complex queries. This integration not only addresses the prevalent knowledge gaps found in LLMs but also enhances their utility for exploring and developing new domains in LLM applications. The approach proves particularly powerful for advancing research and practical implementations across various fields of study.

3 Evaluation

3.1 Datasets

For the KGQA task, WebQuestionSP (WebQSP) [tau Yih et al., 2016] and Complex WebQuestion (CWQ) [Talmor and Berant, 2018] are two commonly used benchmark datasets which are used in many related papers [Luo et al., 2023, Sun et al., 2023], and Freebase [Bollacker et al., 2008] is the background knowledge graph for both datasets. As WebQSP is small in size and easy to be tested on, in this project, we plan to use WebQSP for our model evaluation.

3.2 Metric

In line with prior research papers [Luo et al., 2023, Sun et al., 2023], we use Hit@1 and Hit@10 as our assessment criteria. Hits@1 quantifies the percentage of questions with accurately predicted top-1 answers, while Hit@10 focuses on top-10 answers. And we show the result in Table 1.

4 Discussion

Inspired by ROG and RAG, our method overlooks the limitations of datasets and computational resources by integrating LLMs and KGs. This integration allows the language model to answer unprecedented questions by leveraging knowledge from external sources. Throughout this development, we encountered lots of challenges and struggled on lots of details.

For instance, initially, we utilized only the embeddings of candidate relation paths generated by the LLM to compute cosine similarity. However, this approach did not effectively prune the relation paths. After extensive experimentation, including the use of a greater number of candidate paths' embeddings, we ultimately refined our method. We decided to concatenate each path with the query prior to generating embeddings, which significantly enhanced the effectiveness of pruning.

Besides, as highlighted in the related work, the role of prompts is crucial in training LLMs to adhere better to our instructions and to format the responses as desired. We have optimized prompts used in step 1 and step 4 lots of times in order to get better results. We finally confirmed that our prompt uses one-shot in Alpaca format (Instruction-Input-Response). It's a pity we haven't designed a reasonable chain-of-thought or few-shot prompts, but we believe that higher accuracy in Hit@1 and Hit@10 can be achieved with better prompts. Because the accuracy of LLM in determining whether the answer is obtained in step 4 is not so satisfactory.

Overall, the most significant challenge we encountered in our experiment lies in step 4. Because we observed that in most cases, we could get reasonable relation paths (at least part of the relation path) after pruning(step 3), indicating the potential for high accuracy. Unfortunately, the language model frequently terminated the process prematurely by affirming a solution before we obtained the complete answer. Moreover, it struggled to consistently identify the most suitable nodes for the subsequent step, leading to limited accuracy. We believe there are numerous opportunities for improvement at this stage. For instance, we could explore feeding answers after step 3 to multiple language models simultaneously and only conclude the process if they all confirm the solution. By implementing such strategies, we anticipate overcoming these challenges and enhancing the overall accuracy of our approach.

5 Contribution

Xinyu Bao: Workflow design, prompt engineering, model debugging, and testing.

Haohong Shang: Workflow design, model implementation, and prompt engineering.

Ziyang Xiong: Workflow design, literature review, and poster design.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Liangke Gui, Borui Wang, Qiuyuan Huang, Alex Hauptmann, Yonatan Bisk, and Jianfeng Gao. Kat: A knowledge augmented transformer for vision-and-language, 2022.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 553–561, 2021.

Hangfeng He, Hongming Zhang, and Dan Roth. Rethinking with retrieval: Faithful large language model inference, 2022.

Ruixin Hong, Hongming Zhang, Hong Zhao, Dong Yu, and Changshui Zhang. Faithful question answering with monte-carlo planning, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36, 2024.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*, 2023.

Maxwell Nye, Michael Henry Tessler, Joshua B. Tenenbaum, and Brenden M. Lake. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning, 2021.

Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge, 2018.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph, 2023.

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

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge, 2020.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

Wen tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. The value of semantic parse labeling for knowledge base question answering. In *Annual Meeting of the Association for Computational Linguistics*, 2016. URL `https://api.semanticscholar.org/CorpusID:13905064`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, sep 2014. ISSN 0001-0782. doi: 10.1145/2629489. URL `https://doi.org/10.1145/2629489`.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023.