Optimizing Knowledge Graph to Text as Knowledge-Augmented Prompt with Alignment Tuning for Question Answering

Anonymous ACL submission

Abstract

Large language models have limitations in maintaining up-to-date knowledge and preventing hallucinations. To address these issues, recent research has explored integrating external knowledge sources into language models, with Knowledge Graphs emerging as a particularly promising approach since their structured and factual nature. However, effectively incorporating knowledge graphs into language models remains challenging due to the modality gap and the lack of query-aware knowledge selection in existing knowledge-to-text methods. This paper proposes a Knowledge Graph to Knowledge-Augmented Prompt (KG2P), a framework that optimizes knowledge graph-to-text transformation for language model prompting. KG2P introduces black-box optimization to systematically learn effective knowledge transformation and query-aware alignment to enhance relevance. Unlike previous approaches that rely on rigid linearization or static human annotations, KG2P dynamically adapts knowledge augmentation to improve reasoning in language models. Experimental results on knowledge graph question-answering benchmarks demonstrate that KG2P consistently outperforms existing methods. The findings suggest that task-specific optimization is essential for effectively incorporating structured knowledge into language models, providing a new direction for knowledge-augmented prompting.

1 Introduction

Large language models (LLMs) have demonstrated remarkable success in various natural language processing (NLP) tasks by pretraining on massive text corpora. However, they still suffer from significant limitations, such as hallucinations, where they generate factually incorrect information, and a lack of up-to-date knowledge due to their static training data. To address this issue, extensive research has been conducted on methods, which enable LLMs to generate responses by referencing external knowledge such as Retrieval-Augmented Generation (RAG).

Typically, unstructured text data, such as web documents, is retrieved and utilized, but such data often contains redundant or conflicting information, which may reduce reliability. In contrast, Knowledge Graphs (KGs) provide systematically structured information, making them a more reliable knowledge base. A KG stores information in the form of triples (subject, relation, object), explicitly representing relationships between entities. Due to this structured nature, KGs serve as not just a data repository but also a crucial knowledge source that enhances the factual accuracy and reasoning ability of LLMs.

Despite their advantages, integrating KGs with LLMs presents a fundamental modality gap since KGs store information in graph format, whereas LLMs primarily process textual input. To bridge this gap, researchers have explored KG-to-Text transformation methods that convert structured graph knowledge into textual representations (Li et al., 2020; Agarwal et al., 2021; Moiseev et al., 2022; Kim et al., 2025a). A common approach is linearization, which involves converting KG triples into triple-form text that LLMs can process. For example, Baek et al. (2023a) proposed retrieving relevant triples, linearizing them, and incorporating them into prompts to improve question-answering (QA) performance. However, Wu et al. (2023) criticized this approach for relying on rigid triple-form text rather than free-form text, which is more naturally processed by LLMs. To address this issue, Wu et al. (2023) introduced a rewriting step that transforms structured triples into more fluid, human-like text.

While these methods mitigate some aspects of the modality gap, two key challenges persist. First, it is still unclear how knowledge from KGs should be optimally transformed into text to maximize the effectiveness of LLMs. Since LLMs function as black-box models, little is known about how KG-based knowledge is internally processed when inserted into prompts. In particular, closed LLMs that are provided solely as APIs without open-source code function as complete black boxes. Second, existing KG-to-Text transformation methods do not account for query dependency. The relevance of KG-derived information varies depending on the query, but prior approaches perform textualization independently of the query context. Recent studies (Yasunaga et al., 2021; Kim et al., 2025b) have demonstrated that query-dependent graph representations improve downstream performance, suggesting that KG-to-Text transformation should also be query-aware.

To address these challenges, we propose **K**nowledge **G**raph to Knowledge-Augmented **P**rompt (KG2P), a novel KG-to-Text transformation framework designed to enhance downstream task performance of LLMs. KG2P introduces two major innovations. First, it applies black-box optimization to refine structured graph knowledge transformation for LLMs. Second, KG2P incorporates query dependency into the transformation process, ensuring that only the most relevant KG information is retained when constructing prompts.

We evaluate KG2P on the Knowledge Graph Question Answering (KGQA) task and demonstrate that it outperforms conventional KG-to-Text methods in QA performance. Moreover, our experiments reveal that traditional KG-to-Text techniques, despite being widely adopted in general knowledge conversion tasks, perform suboptimally when used for LLM prompt construction in KGQA. To empirically verify the robustness of our approach, we conduct model cross-experiments, where its effectiveness is tested with different LLM.

2 Related Works

2.1 Knowledge Graph-to-Text

KG-to-Text involves transforming structured triples into free-form text. For example, given a set of triple such as "(Parasite, director, Bong Joon-Ho), (Parasite, Genre, Thriller Film), (Bong Joon-Ho, educated at, Yonsei University)", a KG-to-Text model converts them into "Parasite, a film directed by Bong Joon-Ho who was educated at Yonsei University, is a thriller film".

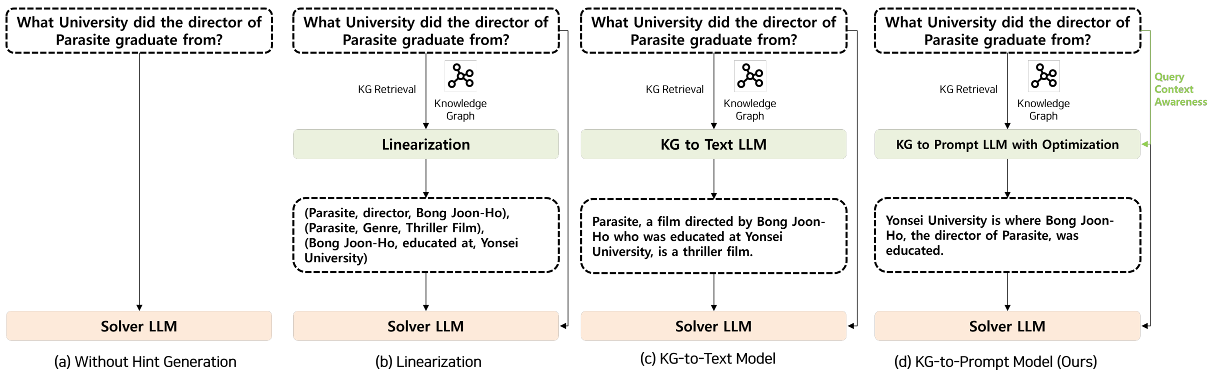
This task is often approached similarly to machine translation using sequence-to-sequence manners (Li et al., 2020; Sutskever, 2014). Agarwal et al. (2021) employed a sequence-to-sequence model, specifically fine-tuning the T5 model (Raffel et al., 2020), to convert KGs into natural text. Similarly, Ribeiro et al. (2021) transformed graph-based data into fluent text using BART and T5 (Lewis et al., 2019; Raffel et al., 2020). These studies showed that these pre-trained language models achieve superior results across various graph domains, demonstrating the effectiveness of sequence-to-sequence architectures in graph-to-text generation tasks. Despite the advancements in KG-to-Text methods, their reliance on human-labeled data is not optimized for a downstream task.

2.2 Knowledge Graph Question Answering and Knowledge-Augmented Prompting

The objective of KGQA tasks is to respond to natural language queries based on facts over KGs (Chakraborty et al., 2019; Fu et al., 2020). Earlier approaches have employed neural semantic parsing methods that leverage deep learning techniques to map natural language questions into structured queries that can be executed against a KG. These methods typically rely on end-to-end training of neural networks that jointly learn representations of both questions and KGs (Yih et al., 2015; Luo et al., 2018). However, these methods have limitations which are difficult to generalize across diverse domain-specific KGs.

Another approach is an information retrieval-based method known as Retrieval-Augmented Graph QA. This approach retrieves relevant information from KGs based on a query and then answers the question using this information. Early methods addressed the multiple-choice QA task by training a neural classifier to utilize retrieved KG information (Sun et al., 2018; Saxena et al., 2020).

Recently, there has been a focus shifted toward employing LLMs for both multiple-choice and general QA tasks (Baek et al., 2023a; Wu et al., 2023). In order for LLMs to utilize the information in KGs, it is necessary to transform the information in the KG into text form. To address these issues Baek et al. (2023a) proposed knowledge-augmented prompting, an approach that incorporates retrieved triples from KGs directly into the prompts in a linearized format. However, this technique sometimes leads to responses that may lack natural linguistic coherence and be difficult for LLMs to process as shown in Figure 1(b). Wu et al. (2023) introduced a refinement by adding a step to rewrite these triples into a natural language



What University did the director of Parasite graduate from?
KG Retrieval
Knowledge Graph
Linearization
(Parasite, director, Bong Joon-Ho), (Parasite, Genre, Thriller Film), (Bong Joon-Ho, educated at, Yonsei University)
KG to Text LLM
Parasite, a film directed by Bong Joon-Ho who was educated at Yonsei University, is a thriller film.
KG to Prompt LLM with Optimization
Query Context Awareness
Yonsei University is where Bong Joon-Ho, the director of Parasite, was educated.
Solver LLM
(a) Without Hint Generation
(b) Linearization
(c) KG-to-Text Model
(d) KG-to-Prompt Model (Ours)

Figure 1: **Different Methods for KG-to-Text Conversion for QA Tasks.** The Linearization and KG-to-Text Model convert a KG into text independently of the question and are not optimized for knowledge-augmented prompting. In contrast, our proposed KG-to-Knowledge Augmented Prompt Model transforms KG into text while considering query contexts and is optimized for knowledge-augmented prompting.

sentence without any further training. This leads to enhanced KGQA performance as shown in Figure 1(c). Despite these advancements, these methods were not specifically optimized for KG-to-Text for a downstream task. Additionally, they failed to consider the context of questions when converting KG-to-Text.

2.3 Alignment Tuning

Ouyang et al. (2022) introduced a methodology to use reinforcement learning to align LLMs with human preference. Analogous to a black-box problem, the internal structure and formulaic representation of human preferences remains unknown. Furthermore, the non-differentiable nature of these preferences precludes traditional back-propagation for training LLMs. To overcome this challenge, Ouyang et al. (2022) developed a model to predict human preferences from survey data, utilizing these predictions as the foundation for a reward function in reinforcement learning known as Reinforcement Learning with Human Feedback (RLHF). Additionally, Rafailov et al. (2024) showed that the method for training LLMs aligns with human preferences by using paired positive and negative response data, without relying on reward models, known as Direct Preference Optimization (DPO).

3 Methodology

A problem that optimizes KG-to-Text for QA is a derivative-free and black-box optimization problem analogous to human preferences. This occurs because the manner in which LLM processes prompts is difficult to model mathematically, similar to human preferences, and the argmax operation and instruction prompt templates make differentiation impossible. Inspired by Rafailov et al. (2024), we propose a novel optimization method that uses positive and negative data pairs as language model preference guides and alignment tuning objective functions. We aim to improve the QA performance of LLMs, maintaining a pre-trained state as a black box without further training, to optimize its input prompts.

3.1 Problem Definition and Approach

In this study, we address the problem of KGQA. KGQA involves generating an answer to a given question q, based on a set of related triples $\{\mathcal{T}_1^q, \cdots, \mathcal{T}_n^q\}$ from KG. In this process, the given set of triples is transformed into a free-form text P which, along with q, is used as a prompt for the Solver LLM to generate the answer. The Solver LLM is not trained, but another LLM, $\mathbf{KG2P}_\theta$, is trained to convert $\{\mathcal{T}_1^q, \cdots, \mathcal{T}_n^q\}$ into P.

Formally, question q and given related triples $\{\mathcal{T}_1^q, \cdots, \mathcal{T}_n^q\}$ are passed into a KG2P template[1] $\mathbf{T_P} : (q, \{\mathcal{T}_1^q, \cdots, \mathcal{T}_n^q\}) \mapsto x_P$. This instruction x_P is transformed into P by $\mathbf{KG2P}_\theta$. Here, θ represents the trainable parameters of the $\mathbf{KG2P}_\theta$. During this process, $\mathbf{KG2P}_\theta$ incorporates the context of the query and generates knowledge-augmented prompt P.

$$P = \mathbf{KG2P}_\theta(x_P) \qquad (1)$$

The question q and the knowledge-augmented prompt P are passed through a QA template[2] $\mathbf{T}_{\text{QA}} : (q, P) \mapsto x_{\text{QA}}$. The Solver LLM takes x_{QA} as input

[1] Details about the prompt are found in the Appendix A
[2] It is also found in the Appendix A

and generates a predicted answer $\hat{y}$. Our work focuses on training θ to ensure that $\hat{y}$ matches the correct answer y.

$$\hat{y} = \textbf{Solver}(x_{\text{QA}}) \qquad (2)$$

We consider scenarios where the Solver LLM cannot be directly trained due to restricted access to its parameters or the prohibitively high cost of training. Therefore, the Solver LLM remains in its pre-trained state without undergoing further training. Thus, our objective is to train $\textbf{KG2P}_\theta$ to provide optimized knowledge-augmented prompts that help the Solver LLM generate correct answers, thereby enhancing QA performance.

3.2 Knowledge Graph to Knowledge-Augmented Prompt

We now present the details of our method, **K**nowledge **G**raph to Knowledge-Augmented **P**rompt (KG2P), a novel KG-to-Text strategy to help solve question-answering.

Our proposed method is divided into two phases. Figure 2 illustrates the overview of KG2P.

- **Exploration (First Phase)** Generating various synthetic knowledge-augmented prompt samples and automatically labeling them as positive or negative.

- **Exploitation (Second Phase)** DPO training the KG2P using the labeled prompt samples.

In the first phase, we generate automatically labeled data for $\textbf{KG2P}_\theta$ training. We adopt the DPO training strategy (Rafailov et al., 2024). In reinforcement learning, random actions are simulated across various situations, and the action search space is explored during training. DPO approximates this process by training on data consisting of positive and negative actions in the same context.

For DPO training, it is necessary to generate various knowledge-augmented prompts for the given triple set with the question and label each knowledge-augmented prompt as either good or bad. Here, we define a 'good' (positive) prompt as one that enables the Solver LLM to generate the correct answer to the given question, while a prompt that fails to do so is labeled as a 'bad' (negative) prompt. Based on this definition, we performed automatic labeling.

Formally, by applying nuclear sampling (Holtzman et al., 2019) to $\textbf{KG2P}_\theta$, we generate a prompt set ($\mathcal{P} = \{P_1, \cdots, P_{|\mathcal{P}|}\}$) of given triple set with question q. For each P_i, the Solver LLM predicts $\hat{y}_i$ corresponding $\mathbf{T_{QA}}(q, P_i)$. If $\hat{y}_i$ matches correct answer y, then P_i is labeled as a positive prompt; else P_i becomes a negative prompt of question q. The $\textbf{KG2P}_\theta$ produces five knowledge-augmented prompts with he nucleus sampling threshold 0.9.

In the second phase, $\textbf{KG2P}_\theta$ is trained using DPO to generate optimized knowledge-augmented prompts. This process consists of two key steps: (1) Supervised Fine-Tuning (SFT) and (2) DPO Training.

First, in the SFT step, $\textbf{KG2P}_\theta$ is pre-trained using only the positive prompts that were automatically labeled in the exploration phase. The goal of this step is to initialize $\textbf{KG2P}_\theta$ so that it can generate high-quality prompts that assist the Solver LLM in accurately predicting the correct answers. $\textbf{KG2P}_\theta$ is an autoregressive language model that is trained to minimize cross-entropy loss. The training dataset consists only of the positive prompts—those that led to correct answers during the exploration phase. After this process, the fine-tuned model is denoted as $\textbf{KG2P}_{\text{SFT}_\theta}$, which serves as the reference model in the subsequent DPO training step.

Next, in the DPO Training step, $\textbf{KG2P}_{\text{SFT}_\theta}$ undergoes further optimization to refine its ability to generate effective prompts. DPO increases the likelihood of generating positive prompts while decreasing the likelihood of generating negative ones. This approach is based on preference learning, where $\textbf{KG2P}_{\text{SFT}_\theta}$ is guided by the differences between positive and negative prompts. The loss function for DPO training is formulated as eq 3.

This loss function encourages $\textbf{KG2P}_{\text{DPO}_\theta}$ to assign higher probabilities to positive prompts compared to the reference model $\textbf{KG2P}_{\text{SFT}_\theta}$, while simultaneously lowering the probabilities of generating negative prompts. The training process follows these steps: (1) collect positive and negative prompts from the exploration phase, (2) initialize $\textbf{KG2P}_{\text{DPO}_\theta}$ with the same weights as $\textbf{KG2P}_{\text{SFT}_\theta}$, and (3) fine-tune $\textbf{KG2P}_{\text{DPO}_\theta}$ by minimizing the DPO loss function.

4 Experiments

4.1 Experiments Setup

We used Wiki5M (Wang et al., 2021) as a knowledge base, which is a subset of Wikidata (Vrandečić and Krötzsch, 2014). This dataset consists of

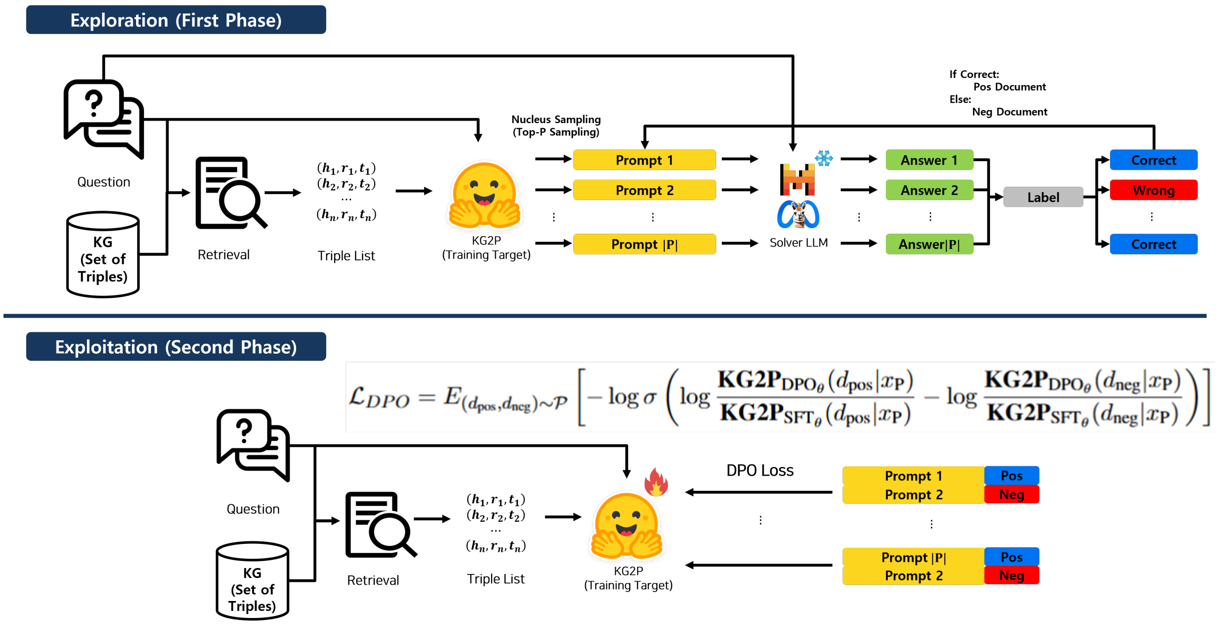

Exploration (First Phase)
Question
KG (Set of Triples)
Retrieval
(h_1, r_1, t_1) (h_2, r_2, t_2) ... (h_n, r_n, t_n)
Triple List
KG2P (Training Target)
Nucleus Sampling (Top-P Sampling)
Prompt 1
Prompt 2
Prompt |P|
Solver LLM
Answer 1
Answer 2
Answer|P|
If Correct: Pos Document
Else: Neg Document
Label
Correct
Wrong
Correct
Exploitation (Second Phase)
$\mathcal{L}_{DPO} = E_{(d_{\text{pos}}, d_{\text{neg}}) \sim \mathcal{P}} \left[-\log \sigma \left(\log \frac{\mathbf{KG2P}_{\text{DPO}_\theta}(d_{\text{pos}}|x_{\text{P}})}{\mathbf{KG2P}_{\text{SFT}_\theta}(d_{\text{pos}}|x_{\text{P}})} - \log \frac{\mathbf{KG2P}_{\text{DPO}_\theta}(d_{\text{neg}}|x_{\text{P}})}{\mathbf{KG2P}_{\text{SFT}_\theta}(d_{\text{neg}}|x_{\text{P}})} \right) \right]$
DPO Loss
Prompt 1 Pos
Prompt 2 Neg
Prompt |P| Pos
Prompt 2 Neg

Figure 2: **Overview of Knowledge Graph to Knowledge Augmented Prompt (KG2P).** In the first phase, data for training KG2H is generated. Triples related to the given question are retrieved in the KG, and the triples are used to create different knowledge-augmented prompt using **KG2P**. Each hint is evaluated to determine whether it is Positive or Negative by comparing labels with answers which is created by Solver LLM using the knowledge-augmented prompt. And then **KG2P**$_\theta$ is train by using $\mathcal{L}_{\text{DPO}}$

.

$$\mathcal{L}_{DPO} = E_{(d_{\text{pos}}, d_{\text{neg}}) \sim \mathcal{P}} \left[-\log \sigma \left(\log \frac{\mathbf{KG2P}_{\text{DPO}_\theta}(d_{\text{pos}}|x_{\text{P}})}{\mathbf{KG2P}_{\text{SFT}_\theta}(d_{\text{pos}}|x_{\text{P}})} - \log \frac{\mathbf{KG2P}_{\text{DPO}_\theta}(d_{\text{neg}}|x_{\text{P}})}{\mathbf{KG2P}_{\text{SFT}_\theta}(d_{\text{neg}}|x_{\text{P}})} \right) \right] \quad (3)$$

approximately 20 million triple sets, created from a combination of about 5 million entities and 1,000 relations. For KGQA datasets, we utilized the Simple Questions (SimQ) (Diefenbach et al., 2017), WebQSP-WD (WebQSP) (Sorokin and Gurevych, 2018), and Mintaka (Sen et al., 2022), where the answers to questions are entities within Wikidata. In our experiments, we focused only on questions with answer entities within Wiki5M. We utilized a text embedding-based retrieval system to identify triples relevant to the given questions, following the approach of Baek et al. (2023a).

Instruction-tuned LLMs are based on transformer (Vaswani et al., 2017) architecture and have shown impressive general natural language performance in responding to various queries. We utilized two instruction-tuned LLMs, LLaMa2[3] (Touvron et al., 2023) and Mistral[4] (Jiang et al., 2023), as both KG2P and Solver LLM. Implementation details are specified in the Appendix B.

For evaluation metrics, we measured (macro-averaged) F1 score and Exact Match score (EM) for QA task (Rajpurkar et al., 2016). F1 score computes the harmonic mean of precision and recall between the predicted answer and label, while the EM score calculates the percentage of predictions that match any one of the ground truth answers exactly.

Additionally, we reported the Bilingual Evaluation Understudy (BLEU) score (Papineni et al., 2002). The BLEU score evaluates the quality of machine-translated text by measuring the n-gram similarity between the translated text and reference translations. As aforementioned, a traditional KG-to-Text task is similar to a machine translation task. So we measure the BLEU score to explore the relationship between QA performance and traditional KG-to-Text performance. For this process, we used WebNLG test dataset (Li et al., 2020).

4.2 Overall QA Performance

In our initial experiment, we designed multiple baseline by varying the KG-to-Text methods. We compared five distinct settings:

- **Without KG Information (Without KG)**:

[3] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[4] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1

In this setting, no related triple set is used alongside the question as shown in Figure1(a).

- **Linearization (Linearization)**: A prompt is generated by the linearization process described in Baek et al. (2023a) as shown in Figure1(b).

- **Pre-trained LLM (Frozen)**: A pre-trained LLM produces knowledge-augmented prompts without any additional training. In this baseline, triples are transformed into a free-form text. This method employs the rewrite step similar to Wu et al. (2023). Wu et al. (2023) did not consider query context in the KG-to-Text conversion.

- **KG-to-Text Model (KG2T)**: This setting employs an LLM trained specifically for the KG-to-Text task using the dataset from Li et al. (2020)[5] as shown in Figure1(c). Although the KG-to-Text Model is optimized to replicate human-labeled data, it is not tailored for knowledge-augmented prompting which aims to improve QA accuracy.

- **Knowledge Graph to Knowledge-Augmented Prompt Model (KG2P-SFT, KG2P-DPO)**: This setting employs our method, which trains the prompts to produce more informative as shown in Figure1(d). We evaluated two versions: the SFT version, referred to as KG2P-SFT ($\mathbf{KG2P}_{\mathrm{SFT}_\theta}$), and the version trained using the DPO approach, referred to as KG2P-DPO ($\mathbf{KG2P}_{\mathrm{DPO}_\theta}$).

Table 1 shows the overall results. And Detailed examples are in Appendix C.

Table 1: **Comparison of KGQA Performance.** The table shows QA performance metrics (F1, EM) based on the prompt methods. **BOLD** indicates the highest score for each dataset regarding the Solver LLM.

Solver LLM	Prompting Method		Easy				Complex	
			SimQ		WebQSP		Mintaka	
			F1	EM	F1	EM	F1	EM
LLaMa2	Without KG		20.56	9.82	47.80	26.22	35.11	29.20
	Linearization		39.57	28.95	47.17	30.81	29.19	22.88
	LLaMa2	Frozen	36.26	24.63	49.11	31.80	39.28	33.22
		KG2T	36.42	24.77	43.93	27.32	25.41	19.26
		KG2P-SFT	35.43	24.49	51.15	33.20	40.84	34.85
		KG2P-DPO	41.21	30.72	**57.51**	**39.58**	**46.19**	**39.88**
	Mistral	Frozen	36.99	26.25	45.88	30.81	25.80	19.65
		KG2T	37.07	25.50	43.91	28.02	25.73	20.20
		KG2P-SFT	37.40	26.31	51.41	33.10	28.32	22.51
		KG2P-DPO	**41.31**	**30.84**	56.51	39.28	34.62	26.19
Mistral	Without KG		22.31	12.40	50.29	31.51	35.53	29.60
	Linearization		38.06	27.96	43.97	28.51	28.17	22.83
	LLaMa2	Frozen	37.02	25.73	47.40	29.31	30.61	24.14
		KG2T	36.58	25.56	40.10	22.63	25.87	19.34
		KG2P-SFT	36.20	25.52	49.73	31.01	32.89	26.69
		KG2P-DPO	**42.36**	32.32	**57.52**	**37.59**	**38.76**	**32.93**
	Mistral	Frozen	34.77	25.60	41.96	26.32	25.48	18.68
		KG2T	36.82	26.35	38.77	23.53	25.29	18.89
		KG2P-SFT	33.41	23.90	43.23	26.22	27.90	21.60
		KG2P-DPO	42.05	**32.60**	49.39	31.90	37.22	30.41

Table 2: Pearson correlation between KG-to-Text performance (BLEU) and QA performance (EM) across different prompting methods. The low or negative correlation values indicate little to no relationship between KG-to-Text and QA performance, suggesting that improvements in one do not necessarily result in improvements in the other.

Solver LLM	Prompting Method		SimQ		WebQSP		Mintaka	
			EM	BLEU	EM	BLEU	EM	BLEU
LLaMa2	Linearization		28.95	4.12	30.81	4.12	22.88	4.12
	LLaMa2	Frozen	24.63	10.38	31.80	10.38	33.22	10.38
		KG2T	24.77	39.26	27.32	39.26	19.26	39.26
		KG2P-SFT	24.49	9.29	33.20	5.89	34.85	9.67
		KG2P-DPO	30.72	6.71	39.58	2.74	39.88	6.53
	Mistral	Frozen	26.25	4.77	30.81	4.77	19.65	4.77
		KG2T	25.50	39.45	28.02	39.45	20.20	39.45
		KG2P-SFT	26.31	14.97	33.10	11.33	22.51	5.59
		KG2P-DPO	30.84	6.39	29.28	4.69	26.19	6.08
Mistral	Linearization		27.96	4.12	28.51	4.12	22.83	4.12
	LLaMa2	Frozen	25.73	10.38	29.31	10.38	24.14	10.38
		KG2T	25.56	39.26	22.63	39.26	19.34	39.26
		KG2P-SFT	25.52	9.14	31.01	9.14	26.69	7.81
		KG2P-DPO	32.32	10.22	37.59	3.28	32.93	5.19
	Mistral	Frozen	25.60	4.77	26.32	4.77	18.68	4.77
		KG2T	26.35	39.45	23.53	39.45	18.89	39.45
		KG2P-SFT	23.90	5.96	26.22	7.28	21.60	5.73
		KG2P-DPO	32.60	6.83	31.90	3.88	30.41	4.55
Pearson Correlation			-0.33		-0.62		-0.44	

The impact of KG Information Providing KG information generally enhanced QA task performance compared to scenarios without it. However, in the case of WebQSP and Mintaka, providing KG information occasionally resulted in performance declines. Notably, KG2P-DPO consistently outperformed across all cases when prompts were provided. This suggests that inappropriate forms of KG information can hinder performance, whereas our method effectively provides beneficial forms.

Triple Form vs Free Form Text Knowledge-augmented prompt using LLMs to produce free-form text is generally expected to outperform simple rule-based linearization, which generates triple-form text. However, the results indicate that this was not always the case. In certain instances with SimQ, WebQSP, and Mintaka, linearization demonstrated better performance compared to free-form text. This supports the findings of Baek et al. (2023a), which suggest that linearization is a simple yet effective method for knowledge-augmented prompting. Nevertheless, KG2P-DPO consistently generated superior prompts compared to linearization, indicating that our method effectively optimizes knowledge-augmented prompting for KGQA.

Impact of DPO Training The Frozen method

[5]https://gitlab.com/shimorina/webnlg-dataset/-/tree/master/release_v3.0/en

considers the query context when converting KG-to-Text. The primary difference lies in whether training is conducted using KG2P-SFT or KG2P-DPO. The performance of KG2P-SFT is not always superior to that of Frozen. However, the performance of KG2P-DPO consistently surpasses that of Frozen. When comparing the two specific strategies of KG2P, the DPO training method demonstrated better results than the SFT training method.

KG2T vs KG2P The Traditional KG-to-Text method involves converting a set of triples from a KG into natural language text, by training human-translated data. This method is expected to enhance readability, thus helping LLMs process KG information more effectively. However, whether the form of human labels is optimal for LLM processing remains uncertain. To investigate this, we measured the KG-to-Text performance on WebNLG test data Li et al. (2020) according to different prompt methods. Table 2 shows the results.

The KG2T, which is directly trained from human-translated labels, exhibited significantly higher BLEU scores compared to other prompting methods. However, KG2T generally underperforms in QA tasks compared to Linearization. This suggests that converting KG-to-Text by mimicking human-translated labels does not help LLMs process information effectively.

Furthermore, although KG2P-DPO achieves superior QA performance, its lower BLEU scores underscore the weak or negative correlation between KG-to-Text generation and QA performance as demonstrated by the Pearson correlation values in Table 2. This lack of correlation highlights that training for QA objectives and traditional KG-to-Text objectives are largely independent. Our findings suggest that optimizing for task-specific KG-to-Text methods, rather than strictly imitating human labels, offers a more promising approach for improving downstream task performance in LLMs.

4.3 Abliation Study of Query Context and Training for Knowledege Augmented Prompting

The two primary factors of our proposed method are 1) training aimed at optimizing knowledge-augmented prompting and 2) the conversion that takes into account the context of the query. We conducted additional experiments to analyze the impact on performance and their interaction. To assess the impact of these factors, we compared the performance of Frozen and KG2P-DPO as used in

Table 3: **Performance Comparison With and Without Query Context and KG2P Training.** 'w/o Q' means prompt method without considering query context. Frozen refers to the LLM in the pre-trained state without KG2P training. By comparing w/o Q and others, we show how query context affects the quality of prompts. Additionally, compare Frozen and KG2P to see the impact of our proposed training method.

Solver LLM	Prompting Method		Easy				Complex	
			SimQ		WebQSP		Mintaka	
			F1	EM	F1	EM	F1	EM
LLaMa2	LLaMa2	Frozen w/o Q	27.05	15.26	42.19	21.14	36.26	26.53
		Frozen	36.26	24.63	49.11	31.80	39.28	33.22
		KG2P w/o Q	37.09	26.05	48.32	30.81	39.40	32.17
		KG2P-DPO	41.21	30.72	57.51	39.58	46.19	39.88
	Mistral	Frozen w/o Q	35.26	24.53	45.46	29.21	28.38	22.70
		Frozen	36.99	26.25	45.88	30.81	25.80	19.65
		KG2P w/o Q	24.36	13.80	34.22	34.20	34.22	28.84
		KG2P-DPO	41.31	30.84	56.51	39.28	34.62	26.19
Mistral	LLaMa2	Frozen w/o Q	30.45	19.68	45.14	27.22	31.00	23.75
		Frozen	37.02	25.73	47.40	29.31	30.61	24.14
		KG2P w/o Q	18.35	9.49	45.11	27.92	30.97	26.08
		KG2P-DPO	42.36	32.32	57.52	37.59	38.76	32.93
	Mistral	Frozen w/o Q	33.40	23.90	42.35	26.02	28.54	22.88
		Frozen	34.77	25.60	41.96	26.32	25.48	18.68
		KG2P w/o Q	38.37	28.46	45.17	27.82	29.81	24.43
		KG2P-DPO	42.05	32.60	49.39	31.90	37.22	30.41

Table 4: **Analysis for Impact of Proposed Factors.** It shows the average EM scores for each case in Table 3. To measure the effect of query context and KG2P training, we calculated the ratio by dividing the results with each factor by the results without. We also compared the performance of KG2P-DPO and the performance of Frozen w/o Q to evaluate the collaborative results of the two factors.

	without KG2P Train	with KG2P Train	ΔKG2P Training
without Query Context	23.57	25.84	2.27 (10%)
with Query Context	26.35	33.69	7.34 (28%)
ΔQuery Context	2.78 (12%)	7.85 (30%)	10.12 (43%)

the §4.2 and along with additional baselines. The details are below:

- **Pre-trained LLM without Query Context (Frozen w/o Q)** As with the Frozen LLM in §4.2, we converted KG-to-Text form using the pre-trained LLM without additional training. And, during the converting process, we did not not consider query context.
- **KG2P-DPO without Query Context (KG2P w/o Q)** As KG2P-DPO in §4.2, we converted KG-to-Text from using our proposal method. However, during the converting process, we do not consider query context.

Table 3 presents the results of our ablation studies. Additionally, for quantitative analysis, Table 4 reports the average EM scores both when each factor is considered and when it is not. When training for knowledge-augmented prompting was not applied, incorporating the query context led to a

12% performance improvement, while the training combined with query context reflection resulted in a 30% improvement. This indicates that incorporating the query context is a significant positive factor in prompt generation.

Moreover, the proposed training method resulted in a 10% performance improvement without incorporating the query context, and a 28% improvement when the query context was incorporated. Compared to the case where neither factor was incorporated, incorporating both factors led to a 43% performance improvement. This demonstrates that our proposed training method effectively optimizes prompt generation for QA tasks. Furthermore, it shows that incorporating query context into the learning process results in positive interactions.

4.4 Experiments of Robustness

Our methodology aims to improve the performance of the Solver LLM on KGQA by generating prompts. If the Solver LLM is replaced with a different LLM, the optimization criteria change, potentially leading to a decline in QA performance and undermining the robustness of the proposed method. We measured the impact on QA performance when the Solver LLM was replaced with an alternative LLM. To quantify this, we calculate the ratio by dividing the swapped Solver LLM's performance by the same Solver LLM used in the train.

The results are shown in Table 5. When the Solver LLM was changed from LLaMa2 to Mistral, there was no significant performance change for SimQ and Mintaka. However, for WebQSP, this change resulted in a 16% performance decline. The overall range of the Ratio was from 0.84 to 1.04, indicating a slight performance drop as expected. When the Solver LLM was switched from Mistral to LLaMa2, performance varied from a maximum 9% decline to a maximum 14% improvement.

Despite some performance drops in both scenarios, our methodology still outperformed other baselines. Even with instances of performance decline, our approach consistently demonstrated superior performance compared to other baselines. The results show that our method is robust even when the solver LLM is substituted.

Table 5: **Results of Robustness Experiments.** It shows the change in EM for each data when the Solver LLM is swapped during testing. The ratio is calculated by dividing the performance when the Solver LLM is swapped by the performance when the Solver LLM remains the same during training and testing.

Solver LLM		KG2P-DPO (LLaMa2)			KG2P-DPO (Mistral)		
Train	Test	SimQ	WebQSP	Mintaka	SimQ	WebQSP	Mintaka
LLaMa2	LLaMa2	30.72	39.58	39.88	30.84	39.28	26.19
LLaMa2	Mistral	30.31	36.19	41.54	31.93	33.00	26.97
Ratio		**0.99**	**0.91**	**1.04**	**1.04**	**0.84**	**1.03**
Mistral	Mistral	32.32	37.59	32.93	32.60	31.90	30.41
Mistral	LLaMa2	30.90	39.28	30.07	29.82	36.29	29.49
Ratio		**0.96**	**1.04**	**0.91**	**0.91**	**1.14**	**0.97**

5 Conclusion

In this study, we propose the Knowledge Graph to Knowledge-Augmented Prompt (KG2P) framework, which effectively integrates LLMs with KGs. Existing KG-to-Text transformation methods are not optimized for enabling LLMs to effectively utilize KG information, as they perform independent transformations without considering the query. This often results in unnecessary information being included or critical information being omitted. To overcome these limitations, our study introduces two key techniques.

First, we employ DPO to refine knowledge-augmented prompts, automatically learning transformations that maximize QA performance. As a result, KG-to-Text transformation is not merely a replication of human annotation but is fine-tuned to enable LLMs to generate more accurate responses. Second, we apply query-aware transformation, ensuring that KG information is tailored to the given query. This optimization allows LLMs to leverage KG information more effectively in QA tasks.

Experiments conducted on KGQA benchmark datasets demonstrate that KG2P consistently outperforms existing KG-to-Text methods. Notably, KG2P generates query-relevant information more effectively than simple triple linearization or traditional KG-to-Text models, providing better prompts for Solver LLMs to derive correct answers. Furthermore, cross-model evaluation with different LLMs, including LLaMa2 and Mistral, confirmed KG2P's robustness and adaptability across various LLM environments.

This study makes a contribution by introducing a new paradigm for integrating LLMs with structured knowledge. While prior KG-to-Text approaches rely on static, human-generated data, our proposed framework directly optimizes LLM response performance.

6 Limitations

Our approach transcends the boundaries of KGQA and offers potential applicability to a range of studies involving the integration of KG with LM. Nonetheless, this investigation remains focused on QA, leaving the exploration of its wider applicability for future research.

It is generally observed that LM with more parameters tend to exhibit better QA performance. Due to resource limitations, our study is unable to explore a wide range of models. We employ models that have 7 billion parameters and enhance their performance using LoRA and quantization techniques. Future investigations into the performance of more advanced LM may potentially yield even better results.

Additionally, various alignment tuning techniques such as Identity Preference Optimisation (IPO) and Kahneman-Tversky Optimization (KTO) could be applied to our study alongside DPO (Azar et al., 2024; Ethayarajh et al., 2024). However, a key limitation of our work is that we did not conduct experiments with different methods and only evaluated DPO.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023a. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Jinheon Baek, Soyeong Jeong, Minki Kang, Jong C Park, and Sung Ju Hwang. 2023b. Knowledge-augmented language model verification. *arXiv preprint arXiv:2310.12836*.

Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer. 2019. Introduction to neural network based approaches for question answering over knowledge graphs. *arXiv preprint arXiv:1907.09361*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Dennis Diefenbach, Thomas Pellissier Tanon, Kamal Deep Singh, and Pierre Maret. 2017. Question answering benchmarks for wikidata. In *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017.*

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. *arXiv preprint arXiv:2007.13069*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Wooyoung Kim, Haemin Jung, and Wooju Kim. 2025a. Knowledge graph as pre-training corpus for structural reasoning via multi-hop linearization. *IEEE Access*, 13:7273–7283.

Wooyoung Kim, Byungyoon Park, and Wooju Kim. 2025b. Query-aware learnable graph pooling tokens as prompt for large language models. *arXiv preprint arXiv:2501.17549*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xintong Li, Aleksandre Maskharashvili, Symon Jory Stevens-Guille, and Michael White. 2020. Leveraging large pretrained models for WebNLG 2020. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 117–124, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194, Brussels, Belgium. Association for Computational Linguistics.

Fedor Moiseev, Zhe Dong, Enrique Alfonseca, and Martin Jaggi. 2022. Skill: Structured knowledge infusion for large language models. *arXiv preprint arXiv:2205.08184*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Nils Reimers and Iryna Gurevych. 2019. 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.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. 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*, pages 4498–4507.

Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1604–1619, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Daniil Sorokin and Iryna Gurevych. 2018. Modeling semantics with gated graph neural networks for knowledge base question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3306–3317. Association for Computational Linguistics.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium. Association for Computational Linguistics.

I Sutskever. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. In *International Conference for Learning Representation (ICLR)*.

A Details of Prompt Template

KG2P Tempalte (T_P)
Please create a short hint paragraph to answer the question reorganizing the triple information, step by step.
Question: *{question q}*
Triple Information: *{triples* $\mathcal{T}_1^q, \cdots, \mathcal{T}_n^q$*}*
Hint Paragraph:

QA Template (T_{QA})
Below are the facts that might be relevant to answer the question. Please provide a short answer(1-3 words in English) to the following question.
Facts: *{Knowledge Augmented Prompt* P*}*
Question: *{question q}*
Answer:

B Implementation Details

B.1 QA Dataset Preprocessing

In this study, we used Wiki5M, a subset of Wikidata, as our knowledge base. The KGQA datasets used in our experiments are based on the entirety of Wikidata, which includes questions that cannot be answered within the scope of Wiki5M. Therefore, questions whose answer entities are not included in Wiki5M were excluded from the experiments. The specific number of excluded questions is shown in Table 6.

Table 6: **Number of Questions for Each Dataset** Original refers to the size provided by the original dataset. Answerable means the count of questions whose answer entities are included in Wiki5M.

	Simple Question		WebQSP-WD		Mintaka	
	Train	Test	Train	Test	Train	Test
Original	19481	5622	3098	1033	14000	4000
Answerable	17700	5071	2763	1003	13275	3811

B.2 Retrieval System

All triples in the Wiki5M dataset were converted into text form using the linearization method, and their vectors were generated using a text embedding model. These vectors were indexed in a triple vector repository denoted as DB. Upon receiving a question, the text embedding model produced an embedding vector for the question z_q, which was used to retrieve relevant triples through Maximum Inner Product Search (MIPS).

Specifically, the argtopk operation searches the k triples based on high inner product scores in the DB. We employed a pre-trained SentenceTransformer[6] (Reimers and Gurevych, 2019) as our text embedding model, and leveraged Faiss (Johnson et al., 2019) to efficiently perform the MIPS operations.

$$\{\mathcal{T}_1^q, \cdots, \mathcal{T}_k^q\} = \text{argtopk}_{z_i \in \text{DB}} \text{InnerProduc}(z_q, z_i) \tag{4}$$

B.3 Computational Resources and Training Details

We utilized an Nvidia RTX 3090 24G GPU. To efficiently use computational resources, we applied 4-bit NormalFloat (NF4) quantization and Low-Rank Adaptation (LoRA) during the training process, and NF4 quantization was also used during the inference process (Hu et al., 2021; Dettmers et al., 2024). Both SFT and DPO[7] were trained for 2 epochs on each dataset with a mini-batch size of 2. For optimization, we used Adafactor (Shazeer and Stern, 2018) with a learning rate of 1e-5.

C Case Study

In §4.2, there are instances where performance decreased despite the provision of KG information. Baek et al. (2023b) interpreted this phenomenon by categorizing two types of errors: *retrieval error*, where irrelevant knowledge is retrieved for a query, and *grounding error*, where an LLM fails to generate the correct answer despite relevant knowledge is provided. Specifically, a retrieval error occurs when the retrieved results do not contain the correct answer entity, while a grounding error occurs when the LLM generates an incorrect answer despite the presence of the answer in the given KG information. We conducted a detailed analysis by categorizing cases into *Without Retrieval Error* and *With Retrieval Error* based on the correct labels for each dataset.

The role of the prompt generator differs depending on the case. In the *Without Retrieval Error* case, the primary role of the prompt generator is to reduce the *grounding error*. Therefore, the prompt generator should transform triples into free-form text that is more easily processed by LLMs. Conversely, in the case *With Retrieval Error*, the given triples are insufficient to generate correct answers. The primary objective of the prompt generator is to extract information related to the question from its parametric knowledge, which is acquired during the pre-training process, to compensate for this insufficiency.

Table 7 presents a case study comparing scenarios with and without Retrieval Error. In the *Without Retrieval Error* case, it is crucial to highlight the relevant triples and eliminate the noise triples that are unnecessary for deriving the correct answer. Other baselines tend to emphasize irrelevant information or overly broad details, such as genres. In contrast, KG2P-DPO accurately highlights the necessary information, aiding the Solver LLM in generating the correct answer.

In the *With Retrieval Error* case, there is insufficient information in retrieval results to derive the correct answer, necessitating the generation of new information using parametric knowledge. The most critical factor in this process is generating relevant information without hallucination. Linearization and KG2T failed to generate new information, resulting in the Solver LLM's inability to produce the correct answer. Frozen and KG2P-SFT generated information but exhibited hallucinations by producing incorrect information. However, KG2P-DPO generated accurate additional information, which enabled the Solver LLM to derive the correct answer.

Yu et al. (2023) demonstrates that leveraging the parametric knowledge inherent in LLMs can lead to the creation of prompt generator enhancing QA task performance. Therefore, KG2P-DPO not only effectively utilizes retrieval results but also enriches answers with additional, relevant information extracted from the vast parametric knowledge in LLMs, demonstrating a robust method for improving accuracy in KGQA tasks.

[6]https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2

[7]Hugging Face Trainer was used for training.
https://huggingface.co/docs/trl/sft_trainer
https://huggingface.co/docs/trl/dpo_trainer

Table 7: **Example of Generated Knowledge Augmented Prompts.** The example illustrates how KG2P-DPO addresses Grounding Error and Retrieval Error. In the case without Retrieval Error, KG2P-DPO highlights the essential knowledge needed to answer the question, thereby eliciting the correct answer. Moreover, in the case with Retrieval Error, KG2P-DPO generates new knowledge to answer the question using parametric knowledge.

	[Without Retrieval Error] Question: What music is in espion, lève-toi? Correct Answer: Ennio Morricone	
Prompting Method	Knowledge-Augmented Prompt	Answer
Linearization	“(espion, lève-toi, instance of, movie)”, “(espion, lève-toi, composer, ennio morricone)”, “(Esra Dalfidan, genre, jazz music)”, “(Cassiane, genre, Devotional music)”, “(espion, lève-toi, genre, Spy movie)”	Jazz
Frozen	The music piece “espion, lève-toi” is featured in a spy movie. Ennio Morricone composed it, and it belongs to the genre of spy music. **(the rest omitted)**	Spy
KG2T	Espion, Lève-Toi is a spy movie with Ennio Morricone as its composer. It is performed in jazz and devotional music. Cassiane performs devotional music. Esra Dalfidan’s genre is jazz music.	Jazz
KG2P-SFT	The music you are looking for is “L’Impero” from the 1965 Italian spy film “Operazione San Gennaro.” This score was composed by Ennio Morricone and features a distinctive blend of jazz and devotional music that adds depth and complexity to the film’s action scenes. **(the rest omitted)**	Jazz
KG2P-DPO	Based on the triple information provided, it can be inferred that the music in “Espion, lève-toi” is Ennio Morricone’s score. This talented Italian composer has created numerous iconic soundtracks for films, including this spy movie. **(the rest omitted)**	Ennio Morricone
	[With Retrieval Error] Question: What airport do you fly into to get to destin fl? Correct Answer: Destin-Fort Walton Beach Airport	
Prompting Method	Knowledge-Augmented Prompt	Answer
Linearization	“(Davin Lake Airport, country, p:ca)”, “(michelle dusan, start point, Miami Airport, Florida)”, “(northwest florida beaches international airport, place served by transport hub, un/locode:uspfn)”, “(via742, destination point, Miami Airport, Florida)”, “(mário lopes leite state airport, country, Brzil)”	MIA
Frozen	To get to Destin, Florida from Davin Lake Airport in Canada, you will need to fly into Northwest Florida Beaches International Airport, which is located near Panama City, Florida. From there, you can take a taxi or rental car to Miami Airport, where you can catch a flight to Miami, and then another flight to Destin. **(the rest omitted)**	Miami Airport
KG2T	The airport serving Destin is Northwest Florida Beaches International Airport which has an Un/Locode of USPFN. Michelle Dusan started at Miami Airport in Florida and via 742 she arrived there. Davin Lake Airport is located in Brazil.	USPFN
KG2P-SFT	To get to Destin, Florida from Davin Lake Airport in Canada, you will need to fly into Northwest Florida Beaches International Airport or Pensacola International Airport, both located in Florida. From there, you can take a taxi or rental car to your final destination in Destin. **(the rest omitted)**	Northwest Florida Beaches International Airport (ECP)
KG2P-DPO	To get to Destin, Florida from your starting point of Miami Airport, you will fly into Destin-Fort Walton Beach Airport, which is located in the United States and serves as a major transportation hub for the region. **(the rest omitted)**	Destin-Fort Walton Beach Airport