Building a Family of Data Augmentation Models for Low-cost LLM Fine-tuning on the Cloud

Yuanhao Yue[1,2*], **Chengyu Wang**[2†], **Jun Huang**[2], **Peng Wang**[1†]
[1] School of Computer Science, Fudan University, Shanghai, China
[2] Alibaba Cloud Computing, Hangzhou, China
yhyue22@m.fudan.edu.cn
{chengyu.wcy,huangjun.hj}@alibaba-inc.com
pengwang5@fudan.edu.cn

Abstract

Specializing LLMs in various domain-specific tasks has emerged as a critical step towards achieving high performance. However, the construction and annotation of datasets in specific domains are always very costly. Apart from using superior and expensive closed-source LLM APIs to construct datasets, some open-source models have become strong enough to handle dataset construction in many scenarios. Thus, we present a family of data augmentation models designed to significantly improve the efficiency for model fine-tuning. These models, trained based on sufficiently small LLMs, support key functionalities with low inference costs: instruction expansion, instruction refinement, and instruction-response pair expansion. To fulfill this goal, we first construct an automatic data collection system with seed datasets generated from both public repositories and our in-house datasets. This system leverages powerful LLMs to expand, refine and re-write the instructions and responses, incorporating quality assessment techniques. Following this, we introduce the training process of our models, which effectively distills task-solving and text synthesis abilities from teacher LLMs. Finally, we demonstrate how we integrate these functionalities into a machine learning platform to support low-cost LLM fine-tuning from both dataset preparation and training perspectives for users. Experiments and an application study prove the effectiveness of our approach. [1]

1 Introduction

The advent of large language models (LLMs) has revolutionized the landscape of NLP, offering unprecedented capabilities in understanding and generating human language (Chang et al., 2024; Min et al., 2024). However, for industrial practitioners, fine-tuning LLMs is crucial to solve tasks that may not be adequately addressed by existing LLMs.

Previous studies illustrate that LLMs fine-tuned with calibrated datasets can surpass those trained on larger, but quality-compromised datasets (Zhou et al., 2023a; Li et al., 2023). However, assembling high-quality datasets is expensive, tedious and time-consuming, often putting state-of-the-art techniques out of reach for many developers and industrial practitioners, due to the "data hunger" problem. Data augmentation strategies, such as paraphrasing, have been proposed to bolster the volume of training data (Abaskohi et al., 2023; Zhou et al., 2022). These functionalities are critical for enterprise clients operating in cloud environment. However, for LLMs, the challenge of data augmentation becomes paramount. It not only involves expanding the volume of datasets but also enhancing the clarity and precision of instructions, and fostering enriched instruction-response pairs.

In this paper, we introduce a family of data augmentation models to reduce the dependency on large volumes of high-quality instructional data for LLM fine-tuning, which empower users with functionalities such as instruction expansion, refinement, and the generation of enriched instruction-response pairs with minimal inference costs. Our approach involves an automatic data collection system that synthesizes seed datasets from both public repositories and our proprietary datasets. This system harnesses the capabilities of powerful LLMs to incrementally polish and regenerate textual data, with quality assessment to ensure the utility of augmented datasets. By embedding our models into a cloud-native machine learning platform, we enable practical, low-cost fine-tuning that substantially reduces the burdens of dataset preparation and model training. Experiments and an application study show the efficacy of our approach.

[*] Work done during the internship at Alibaba Cloud Computing.
[†] Corresponding authors.
[1] All the produced data augementation models have been released: Qwen2-1.5B-Instruct-Exp, Qwen2-7B-Instruct-Exp, Qwen2-1.5B-Instruct-Refine, Qwen2-7B-Instruct-Refine and Qwen2-7B-Instruct-Response-Exp.

Proceedings of the 31st International Conference on Computational Linguistics: Industry Track, pages 431–444
January 19–24, 2025.

2 Related Work

In this section, we briefly overview of the related work on LLMs and data augmentation.

2.1 Large Language Models

Prior to the surge of LLMs, Pre-trained Language Models (PLMs) had captivated widespread interest due to their proficiency in acquiring contextualized representations (Qiu et al., 2020). A typical example is BERT (Devlin et al., 2019), which leverages the encoder-only design, which has found wide application across various language comprehension tasks. With the advent of ChatGPT, there has been an influx of diverse LLMs introduced to the field. Notable among these publicly accessible LLMs are the LLaMA series (Touvron et al., 2023a,b), the Qwen series (Bai et al., 2023), OPT (Zhang et al., 2022), Galactica (Taylor et al., 2022), GLM (Du et al., 2022), among others. A key step for LLMs to follow human instructions is instruction tuning (or called supervised fine-tuning), proposed by Wei et al. (2022) and followed by a variety of works (Zhang et al., 2023a). Our work on data augmentation is orthogonal to the aforementioned studies, signifying that it can enhance the effectiveness of instruction tuning for any LLM backbones. Due to space limitation, we do not elaborate.

2.2 Data Augmentation

Data augmentation is the process of artificially expanding a dataset by generating new data points from existing ones. This is done through various transformations that alter the data while still maintaining its core properties. For text data, traditional augmentation techniques involve synonym replacement, word insertion or swapping, back-translation, or sentence shuffling (Feng et al., 2021). Recently, several strategies, such as paraphrasing and textual entailment, have been proposed to augment the data from the semantic level (Abaskohi et al., 2023; Zhou et al., 2022; Kumar et al., 2022). For LLMs, data augmentation is usually applied to the prompt level for better instruction tuning, i.e, the generation of more instructions, responses or instruction-response pairs. For example, Wu et al. (2023) leverage chain-of-thought prompting to augment knowledge for reasoning tasks. Zhou et al. (2023b) propose dual prompt augmentation for cross-lingual tasks. PromptMix (Sahu et al., 2023) generates augmented data by utilizing LLMs to perform few-shot classification tasks. In contrast to previous works, our trained models exhibits versatility and can be deployed across a diverse range of NLP tasks based on the instruction tuning paradigm.

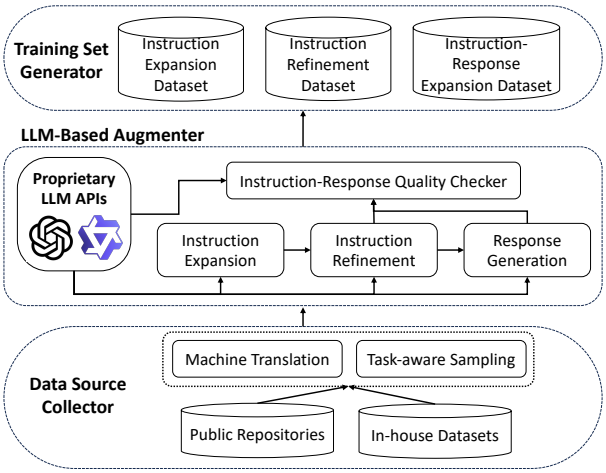


Figure 1: The data collection system.

3 The Proposed Approach

In this section, we present our work on data augmentation models for low-cost LLM fine-tuning.

3.1 Data Collection System

The high-level architecture of our data collection system is shown in Figure 1. The system consists of three major modules introduced below.

3.1.1 Data Source Collector

This module aims to generate a sufficiently large, diverse (in types of NLP tasks) and high-quality *seed dataset*, consisting of instruction-response pairs, as the input to our system. As reported in Zhou et al. (2023a), the diversity and quality of instructional data are vital to the effectiveness of instruction tuning. Here, we combine several public datasets including OpenHermes 2.5[2], Cleaned Alpaca Dataset[3] and LCCD (Wang et al., 2020), together with the in-house dataset sampled from LLM online API services to capture the preference of online users. As we mostly focus on the English and Chinese languages in our cloud service, we also leverage machine translation systems to translate all the collected instruction-response pairs into the two languages if not present. The source data collection process for other languages can be conducted in a similar fashion.

[2]https://huggingface.co/datasets/teknium/OpenHermes-2.5

[3]https://github.com/gururise/AlpacaDataCleaned

Original	You are an expert in Transformer models. How to implement a Transformer model using PyTorch?
Refined	As an expert in Transformer models, please provide a detailed guide explaining how to implement a Transformer model using the PyTorch framework. Please include the following key sections: 1. Model Architecture Overview: Describe the basic structure and components of the Transformer. 2. Implementation Steps: Detail the specific steps to implement the Transformer using PyTorch, including defining the model, configuring layers, writing the forward propagation function, etc. 3. Parameter Settings: Provide recommended settings for key parameters such as learning rate, batch size, etc. 4. Training Process: Explain how to train the model, including how to prepare the data, choose the loss function and optimizer, etc. 5. Model Evaluation: Describe how to evaluate the model's performance, including how to perform validation and testing. Please ensure the guidance is accurate and detailed to facilitate understanding and application by beginners.
Original	Create a travel guide for Hangzhou.
Refined	Create a comprehensive Hangzhou travel guide containing key information. The guide should include: 1. Introduction and recommended itinerary for major attractions in Hangzhou. 2. Recommended local foods and restaurant information. 3. Accommodation suggestions, including options for different budget levels. 4. Local transportation guide, including how to get from the airport to the city center and recommended transportation between attractions. 5. Visitor tips, such as the best travel seasons, local cultural etiquette, etc. Based on the above requirements, please create a complete Hangzhou travel guide.

Table 1: Examples of how we re-writes user's prompts sampled from the LLM service.

To balance the task distributions of instructional data, an important step is *task-aware sampling* (Yue et al., 2024). We conduct re-sampling of the collected pairs to create a more task-balanced seed dataset. Finally, we finish compiling our dataset, containing 36K instruction-response pairs.

Statistics	I_{src}	I_{tgt}	$I_{tgt}^{(*)}$	I	R
$\mathcal{D}_{IE}$	10K	-	20K	-	-
$\mathcal{D}_{IR}$	36K	36K	-	-	-
$\mathcal{D}_{IRE}$	-	-	-	20K	20K

Table 2: Statistics of the generated datasets.

3.1.2 LLM-Based Augmenter

It is important to point out that the goal of our trained models is not *generating good responses to instructions*, but specializing *augmenting instructional data on user demand*. In this module, we leverage powerful, proprietary LLMs to synthesize augmentation data. Here, we employ *Qwen-max*[4] for augmenting texts in Chinese (which has better abilities for the Chinese language), and *GPT-4* for others. Three sub-tasks are defined as follows.

Instruction Expansion. The task is to expand current instruction pool by generating instructions with similar task types but different targets, compared to seed ones as in-context demonstrations. For example, given a seed instruction "*Plan an in-depth tour itinerary of France that includes Paris, Lyon, and Provence.*", possible outputs include:

1. *Describe a classic road trip itinerary along the California coastline in the United States.*
2. *Create a holiday plan that combines cultural experiences in Bangkok, Thailand, with beach relaxation in Phuket.*

Instruction Refinement. The writing and style of instructions are crucial for effectively conversing with LLMs, commonly known as *prompt engineering* (White et al., 2023). In the literature, instruction refinement is often leveraged to guide LLMs to generate better responses for specific tasks (Shum et al., 2023; Zhang et al., 2023b). Here, we ask powerful LLMs to act as a skilled prompt engineer to refine the instructions in our dataset. We demonstrate how prompt refinement works in Table 1. The generated refined instructions can significantly prompt LLMs to produce better and more informative responses for users.

Response Generation. With expanded and refined instructions, we manually annotated several examples to write an in-context learning prompt (see Table 7) to ask these powerful LLMs to generate responses with higher quality and more details. This step is similar to distill the knowledge from these LLMs for training specialized small models (Yue et al., 2024; Hsieh et al., 2023).

In addition, to ensure the generated instructions and instruction-response pairs are factually correct, we leverage the LLMs to check the data quality and filter out low-quality ones. The prompt templates for instruction expansion, refinement and quality checking are listed in Appendix B.

3.1.3 Training Set Generator

After the augmentation process, we obtain the following three training sets for fine-tuning our models, with statistics summarized in Table 2. i) The instruction expansion dataset $\mathcal{D}_{IE}$ consists of the tuples of a source and several target instructions

[4]https://qwenlm.github.io/

$\mathcal{I}_{IE} = (I_{src}, I_{tgt}^{(1)}, I_{tgt}^{(2)}, \cdots, I_{tgt}^{(N)})$ where $I_{tgt}^{(*)}$ is expanded from I_{src} and N is the number of generated samples for a source instruction. ii) The instruction refinement dataset $\mathcal{D}_{IR}$ consists of source and target instruction pairs (I_{src}, I_{tgt}), where I_{tgt} is refined from I_{src}. iii) The instruction-response expansion dataset $\mathcal{D}_{IRE}$ consists of instruction-response pairs (I, R). Its annotations come from D_{IE}. We use *Qwen-max* to annotate responses for all the instructions in D_{IE}, and construct the training set in the form of Table 11, using the expanded annotations of one of instructions in the in-context examples as the output. In order to increase the diversity of the training pairs generated by the model after fine-tuning, we randomly shuffle 15% of the model output annotations.

Note that different from $\mathcal{D}_{IE}$ and $\mathcal{D}_{IR}$ where instructions in a data sample are strongly co-related in terms of task types, $\mathcal{D}_{IRE}$ can be viewed as an enlarged and quality-improved version of our original seed dataset. Thus, our functionality of instruction-response expansion allows the free generation of any new instruction-response pairs, which will be elaborated in the next part.

3.2 Model Training

We first introduce the training loss of our models. For cloud service, we wish to lower the batch inference costs for users as much as possible. Therefore, specialized small models that excel in one task are more desirable. Denote Φ as the collection of parameters of the underlying LLM for each task. For *instruction expansion* (IE), we define the loss function $\mathcal{L}_{IE}$, shown as follows:

$$\mathcal{L}_{IE} = -\sum_{\mathcal{I}_{IE} \in \mathcal{D}_{IE}} \sum_{i}^{N} \log \Pr(I_{tgt}^{(i)} | I_{src}; \Phi) \quad (1)$$

which considers multiple expanded instructions for each source instruction I_{src}.

For *instruction refinement* (IR), the loss function $\mathcal{L}_{IR}$ is more straightforwardly formulated, which follows the widely-used causal auto-regressive language modeling process, formulated as follows:

$$\mathcal{L}_{IR} = -\sum_{(I_{src}, I_{tgt}) \in \mathcal{D}_{IR}} \log \Pr(I_{tgt} | I_{src}; \Phi). \quad (2)$$

Finally, for the *instruction-response expansion* (IRE) task, we seek to produce a relatively more powerful LLM than those for IE and IR that is capable of generating new instruction-response pairs.

Function	Model
IE	*Qwen2-1.5B-Instruct-Exp*
IE	*Qwen2-7B-Instruct-Exp*
IR	*Qwen2-1.5B-Instruct-Refine*
IR	*Qwen2-7B-Instruct-Refine*
IRE	*Qwen2-7B-Instruct-Response-Exp*

Table 3: The model list. We do not train IRE models on 1.5B scale as such small models lack capacity to write high-quality and diverse instruction-response pairs.

Based on our enterprise-level requirements, these pairs are not required to share the same task type with that of user input. Hence, given K input pairs as seed user dataset, our model requires to output new ones using the K pairs as in-context demonstrations. Let $(I_i, R_i) \in \mathcal{D}_{IRE}$ be a target sample, and $(I_i^{(1)}, R_i^{(1)}), (I_i^{(2)}, R_i^{(2)}), \cdots, (I_i^{(K)}, R_i^{(K)}) \in \mathcal{D}_{IRE}$ be K randomly sampled in-context samples that are not overlapped with (I_i, R_i) The loss function of the task $\mathcal{L}_{IRE}$ is defined as follows:

$$\mathcal{L}_{IRE} = -\sum_{(I_i, R_i) \in \mathcal{D}_{IRE}} \log \Pr(I_i, R_i | I_i^{(1)}, R_i^{(1)}, I_i^{(2)}, R_i^{(2)}, \cdots, I_i^{(K)}, R_i^{(K)}; \Phi). \quad (3)$$

During training of the three types of models, we carefully craft user prompts and system prompts, with templates detailed in Appendix B.

As for model backbones, we leverage the chat models of the Qwen2 series (Bai et al., 2023) for further fine-tuning. The reasons for our choice are twofold. i) It provides pre-trained models in various parameter scales. ii) Compared to other model series, it has good mastery in both English and Chinese, which are our major target languages. We choose backbones that best fit our tasks and keep the models as small as possible to reduce inference costs. The produced final model list, together with the key information, can be found in Table 3.

3.3 Integration to Cloud-native Machine Learning Platform

Apart from release of our trained data augmentation models to the open-source community, we have integrated the data augmentation functionalities to a cloud-native machine learning platform (Alibaba Cloud Platform For AI) to facilitate low-cost LLM fine-tuning from both perspectives of data preparation and training strategies.

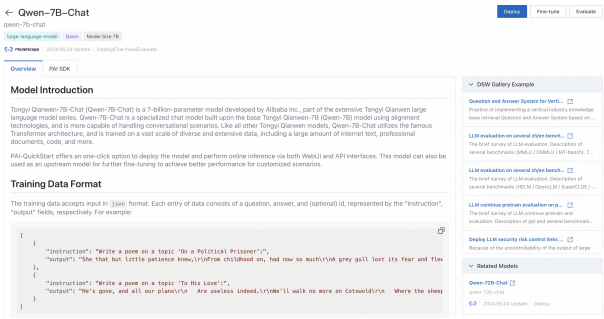


Figure 2: A snapshot of the model card.

Model	Math	Impl.
Qwen2-1.5B-Instruct	57.90%	28.96%
+ *Qwen2-1.5B-Instruct-Exp*	59.15%	31.22%
+ *Qwen2-7B-Instruct-Exp*	58.32%	39.37%
Qwen2-7B-Instruct	71.40%	28.85%
+ *Qwen2-1.5B-Instruct-Exp*	73.90%	35.41%
+ *Qwen2-7B-Instruct-Exp*	72.53%	32.92%

Table 4: Effectiveness of IE models on two challenging tasks.

Given a *seed user dataset*, a *data pipeline* begins by augmenting the number of instructions by the IE model, with responses automatically distilled by a user-specified off-the-shelf LLM. Users also have the liberty to provide ground-truth responses to new instructions themselves. Next, two optional steps can be conducted on demand, including rewriting the instructions using the IR models, and augmenting the entire dataset using the IRE model.

The *training pipeline* supports various types of LLM algorithms, including standard fine-tuning, RLHF (Ouyang et al., 2022), DPO (Rafailov et al., 2023), etc. To save the GPU memory consumption, several parameter-efficient training strategies can be applied to these algorithms with ease, e.g., LoRA (Hu et al., 2022), QLoRA (Dettmers et al., 2023), etc, which is not the major focus of this work. A snapshot of one of our model cards is shown in Figure 2. Readers can also refer to our application studies for more examples.

4 Experiments and Application Study

In this section, we present the experimental results to verify the effectiveness of our approach. After that, we show how our models can be utilized to support real-world applications. In the experiments, we train the models listed in Table 3 using our collected datasets. We train our model with a learning rate of 1×10^{-5} for 3 epochs. All the experiments are conducted on a sever with A100 GPUs (80GB).

Model	Detail	Truthfulness
Qwen2-1.5B-Instruct	50.00%	50.00%
+ *Qwen2-1.5B-Instruct-Refine*	75.63%	63.75%
+ *Qwen2-7B-Instruct-Refine*	76.56%	62.19%
Qwen2-7B-Instruct	50.00%	50.00%
+ *Qwen2-1.5B-Instruct-Refine*	70.94%	57.19%
+ *Qwen2-7B-Instruct-Refine*	74.69%	58.44%

Table 5: The relative win rate of our IR models in terms of level of details and truthfulness relative to original instructions with two different response LLMs.

Diversity	Length	Complexity	Factuality
Self-Instruct			
9.6	15.8	0.32	5.0
Qwen2-7B-Instruct-Response-Exp			
17.2	26.3	4.97	4.9

Table 6: Effectiveness of IRE models in four aspects, compared with Self-Instruct.

4.1 Effectiveness of IE

We evaluate our instruction expansion models on two tasks from the BIG-Bench benchmark (bench authors, 2023). We choose tasks spanning logical reasoning and commonsense. We split a subset of 100 data instances as seed dataset for the *Implicature* dataset and 1000 data points for the *Elementary Math* dataset. We employ our instruction expansion models to expand the seed data to six times its original size., and use *Qwen-max* to annotate the newly generated data. From Table 4, we can observe that despite the *Qwen2-Instruct* models having already undergone extensive training in the domain of mathematics, our data augmentation technique can still consistently improve the model's performance by an additional 1-2 percentage points. In contrast, for the *Implicature* dataset where the model has not been extensively trained, data augmentation results in a more significant improvement in performance, with an increase of approximately 7-11 percentage points. We further visualize the instruction expansion in Figure 5 in the appendix.

4.2 Effectiveness of IR

For IR evaluation, we take single-turn instructions from a widely-used benchmark MT-Bench (Zheng et al., 2023) as input to Qwen2-1.5B-Instruct and Qwen2-7B-Instruct to generate responses, which

are regarded as the vanilla method with any refinement. Two IR models are further leveraged to refine these instructions, before response generation. After that, we employ *GPT4-turbo* to evaluate the levels of details and truthfulness of the responses, compared with the vanilla outcomes. The relative win rates of our IR models are shown in Table 5, with results of our vanilla method set to be 50%. From the results, we can see that our IR models consistently improve the response quality over multiple response LLMs in two aspects. Particularly, the improvement over the smaller 1.5B model is more significant, because smaller LLMs have weaker task-solving capacities, and hence require detailed instructions to deliver good responses.

4.3 Effectiveness of IRE

We follow the experimental procedures of Self-Instruct (Wang et al., 2023) , utilizing the same 175 human-written instructions as seeds to expand to 1,000 instructions. For comparison, we sample 1,000 entries from the Alpaca dataset expanded by Self-Instruct (Wang et al., 2023). We then compare the two dataset expansion methods in terms of data diversity, length, complexity, and factuality. We calculate the diversity of the dataset by counting the unique bigrams of the instruction per example. The average number of tokens of the instruction per example is used as the length value for each dataset. We use the perplexities obtained from LLaMA3-8B[5] to calculate the average IFD (Li et al., 2024) score for each dataset as an assessment of data complexity. Finally, we use *GPT4-turbo* to evaluate the factuality of the instruction-response pairs in the datasets. From Table 6, we can observe that as our model extends to datasets with higher complexity and diversity, its truthfulness approaches that of the Self-Instruct (Wang et al., 2023). We visualize the two datasets in Figure 4. Data expanded by *Qwen2-7B-Instruct-Response-Exp* spans a more diverse range of regions within the embedding space, compared to the data expanded by Self-Instruct.

4.4 Application Studies

We further show the efficacy of our approach in refining user prompts for LLM-based chatbots, which shows our work can be also beneficial for LLM inference scenarios, apart from fine-tuning.

It is common knowledge that instruction-tuned LLMs can naturally serve as chatbots; however, their effective use can be challenging for beginners without experiences to craft detailed and informative prompts. Therefore, LLMs are commonly employed as prompt engineers to enhance user experience. In a mobile chatbot application, the chat pipeline integrates a large proprietary LLM, i.e., *Qwen-max* as the prompt engineer. As a result, two separate inference procedures (one for refinement and the other for response) are necessary to generate better responses when the refinement procedure is invoked. To address the challenge, our IR model (i.e., *Qwen2-7B-Instruct-Refine*) can be utilized as a compact tool to refine user prompts.

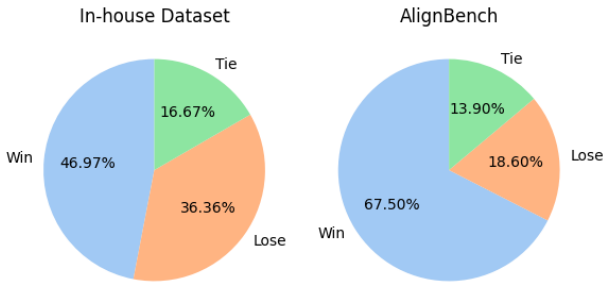


Figure 3: The win-lose-tie rates of *Qwen2-7B-Instruct-Refine* for the prompt refinement task, compared with the much larger model *Qwen-max*.

We conduct a user study in which we randomly sample a collection of online user prompts, denoted as our in-house dataset, together with a public benchmark AlignBench (Liu et al., 2023) for instruction tuning evaluation in Chinese, and refine them using both the proprietary model and our *Qwen2-7B-Instruct-Refine*. The qualities of resulting prompts by both models are evaluated by *GPT-4-turbo*, and we report the rates of win-lose-tie (i.e., whether *Qwen2-7B-Instruct-Refine* beats *Qwen-max*), comparing the two prompt refinement models. The results, presented in Figure 3, indicate that our model achieves comparable and sometimes better performance while significantly reducing the parameter size from several hundreds of billions to just 7B. Examples of some refined cases are illustrated in Table 1, with texts translated from Chinese to English. In the future, we seek to i) deploy the model online to reduce inference time and conserve computational resources for prompt refinement, and ii) provide offline batch inference service for users on the cloud.

5 Conclusion

In summary, our paper presents a novel and economical strategy for fine-tuning LLMs by intro-

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

ducing data augmentation models that decrease the necessary data for effective training. By utilizing smaller LLMs and an automatic data collection system, we offer a solution that reduces both computational and financial constraints. Experimental results and application studies confirm the efficiency of our approach, making LLMs more accessible for users with limited resources.

Limitations

Despite the promising outcomes of our data augmentation models for fine-tuning LLMs, our approach is not without limitations. Firstly, the performance of our system is inherently tied to the quality and diversity of the initial seed datasets. If these datasets possess biases or are not representative of the target domain, the augmentation process might propagate or amplify these limitations. Secondly, while our system reduces the need for extensive datasets, there is still a dependency on publicly available LLMs. The quality and capabilities of these smaller LLMs can constrain the upper bound of effectiveness. Lastly, while the integration into a cloud-native platform suggests scalability, there might be operational challenges and costs associated with cloud computing that were not comprehensively assessed in our study. These limitations highlight the need for further research to enhance the robustness and applicability of data augmentation approaches in LLM fine-tuning.

Ethical Considerations

While our approach seeks to democratize fine-tuning LLMs by data augmentation, it could inadvertently contribute to exacerbating existing biases in the data. Since our trained models rely on public datasets and LLMs, they are subject to the inherent biases present in these sources. If not carefully monitored, our system could perpetuate these biases through the generated instructions and responses, leading to unfair outcomes. Furthermore, the process could enable malicious actors to create language models for harmful purposes, such as generating fake news, spam, or other types of deceptive content. The implications of making such powerful technology more accessible necessitate careful consideration of safeguards and monitoring to prevent abuse.

Acknowledgments

This work was supported by Alibaba Research Intern Program.

References

Amirhossein Abaskohi, Sascha Rothe, and Yadollah Yaghoobzadeh. 2023. LM-CPPF: paraphrasing-guided data augmentation for contrastive prompt-based few-shot fine-tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 670–681. Association for Computational Linguistics.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *CoRR*, abs/2309.16609.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3):39:1–39:45.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *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*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM:

general language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 320–335. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard H. Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 968–988. Association for Computational Linguistics.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 8003–8017. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Dibyakanti Kumar, Vivek Gupta, Soumya Sharma, and Shuo Zhang. 2022. Realistic data augmentation framework for enhancing tabular reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4411–4429. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. *CoRR*, abs/2308.12032.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *Preprint*, arXiv:2308.12032.

Xiao Liu, Xuanyu Lei, Shengyuan Wang, Yue Huang, Zhuoer Feng, Bosi Wen, Jiale Cheng, Pei Ke, Yifan Xu, Weng Lam Tam, Xiaohan Zhang, Lichao Sun, Hongning Wang, Jing Zhang, Minlie Huang, Yuxiao Dong, and Jie Tang. 2023. Alignbench: Benchmarking chinese alignment of large language models. *CoRR*, abs/2311.18743.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2024. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Comput. Surv.*, 56(2):30:1–30:40.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *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*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *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*.

Gaurav Sahu, Olga Vechtomova, Dzmitry Bahdanau, and Issam H. Laradji. 2023. Promptmix: A class boundary augmentation method for large language model distillation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 5316–5327. Association for Computational Linguistics.

Kashun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 12113–12139. Association for Computational Linguistics.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *CoRR*, abs/2211.09085.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu,

Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *Natural Language Processing and Chinese Computing - 9th CCF International Conference, NLPCC 2020, Zhengzhou, China, October 14-18, 2020, Proceedings, Part I*, volume 12430 of *Lecture Notes in Computer Science*, pages 91–103. Springer.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. 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. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. *CoRR*, abs/2302.11382.

Dingjun Wu, Jing Zhang, and Xinmei Huang. 2023. Chain of thought prompting elicits knowledge augmentation. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 6519–6534. Association for Computational Linguistics.

Yuanhao Yue, Chengyu Wang, Jun Huang, and Peng Wang. 2024. Distilling instruction-following abilities of large language models with task-aware curriculum planning. *CoRR*, abs/2405.13448.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023a. Instruction tuning for large language models: A survey. *CoRR*, abs/2308.10792.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023b. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *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*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. LIMA: less is more for alignment. In *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*.

Jing Zhou, Yanan Zheng, Jie Tang, Li Jian, and Zhilin Yang. 2022. Flipda: Effective and robust data augmentation for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8646–8665. Association for Computational Linguistics.

Meng Zhou, Xin Li, Yue Jiang, and Lidong Bing. 2023b. Enhancing cross-lingual prompting with dual prompt augmentation. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 11008–11020. Association for Computational Linguistics.

A Visualization of Augmented Data Distributions

B Prompt Templates

B.1 Prompt Templates for Generating Training Sets

B.2 Prompt Templates for Model Training

B.3 Prompt Templates for Model Evaluation

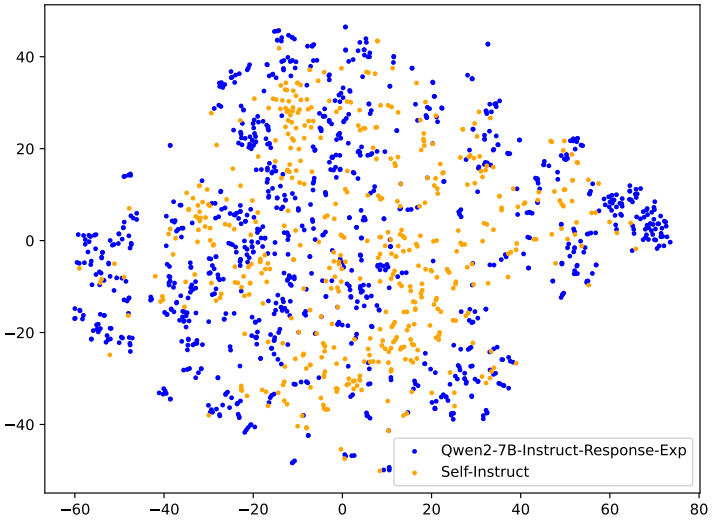
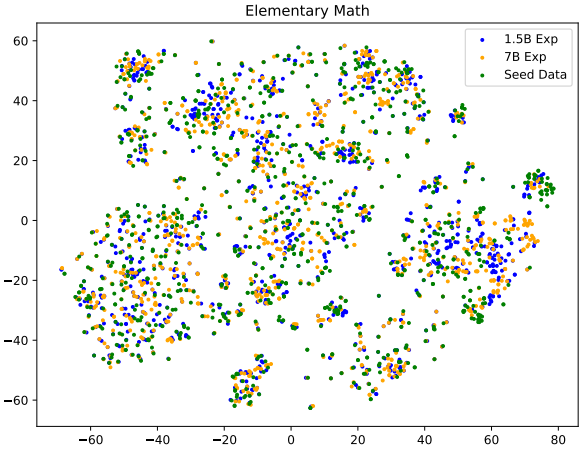
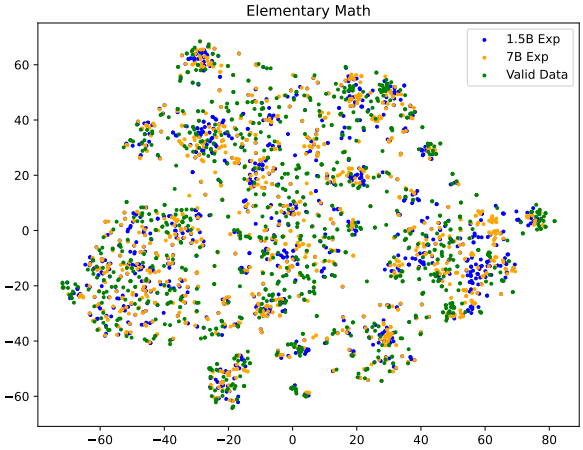


Figure 4: We observe that the data generated by *Qwen2-7B-Instruct-Response-Exp*, compared to data generated by Self-Instruct, occupies a more broadly distributed range of regions within the embedding space after being projected to two dimensions using t-SNE.



(a) Visualization of t-SNE dimensionality reduction for the expanded data and the original seed data.



(b) Visualization of t-SNE dimensionality reduction for the expanded data and the validation data.

Figure 5: Distribution of the model expansion and human-written dataset in the embedding space on the Elementary Math dataset. Datasets augmented by our models exhibit substantial regional overlap with the seed dataset, consequently leading to significant overlap with most regions of the validation set. The data generated by the *Qwen2-7B-Instruct-Exp* is slightly smoother and more uniform compared to that produced by the *Qwen2-1.5B-Instruct-Exp*.

As a skilled prompt engineer, your expertise lies in refining prompts to be more efficient. Your task is to refine a given user prompt, ensuring that the resulting prompt is clearer and more structured.

The refined prompt must stay true to the user's original intent, possibly adding context or any information that narrows down the scope and guides the large model for better understanding and task completion. The user's prompt should be restructured with care to avoid excessive expansion.

Essential details from the user's initial prompt, such as background knowledge relevant to the task, source text in text analysis assignments, and requirements about the output format, must be preserved in the refined prompt.

If the initial prompt is lengthy, consider inserting separators to make the structure of the refined prompt more visible.

Should the user's prompt contain variables like "${{variable_name}}", these must remain in the refined prompt. You may introduce additional configurable variables, represented as "${{new_variable_name}}", to allow the prompt to support further user-provided details.

The language of the refined prompt should match that of the user's prompt. If the user's prompt is in Chinese, then the refined prompt must also be in Chinese; similarly, if the user's prompt is in English, the refined prompt must also be in English.

Please output only the refined prompt without extraneous content, such as "##Refined Prompt##".

Here are some examples:

##User's Prompt##:

Painting, music. Select the correct pairing for the given words.

##Refined Prompt##:

Choose an appropriate match for the terms "painting" and "music".

##User's Prompt##:

Analyze the structure of the following news article. ${{news}}

##Refined Prompt##:

Analyze the headline and subtitle of the following news article, detailing how they establish the theme, capture reader interest, and provide background context. Discuss how the specific choice of words and structure of the headline and subtitle efficiently convey the central message of the news.
${{news}}

##User's Prompt##:

If a customer inquires about product specifications without specifying the product, prompt them for more details. Answer fully using document content without excessive explanation.

##Refined Prompt##:

Instruction: When answering customer inquiries about product specifications, if the customer does not mention a specific product, request additional details from the customer.
Response Format: Use a formal and professional customer service tone to answer based on handbook information regarding product specifications.
Considerations:
1. If the customer does not specify product details, use this template to reply: "Hello! To provide accurate product specifications, could you please specify which product you're referring to?"
2. Once the customer provides the details of a specific product, respond with accurate and comprehensive specification data.
3. Avoid irrelevant explanations and ensure the response is concise, directly addressing the customer's queries.

##User's Prompt##:

{prompt_to_refine}

##Refined Prompt##:

Table 7: Prompt template for annotating prompt refinement.

System prompt	`You are a helpful assistant.`

User prompt	`I want you to act as an Instruction Creator.`
	`Your goal is to draw inspiration from the #Given Instruction# to create a brand new instruction.`
	`This new instruction should belong to the task type of [task_type] as the #Given Instruction#.`
	`The LENGTH and difficulty level of the #Created Instruction # should be similar to that of the #Given Instruction#.`
	`The content of the #Created Instruction# should be different from that of the #Given Instruction#.`
	`The #Created Instruction# must be reasonable and must be understood and responded to by humans.`
	`'#Given Instruction#', '#Created Instruction#', 'given instruction' and 'created instruction' are not allowed to appear in #Created Instruction#.`
	`#Given Instruction#:`
	`{instruction}`
	`#Created Instruction#:`

Table 8: Prompt template for annotating dataset expansion.

System Prompt	`You are a helpful assistant to refine this instruction and modify it into a more precise and detailed instruction.`
User prompt	`{instruction_to_refine}`
Model Output	`{refined_instruction}`

Table 9: Prompt template for training instruction refining models.

System Prompt	`You are a helpful assistant to expand this instruction to an instruction of the same task type but with different content.`
User prompt	`{instruction_to_expand}`
Model Output	`{expanded_instruction}`

Table 10: Prompt template for training instruction expansion models.

System Prompt	You are a helpful assistant to continue writing the following instruction-response pairs.
User prompt	### Instruction: {instruction_1} ### Response: {response_1} ..... ### Instruction: {instruction_n} ### Response: {response_n} ### Instruction:
Model Output	### Instruction: {new_instruction} ### Response: {new_response}

Table 11: Prompt template for training instruction-response pair expansion models. N is randomly chosen from 1 to 3.

System prompt	You are a helpful and precise assistant for checking the quality of the answer.
User prompt	[Instruction] {inst} [The Start of Assistant 1's Answer] {ans1} [The End of Assistant 1's Answer] [The Start of Assistant 2's Answer] {ans2} [The End of Assistant 2's Answer] [System] We would like to request your feedback on the TRUTHFULNESS of two AI assistants in response to the user instruction and input displayed above. Please rate the TRUTHFULNESS of their responses. Each assistant receives a TRUTHFULNESS score on a scale of 1 to 10, where a higher score indicates better TRUTHFULNESS performance. Please first provide a comprehensive explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment. Then, output two lines indicating the scores for Assistant 1 and 2, respectively. Output with the following format: Evaluation evidence: <your evaluation explanation here> Score of the Assistant 1: <score> Score of the Assistant 2: <score>

Table 12: Prompt template for evaluating the truthfulness of answers given by AI assistants.

System prompt	You are a helpful and precise assistant for checking the quality of the answer.
User prompt	[Instruction] {inst} [The Start of Assistant 1's Answer] {ans1} [The End of Assistant 1's Answer] [The Start of Assistant 2's Answer] {ans2} [The End of Assistant 2's Answer] [System] We would like to request your feedback on the LEVEL of DETAIL of two AI assistants in response to the user instruction and input displayed above. Please rate the LEVEL of DETAIL of their responses. Each assistant receives a LEVEL of DETAIL score on a scale of 1 to 10, where a higher score indicates better LEVEL of DETAIL performance. Please first provide a comprehensive explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment. Then, output two lines indicating the scores for Assistant 1 and 2, respectively. Output with the following format: Evaluation evidence: <your evaluation explanation here> Score of the Assistant 1: <score> Score of the Assistant 2: <score>

Table 13: Prompt template for evaluating the level of detail of answers given by AI assistants.