Putting People in LLMs' Shoes: Generating Better Answers via Question Rewriter

Junhao Chen, Bowen Wang*, Zhouqiang Jiang, Yuta Nakashima

Osaka University, Japan
{junhao, zhouqiang}@is.ids.osaka-u.ac.jp, {wang, n-yuta}@ids.osaka-u.ac.jp

Abstract

Large Language Models (LLMs) have demonstrated significant capabilities, particularly in the domain of question answering (QA). However, their effectiveness in QA is often undermined by the vagueness of user questions. To address this issue, we introduce single-round instance-level prompt optimization, referred to as question rewriter. By enhancing the intelligibility of human questions for black-box LLMs, our question rewriter improves the quality of generated answers. The rewriter is optimized using direct preference optimization based on feedback collected from automatic criteria for evaluating generated answers; therefore, its training does not require costly human annotations. The experiments across multiple black-box LLMs and long-form question answering (LFQA) datasets demonstrate the efficacy of our method. This paper provides a practical framework for training question rewriters and sets a precedent for future explorations in prompt optimization within LFQA tasks.

Code — https://github.com/3244we/Question-Rewriter
Extended version — https://arxiv.org/abs/2408.10573

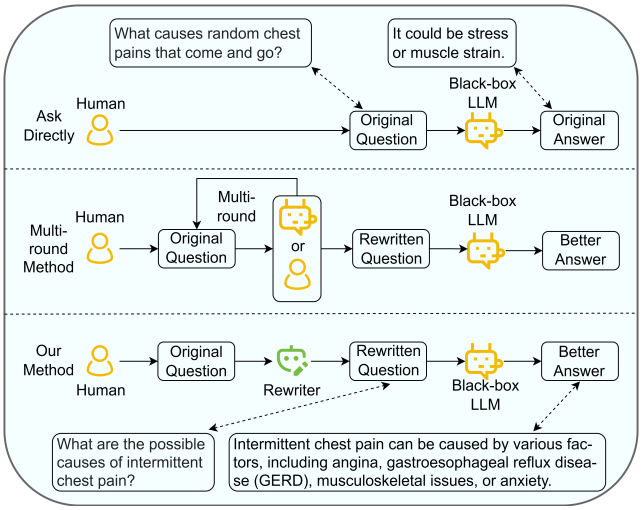


Figure 1: The original questions posed by the user are difficult for black-box LLMs to understand, resulting in poor answers. However, when the questions are rewritten by the rewriter, they become easier for LLMs to comprehend, leading to better answers.

Introduction

Large language models (LLMs) have incorporated extensive world knowledge through learning vast publicly available corpora (Roberts, Raffel, and Shazeer 2020). It becomes increasingly common for people to seek knowledge from LLMs, especially in fields such as medicine and law (Atallah et al. 2023; Harrington 2023; Wang et al. 2024). However, a near-paradoxical issue arises: *People ask questions to get knowledge, while lack of knowledge often leads to poorly formulated or vague questions, hindering LLMs from providing precise answers* (Kim et al. 2023; Zhang et al. 2024). Fine-tuning can enhance LLMs' ability to understand vague questions, but most popular LLMs are black-box models, and their parameters are inaccessible. Thus, a step of transforming user questions into a format that LLMs can understand better, known as question rewriting, is crucial for question answering (QA).

Question rewriting is closely related to prompt optimization. A prompt is an input to LLMs that guides them in generating a specific response, including a question (possibly with some instructions), a conversation history, etc. (Liu et al. 2023). Question rewriting is prompt optimization solely for questions. Previous work on prompt optimization primarily focused on optimizing *task-level* prompts. They decompose prompts into task-level instructions and *instance-level* inputs, optimizing task-level instructions for better performance across all instances of the task (Fernando et al. 2023; Guo et al. 2023; Kong et al. 2024).

Recent studies have shown that directly optimizing the prompts at the instance level offers more flexibility in prompt editing tailored for a given prompt(Lin et al. 2024) and can lead to better responses (Srivastava et al. 2023). By obtaining feedback from humans or LLMs, they iteratively refine a given prompt, which requires multi-round interactions. In addition, previous prompt optimization is mainly applied to arithmetic reasoning (Cobbe et al. 2021) and short-form question answering (SFQA) (Kwiatkowski et al. 2019), where the latter involves answers in few words.

*Corresponding author.

These tasks do not necessarily cover real-world QA scenarios.

This paper proposes single-round instance-level prompt optimization, referred to as *question rewriter*, aiming at optimizing questions for long-form question answering (LFQA), which is closer to real-world QA scenarios (Bhat et al. 2023). The question rewriter serves as an intermediary between users and black-box LLMs to rewrite questions, as shown in Figure 1. When a user submits a question, our question rewriter scutches it to contextualize the question for black-box LLMs to generate a more accurate answer.

The key to our method lies in obtaining supervising signals for optimizing the question rewriter. Different from arithmetic reasoning and SFQA, there is no unique best answer for LFQA questions; therefore, obtaining the optimal rewritten question as the ground truth to train a question rewriter is not trivial (Radford and Narasimhan 2018). Our method, in contrast, assumes the presence of automatic criteria to evaluate generated answers, which are typically provided in LFQA datasets, and uses them to identify better and worse rewritten questions. With such supervising signals, we propose to use direct preference optimization (DPO) (Rafailov et al. 2023) to train our question rewriter. Thanks to this design choice, our method does not necessitate costly human interactions used in reinforcement learning from human feedback (Bai et al. 2022) and a differentiable reward model in proximal policy optimization (Schulman et al. 2017).

Contribution. Our question rewriter is single-round prompt optimization without human interventions, which has not been explored so far. We experimentally show across multiple datasets and LLMs for answer generation that, with optimization by automatic evaluation criteria, the question rewriter can generate questions that end up with better answers. Intiguingly, our analysis implies that the question rewriter learns to generate non-leading and concise questions in a professional tone, which aligns with our intuitions when engineering a prompt.

Related Work

Early work for prompt optimization focused on white-box models (Shin et al. 2020; Shi et al. 2023; Li and Liang 2021; Lester, Al-Rfou, and Constant 2021; Zhong, Friedman, and Chen 2021). Due to the prevalent nature of black-box models such as GPT-3 (Patel et al. 2023) and Claude (Anthropic 2024), the following work targeted at these black-box models. Most works decomposed prompts into task-level (i.e., instructions) and instance-level (i.e., specific queries) and optimizing only task-level instructions. Some work assumed that input (i.e., text embeddings) and output (i.e., logits) are accessible and leveraged them to optimize prompts (Sun et al. 2022b,a; Chai et al. 2022). Other recent work has attempted to remove this assumption. Prasad et al. (2023) and Pryzant et al. (2023) evaluate task-level instructions with small edits (e.g., replacing some phrases with their synonyms) and find better ones step by step. Evolutionary algorithms (Fernando et al. 2023; Guo et al. 2023), reinforce learning (Diao et al. 2023; Kong et al. 2024), and planning-based methods (Wang et al. 2023) have also been adopted.

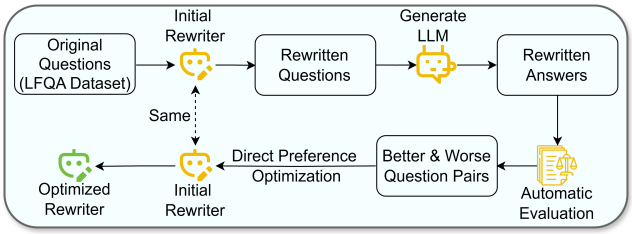


Figure 2: Pipeline of our method.

Some work fully utilized the inherent capabilities of LLMs to refine prompts. Zhou et al. (2023) leverages an LLM to generate and refine the prompts iteratively, and Yang et al. (2023) provides the prompt optimization trajectory to an LLM, allowing for discovering inherent patterns and optimizing the prompt progressively. Other notable efforts, such as InstructZero (Chen et al. 2023) and INSTINCT (Lin et al. 2023), have transformed black-box optimization into an iterative optimization problem with white-box LLMs.

All these works are task-level prompt optimization. Recent studies have shown that instance-level prompt optimization results in better performance by offering more specific prompts (Lin et al. 2024; Srivastava et al. 2023). These works iteratively refine a prompt at the instance level by obtaining feedback from humans or ChatGPT. We also adopt the instance-level approach, but unlike the other instance-level method, ours does not require feedback from LLMs or humans for multi-round optimization. We instead use preference optimization (Rafailov et al. 2023) for our question rewriter that can optimize the questions at once without any feedback nor iterative refinement. We evaluate our method over LFQA tasks, whereas previous works mainly use arithmetic reasoning (Cobbe et al. 2021) and SFQA (Kwiatkowski et al. 2019). LFQA is closer to real-world QA scenarios (Bhat et al. 2023) and can highlight differences in the generated text.

Method

Our question rewriter R learns to rewrite questions so that an LLM can give a better answer for a rewritten question. We design our method under the assumption that the goodness of the answer to a certain question is automatically judgeable. With this assumption, we can sample rewritten questions and the corresponding answers using LLMs, contrasting them to learn desirable questions.

Figure 2 shows the pipeline of our method. Let $\mathcal{D} = \{(q, a)\}$ denote a training dataset of pairs of question q and answer a, with an associated set $\mathcal{C} = \{c\}$ of automatic evaluation criteria c. Firstly, our pipeline rewrites questions for $q \in \mathcal{D}$. Then, $c \in \mathcal{C}$ evaluates the rewritten questions to make a set $\mathcal{P} = \{(\hat{q}, \check{q})\}$ of pairs of a better question $\hat{q}$ and a worse question $\check{q}$. Finally, we use DPO (Rafailov et al. 2023) to train R with $\mathcal{P}$.

Sampling Rewritten Questions

We use a pre-trained LLM R_0 to sample rewritten questions without fine-tuning as it offers sufficient capability for ini-

tial rewriting solely through prompt engineering. We use top-p sampling (Radford et al. 2019) to generate K different rewritten questions $\mathcal{Q}(q) = \{r_k(q)|k = 1, \ldots, K\}$ of $q \in \mathcal{D}$, where $r_k(q)$ is the k-th rewritten question for q, with the predefined prompt t, i.e., t equals:

```
Rewriting question to make it more
understandable, just give me the
rewritten question without any other
word:
```

followed by q.

Making Better and Worse Question Pairs

Datasets for LFQA typically provide methods for evaluating generated answers. For instance, some datasets (Manes et al. 2024) are inspired by FActScore (Min et al. 2023) and annotate the facts required to answer each question, allowing LLMs to assess whether the corresponding facts are implied by or contradict the generated answers to derive scores for comprehensiveness and precision. Other datasets (Lin, Hilton, and Evans 2022) offer extensive binary annotations used to train classifiers to determine whether answers conform to certain attributes like truthfulness. Additionally, some datasets[1] are in the form of preference datasets, which provide pairs of samples, where one is better than the other. Such datasets can be used to train reward models to evaluate whether answers align with human preferences. We can use these automatic evaluation criteria as $\mathcal{C}$ to evaluate rewritten questions. Such automatic evaluation criteria substitute the human feedback typically used in previous methods (Rafailov et al. 2023) to make $\mathcal{P}$.

Let L denote a pre-trained LLM for answer generation. For a question-answer pair $(q, a) \in \mathcal{D}$, we generate answers for all $q' \in \mathcal{Q}(q)$ as $a' = L(q')$. We also generate the answer to the original question q as $\tilde{a} = L(q)$, which serves as the baseline to judge the goodness of rewritten questions.

To make better-worse pairs, we first identify $q' \in \mathcal{Q}(q)$ that gives better answers and worse answers, collectively denoted by $\mathcal{Q}_+(q)$ and $\mathcal{Q}_-(q)$, respectively. Observing that criterion $c \in \mathcal{C}$ is often numerical[2], we judge q' is better if a' is larger than or equal to $\tilde{a}$ in terms of *all* criteria and a' is larger than $\tilde{a}$ at least one criterion,[3] i.e.,

$$\mathcal{Q}_+(q) = \{q' \in \mathcal{Q}(q) | \forall_{c \in \mathcal{C}}\, c(a') \geq c(\tilde{a}), \exists_{c \in \mathcal{C}}\, c(a') > c(\tilde{a})\}. \quad (1)$$

$\mathcal{Q}_-(q)$ is defined in the opposite way, i.e., a' should be always worse than or equal to $\tilde{a}$ and a' should be worse than $\tilde{a}$ for at least one criterion.

A better and worse question pair is created by picking one rewritten question from $\mathcal{Q}_+(q)$ and the other from $\mathcal{Q}_-(q)$. As we wish to train a model R to generate good questions, we rank rewritten questions in $\mathcal{Q}_+$ according to a certain composition of all $c \in \mathcal{C}$,[4] and use the top N_+ questions. The set of chosen better questions is denoted by $\mathcal{Q}^\star_+(q)$. On the other hand, to avoid following DPO training only with easy negatives, we randomly choose N_- questions in $\mathcal{Q}_-(q)$ and pair each of them with $\hat{q} \in \mathcal{Q}^\star_+(q)$. Formally, letting $\mathcal{S}$ denote randomly sampled N_- questions from $\mathcal{Q}_-(q)$ without replacement, the set $\mathcal{P}(q)$ of better and worse question pairs for q is given by:

$$\mathcal{P}(q) = \{(\hat{q}, \check{q}) | \hat{q} \in \mathcal{Q}^\star_+(q), \check{q} \in \mathcal{S}(\mathcal{Q}_-(q))\}. \quad (2)$$

$\mathcal{P}(q)$ contains $N_+ \times N_-$ pairs when $|\mathcal{Q}_+(q)| \geq N_+$ and $|\mathcal{Q}_-(q)| \geq N_-$; otherwise, $|\mathcal{P}(q)|$ is smaller. The comparison of the different sampling combination[5] for $\mathcal{P}(q)$ can be found in the extended version appendix.

Optimizing Question Rewriter

Training our question rewriter R is costly when it requires human feedback or a reward model that learns the human feedback. Fortunately, LFQA tasks typically offer automatic criteria to evaluate the goodness of generated answers. We can use the criteria to (indirectly) evaluate rewritten questions by evaluating their answers.

Let $P_R(q'|t, q)$ denote the average probability of tokens in q' given the predefined prompt t and the original question q with R, given by:

$$P_R(q') = \frac{1}{T} \sum_{k=1}^{K} p_R(w_k | t, q, w_{1:k-1}), \quad (3)$$

where K is the length of q'; $p_R(w_t|t, q, w_{1:k-1})$ is the probability of token w_t given t, q, and a set $w_{1:k-1}$ of tokens generated by the $(k-1)$-th step (i.e., $q' = w_{1:K}$). $P_{R_0}(q')$ is defined likewise for the initial question rewriter R_0. DPO's training loss is given by:

$$L = -\mathbb{E}\left[\log \sigma \left(\beta \log \frac{P_R(\hat{q})}{P_{R_0}(\hat{q})} - \beta \log \frac{P_R(\check{q})}{P_{R_0}(\check{q})}\right)\right] \quad (4)$$

where σ is sigmoid, and β is a hyperparameter that controls how much R deviates from R_0, and the expectation is computed over $q \sim \mathcal{D}$ and $(\hat{q}, \check{q}) \sim \mathcal{P}(q)$.

To mitigate the risk of overfitting, we use dropout in the model. Also, the original LFQA dataset is divided into three parts: training, validation, and testing. R is trained on the training set (i.e., $\mathcal{D}$) for one epoch, and we select the best model that most prefer $\hat{q}$'s to $\check{q}$'s. Specifically, we define preference score PS as

$$\text{PS} = \mathbb{E}\left[\mathbf{1}[P_R(\hat{q}|t, q) > P_R(\check{q}|t, q)]\right], \quad (5)$$

where $\mathbf{1}[\cdot]$ gives 1 if the given condition is satisfied, and otherwise 0; the expectation is computed for all the q from the validation set and $(\hat{q}, \check{q}) \sim \mathcal{P}(q)$.

[1] https://huggingface.co/datasets/tasksource/oasst1_pairwise_rlhf_reward

[2] For example, $c(a')$ may be the probability of generated answer a' is truthful.

[3] It should be noted that a criterion may not always be *larger-is-better*. For *smaller-is-better* criteria, we just replace "larger than" with "smaller than" (and so in the inequalities).

[4] For example, if all $c \in \mathcal{C}$ are a *larger-is-better* type of criteria, the product of all criteria $\prod_c c(a')$ can be used. We design the composition for the set of criteria that a respective dataset provides.

[5] For instance, $\mathcal{P}(q) = \{(\hat{q}, \check{q}) | \hat{q} \in \mathcal{S}(\mathcal{Q}_+(q)), \check{q} \in \mathcal{Q}^\star_-(q)\}$, where $\mathcal{Q}^\star_-(q)$ denotes the set of bottom N_- questions.

Dataset	Training	Validation	Testing	Total
K-QA	101	50	50	201
TruthfulQA	407	205	205	817
OASST1QA	1,000	93	93	1,186

Table 1: Statistics of LFQA datasets used to evaluate our method. Columns for **Training**, **Validation**, and **Testing** give the numbers of samples in respective dataset splits.

Experiments

Experimental Setup

Dataset We evaluate three distinct LFQA datasets, each equipped with automated evaluation criteria.

K-QA (Manes et al. 2024), sourced from the medical domain, is designed to evaluate the factual comprehensiveness and precision of answers through metrics S_{comp} and S_{cont}, employing a FActScore type method (Min et al. 2023). To combine these two criteria for ranking rewritten questions in $\mathcal{Q} + (q)$, we first use S_{cont} to rank them, and then use S_{comp} if S_{cont} is the same for multiple questions.

TruthfulQA (Lin, Hilton, and Evans 2022), covering multiple domains including health and law, assesses the truthfulness (S_{truth}) and informativeness (S_{info}) of answers. The evaluation criteria are implemented as binary classifiers. We use the probabilities for positive classes (*truthful* for S_{truth} and *informative* for S_{info}). An overall score (S_{overall}) is computed as the product of these scores. For better rewritten pair ranking, we use S_{overall}.

OASST1QA, derived from the multi-turn dialogue alignment dataset OASST1[6], incorporates a criterion S_{pref} that measures human preference for answers using a pre-trained reward model. This dataset provides a single criterion for evaluation (i.e., $|\mathcal{C}| = 1$), so we directly use S_{pref} for ranking better rewritten questions.

More details about these datasets and their evaluation criteria can be found in the extended version appendix. Table 1 summarizes the statistics on the datasets.

LLMs The base model of our question rewriter R (and R_0 is Llama3-8B-instruct, and the answer generation model L is also Llama3-8B-instruct because it is one of the most powerful but small LLMs. R is fine-tuned with our method, while L is frozen. Subsequently, we evaluate the generalizability of R on multiple answer generation LLMs, including Llama3-8B-instruct[7], mistral-7B-instruct-v0.2[8], zephyr-7B-beta[9], gemma-1.1-7B-it[10], gpt-3.5-turbo-1106, and gpt-4o-2024-05-13. They will be referred to as Llama3-8B, Mistral-7B-v0.2, Zephyr-7B-beta, Gemma-1.1-7B, GPT-3.5, and GPT-4o, respectively. It is worth noting that we only use L as Llama3-8B-instruct to build P for training R, and then test the generalizability of R on other models.

[6]https://huggingface.co/datasets/OpenAssistant/oasst1

[7]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[8]https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

[9]https://huggingface.co/HuggingFaceH4/zephyr-7b-beta

[10]https://huggingface.co/google/gemma-1.1-7b-it

Hyperparameters We borrowed open-source code for DPO training over all three datasets[11], which also provides the code for supervised fine-tuning of automatic criteria S_{truth} and S_{info} for TruthfulQA. During DPO training, we set the dropout rate to 0.8, the training batch size to 32, and the testing batch size to 64, maintaining all other parameters at their default settings in the source code. For sampling rewritten questions, we use top-p sampling, where the cumulative probability for top-p sampling is set to 0.999, and the temperature of R_0 is 1, to ensure diversity. We sample 100 unique rewritten questions for each of the original questions and terminate the sampling after 10,000 attempts. N_+ and N_- are defaulted to (10, 20), (5, 10), and (4, 5) in K-QA, TQA, and OQA respectively. When multiplied by the number of samples in the corresponding training sets, they are around 20,000. The maximum token length is set to 512 during feedback collection and testing. During testing, to ensure reproducibility, we generate answers using greedy sampling.

Device All our testing and training, except for the DPO training of OASST1QA, are conducted on a system equipped with four NVIDIA A100-40GB-PCIE. Due to the extensive length of OASST1QA, we only used samples whose question plus the prompt t and rewritten questions q' for question rewriting is less than or equal to 512 tokens and conducted the DPO training on a system with four NVIDIA A100-80GB-PCIe.

Baselines In our experiments across different datasets and models, we compare our method with both the original questions and the initial Llama3-8B-instruct rewriter (without fine-tuning). To demonstrate the effectiveness of our approach, we also compare it with the widely used task-level prompting method, Zero-Shot Chain-of-Thought (Zero-Shot CoT) (Kojima et al. 2022). Other instance-level methods, such as PRoMPTed, require multiple rounds of interactions with humans or LLMs to obtain feedback and iteratively modify the prompt or question during inference, which is extremely costly in our QA scenarios. Therefore, we only perform comparisons with PRoMPTed and other question rewriting methods on the K-QA dataset and Llama3-8B-instruct.

Result across Models and Datasets

Table 2 summarizes our experimental results over three LFQA datasets. Our method demonstrates superior performance in most combinations of LLMs and datasets.

For the K-QA dataset, our method consistently shows the highest S_{comp} scores across all models, especially with GPT-4o, where the improvement is most significant. Furthermore, it achieves the lowest S_{cont} with half of the models and the second lowest in the rest. Notably, our method trained an effective question rewriter using only 151 samples (i.e., training set plus validation set), implying that our method requires only a small number of annotated samples in a real-world QA scenario. Table 3 shows an example from K-QA,

[11]https://github.com/eric-mitchell/direct-preference-optimization

Model	Method	K-QA		TruthfulQA			OASST1QA
		$S_{\text{comp}} \uparrow$	$S_{\text{cont}} \downarrow$	$S_{\text{truth}} \uparrow$	$S_{\text{info}} \uparrow$	$S_{\text{overall}} \uparrow$	$S_{\text{pref}} \uparrow$
Llama-3-8B	Original	0.4573	0.4400	0.7683	0.9664	0.7397	0.8654
	Zero-Shot CoT	0.4579	**0.4000**	0.7476	0.9306	0.6938	0.8838
	Initial Rewriter	0.4262	0.5000	0.7914	0.9564	0.7566	0.8748
	Ours	**0.4600**	**0.4000**	**0.8059**	**0.9668**	**0.7789**	**0.9104**
Mistral-7B-v0.2	Original	0.4374	**0.2200**	0.8364	**0.9834**	0.8227	0.8281
	Zero-Shot CoT	0.4428	0.2800	0.8423	0.9737	0.8199	**0.8908**
	Initial Rewriter	0.4177	0.3400	0.7916	0.9689	0.7670	0.8381
	Ours	**0.4899**	0.2600	**0.8474**	0.9788	**0.8296**	0.8762
Zephyr-7B-beta	Original	0.4396	0.3400	0.7644	**0.9826**	0.7518	0.6369
	Zero-Shot CoT	0.4333	0.3200	0.7081	0.9705	0.6867	0.7606
	Initial Rewriter	0.4666	**0.2200**	0.7353	0.9723	0.7167	0.6417
	Ours	**0.4702**	0.2600	**0.7709**	0.9775	**0.7528**	**0.7768**
Gemma-1.1-7B	Original	0.4010	0.5400	0.6780	**0.9716**	0.6554	0.7428
	Zero-Shot CoT	0.4516	0.5000	0.7216	0.9454	0.6752	0.8632
	Initial Rewriter	0.4928	0.4400	0.6415	0.9617	0.6124	0.7955
	Ours	**0.4956**	**0.2200**	**0.7224**	0.9558	**0.6888**	**0.9034**
GPT-3.5-turbo	Original	0.4909	0.3200	0.7451	**0.9804**	0.7303	0.7294
	Zero-Shot CoT	0.4748	**0.1600**	0.7413	0.9781	0.7237	0.8222
	Initial Rewriter	0.4454	0.3000	0.7325	0.9768	0.7164	0.7353
	Ours	**0.4978**	0.2800	**0.7574**	0.9682	**0.7309**	**0.8994**
GPT-4o	Original	0.5167	0.2800	0.8812	**0.9790**	0.8631	0.8532
	Zero-Shot CoT	0.4903	0.3000	0.8739	0.9611	0.8400	0.8471
	Initial Rewriter	0.5255	**0.2400**	0.8593	0.9683	0.8329	0.8461
	Ours	**0.6253**	**0.2400**	**0.8880**	0.9722	**0.8641**	**0.9100**

Table 2: Comparison of different question rewriting methods across multiple datasets and LLMs for answer generation. We use automatic evaluation criteria associated with each dataset. **Bold** indicates the best method. Underline indicates the second-best.

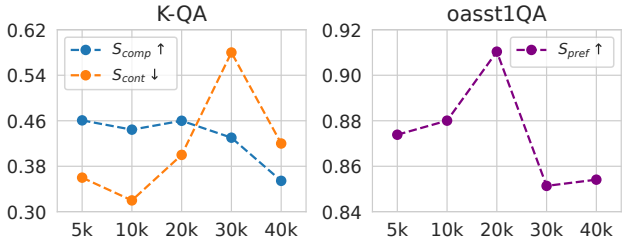


Figure 3: Evaluating the impact of N_+ and N_- on the performance over K-QA and OASST1QA.

in which S_{comp} increases and S_{cont} decreases after rewriting the original question.

For the TruthfulQA dataset, all methods generally reduce the informativeness (i.e., S_{info}) of the answers, and only ours gains the truthfulness score (i.e., S_{truth}). This is typical behavior in the TruthfulQA dataset as these two criteria exhibit a trade-off relationship (Lin, Hilton, and Evans 2022). Our method can increase S_{overall}, while the others reduce it.

For OASST1QA, our method outperforms others except Mistral-7B-v0.2, where Zero-Shot CoT performs best.

Overall, our method excels in all metrics and all datasets, not only on the Llama3-8B-instruct used for training R but also on the other LLMs, demonstrating their generalizability to answer generation models. Thanks to this generalizability, the rewriter R can be deployed without adjusting to individual (black-box) answer generation models.

Impact of N_+ and N_-

The number of better and worse question pairs $\mathcal{P} = \cup_{q \in \mathcal{D}} \mathcal{P}(q)$ is determined by N_+, N_-, and $|\mathcal{D}|$ (i.e., $|\mathcal{P}| = N_+ \times N_- \times |\mathcal{D}|$), where the choice of N_+ and N_- is rather arbitrary. To explore the impact of N_+ and N_- on the performance, we evaluated our method with varying N_+ and N_- over K-QA OASST1QA with Llama-3-8B[12]. The results are summarized in Figure 3.

For the K-QA dataset, finding the optimal values of N_+ and N_- is not easy since multiple criteria are involved, but we would say $|\mathcal{P}|$ between 5k and 20k works well. Beyond 20k, S_{comp} starts to decrease, and S_{cont} spikes. In OASST1QA, the performance increases along with $|\mathcal{P}|$ increases by 20k, then it decreases. The performance drops

[12]We evaluated (N_+, N_-) in $\{(5,10),(10,10),(10,20),(15,20),(20,20)\}$ for K-QA and in $\{(1,5),(2,5),(4,5),(5,6),(5,8)\}$ for OASST1QA, which corresponds to $|\mathcal{P}|$ being 5k, 10k, 20k, 30k, and 40k, respectively.

Original Question	How do I know if it's a canker sore versus herpes?
Original Answer S_{comp} ↑: 2/6 S_{cont} ↓: 1	Canker sores and cold sores (caused by herpes simplex virus) are both types of oral lesions but they have distinct characteristics. **Canker Sore:** • **Location**: Canker sores typically occur on the inside of the mouth, tongue, lips or throat. **Cold Sore (Herpes Simplex Virus):** • **Location**: Cold sores typically occur on the lips, around the mouth, or on the nose. ...
Rewritten Question	How can I distinguish between a canker sore and a herpes outbreak based on their symptoms and characteristics?
Rewritten Answer S_{comp} ↑: 3/6 S_{cont} ↓: 0	**Canker Sores (Aphthous Ulcers):** • **Location**: Typically occur on the inside of the mouth, tongue, lips, or throat. **Herpes Simplex Virus (HSV) Outbreaks:** • **Location**: Can occur anywhere on the body but most commonly on the mouth, lips, or genital area. • **Appearance**: Small, fluid-filled blisters that eventually crust over and form a scab. ...

Table 3: An example of original and rewritten questions in K-QA with their answers. Text highlighted in green is consistent with the must-to-have fact, whereas red contradicts the must-to-have fact. Omitted text in answers, represented by "...", are irrelevant to any must-to-have fact.

Metric	Orig.	Ours	PRewrite	T5	PRoMPTed
S_{comp} ↑	0.4573	**0.4600**	0.4409	0.4160	**0.5012**
S_{cont} ↓	0.4400	**0.4000**	**0.3600**	**0.3600**	0.5400

Table 4: Comparison of different question rewriting methods. **Bold** indicates the optimization method is better than the original results. **Orig.** indicates the original results. **T5** indicates the replicated result of ill-formed rewriting.

when $|\mathcal{P}|$ larger than 20k is attributed to overfitting during DPO training. These results highlight the necessity of adjusting N_+ and N_- for each dataset.

Comparison with Other Methods

To validate the superiority of our method, we compared it with other question rewriting methods on the K-QA dataset and Llama3-8B-instruct. In addition to the PRoMPTed method mentioned above, we considered two other closely related methods. The first method employs a transformer model to rewrite ill-formed questions into well-structured ones, while it does not evaluate the impact on answer quality (Chu et al. 2020). We replicated this method using T5-Flan-base and performed better on their original dataset. The second method, PRewrite (Kong et al. 2024), is an RL-based task-level prompt optimization method that uses RLHF, requiring a differentiable reward. However, some LFQA datasets, such as K-QA, lack differentiable rewards. To address this, we implemented DPO, a variant of RLHF, to replicate it. We used prompts from it:

```
Rewrite the following instruction via
rephrasing and/or adding specific
requirements. Add instructions that
would be helpful to solve the problem
correctly. Output the new instruction
only.
```

to optimize the original task instruction: "provide the answer:", and then appended the optimized instruction to the original questions to obtain answers. The optimized instruction can be found in the extended version appendix.

As shown in Table 4, only our method demonstrates improvements in both metrics. While the PRoMPTed method improves S_{comp}, it significantly compromises S_{cont}. This highlights the superiority of our method in balancing these competing objectives and further confirms that our method outperforms other question rewriting methods.

	K-QA		TruthfulQA			OQA
Rewriter	S_{comp} ↑	S_{cont} ↓	S_{truth} ↑	S_{info} ↑	S_{overall} ↑	S_{pref} ↑
Original	0.4573	0.4400	0.7683	0.9664	0.7397	0.8654
Rw-K	**0.4600**	**0.4000**	**0.7834**	0.9535	**0.7454**	**0.8759**
Rw-T	0.4104	**0.2800**	**0.8059**	**0.9668**	**0.7789**	**0.8839**
Rw-O	0.4510	**0.4200**	0.7622	0.9373	0.7155	**0.9104**

Table 5: Performance of rewriters across datasets on Llama3-8B-instruct: OQA represents OASST1QA, Rw-K, Rw-T, and Rw-O represent rewriters trained on K-QA, TruthfulQA and OASST1QA, respectively. **Bold** indicates the rewriter performing better than the corresponding original for this LLM.

Cross Dataset Generalizability

We explored the performance of rewriters trained on Llama3-8B-instruct across different datasets. Table 5 shows that rewriters trained in the K-QA and TruthfulQA datasets can optimize the generated answers on the OASST1QA dataset, but each fails on one metric in their respective datasets. In contrast, rewriters trained on the OASST1QA dataset are almost ineffective on the other two datasets. This suggests that training on more complex LFQA datasets, which include multiple automatic evaluation criteria, yields rewriters with better generalizability.

Discussion

Our question rewriters can significantly improve the representation of questions, making them more likely to obtain higher-quality answers with LLMs. To quantitatively analyze how attributes impact the evaluation criteria of generated answers, we study 50 original questions in KQA's test set and their rewritten versions, resulting in 100 questions in total. We adopt 10 attributes: non-leadingness, word choice, tone, conciseness, neutrality, grammar and spelling, structure, politeness, clarity, and emotion, which are identified by an LLM. For each attribute and each question, we use GPT-4o to assign a score ranging from 1 to 5 to the 100 questions. The definitions of attributes and the prompt templates are available in the extended version appendix.

To explore the important attributes that determine evaluation criteria S_{comp} and S_{cont}, we use a random forest regressor that takes the attribute scores as input and predicts either S_{comp} or S_{cont} of each question. We train the regressors with these 100 questions and use the predictions again for them.[13] The regressors yielded R^2 values of 0.56 and 0.55 for S_{comp} and S_{cont}, respectively, demonstrating that the attribute scores are significantly correlated with the criteria. The random forest regressors provide feature importance, which, in our case, corresponds to the importance of each attribute.

In addition to the attribute importance, we also examine whether each attribute has a positive or negative impact on the evaluation criteria. To this end, we define the impact by:

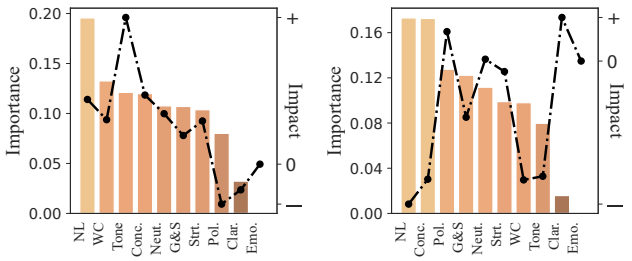
$$I_{la} = \hat{S}_{la} - \check{S}_{la} \tag{6}$$

where $l \in \{\text{comp}, \text{cont}\}$ and a is one of the 10 attributes; $\hat{S}_{la}$ and $\check{S}_{la}$ are the averages of evaluation criterion S_l of questions whose attribute score for a are among the top-50 and bottom-50, respectively. Specifically, $\hat{S}_{la}$ is given by:

$$\hat{S}_{la} = \frac{1}{50} \sum_{S_l \in \hat{\mathcal{S}}_{la}} S_l, \tag{7}$$

where $\hat{\mathcal{S}}_{la}$ is the set of scores S_l of questions whose attribute scores are among top 50. $\check{S}_{la}$ is defined likewise. A higher $\hat{S}_{la}$, for instance, means that the attribute a is positively correlated with S_l.

[13] We consider the choice of using the test set for both training the regressors and analysis is reasonable as we wish to show how much the attribute scores can explain criteria S_{comp} and S_{cont}.



(a) S_{comp} (b) S_{cont}

Figure 4: The importance and impact of attributes conciseness (Conc.), structure (Strt.), word choice (WC), emotion (Emo.), non-leadingness (NL), grammar and spelling (G&S), neutrality (Neut.), tone, clarity (Clar.), and politeness (Pol.). The bar plots are important, while the line plots are impact.

As shown in Figure 4, for S_{comp}, *non-leadingness*, word choice, and tone are the most contributing attributes to regression, while for S_{comp}, *non-leadingness*, *conciseness*, and *politeness* are important. Intriguingly, *no-leadingness* for example, which means a question does not give some implication of a certain answer, is the most contributing attribute for both criteria. It is not straightforward to interpret this result, but being or not being leading may give some cues about attributes to be used for regression. At least, these attributes are somehow correlated with the evaluation criteria, so the impact of these attributes can be meaningful.

As for the impact, we can see *tone* gives a positive impact to S_{comp}, while *non-leadingness* and *conciseness* are negatively correlated with S_{cont}. A higher attribute score for *tone* means the question is written in a formal language and a professional manner. We can reason that a formal language triggers expert knowledge encompassed in an LLM, which is likely to be written also in a formal language with proper wordings. Meanwhile, being non-leading and concise leads to lower S_{cont}, which is preferable. These results also make much sense; extra text in a question can lead to knowledge that is still relevant to the extra text but irrelevant to the question. Overall, our importance and impact analysis unveils that our question rewriter learns to generate professional, non-leading, and concise questions, which align with our intuitions, solely through supervision by S_{comp} and S_{cont}.

Conclusions and Future Work

This paper proposes single-round instance-level question optimization for LFQA tasks, coined *question rewriter*. We employ DPO to optimize the question rewriter with automatic evaluation criteria. Our experimental results demonstrated that our question rewriter can generate questions that give better answers in terms of the automatic evaluation criteria. Meanwhile, although our method demonstrates some degree of cross-domain generalizability, it still has limitations in performance. Therefore, exploring completely domain-agnostic methods would be an interesting direction for future research.

Acknowledgments

This work was supported by World Premier International Research Center Initiative (WPI), MEXT, Japan. This work is also supported by JST ACT-X Grant Number JPMJAX24C8 and JSPS KAKENHI No. 24K20795. This work is partly supported by JSPS KAKENHI No. JP23H00497, JST CREST Grant No. JPMJCR20D3, and JST FOREST Grant No. JPMJFR216O.

References

Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku.

Atallah, S.; Banda, N.; Banda, A.; and Roeck, N. 2023. How large language models including generative pre-trained transformer (GPT) 3 and 4 will impact medicine and surgery. *Techniques in Coloproctology*, 27(8): 609–614.

Bai, Y.; Jones, A.; Ndousse, K.; Askell, A.; Chen, A.; DasSarma, N.; Drain, D.; Fort, S.; Ganguli, D.; Henighan, T.; et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Bhat, M. M.; Meng, R.; Liu, Y.; Zhou, Y.; and Yavuz, S. 2023. Investigating Answerability of LLMs for Long-Form Question Answering. *CoRR*, abs/2309.08210.

Chai, Y.; Wang, S.; Sun, Y.; Tian, H.; Wu, H.; and Wang, H. 2022. Clip-Tuning: Towards Derivative-free Prompt Learning with a Mixture of Rewards. In Goldberg, Y.; Kozareva, Z.; and Zhang, Y., eds., *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, 108–117. Association for Computational Linguistics.

Chen, L.; Chen, J.; Goldstein, T.; Huang, H.; and Zhou, T. 2023. InstructZero: Efficient Instruction Optimization for Black-Box Large Language Models. *CoRR*, abs/2306.03082.

Chu, Z.; Chen, M.; Chen, J.; Wang, M.; Gimpel, K.; Faruqui, M.; and Si, X. 2020. How to ask better questions? a large-scale multi-domain dataset for rewriting ill-formed questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 7586–7593.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *CoRR*, abs/2110.14168.

Diao, S.; Huang, Z.; Xu, R.; Li, X.; Lin, Y.; Zhou, X.; and Zhang, T. 2023. Black-Box Prompt Learning for Pre-trained Language Models. *Trans. Mach. Learn. Res.*, 2023.

Fernando, C.; Banarse, D.; Michalewski, H.; Osindero, S.; and Rocktäschel, T. 2023. Promptbreeder: Self-Referential Self-Improvement Via Prompt Evolution. *CoRR*, abs/2309.16797.

Guo, Q.; Wang, R.; Guo, J.; Li, B.; Song, K.; Tan, X.; Liu, G.; Bian, J.; and Yang, Y. 2023. Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers. *CoRR*, abs/2309.08532.

Harrington, S. A. 2023. The Case for Large Language Model Optimism in Legal Research from a Law & Technology Librarian. *Available at SSRN*.

Kim, G.; Kim, S.; Jeon, B.; Park, J.; and Kang, J. 2023. Tree of Clarifications: Answering Ambiguous Questions with Retrieval-Augmented Large Language Models. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 996–1009. Singapore: Association for Computational Linguistics.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large Language Models are Zero-Shot Reasoners. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Kong, W.; Hombaiah, S. A.; Zhang, M.; Mei, Q.; and Bendersky, M. 2024. PRewrite: Prompt Rewriting with Reinforcement Learning. *CoRR*, abs/2401.08189.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A. P.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; Toutanova, K.; Jones, L.; Kelcey, M.; Chang, M.; Dai, A. M.; Uszkoreit, J.; Le, Q.; and Petrov, S. 2019. Natural Questions: a Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguistics*, 7: 452–466.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In Moens, M.; Huang, X.; Specia, L.; and Yih, S. W., eds., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, 3045–3059. Association for Computational Linguistics.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In Zong, C.; Xia, F.; Li, W.; and Navigli, R., eds., *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, 4582–4597. Association for Computational Linguistics.

Lin, S.; Hilton, J.; and Evans, O. 2022. TruthfulQA: Measuring How Models Mimic Human Falsehoods. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, 3214–3252. Association for Computational Linguistics.

Lin, X.; Dai, Z.; Verma, A.; Ng, S.; Jaillet, P.; and Low, B. K. H. 2024. Prompt Optimization with Human Feedback. *CoRR*, abs/2405.17346.

Lin, X.; Wu, Z.; Dai, Z.; Hu, W.; Shu, Y.; Ng, S.; Jaillet, P.; and Low, B. K. H. 2023. Use Your INSTINCT: INSTruction optimization usIng Neural bandits Coupled with Transformers. *CoRR*, abs/2310.02905.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023. Pre-train, Prompt, and Predict: A Systematic Sur-

vey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.*, 55(9): 195:1–195:35.

Manes, I.; Ronn, N.; Cohen, D.; Ber, R. I.; Horowitz-Kugler, Z.; and Stanovsky, G. 2024. K-QA: A Real-World Medical Q&A Benchmark. *CoRR*, abs/2401.14493.

Min, S.; Krishna, K.; Lyu, X.; Lewis, M.; Yih, W.; Koh, P. W.; Iyyer, M.; Zettlemoyer, L.; and Hajishirzi, H. 2023. FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, 12076–12100. Association for Computational Linguistics.

Patel, A.; Li, B.; Rasooli, M. S.; Constant, N.; Raffel, C.; and Callison-Burch, C. 2023. Bidirectional Language Models Are Also Few-shot Learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Prasad, A.; Hase, P.; Zhou, X.; and Bansal, M. 2023. GrIPS: Gradient-free, Edit-based Instruction Search for Prompting Large Language Models. In Vlachos, A.; and Augenstein, I., eds., *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, 3827–3846. Association for Computational Linguistics.

Pryzant, R.; Iter, D.; Li, J.; Lee, Y. T.; Zhu, C.; and Zeng, M. 2023. Automatic Prompt Optimization with "Gradient Descent" and Beam Search. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, 7957–7968. Association for Computational Linguistics.

Radford, A.; and Narasimhan, K. 2018. Improving Language Understanding by Generative Pre-Training.

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

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Roberts, A.; Raffel, C.; and Shazeer, N. 2020. How Much Knowledge Can You Pack Into the Parameters of a Language Model? In Webber, B.; Cohn, T.; He, Y.; and Liu, Y., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, 5418–5426. Association for Computational Linguistics.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347.

Shi, W.; Han, X.; Gonen, H.; Holtzman, A.; Tsvetkov, Y.; and Zettlemoyer, L. 2023. Toward Human Readable Prompt Tuning: Kubrick's The Shining is a good movie, and a good prompt too? In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, 10994–11005. Association for Computational Linguistics.

Shin, T.; Razeghi, Y.; IV, R. L. L.; Wallace, E.; and Singh, S. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In Webber, B.; Cohn, T.; He, Y.; and Liu, Y., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, 4222–4235. Association for Computational Linguistics.

Srivastava, S.; Huang, C.; Fan, W.; and Yao, Z. 2023. Instance Needs More Care: Rewriting Prompts for Instances Yields Better Zero-Shot Performance. *CoRR*, abs/2310.02107.

Sun, T.; He, Z.; Qian, H.; Huang, X.; and Qiu, X. 2022a. BBTv2: Pure Black-Box Optimization Can Be Comparable to Gradient Descent for Few-Shot Learning. *CoRR*, abs/2205.11200.

Sun, T.; Shao, Y.; Qian, H.; Huang, X.; and Qiu, X. 2022b. Black-Box Tuning for Language-Model-as-a-Service. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvári, C.; Niu, G.; and Sabato, S., eds., *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, 20841–20855. PMLR.

Wang, B.; Chang, J.; Qian, Y.; Chen, G.; Chen, J.; Jiang, Z.; Zhang, J.; Nakashima, Y.; and Nagahara, H. 2024. DiReCT: Diagnostic Reasoning for Clinical Notes via Large Language Models. *CoRR*, abs/2408.01933.

Wang, X.; Li, C.; Wang, Z.; Bai, F.; Luo, H.; Zhang, J.; Jojic, N.; Xing, E. P.; and Hu, Z. 2023. PromptAgent: Strategic Planning with Language Models Enables Expert-level Prompt Optimization. *CoRR*, abs/2310.16427.

Yang, C.; Wang, X.; Lu, Y.; Liu, H.; Le, Q. V.; Zhou, D.; and Chen, X. 2023. Large Language Models as Optimizers. *CoRR*, abs/2309.03409.

Zhang, T.; Qin, P.; Deng, Y.; Huang, C.; Lei, W.; Liu, J.; Jin, D.; Liang, H.; and Chua, T. 2024. CLAMBER: A Benchmark of Identifying and Clarifying Ambiguous Information Needs in Large Language Models. *CoRR*, abs/2405.12063.

Zhong, Z.; Friedman, D.; and Chen, D. 2021. Factual Probing Is [MASK]: Learning vs. Learning to Recall. In Toutanova, K.; Rumshisky, A.; Zettlemoyer, L.; Hakkani-Tür, D.; Beltagy, I.; Bethard, S.; Cotterell, R.; Chakraborty, T.; and Zhou, Y., eds., *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, 5017–5033. Association for Computational Linguistics.

Zhou, Y.; Muresanu, A. I.; Han, Z.; Paster, K.; Pitis, S.; Chan, H.; and Ba, J. 2023. Large Language Models are Human-Level Prompt Engineers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.