Learning to Paraphrase for Alignment with LLM Preference

Junbo Fu[1], Guoshuai Zhao[1*], Yimin Deng[1], Yunqi Mi[1], Xueming Qian[2]
[1]School of Software Engineering, Xi'an Jiaotong University
[2]School of Information and Communication Engineering, Xi'an Jiaotong University
{guoshuai.zhao, qianxm}@xjtu.edu.cn,{junbofu, dymanne, miyunqi}@stu.xjtu.edu.cn

Abstract

Large Language Models (LLMs) exhibit the issue of paraphrase divergence. This means that when a question is phrased in a slightly different but semantically similar way, LLM may output a wrong response despite being able to answer the original question correctly. Previous research has regarded this issue as a problem of the model's robustness to question paraphrase and proposed a retraining method to address it. However, retraining faces challenges in meeting the computational costs and privacy security demands of LLMs. In this paper, we regard this issue as a problem of alignment with model preferences and propose PEARL (**P**reference-driv**E**n p**A**raphRase **L**earning). This is a black-box method that enhances model performance by paraphrasing questions in expressions preferred by the model. We validate PEARL across six datasets spanning three tasks: open-domain QA, commonsense reasoning, and math word problem. Extensive experiments demonstrated not only the outstanding performance but also the composability, transferability, and immense potential of PEARL, shedding new light on the black-box tuning of LLMs[1].

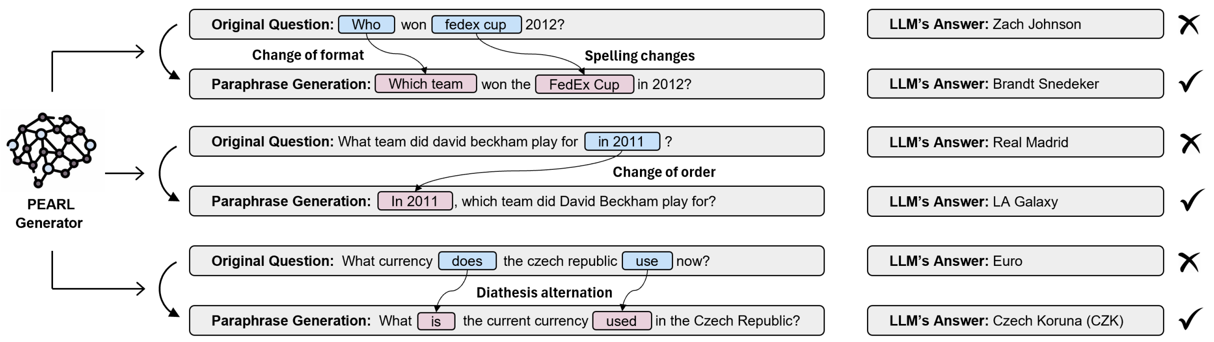
1 Introduction

The era of Large Language Models (LLMs) has arrived. With the continuous increase in the scale of parameters and the ongoing development of pre-training methods, LLM's overall performance is becoming increasingly powerful (Kaplan et al., 2020; Wei et al., 2021; Bai et al., 2022). However, the rapid development of LLM has also brought about some issues. Firstly, the surge in parameter scale has led to a substantial increase in computational resources required for tuning them. This limitation restricts the participation of many organizations and individuals, which is not conducive to the healthy development of the community. Although some research has proposed methods to optimize the performance of LLMs with lower computational requirements (Li and Liang, 2021; Liu et al., 2022; Hu et al., 2021), they primarily are white-box tuning methods, which necessitate access to model parameters. With the continuous evolution of regulations such as GDPR, the security and privacy concerns surrounding LLM are increasingly being emphasized (Wu et al., 2023; Yidong et al., 2023). As a result, some designers of LLMs are opting to provide services to users through API rather than exposing the complete model. This trend has made it increasingly challenging to apply white-box tuning. Therefore, black-box methods are gaining attention, such as retrieving relevant examples (Rubin et al., 2022; Ye et al., 2023; Li et al., 2023) and suitable prompts (Jiang et al., 2020; Cheng et al., 2023) to add to LLM's input.

A notable phenomenon is the issue of paraphrase divergence in language models. Indeed, when a question is phrased in a slightly different but semantically similar way, the language model can output a wrong response despite being able to answer the original question correctly. Figure 1 shows three such examples. This can be explained as the language model does not only learn the knowledge itself from the corpus during pre-training but also learns the expression pattern associated with specific knowledge (Heinzerling and Inui, 2021). Gan and Ng (2019) regard this issue as a problem of the model's robustness to question paraphrase and proposed a retraining method to enhance the robustness of the model. However, this method is white-box and requires tuning all parameters of the model, making it difficult to meet the computational costs and privacy security demands of LLMs.

To address these, we regard the paraphrase divergence of LLM as a problem of alignment with model preferences and propose PEARL (**P**reference-driv**E**n p**A**raphRase **L**earning). This

[*]Corresponding author
[1]https://github.com/630bdd/PEARL



Original Question: Who won fedex cup 2012?
Change of format
Spelling changes
Paraphrase Generation: Which team won the FedEx Cup in 2012?
LLM's Answer: Zach Johnson
LLM's Answer: Brandt Snedeker
PEARL Generator
Original Question: What team did david beckham play for in 2011 ?
Change of order
Paraphrase Generation: In 2011, which team did David Beckham play for?
LLM's Answer: Real Madrid
LLM's Answer: LA Galaxy
Original Question: What currency does the czech republic use now?
Diathesis alternation
Paraphrase Generation: What is the current currency used in the Czech Republic?
LLM's Answer: Euro
LLM's Answer: Czech Koruna (CZK)

Figure 1: How PEARL Generator works. When paraphrasing questions with different semantics, the PEARL Generator adaptively selects different paraphrase types.

is a black-box method that enhances model performance by paraphrasing questions in expressions preferred by the model. Specifically, we train a PEARL Generator to learn the model's preferences for expressions. During inference, the PEARL Generator generates paraphrases in alignment with model preference and then feeds them into the model. In this paper, the training of the PEARL Generator uses a seq2seq approach. Considering the aspects of the diversity of paraphrase types and the preservation of semantics, we propose a prompt-based method for the automatic construction of final training sets. It's worth noting that the parameter scale of the PEARL Generator we train is significantly smaller than that of our target LLM, and it requires only a small amount of training data.

Fundamentally, PEARL is a process of learning model preference for expressions and incorporates the preference into questions to guide the model in producing correct answers. This parallels prompt learning, where adjusting prompts caters to the expressions that model is accustomed to during pre-training (Liu et al., 2023). However, prompt learning involves incorporating the preference information into the prompts that are independent of the questions. It is important to note that prompts are typically task-specific or domain-specific, meaning that the preference information learned through prompt learning is preserved in a fixed format within the prompts for a particular task or domain. However, questions are not specific to any domain or task, they are constantly changing. This necessitates dynamically incorporates the model's preference information into the questions. The PEARL Generator can incorporate preference adaptively during paraphrase by learning model preferences for expressions under different semantics. As depicted in Figure 1, the adaptability of the PEARL Generator can be concretely manifested in its ability to apply different paraphrase types to different questions. The paraphrase type represents the lexical variables manipulated during paraphrase generation (Wahle et al., 2023) .

To validate the effectiveness of our approach, we conducted extensive experiments on six datasets across three tasks: open-domain QA, commonsense reasoning, and math word problem. The experiments demonstrate that PEARL significantly enhances model performance. The contributions of this paper are as follows:

- We proposed PEARL, a method that addresses the paraphrase divergence in LLMs by paraphrasing questions in alignment with model preference. PEARL is a black-box method, requiring significantly fewer trainable parameters than the target LLM and only a small amount of training data.
- We propose a prompt-based method for the automatic construction of training sets, aiming to enhance training effectiveness by increasing diversity in paraphrase types while ensuring the preservation of semantics.
- Experiments conducted on six datasets across three tasks demonstrate not only the excellent performance, but also the composability, transferability, and significant potential of PEARL, shedding new light on the black-box tuning of LLMs.

2 Related Work

Paraphrases are texts that convey identical meanings while using different words or structures (Bhagat and Hovy, 2013; Vila et al., 2014), which is a reflection of the complexity and diversity of human language. Given a sentence, paraphrase generation aims to craft its paraphrases that are different from the original sentence, while maintaining the original meaning (Zhou and Bhat, 2021). Many downstream tasks in natural language processing leverage paraphrase generation, such as question answering (Dong et al., 2017; Gan and Ng, 2019),

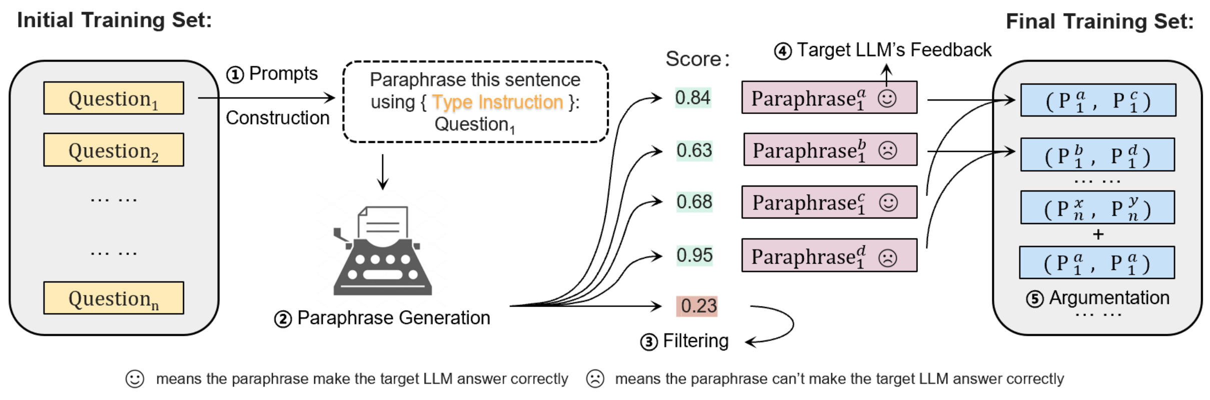


Figure 2: The construction process of the training set for PEARL Generator.

semantic parsing (Cao et al., 2020), dialogue systems (Liang et al., 2022; Panda et al., 2021) and machine translation (Thompson and Post, 2020). In the era of LLMs, leveraging language models for end-to-end paraphrasing ensures both substantial accuracy and diversity while maintaining flexibility and ease of use (Cegin et al., 2023). Therefore, this approach has become a mainstream choice for paraphrase generation. In this paper, all paraphrases are generated through this end-to-end approach.

Currently, deploying large language models as services has become a common practice, such as GPT-4 (Achiam et al., 2023), Claude 3 (Anthropic, 2024). On one hand, fine-tuned models in specific domains may possess knowledge that implicates privacy concerns. On the other hand, we cannot ascertain whether the answers directly provided by the model adhere to legal regulations and ethical norms. Sun et al. were the first to propose the concept of LMaaS (Language Model as a Service) (Sun et al., 2022), where LLMs are provided to users through interfaces to accomplish various downstream tasks. And they advocated for optimizing model performance through black-box methods. The PEARL method we propose is indeed a black-box approach, enabling optimization of model performance while requiring only access to input-output pairs of the LLMs.

3 Training of PEARL Generator

We choose to directly treat preference-driven paraphrase generation as a seq2seq task. Therefore, the key to training the PEARL Generator lies in constructing effective training sets. Given an initial training set $Q = \{q_1, q_2, \ldots, q_n\}$ comprising n questions, our goal is to construct a set of paraphrased sentence pairs $P = \left\{(p_1^a, p_1^c), \left(p_1^b, p_1^d\right), \ldots, (p_n^x, p_n^y)\right\}$, where each pair consists of two sentences that are synonymous and all paraphrases originate from questions in the initial training set Q. The former p_n^x in the sentence pair is the input to the PEARL Generator, which represents the question that is not aligned with model preference. The latter p_n^y in the sentence pair is the target output of the PEARL Generator, representing the question aligned with model preference.

The process of constructing paraphrased sentence pairs can be summarized as generating paraphrases for the questions in the initial training set according to certain rules, and then determining the preference for each question and their paraphrases based on whether the target LLM can correctly answer them. Many existing pre-trained or tuned language models can be used as paraphrase generators directly given relevant prompts. However, their generation tends to be limited to a few types of paraphrases. For the PEARL Generator, we aim to include a sufficient variety of paraphrase types in its training data to increase its opportunities to produce effective paraphrases. Therefore, we choose to include the type instruction in the prompts of the paraphrase generator, as shown in Figure 2. The type classification we use is detailed in Figure 3.

Secondly, maintaining the semantic essence of the paraphrases is also crucial. If the paraphrases generated by the PEARL Generator alter the fundamental meaning, the target LLM would not be able to provide correct answers. But this aspect cannot be solely ensured by the capabilities of the paraphrase generator itself, we require the assistance of machine evaluation metrics to help filter out incorrect paraphrases. In this paper, we choose BLUERT (Sellam et al., 2020) as the metric for filtering. When the score between the original question and its paraphrase falls below a predetermined threshold θ, we filter out the paraphrased question.

Finally, we divide synonymous paraphrases into two sets: the preferred paraphrase set p_n^+ and the non-preferred paraphrase set p_n^- based on feedback from the target LLM. Then obtain the set of paraphrased sentence pairs $P = \bigcup_{n=1}^{N} (P_n^+ \times P_n^-)$ by taking their Cartesian Product. Moreover, PEARL Generator does not need to paraphrase all questions since some of them may already align with the target LLM's preferences. Therefore, we augment the training set with a certain amount of identical sentence pairs I, allowing the PEARL Generator to understand that it can skip paraphrasing questions that are already in alignment with the model's preferences. We control the ratio between the two types of sentence pairs using a parameter λ to obtain the final training set $F = P \cup I$.

4 Experiment

4.1 Set Up

4.1.1 Implementation Details

We tune the FLAN-T5-L (770M parameters) (Chung et al., 2022) to train a PEARL Generator for each task, with details of the composition of the initial training set outlined in Section 4.1.2. We set the ratio parameter λ for data augmentation to 0.2 and the filtering threshold θ to 0.4 in all three tasks. The learning rate for the training of three PEARL Generators is set to $1e-4$, with a batch size of 32 and training epochs set to 3. In this paper, the paraphrase generation model utilized in our training sets construction is COEDIT-XXL (Raheja et al., 2023), an open-source text editing system.

We selected MPT-instruct-7B (Team et al., 2023a) as the primary target LLM to validate the effectiveness of PEARL. Additionally, we employed TinyLlama-1.3B (Zhang et al., 2024), RedPajama-instruct-3B (Computer, 2023), BLOOMz-7B (Muennighoff et al., 2023), and MPT-instruct-30B (Team et al., 2023b) to assess the transferability of the PEARL Generator across models of different series and parameter scales. To ensure the reproducibility of experiment results, we utilized a greedy decoding strategy with max new tokens set to 256 for all target LLMs.

We utilize EM as the evaluation metric for question-answer tasks and accuracy for multiple-choice tasks. The classification of paraphrase types in this paper follows the scheme established by Wahle et al. (2023) in Figure 3.

4.1.2 Datasets

We conduct experiments on three tasks including commonsense reasoning, math word problems, and open-domain QA. For commonsense reasoning, we choose CommonsenseQA (CSQA) (Talmor et al., 2019) and SocialIQA (SiQA) (Sap et al., 2019). For math word problem, we choose SVMAP (Patel et al., 2021). For open-domain QA, we choose WebQuestionSP (WebQSP) (Yih et al., 2016), ComplexWebQuestion (CWQ) (Talmor and Berant, 2018) and ComplexQuestions (CompQ) (Bao et al., 2016). The detailed information for all these existing datasets can be found in Appendix A.

	open-domain QA	Commonsense Reasoning	Math Word Problem
Size	1618	1591	1320

Table 1: The size of the final training sets for three tasks.

We directly use the test set of an original split from each existing dataset for evaluation. For datasets without openly available test sets, we employ the dev set for evaluation. Since we opt to train a PEARL Generator for each task, the initial training set for each task is sourced from the existing training sets of that task. We randomly select part of the data from each existing training set and add it to the corresponding task's initial training set. The size of the final training sets is shown in Table 1.

4.2 Main Result

We show the performance comparison in the six datasets in Table 2 to validate the effectiveness of PEARL. We selected one baseline and three comparative methods. To ensure fairness in comparison, we categorize methods into white-box and black-box as follows:

Black-Box Methods

Manual Prompt: We follow the official recommended template for manual prompt construction and use this method as our baseline for comparison.

UPRISE (Cheng et al., 2023): This is a universal prompt auto-retrieval method that tunes a lightweight prompt retriever based on contrastive learning. In this paper, we retrieve corresponding prompts from the UPRISE prompt pool for each question and add them to the prompt template as a comparative method.

EPR (Rubin et al., 2022): This is an ICL(In-Context Learning) method that trains a dense retriever to retrieve examples relevant to the input from an example pool. In this paper, we use question-answer pairs from the initial training set

Method	CompQ EM	WebQSP EM	CWQ EM	CSQA acc	SiQA acc	SVAMP EM	Avg.	Gains
White-Box Method								
LORA	32.25	51.49	40.58	50.29	42.12	26.30	40.51	-
Black-Box Method								
Manual Prompt	27.00	43.20	32.71	25.55	34.75	26.00	31.54	-
+PEARL	29.88	44.97	34.78	26.78	35.52	**27.20**	33.19	5.23%↑
UPRISE	30.13	44.66	33.84	27.76	36.54	-	34.59	-
+PEARL	31.25	45.64	35.55	**30.06**	**39.66**	-	36.43	5.32%↑
EPR	33.75	46.31	45.67	26.13	37.41	21.60	35.15	-
+PEARL	**36.25**	**49.24**	**46.46**	27.35	39.20	23.50	**37.00**	5.27%↑

Table 2: Comparative experiments on six datasets across three tasks. We separate the comparison between black-box and white-box methods and highlight the best-performing black-box method on each dataset in **bold**. Due to the absence of relevant prompts for the math word problem task in the UPRISE prompt pool, the results of this method are missing on the SVAMP dataset.

as the example pool, retrieving corresponding examples from it for each question and adding them to the prompt template as a comparative method.

White-Box Methods

LORA (Hu et al., 2021): Low-Rank Adaptation is a PEFT (Parameter-Efficient Fine-Tuning) method. In this paper, we directly utilize the initial training sets of three tasks to conduct LORA Tuning on the target LLM as a comparative method.

On one hand, we directly compare PEARL with the other methods. On the other hand, considering that PEARL is a novel method whose optimization aspects do not conflict with those of previous methods, we also employ a more persuasive comparative approach which is combination. Specifically, we combine PEARL with black-box comparative methods to observe its enhancement. The detailed experimental settings of the comparative methods and prompt templates we used are provided in Appendix B and C, respectively.

Firstly, our method achieved an average improvement of 1.65% across the six datasets compared to the baseline, which provides initial validation of the effectiveness of our approach. Furthermore, when our method was combined with the two black-box methods, it exhibited improvements over the original methods on all datasets, achieving average enhancements of 1.84% and 1.85%, respectively. This not only further validates the composability of our method but also indicates that PEARL indeed addresses an aspect overlooked by previous black-box methods. Moreover, it is noticeable that the improvement achieved by combining PEARL with the two comparative black-box methods is more significant compared to using it with only manual prompts. This implies that PEARL may exhibit a synergistic effect when combined with others, potentially yielding greater than additive improvements. It's worth noting that the EPR method relies on semantic relevance to retrieve examples. However, the math word problem task requires logical guidance and semantically similar questions may contain conflicting mathematical logic. Therefore, EPR's performance on SVAMP is not as good as Manual Prompt. Nevertheless, what's important is that our method achieves improvement on SVAMP both when combined with manual prompt and EPR. Finally, the overall performance of the best-performing black-box method which combines PEARL approaches LORA across the six datasets. Moreover, their performance surpasses LORA on the CompQ, CWQ, and SVAMP datasets. The analysis in Section 6 indicates that PEARL still has plenty of untapped potential waiting to be explored.

4.3 Analysis

4.3.1 What Preference do LLM have?

Describing what expression LLM prefers is a difficult question to answer clearly. LLM may exhibit different preferences for various semantics. And these preferences are challenging to articulate directly with precise terms. Hence, we opt to concretize the model preferences based on paraphrase types used in PEARL, leveraging crowdsourcing. More specifically, we represent the model preference as the distribution of paraphrase types that enable the model to correctly answer the question it previously failed to answer correctly. Correspondingly, we represent the preferences learned by the PEARL Generator as the distribution of paraphrase

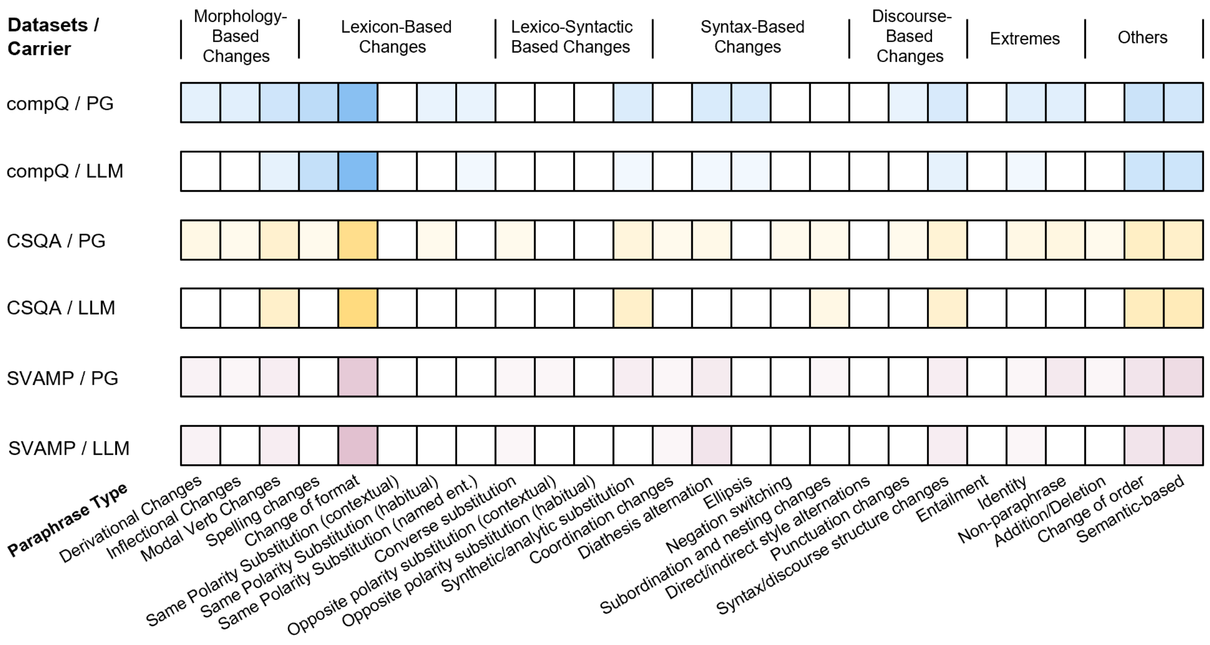


Figure 3: Analysis of model preferences for MPT-instruct-7B. "PG" stands for PEARL Generator. The top section of the image displays the top-level classification of paraphrase types used in our paper, while the bottom section displays the second-level classification. The varying shades of color in different squares represent the distribution of paraphrase types, with darker colors indicating a higher proportion of the paraphrase type in the model's preferences.

types for all paraphrases within PEARL.

We selected one dataset from each of the three tasks for the analysis of model preferences, and the results are depicted in Figure 3. From the top-level classification perspective, the paraphrase types corresponding to model preferences primarily concentrate within three categories: Lexicon-Based Changes, Morphology-Based Changes, and Others. From the second-level classification perspective, the model's preferences for expression mainly manifest in its sensitivity towards three paraphrase types: Change of format, Semantic-based, and Change of order. Among these, the model exhibits particularly high sensitivity to Change of format. Furthermore, it can be observed that the preferences learned by the PEARL Generator are generally consistent with the preferences exhibited by the model, although there are also inconsistencies in some types. For example, in the SVAMP dataset, the PEARL Generator indicates that the model should be sensitive to Synthetic/analytic substitution, but in reality, the model does not exhibit sensitivity to this type. Conversely, in the CSQA dataset, the model demonstrates a stronger sensitivity to Synthetic/analytic substitution compared to what the PEARL Generator has learned.

4.3.2 Transfer of PEARL Generator

Although the pre-training corpora of various LLMs differ, they contain some overlapping text. Therefore, we speculate that there may be some shared preferences among different LLMs, enabling the PEARL Generator to still function when transferred to other target LLMs. Hence, we attempted to employ the PEARL Generator trained on MPT-instruct-7B to LLMs of different series. We also tried employing it on LLMs of the same series but with different parameter scales. The results are shown in Table 3.

Overall, the PEARL Generator still contributes to improving the performance of the target LLM after being transferred, achieving average enhancements of 0.52%, 1.23%, 2.94%, and 0.83% on four LLMs, respectively. And there is an intriguing observation can be observed. When considering solely the scale of model parameters, the performance of the PEARL Generator tends to improve as the size of the target model for transfer becomes closer to that of the original model. Models with similar parameter scales tend to exhibit more similar preferences, offering a potential explanation for this phenomenon. However, we posit that differences in the inherent capabilities of models may also contribute to this experimental observation.

Additionally, the transferred PEARL Generator resulted in performance decreases for some LLMs on a few datasets. This is because the PEARL Generator generated more counterproductive paraphrases than productive paraphrases on

LLM	CompQ EM	WebQSP EM	CWQ EM	CSQA acc	SiQA acc	SVAMP EM	Avg.	Gains
TinyLlama-1.3B	24.75	46.19	31.51	**28.58**	32.91	17.50	30.24	-
+PEARL GENERATOR	**26.13**	**46.25**	**32.23**	27.35	**34.80**	**17.80**	**30.76**	1.72%↑
RedPajama-instruct-3B	13.50	26.67	22.90	32.84	38.89	**12.60**	24.57	-
+PEARL GENERATOR	**16.25**	**27.70**	**26.31**	**33.25**	**40.38**	10.90	**25.80**	5.01%↑
BLOOMz-7B	13.88	27.88	22.93	28.17	33.57	19.10	24.26	-
+PEARL GENERATOR	**16.88**	**32.21**	**26.09**	**31.94**	**35.36**	**20.70**	**27.20**	12.12%↑
MPT-instruct-30B	34.25	46.49	35.61	**54.63**	50.51	57.30	46.47	-
+PEARL GENERATOR	**36.25**	**47.77**	**37.14**	53.15	**51.38**	**58.10**	**47.30**	1.79%↑

Table 3: The performance of the PEARL Generator trained on MPT-instruct-7B when transferred to other target LLMs. We highlight in **bold** the superior performance before and after using the PEARL Generator.

	CompQ EM	WebQSP EM	CWQ EM	CSQA acc	SiQA acc	SVAMP EM	Avg.
PEARL	**29.88**	**44.97**	**34.78**	**26.78**	**35.52**	27.20	**33.19**
-w/o Type Instruction	28.88	44.17	34.04	24.82	34.44	**28.10**	32.41
-w/o Filter	27.00	41.37	30.12	24.57	34.70	25.70	30.58
-w/o Augmentation	26.50	42.04	30.72	24.90	35.21	26.40	30.96

Table 4: Ablation study on the effectiveness of our training set construction strategies across six datasets. We highlight in **bold** the best-performing method on each dataset.

Gini Coefficient	CompQ	CSQA	SVAMP
PEARL	0.8748	0.8731	0.8351
-w/o Type Instruction	0.8032	0.8286	0.7913

Table 5: Gini Coefficients of the paraphrase type distributions generated by the PEARL Generator.

Proportion	CompQ	CSQA	SVAMP
PEARL	3.75%	7.53%	7.7%
-w/o Argumentation	7.88%	12.61%	10.1%

Table 6: Proportions of the counterproductive paraphrase generated by the PEARL Generator.

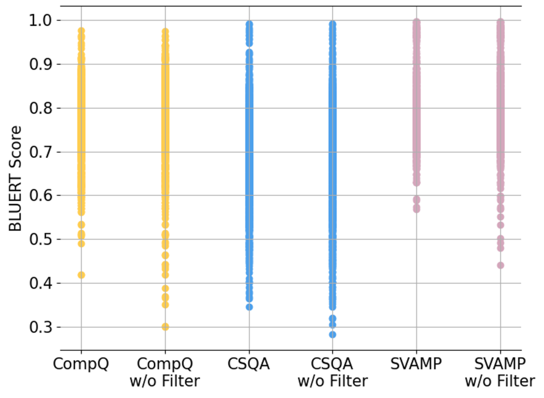


Figure 4: BLUERT scores of the paraphrases generated by the PEARL Generator relative to the original questions.

these datasets. Counterproductive paraphrase refers to paraphrase that leads the model to incorrectly answer questions it would have originally answered correctly, while productive paraphrase has the opposite effect. We believe that the reason for this phenomenon is that the transfer causes a significant gap between the preferences learned by the PEARL Generator and the actual preferences of the target LLMs, which becomes more pronounced on certain datasets.

4.3.3 Ablation Study

We conducted ablation experiments on the six datasets to validate our strategies in training set construction. Table 4 presents the results of our ablation experiments, from which we can observe the following findings:

(a) Without paraphrase type instructions, the overall performance of PEARL decreases. In Table 5, we present the Gini Coefficients of the paraphrase type distributions generated by the PEARL Generator before and after removing type instructions on three representative datasets from each task. The Gini Coefficient is a metric used to measure the degree of inequality in a distribution, ranging from 0 to 1, where values closer to 0 indicate a more even distribution across categories. After incorporating type instructions, the Gini Coefficients on all three datasets have decreased. This indicates that the final training sets under type instruction tend to encourage the PEARL Generator to generate more diverse paraphrase types. We believe this can increase the opportunities for the PEARL Generator to produce effective paraphrases, thereby enhancing the performance of the PEARL.

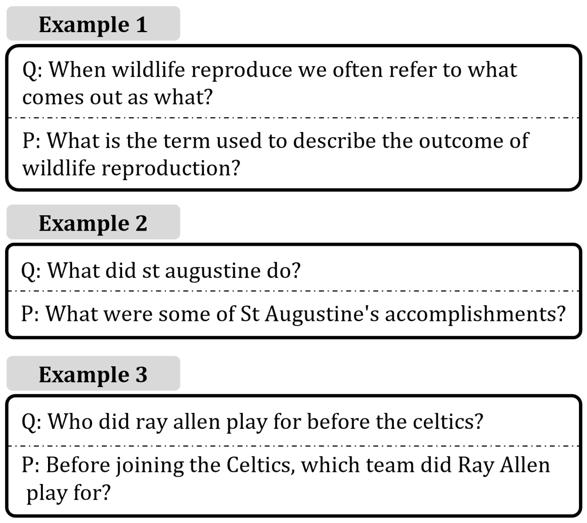
(b) After removing the filter strategy, the performance of PEARL significantly decreases. In Figure 4, we illustrate the BLUERT scores of the paraphrases generated by the PEARL Generator relative to the original questions before and after removing the filter strategy on three datasets. After incorporating the Filter strategy, the number of outliers in the scores significantly decreases and these outliers likely represent incorrect paraphrases. We believe that our filtering strategy successfully removes incorrect paraphrases from the training set, thereby ensuring semantic coherence before and after paraphrasing by the PEARL Generator.

(c) Without the augmentation of identical sentence pairs in the training set, the performance of PEARL significantly decreases. From Figure 3, it can be seen that the paraphrase types used by the PEARL Generator for the three tasks all include a certain proportion of Non-paraphrase types. This indicates that the PEARL Generator has learned to preserve the expression for some questions. Additionally, in Table 6, we compiled the proportions of counterproductive paraphrases generated by the PEARL Generator before and after removing data augmentation. It can be observed that our augmentation strategy significantly reduces the counterproductive paraphrase. Therefore, we believe that the PEARL Generator learns what expressions already satisfy the model preferences through this strategy, and by not altering them, it reduces the likelihood of its paraphrase having a counterproductive effect.

4.3.4 Empirical Suggestions on Paraphrase

In the era of LLMs, some research has provided empirical suggestions on how to prompt LLMs (Saravia, 2022; Nigh., 2023). Based on our analysis of the LLM's preferences for expression, we also provide three empirical suggestions on how to paraphrase questions to align with model preferences. The detailed suggestions are as follows, with corresponding examples in Figure 5

Emphasize the requirement for clarity. Make your questions as clear and specific as possible, even if some emphasis on restrictive requirements may seem unnecessary to you. For instance, in Example 1, it is better to specify directly what you want the LLM to answer, such as 'term', rather than posing a broader question like "what comes out as what". In Example 2, a clearer emphasis on wanting information about "accomplishments" is preferred over a broad inquiry into "what did St. Augustine do".


Example 1

Q: When wildlife reproduce we often refer to what comes out as what?

P: What is the term used to describe the outcome of wildlife reproduction?

Example 2

Q: What did st augustine do?

P: What were some of St Augustine's accomplishments?

Example 3

Q: Who did ray allen play for before the celtics?

P: Before joining the Celtics, which team did Ray Allen play for?

Figure 5: Three examples of effective paraphrases. 'Q' represents the original question and 'P' represents the paraphrased question."

Precede with essential information. Emphasize significant modifying details by placing them at the beginning of the sentence, including restrictive elements such as time, place, manner, reason, purpose, and conditions. For example, in Example 3, the explanatory clause 'Before joining the Celtics' is emphasized by being placed at the beginning of the sentence as an adverbial phrase.

Pay attention to spelling. Correct and meticulous spelling helps ensure accurate responses from the LLMs. when it comes to proper nouns, titles, honorifics, as well as abbreviations, and acronyms, it's important to observe capitalization. In Examples 2 and 3, the paraphrases demonstrate the correct spelling of both personal names and the team name, as opposed to the original questions.

5 Conclusion

In this paper, we regard the paraphrase divergence of LLM as a problem of alignment with model preference. We propose PEARL, a black-box method that enhances model performance by paraphrasing questions in expressions preferred by the model. Extensive experiments on six datasets across three tasks validate the effectiveness, composability, and transferability of PEARL. Furthermore, our analysis experiments concretize LLM's preferences for expression by analyzing paraphrase types and uncovering significant untapped potential in PEARL.

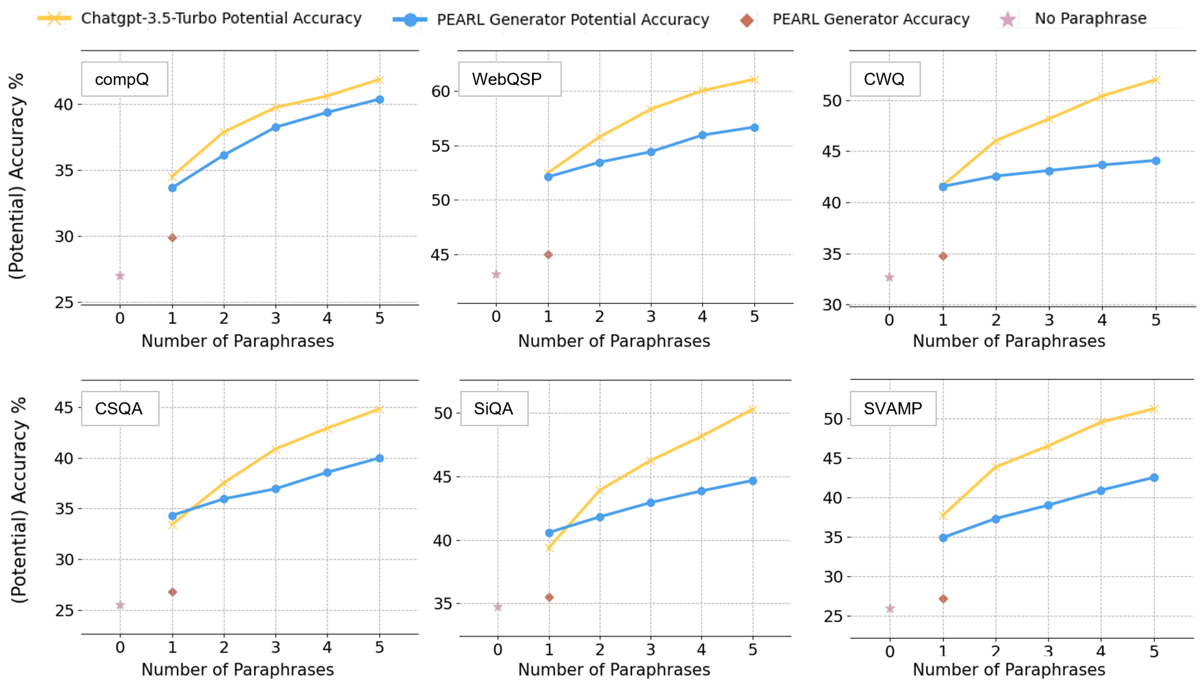


Figure 6: Potential Accuracy of the PEARL Generator compared to ChatGPT-3.5-turbo as the number of paraphrase ranges from 1 to 5.

6 Limitations

We believe that the limiting factors of the PEARL, or rather, where its potential lies, can mainly be attributed to two aspects: One is the PEARL Generator's ability to learn preferences, including how many preferences it can learn and whether the learned preferences are accurate. Another aspect is the PEARL Generator's paraphrase capability itself, including the diversity of paraphrase manner, the fluency and naturalness of sentences after paraphrase, semantic coherence before and after paraphrase, and several other factors. Although we have enhanced the paraphrase capability of the PEARL Generator to some extent through certain strategies in training set construction, we believe that there is still considerable optimization potential.

To explore the potential of PEARL, we have devised two strategies to simulate the alleviation of these two limitations. For the first aspect, we introduce the concept of "Potential Accuracy". Specifically, we perturb the paraphrase generation of the PEARL Generator by adding some prompts before its input, aiming to produce different paraphrases. The specific prompts used can be found in Appendix D. If among these paraphrases there exists one that allows the model to answer correctly, we consider this paraphrase effective and calculate the accuracy based on it, referred to as potential accuracy. For the second aspect, we opt to replace the PEARL Generator with a more powerful paraphrase generation model.

Figure 6 illustrates the results. For the PEARL Generator, when the number of paraphrases is equal to one, its potential accuracy significantly exceeds its accuracy by a noticeable margin. Moreover, as the number of paraphrases increases, its potential accuracy continues to improve. We believe that this indicates there is significant potential to be explored in both the PEARL Generator's ability to learn preferences and its paraphrase capability. As for how to fully exploit this potential, we will continue to investigate in our future work.

Acknowledgments

We are grateful to the anonymous reviewers and the area chair for their insightful comments and suggestions. This work is in part funded by the NSFC, China under Grants 62372364; in part by China Postdoctoral Science Foundation (2020M683496); in part by Shaanxi Provincial Technical Innovation Guidance Plan, China under Grant 2024QCY-KXJ-199; in part by Key RD Plan of Xianyang City, China under Grant L2023-ZDYF-QYCX-002; and in part by the National Postdoctoral Innovative Talents Support Program, China (BX20190273).

References

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

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514, Osaka, Japan. The COLING 2016 Organizing Committee.

Rahul Bhagat and Eduard Hovy. 2013. Squibs: What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817, Online. Association for Computational Linguistics.

Jan Cegin, Jakub Simko, and Peter Brusilovsky. 2023. ChatGPT to replace crowdsourcing of paraphrases for intent classification: Higher diversity and comparable model robustness. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1889–1905, Singapore. Association for Computational Linguistics.

Daixuan Cheng, Shaohan Huang, Junyu Bi, Yuefeng Zhan, Jianfeng Liu, Yujing Wang, Hao Sun, Furu Wei, Weiwei Deng, and Qi Zhang. 2023. UPRISE: Universal prompt retrieval for improving zero-shot evaluation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12318–12337, Singapore. Association for Computational Linguistics.

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

Together Computer. 2023. Redpajama: An open source recipe to reproduce llama training dataset.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

Wee Chung Gan and Hwee Tou Ng. 2019. Improving the robustness of question answering systems to question paraphrasing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6065–6075, Florence, Italy. Association for Computational Linguistics.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1772–1791, Online. Association for Computational Linguistics.

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*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv*, abs/2001.08361.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4644–4668, Toronto, Canada. Association for Computational Linguistics.

Yunlong Liang, Fandong Meng, Ying Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2022. Emotional conversation generation with heterogeneous graph neural network. *Artificial Intelligence*, 308:103714.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15991–16111, Toronto, Canada. Association for Computational Linguistics.

Matt Nigh. 2023. Chatgpt3-free-prompt-list: A free guide for learning to create chatgpt3 prompts. *https://github.com/mattnigh/ChatGPT3-Free-Prompt-List*.

Subhadarshi Panda, Caglar Tirkaz, Tobias Falke, and Patrick Lehnen. 2021. Multilingual paraphrase generation for bootstrapping new features in task-oriented dialog systems. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 30–39, Online. Association for Computational Linguistics.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Vipul Raheja, Dhruv Kumar, Ryan Koo, and Dongyeop Kang. 2023. CoEdIT: Text editing by task-specific instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5274–5291, Singapore. Association for Computational Linguistics.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Elvis Saravia. 2022. Prompt Engineering Guide. *https://github.com/dair-ai/Prompt-Engineering-Guide*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*, pages 20841–20855. PMLR.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

MN Team et al. 2023a. Introducing mpt-7b: A new standard for open-source, commercially usable llms.

MosaicML NLP Team et al. 2023b. Introducing mpt-30b: Raising the bar for open-source foundation models.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121, Online. Association for Computational Linguistics.

Marta Vila, M Antònia Martí, Horacio Rodríguez, et al. 2014. Is this a paraphrase? what kind? paraphrase boundaries and typology. *Open Journal of Modern Linguistics*, 4(01):205.

Jan Philip Wahle, Bela Gipp, and Terry Ruas. 2023. Paraphrase types for generation and detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12148–12164, Singapore. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Xinwei Wu, Junzhuo Li, Minghui Xu, Weilong Dong, Shuangzhi Wu, Chao Bian, and Deyi Xiong. 2023. DEPN: Detecting and editing privacy neurons in pretrained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2875–2886, Singapore. Association for Computational Linguistics.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. In *International Conference on Machine Learning*, pages 39818–39833. PMLR.

Wang Yidong, Yu Zhuohao, Zeng Zhengran, Yang Linyi, Heng Qiang, Wang Cunxiang, Chen Hao, Jiang Chaoya, Xie Rui, Wang Jindong, et al. 2023. Pandalm: Reproducible and automated language model assessment.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Datasets Details

Open-Domain QA:

WebQuestionSP (WebQSP) (Yih et al., 2016) consists of 4737 examples containing NL questions and answers with semantic parses. Originally it was split into 3,298 questions as a train set and 1,639 questions as a test set.

ComplexWebQuestion (CWQ) (Talmor and Berant, 2018) is a dataset for answering complex questions, which is constructed by programmatically generating more complex formal queries from WebQuestionsSP, then generating pseudo-NL questions for crowd workers to improve into NL questions. It was split into 2848/250/1639 examples for training, validation, and testing.

ComplexQuestions (CompQ) (Bao et al., 2016) contains 2,100 complex questions and was collected by mining a Bing search query log for questions with multi-constraint. The dataset was split into 1,300 training and 800 testing questions.

Commonsense Reasoning:

CommonsenseQA (CSQA) (Talmor et al., 2019) is a multiple-choice question-answering dataset that requires different types of commonsense knowledge to predict the correct answers. Each option includes one correct answer and four distractor answers. It was officially split into 9741/1221/1140 examples for training, validation, and testing.

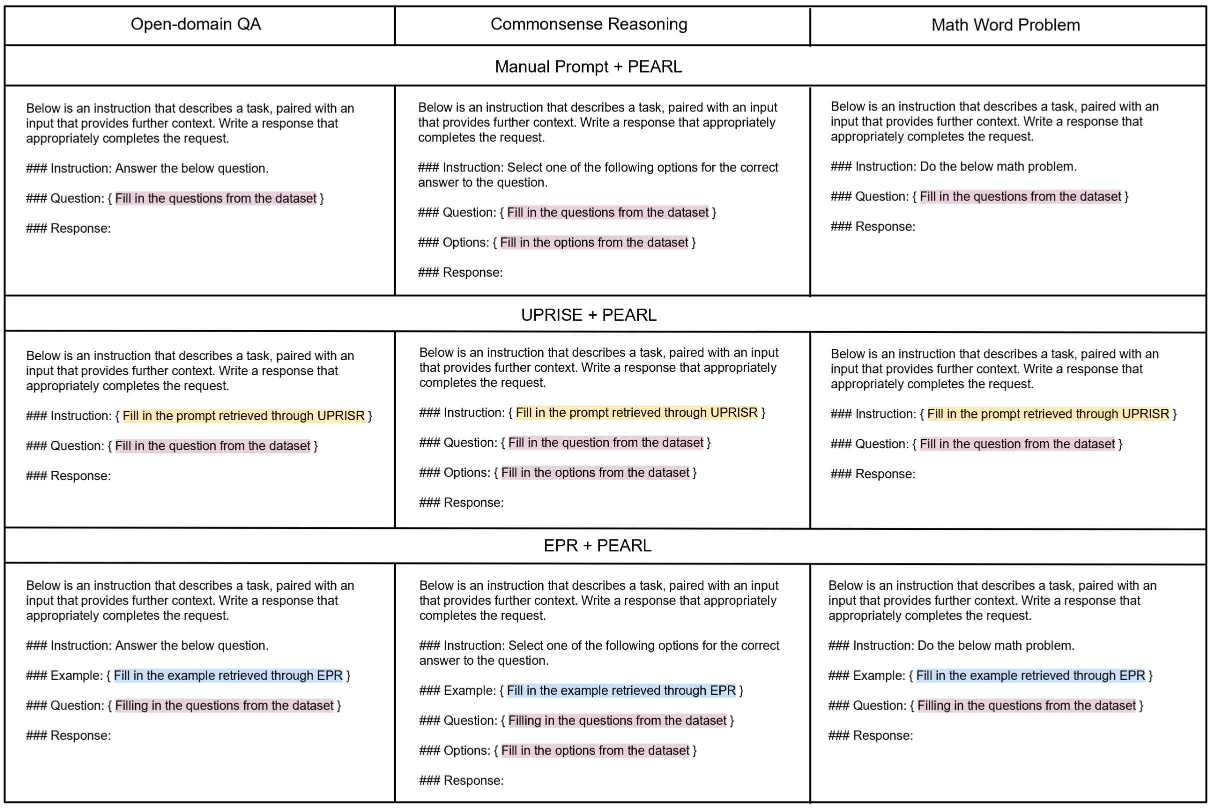
SocialQA (SiQA) (Sap et al., 2019) is a question-answering benchmark for testing social commonsense intelligence, which focuses on reasoning about people's actions and their social implications. Each option includes one correct answer and two distractor answers. It was split into 33410/1954/2059 examples for training, validation, and testing.

Math Word Problem:

SVMAP (Patel et al., 2021) is a challenge set for elementary-level Math Word Problems (MWP). An MWP consists of a short Natural Language narrative that describes a state of the world and poses a question about some unknown quantities. We follow Patel et al. (2021) to use the Combination of full MAWPS (Koncel-Kedziorski et al., 2016) and ASDiv-A (Miao et al., 2020) which has 3591 examples as the train set.

B Set Up for Comparative Experiment

LORA:



Open-domain QA	Commonsense Reasoning	Math Word Problem
	Manual Prompt + PEARL	
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Answer the below question. ### Question: { Fill in the questions from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Select one of the following options for the correct answer to the question. ### Question: { Fill in the questions from the dataset } ### Options: { Fill in the options from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Do the below math problem. ### Question: { Fill in the questions from the dataset } ### Response:
	UPRISE + PEARL	
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: { Fill in the prompt retrieved through UPRISR } ### Question: { Fill in the question from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: { Fill in the prompt retrieved through UPRISR } ### Question: { Fill in the question from the dataset } ### Options: { Fill in the options from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: { Fill in the prompt retrieved through UPRISR } ### Question: { Fill in the question from the dataset } ### Response:
	EPR + PEARL	
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Answer the below question. ### Example: { Fill in the example retrieved through EPR } ### Question: { Filling in the questions from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Select one of the following options for the correct answer to the question. ### Example: { Fill in the example retrieved through EPR } ### Question: { Filling in the questions from the dataset } ### Options: { Fill in the options from the dataset } ### Response:	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: Do the below math problem. ### Example: { Fill in the example retrieved through EPR } ### Question: { Filling in the questions from the dataset } ### Response:

Figure 7: Prompts for MPT-instruct-7B.

Hyper-parameters	
rank	64
lora_alpha	128
lora dropoute	0.05
target modules	Wqkv
training epoch	3

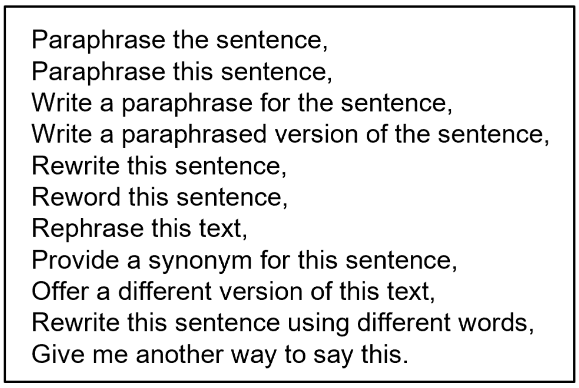
Table 7: Hyper-parameters.

We utilized the official repository[2] of MPT for LORA Tuning, employing training data consistent with the initial training set used for training the PEARL Generator. For specific parameter settings, please refer to Table 7.

UPRISE:

We employed the officially trained retriever and pre-constructed prompt pool. For the task settings in open-domain question answering and commonsense reasoning, as well as the parameter settings in retrieval, we adhered to the default settings provided in the official code repository[3].

EPR:



Paraphrase the sentence,
Paraphrase this sentence,
Write a paraphrase for the sentence,
Write a paraphrased version of the sentence,
Rewrite this sentence,
Reword this sentence,
Rephrase this text,
Provide a synonym for this sentence,
Offer a different version of this text,
Rewrite this sentence using different words,
Give me another way to say this.

Figure 8: Prompts for PEARL Generator.

We opted to train an example retriever for each task, using training data consistent with the initial training set employed for training the PEARL Generator. All parameter settings for training and retrieval were adhered to the default settings provided in the official code repository[4].

C Prompt of LLM

We followed the official instruction prompt templates to set up prompts for MPT-instruct-7B. The specific prompts used for different comparison ex-

[2]https://github.com/mosaicml/llm-foundry/

[3]https://github.com/microsoft/LMOps/tree/main/uprise

[4]https://github.com/ohadrubin/epr

periments across various tasks can be found in Figure 7.

D Prompt of PEARL Generator

In Section 4.3.4, when generating paraphrases with the PEARL Generator, we perturb it by adding some prompts to the input. For specific prompts, please refer to Figure 8.