Clustering and Ranking: Diversity-preserved Instruction Selection through Expert-aligned Quality Estimation

Yuan Ge[1*], **Yilun Liu**[2✉], **Chi Hu**[1], **Weibin Meng**[2], **Shimin Tao**[2], **Xiaofeng Zhao**[2],
Hongxia Ma[2], **Li Zhang**[2], **Boxing Chen**[3], **Hao Yang**[2], **Bei Li**[1], **Tong Xiao**[1,4], **Jingbo Zhu**[1,4]

[1] Northeastern University, Shenyang, China
[2] Huawei, Beijing, China
[3] Huawei Canada, Toronto, Canada
[4] NiuTrans Research, Shenyang, China

Abstract

With contributions from the open-source community, a vast amount of instruction tuning (IT) data has emerged. Given the significant resource allocation required for training and evaluating models, it is advantageous to have an efficient method for selecting high-quality IT data. However, existing methods for instruction data selection have limitations such as relying on fragile external APIs, being affected by biases in GPT models, or reducing the diversity of the selected instruction dataset. In this paper, we propose an industrial-friendly, expert-aligned and diversity-preserved instruction data selection method: **C**lustering **a**nd **R**anking (**CaR**). CaR employs a two-step process: first, it ranks instruction pairs using a high-accuracy (84.25%) scoring model aligned with expert preferences; second, it preserves dataset diversity through clustering. In our experiment, CaR efficiently selected a mere 1.96% of Alpaca's IT data, yet the resulting AlpaCaR model surpassed Alpaca's performance by an average of 32.1% in GPT-4 evaluations. Moreover, we find that data selecting is a consistent paradigm whether the pre-trained model is more capable or the model parameters scaling up. Our approach employs compact models with 550M parameters and incurs just 11.2% of the financial outlay of current methods, enhancing its industrial deployability.

1 Introduction

Language Models (LMs) acquire the capability to follow instructions through Instruction Tuning (IT) (Radford et al., 2019; Brown et al., 2020; Zhang et al., 2023), which aligns Large Language Models (LLMs) with critical human standards such as security, privacy, and legal compliance. Self-instruct proposes a novel methodology that utilizes LMs to construct IT datasets (Wang et al., 2022),

---
* Work done during an internship at Huawei.
✉ Corresponding author (liuyilun3@huawei.com).

Figure 1: Compares the performance of the proposed AlpaCaR model to established baseline models over four test sets. Our AlpaCaR achieves the best model performance with the smallest amount of instruction tuning data.

greatly improving the efficiency of instruction generation. Alpaca leveraged a similar strategy (Taori et al., 2023), utilizing text-davinci-003 to construct the Alpaca_52k dataset, and subsequent IT on LLaMA-7B model (Touvron et al., 2023) led to the creation of Alpaca.

Despite these advancements, the quality of instructions remains paramount over their quantity. Zhou et al. (2023) carefully curated 1,000 instructions, ensuring data quality and diversity by human being, resulting in LIMA model significantly outperforming the Alpaca. Nevertheless, creating high-quality instruction sets through manual annotation is both time-consuming and labor-intensive (Chiang et al., 2023). A promising approach to mitigate this challenge involves filtering a small subset of high-quality and diverse instructions from the vast amounts of existing instruction data.

Alpagasus (Chen et al., 2023) introduced a

IQS	Comet$_{Instruct}$	GPT-4	GPT-3.5
84.25%	72.44%	63.19%	57.48%
78.12%	45.00%	65.00%	56.25%

Table 1: Accuracy of the IQS, Comet$_{Instruct}$ and GPT models on test sets. Reflecting the alignment of the model with human preferences in the task of Instruction Pairs Quality Estimation. The second row presents results for instruction pairs sourced from the *IQE test set*, while the third row shows acc on instruction pairs from *Vicuna_80*, demonstrating the models' generalization to other distributions, see more details in Appendix C.1. The IQS and Comet$_{Instruct}$ model were fine-tuned as described in Appendix C.2, while the GPT model used prompts referenced in the Appendix B.2.

straightforward yet effective method that utilizes GPT-3.5-Turbo to filter roughly 9k instructions, surpassing Alpaca's performance. However, this approach overlooks data diversity, and GPT's evaluations rated 17.3% instruction pairs generated by text-davinci-003 above 4.5 and 74.9% above 4.0, demonstrating GPT's self-enhancement bias Zheng et al. (2023), rendering it unsuitable for assessing instructions generated by models within the same series. Therefore, more authentic human preferences should be used to filter instruction sets. Moreover, relying on fragile and expensive external GPT APIs limits Alpagasus in industrial deployment, especially in low-computation resource scenarios.

In this work, we propose an effective and efficient method for selecting instruction pairs — **C**lustering **a**nd **R**anking (**CaR**). CaR consists of two steps. The first is ranking through quality estimation on instruction pairs, where an expert-aligned scoring model (with 550M parameters only) achieves an accuracy of 84.25% with expert preferences. Then, a clustering step ensures the overall diversity of the dataset, minimizing potential capability gaps. Our contributions are summarized as follows:

- We introduce Instruction Pair Quality Estimation (IQE), a new stage before IT process which aims to use the assessment results of instruction datasets as an aid for the actual fine-tuning of language models and evaluation on benchmarks, reducing the time and computational expenses for model performance validation in IT process by over 90%.

- We propose a novel quality evaluation paradigm for IT dataset that is independent

of external APIs and aligns well with human experts' preferences. As shown in Table 1, our small Instruction pair Quality Scoring (IQS) model, compared to GPT-4, achieves a 21.05% improvement in aligning with human preferences for data quality.

- We propose CaR, an instruction selection method that aligns with expert insights and preserves diversity, showcasing significant enhancements in model performance and training efficiency. As shown in Fig. 1, CaR uses a small model to filter high-quality instruction data, achieving an average performance exceeding Alpaca by about 13.3% to 32.8% on the Alpaca_52k dataset using only a 1.96% subset of instructions. This implies a reduction of 98% in training time and resources.

- In section 5, experiments found that the data selecting paradigm is effective even with *more adequate pre-training* (LLaMA 1–LLaMA 3) or *model parameter scaling* (7B–30B). However, data selecting methods at *higher data quality*, such as Alpaca-GPT4 (Peng et al., 2023), are still challenging.

In addition, we released our code and models to facilitate future research and industrial endeavors[1].

2 Method

2.1 Motivation

Our work is motivated by the challenges of data quality in instruction tuning and the limitations of existing approaches.

From Quality Estimation to Instruction Pair Quality Estimation. Quality estimation is a crucial task in machine translation (MT), enabling the assessment of MT models' effectiveness and the selection of high-quality translations for specific purposes, such as manual post-editing. Similarly, LLMs' IT process faces the challenge of rapidly shifting from rare to abundant instruction pairs with inconsistent quality. Ensuring the quality of IT datasets presents a significant challenge, necessitating adjustments to the pre-trained model, executing inference on test datasets, and undergoing evaluation by LLM or human annotators. These processes are not only time-intensive but also demand considerable computational resources. To address this,

---

[1] https://github.com/IronBeliever/CaR

we propose a paradigm shift from evaluating model performance to assessing IT datasets via IQE. Our goal is to perform a coarse screening of a large number of instructions using IQE, followed by refining and selecting the optimal LLM with minimal datasets to reduce the overall computational cost associated with instruction filtering and verification.

GPT as a Judge Exhibits Systematic Bias. Researchers often use GPT preferences as a proxy for human preferences in scenarios requiring human feedback, due to time and cost considerations (Zhou et al., 2023; Rafailov et al., 2023; Dubois et al., 2023; Lee et al., 2023). However, GPT-4 has been shown to exhibit systemic biases in its evaluations, including positional bias, verbosity bias, and self-enhancement bias (Zheng et al., 2024a; Wang et al., 2023a). While researchers generally view Alpaca 52k as needing improvement (Alpaca-DataCleaned [2] ; Liu et al., 2023b), GPT's evaluations rated 9k instruction pairs above 4.5 and 39k above 4.0. Introducing more realistic human preferences for instruction filtering could further enhance model performance.

Instruction Diversity Inspires LLMs' Multi-tasks Capability. Recent studies have highlighted the importance of data diversity in improving the performance of LLMs (Zhou et al., 2023; Chen et al., 2023). Dong et al. (2023) found that combining training data from various tasks boosts LLMs' performance in low-resource scenarios. Inspired by these findings, we posit that integrating instructions from different tasks enhances LLMs' capabilities in low-resource settings. Consequently, ensuring the diversity of the IT dataset is paramount, particularly when dealing with large-scale models and limited high-quality data for each task.

2.2 Clustering and Ranking Method

Considering the aforementioned motivations, we propose a straightforward yet effective data selection framework, Cluster and Ranking, which integrates the dimensions of quality and diversity. Inspired by Zhou et al. (2023)'s work, we first select a subset that ensures the retention of a large number of high-quality instructions, then supplement a small number of high-quality instructions from each cluster to enhance data diversity while

preserving instruction quality. As illustrated in Fig. 2, the framework begins by evaluating the entire dataset using the IQS model, assigning a $score_i$ to each instruction $pair_i$. Subsequently, the cluster model is employed to partition all candidate instruction pairs into k clusters. Finally, all instruction pairs are sorted based on their scores, and the top n_1 pairs are selected; Within each cluster, the top n_2 pairs are chosen based on their scores. The resulting high-quality sub-dataset with preserved diversity is curated by deduplicating $n_1 + k * n_2$ pairs of instructions and is intended for the training of AlpaCaR.

Sections 2.3 and 2.4 provide a comprehensive discussion of the ranking and clustering methodologies implemented in CaR.

2.3 Single Instruction Pair Quality Estimation

To explore the IQE task, we adapt the Comet framework (Rei et al., 2020) and develop a suitable framework for leveraging expert preference. Our training data is derived from expert-revised dataset (Liu et al., 2023b), consisting of 3,751 instruction pairs from Alpaca_52k that were refined by linguistic experts to enhance fluency, accuracy, and semantic coherence between questions and responses. We categorize unedited instructions and responses from text-davinci-003 as *GPT Preference*, and expert-revised instructions as *Expert Preference*. To enable the model to discern features across these categories, we curated 2,541 markedly distinct instructions from the expert-revised dataset, ensuring an edit distance above a small threshold. These instruction pairs are then randomly allocated them into training, validation, and test sets following an 8:1:1 distribution.

Initially, we experimented with the translation ranking model architecture from the Comet framework to leverage the paired annotations in expert-revised better. In Fig. 10 (left), $Comet_{instruct}$ optimizes the model using instruction and input as anchors, minimizing semantic distance to human-preferred responses while maximizing distance to GPT-generated outputs. This approach achieves 72.44% accuracy on the test set but fails to fully leverage the improvements about *Input* made by experts. To address this, as illustrated in Fig. 10 (right), we retained the pre-trained XLM-RoBERTa large in $Comet_{instruct}$ and directly concatenated the instruction pair components to train the IQS model. As shown in Table 1, our IQS model outperforms GPT-3.5 (version: GPT-3.5-Turbo) and

---

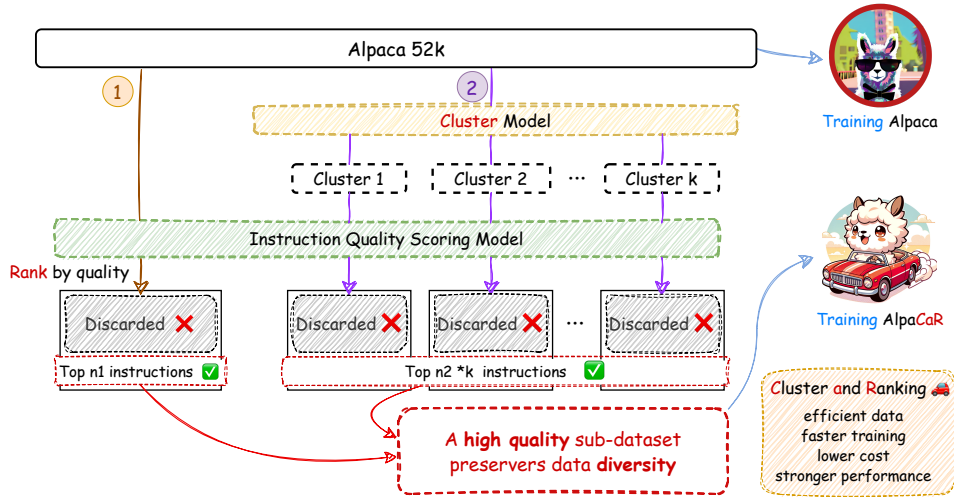[2] https://github.com/gururise/AlpacaDataCleaned

Figure 2: An overview of Cluster and Ranking (CaR) method. Unlike directly training Alpaca with the entire Alpaca_52k dataset, CaR first uses the IQS model to score all instructions (brown arrow). Then it selects the top n_1 instructions ranked by quality. Next, a clustering model (violet arrow) groups all instructions into k clusters, selecting n_2 from each. These are concatenated and deduplicated to form a diverse, high-quality sub-dataset for training AlpaCaR.

GPT-4 (version: GPT-4-1106-preview). Further analysis reveals that GPT-4 favors original instructions in 62.2% of incorrect cases, showing that even advanced GPT models often prefer GPT-aligned instructions. Additionally, GPT-4 struggles to recognize nuanced semantic changes made by experts in 37.8% of incorrect cases, revealing its difficulty in recognizing expert and nuanced semantic changes with minimal adjustments. Despite GPT-4's strong alignment with human preferences in most general tasks, its subpar performance on the expert-revised dataset highlights a subtle gap between expert preferences and GPT preferences.

2.4 Diversity

Within the instruction filtering framework, it is imperative to filter out a minimal subset of data from a vast array of instructions, resulting in a limited number of instructions per task. In such low-resource scenarios, Dong et al. (2023) has demonstrated that blending training data from various tasks enhances the LLMs' proficiency across different abilities. Intuitively, by assigning a task label to each instruction pair, we can preserve instruction pairs associated with a broader range of tasks, thereby facilitating cross-task instruction synergy and enhancing model performance. To determine task labels for instruction pairs, we evaluated manual labeling, classification models, and clustering models, selecting clustering for our study. Manual labeling, though more accurate, is labor-intensive

and less adaptable to various datasets. We hypothesize that instruction pairs within the same task are semantically close, allowing their distribution to be learned via classification models. Nonetheless, such models may struggle with flexibility when faced with out-of-domain data.

To enhance the method's versatility, we opted for an unsupervised clustering-based approach to preserve data diversity. A clustering algorithm can identify semantically close instruction pairs and form clusters for different tasks. Moreover, this choice allows for efficient adaptation to different datasets without retraining from scratch by forming new clusters when encountering out-of-domain instruction pairs.

Regarding the clustering methodology, we employ the k-Means algorithm. Initially, a sentence-transformers model is used to map sentences to a 384-dimensional dense vector space. Subsequently, semantic features are PCA-reduced to retain 95% of dimensions. Finally, by setting the number of clusters as $k = \sqrt{n/2}$, all 52k instruction pairs are clustered into 161 clusters. The diversity of the instruction sub-dataset is maintained by adjusting the quantity of instruction pairs within each cluster.

3 Experimental Setup

To compare AlpaCaR with other models, we obtain a single response for each test set sample using a fixed prompt (Taori et al., 2023). Judge LLMs are then compare responses generated by LLMs

Method	Num	Size	PandaLM			Vicuna			CoachLM			Self-instruct		
			WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$
Alpaca-PandaLM	52k	7B	1.224	49.4%	72.9%	0.288	8.8%	20.0%	0.867	28.7%	58.0%	1.075	42.9%	64.7%
Alpaca-cleaned	52k	7B	1.276	53.5%	74.1%	0.300	8.8%	21.3%	0.953	35.3%	60.0%	1.083	42.5%	65.9%
Vicuna	70k	7B	1.276	53.5%	74.1%	0.688	17.5%	51.3%	0.787	23.3%	55.3%	0.877	25.8%	61.9%
Alpaca	52k	7B	1.341	54.1%	80.0%	0.363	11.3%	25.0%	0.913	32.7%	58.7%	1.139	42.9%	71.0%
Alpagasus	9k	7B	1.324	54.1%	78.2%	0.463	13.8%	32.5%	0.807	25.3%	55.3%	1.123	44.4%	67.9%
AlpaCaR	1k	7B	**1.594**	**70.6%**	**88.8%**	**0.813**	**27.5%**	**53.8%**	**1.020**	**37.3%**	**64.7%**	**1.448**	**61.9%**	**82.9%**
Alpaca	52k	13B	1.365	56.5%	80.0%	0.363	8.8%	27.5%	0.940	30.7%	63.3%	1.155	45.2%	70.2%
Alpagasus	9k	13B	1.347	54.7%	80.0%	0.338	6.3%	27.5%	0.880	28.0%	60.0%	1.230	48.4%	74.6%
AlpaCaR	1k	13B	**1.535**	**65.9%**	**87.6%**	**1.025**	**37.5%**	**65.0%**	**1.153**	**44.0%**	**71.3%**	**1.357**	**56.3%**	**79.4%**
Alpaca	52k	30B	1.276	50.0%	77.6%	0.425	11.3%	31.3%	0.900	28.0%	62.0%	1.155	43.7%	71.8%
Alpagasus	9k	30B	1.382	57.1%	81.2%	0.438	8.8%	35.0%	0.920	30.0%	62.0%	1.214	46.8%	74.6%
AlpaCaR	1k	30B	**1.553**	**67.1%**	**88.2%**	**0.950**	**28.8%**	**66.3%**	**1.120**	**43.3%**	**68.7%**	**1.377**	**57.1%**	**80.6%**

Table 2: Comparative analysis of AlpaCaR and existing methods in the primary experiment. Winning rates are determined relative to the reference responses of the test sets, providing a quantitative measure of performance.

against each other or human reference responses, identifying their preferred responses. PandaLM, GPT-4 and human are used as judge, yielding consistent evaluation conclusions.

3.1 Test Datasets

To avoid confusion arising from the similarity in naming between models and datasets, we use the format "ModelName_DatasetSize" to represent datasets. Following previous methodologies, we assess four datasets: Self-instruct_252 (Li et al., 2023b), Vicuna_80 (Chiang et al., 2023), PandaLM_170 (Wang et al., 2023b), and CoachLM_150 (Liu et al., 2023b). This approach covers a broader range of instructions, minimizing evaluation bias.

3.2 Generations

For each test instruction, a single response is generated from each baseline model using LLaMA-Factory's default settings (Zheng et al., 2024b): temperature=0.95, top_p=0.7, top_k=50, no beam search, and a maximum token length to 512.

3.3 Evaluate Metrics

For each sample, the judge model receives a single instruction and two candidate responses. It labels the winning response or a tie if both stand out significantly. To address potential bias of LLM judges preferring specific positions, we tested the results twice by swapping the response order and define the final judgment based on:

- *win* : win twice, or win once and tie once
- *lose* : lose twice, or lose once and tie once
- *tie* : tie twice, or win once and lose once

We compute three types of winning rates: (1) WS, a winning score formulated as WS=1 + $\frac{\#win-\#lose}{\#all}$. (2) WR, which considers wins cases and is given by WR= $\frac{\#win}{\#all}$, where $\#all$ is the number of test set samples; (3) QS, a quality score that measures the ratio of responses reaching the reference level, formulated as QS= $\frac{\#win+\#tie}{\#all}$.

Evaluation Approach: (1) GPT-4 Turbo, currently the most powerful LLM widely used to replace manual responses quality assessments, with prompts designed by Chiang et al. (2023). However, this method faces limitations due to API dependency and inherent biases. (2) PandaLM, an open-source evaluation model that can be deployed locally, providing efficient LLM assessments (Wang et al., 2023b). Trained on 300k samples using GPT-3.5, it effectively mitigates biases and achieves 88.3% of GPT-4's evaluation capability. (3) Human, three experts with an average of 12.57 years of experience independently conducted comparisons based on the criteria in Appendix E After comprehensive consideration, we use the evaluation results of PandaLM to measure the model's instruction-following ability in most experiments, while some key principal experiments utilize GPT-4 and human for assessment. The prompt for GPT-4's evaluation is designed by Chiang et al. (2023), as detailed in the Appendix B.1.

4 Results and Analysis

In this section, we compared AlpaCaR with baseline models, including Alpaca, Alpaca-PandaLM, Alpaca-cleaned, Alpagasus, and Vicuna. We replicated all baseline models at a 7B scale and demonstrated the superiority of AlpaCaR at 13B and 30B scales.
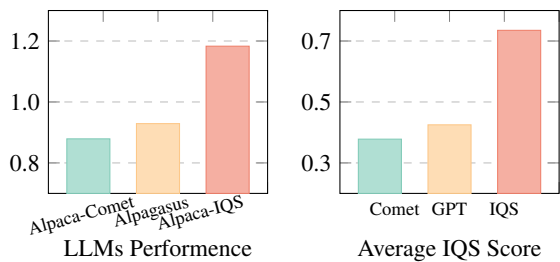
Figure 3: Consistency between IQS scores and the performance of LLMs.

Figure 4: Model performances with varying n_1.

Figure 5: Performances with varying n_2.

4.1 Comparison with Baselines

We conduct a comparative analysis of two established baseline LLMs, Alpaca and Vicuna, which were fine-tuned using 52,000 text instructions through text-davinci-003 and 70,000 ChatGPT dialogues, respectively. Furthermore, we explore three models that advance upon Alpaca: Alpaca-PandaLM and Alpaca-cleaned, which employ instructional enhancement methods, and Alpagasus, which incorporates an instruction filtering method. All models were trained with identical hyperparameter settings. As delineated in Table 2, AlpaCaR, at the 7B scale, outperforms not only the foundational models of Alpaca and Vicuna but also Alpaca-PandaLM, Alpaca-cleaned, and Alpagasus. Overall, AlpaCaR achieves significant performance improvements over Alpaca across the 7B, 13B, and 30B scales, validating the efficacy of the CaR method. The notable performance gains of AlpaCaR, accomplished with reduced data usage compared to Alpagasus, underscore the importance of leveraging high-quality human preferences and data diversity in enhancing model performance.

4.2 Reliability of IQE Results

To verify whether the IQE results genuinely reflect the performance of LLMs after IT, we examined the correlation between scores given by the IQS model and the performance of fine-tuned LLMs on test sets. Given that Alpagasus obtained 9k instructions rated above 4.5 using GPT-3.5-Turbo, we similarly selected the top 9k instructions ranked by IQS model and Comet model. We then calculated the average score for the three IT sub-datasets using the IQS model, fine-tuned LLaMA-7B, and tested its performance by averaging models' winning scores on four datasets against reference. As illustrated in Fig. 3, the average IQS score and the fine-tuned model's performance are generally consistent, indicating that IQE results can approximately reflect
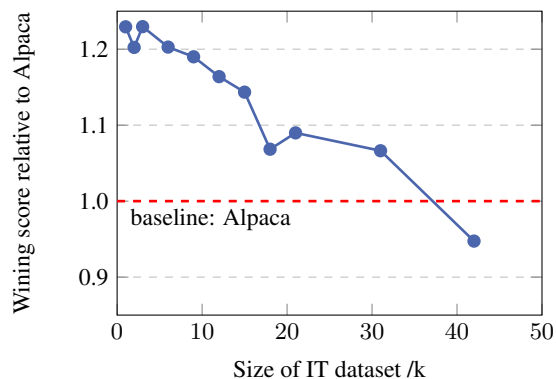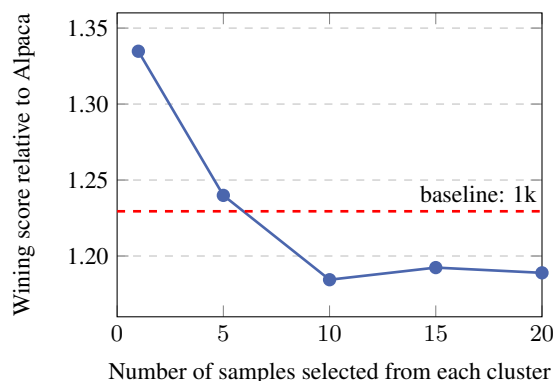
the performance of LLMs after fine-tuning.

4.3 Ablation Study

Quality Dimension. To illustrate the significance of data quality, we employed the IQS model's score to rank 52,000 instructions. Subsequently, we extracted subsets of the top 1,000, 2,000 and up to 42,000 instructions to train LLaMA-7B. In Fig. 4, the horizontal axis represents the size of instruction dataset, where a higher count signifies more instructions of relatively lower quality, while the vertical axis shows the winning score relative to Alpaca. The results indicate that models trained with selected data generally surpass the one trained with the entire dataset. As more instructions of relatively lower quality are included, the performance of the LLM generally declines. Remarkably, the model approaches its optimal performance with a mere 1,000 high-quality IT data. Therefore, in the CaR method, we select $n_1 = 1000$ instructions to ensure the chosen IT sub-dataset is of high quality.

Selection of n_2: Trade-off between Quantity and Quality. We compared the number of samples selected from each cluster after k-means clustering.

Method	Vicuna			Self-instruct		
	WS↑	WR↑	QS↑	WS↑	WR↑	QS↑
40×4	0.625	20.0%	31.3%	1.226	48.4%	61.3%
80×2	0.600	18.8%	30.0%	1.290	52.4%	64.5%
160×1	0.688	23.8%	34.4%	1.365	59.5%	68.3%

Table 3: Ablation on Diversity: Models with more diverse instruction sets perform better. (160×1 means 1 highest IQS-scored sample per 160 clusters)
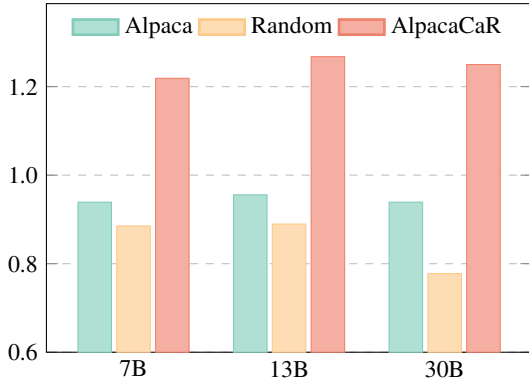

Figure 6: Compare AlpaCaR with baselines, including Alpaca and randomly selected 1k instructions.

Fig. 5 demonstrates that, compared to using only 1k high-quality data selected by IQS model, the CaR method enhances performance when a small number of samples (up to 5) are selected from each cluster. Selecting too many samples can negatively impact the overall quality of the IT sub-dataset and the performance of the LLMs. Moreover, the CaR method achieves nearly optimal performance by selecting $n_2 = 1$ sample from each cluster, thus enhancing the diversity of the IT sub-dataset.

Importance of Diversity. An ideal IT dataset should encompass a rich variety of data, but determining the optimal number of instructions per cluster required for the model to effectively correspond to the task remains a challenge. We designed experiments to demonstrate the importance of diversity and explore values of n_2, the trade-off between the number and quality of samples per cluster.

Designing strict ablation experiments in this context is challenging due to the difficulty in ensuring consistent instruction set quality while maintaining the same number of instructions. To explore this, we established three experimental groups with increasing diversity (baseline: reference response). In Table 3, the winning rates on the Self-Instruct and Vicuna test sets show that models with more diverse instruction sets perform better.
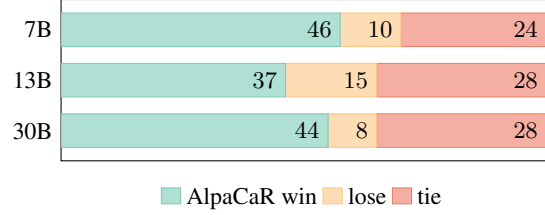
Figure 7: GPT-4 result on Vicuna_80 dataset: AlpaCaR vs. Alpaca.

4.4 Compare with Random & GPT-4 Result

Fig. 6 presents the results of ablation experiments, revealing that randomly selecting 1,017 instruction pairs from 52k dataset leads to a decrease in model performance compared to Alpaca. In contrast, the instruction pairs selected by the CaR method show significant improvements at 7B (29.8%), 13B (32.7%), and 30B (33.1%) scales.

Furthermore, to address cost considerations, we employed GPT-4's evaluation framework exclusively on four datasets to compare AlpaCaR against Alpaca. As depicted in Fig. 7 and elaborated upon in Appendix D, GPT-4 exhibited similar evaluative outcomes: AlpaCaR outperformed baseline in the majority of instances, thereby substantiating the efficacy of the CaR method. Employing CaR, which involves selecting 1.96% of the dataset, has proven to yield superior preferences across a variety of parameter scales.

4.5 Human Evaluation

We have formulated detailed evaluation criteria, covering seven aspects: fluency, relevance, correctness, consistency, satisfaction, informativeness and security, which are further categorized into 27 primary and 58 secondary classifications. Additional details are provided in Appendix E.

We compared AlpaCaR 30B vs. Alpaca 30B on Vicuna_80 test set. The human evaluation results demonstrated that AlpaCaR performed at least as well as Alpaca across all categories and was preferred by language experts in the vast majority of cases. The specific results are shown in Table 4.

Table 7 in Appendix F displays *case study* from the math category. We found that under strict evaluation criteria, experts believed that neither model provided the correct final answer, resulting in a tie. However, a more detailed analysis reveals that *AlpaCaR utilized CoT to explore the correct reasoning steps*, although errors occurred after certain steps. In contrast, Alpaca simply provided a con-

Category	win	lose	tie	WS$^\uparrow$
Writing	8	1	1	1.700
Roleplay	5	0	5	1.500
Common-sense	9	0	1	1.900
Fermi	7	2	1	1.500
Counterfactual	7	0	3	1.700
Coding	3	3	1	1.000
Math	0	0	3	1.000
Generic	6	0	4	1.600
Knowledge	7	2	1	1.500
Total	52	8	20	**1.550**

Table 4: Human evaluation results on Vicuna_80 dataset: AlpaCaR_30B vs. Alpaca_30B.

Method	Vicuna			Self-instruct		
	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$
Alpaca	0.338	10.00%	16.88%	1.206	45.63%	60.32%
mixed-181k	0.875	28.80%	43.75%	1.349	52.38%	67.46%
CaR_50k	1.113	33.75%	55.62%	1.500	63.89%	75.00%

Table 5: CaR is a stable and effective framework even on larger datasets

fusingly incorrect answer. We hypothesize that the IQS model has learned experts' preferences for detailed reasoning processes presented in the training data. Consequently, during subset selection, the IQS model favors instruction pairs that showcase meticulous reasoning, resulting in the fine-tuned AlpaCaR exhibiting more comprehensive thought processes in the form of CoT reasoning.

4.6 Larger Instruction Tuning Datasets

To further explore the performance of CaR in more massive and complex datasets, we conducted additional experiments on even larger instruction datasets. Following recent work (Du et al., 2023; Liu et al., 2023a), we combined five instruction tuning datasets, including Alpaca, Dolly_v2 (Conover et al., 2023), Alpaca-evol-instruct (Xu et al., 2023), HC3 (Guo et al., 2023), and LIMA (Zhou et al., 2023), to obtain a large-mixed-dataset containing 181,253 instructions. Then we used CaR to filter the large-mixed dataset and obtained CaR_50k containing 50k instructions.

Table 5 shows that the model fine-tuned on 50k instructions selected by CaR outperforms Alpaca at the same number of instructions using LLaMA 2 7B as the base pre-trained model. In addition, the model fine-tuned using CaR_50k outperforms the one using mixed-181k instruction tuning dataset.

This illustrates that the bottleneck of Alpaca is not that pre-trained LLaMA cannot learn more knowledge from more instructions, but rather that

Method	Selection	Training	Total
Alpaca	0$	733.35$	733.35$
Alpagasus	12.66$	104.18$	116.84$
AlpaCaR	0.02$	13.07$	13.09$

Table 6: Cost comparison of 30B scale.

the limited quality of instruction dataset restricts the model's performance. It also demonstrates that CaR is a stable and effective framework even on larger datasets. CaR framework can filter 50k high-quality instructions from 181k instruction pairs to get stronger model performances with less training overheads.

4.7 Cost Comparison

Here, we compare the computational costs of AlpaCaR, Alpaca, and Alpagasus, focusing on instruction evaluation and full parameter fine-tuning at the 30B scale, as detailed in Table 6. For instruction evaluation using an API-based method, we refer to the official pricing [3], while for model training or inference, we consider the rental costs of GPUs [4]. In summary, training AlpaCaR significantly saves both time and costs, compared to Alpaca or Alpagasus.

5 Is the Benefit Derived from Data Selecting Universally Applicable?

Filtering a high quality instruction sub-dataset to supervised fine-tuning LLaMA 1 significantly reduces computational cost and effectively improves LLM performances. More crucially, it is essential to ascertain whether data screening constitutes a consistent paradigm for performance enhancement, particularly as pre-trained model become increasingly powerful and model parameters scaling up. In this section, we used the average WS on Vicuna_80 and Self-instruct_252 test set to explore the generalization of data selection.

A consistent paradigm when *pre-training is more adequate*? Base pre-trained LLMs acquire knowledge through pre-training. LLaMA 1, LLaMA 2, and LLaMA 3 were pre-trained using 1T, 2.4T, and 15T tokens, respectively. When pre-trained models exhibit strong capabilities, can they discern the quality of fine-tuning instructions, rendering instruction selecting redundant? To investigate this, we employed LLaMA 1 7B, LLaMA

---

[3] https://openai.com/pricing
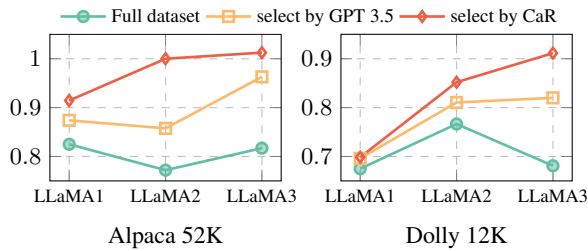[4] https://www.leadergpu.com/

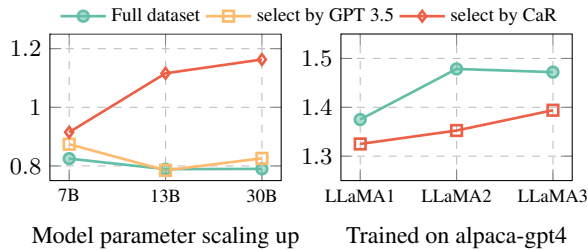Figure 8: Impact of data selection as *pre-trained model become more powerful*.

Figure 9: Impact of data selection as *models parameters* or *instruction quality* increase.

2 7B, and LLaMA 3 8B pre-trained models, comparing fine-tuning using the full dataset or subsets filtered by GPT-3.5 Turbo or CaR. Fig. 8 shows the results on Alpaca_52k and Dolly_15k IT datasets. The findings suggest that even as base pre-trained LLMs become more powerful, models fine-tuned on filtered data surpass those trained on full instructions. LLaMA 3 8B is more susceptible to low-quality instructions, impeding its ability to follow instructions in downstream tasks.

A consistent paradigm when *model size scaling up*? Many new capabilities and phenomena emerge as the model parameters scaling up. Thus another question is whether instruction tuning data selection is still important as the parameters increase. We experimented the performance of the model fine-tuned by full versus selected instructions at the 7B-30B scale, due to limited computational conditions. As shown on the left side of Fig. 9 (left), The horizontal direction showed no significant improvement in model performance even as the model size increased. However, the vertical direction showed that the model performs better using instructions selected by GPT-3.5 or CaR at all scales.

A consistent paradigm when *instructions quality improves*? Alpaca-GPT4 (Peng et al., 2023) contains instruction generated by GPT-4 using Alpaca prompts, which quality significantly improved

compared to Alpaca. Distinguishing high-quality instructions remains a challenge when instruction quality generally improves. As depicted in Fig. 9 (right), models trained by CaR-selected instructions are inferior to full instructions. We argue that the IQS model cannot significantly discriminate instruction quality in such a high-quality data distribution, so randomly filtering instructions caused performance degradation similar to Fig. 6. A similar phenomenon occurs when using LLMs to select instructions. Qwen1.5-110B-chat and Qwen-max scored more than 1,800 of the 2,000 instructions in the Alpaca-GPT4 dataset as perfect score, indicating that the quality of the evaluated instructions in this situation approaching the boundaries of the LLMs' capabilities. So data selecting methods at *higher data quality* are still challenging, and maybe gradient-based (Xia et al., 2024) or in-context learning-based (Li et al., 2023c) methods demonstrate greater potential.

6 Conclusion

In this paper, we focus on exploring and resolving the issue of instruction selection during supervised fine-tuning stage. We introduce the CaR method and examine two perspectives that are warrant considered: (1) Evaluating instruction quality using more authentic human preferences: models trained with data annotated by linguistic experts show higher agreement rates and the selected instructions lead to better-performing models. (2) Instruction diversity inspires LLMs' stronger capability: Under our selection framework, preserving a small number of instructions for different tasks through cluster improves model performance. Experimental results show that fine-tuning LLaMA (ranging from 7B to 30B parameters) with a 1.96% subset of instructions selected by CaR outperforms models trained on full datasets or data selected by GPT. Moreover, data selecting methods using GPT-family or CaR is a consistent paradigm whether the pre-trained model is more capable or the model parameters scaling up, while those at higher data quality are still challenging. Additionally, our approach can be deployed locally without relying on APIs, thereby enabling a more efficient instruction selection approach in low-computation resource environments.

7 Limitation

Despite the outstanding performance of CaR across multiple test sets, its experiments were confined to filtering on only several datasets. The diverse formats of different open-source instruction sets pose challenges for the academic community interested in instruction filtering tasks. In the future, we plan to validate the effectiveness of CaR on more datasets such as WizardLM_evol_instruct_70k (Xu et al., 2023). Moreover, while CaR is primarily used for single-turn dialogue instruction filtering, exploring its application in multi-turn dialogue instruction filtering presents an attractive direction for future research.

8 Potential Risk & Ethical Consideration

We reveal the following potential risks of our research based on ethical considerations:

1. Quality of instruction data: While the proposed method aims to select high-quality instruction data, there is still a risk that the selected subset may not fully represent the diversity and complexity of the entire dataset. This could potentially lead to biased or incomplete training of models and cause adverse social impact.

2. Bias and fairness: As with any AI research, there is a need to ensure fairness and mitigate biases. The selection process and scoring model used in CaR should be carefully monitored to prevent any unintentional biases, such as favoring certain types of instructions or excluding underrepresented groups.

3. Industrial deployment and responsible use: As the method is designed for industrial scenarios, it is important to consider the responsible use of the developed models. Ensuring that the models are not used for unethical purposes or harmful applications is crucial. Additionally, monitoring and addressing any unintended consequences or biases that may emerge during deployment should be a priority.

9 Acknoledgement

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. 2023. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*.

Xu Chu, Ihab F Ilyas, Sanjay Krishnan, and Jiannan Wang. 2016. Data cleaning: Overview and emerging challenges. In *Proceedings of the 2016 international conference on management of data*, pages 2201–2206.

Mike Conover, Matt Hayes, Ankit Mathur, Xiangrui Meng, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, et al. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

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

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpacafarm: A simulation framework for methods

that learn from human feedback. *arXiv preprint arXiv:2305.14387*.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Mustafa Hajij, Ghada Zamzmi, Karthikeyan Natesan Ramamurthy, and Aldo Guzman Saenz. 2021. Data-centric ai requires rethinking data notion. *arXiv preprint arXiv:2110.02491*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2023a. Generative judge for evaluating alignment. *arXiv preprint arXiv:2310.05470*.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023b. Self-alignment with instruction back-translation. *arXiv preprint arXiv:2308.06259*.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Junhao Liu, Tongliang Liu, Fei Huang, et al. 2023c. One shot learning as instruction data prospector for large language models. *arXiv preprint arXiv:2312.10302*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023a. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*.

Xiaoyong Liu and W Bruce Croft. 2004. Cluster-based retrieval using language models. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193.

Yilun Liu, Shimin Tao, Xiaofeng Zhao, Ming Zhu, Wenbing Ma, Junhao Zhu, Chang Su, Yutai Hou, Miao Zhang, Min Zhang, et al. 2023b. Automatic instruction optimization for open-source llm instruction tuning. *arXiv preprint arXiv:2311.13246*.

Mohammad Motamedi, Nikolay Sakharnykh, and Tim Kaldewey. 2021. A data-centric approach for training deep neural networks with less data. *arXiv preprint arXiv:2110.03613*.

Yongyu Mu, Abudurexiti Reheman, Zhiquan Cao, Yuchun Fan, Bei Li, Yinqiao Li, Tong Xiao, Chunliang Zhang, and Jingbo Zhu. 2023. Augmenting large language model translators via translation memories. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10287–10299, Toronto, Canada. Association for Computational Linguistics.

Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, et al. 1999. The pagerank citation ranking: Bringing order to the web.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Yizhou Sun, Jiawei Han, Peixiang Zhao, Zhijun Yin, Hong Cheng, and Tianyi Wu. 2009. Rankclus: integrating clustering with ranking for heterogeneous information network analysis. In *Proceedings of the 12th international conference on extending database technology: advances in database technology*, pages 565–576.

Hongyin Tang, Xingwu Sun, Beihong Jin, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Improving document representations by generating pseudo query embeddings for dense retrieval. *arXiv preprint arXiv:2105.03599*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023a. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, et al. 2023b. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

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

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

Zhiqiang Yuan, Junwei Liu, Qiancheng Zi, Mingwei Liu, Xin Peng, and Yiling Lou. 2023. Evaluating instruction-tuned large language models on code comprehension and generation. *arXiv preprint arXiv:2308.01240*.

Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. 2023. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024a. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024b. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

A Related work

Quality Estimation and Comet framework. Quality estimation is a pivotal task in machine translation, involving scoring or ranking translation results to select higher-quality data. Comet (Rei et al., 2020) leverages input and reference translations to accurately assess translation quality, employing two architectures: the Estimator model and the Translation Ranking model. The Estimator model directly predicts quality scores for each evaluation instance, while the Translation Ranking model learns parameters from paired evaluation data to predict reasonable quality scores.

Algorithm - Data Lifecycle. In the modern era of deep learning, high-quality data has become the cornerstone for training robust and effective models. Over the past decade, there has been a growing emphasis on the collection and curation of superior data (Chu et al., 2016; Motamedi et al., 2021). The emergence of data-centric AI has underscored the belief that data quality is as crucial as algorithmic advancements within the AI/ML lifecycle (Hajij et al., 2021; Zha et al., 2023). This paradigm shift has been particularly evident since the introduction of the Transformer architecture (Vaswani et al., 2017), which has revolutionized the field of language modeling. Rather than focusing on disruptive innovations in model structure, researchers have concentrated on leveraging the effectiveness of the Transformer architecture by stacking transformer blocks to create more potent models. Additionally, significant improvements in model performance have been achieved through the construction of task-specific datasets and the enhancement of data quality (Zhou et al., 2023; Chen et al., 2023; Li et al., 2023c).

Futher perspective of clustering and ranking. Many domains have employed methods similar to clustering and ranking. In information retrieval, Google extensively utilizes the PageRank algorithm (Page et al., 1999) to calculate the importance of hyperlinks between webpages. Liu et al. developed a cluster-based retrieval model by constructing language models for clusters (Liu and Croft, 2004), combining documents within the same cluster and searching/ranking clusters based on query generation likelihood. Tang et al. enhanced the Bi-encoder's performance in dense information retrieval tasks by using clustering algorithms to generate "pseudo-query embeddings" (Tang et al., 2021).

Selecting suitable data for LLM inference is crucial in the RAG field, as discussed by Yuan et al. (2023) and Mu et al. (2023), who explore methods for finding appropriate demonstrations to improve LLM performance. In the network domain, Sun et al. introduced the RankClus framework (Sun et al., 2009), which integrates clustering and ranking methods to strengthen heterogeneous information network analysis.

Evaluation of LLMs. Evaluating the open-domain instruction-following capabilities of LLMs presents a significant challenge. Currently, the prevailing approach involves employing human evaluators or GPT-4 to compare the inference response of different models. Consequently, recent studies, including PandaLM (Wang et al., 2023b), Vicuna (Chiang et al., 2023), CoachLM (Liu et al., 2023b), and Self-Instruct (Wang et al., 2022), have curated and provided their own instruction sets to evaluate instruction-finetuned LLMs. Additionally, leaderboards such as MT-Bench (Zheng et al., 2024a), Alpaca-Eval (Dubois et al., 2023), and Chatbot Arena (Chiang et al., 2024) have been established to measure the instruction-following abilities of these models. PandaLM (Wang et al., 2023b) and Auto-J (Li et al., 2023a) efforts focus on training LLMs to provide more impartial and accurate evaluations. By leveraging these latest advancements, we aim to evaluate our model's performance using human-generated instruction sets, ensuring a comprehensive and rigorous assessment of its capabilities in following open-ended instructions.

B Evaluate Prompts

B.1 IQE Prompt

> [The Start of Assistant A's Instruction and Answer]
> {Instruction pair 1}
> [The End of Assistant A's Instruction and Answer]
> [The Start of Assistant B's Instruction and Answer]
> {Instruction pair 2}
> [The End of Assistant B's Instruction and Answer]
> [System]
> We would like to request your feedback on the performance of two AI assistants in response to the user question displayed above. Please rate the helpfulness, relevance, accuracy, level of details of their responses. Each assistant receives an overall score on a scale of 1 to 10, where a higher score indicates better overall performance. Please first output a single line containing only two values indicating the scores for Assistant 1 and 2, respectively. The two scores are separated by a space. In the subsequent line, please 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.

B.2 Response Comparison Prompt

> [Question]
> {Instruction}
> [The Start of Assistant 1's Answer]
> {Response 1}
> [The End of Assistant 1's Answer]
> [The Start of Assistant 2's Answer]
> {Response 2}
> [The End of Assistant 2's Answer]
> [System]
> Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Begin your evaluation by comparing the two responses and provide a short explanation. Avoid any positional biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.

C Specifics about Instruction Quality Estimation

C.1 Evaluation Metric of IQE

The second row of Table 1 presents results for instruction pairs sourced from the IQE test set, which are instructions revised by language expert. The third row shows accuracy on instruction pairs from Vicuna_80, demonstrating the models' generalization to other distributions. The instructions are provided by the dataset, while language experts evaluates the quality of two responses generated by different models, establishing the ground truth labels. In the calculation of accuracy, if the absolute difference between the scores of two responses is less than 0.01 assigned by IQS or $\text{Comet}_{Instruct}$, the outcome is considered a "Tie".

C.2 Model Architecture of IQS and $\text{Comet}_{instruct}$

In the IQE task, the IQS model and Comet model correspond to the Estimator model architecture and Translation Ranking model architecture in the Comet framework, respectively. As shown in Fig. 10, The $\text{Comet}_{instruction}$ model concatenates instructions with input to form anchors. It then feeds pairs of better and worse responses into the model. Finally, the model is trained using a triplet margin loss function to distinguish between the superior
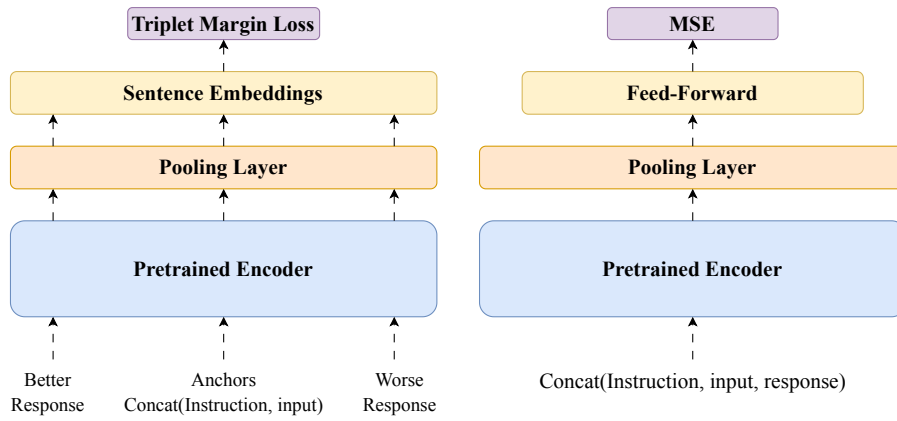
Figure 10: Detailed architecture of Comet$_{instruct}$ model(left) and Instruction pair quality scoring model(right).
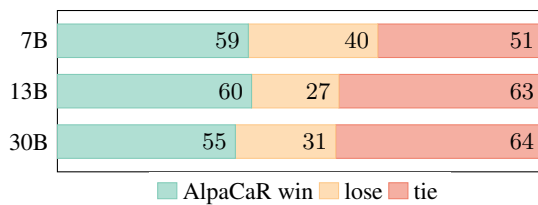

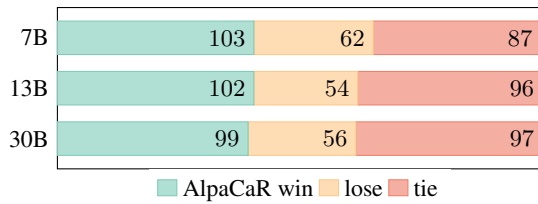Figure 11: GPT-4 result on CoachLM_150 dataset: AlpaCaR vs. Alpaca.

Figure 12: GPT-4 result on Self-instruct_252 dataset: AlpaCaR vs. Alpaca.
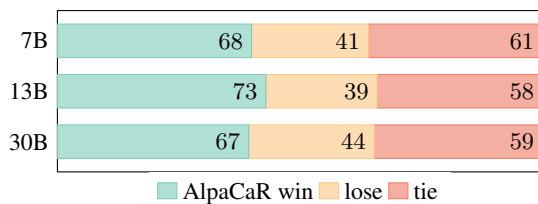

Figure 13: GPT-4 result on Pandalm_170 dataset: AlpaCaR vs. Alpaca.

and inferior responses. The IQS model concatenates instruction pairs and then trains the model using Mean Squared Error as the loss function.

D More Results about GPT-4 Evaluations

As illustrated in Fig. 11, 12, 13. Baseline: reference responses.

E Specifics about Human Evaluation Criteria

- **Fluency**
  - Redundancy: verbose repetition (>=2)
  - Redundancy: extraneous content clutter
  - Missing: incomplete response
  - Error: syntax & semantics issues
  - Errors: word & symbol errors
- **Relevance**
  - Irrelevant
  - Incorrect understanding of object / personality / relationship
  - Poor compliance
- **Correctness**
  - Incorrect answer to the question
  - Incorrect derivation process
- **Consistency**
  - Intrasentence consistency
  - Contextual consistency
  - Content Logical Consistency
  - Query response consistency
- **Satisfaction**
  - Unclear views, no precise conclusions given (advice / recommendations)
  - Insufficiently organized and comprehensive content
  - Poor extensibility, almost a carbon copy of the query information, lack of embellishments
  - Lack of emotional feedback
  - Not careful enough to reply
  - Biases in characterization and speaker tone
  - Overly subjective and unsubstantiated
  - Some of the recommendations are ineffective, irrelevant to the issue or lack specific advice and assistance
  - Indeed a problem, but a subjective experience problem that is hard to describe
- **Informativeness**
  - Missing interpretive information
  - Lack of richness
- **Security**
  - Minor or moderate security violations
  - Serious security violations

Instruction	Input	AlpaCaR_30B	Alpaca_30B	Human evaluation
Given that f(x) = 5x^3 - 2x + 3, find the value of f(2).	\	The value of f(2) = 5(2)^3 - 2(2) + 3 = 5(8) - 4 + 3 = 44.	The value of f(2) is 3.	Tie
Solve for x in the equation 3x + 10 = 5(x - 2).	\	The equation 3x + 10 = 5(x - 2) can be rearranged to solve for x. First, subtract 5(x - 2) from both sides to get 3x - 5x + 10 = 5(x - 2) - 5x + 10. Then, x = 5.	x = 7	Tie
If the endpoints of a line segment are (2, -2) and (10, 4), what is the length of the segment?	\	The length of the line segment can be found using the formula for the distance between two points: $d = \sqrt{(x2 - x1)2 + (y2 - y1)2}$ In this case, x2 = 10, x1 = 2, y2 = 4, and y1 = -2. Plugging these numbers into the formula, we get $d = \sqrt{(10 - 2)2 + (4 - (-2))2}$ Similarly, we can calculate the length of the line segment using the Pythagorean-Pythagorean-Circles Theorem: $d = \sqrt{(x2 + y2)}$ In this case, x2 = 10, y2 = 4, so $d = \sqrt{(10 + 4)} = 5$. Therefore, the length of the line segment is 5.	The length of the line segment is 8.	Tie

Table 7: Case study of LLMs responses in vicuna_80 math category.

F Case study

As illustrated in Table 7.

G Profile of Involved Language Experts

To ensure a comprehensive and rigorous human evaluation of LLM abilities, we established a collaboration with the language service center of a prominent international corporation. We recruited a team of highly educated, multilingual language experts with diverse skills in translation, localization, writing, and testing, who dedicated their full-time efforts to this task. Specifically, three experts possessing an average experience of 12.57 years, are responsible for conducting a human evaluation of AlpaCaR and other LLMs.

H Discussion of CaR framework

Selecting top-n ranked samples for each cluster is indeed an intuitive and interesting idea that integrates the two steps of clustering and ranking. We have also experimented with this setting in our early research. However, a challenge arises when the predefined number of clusters $k = \sqrt{Number_{instructions/2}} = 161$ is used. When top-n is small, the resulting dataset size is insufficient for the model to achieve good instruction-following capacity. Conversely, when top-n is large, it introduces more low-quality instruction pairs, which negatively impacts the performance of LLMs. An

Top-n	Vicuna			Self-instruct		
	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$	WS$^\uparrow$	WR$^\uparrow$	QS$^\uparrow$
10	1.188	55.00%	90.00%	1.230	45.63%	77.38%
20	1.375	51.25%	83.75%	1.167	42.86%	73.81%
30	1.300	57.50%	85.00%	1.111	38.49%	72.62%
CaR(ours)	**1.475**	**58.75%**	**88.75%**	**1.310**	**51.98%**	**78.97%**

Table 8: Discussion of CaR framework: k × top-n v.s. n_1 + k × n_2

early version of our experimental results (baseline: Alpaca 52k) is shown in Table 8.

The experimental results indicate that this combinatorial approach performs less effectively than treating the two components separately. Our idea is to additionally and separately extract the top n_1 instructions using only the ranking step to ensure that most high-quality instructions are included (as indicated in section 2.2) while using a smaller top n_2 to prevent the inclusion of a large number of low-quality instruction pairs. Experimenting with different values of k might alleviate this problem, but we aim to propose a more automated process and avoid involving additional hyperparameter tuning.