FiNE: Filtering and Improving Noisy Data Elaborately with Large Language Models

Junliang He[◇♡] **Ziyue Fan**[◇♡] **Shaohui Kuang**[♡] **Xiaoqing Li**[♡]
Kai Song[♡] **Yaqian Zhou**[◇] **Xipeng Qiu**[◇*]
[◇]School of Computer Science, Fudan University
[♡]ByteDance
{jlhe22,zyfan22}@m.fudan.edu.cn
xpqiu@fudan.edu.cn

Abstract

Data is the lifeblood of large language models (LLMs). While the quantity of open-source data available for training LLMs is substantial, its integrity[1] often falls short. For instance, the open-source chat version of Yi-1.5-9B scores 5.20 on AlignBench, while the Chinese Alpaca-GPT4 version scores 4.12. This discrepancy makes it challenging for developers to create models that excel in downstream tasks and instruction following. Therefore, it is essential to improve data integrity. Currently, there are two mainstream methods for enhancing data integrity: data filtering and data augmentation. Due to the labor-intensive and time-consuming nature of performing these tasks manually, some of these efforts are now being undertaken by LLMs, owing to their high alignment with human preferences. However, we have found that performing data filtering or data augmentation with LLMs has limited effectiveness in improving data integrity. In this work, we propose FiNE (**F**iltering and **I**mproving **N**oisy data **E**laborately), a method that performs refined filtering and improvement of training data with LLMs. Using the data obtained through our method to train Yi-1.5-9B, the performance gap on AlignBench between our model and the open-source chat version is reduced from 1.08 to 0.35. Additionally, on HalluQA, our model surpasses the open-source chat version by 8.45[2].

1 Introduction

Large language models (LLMs), such as ChatGPT [3], GPT-4 (OpenAI et al., 2024), and Claude3 (Anthropic, 2024), have become the predominant approach in natural language processing due to their impressive performance. Training these LLMs involves three stages (Ouyang et al., 2022): pre-training, supervised fine-tuning (SFT), and reinforcement learning. Among these, SFT is more commonly performed, requiring significantly fewer resources than pre-training and serving as a reinforcement learning prerequisite.

Model	AlignBench ↑	HalluQA ↑
Yi-1.5-9B-Chat	5.20	45.11
Yi-1.5-9B-MOSS	4.00 Δ 1.20	31.78 Δ 13.33
Yi-1.5-9B-Alpaca-GPT4	4.12 Δ 1.08	46.00
Yi-1.5-9B-Firefly	3.05 Δ 2.15	22.89 Δ 22.22

Table 1: Performance of Yi-1.5-9B models trained on different datasets. Δ indicates the performance drop compared to the corresponding open-source chat model on the same benchmark. In this work, we use GPT-4-1106-preview as a judge.

The integrity of data is a critical determinant of SFT performance (Zhou et al., 2023). In the English domain, there already exist some high-integrity SFT datasets, such as UltraChat (Ding et al., 2023) and InfInstruct[4]. Meanwhile, in the Chinese domain and after we utilized fastText (Joulin et al., 2017) to filter out non-Chinese data, there is a substantial amount of open-source supervised fine-tuning data, such as Alpaca-GPT4 (48818 Chinese examples) (Peng et al., 2023), Firefly (1475561 Chinese examples) (Yang, 2023), and MOSS (456276 Chinese examples) (Sun et al., 2024). However, a significant performance gap remains on AlignBench (Liu et al., 2023) and HalluQA (Cheng et al., 2023) compared to their open-source chat counterparts, whose training data is not publicly available as illustrated in Table 1. Therefore, the integrity of these datasets needs improvement.

Currently, there are two mainstream approaches

[*]Corresponding author.

[1]Since quality is a dimension for assessing data, to avoid confusion, we use "integrity" to denote the overall "quality" of data.

[2]We will release our code and data at https://github.com/jlhe2000/FiNE

[3]https://openai.com/blog/chatgpt

[4]https://huggingface.co/datasets/BAAI/Infinity-Instruct

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 8686–8707

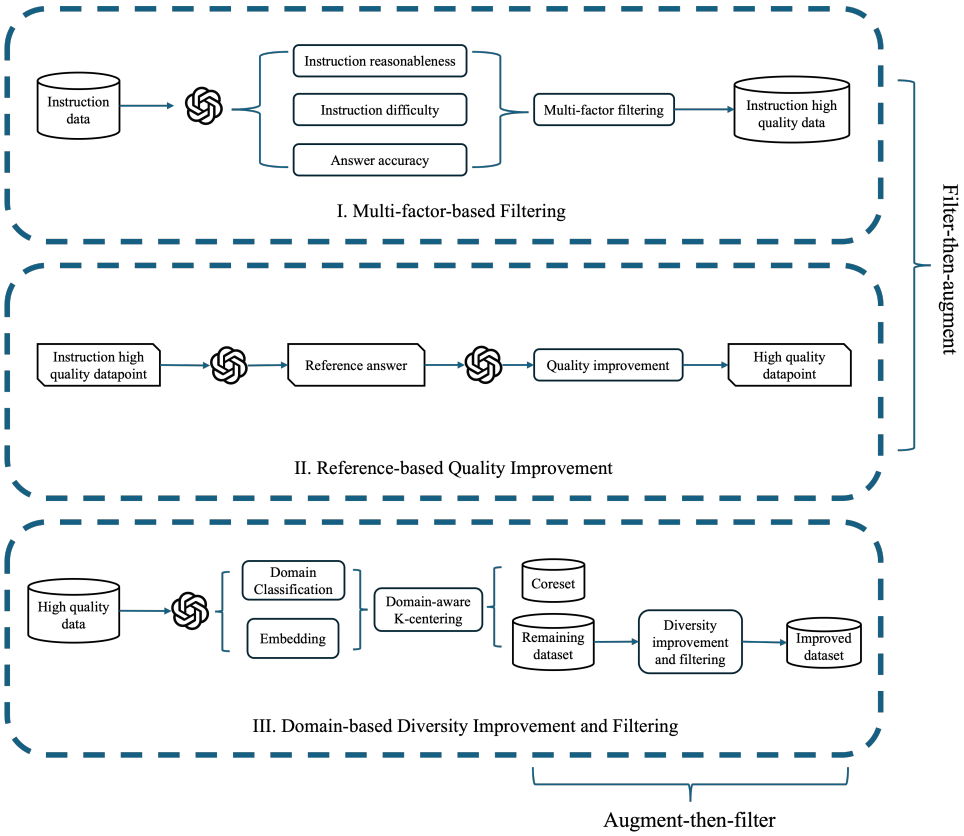

Figure 1: Pipeline of FiNE. (1) In the first stage, we use LLMs to evaluate data quality based on carefully designed factors, ensuring we do not overlook potentially usable data. (2) In the second stage, we first extract the knowledge of the data from the LLM and then have the LLM refer to this knowledge to evaluate and improve the quality of the answer. (3) In the final stage, we identify and expand non-diverse data to restore the diversity lost due to filtering. Then, we combine filtering methods to prevent data duplication, further ensuring diversity.

to improving data integrity. The first is data filtering (Du et al., 2023; Das and Khetan, 2024; Xu et al., 2023b; Xia et al., 2024; Chen and Mueller, 2024; Alcoforado et al., 2024; Chen et al., 2024), which involves removing low-quality data within a given dataset. The second is data augmentation (Xu et al., 2023a, 2024; Li et al., 2024a; Ding et al., 2023; Wang et al., 2024), which involves creating new data based on a high-quality seed dataset (which can also be an empty set). Due to the labor-intensive and time-consuming nature of performing these tasks manually, some of these efforts are now being undertaken by LLMs, owing to their high alignment with human preferences.

However, we encountered significant issues when using LLMs to filter the data, such as bias toward complexity and unreasonableness (3.1), opaque knowledge awareness (3.2), and quality-centric oversight (3.3). Also, we have observed that in data augmentation, LLMs may generate repetitive data (3.3). Therefore, performing data filtering or data augmentation alone is insufficient to improve data integrity, and these two approaches can be combined.

To address the issues above, we propose FiNE, which stands for **F**iltering and **I**mproving **N**oisy data **E**laborately with Large Language Models, to perform refined filtering and enhancement of noisy data. FiNE comprises three stages: Multi-factor-based Filtering (3.1), Reference-based Quality Improvement (3.2), and Domain-based Diversity Improvement and Filtering (3.3) as shown in Figure 1.

Our contributions are as follows:

(1) We combine data filtering and augmentation, proposing a more refined and rational process than previous methods.

(2) We open-source two high-integrity Chinese SFT datasets, the FiNE-Alpaca and the FiNE-Firefly, filtered and improved from Alpaca-GPT4 and Firefly, respectively.

(3) Experimental results indicate that fine-tuning LLMs with the high-integrity FiNE-Alpaca (15k) and FiNE-Firefly (120k, see Appendix A) yields su-

perior performance compared to using their entire original dataset (49k and 1500k), and significantly narrows the performance gap with their corresponding open-source chat models.

2 Related Work

There are three crucial evaluation criteria for data integrity: quality, diversity, and importance (i.e. necessity) (Qin et al., 2024).

2.1 Quality-Based Methods

Instruction Mining (Cao et al., 2023) developed metrics to predict validation set loss, serving as a basis for measuring data quality. MoDS (Du et al., 2023) employs a reward model to evaluate and filter low-quality data. Alpagasus (Chen et al., 2024) uses an LLM to score and filter data. Moreover, Alpaca-GPT4 (Peng et al., 2023) leverages GPT-4 to generate higher-quality responses.

2.2 Diversity-Based Methods

Currently, there are two mainstream methods for data diversity. The first method relies on discrete semantic diversity, such as UltraChat (Ding et al., 2023), which employs predefined meta-topics and subtopics; GLAN (Li et al., 2024a), which uses LLMs to continuously break down knowledge; and InsTag (Lu et al., 2023), which trains a tagging model to classify data.

The second approach employs continuous semantic diversity. For instance, DEFT (Das and Khetan, 2024) uses sentence embeddings to cluster the data and selects samples that are the farthest and closest to the cluster centers. Similarly, in the second stage of MoDS (Du et al., 2023), the K-Center greedy algorithm (Sener and Savarese, 2018) is applied to ensure diversity. Additionally, LIFT (Xu et al., 2023b) selects samples with the highest row variance from the dimensionally reduced sentence embedding matrix.

2.3 Necessity-Based Methods

Similar to quality assessment, MoDS (Du et al., 2023) employs a reward model to evaluate the performance of the initial model on non-diverse data. Instruction-Following Difficulty (Li et al., 2024b) has been proposed to measure the difficulty for a model to learn from data. LESS (Xia et al., 2024) uses gradient information from LoRA (Hu et al., 2021) to ensure necessity. CodecLM (Wang et al., 2024) measures necessity by the quality gap in response to the data between a target model and a strong model.

3 Methodology

Since necessity is model-dependent, FiNE focuses primarily on data quality and diversity. FiNE consists of three main steps. In the first two steps, we use a filter-then-augment approach to remove low-quality data and then improve the rest. In the third step, we use an augment-then-filter approach to increase data diversity while preventing duplicate data.

3.1 Multi-factor-based Filtering

The first issue is the bias toward complexity and unreasonableness. The LLM tends to assign lower scores to complex or unreasonable data, as shown in Appendix B.1. This issue happens because these data often have inaccurate or unhelpful answers.

To address this issue, we propose multi-factor-based filtering. Unlike approaches that use LLMs to assess only data accuracy and inspired by CriticLLM (Ke et al., 2024), we prompt the LLM to evaluate every data in the dataset based on the following three aspects:

Instruction Reasonableness refers to whether the instruction is fully answerable, aligns with common sense, and is safe without being offensive. **Instruction Complexity** refers to the difficulty of responding to the instruction, including the amount and rarity of required knowledge, the difficulty of analysis, or the complexity of reasoning. **Answer Accuracy**. Similar to Alpagasus(Chen et al., 2024), this evaluates the correctness of the response.

Let s_r be the instruction reasonableness score, s_d be the instruction difficulty score, s_a be the answer accuracy score and (I, A) be an instruction-answer pair from the original dataset. We prompt the LLM to obtain these three scores:

$$s_r, s_a, s_d = \text{LLM}(I, A; P_{\text{M}})$$

Where P_{M} is the prompt to get those three scores and can be found in Appendix C.

We retain the instruction if it meets any of these three conditions.

$$s_r < \theta_1 \wedge s_a \geq \theta_2 \quad (1)$$

$$s_r \geq \theta_3 \wedge s_d \geq \theta_4 \quad (2)$$

$$s_r \geq \theta_5 \wedge s_d < \theta_6 \wedge s_a \geq \theta_7 \quad (3)$$

Where $\{\theta_i\}$ are the thresholds, with i ranging from 1 to 7.

In other words, (1) includes instructions that may be unreasonable but ensure answer accuracy, contributing to the dataset and model's truthfulness and harmlessness. (2) includes data with reasonable but complicated instructions, even if the answers are not accurate (to be improved later), enhancing dataset complexity. (3) includes data with reasonable, not highly difficult instructions but accurate responses, ensuring fundamental quality and contributing to data diversity. According to the amount of data remaining after filtering, we set the thresholds of Alpaca-GPT4 as follows: $\theta_1 = 7$, $\theta_2 = 7$, $\theta_3 = 8$, $\theta_4 = 8$, $\theta_5 = 8$, $\theta_6 = 7$ and $\theta_7 = 9$.

3.2 Reference-based Quality Improvement

After the first stage, some answers may still be inaccurate. In this stage, we aim to improve their quality. While a stronger LLM could re-score and enhance the answers, we identified an issue: opaque knowledge awareness. During scoring, the LLM does not output intermediate steps in response to the instruction, making it unaware of its knowledge. This results in misjudgments in some data, as shown in Appendix B.2.

To address this challenge, we implement reference-based quality improvement. Rather than having the LLM score the data directly using scoring criteria, we first have the LLM act as an assistant, following the criteria to generate a step-by-step reference answer. We summarize this process as follows:

$$R = \text{LLM}(I; P_{\text{R}})$$

where R is the reference answer and P_{R} is the prompt based on the "3H" principle (Bai et al., 2022) to get R which is provided in Appendix C.

Then, we prompt the LLM to compare the original data answer with the reference answer based on the scoring criteria and select the answer that the LLM prefers as the final answer:

$$A_{\text{final}} = \text{LLM}(A_{\text{orig}}, R, I; P_{\text{A}})$$

where A_{final} is the final answer A_{orig} is original answer in the data and P_{A} is the prompt to compare A_{orig} with R.

In cases where the answers are deemed comparable, we select the answer with higher complexity, measured by the number of lines.

$$A_{\text{final}} = \max(A_{\text{orig}}, R, \text{by lines})$$

3.3 Domain-based Diversity Improvement and Filtering

After filtering and improving, we obtained a reasonably high-quality dataset. However, the LLM-based filtering processes do not consider diversity. This quality-centric oversight lacks diversity and balance, as shown in Appendix B.3. In Figure 5, the math ability data (less than 1k) is significantly less than other data (about 5k). In Figure 6, the bottom-right area contains sparsely distributed data compared to other sections. Training a model with unevenly distributed data may result in poor performance across different real-world scenarios (Shao et al., 2024). Therefore, we propose domain-based diversity improvement and filtering. This solution consists of three sub-steps: Domain-aware K-Center, Domain-aware Diversity Improvement, and Diversity Filtering. Each of the following sections will introduce these sub-steps.

3.3.1 Domain-aware K-Center

First, we aim to identify the data segments that exhibit the highest and lowest levels of diversity. Discrete semantics methods often misclassify multi-class or ambiguous data. While more flexible, continuous semantics methods suffer from unclear vector space distances influenced by various factors. Unlike previous work, we integrate continuous and discrete semantic-based methods for assessing data diversity and propose a novel approach called domain-aware K-Center.

Initially, for each instruction I, we prompt an LLM to assign it a category C_j:

$$C_j = \text{LLM}(I; P_{\text{C}})$$

Appendix C provides P_{C}, the prompt to get the domain of instruction.

We design a classification system with three categories: (1) **Mathematical Ability** (C_1): reasoning, mathematical, and code data; (2) **Question Answering** (C_2): knowledge-related questions; (3) **Text Generation and Others** (C_3): instruction-following and creative writing. For each category, we embed the instruction using text-embedding-3-large[5] and apply the K-Center algorithm to obtain the coreset and remaining data.

Let D_j denote the dataset in category C_j, S_j denote the coreset, and T_j denote the remaining data:

[5] https://platform.openai.com/docs/guides/embeddings/embedding-models

$$S_j = \text{KCenter}(D_j, \emptyset, k)$$

$$D_j = S_j \cup T_j$$

where $S_j \cap T_j = \emptyset$ and KCenter is described in Appendix F. We set $k = 0.2 \times |D_j|$ in this work.

We then merge the core sets and the remaining data from all categories to form the core set and the remaining data for the entire dataset.

$$S = \bigcup_{j=1}^{3} S_j, T = \bigcup_{j=1}^{3} T_j$$

This approach offers several advantages: (1) It ensures that data processed by the K-Center algorithm belong to the same domain, preventing domain-induced embedding distances and focusing on content diversity. (2) The K-Center algorithms for multiple categories can run in parallel, reducing the computational load due to fewer samples per category. Experimental results demonstrate the method's efficiency in identifying highly diverse core sets.

3.3.2 Domain-aware Diversity Improvement

After identifying highly diverse core sets and the remaining data, we aim to enhance the latter's diversity. Unlike the Wizard's uniform approach, we use varied techniques tailored to different instruction domains. This method is domain-aware diversity improvement.

Math Ability. We refer to WizardMath (Luo et al., 2023) and use the EvolInstruct (Xu et al., 2023a) to evolve math data through four iterations. We then incorporate these four iterations of data into the dataset.

Let $T_1^{(i)}$ denote the data for mathematical ability after the i-th iteration, where $i \in \{0, 1, 2, 3, 4\}$. The following description outlines the evolution process:

$$T_1^{(0)} = T_1$$

$$T_1^{(i)} = \text{LLM}(T_1^{(i-1)}; P_{\text{EI}}) \quad \text{for} \quad i = 1, 2, 3, 4$$

Where P_{EI} is the prompt used in EvolInstruct to get new math data.

We do not filter as this data segment extends depth, and the mathematical category is imbalanced. Instead, we add it directly to the dataset to achieve balance.

$$D' = (S \cup T) \cup \bigcup_{i=1}^{4} T_1^{(i)}$$

Q&A. For each reasonable Q&A data point, we instruct the LLM to ensure the new question assesses comprehensive knowledge, requiring responses to include several specific, parallel knowledge points.

$$T_2^{(1)} = \text{LLM}(T_2; P_{\text{QA}})$$

Appendix C provides P_{QA}, the prompt to get new Q&A data.

Text Generation and Others. Like Q&A, we require the LLM to ensure the fine-tuned model answers new instructions with difficulty, incorporating multiple constraints or processing/output requirements.

$$T_3^{(1)} = \text{LLM}(T_3; P_{\text{TG}})$$

Appendix C provides P_{TG}, the prompt to get new text generation data.

3.3.3 Diversity Filtering

We identified duplicate data in the Magpie[6] (Xu et al., 2024) and Wizard[7] (Xu et al., 2023a) datasets. Appendix B.4 provides examples. This duplication likely occurs because the LLM cannot access existing instructions when generating new instructions. To address this, we apply the K-Center algorithm to the newly generated Q&A and text generation data, using the existing dataset (post-math extension) as the existing set. We select k' data points farthest from the existing set and add them to form the final dataset.

Let D_{final} denote the final dataset:

$$T_{\text{new}} = T_2^{(1)} \cup T_3^{(1)}$$

$$D_{\text{final}} = D' \cup \text{KCenter}(T_{\text{new}}, D', k')$$

Considering the balance across domains, we set $k' = 0.18 \times |T_{\text{new}}|$.

4 Experiments

4.1 Models

We use the fast and powerful LLM for the initial phase, Doubao-pro-32k[8]. As the data volume decreased, we switched to GPT-4-1106-preview for subsequent experiments. For training base models, considering both data volume and efficiency, we

[6] https://huggingface.co/datasets/Magpie-Align/Magpie-Reasoning-150K

[7] https://huggingface.co/datasets/WizardLMTeam/WizardLM_evol_instruct_70k

[8] https://www.volcengine.com/product/doubao

Model	PA	CU	BT	MC	TW	CQA	RP	LR	CR	CL	TS
Yi-1.5-9B-Chat	5.25	4.69	5.82	5.34	5.75	6.58	5.98	4.12	4.73	5.68	**5.20**
Yi-1.5-9B-Alpaca-GPT4	4.77	4.64	5.06	3.70	4.56	5.45	4.73	3.04	3.37	4.87	4.12
Yi-1.5-9B-FiNE	5.35	5.21	4.85	4.71	5.40	6.39	6.04	3.62	4.16	5.54	<u>4.85</u>
Qwen2-7B-Instruct	6.15	5.79	5.40	5.57	5.96	7.11	6.74	3.74	4.66	6.19	**5.42**
Qwen2-7B-Alpaca-GPT4	5.23	5.05	4.85	4.30	4.81	5.53	5.21	3.25	3.78	5.11	4.45
Qwen2-7B-FiNE	5.40	4.84	5.47	5.19	5.19	6.53	5.96	3.61	4.40	5.56	<u>4.98</u>
GLM4-9B-Chat	5.94	6.26	5.62	5.16	5.65	7.13	6.70	3.99	4.57	6.22	**5.40**
GLM4-9B-Alpaca-GPT4	5.13	4.34	5.19	4.20	4.67	5.61	5.03	3.27	3.73	5.00	4.36
GLM4-9B-FiNE	5.59	4.90	5.76	5.30	5.15	6.34	5.28	3.62	4.46	5.50	<u>4.98</u>
InternLM2.5-7B-Chat	6.15	6.28	5.41	5.22	6.44	7.34	6.51	3.65	4.44	6.36	**5.40**
InternLM2.5-7B-Alpaca-GPT4	5.03	5.29	4.74	3.63	4.52	5.76	4.88	2.51	3.07	5.04	4.05
InternLM2.5-7B-FiNE	5.30	5.91	5.32	4.93	5.04	6.47	5.41	3.79	4.36	5.58	<u>4.97</u>

Table 2: Performance of models on AlignBench. Abbreviations: **PA** - Professional Ability, **CU** - Chinese Understanding, **BT** - Basic Tasks, **MC** - Mathematical Calculation, **TW** - Text Writing, **CQA** - Comprehensive Q&A, **RP** - Role Playing, **LR** - Logical Reasoning, **CR** - Chinese Reasoning, **CL** - Chinese Language, **TS** - Total Score. The highest scores are in **bold**, and the second highest scores are <u>underlined</u>.

selected Yi-1.5-9B (01.AI et al., 2024), Qwen2-7B (Yang et al., 2024), GLM4-9B (GLM et al., 2024) and InternLM2.5-7B (Cai et al., 2024). We use vLLM (Kwon et al., 2023) for efficient inference. See Appendix E for more training details.

4.2 Benchmarks

AlignBench (Liu et al., 2023) AlignBench is a comprehensive benchmark consisting of 683 samples for evaluating the alignment capabilities of LLMs in Chinese. It uses a rule-calibrated, multidimensional LLM-as-Judge approach with Chain-of-Thought (Wei et al., 2023) reasoning for reliable and interpretable evaluations.

HalluQA (Cheng et al., 2023) HalluQA is a benchmark consisting of 450 samples designed to evaluate hallucinations in LLMs. It features adversarial questions across various domains, including Chinese history, culture, and social phenomena. An automated LLM-based method assesses whether the model outputs hallucinate.

4.3 Main Results

The performance of the model trained on FiNE-Alpaca, evaluated using AlighBench, is presented in Table 2. Despite utilizing only a subset (10k) of Alpaca-GPT4 data (49k) and augmented data (5k), FiNE significantly enhances performance, narrowing the gap with open-source chat models. Figure 2 depicts the geometric distribution of the FiNE-Alpaca. The augment-then-filter method mitigates sparsity in the region where the x-axis ranges from 50 to 100 and the y-axis ranges from -100 to -50. This section presents the results of FiNE-Alpaca. For details on FiNE-Firefly, refer to Appendix A.

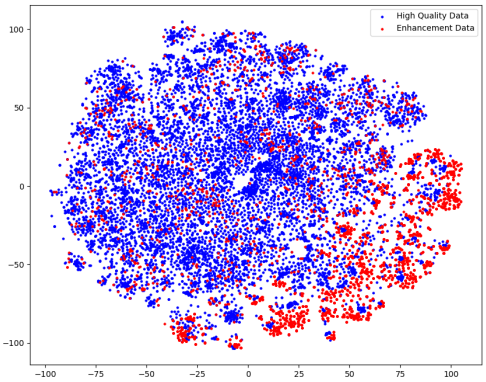

Figure 2: Geometric Distribution (t-SNE (Van der Maaten and Hinton, 2008) plot) of FiNE-Alpaca. High-quality data is the data after the first two stages, and enhancement data is the data augmented from the third stage.

Table 3 shows the performance on HalluQA. Models trained with FiNE-Alpaca outperform those trained on Alpaca-GPT4 and perform on par with or exceed those trained on its open-source chat versions. This result indicates FiNE's superior ability to retain and enhance truthful data.

5 Ablation

Effectiveness of Each Component. We conducted detailed ablation experiments to assess the effec-

Model	Misleading	Knowledge	Misleading-hard	Overall
Yi-1.5-9B-Chat	64.00	33.99	30.43	45.11
Yi-1.5-9B-Alpaca-GPT4	69.71	32.04	27.54	<u>46.00</u>
Yi-1.5-9B-FiNE	79.43	38.35	39.13	**53.56**
Qwen2-7B-Instruct	74.29	39.81	34.78	<u>52.44</u>
Qwen2-7B-Alpaca-GPT4	72.57	32.52	30.43	47.78
Qwen2-7B-FiNE	79.43	39.32	31.88	**53.78**
GLM4-9B-Chat	68.00	40.29	37.68	**50.67**
GLM4-9B-Alpaca-GPT4	61.14	37.38	21.74	44.22
GLM4-9B-FiNE	73.71	36.89	30.43	<u>50.22</u>
InternLM2.5-7B-Chat	69.14	44.17	40.58	<u>53.33</u>
InternLM2.5-7B-Alpaca-GPT4	70.29	34.47	23.19	46.67
InternLM2.5-7B-FiNE	76.57	42.23	31.88	**54.00**

Table 3: Non-hallucination rate of various models on HalluQA. The highest scores are in **bold**, and the second highest scores are <u>underlined</u>.

tiveness of each FiNE component. Table 4 shows that each part positively impacts performance.

Model	AlignBench	HalluQA
Yi-1.5-9B-FiNE	4.85	53.56
-DDIF	4.51	52.00
-RQI	4.40	46.67
-MF	4.12	46.00

Table 4: Performance after progressively removing each component in FiNE. Abbreviations: **DDIF** - domain-based diversity improvement and filtering, **RQI** - reference-based quality improvement, **MF** - multi-factor-based filtering.

Multi-factor-based Filtering vs. Alpagasus. We compared FiNE's first-stage filtering with an equivalent amount of Alpagasus (Chen et al., 2024) and random selection. Table 5 demonstrates FiNE's ability to filter out low-quality data while retaining high-quality data.

Model	AlignBench	HalluQA
Yi-1.5-9B-Alpaca-GPT4	4.12	46.00
+MF	**4.40**	<u>47.11</u>
+*Alpagasus*	<u>4.29</u>	42.89
+*Random Selection*	4.19	**47.33**

Table 5: Comparison between FiNE's multi-factor based filtering, Alpagasus, and random selection. The highest scores are in **bold**, and the second highest scores are <u>underlined</u>.

All Reference Answers vs. Preferred Answers. We compared the performance of using only generated reference answers versus LLM-preferred answers for enhancing answer quality, as shown in Table 6. The initial filtering stage retains some high-quality answers, making it necessary to compare reference answers with original answers.

Model	AlignBench	HalluQA
Yi-1.5-9B-MF	4.40	47.11
+*Preferred Answers*	**4.51**	**52.00**
+*Generated Answers*	4.47	50.44

Table 6: Comparison between using all LLM generated reference answers and LLM preferred answers. The highest scores are in **bold**.

Effect of Multi-factor Based Filtering Proportions. Using different thresholds, we studied how varying data retention after multi-factor filtering affects model performance (see Appendix D). Figure 3 shows that initially, more retained data improves performance due to higher diversity and quality. However, further increases lead to performance decline, likely due to inaccurate answers in some data.

Effect of Diversity Filtering Proportions. We examined how retained data volume after diversity filtering affects model performance (Figure 4). For AlignBench, performance initially improves with more data but later declines due to increased redundancy. In contrast, HalluQA shows steady performance improvement, stabilizing as the dataset grows, likely due to broader knowledge and more varied misleading scenarios in augmented instructions.

Domain-aware Diversity Improvement vs. Wizard Breadth Evolving. We compared domain-

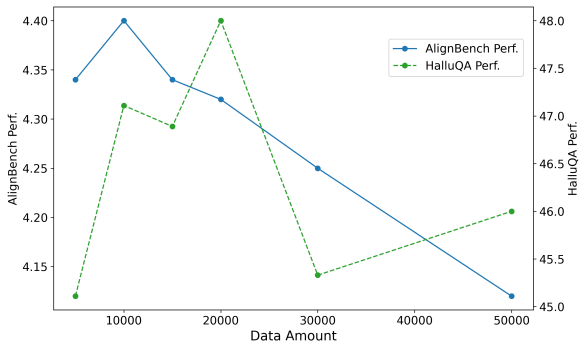

Figure 3: The relationship between the amount of retained data after multi-factor based filtering and model performance.

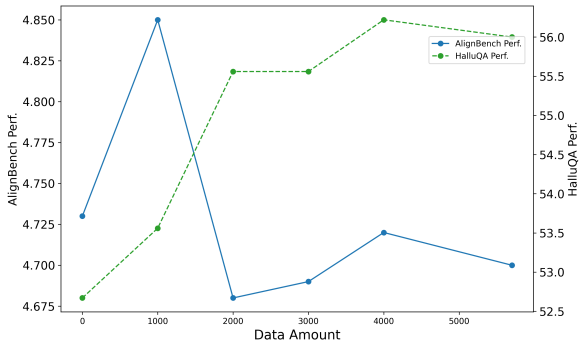

Figure 4: The relationship between the amount of data retained after diversity filtering and model performance.

aware diversity improvement with wizard breadth evolving. Both methods were applied to each Q&A and text generation data point after math augmentation. As shown in Table 7, field-specific augmentation results in more reasonable and higher-quality generated instructions.

Model	AlignBench
Yi-1.5-9B-MF+RQI	4.51
+DDI	**4.70**
+*Wizard Breadth*	4.63

Table 7: Comparison of domain-aware diversity improvement (**DDI**) with wizard breadth evolving. The highest scores are in **bold**.

Effect of Diversity Filtering. We also conducted experiments to verify the effectiveness of diversity filtering, as shown in Table 8. We can see that filtering diverse data based on vector distance after data augmentation is essential because not filtering and random filtering result in less performance improvement.

Domain-Aware K-Center vs. K-Center. We compared the Domain-Aware K-Center algorithm with the standard K-Center algorithm, as illustrated in Table 9. Domain-Aware K-Center more accurately identifies diverse data segments because, after reducing the large data volume, the model's performance drop trained on the coreset identified using Domain-Aware K-Center is less significant.

Model	AlignBench
Yi-1.5-9B-MF+RQI+DDI	<u>4.70</u>
+*Diversity Filtering*	**4.85**
+*Random Selection*	4.61

Table 8: Comparison of diversity filtering with random selection after domain-aware diversity improvement. The highest scores are in **bold**, and the second highest scores are <u>underlined</u>.

Model	AB Perf.	HQ Perf.
Yi-1.5-9B-MF+RQI	**4.51**	**52.00**
+*Domain-aware K-Center*	<u>4.23</u>	<u>49.11</u>
+*K-Center*	4.09	46.22

Table 9: Performance comparison between the Domain-Aware K-Center algorithm and the standard K-Center algorithm. The highest scores are in **bold**, and the second highest scores are <u>underlined</u>.

6 Conclusion

In this work, we propose a method to improve data integrity by combining data filtering and augmentation using large language models focusing on data quality and diversity. Initially, we employ a filter-then-augment approach, where low-quality noisy data is filtered based on meticulously designed metrics across multiple factors, followed by the remaining data quality improvement. Subsequently, we adopt an augment-then-filter strategy, identifying low-diversity noisy data first and then compensating for the diversity lost during filtering while simultaneously preventing redundancy in the data generated by the LLM during augmentation. When applied to Alpaca-GPT4 and Firefly to fine-tune LLMs, our method significantly improves their instruction-following capabilities and truthfulness, narrowing the gap with their corresponding open-source chat versions. This method and dataset enable downstream developers to create LLMs that perform well on downstream tasks and possess good instruction-following ability.

7 Limitations

Our approach, which leverages LLMs for data filtering and augmentation, imposes specific resource requirements, particularly when handling larger datasets. Furthermore, the effectiveness of our method is highly dependent on the capabilities of the LLMs utilized. Less powerful models may not accurately filter, respond to, or generate appropriate instructions.

8 Ethics Statement

In this work, we utilized two advanced LLMs, Doubao-pro-4k and GPT-4-1106-preview, to filter and enhance a new dataset based on open-source Chinese datasets. The models did not access the internet or invoke external tools; they generated data solely based on their internal distributions. To the best of our knowledge, the developers of both models have implemented stringent safety controls. Additionally, our prompts were crafted with a focus on helpfulness, truthfulness, and harmlessness. Therefore, we believe that the resulting data is safe and does not pose significant negative risks to society.

9 Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. This work was supported by the National Key Research and Development Program of China (No.2022CSJGG0801). The computations in this research were performed using the CFFF platform of Fudan University.

References

01.AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2024. Yi: Open foundation models by 01.ai. *Preprint*, arXiv:2403.04652.

Alexandre Alcoforado, Thomas Palmeira Ferraz, Lucas Hideki Okamura, Israel Campos Fama, Arnold Moya Lavado, Bárbara Dias Bueno, Bruno Veloso, and Anna Helena Reali Costa. 2024. From random to informed data selection: A diversity-based approach to optimize human annotation and few-shot learning. *Preprint*, arXiv:2401.13229.

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, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. 2023. Instruction mining: When data mining meets large language model finetuning. *Preprint*, arXiv:2307.06290.

Jiuhai Chen and Jonas Mueller. 2024. Automated data curation for robust language model fine-tuning. *Preprint*, arXiv:2403.12776.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. Alpagasus: Training a better alpaca with fewer data. *Preprint*, arXiv:2307.08701.

Qinyuan Cheng, Tianxiang Sun, Wenwei Zhang, Siyin Wang, Xiangyang Liu, Mozhi Zhang, Junliang He, Mianqiu Huang, Zhangyue Yin, Kai Chen, and Xipeng Qiu. 2023. Evaluating hallucinations in chinese large language models. *Preprint*, arXiv:2310.03368.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Preprint*, arXiv:2205.14135.

Devleena Das and Vivek Khetan. 2024. Deft: Data efficient fine-tuning for pre-trained language models via unsupervised core-set selection. *Preprint*, arXiv:2310.16776.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *Preprint*, arXiv:2305.14233.

Qianlong Du, Chengqing Zong, and Jiajun Zhang. 2023. Mods: Model-oriented data selection for instruction tuning. *Preprint*, arXiv:2311.15653.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

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. *Preprint*, arXiv:2106.09685.

Neel Jain, Ping yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Neftune: Noisy embeddings improve instruction finetuning. *Preprint*, arXiv:2310.05914.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2024. Critiquellm: Towards an informative critique generation model for evaluation of large language model generation. *Preprint*, arXiv:2311.18702.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng, Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, and Furu Wei. 2024a. Synthetic data (almost) from scratch: Generalized instruction tuning for language models. *Preprint*, arXiv:2402.13064.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024b. 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. *Preprint*, arXiv:2311.18743.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Instag: Instruction tagging for analyzing supervised fine-tuning of large language models. *Preprint*, arXiv:2308.07074.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *Preprint*, arXiv:2308.09583.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix,

Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff 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 Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Yulei Qin, Yuncheng Yang, Pengcheng Guo, Gang Li, Hang Shao, Yuchen Shi, Zihan Xu, Yun Gu, Ke Li, and Xing Sun. 2024. Unleashing the power of data tsunami: A comprehensive survey on data assessment and selection for instruction tuning of language models. *Preprint*, arXiv:2408.02085.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. *Preprint*, arXiv:1910.02054.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. *Preprint*, arXiv:1708.00489.

Yunfan Shao, Linyang Li, Zhaoye Fei, Hang Yan, Dahua Lin, and Xipeng Qiu. 2024. Balanced data sampling for language model training with clustering. *Preprint*, arXiv:2402.14526.

Tianxiang Sun, Xiaotian Zhang, Zhengfu He, Peng Li, Qinyuan Cheng, Xiangyang Liu, Hang Yan, Yunfan Shao, Qiong Tang, Shiduo Zhang, Xingjian Zhao, Ke Chen, Yining Zheng, Zhejian Zhou, Ruixiao Li, Jun Zhan, Yunhua Zhou, Linyang Li, Xiaogui Yang, Lingling Wu, Zhangyue Yin, Xuanjing Huang, Yu-Gang Jiang, and Xipeng Qiu. 2024. Moss: An open conversational large language model. *Machine Intelligence Research*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Zifeng Wang, Chun-Liang Li, Vincent Perot, Long T. Le, Jin Miao, Zizhao Zhang, Chen-Yu Lee, and Tomas Pfister. 2024. Codeclm: Aligning language models with tailored synthetic data. *Preprint*, arXiv:2404.05875.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *Preprint*, arXiv:2402.04333.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023a. Wizardlm: Empowering large language models to follow complex instructions. *Preprint*, arXiv:2304.12244.

Yang Xu, Yongqiang Yao, Yufan Huang, Mengnan Qi, Maoquan Wang, Bin Gu, and Neel Sundaresan. 2023b. Rethinking the instruction quality: Lift is what you need. *Preprint*, arXiv:2312.11508.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *Preprint*, arXiv:2406.08464.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Jianxin Yang. 2023. Firefly. `https://github.com/yangjianxin1/Firefly`.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

A Additional Results On Firefly Dataset

We further applied FiNE to another larger Firefly (Yang, 2023) dataset (1500k), resulting in a relatively smaller-scale FiNE-Firefly (125k). Subsequently, we utilized FiNE-Firefly to train Yi-1.5-34B and evaluated its performance on the benchmark.

A.1 Thresholds of FiNE on Firefly Dataset

According to the amount of data remaining after filtering, we set the thresholds of multi-factor-based filtering as follows: $\theta_1 = 7$, $\theta_2 = 8$, $\theta_3 = 9$, $\theta_4 = 7$, $\theta_5 = 9$, $\theta_6 = 7$ and $\theta_7 = 7$.

For domain-aware K-Center, the same as Alpagasus, we set $k = 0.2 \times |D_j|$. Then for domain-aware diversity improvement, we set $k' = 0.26 \times |T_{new}|$

A.2 AlignBench Results

We evaluated the performance of the Yi-1.5-34B model on AlignBench using the Firefly dataset and the FiNE-Firefly dataset. The results were compared with its open-source Chat version, as shown in Table 10.

Similar to FiNE-Alpaca, our approach, despite utilizing only a tiny portion of the original dataset (100k out of 1500k) along with 25k augmented data samples, successfully improved the model's performance from 3.39 to 5.27. Our method also reduced the performance gap with its open-source counterpart from 2.43 to 0.55.

A.3 HalluQA Results

Similarly, we also evaluated the performance on HalluQA, as shown in Table 11. Our approach significantly improved the model's truthfulness, increasing the performance from 23.33 to 62.89, an enhancement of nearly 40 percentage points and surpassing its corresponding open-source chat version.

B Examples

B.1 Examples show the bias toward complexity and unreasonableness

Table 12 gives examples that show the bias toward complexity and unreasonableness.

B.2 Examples show opaque knowledge awareness

Table 13 provides an example that shows opaque knowledge awareness.

B.3 Examples show quality-centric oversight.

Figure 5 and Figure 6 show the discrete and continuous distribution of instructions after the first two stages, respectively.

B.4 Repetitive Training Data in Magpie and Wizard

Table 14 and Table 15 show repetitive training examples of Wizard and Magpie, respectively.

C Prompts Used in This Work

Table 16 - Table 22 provide all prompts used in this work. We have translated all prompts into English.

Model	PA	CU	BT	MC	TW	CQA	RP	LR	CR	CL	TS
Yi-1.5-34B-Chat	6.64	6.12	5.82	5.44	6.45	6.79	6.98	4.89	5.16	6.47	**5.82**
Yi-1.5-34B-Firefly	4.37	4.38	4.54	2.99	3.93	3.00	4.23	2.42	2.71	4.08	3.39
Yi-1.5-34B-FiNE	5.73	5.52	6.00	5.12	5.91	7.18	6.06	3.82	4.47	6.07	5.27

Table 10: Performance of models on AlignBench. The highest scores are in **bold**, and the second highest scores are underlined.

Model	Misleading	Knowledge	Misleading-hard	Overall
Yi-1.5-34B-Chat	80.57	49.51	50.72	61.78
Yi-1.5-34B-Firefly	20.57	28.16	15.94	23.33
Yi-1.5-34B-FiNE	86.86	48.06	46.38	**62.89**

Table 11: Non-hallucination rate of models on HalluQA. The highest scores are in **bold**, and the second highest scores are underlined.

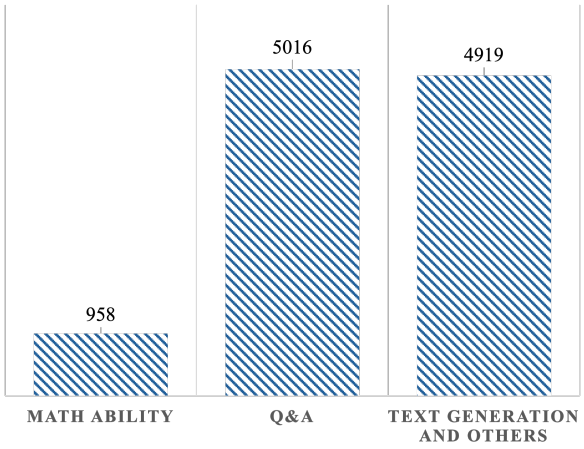

Figure 5: Domain Distribution of data after the first stage. We utilized GPT-4-1106-preview to classify the data.

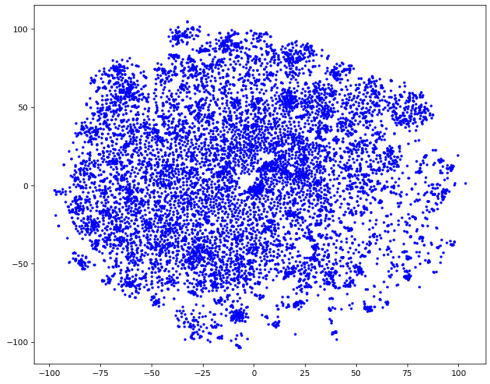

Figure 6: Geometric Distribution of high-quality data after first two stages.

D Different Thresholds For Different Amounts of Data

Table 23 shows the values of θ parameters for different data amounts.

E Training Details

During training, we used the default dialogue templates of each model and only calculated loss on the output. We set the learning rate to 1e-5 (1e-6 for GLM4), the global batch size to 64, the weight decay to 0.1, and the warmup ratio to 0.1. We conducted training over two epochs and configured the NEFTune (Jain et al., 2023) noise alpha to be 5. We use ZeRO (Rajbhandari et al., 2020) and FlashAttention (Dao et al., 2022) for efficient training. All experiments were conducted on four 40GB A800 GPUs.

F K-Center

Algorithm 1 shows the K-center algorithm used in this work.

Algorithm 1 K-Center (Sener and Savarese, 2018)

Input: dataset $\{x_i\}$, existing set S^0 and a budget k Initialize $S = S^0$

repeat

$u = \text{argmax}_{i \in [n] \setminus s} \min_{j \in s} \Delta(x_i, x_j)$

$S = S \cup \{u\}$

Until $|S| = k + |S^0|$

return $S \backslash S^0$

Instruction	Answer	Alpagasus Score	FiNE Score

What are the Einstein-Brillouin-Keller quantization conditions?	The Einstein-Brillouin-Keller quantization conditions, ... Overall, these conditions describe the motion of free particles in a quantized world, providing important guidance for understanding particle motion.	Score: 3
		
		Explanation: ... Therefore, there is room for improvement in accuracy, and a score of 3 is given considering all factors.
		
		(Filtered)
			Explanation: ... The question is clear, precise, and well-formulated, demonstrating a high level of depth. ... Rating:
			{
			"User Question Difficulty": 9,
			"User Question Reasonableness": 9,
			"Assistant Response Accuracy": 9
			}
			
			(Retained)

Render a 3D model of a house.	Sorry, as a language model, I cannot render 3D models. But I can provide some suggestions. ... If you are a beginner, I recommend searching for tutorials on 3D modeling and rendering techniques. These skills can help you quickly learn how to render a 3D model of a house.	Score: 1
		
		Explanation: The instruction is to render a 3D model of a house, ... Therefore, the accuracy can only be given a score of 1.
		
		(Filtered)
			Explanation: ... but it is not very specific or in-depth, making it moderately reasonable. ... The response is fairly accurate but lacks some specific operational steps and examples, leaving room for improvement.
			{
			"User Question Reasonableness": 6,
			"User Question Difficulty": 6,
			"Assistant Response Accuracy": 7
			}
			
			(Retained)

Table 12: Examples show the bias toward complexity and unreasonableness. We have translated the data into English.

Instruction	Answer	Direct Score	FiNE Result
Find the median of a given list of numbers. Input: 9, 2, 7, 4, 1, 6, 8	To find the median of this list of numbers, you first need to sort these numbers in ascending order. The sorted order is: 1, 2, 4, 6, 7, 8, 9 In this case, the median is 6, which is the ((n+1)/2)th number in the ordered list, where n is the number of elements in the list.	... However, the assistant made an error in calculating the median. The correct median should be the middle number of the list, which is 7, not 6. ... Therefore, I would give the following score: Score: [[1]] (Wrong)	LLM Answer: ... Step 1: Sort the list. The sorted list is: 1, 2, 4, 6, 7, 8, 9 Step 2: ... the median is the number at the middle position, which is the 4th number. ... Therefore, the median of the given list of numbers is 6. Compare: ... Result: [Comparable] (Right)

Table 13: Examples show opaque knowledge awareness. We have translated the data into English.

ID	Instruction	Reason for Repetition
Wizard Dataset		
63403	List the steps of a task. Making a cake	Instructions about
10454	Explain the process of baking a cake.	making a cake
759	Come up with three different ways to save money.	Instructions about saving money
8151	List three tips to save money.	
36526	Come up with three ways to increase customer loyalty.	Instructions about improving customer relations
35439	Give 3 ways to improve the customer experience	
32062	Name five U.S. presidents	
28090	Name 3 US presidents	Instructions about
9419	Name a famous US President	naming U.S. presidents
57288	Name three famous American presidents.	
9573	Design a web page.	Instructions about
3077	Design a website page layout.	web page design

Table 14: Repeated Training Data Example - Wizard Dataset

ID	Instruction	Reason for Repetition
Magpie Dataset		
56432	Generate a function in Python that takes a number and returns its factorial using recursion.	Instructions about recursive factorial in Python
86712	Write a short piece of recursive Python code to find the factorial of a number.	
140315	Create a recursive implementation of an iterative function to calculate the factorial of a given number.	
129620	Given the quadratic equation $ax^2 + bx + c = 0$, write a Python program to find the roots of the quadratic equation. Also, print the type of roots (real and distinct, real and equal or imaginary) based on the discriminant.	Instructions about solving quadratic equations in Python
6143	Given the quadratic equation $ax^2 + bx + c = 0$, write a Python function that takes three integers a, b, and c, and uses the quadratic formula to return the roots of the equation. Ensure that your function returns the roots as complex numbers, taking into account that the square root of the discriminant $b^2 - 4ac$ might be negative, and handle it appropriately.	
108901	Write a Python code to create a list of prime numbers from 1 to 100.	Instructions about finding prime numbers in Python
78996	Write a python program to find the prime numbers between 1 to 100	
117718	How many perfect squares less than 1000 have a last digit of 2, 3, or 4?	Instructions about perfect squares with specific last digits
9189	How many perfect squares less than 1000 have a ones digit of 2, 3 or 4?	
32974	1. How many perfect squares less than 100 have a ones digit of 2, 3, or 4?	

Table 15: Repeated Training Data Example - Magpie Dataset

As a fair judge, evaluate the quality of the conversation. Your evaluation should consider the following factors:

1. The challenging nature of user input

Score 1-2:

Common sense or basic knowledge in daily life. The question is very simple, almost no thinking or research is needed to answer.

Score 3-4:

Knowledge at the elementary or middle school level. The question is simple, but requires some basic common sense or foundational knowledge.

Score 5-6:

Knowledge from high school or university basic courses. The question is of medium difficulty, requiring some professional knowledge or understanding.

Score 7-8:

Knowledge from advanced university courses or professional fields. The question is difficult, requiring in-depth professional knowledge and understanding.

Score 9-10:

Professional knowledge at the graduate level or higher, possibly requiring interdisciplinary understanding. The question is very difficult, requiring highly professional knowledge and in-depth analysis.

2. The reasonableness of user input

Score 1-2:

The question lacks logic or reality basis, unable to provide useful answers. The question is unreasonable, vague or meaningless, difficult to answer.

Score 3-4:

The question has some logic, but lacks specific background or details, the answer may not be comprehensive. The question is partially reasonable, but some parts are unclear or lack background information.

Score 5-6:

The question has logic and reality basis, but could be clearer by providing more details. The question is reasonable, but some parts could be more explicit or specific.

Score 7-8:

The question has clear background and details, easy to understand and answer. The question is very reasonable, clear and explicit.

Score 9-10:

The question is not only clear and explicit, but also involves in-depth background and details, the answer requires professional knowledge and analysis. The question is completely reasonable, clear and deep.

3. The accuracy of the AI assistant's answer

Score 1-2:

Completely inaccurate, failed to answer the question. The answer is irrelevant or completely wrong, unable to provide useful information.

Score 3-4:

Partially accurate, but most of the content is irrelevant or wrong. The answer contains some correct information, but most of the content is irrelevant or has obvious errors.

Score 5-6:

Mostly accurate, but with some errors or omissions. The main content of the answer is correct, but lacks details or has some errors.

Score 7-8:

Very accurate, with only minor detail errors. The answer is almost completely correct, with only minor details possibly inaccurate or missing.

Score 9-10:

Completely accurate, comprehensive and error-free. The answer is not only completely correct, but also provides detailed explanations and relevant background information.

If you do not find obvious advantages or disadvantages, you should give a score of around 7. You should seriously consider the above guidelines, and then give your score (1-10) for the reasonableness of the user's question, the difficulty, and the accuracy of the assistant's reply, and avoid giving these scores without sufficient explanation. Do not let the length of the answer affect your evaluation.

First output your explanation, then strictly output the score in the following format: {"Reasonableness of User Question": Score, "Difficulty of User Question": Score, "Accuracy of Assistant's Reply": Score}

For example {"Difficulty of User Question": 7, "Reasonableness of User Question":7, "Accuracy of Assistant's Reply": 7}

Table 16: Prompt For Multi-factor Based Filtering

You are a helpful, harmless, and honest AI assistant based on a language model, with the following characteristics:

1. Broad knowledge:

Trained with a large amount of data, it can provide helpful high-quality information and suggestions in a wide range of fields.

2. Self-awareness:

It can realize that it is an offline language model that can only generate text and cannot interact with the real world. When encountering commands, intentions, or knowledge that cannot be executed or need to be queried, it needs to inform the user.

3. Safe and harmless:

It will not generate harmful, discriminatory, or inappropriate content.

4. Honest and transparent:

It will not deliberately mislead users and try to provide accurate information. When the user input is unreasonable or incomplete, it will inform the user.

5. Step-by-step thinking process:

When replying to the user, if necessary, please output your step-by-step thinking process.

User input:

{question}

Assistant reply:

Table 17: Prompt For Reference Answer

You are an expert in comparing answers. Given a question and two answers, you can compare which one is better. Please avoid any position and length biases and only analyze from the content perspective.

Scoring Factors

A helpful, harmless, and honest dialogue assistant based on a pre-trained language model, with the following features:

1. Knowledgeable:

After a large amount of pre-training data, it can provide high-quality information and suggestions in a wide range of fields. It can guarantee the correctness of the answers and will inform the user when there is a lack of information.

2. Self-awareness:

It can realize that it is an offline language model that can only generate text and cannot interact with the real world. It needs to inform the user when encountering commands, intentions, or knowledge that cannot be executed or need to be queried.

3. Safe and harmless:

It will not generate harmful, discriminatory, or inappropriate content.

4. Honest and transparent:

It will not deliberately mislead users, try to provide accurate information, and indicate when uncertain.

Requirements

1. First, analyze according to the scoring factors and give your step-by-step comparison process.

2. Then, give your comparison result. If answer 1 is better, output [Answer 1]. If answer 2 is better, output [Answer 2]. If the two answers are comparable, output [Comparable].

User Input

{current_in}

Answer 1

{answer1}

Answer 2

{answer2}

Strictly follow the format below and do not output any other content:

Comparison Process

xxx

Comparison Result

xxx

Table 18: Prompt For Comparison Between Original Answer and Reference Answer

You are a classification expert for the capabilities required for a command/dialogue. Here are some predefined categories (one per line):

Categories

1. Text Generation:

Generate coherent and meaningful text based on input data.

2. Mathematical Ability:

Understand and handle mathematical problems, including arithmetic, algebra, etc.

3. Knowledge Q&A:

Answer user's questions based on knowledge.

4. Others:

Besides the above abilities, including sentiment analysis, logical reasoning, machine translation, text classification, and other broad NLP applications.

Requirements

Given a command/dialogue, please categorize it according to the above categories based on the required capabilities. Here are the commands/dialogues (separated by "'). Each command/dialogue has only one category, please choose the one that is closest to it.

User Input

{data}

The command/dialogue is a whole, do not classify each command/dialogue separately, give your category based on the whole. The classification result can only be one of the above categories, do not output other content. Please write out your analysis process first, then give your answer, strictly follow the format below, do not output other content.

Analysis Process

xxx

Classification Result

xxx.

Table 19: Prompt For Domain Classification

Act in the capacity of a professional math teacher. Your goal is to accurately solve a mathematical word problem. To achieve this goal, you have one task.

Write out a detailed solution to the given problem.

You have two principles to do this.

Ensure the solution is step-by-step.

Ensure the final answer is correct.

Given problem: {input}

Only output your detailed solution, do not output any other content.

Table 20: Prompt For Answer of Augmented Math Instruction

You will receive an instruction. Please output a new instruction in Chinese according to the given instruction and the following principles. To do this, you have the following principles.
For Text Generation:
Ensure that the language model fine-tuned by instructions follows the new instructions with a certain degree of difficulty (the new instructions should include several restrictions or processing/output requirements).
For Q&A:
Ensure that the new instructions examine the rich knowledge of the language model fine-tuned by instructions (the reply to the new instructions contains several more specific parallel knowledge points).
Ensure that the new instruction examines different knowledge in the same field as the given instruction.
Ensure that the new instruction conforms to common sense.
Ensure that there is no lack of any input information in the new instruction. That is, the answer can be obtained only based on the new instruction.
Please do not include solutions in the new instruction.
[Start of given instruction]
{question}
[End of given instruction]
Output in the following format:
[New instruction]
xxx

Table 21: Prompt For Q&A And Text Generation Instruction Augmentation

You are a helpful, harmless, and honest AI assistant based on a language model that responds in Chinese with the following characteristics:
1. Broad knowledge: After a large amount of data training, it can provide helpful, high-quality information and suggestions in a wide range of fields.
2. Self-awareness: It can realize that it is an offline language model that can only generate text and cannot interact with the real world. When encountering commands, intentions, or knowledge that cannot be executed or need to be queried, it needs to inform the user.
3. Safe and harmless: It will not generate harmful, discriminatory, or inappropriate content.
4. Honest and transparent: It will not deliberately mislead users and try to provide accurate information. When the user input is unreasonable or incomplete, it will tell the user.
5. When replying to the user, if necessary, please output your step-by-step thinking process.
[User Input]
{input}
[Assistant Reply]

Table 22: Prompt For Answer of Augmented Q&A And Text Generation Instruction

Data amount	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7
5000	7	8	8	8	8	6	7
10000	7	7	8	8	8	7	9
15000	7	7	8	9	8	8	9
20000	7	7	8	7	8	7	8
30000	7	7	7	7	7	7	9

Table 23: Values of θ parameters for different data amounts.