arXiv:2308.09729v4 [cs.AI] 15 Sep 2023

MindMap: Knowledge Graph Prompting Sparks Graph of Thoughts in Large Language Models

Yilin Wen* **Zifeng Wang[1]*** **Jimeng Sun[1]**

[1] University of Illinois Urbana-Champaign, Champaign, IL
yilinwen510@gmail.com, zifengw2@illinois.edu, jimeng@illinois.edu

Abstract

LLMs usually exhibit limitations in their ability to incorporate new knowledge, the generation of hallucinations, and the transparency of their decision-making process. In this paper, we explore how to prompt LLMs with knowledge graphs (KG), working as a remedy to engage LLMs with up-to-date knowledge and elicit the reasoning pathways from LLMs. Specifically, we build a prompting pipeline that endows LLMs with the capability of comprehending KG inputs and inferring with a combined implicit knowledge and the retrieved external knowledge. In addition, we investigate eliciting the *mind map* on which LLMs perform the reasoning and generate the answers. It is identified that the produced mind map exhibits the reasoning pathways of LLMs grounded on the ontology of knowledge, hence bringing the prospects of probing and gauging LLM inference in production. The experiments on three question & answering datasets also show that `MindMap` prompting leads to a striking empirical gain. For instance, prompting a GPT-3.5 with `MindMap` yields an overwhelming performance over GPT-4 consistently. We also demonstrate that with structured facts retrieved from KG, `MindMap` can outperform a series of prompting-with-document-retrieval methods, benefiting from more accurate, concise, and comprehensive knowledge from KGs. For reproducing our results and extending the framework further, we make our codebase available at https://github.com/wyl-willing/MindMap.

1 Introduction

Scaling large language models (LLMs) to billions of parameters and a training corpus of trillion words was proved to induce surprising performance in various tasks (Brown et al. 2020; Chowdhery et al. 2022). Pre-trained LLMs can be adapted to domain tasks with further fine-tuning (Singhal et al. 2023) or be aligned with human preferences with instruction-tuning (Ouyang et al. 2022). Nonetheless, several hurdles lie in the front of steering LLMs in production:

- **Inflexibility**. The pre-trained LLMs possess outdated knowledge and are inflexible to parameter updating. Fine-tuning LLMs can be tricky because either collecting high-quality instruction data and building the training pipeline can be costly (Cao, Kang, and Sun 2023), or continually fine-tuning LLMs renders a risk of catastrophic forgetting (Razdaibiedina et al. 2022).
- **Hallucination**. LLMs are notoriously known to produce hallucinations with plausible-sounding but wrong outputs (Ji et al. 2023), which causes serious concerns for high-stake applications such as medical diagnosis.
- **Transparency**. LLMs are also criticized for their lack of transparency due to the black-box nature (Danilevsky et al. 2020). The knowledge is implicitly stored in LLM's parameters, thus infeasible to be validated. Also, the inference process in deep neural networks remains elusive to be interpretable.

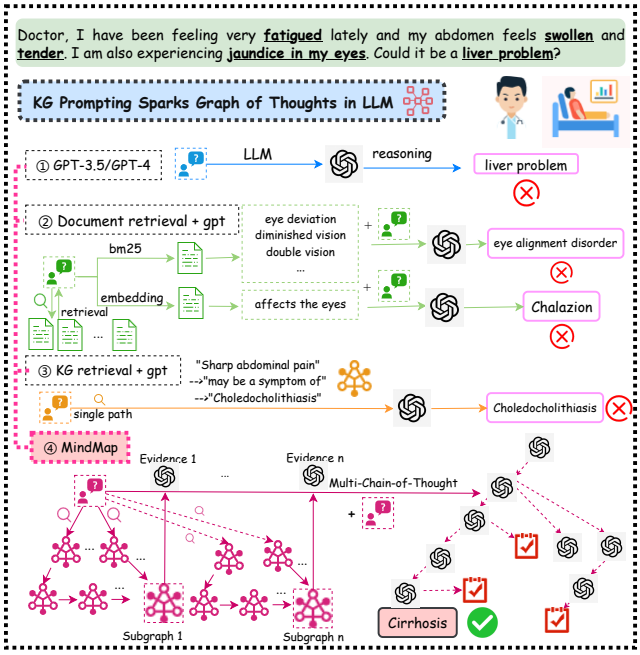

Figure 1: A conceptual comparison between our method and the other prompting baselines: LLM-only, document retrieval + LLM, and KG retrieval + LLM.

As a classic way to build large-scale structural knowledge bases, knowledge graphs (KG) are established by the triples of entities and relations, i.e., $\{\texttt{head}, \texttt{relation}, \texttt{tail}\}$. They can provide explicit knowledge representation and in-

*These authors contributed equally.

Preprint.

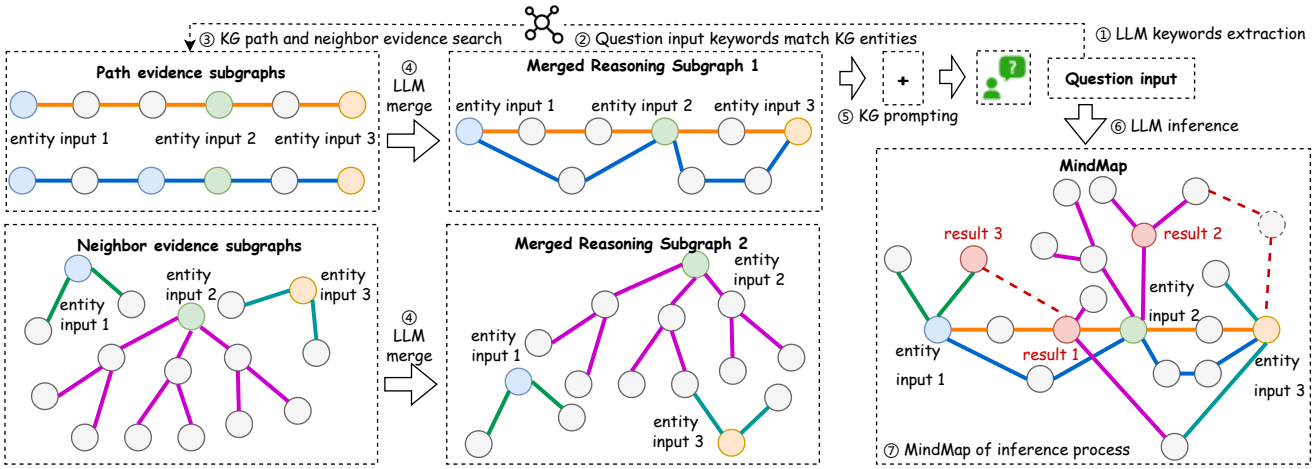

Figure 2: A conceptual demonstration of evidence query sub-graphs, merged reasoning sub-graphs, and mind map. The entity inputs $\mathcal{V}_q$ is identified from the input. Lines and circles of the same color indicate that they correspond. The red dashed lines in the MindMap box illustrate the augmentation operation based on the knowledge of LLM.

terpretable reasoning paths. Besides, KGs are amenable to continual modifications to debug the existing knowledge or add new knowledge. Due to their flexibility, preciseness, and interpretability, KGs emerged as a promising complement to the drawbacks of LLMs (Pan et al. 2023). For instance, KG triples were added to the training of LLMs (Zhang et al. 2019b) or KG encoders were entangled with LLM layers for joint inference and optimization on graph and text data (Zhang et al. 2022). By contrast, our work pivots on the synergistic inference of KGs and fixed LLMs, which is applicable to powerful pre-trained LLMs, such as commercial LLM-as-service APIs. In general, the prior arts in this venue can be categorized into two genres:

- **Retrieval-Augmented LLM Inference**. Researchers tried to retrieve documents to augment LLM inference (Lewis et al. 2020) while suffering from inaccurate retrieval and lengthy documents (Liu et al. 2023a). Recently, several attempts were made to incorporate extracted KG triples into the prompt to LLMs to answer KG-related questions (Baek, Aji, and Saffari 2023). However, this approach treats KG inputs as plain text and ignores their graphical structure, which causes the generated response to be hard to validate and vulnerable to hallucinations.
- **Graph Mining with LLMs**. There were also attempts to prompt LLMs to comprehend graphical inputs, while they primarily experimented with graph mining tasks, e.g., edge detection and graph summarization (Guo, Du, and Liu 2023; Chen et al. 2023). It was rarely explored in text generation tasks that require complex reasoning across multiple evidence graphs grounded on KGs.

The goal of this work is to build a plug-and-play prompting approach to elicit the graph-of-thoughts reasoning capability in LLMs. We call our method `MindMap` because it enables LLMs to comprehend graphical inputs to build their own mind map that supports evidence-grounded generation. A conceptual demonstration of our framework is in Figure 2. Specifically, `MindMap` sparks the graph of thoughts of LLMs that (1) consolidates the retrieved facts from KGs and the implicit knowledge from LLMs, (2) discovers new patterns in input KGs, and (3) reasons over the mind map to yield final outputs. We conducted experiments on three datasets to illustrate that `MindMap` outperforms a series of prompting approaches by a large margin. This work underscores how LLM can learn to conduct synergistic inference with KG, combining the implicit and explicit knowledge to enable transparent and reliable inference.

2 Related Work

Prompt Engineering. The "pre-train, prompt, and predict" paradigm has become the best practice for natural language processing in few-shot or zero-shot manners (Liu et al. 2023b). The core insight is LLMs are able to adapt to new tasks following the input context and instructions via in-context learning (Brown et al. 2020), especially with instruction tuning (Wei et al. 2022) and alignment (Ouyang et al. 2022). Retrieval-augmented generation emerged as a way to dynamically inject additional evidence for LLM inference (Lewis et al. 2020). The common practice is to query a dense embedding database to find the relevant document pieces to the input user questions, then put the retrieved corpus back to the prompt input. However, documents can be lengthy, thus not fitting into the context length limit of LLM. It was also identified even though we can build long documents as prompts, LLMs usually fail to capture information in the middle of the prompt and produce hallucinations (Liu et al. 2023a). Another line of research looks to prompt to elicit the intermediate reasoning steps of LLMs in chains (Wei et al. 2023) and trees (Yao et al. 2023), while these approaches all focus on eliciting the implicit knowledge from LLMs. Nonetheless, our work explores sparking the reasoning of LLMs on graph inputs, with an emphasis on joint reasoning with implicit and external explicit knowledge.

Knowledge Graph Augmented LLM. Researchers have explored using knowledge graphs (KGs) to enhance LLMs in two main directions: (1) integrating KGs into LLM pre-

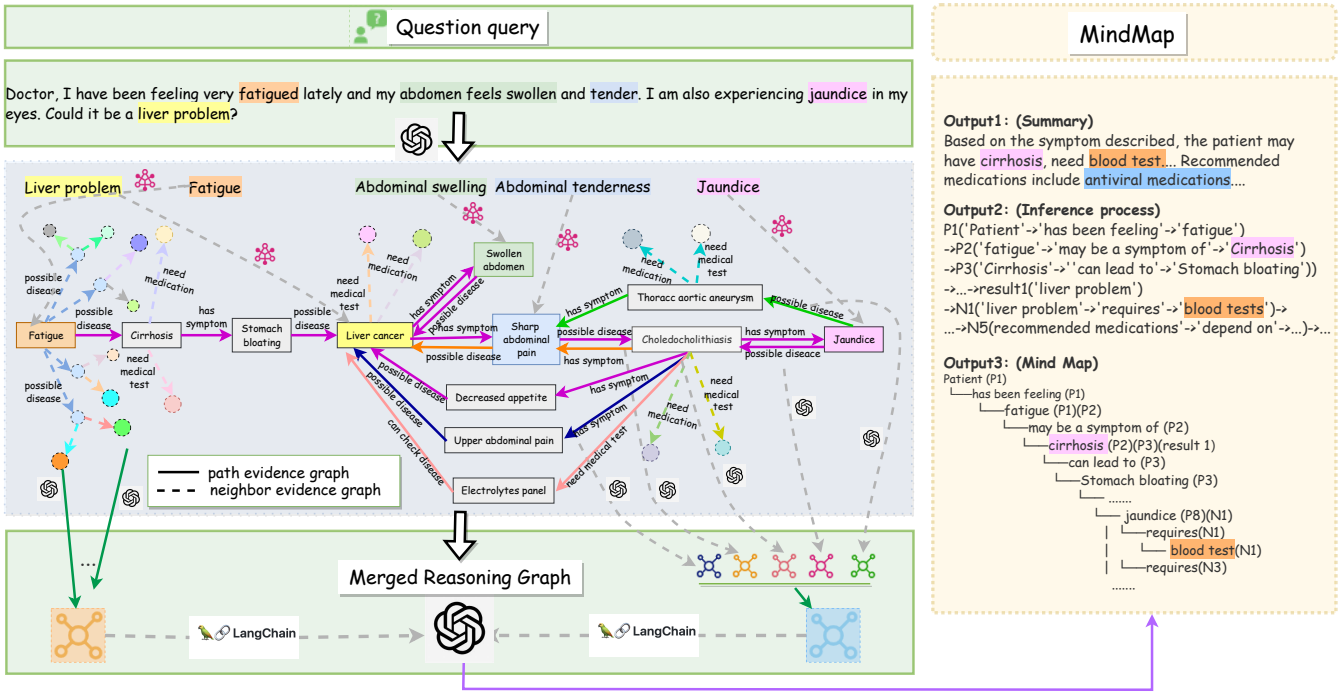

Figure 3: An overview of the architecture of our proposed `MindMap`. The left part illustrates the components of *evidence graph mining* and *evidence graph aggregation*, while the right part shows how LLM consolidates the knowledge from LLM and KG and builds its own *mind map*.

training and (2) injecting KGs into LLM inference. For (1), it is a common practice to design knowledge-aware training objectives by either putting KG entities and relations into the training data (Zhang et al. 2019b; Sun et al. 2021) or applying KG prediction tasks, e.g., link prediction, as additional supervision (Yasunaga et al. 2022). However, when scaling the pre-training data to a web-scale corpus with trillion words, it is intractable to find or create KGs with approximate scale. More importantly, although these methods directly compress KG knowledge into LLM's parameters via supervision, they do not mitigate the fundamental limits of LLMs in flexibility, reliability, and transparency.

For (2), the early efforts were centered around fusing KG triples into the inputs of LLMs via attention (Liu et al. 2020; Sun et al. 2020) or attaching graph encoders to LLM encoders to process KG inputs (Wang et al. 2019). The follow-ups further adopted graph neural networks in parallel to LLMs for joint reasoning (Yasunaga et al. 2021) and added interactions between text tokens and KG entities in the intermediate layers of LLMs (Zhang et al. 2022; Yao, Li, and Zhao 2023). Witnessing the recent success of pre-trained LLMs, the research paradigm is shifting to prompting fixed pre-trained LLMs with graphical inputs. This line of research includes prompting LLMs for KG entity linking prediction (Choudhary and Reddy 2023; Sun et al. 2023), graph mining (Guo, Du, and Liu 2023), and KG question answering (Baek, Aji, and Saffari 2023). While these approaches permit LLMs to comprehend graph inputs, they either target KG tasks exclusively or recall retrieved facts and translate them to plain texts, ignoring the structure of KG.

3 Method

We show the framework of `MindMap` in Figure 3, which comprises three main components:

1. **Evidence graph mining**: We begin by identifying the set of entities $\mathcal{V}_q$ from the raw input and query the source KG $\mathcal{G}$ to build multiple *evidence sub-graphs* $\mathcal{G}_q$.
2. **Evidence graph aggregation**: Next, LLMs are prompted to comprehend and aggregate the retrieved evidence sub-graphs to build the *reasoning graphs* $\mathcal{G}_m$.
3. **LLM reasoning on the mind map**: Last, we prompt LLMs to consolidate the built reasoning graph and their implicit knowledge to generate the answer and build a *mind map* explaining the reasoning process.

3.1 Step I: Evidence Graph Mining

Discovering the relevant evidence sub-graphs $\mathcal{G}_q$ from the external KG breaks down into two main stages.

Entity Recognition We start by prompting LLMs to extract the key entities from the question query Q via in-context learning. We build a set of exemplars $\{Q_k, M_k\}_k^K$ where each consists of a pair of input sentences Q_k and the corresponding entity set M_k. We prepend the exemplars to the candidate query Q as the context and ask LLMs to generate the entity set M. We further perform *entity linking* since the extracted entities M may not all belong to the entity set in $\mathcal{G}$. Specifically, we encode all the entities in $\mathcal{G}$ and the in M with a BERT encoder into dense embeddings $H_{\mathcal{G}}$ and H_M, respectively. We link each entity in M to its nearest

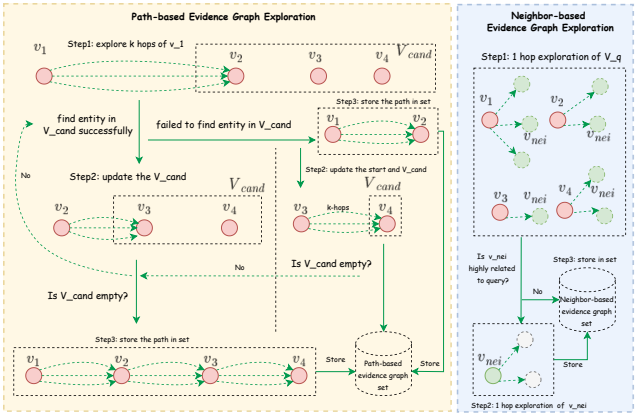

Figure 4: The process of two evidence sub-graphs exploration methods.

neighbor entity in $\mathcal{G}$ by comparing their cosine similarity, which yields an initial entity set $\mathcal{V}_q$ for building the evidence sub-graphs in the next step.

Evidence sub-graphs Exploration We define the source knowledge graph by $\mathcal{G} = \{\langle u, r, o\rangle \,|u \in \psi, r \in \varphi, o \in \mathcal{L}\}$, where ψ_q, φ_q, and $\mathcal{L}_q$ represent the entity set, relation set, and textual set, respectively. The objective of this stage is to build the evidence sub-graphs $\mathcal{G}_q = \{\mathcal{G}_q^1, \mathcal{G}_q^2, \dots\}$ based on the extracted entities $\mathcal{V}_q$. An evidence sub-graph is defined by $\mathcal{G}_q^* = (\mathcal{V}_q, \mathcal{E}_q, \psi_q, \varphi_q, \mathcal{L}_q)$, where $\mathcal{V}_q$ is the node set, $\mathcal{E}_q$ is the edge set where each edge $e = \langle v, v'\rangle$, $v, v' \in \mathcal{V}_q$.

We explore the source KG to build evidence sub-graphs with *path-based* and *neighbor-based* exploration, as shown in Figure 4.

- **Path-based Evidence Graph Exploration** connects entities in $\mathcal{V}_q$ by tracing their intermediary pathways within $\mathcal{G}$: **(a)** Choose one node in $\mathcal{V}_q^0$ as the start node v_1. Place the remaining nodes in a candidate node set $\mathcal{V}_{cand}$. Explore at most k hops from v_1 to find the next node v_2, where $v_2 \in \mathcal{V}_{cand}$. If v_2 is successfully reached within k hops, update the start node as v_2 and remove v_2 from $\mathcal{V}_{cand}$. If v_2 cannot be found within k hops, connect the segments of paths obtained so far and store them in $\mathcal{G}_q^{\text{path}}$. Then choose another node $v_1{}'$ from $\mathcal{V}_{cand}$ as the new start node, and remove both v_1 and v_2 from $\mathcal{V}_{cand}$. **(b)** Check if $\mathcal{V}_{cand}$ is empty. If it is not empty, iterate **(a)** to find the next segment of the path. If it is empty, connect all segments to build a set of sub-graphs and put them into $\mathcal{G}_q^{\text{path}}$.
- **Neighbor-based Evidence Graph Exploration** aims to incorporate more query-related evidence into $\mathcal{G}_q$. It has two steps: **(a)** Expand for each node $v \in \mathcal{V}_q$ by 1-hop to their neighbors $\{v'\}$ to add triples $\{(v, e, v')\}$ to $\mathcal{G}_q^{\text{nei}}$. **(b)** We will check if every v' is semantically related to the question. If so, we further expand the 1-hop neighbors of v', adding triples (v_{nei}, e', v') to $\mathcal{G}_q^{\text{nei}}$.

We will also update $\mathcal{V}_q$ with all newly added intermediate nodes in bridging pathways. After exploration, we perform pruning over $\mathcal{G}_q^{\text{path}}$ and $\mathcal{G}_q^{\text{nei}}$ to reduce the information overhead while keeping diversity: **(a)** Cluster all sub-graphs by their head entities, which were extracted from the raw query and filled in $\mathcal{V}_q$ before the exploration. **(b)** Check each cluster and randomly sample k if the number of graphs exceed the pre-defined threshold. By merging the remained sub-graphs, we can obtain the final evidence graphs $\mathcal{G}_q$.

3.2 Step II: Evidence Graph Aggregation

In this step, LLM is prompted to aggregate the diverse evidence sub-graphs $\mathcal{G}_q$ into a merged *reasoning graph* $\mathcal{G}_m$, in order to bring a global view of all evidence sub-graphs for the output generation in Step III.

We add the instruction to prompt LLM to describe the structure of each evidence sub-graph $\mathcal{G}_q^*$ with natural language. It generates the evidence route set $\mathcal{C}_q = \left\{c_q^{(1)}, \dots, c_q^{(n)}\right\}$, where each $c_q^{(*)}$ represents a structured pathway, e.g., *"(Fatigue, Nausea) - IsSymptomOf - Liver-Problem"*. We perform the prompting for all path-based sub-graphs $\mathcal{G}_q^{\text{path}}$ and all neighbor-based sub-graphs $\mathcal{G}_q^{\text{nei}}$ separately, producing the reasoning graphs $\mathcal{G}_m = \left\{\mathcal{G}_m^{\text{path}}, \mathcal{G}_m^{\text{nei}}\right\}$. This design offers two advantages: (1) LLM adopts a holistic view of small sub-graphs, condensing them into a unified and succinct format, and (2) LLM aids in the disambiguation of similar entities in natural language.

3.3 Step III: LLM Reasoning with Mind Map

In this step, LLMs are prompted with the merged two evidence graphs $\mathcal{G}_m^{\text{path}}$ and $\mathcal{G}_m^{\text{nei}}$ to produce the final outputs.

Prompting for Graph Reasoning To trigger LLM to generate a mind map and find final results, we provide a prompt that consists of five components: *system instruction*, *question*, *evidence graphs* $\mathcal{G}_m$, *graph-of-thought instruction*, and *exemplars*. Specifically, the *graph-of-thought instruction* asks LLMs to think step by step to produce the final answer while **(a)** comprehending the input evidence graphs, **(b)** building its own mind map for reasoning, and **(c)** grounding the mind map with the evidence graphs. We provide the used prompt template in Table 8 in Appendix D. The final answers are with the detailed graph reasoning pathways on the mind map, where each entity in the graph is grounded on the input evidence graph $\mathcal{G}_m$, as shown in the right box in Figure 3.

Synergistic Inference with LLM and KG Knowledge We observed that previous retrieval-augmented LLMs tend only to rephrase the retrieved facts while overlooking the knowledge of LLM itself. However, we find `MindMap` encourages LLM to enrich the retrieved evidence graphs with its own knowledge, hence achieving a synergistic inference. We conjecture LLM achieves it by (1) *Language Understanding* as LLM can comprehend and extract the knowledge from $\mathcal{G}_m$ that is described by natural language and (2) *Knowledge Reasoning* as LLM leverages its implicit knowledge when producing the final answer grounded on the mind map built based on $\mathcal{G}_m$. As illustrated in Figure 2, LLM may identify new links connecting existing nodes with the contextual graph information, thereby enabling graph-of-thoughts and producing new insights.

Table 1: The statistics of the used datasets.

Dataset	Domain	Multi-task	Question	KG	Node	Triple	Relationship
GenMedGPT-5k	English Clinical Q&A	Disease, Drug, Test	714	EMCKG	1122	5802	6
CMCQA	Chinese Long Dialogue	Disease, Drug, Test, Food	468	CMCKG	62282	506490	12
ExplainCPE	5-way Choice Question	Option, Explanation	400	CMCKG	62282	506490	12

Table 2: The BERTScore and GPT4 ranking of all methods for **GenMedGPT-5k**.

	BERT Score			GPT4 Ranking
	Precision	Recall	F1 Score	(Average)
MindMap	**0.7936**	0.7977	**0.7954**	**1.8725**
GPT-3.5	0.7612	0.8003	0.7800	4.8571
GPT4	0.7689	0.7893	0.7786	4.1764
BM25 Retriever	0.7693	0.7981	0.7831	3.5546
Embedding Retriever	0.7690	0.8038	0.7857	3.1232
KG Retriever	0.7717	0.8030	0.7868	3.4159

4 Experiments

We evaluate our method for a suite of question & answering tasks that require sophisticated reasoning and domain knowledge and compare it with retrieval-based baselines.

4.1 Experimental Setup

We involved three Q&A datasets: (1) *GenMedGPT-5k* (Li et al. 2023b), (2) *CMCQA* (Weng 2022), (3) *ExplainCPE* (Li et al. 2023a). The statistics of these datasets are in Table 1. We describe the data curation process in Appendix A. In addition, we established two KGs, *EMCKG* and *CMCKG*, to support the KG-enhanced methods. Please refer to Appendix B for their implementation and statistics.

We pick the vanilla GPT-3.5 and GPT-4 as the baselines that infer with only implicit knowledge. We then add three retrieval-augmented baselines: *BM25 retriever*, *Text Embedding retriever*, and *KG retriever*. Unless otherwise stated, we use GPT-3.5 as the backbone for all retrieval-based methods. Please refer to Appendix C for the details of these baselines.

4.2 Medical Question Answering

We used GenMedGPT-5K to test how LLMs deal with question-answering in the medical domain, where LLMs need to answer with disease diagnosis, drug recommendation, and test recommendation.

Evaluation Metrics We used *BERTScore* (Zhang et al. 2019a) and *GPT-4 Rating* to evaluate the performance of all methods quantitatively. *BERTScore* can measure the semantic similarity between the generated answer with the reference answer. We also leveraged GPT-4 to comprehend the answers to (1) rank the quality of answers of all methods using the groundtruth as the reference and (2) make a pairwise comparison between the answers of two selected methods on four axes: *response diversity and integrity*, *overall factual correctness*, *correctness of disease diagnosis*, and *correctness of drug recommendation*.

Results Table 2 presents the BERTScore and GPT-4 ranking scores. We identify that all methods yield similar BERTScore where MindMap shows marginal improvement over the others, which means all methods generate answers that are similar to the real answers. This might be attributed to the similarity in tone among all responses, resembling the language commonly used by clinicians, but BERTScore only measures the overall semantic similarity. However, we find that MindMap shows a significant improvement over others when ranked by GPT-4, where it gets an average ranking of 1.8725. In comparison, the vanilla baseline GPT-3.5 and GPT-4 make the worst answers, ranked 4.85 and 4.17, respectively. The main reason is that answering medical questions requires a comprehensive exploration and understanding of domain knowledge, while the inference with implicit knowledge of GPT models can lead to plausibly looking but wrong answers. By contrast, our method conducts a thorough exploration of the external KG and provides evidence-grounded answers.

We dive deep into the pair-wise comparison of these methods in more granular aspects. The results are in Table 3. It is observed that MindMap yields all-around superior performances over other baselines. When compared against the implicit knowledge baselines, MindMap gets a winning rate of 88.21% over GPT-3.5 and 82.97% over GPT-4, on average of four aspects. This further emphasizes the importance of introducing external knowledge to mitigate hallucinations of LLM. In addition, it is found MindMap also outperforms the other retrieval-based methods by a great margin. It is noteworthy that the *KG Retriever* baseline retrieves facts from the same KG as MindMap. Nonetheless, MindMap is advanced by the synergistic inference of KG knowledge and implicit LLM knowledge, with a comprehensive exploration of the source KG, thus producing accurate and diverse answers. The other retriever baselines, however, suffer from inaccurate retrieval, insufficient exploration of the knowledge base, or lengthy inputs.

4.3 Long Dialogue Question Answering

We conducted experiments on the CMCQA dataset where the dialogues are long and require more complex reasoning to cover all facts required for the answer. The GPT-4 ranking results are exhibited in Table 4, where MindMap still ranks better than most baselines but is similar to the KG Retriever. We further illustrate the pairwise winning rate results in Table 5, where MindMap consistently outperforms baselines by the GPT-4 judge.

Specifically, compared to the performance on GenMedGPT-5K, the gap between our method and baselines is narrowed. This is due to the used KG does not cover all facts required to answer the questions in CMCQA.

Table 3: The pair-wise comparison by GPT-4 on the winning rate of MindMap v.s. baselines on diversity & integrity score (%), fact total match score (%), and disease diagnosis (%), on **GenMedGPT-5k**.

MindMap vs Baseline	GPT-3.5			BM25 Retriever			Embedding Retriver			KG Reriever			GPT-4		
Metries	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose
Diversity & integrity	100	-	-	100	-	-	100	-	-	100	-	-	100	-	-
Total factualness	**80.11**	-	19.89	66.67	-	33.33	76.05	-	23.95	73.53	-	26.47	75.77	-	24.23
Disease diagnosis	84.73	0.14	15.13	75.91	1.26	22.83	77.03	1.96	21.01	66.67	2.94	30.39	73.11	1.40	25.49
Drug recommendation	88	5	7	87	8	5	72	13	15	74	19	7	83	8	9
Average	**88.21**	1.285	10.505	**82.395**	2.315	15.29	**81.27**	3.74	14.99	**78.55**	5.485	15.965	**82.97**	2.35	14.68

Table 4: The BERTScore and GPT-4 ranking of all methods for **CMCQA** dataset.

	BERT Score			GPT-4 Ranking
	Precision	Recall	F1 Score	(Average)
MindMap	**0.9415**	0.9321	0.9367	**2.3**
GPT-3.5	0.9385	0.9361	0.9372	3.4
GPT-4	0.9355	0.9358	0.9356	3.6
BM25 Retriever	0.9365	0.9348	0.9356	3.7
Embedding Retriever	0.9357	0.9334	0.9345	5.4
KG Retriever	0.9318	0.9348	0.9332	2.3

Nonetheless, MindMap is still better than all retrieval-based baselines, including KG Retriever. We conjecture the previous retrieval-based methods are prone to answering by inferring only based on the retrieved external knowledge. However, this simple prompting strategy undermines LLM's ability to comprehend intricate logic and complex dialogue using its implicit knowledge. On the contrary, MindMap combines external knowledge and its implicit knowledge in the graph reasoning process, thus benefiting from both to produce more accurate answers.

4.4 Generate with Partial Knowledge from KG

We investigate the performance of MindMap when retrieving from a KG that only partially covers the knowledge required for answering questions in *ExplainCPE*. It is crucial because it is common in production that LLM needs to generate the answer by combining its implicit knowledge and retrieved knowledge.

Evaluation Metrics We assess all methods on the *accuracy* of the generated choice and the *quality* of the generated explanations. To evaluate the explanation quality, we also employ BERTScore and GPT-4 ranking. Specifically, we ask the GPT-4 rater to prioritize the correctness of the explanation over the helpfulness or integrity.

Results We show the accuracy of all methods in Table 6. We find MindMap achieves better accuracy than most baselines, which further verifies the superiority of MindMap over document retrieval prompting methods. We also identify that simply putting the retrieved knowledge to the prompt sometimes undermines the answer quality, as KG Retriever and BM25 Retriever all perform worse than the vanilla GPT-3.5 model. We checked the wrong answers and found the retrieved facts could only partially answer the questions because of the mismatch between the used external knowledge bases and the questions. This finding emphasizes combining the explicit and implicit knowledge of LLMs to achieve robust inference. It is also witnessed GPT-4 is better than GPT-3.5+MindMap. We conjecture the test questions, drawn from National Licensed Pharmacist Examination, were in the pre-training corpus of GPT-4 and stored in GPT-4's internal memory.

We conducted an ablation analysis on the design of instruction prompts. We find that removing the prompt $\mathbf{p}_1$ that explicitly instructs LLM to "*combine with the knowledge you already have*" improves the model's performance by 8.2%. In addition, Table 7 signals the superior performance of MindMap in generating rationales for the answers, achieving a ranking of 2.98 by GPT-4.

4.5 In-depth Analysis

We further conducted an in-depth analysis of the cases by MindMap, focusing on the discussion of the following aspects.

How is MindMap robust to mismatched facts? The example question in Figure 5 in Appendix E contains multiple symptom facts, some of which are misleading. In this scenario, certain facts, such as *'jaundice in my eyes'*, contribute incorrect information when retrieving related evidence. For instance, baseline models like document retrievers retrieve knowledge associated with *'eye'* when searching for evidence, leading to failure in obtaining the correct disease. The recommended drugs and tests are also unrelated to liver disease. In contrast, our model MindMap accurately identifies the disease as *'cirrhosis'* and recommends the appropriate test, *'blood test'*, demonstrating its robustness.

How does MindMap aggregate evidence graphs considering entity semantics? As shown in Figure 6 in Appendix E, nodes like *'vaginitis'* and *'atrophic vaginitis'* appear in different evidence sub-graphs, but they are semantically identical. MindMap enables LLMs to disambiguate and merge different evidence graphs for reasoning. The final mind map outputs also locate the entities to the input evidence graphs. Additionally, Figure 6 demonstrates the GPT-4 rater's preference for the total factual correctness and disease diagnosis factual correctness of all methods. For total factual correctness, the GPT-4 rater points out that MindMap provides more specific disease diagnosis results, while the baseline only gives vague mentions and lacks treatment options. For disease diagnosis factual correctness, the

Table 5: The pair-wise comparison by GPT-4 on the winning rate of MindMap v.s. baselines on disease diagnosis and drug recommendation on CMCQA.

MindMap vs Baseline	GPT-3.5			BM25 Retriever			Embedding Retriver			KG Reriever			GPT-4		
Metrics	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose
Disease diagnosis	35.68	39.96	24.36	30.98	50.21	18.80	37.18	42.74	20.08	34.40	45.51	20.09	27.99	47.22	24.79
Drug recommendation	47.32	30.62	22.06	47.11	29.34	23.55	44.97	32.12	22.91	44.33	31.26	24.41	44.11	29.76	26.12
Average	41.5	35.29	23.21	39.045	39.775	21.175	41.075	37.43	21.495	39.365	38.385	22.25	36.05	38.49	25.455

Table 6: The accuracy of all methods for **ExplainCPE**. We calculate the rates of correct, wrong, and failed responses.

Method	Accuracy Rate(%)		
	Correct	Wrong	Failed
GPT-3.5	52.2	47.2	0.5
BM25 Retriever	50	44.2	5.7
Embedding Retriever	54.2	45.2	0.5
KG Retriever	42	44	14
GPT-4	72	27.7	0.2
MindMap	**61.7**	**37.7**	**0.5**
w/o prompt template $\mathbf{p}_1$	53.5	46	0.5

Table 7: Quantitative comparison with BERTScore and GPT-4 preference ranking between MindMap and baselines in ExplainCPE dataset.

	BERT Score			GPT-4 Ranking
	Precision	Recall	F1 Score	(Average)
MindMap	0.9335	0.9376	0.9354	**2.98**
GPT-3.5	0.9449	0.9487	0.9467	3.0425
GPT-4	0.9487	0.9529	0.9507	3.0075
BM25 Retriever	0.9413	0.9411	0.9411	3.6675
Embedding Retriever	0.9440	0.9459	0.9449	4.3175
KG Retriever	0.9354	0.9373	0.9362	3.985

GPT-4 rater thinks the diagnosis by MindMap matches the groundtruth better.

How does MindMap visualize the inference process and evidence sources? Figure 7 in Appendix E presents a complete answer to a question from CMCQA. The output consists of three components: a summary, an inference process, and a mind map. The summary extracts the correct and specific result from the mind map, and the inference process shows multiple chains of reasoning starting from the extracted entities on the evidence graph $\mathcal{G}_m$. And the mind map unifies all the inference chains into the reasoning graph. It offers an intuitive understanding of the knowledge connections involved in each reasoning step and the original sources of evidence query sub-graphs.

How does MindMap leverage LLM knowledge for various tasks? Figure 8 in Appendix E demonstrates the performance of MindMap on different types of questions. Questions (a) and (d) are drug-related questions that require in-depth knowledge. In these tasks, MindMap exhibits a clear advantage over other methods. Questions (b) and (f) are disease knowledge questions. In these cases, the retrieval methods achieve comparable results to MindMap, indicating that the incorporation of external knowledge relieves hallucinations of LLM. The most significant observation comes from question (c), which involves general knowledge. In this case, LLM models like GPT-3.5 perform better, while the retrieval method performs poorly. It suggests that the retrieval methods have to some extent, overlooked the knowledge that LLM has learned. Conversely, MindMap performs equally well as the GPT-3.5 in handling general knowledge questions. This demonstrates that MindMap effectively synergizes the knowledge of LLM itself with KG knowledge for reasoning purposes.

5 Conclusion

This paper introduced knowledge graph (KG) prompting that 1) endows LLMs with the capability of comprehending KG inputs and 2) facilitates LLMs inferring with a combined implicit knowledge and the retrieved external knowledge. We then investigate eliciting the mind map, where LLMs perform the reasoning and generate the answers with rationales represented in graphs. Through extensive experiments on three question & answering datasets, we demonstrated that our approach, MindMap, achieves remarkable empirical gains over vanilla LLMs and retrieval-augmented generation methods. We envision this work opens the door to fulfilling reliable and transparent LLM inference in production.

References

Ali, R.; Tang, O. Y.; Connolly, I. D.; Fridley, J. S.; Shin, J. H.; Sullivan, P. L. Z.; Cielo, D.; Oyelese, A. A.; Doberstein, C. E.; Telfeian, A. E.; et al. 2022. Performance of ChatGPT, GPT-4, and Google bard on a neurosurgery oral boards preparation question bank. *Neurosurgery*, 10–1227.

Ateia, S.; and Kruschwitz, U. 2023. Is ChatGPT a Biomedical Expert?–Exploring the Zero-Shot Performance of Current GPT Models in Biomedical Tasks. *arXiv preprint arXiv:2306.16108*.

Baek, J.; Aji, A. F.; and Saffari, A. 2023. Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering. *arXiv preprint arXiv:2306.04136*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Cao, Y.; Kang, Y.; and Sun, L. 2023. Instruction Mining: High-Quality Instruction Data Selection for Large Language Models. *arXiv preprint arXiv:2307.06290*.

Chen, Z.; Mao, H.; Li, H.; Jin, W.; Wen, H.; Wei, X.; Wang, S.; Yin, D.; Fan, W.; Liu, H.; et al. 2023. Exploring the Potential of Large Language Models (LLMs) in Learning on Graphs. *arXiv preprint arXiv:2307.03393*.

Choudhary, N.; and Reddy, C. K. 2023. Complex Logical Reasoning over Knowledge Graphs using Large Language Models. *arXiv preprint arXiv:2305.01157*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2022. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Danilevsky, M.; Qian, K.; Aharonov, R.; Katsis, Y.; Kawas, B.; and Sen, P. 2020. A Survey of the State of Explainable AI for Natural Language Processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, 447–459.

Guo, J.; Du, L.; and Liu, H. 2023. GPT4Graph: Can Large Language Models Understand Graph Structured Data? An Empirical Evaluation and Benchmarking. *arXiv preprint arXiv:2305.15066*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12): 1–38.

Jia, Z.; Pramanik, S.; Saha Roy, R.; and Weikum, G. 2021. Complex temporal question answering on knowledge graphs. In *Proceedings of the 30th ACM international conference on information & knowledge management*, 792–802.

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

Li, D.; Yu, J.; Hu, B.; Xu, Z.; and Zhang, M. 2023a. ExplainCPE: A Free-text Explanation Benchmark of Chinese Pharmacist Examination. *arXiv preprint arXiv:2305.12945*.

Li, Y.; Li, Z.; Zhang, K.; Dan, R.; Jiang, S.; and Zhang, Y. 2023b. ChatDoctor: A Medical Chat Model Fine-Tuned on a Large Language Model Meta-AI (LLaMA) Using Medical Domain Knowledge. *Cureus*, 15(6).

Liu, N. F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; and Liang, P. 2023a. Lost in the Middle: How Language Models Use Long Contexts. *arXiv preprint arXiv:2307.03172*.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9): 1–35.

Liu, W.; Zhou, P.; Zhao, Z.; Wang, Z.; Ju, Q.; Deng, H.; and Wang, P. 2020. K-BERT: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 2901–2908.

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

Pan, S.; Luo, L.; Wang, Y.; Chen, C.; Wang, J.; and Wu, X. 2023. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *arXiv preprint arXiv:2306.08302*.

Peng, B.; Galley, M.; He, P.; Cheng, H.; Xie, Y.; Hu, Y.; Huang, Q.; Liden, L.; Yu, Z.; Chen, W.; et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Razdaibiedina, A.; Mao, Y.; Hou, R.; Khabsa, M.; Lewis, M.; and Almahairi, A. 2022. Progressive Prompts: Continual Learning for Language Models. In *The Eleventh International Conference on Learning Representations*.

Roberts, A.; Raffel, C.; and Shazeer, N. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Robertson, S.; Zaragoza, H.; et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4): 333–389.

Sharma, A.; and Kumar, S. 2023. Ontology-based semantic retrieval of documents using Word2vec model. *Data & Knowledge Engineering*, 144: 102110.

Singhal, K.; Azizi, S.; Tu, T.; Mahdavi, S. S.; Wei, J.; Chung, H. W.; Scales, N.; Tanwani, A.; Cole-Lewis, H.; Pfohl, S.; et al. 2023. Large language models encode clinical knowledge. *Nature*, 1–9.

Sun, J.; Xu, C.; Tang, L.; Wang, S.; Lin, C.; Gong, Y.; Shum, H.-Y.; and Guo, J. 2023. Think-on-Graph: Deep and Responsible Reasoning of Large Language Model with Knowledge Graph. *arXiv preprint arXiv:2307.07697*.

Sun, T.; Shao, Y.; Qiu, X.; Guo, Q.; Hu, Y.; Huang, X.-J.; and Zhang, Z. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, 3660–3670.

Sun, Y.; Wang, S.; Feng, S.; Ding, S.; Pang, C.; Shang, J.; Liu, J.; Chen, X.; Zhao, Y.; Lu, Y.; et al. 2021. ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Wang, C.; Cheng, S.; Xu, Z.; Ding, B.; Wang, Y.; and Zhang, Y. 2023. Evaluating open question answering evaluation. *arXiv preprint arXiv:2305.12421*.

Wang, X.; Kapanipathi, P.; Musa, R.; Yu, M.; Talamadupula, K.; Abdelaziz, I.; Chang, M.; Fokoue, A.; Makni, B.; Mattei, N.; et al. 2019. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 7208–7215.

Wei, J.; Bosma, M.; Zhao, V.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2022. Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*.

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

Weng, Y. 2022. A large Chinese Medical CQA. https://github.com/WENGSYX/CMCQA.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yao, Y.; Li, Z.; and Zhao, H. 2023. Beyond Chain-of-Thought, Effective Graph-of-Thought Reasoning in Large Language Models. *arXiv preprint arXiv:2305.16582*.

Yasunaga, M.; Bosselut, A.; Ren, H.; Zhang, X.; Manning, C. D.; Liang, P. S.; and Leskovec, J. 2022. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35: 37309–37323.

Yasunaga, M.; Ren, H.; Bosselut, A.; Liang, P.; and Leskovec, J. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K. Q.; and Artzi, Y. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Zhang, X.; Bosselut, A.; Yasunaga, M.; Ren, H.; Liang, P.; Manning, C. D.; and Leskovec, J. 2022. GreaseLM: Graph reasoning enhanced language models. In *International Conference on Learning Representations*.

Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; and Liu, Q. 2019b. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 1441–1451.

A Construction of Datasets

- **GenMedGPT-5k** is a 5K generated dialogue between patients and GPT-3.5 grounded on a disease database[1]. The question describes the symptoms of the patient during the consultation, which comes from the iCliniq database. Based on the database, the generated answers cover the diagnosis, symptoms, recommended treatments, and medical tests. We sampled 714 dialogues as the test set.
- **CMCQA** contains multi-round dialogues between patients and doctors in Chinese. It covers materials from 45 clinical departments such as andrology, gynecology, and obstetrics and gynecology. We simplified the setup by merging the patient's questions and the clinician's answers to build the one-round Q&A. We sampled 468 from all to build the test set.
- **ExplainCPE** is a 5-way choice question dataset from the Chinese National Licensed Pharmacist Examination. Answering the questions requires a series of capabilities, including logical reasoning, drug knowledge, scenario analysis, mathematical calculation, disease knowledge, and general knowledge. The answers include the correct options and the explanations. We extracted 400 samples related to disease diagnosis and treatment recommendations from the original dataset for testing.

B Implementation of Knowledge Graph

- **EMCKG** We utilized a disease database[2] to build the KG from scratch to support the knowledge source for the inference on GenMedGPT-5k. This database encompasses a diverse set of diseases and the corresponding symptoms, medical tests, treatments, etc. The entities in the EMCKG include disease, symptom, drug recommendation, and test recommendation. The relationships in the EMCKG include *'possible_disease'*, *'need_medical_test'*, *'need_medication'*, *'has_symptom'*, *'can_check_disease'*, *'possible_cure_disease'*. In total, the yielded KG contains of 1122 nodes and 5802 triples.
- **CMCKG** We established a KG based on *QASystemOnMedicalKG*[3] to support KG-augmented inference on CMCQA and ExplainCPE. The CMCKG includes various entities such as disease, symptom, syndrome, recommendation drugs, recommendation tests, recommendation food, and forbidden food. The relationships in the CMCKG include *'has_symptom'*, *'possible_disease'*, *'need_medical_test'*, *'has_syndrome'*, *'need_recipe'*, *'possible_cure_disease'*, *'recipe__is_good_for_disease'*, *'food__is_good_for_disease'*, *'food__is_bad_for_disease'*, *'need_medication'*, *'need_food'*, and *'forbid_food'*. In total, the KG contains 62282 nodes, 12 relationships, and 506490 triples.

[1]https://github.com/Kent0n-Li/ChatDoctor

[2]https://github.com/Kent0n-Li/ChatDoctor/blob/main/format_dataset.csv

[3]https://github.com/liuhuanyong/QASystemOnMedicalKG/blob/master/data/medical.json

C Implementation of Baselines

- **GPT-3.5 & GPT-4** We evaluate the performance of the recent dominant LLM models as two baselines, using *gpt-3.5-turbo* (Wang et al. 2023; Ateia and Kruschwitz 2023) and *gpt-4*[4] (Ali et al. 2022; Guo, Du, and Liu 2023) API respectively. In order to ensure a consistent input, we set the temperature to 0. For choice question datasets MedQA-USMLE, we additionally add the input "*First output current choice, then output explanation.*" to prompt the output format.
- **BM25 document retriever + GPT-3.5** We compare with existing BM25 document retriever methods (Roberts, Raffel, and Shazeer 2020; Peng et al. 2023), which use BM25 retrieval scores (Robertson, Zaragoza et al. 2009) as logits when calculating $p(z|x)$. For fair comparisons, we use the same KG database as our method to generate different document files. Specifically, we use the GPT-3.5 API to convert all knowledge data centered on one disease into natural language text as the content of a document. For GenMedGPT-5k and USMLE, we make 99 documents based on English medical KG $\mathcal{G}_{English}$. For CMCQA and ExplainCPE, we make 8808 documents based on Chinese medical KG $\mathcal{G}_{Chinese}$. For each question query, k gold document contexts retrieved from a bm25 score retriever.
- **Text embedding document retrieval + GPT-3.5** Same as bm25 document retriever methods, text embedding document retrieval methods (Sharma and Kumar 2023; Lewis et al. 2020) retrieve the top k documents for each question query. The difference is that in this method we train a word2vec embedding as the evidence source for document ranking.
- **KG retrieval + GPT-3.5** We compare with existing KG retrieval methods (Jia et al. 2021; Sun et al. 2023), which aim to find the shortest KG path between every pair of question entities. The final prompt is then retrieved from KG to guide GPT-3.5 model to answer questions. For fair comparisons, we use the same preliminary process as our method to recognize the entities in question query. The key differences between `MindMap` and these are that they do not think on multiple evidence KG sub-graphs with multi-thought in LLM, and without backtracking evidence sources.

D Prompt Engine

Table 8 presents the final prompt used by `MindMap` for generating results and constructing a mind map. The prompt consists of a system message acknowledging the AI's expertise as a doctor, a user message representing the patient's input, and an AI message incorporating knowledge obtained from an external KG. The Langchain technique is employed to create the prompt. The response consists of a summary answer to the query, the inference process, and a mind map. Table 9 illustrates an example of the pairwise ranking evaluation using the GPT-4 rater.

[4]https://openai.com/gpt-4

E In-depth Analysis

We select four examples for in-depth analysis, as shown in Figure 5, 6, 7, and 8.

- Figure 5 presents an example from GenMedGPT-5k. It includes the question, reference response, the response generated by `MindMap`, responses from baselines, and the factual correctness preference determined by the GPT-4 rater. This example is used to discuss the robustness of `MindMap` in handling mismatched facts.
- Figure 6 illustrates another example from GenMedGPT-5k. It displays the question query, reference response, summary responses from both `MindMap` and baseline models, a mind map generated by `MindMap`, and specific preferences in terms of factual correctness and subtask disease fact match determined by the GPT-4 rater. This example shows the ability of `MindMap` to aggregate evidence graphs.
- Figure 7 showcases an example from CMCQA. It includes the question query, a summary answer, the inference process, and the generated mind map by `MindMap`. This example provides insights into the visualization of the final output produced by `MindMap`.
- Figure 8 demonstrates an example from ExplainCPE. It consists of six questions categorized into three different question types and evaluates the accuracy of `MindMap` and baseline models. This example allows us to examine the performance of `MindMap` across various tasks.

F Pairwise Ranking Evaluation

For each pair of answers, as an example in Table 9, raters were asked to select the preferred response or indicate a tie along the following axes (with exact instruction text in quotes):

- **Diversity and integrity:** "According to the result in reference output, which output is better."
- **Total factual correctness:** "According to the facts of disease diagnosis and drug and tests recommendation in reference output, which output is better match."
- **Disease diagnosis:** "According to the disease diagnosis result in reference output, which output is better match."
- **Drug recommendation:** "According to the drug recommendation result in reference output, which output is better match."

Note that for the second dataset CMCQA, since the reference label is derived from the actual dialogue answer, it may not contain facts. When the GPT-4 rater performs pairwise ranking evaluation, it is very easy to judge it as a tie. Therefore, we add an additional instruction: "*If they are the same, output "2". Try to output "1" or "0"*", so as to force the rater to make a preference judgment.

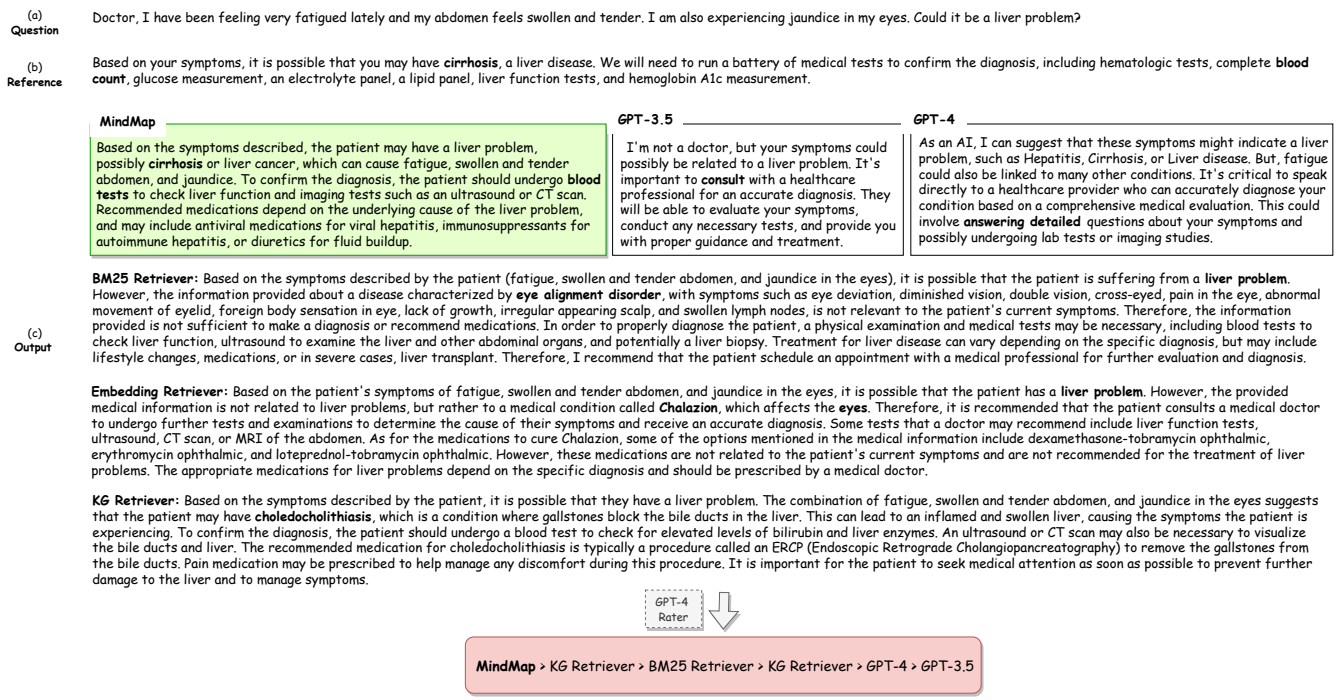

Figure 5: **A case compares `MindMap` and baselines with mismatched retrieved knowledge**, evaluated by the GPT factual correctness preference rater.

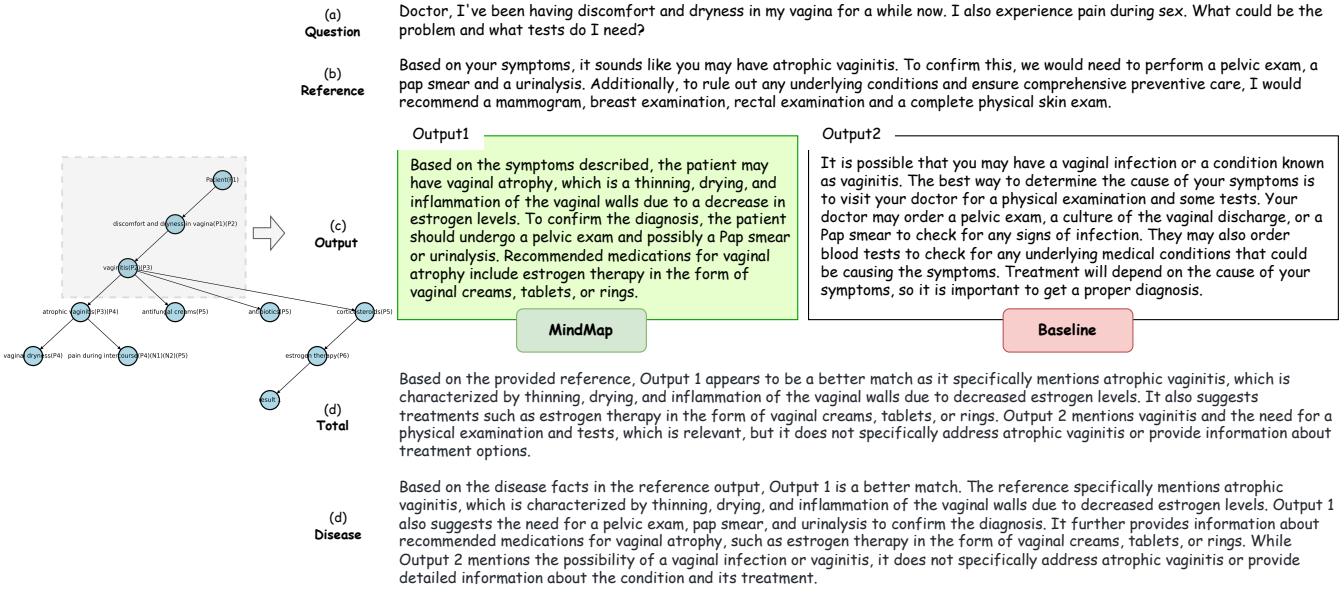

Figure 6: **Factually correctness evaluation in GenMedGPT-5k using GPT-4 preference ranking**: `MindMap` shows a strong ability in fact-matching subtasks of question-answering by generating a mind map.

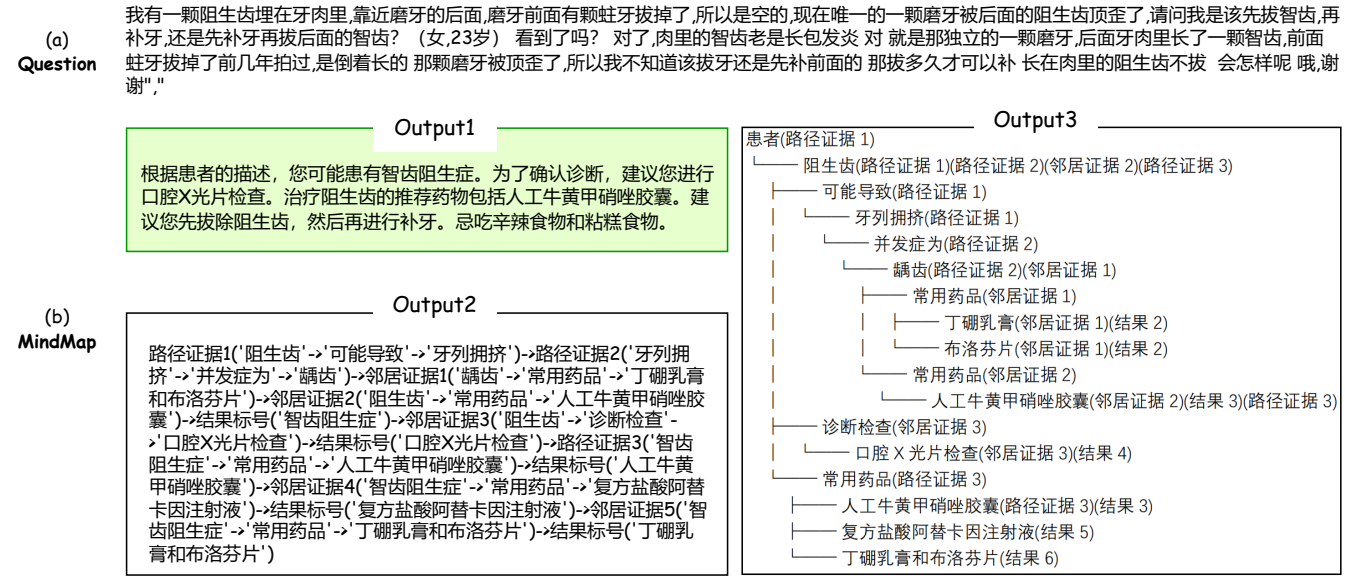

Figure 7: **An example to show the visualization of `MindMap`**. By generating mind maps, our approach guides LLM to obtain the correct factual outputs for different subtasks.

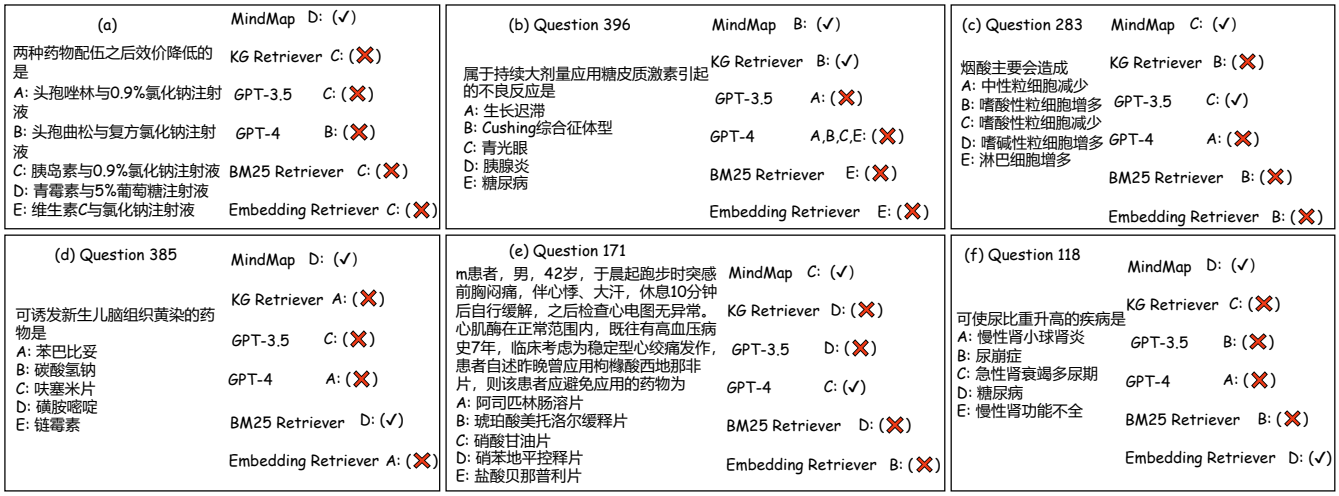

Figure 8: **Case examples of multi-choice reasoning in ExplainCPE**, comparing predictions by other Baselines and our model (`MindMap`).

```
def final_answer(str,response_of_KG_list_path,response_of_KG_neighbor):
    messages  = [
    SystemMessage(content="You are an excellent AI doctor, and you can diagnose diseases
        and recommend medications based on the symptoms in the conversation. "),
    HumanMessage(content="Patient input:"+ input_text[0]),
    AIMessage(content="Combine the knowledge you already have, you have some extra medical
         knowledge information in the following:\n\n" +  '###'+ response_of_KG_list_path +
         '\n\n' + '###' + response_of_KG_neighbor),
    HumanMessage(content="What disease does the patient have? What tests should patient
        take to confirm the diagnosis? What recommened medications can cure the disease?
        Think step by step.\n\n\n"
    + "Output1: The answer includes disease and tests and recommened medications.\n\n"
    +"Output2: Show me inference process as a string about extract what knowledge from
        which Path-based Evidence or Neighor-based Evidence, and in the end infer what
        result. \n Transport the inference process into the following format:\n Path-based
         Evidence number('entity name'->'relation name'->...)->Path-based Evidence number
        ('entity name'->'relation name'->...)->Neighbor-based Evidence number('entity name
        '->'relation name'->...)->Neighbor-based Evidence number('entity name'->'relation
        name'->...)->result number('entity name')->Path-based Evidence number('entity name
        '->'relation name'->...)->Neighbor-based Evidence number('entity name'->'relation
        name'->...). \n\n"
    +"Output3: Draw a decision tree. The entity or relation in single quotes in the
        inference process is added as a node with the source of evidence, which is
        followed by the entity in parentheses.\n\n"
    + "There is a sample:\n"
    + """... """)]
    result = chat(messages)
    output_all = result.content
    return output_all
```

Table 8: The prompt template for final input to LLM. Its input is the question query and the merged graph generated by the LLM.

```
def prompt_comparation(reference,output1,output2):
    template = """
    Reference: {reference}
    \n\n
    output1: {output1}
    \n\n
    output2: {output2}
    \n\n
    According to the facts of disease diagnosis and drug and tests recommendation in
        reference output, which output is better match. If the output1 is better match,
        output '1'. If the output2 is better match, output '0'. If they are same match,
        output '2'.
    """
    prompt = template.format(reference=reference, output1=output1, output2=output2)
    response = openai.ChatCompletion.create(
      model="gpt-4",
      messages=[
          {"role": "system", "content": "You are an excellent AI doctor."},
          {"role": "user", "content": prompt}
      ]
    )
    response_of_comparation = response.choices[0].message.content
    return response_of_comparation
```

Table 9: The prompt template for GPT-4 rater to evaluate the factual correctness between our method and baselines, the reference is the answer or explanation label.