RECOST: External Knowledge Guided Data-efficient Instruction Tuning

Qi Zhang, Yiming Zhang, Haobo Wang, Junbo Zhao[*]
Zhejiang University
{cheung_se,yimingz,wanghaobo,j.zhao}@zju.edu.cn

Abstract

In the current landscape of large language models (LLMs), the process of instruction tuning serves as an essential step. Considering the high computing power overhead, data-efficient instruction tuning was proposed to reduce the training data size in this process, aiming at selecting high-quality instructional data. Nevertheless, we argue that most current data-efficient instruction-tuning methods are highly dependent on the quality of the original instruction-tuning dataset. When it comes to datasets synthesized by LLMs, a common scenario in this field, dirty samples will even be selected with a higher probability than other samples. To address these challenges, we utilized external knowledge (relevant examples or paragraphs) to evaluate those samples synthesized by LLMs with an in-context-based relative predictive entropy. Based on the new metric, we proposed a framework, dubbed as **RECOST**, which integrates external-knowledge-base reranking and diversity-consistent sampling into a single pipeline. Through extensive experiments on several synthetic datasets (Alpaca and Alpaca-gpt4), we demonstrate the effectiveness of our method and achieve even better results with only **1%** of the full dataset.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020) have demonstrated their remarkable capabilities in numerous fields of natural language processing (NLP) with the advancing of training datasets and the scale of model parameters. Behind this phenomenon, instruction tuning serves as an essential step to help pre-trained LLMs align to human cognition (Ouyang et al., 2022; Peng et al., 2023; Chung et al., 2022). Instruction tuning refers to fine-tuning the LLMs on instruction-response pairs to endow LLMs with instruction-following capability and activate the knowledge gained in the pre-training period.

In the past two years, three types of instruction-following datasets have emerged (Wang et al., 2023): those based on traditional NLP tasks (eg. Flan v2 (Wei et al.; Longpre et al., 2023)), those based on high-quality manual annotation (eg. LIMA (Zhou et al., 2023)), and those synthesized by LLMs (eg. Alpaca (Taori et al., 2023)). Among these, LIMA asserts that the quality of instruction-following datasets is far more important than their quantity. Thus, data-efficient instruction tuning is proposed to reduce the data size in instruction tuning without compromising the models' performance (Zhou et al., 2023; Chen et al., 2023a).

Contemporary works in data-efficient instruction tuning predominantly concentrate on selecting high-quality data from instruction datasets synthesized by LLMs (Li et al., 2023a; Chen et al., 2023b; Li et al., 2023c). Most approaches involve evaluation based on its proposed metrics, primarily centered around metrics related to predictive entropy (Kadavath et al., 2022). We simply conclude these methods to synthetic-knowledge-guided ones as they select data points by using synthetic data points as prior knowledge.

However, Duan et al. figured out that predictive entropy is not reliable enough. As shown in Figure 1, **selection based on predictive entropy still results in the sampling of a non-negligible proportion of noisy data**. What's even more surprising is that the higher the quality ranking, the greater the probability that samples contain noise, which is completely contrary to the original design intention. We argue that these methods heavily rely on the quality of the curated datasets which serve as the pre-experience (Li et al., 2023a) or test set (Li et al., 2023c) in their processes and are susceptible to the influence of outlier samples within the original datasets. Therefore, in data selection under synthetic datasets scenarios, we argue that

[*]Corresponding author.

synthetic-knowledge-free methods are still underexplored. This issue currently imposes significant limitations on the development of the field of data-efficient instruction-tuning.

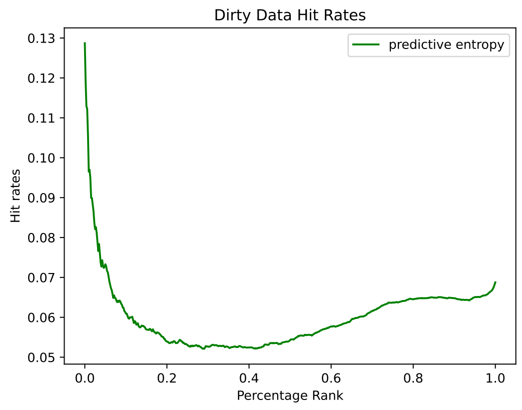


Figure 1: The dirty data hit rates according to its predictive entropy calculated by LLaMA-2-7b. The horizontal axis represents the percentage ranking, while the vertical axis denotes the proportion of corrupted data within the data preceding that percentage threshold. Given the number of dirty data in the top i data points as d_i, the hit rate at i is calculated by d_i/i. The dirty data is collected by comparing Alpaca with Alpaca-cleaned.

To address the challenge posed by the limitations of the predictive entropy in vanilla LLMs as outlined above, we utilize external information to evaluate samples synthesized by LLMs. Despite the suboptimal performance of this dataset in generative tasks (Wang et al., 2023), its authenticity is significantly assured. But in the data-efficient instruction-tuning scenario of LLM, this cost is unacceptable. Recognizing the importance of maintaining efficiency, we instead intuitively leverage pre-trained LLMs' intrinsic in-context learning (ICL) capabilities, treating these truthful samples as demonstrations. Building on this foundation, we introduce a concept: in-context-knowledge-based relative predictive entropy, which serves as another dimension of uncertainty for vanilla LLMs.

In this paper, we propose **RECOST** (REtrieval, RE-rank, COreset sampling, and Supervised fine-Tuning), a framework that encompasses an in-context-knowledge-based re-ranking module and a diversity-consistent sampling module to avoid an overly homogeneous data distribution after re-ranking. With extensive experiments on synthetic datasets including Alpaca and Alpaca-gpt4, **RECOST** demonstrates its superiority over previous methods and surpasses remarkably the full-trained model with merely 1% and 10% training data on three benchmarks including the Alpagasus test sets (Chen et al., 2023b; Li et al., 2023a), the Open-LLM benchmark (Gao et al., 2023) and AlpacaEval (Li et al., 2023b) benchmark.

All in all, our work explores how to instruct-tune LLMs under data-efficient scenarios with synthetic datasets. Our contributions can be summarized as follows:

- We firstly propose **RECOST**, a method to mine high-quality data points from a synthetic dataset with consideration of the truthful-knowledge-based uncertainty and diversity.
- We conduct extensive experiments on multiple synthetic datasets and our method surpasses the fully trained model by utilizing only 1% of the full dataset.

2 Related Work

2.1 Instruction Tuning

Instruction tuning has been regarded as an essential step to align pre-trained LLMs with human cognition (Ouyang et al., 2022; Chung et al., 2022; Peng et al., 2023). This methodology refers to the supervised fine-tuning of pre-trained LLMs on datasets designed for instruction following. Each dataset entry comprises a pair, including an instruction and its corresponding response.

2.2 Data-efficient Instruction Tuning

As LIMA (Zhou et al., 2023) makes the statement that *less is more for alignment*, a new realm called data-efficient instruction tuning appears to help work out the bottleneck of data quality in this field. Recent works on data-efficient instruction tuning mainly quantify the quality of instruction data in two folds: the feedback from close-source LLMs and the score calculated by open-source LLMs based on a proposed metric.

Alpagasus (Chen et al., 2023b) first dives into this field by employing ChatGPT to quantify the quality of instruction data by rating each data point, selecting those with higher scores, and supervised fine-tuning the LLMs with the data points with top scores. This work exploits a feasible way to sample high-quality data with feedback from close-sourced LLMs. Upon this, Deita (Liu et al., 2024) utilizes the feedback from ChatGPT to train scorer models to quantify the instruction data from the dimension of complexity and quality.

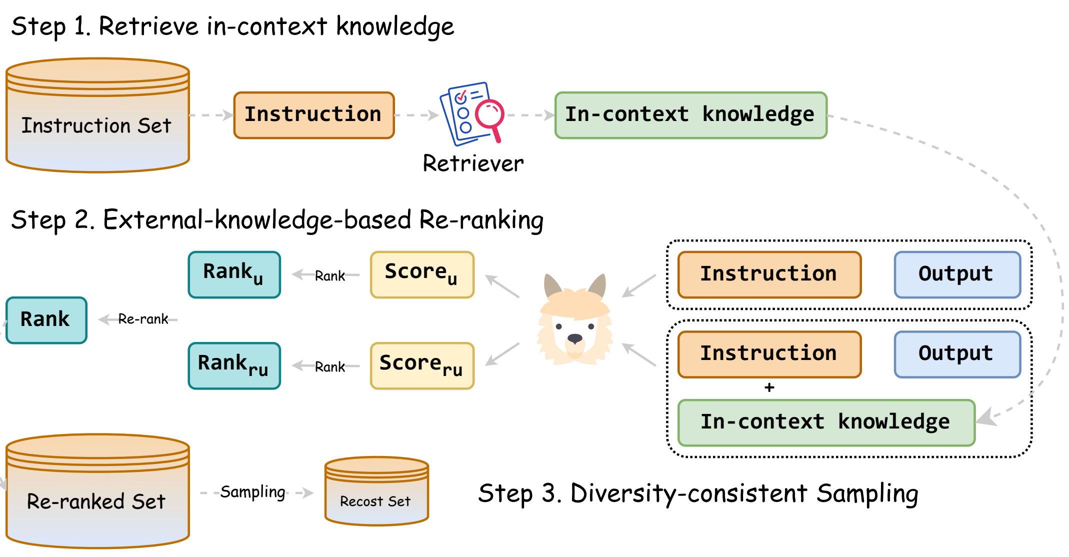


Figure 2: Overview of our proposed method. We start by retrieving in-context knowledge for each under-quantified data point. Two scores are produced by the vanilla LLaMA model on conditions with in-context knowledge or without that. The under-selected data points will be re-ranked by two ranks according to the produced two types of scores. Diversity-consistent sampling will be employed to select the qualified data points to finally supervised fine-tune the language models.

Another type of method, resembling the paradigm of active learning, tends to select data points based on the under-tuned model itself and an initial subset of the full dataset. Instruction mining (Cao et al., 2023) summarizes the performance of several common metrics on data-efficient instruction tuning and proposes a complex equation to calculate the instruction data quality explicitly. Li et al. proposes Instruction-Following Difficulty (IFD), a self-guided method for mining data points with higher IFD scores. Nuggets (Li et al., 2023c) introduces one-shot learning as implicit instruction tuning to guide the data selection for instruction tuning. Nuggets show promising results on some benchmarks while the selection process is computationally costly. Our method can also be regarded as one type of this but is more computationally efficient.

2.3 Synthetic Instruct-following Datasets

Self-instruct (Wang et al., 2022) serves as the milestone research in utilizing LLMs to synthesize instruct-following datasets. It starts from a small seed dataset and produces a fantastic instruction dataset with GPT3. Upon this method, Alpaca (Taori et al., 2023) was proposed by transferring self-instruct to OpenAI's text-davinci-003 engine. Alpaca-gpt4 shares the same prompts with the original Alpaca while using GPT4's response (Peng et al., 2023) as the answer to the prompts. Evol-instruct provides another paradigm for data synthesis by using ChatGPT to change the complexity of instructions and has generalized to multiple domains (Xu et al., 2023; Luo et al., 2023b,a). In this paper, we primarily take synthetic instruct-following datasets into consideration.

3 Methodology

In this section, we will dive into the main methodology of **RECOST**. Figure 2 briefly illustrates the framework of our method.

3.1 Preliminaries

We start by defining the concept of predictive entropy. Predictive Entropy (PE), described in (Kadavath et al., 2022), is a popular metric to measure the uncertainty of LLMs. It's defined as the entropy over the whole response y, which is equivalent to the accumulation of the token-wise entropy:

$$\begin{aligned} PE(x,y) &= -\log p(y|x) \\ &= \sum_i -\log p(z_i|y_{<i},x) \end{aligned} \tag{1}$$

where x, y, z_i refer to the instruction, the response and the i-th token in y, respectively.

3.2 Motivation and Methodological Prelude

In this section, we will give an overview to elucidate the dilemma of current data-efficient instruction tuning and the insights of RECOST.

Most methods in data-efficient instruction tuning rely on building their metrics with predictive entropy. However, as highlighted by Duan et al., predictive entropy fails to adequately characterize LLMs' uncertainty. Besides, as shown in Figure 1 and Table 1, after sorting samples with their corresponding predictive entropy, the hit rate of dirty data has consistently maintained at an unacceptable level.

Rank	<100	<200	<300	<400	<500
Hit Rates	12.9%	11.9%	11.3%	11.2%	10.6%

Table 1: The hits rates on dirty data when sorting samples with predictive entropy.

We hypothesize that this may be attributable to its excessively high correlation with response length, coupled with the tendency of LLMs to produce longer responses due to hallucination while synthesizing data.

Besides, since methods such as IFD (Li et al., 2023a) introduce a subset of the synthetic dataset as the so-called pre-experience, the potential noise in sampled synthetic data points might further exert a negative impact on language models.

Thus, we come up with introducing external knowledge to propose a new metric and guide the data selection process. Originating from traditional NLP tasks, despite its poor performance in generative tasks (Wang et al., 2023), Flan v2 is selected as an external knowledge source due to its low cost, vast data volume, and reliability. Considering efficiency, we refrain from fine-tuning the model on external knowledge; instead, we leverage the in-context learning capabilities of pre-trained LLMs to measure their external-knowledge-based conditional predictive entropy. Recall the equation 1, we here define it as in-context-knowledge-guided predictive entropy PE_{ic} based on the in-context knowledge:

$$\begin{aligned} PE_{ic}(s,x,y) &= -\log p(y|x,s) \\ &= \sum_i -\log p(z_i|y_{<i},x,s) \end{aligned} \quad (2)$$

where s refers to the in-context knowledge. To further mitigate the randomness of demonstration selection, a retrieval technique is introduced for robustness.

Based on the foregoing, we will elaborate on the external-knowledge-based re-ranking module and the diversity-consistent sampling module of RECOST in the following sections.

3.3 External-knowledge-based Re-ranking

Previous research has shown that retrieved similar demonstrations can help improve LLMs' in-context learning ability (Rubin et al., 2021; Luo et al., 2024; Wang et al., 2024; Pan et al., 2024; Gao et al., 2024). Intuitively, when it comes to synthetic samples, reliable data points will gain more from retrieved demonstrations than those less reliable ones. Therefore, based on predictive entropy and our proposed in-context-knowledge-guided predictive entropy, we further define the difference between the two entropies as the relative predictive entropy (RPE):

$$PE_r(x,y) = PE(x,y) - PE_{ic}(s,x,y) \quad (3)$$

On the one hand, as similar demonstrations can promote language models' in-context learning ability (Luo et al., 2024), those data points with higher relative uncertainty can also be regarded as the more reliable ones. On the other hand, it's acknowledged that in-context learning is conducting an explicit gradient descent. From this perspective, a higher relative uncertainty signifies a greater sensitivity to external knowledge, thereby rendering such samples more amenable to learning.

Furthermore, to take both uncertainty and relative uncertainty into consideration, we define the mixed rank R_m as the weighted average of the two ranks corresponding to the two types of uncertainty:

$$R_m^{(i)} = w * R_u^{(i)} + (1-w) * R_{ru}^{(i)} \quad (4)$$

where $R_u^{(i)}$ and $R_{ru}^{(i)}$ refer to the ranks of the i-th data point in the degree of uncertainty and relative uncertainty respectively.

At last, we re-rank all the data points with their corresponding mixed rank to generate the re-ranked instruction dataset.

3.4 Diversity-consistent Sampling

To enable the diversity of the sampled subset, we add an additional stage after re-ranking to increase the diversity further. Core-set sampling (Sener and Savarese, 2018) is a technique for selecting a representative subset of a dataset, allowing for efficient approximation of solutions to problems by reducing computational complexity without significantly compromising result quality.

However, integrating greedy core-set sampling into our framework directly fails to make itself aware of the mixed ranks in Equation 4. Thus, in this section, we introduce a sliding window mechanism, which is illustrated in Algorithm 1, to consider the diversity of the sampled subset with awareness of its overall uncertainty.

Algorithm 1 Core-set sampling with a sliding window.

Input: The re-ranked instruction dataset D, the sampling size s, the initial subset size s_i, the sliding window size w, and the tolerance t.

Output: The sampled subset D_s.

```
1:  $D_s \leftarrow D[: s_i]$
2:  $W \leftarrow D[s_i : s_i + w]$
3:  for $j \leftarrow 1$ to $w$ do
4:      $T[j] \leftarrow t$
5:  end for
6:  for $k \leftarrow 1$ to $s - s_i$ do
7:      $d \leftarrow \mathbf{FarthestFirst}(D_s, W)$
8:      $D_s \leftarrow D_s \cup W[d]$
9:      $W.\mathbf{pop}(d); T.\mathbf{pop}(d)$
10:     $\mathbf{update}(W, T)$
11:     $W.\mathbf{push}(d); T.\mathbf{push}(t)$
12: end for
13: return $D_s$
```

We start by sampling the top data points in the re-ranked instruction as an initial set. Similar to conventional core-set sampling, our algorithm also samples data points iteratively. However, within each iteration, a sliding window mechanism is introduced to enable the algorithm to only sample from the under-selected data points with higher uncertainty and relative uncertainty. Moreover, to further increase the diversity of the sampled subset, we introduce a tolerance t for each data point in the sliding window to make sure each of them can only be considered for t times at most. After every iteration, data points with zero tolerance will be erased from the sliding window.

4 Experiments Settings

4.1 Datasets

Self-instruct (Wang et al., 2022) is a milestone method for constructing instruction-tuning datasets by distilling from closed-sourced LLMs. The Alpaca (Taori et al., 2023) dataset employed self-instruct method to distill instruction data from ChatGPT or GPT-4 (Cao et al., 2023) and serves as a commonly used dataset for instruction tuning. In this paper, we start with ChatGPT and GPT-4 versions of Alpaca.

4.2 Benchmarks

For evaluation, we use three benchmarks to evaluate our method.

For the general ability of instruction-tuned language models, we use the OpenLLM benchmark[1], which includes four datasets: Arc (Yadav et al., 2019), Hellaswag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), and Truthfulqa (Lin et al., 2022). Following the general setting, we use 25-shot for Arc dataset, 10-shot for the Hellaswag dataset, 5-shot for the MMLU dataset, and 0-shot for the Truthfulqa dataset.

Another way to evaluate the instruction-tuned language models is to use LLMs to evaluate the responses generated by those models. We use the AlpacaEval[2] benchmark (Li et al., 2023b) to evaluate the open-end generation ability. To be specific, we use GPT4-turbo to choose the preferred response generated by our fine-tuned models and OpenAI's text-davinci-003 for a given instruction. The overall win rate will be devoted to evaluating the generation ability of language models.

Moreover, we follow previous research with similar settings and evaluate our proposed method with fully trained models Alpagasus test sets (Vicuna, Koala, WizardLM, and Self-Instruct) and IFD's additional test set (LIMA). In detail, we use GPT4 to rate two responses from two models on a scale of 1 to 10, which implies accuracy and relevance. To dismiss the potential positional bias, we also follow the 'Win-Tie-Lose' rule to judge the two responses both obversely and reversely:

- **Wins**: RECOST wins twice, or wins once and draws once.
- **Ties**: RECOST draws twice, or wins once and loses once.
- **Loses**: RECOST loses twice, or loses once and draws once.

4.3 Implementation Details

For the retrieving period, we use a subset of Flan v2[3] as the knowledge source considering time ef-

[1]https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

[2]https://github.com/tatsu-lab/alpaca_eval

[3]https://huggingface.co/datasets/sordonia/flan-10k-flat

Method	**Data size**	OpenLLM Benchmark					**AlpacaEval**
		Arc	Hellaswag	MMLU	Truthfulqa	**Avg**	
LIMA (Zhou et al., 2023)	1000	55.55	81.55	47.74	47.23	58.02	26.58
Alpaca Results							
Full Alpaca	52002 (100%)	54.18	78.21	45.80	42.05	*55.06*	*27.75*
IFD (Li et al., 2023a)	3111 (6%)	57.94	80.37	44.19	40.62	55.78	36.78
Random*	**520 (1%)**	54.10	78.22	47.52	39.77	54.90	26.52
Predictive Entropy*	**520 (1%)**	55.03	77.20	45.18	43.84	55.31	35.59
RECOST (ours)	**520 (1%)**	56.48	77.73	45.80	44.27	**56.07**	**39.19**
Alpaca-gpt4 Results							
Full Alpaca-gpt4	52002 (100%)	56.57	80.72	49.06	54.51	*60.21*	*61.80*
Random*	**5200 (10%)**	55.63	80.87	48.52	51.27	59.07	55.92
Predictive Entropy*	**5200 (10%)**	57.59	81.19	47.95	52.13	59.72	60.39
RECOST (ours)	**5200 (10%)**	57.68	80.63	48.53	52.11	**59.74**	**63.35**

Table 2: Performance of RECOST on the OpenLLM Benchmark and AlpacaEval. The methods marked with * are the data-efficient instruction-tuning results we implemented based on the corresponding metrics, for better comparison with our metric. RECOST outperforms all previous methods on AlpacaEval and is comparable on OpenLLM Benchmark with a fraction of training data of them.

ficiency, which includes 10 million samples. Following the settings of Zhang et al., we use llm-embedder as our retriever and retrieve 5 related demonstrations for each data point.

We choose LLaMa-2-7b as the base model to validate our proposed method. All the models are trained with the Adam optimizer with a batch size of 64 and a 2e-5 learning rate for 3 epochs as official Alpaca.

5 Experiments Results and Analysis

In this section, we present the main results of our method on the three benchmarks mentioned in Section 4.2. Moreover, we conduct extensive ablation studies based on several possible factors.

5.1 OpenLLM Benchmark and AlpacaEval Benchmark

The experimental results of OpenLLM Benchmark and AlpacaEval Benchmark are presented in Table 2.

Concerning the OpenLLM Benchmark, attributable to its constrained output domain, abbreviated output duration, and the assurance of a stochastic interval, our methodology realizes a marginal enhancement. In the context of AlpacaEval, as it constitutes an open-ended generation endeavor, our strategy markedly surpasses alternative methodologies by supplying data of superior quality.

Specifically, for the Alpaca dataset, RECOST outperforms remarkably the fully trained Alpaca model and random selection on both benchmarks. Besides, it surpasses the previous method with even less data. As for the Alpaca-gpt4 dataset, which is of superior quality, RECOST still achieves comparable outcomes to those of full training while only utilizing 10% of the dataset volume, significantly surpassing the results of random sampling.

5.2 Alpagasus Test Set

Following Alpagasus (Chen et al., 2023b) and IFD (Li et al., 2023a), we further compare our method with fully trained models on Alpagasus test sets (Vicuna, Koala, WizardLM, and Self-Instruct) and IFD's additional test set (LIMA). Figure 3a gives

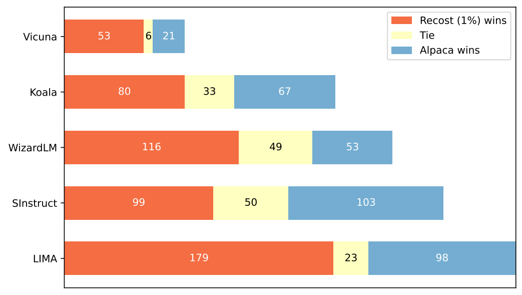
As shown in Figure 3, RECOST shows promising results over the fully trained models on both datasets and only lags on one of the test sets on the Alpaca dataset.

5.3 Ablation Study

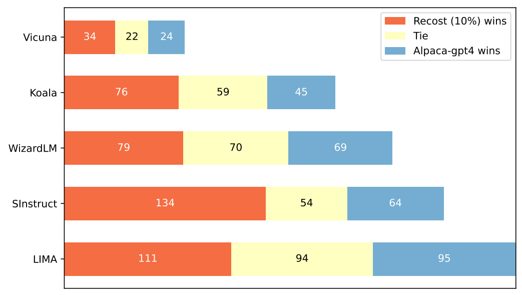
To evaluate the robustness of our method, we conduct extensive ablation experiments based on the following factors.

5.3.1 Ablation on Knowledge Source

Different External Knowledge Sources Besides Flan v2, inspired by retrieval-based generation (RAG), we also explore our method with knowledge from raw text. Following the settings of Self-RAG, we retrieve related paragraphs with the under-selected instruction data. To be specific, we also



(a) RECOST vs. Full Alpaca.



(b) RECOST vs. Full Alpaca-gpt4.

Figure 3: Results on Alpagasus Test Set. Figure 3a and Figure 3b demonstrate RECOST's performance compared to models fully trained on Alpaca and Alpaca-gpt4 respectively.

Method	Data size	OpenLLM	AlpacaEval
Alpaca Results			
Full Alpaca	52002	55.06	27.75
RECOST (Wiki)	520	55.51	37.76
RECOST (Flan v2)	520	**56.07**	**39.19**
Alpaca-gpt4 Results			
Full Alpaca-gpt4	52002	**60.21**	61.80
RECOST (Wiki)	5200	59.78	61.61
RECOST (Flan v2)	5200	59.74	**63.35**

Table 3: Performance of RECOST based on different knowledge sources. RECOST (Wiki) and RECOST (Flan v2) refer to RECOST with Wikipedia and Flan v2 as the external knowledge source respectively.

AlpacaEval	In-context Knowledge Type		
	Synthetic	Random	Retrieved
helpful_base	41.86	44.19	48.06
koala	31.41	37.18	41.03
oasst	38.30	40.96	41.49
selfinstruct	28.29	19.52	28.57
vicuna	41.25	46.25	47.50
Overall	34.76	34.70	**39.19**

Table 4: Performance of RECOST with different in-context knowledge types on AlpacaEval benchmark.

retrieve five paragraphs as in-context knowledge for every data point. As shown in Table 3, **RECOST** with Wikipedia as external knowledge also achieves promising results on both datasets. However, due to the lack of RAG capability for vanilla LLMs, RECOST (Wiki) is slightly inferior to RECOST (Flan v2) but still comparable with previous work.

Different External Knowledge Types To find out how external knowledge affects the final performance within our RECOST, we further explore this by conducting two experiments: one for random truthful demonstrations and another for synthetic demonstrations.

Specifically, synthetic in-context knowledge refers to retrieving similar demonstrations from the Alpaca dataset itself, while random in-context knowledge refers to utilizing random demonstrations as in-context knowledge. Table 4 shows the effects in dimensions of both truthfulness and similarity of in-context knowledge. The original RECOST outperforms the two mentioned in-context knowledge types on all subsets and the overall score of AlpacaEval benchmark.

Besides, we further explore the robustness of relative uncertainty as the number of retrieved demonstrations changes. In Section A.1, we use the Jaccard similarity coefficient to compare the similarity of the top 10% examples sorted by our proposed relative uncertainty with 1 to 5 demonstrations retrieved.

5.3.2 Ablation on Diversity Sampling

The effectiveness of diversity-consistent sampling. To further evaluate the effectiveness of the diversity-consistent sampling module of RECOST, we compare RECOST under different settings. As presented in Table 5, with diversity-consistent sampling, RECOST achieves slight improvement on both benchmarks.

Weigh between uncertainty and diversity. Our experimental results also imply that the balance between uncertainty and diversity should not be neglected. Table 6 gives results of RECOST under variant diversity. By appropriately adjusting tolerance, we can enhance performance. However,

Dataset	Setup	OpenLLM	AlpacaEval
Alpaca	w/o. diversity sampling	56.05	37.80
	w/. diversity sampling	**56.07**	**39.19**
Alpaca-gpt4	w/o. diversity sampling	59.38	62.80
	w/. diversity sampling	**59.60**	**63.50**

Table 5: Ablation study on diversity sampling.

excessively high diversity can detrimentally impact efficiency. Here we simply use the mean cosine similarity as the diversity of the selected datasets, where lower mean cosine similarity leads to datasets with higher diversity.

Tolerance	-	468	104	52	26
Mean CosSim	0.6852	0.6840	0.6815	0.6805	0.6794
AlpacaEval	37.80	39.19	38.14	38.57	36.34

Table 6: Performance of RECOST with increasing diversity. Tolerance '-' refers to RECOST without diversity sampling.

5.4 Analysis

5.4.1 Effectiveness of RECOST

Moreover, we make a comparison of dirty data hit rates on RECOST, predictive entropy, and IFD. Similar to Figure 1, we calculate the hit rates of three metrics. As presented in Figure 4, our proposed RECOST outperforms both metrics remarkably on top 20% samples. Besides, we further compare the performance of the three metrics above in Section A.3.

5.4.2 The Effects of Mixed Weight

To find the optimal mixed weight w, we conduct extensive experiments with different weights on both datasets. We set the mixed weight w to 0, 0.25, 0.5, 0.75, and 1. In particular, $w = 1$ refers to vanilla predictive entropy while $w = 0$ refers to our proposed relative predictive entropy.

As presented in Figure 5, the performance of the models on AlpacaEval demonstrates an initial increase followed by a decline, peaking at values of $w = 0.5$ and $w = 0.75$, respectively. Specifically, due to the overall inferior quality of the Alpaca dataset in comparison to Alpaca-gpt4, coupled with its shorter average response length, a greater emphasis on weighting is inclined towards selecting longer samples, which in turn yields superior outcomes.

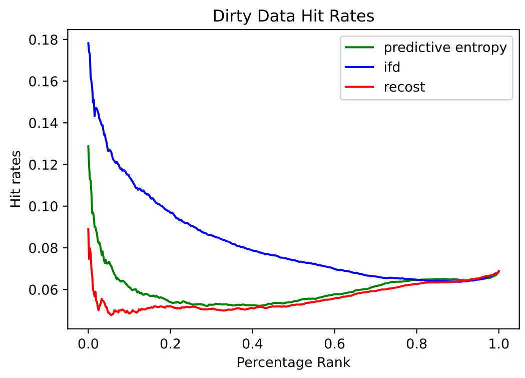


Figure 4: Comparison of dirty data hit rates under different metrics on the Alpaca dataset calculated by LLaMA-2-7b. The horizontal axis represents the percentage ranking, while the vertical axis denotes the proportion of corrupted data within the data preceding that percentage threshold. Given the number of dirty data in the top i data points as d_i, the hit rate at i is calculated by d_i/i. The dirty data is collected by comparing Alpaca with Alpaca-cleaned.

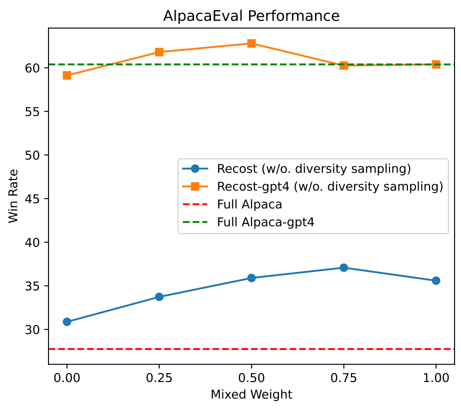


Figure 5: Performance of relative predictive entropy as the change of mixed weight w.

6 Conclusion

In this paper, we demonstrate **RECOST**, an effective method to select high-quality instruction data from synthetic instruction datasets with the help of external knowledge. Our method outperforms the fully-trained models with only 1% training data, which surpasses the previous methods under the same settings. Overall, our approach underscores the significance and efficacy of integrating our proposed relative uncertainty into data-efficient instruction tuning for synthetic datasets, providing a viable avenue for this field.

7 Limitations

The primary limitation of our work lies in the necessity of incorporating additional external knowledge. However, thanks to the development of traditional NLP tasks during the pre-LLM era and the current advancements in retrieval-based ICL, we can easily obtain a vast amount of authentic and reliable external knowledge to meet our requirements. Overall, our research empirically validates the feasibility of integrating exogenous knowledge in the data filtering process based on synthetic data. Although this introduces a minor overhead in data preprocessing, it significantly outperforms previous methods within an acceptable cost margin, offering new perspectives in the realm of data efficiency.

8 Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (No. 226-2024-00049), the Pioneer R&D Program of Zhejiang (No. 2024C01035), and NSFC under Grants (No. 62206247).

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.

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

Hao Chen, Yiming Zhang, Qi Zhang, Hantao Yang, Xiaomeng Hu, Xuetao Ma, Yifan Yanggong, and Junbo Zhao. 2023a. Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023b. Alpagasus: Training a better alpaca with fewer data.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. 2023. Shifting attention to relevance: Towards the uncertainty estimation of large language models.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Lirong Gao, Ru Peng, Yiming Zhang, and Junbo Zhao. 2024. Dory: Deliberative prompt recovery for llm. *arXiv preprint arXiv:2405.20657*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know.

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

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

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*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*.

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

Man Luo, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. 2024. In-context learning with retrieved demonstrations for language models: A survey.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023b. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Kaihang Pan, Juncheng Li, Wenjie Wang, Hao Fei, Hongye Song, Wei Ji, Jun Lin, Xiaozhong Liu, Tat-Seng Chua, and Siliang Tang. 2024. I3: Intent-introspective retrieval conditioned on instructions.

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

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

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`.

Liang Wang, Nan Yang, and Furu Wei. 2024. Learning to retrieve in-context examples for large language models.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? exploring the state of instruction tuning on open resources.

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.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

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*.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2019. Quick and (not so) dirty: Unsupervised selection of justification sentences for multi-hop question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

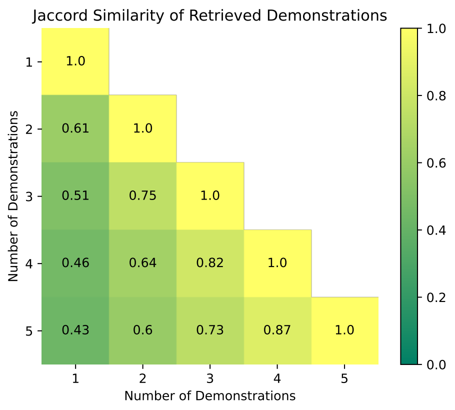
Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. 2023. Retrieve anything to augment large language models.

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.

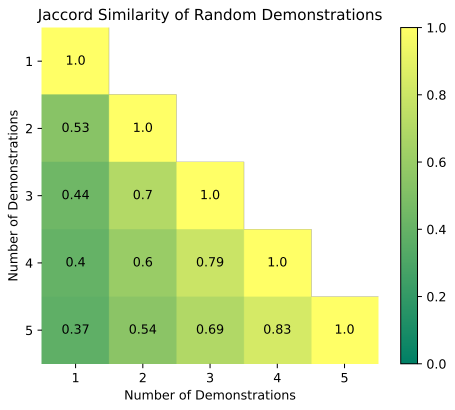
A Appendix

A.1 Robustness Analysis of Relative Uncertainty

As presented in 6, with the increase in the number of demonstrations, RECOST with retrieved demonstrations tends to be more stable than that with random demonstrations.



(a)



(b)

Figure 6: The robustness of relative uncertainty as the number of demonstrations changed when we used random samples as demonstrations. In Figure 6a, we use retrieved demonstrations, while we use random demonstrations in Figure 6b.

A.2 Performance with Different Numbers of Retrieved Demonstrations

Furthermore, with different numbers of retrieved demonstrations, the performance of RECOST varies slightly as shown in Table 7:

Numbers of Retrieved Demonstrations	OpenLLM	AlpacaEval
3	56.13	37.36
4	56.02	39.25
5	56.07	39.19

Table 7: Performance with different numbers of retrieved demonstrations.

A.3 Effectiveness of RECOST

We present the comparison between the aforementioned metrics in Table 8. RECOST outperforms all baselines in both benchmarks.

Method	Data size	OpenLLM	AlpacaEval
Full Alpaca	52002	55.06	27.75
IFD	3111	55.78	36.78
PE	520	55.31	35.59
RECOST	520	**56.07**	**39.19**

Table 8: Comparison of benchmark scores under different metrics on the Alpaca dataset.