SELECTLLM: Can LLMs Select Important Instructions to Annotate?

Anonymous ACL submission

Abstract

Instruction tuning benefits from large and diverse datasets, however creating such datasets involves a high cost of human labeling. While synthetic datasets generated by large language models (LLMs) have partly solved this issue, they often contain low-quality data. One effective solution is selectively annotating unlabelled instructions, especially given the relative ease of acquiring unlabeled instructions or texts from various sources. However, how to select unlabelled instructions is not well-explored, especially in the context of LLMs. Further, traditional data selection methods, relying on input embedding space density, tend to underestimate instruction sample complexity, whereas those based on model prediction uncertainty often struggle with synthetic label quality. Therefore, we introduce SELECTLLM, an alternative framework that leverages the capabilities of LLMs to more effectively select unlabeled instructions. SELECTLLM consists of two key steps: Coreset-based clustering of unlabelled instructions for diversity and then prompting a LLM to identify the most beneficial instructions within each cluster. Our experiments demonstrate that SELECTLLM matches or outperforms other state-of-the-art methods in instruction tuning benchmarks. It exhibits remarkable consistency across human and synthetic datasets, along with better cross-dataset generalization, as evidenced by a 10% performance improvement on the Cleaned Alpaca test set when trained on Dolly data.[1]

1 Introduction

Instruction tuning, which fine-tunes language models (LMs) to follow instructions constructed from a massive number of diverse and different tasks, has shown impressive generalization performance on various unseen tasks (Wei et al., 2022; Chung et al., 2022). However, creating large and diverse annotated instruction datasets is a major challenge due to the significant cost of human labeling. While synthetic datasets generated by advanced large language models (LLMs) have partly addressed this issue (Taori et al., 2023; Wang et al., 2022a), they often contain low-quality data, highlighting the need for more focus on dataset refinement (Zhou et al., 2023; Cao et al., 2023).

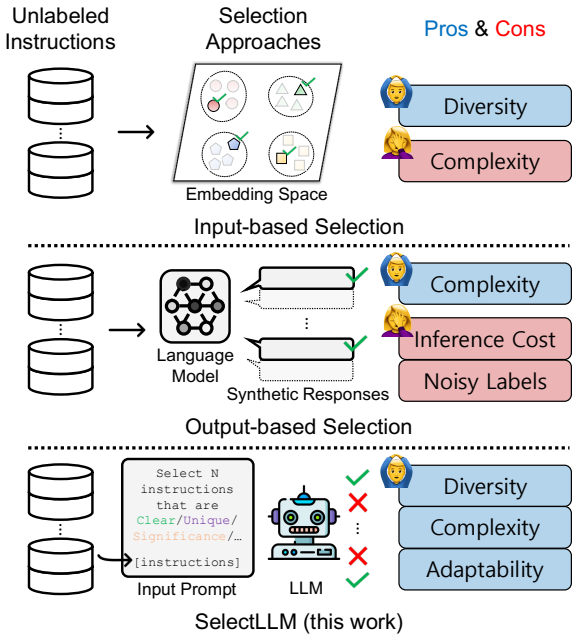

Figure 1: Conceptual comparison between previous approaches to select instructions and SELECTLLM. Focusing on input instructions (**top**) is unable to consider the difficulty or uncertainty of response. Output-based methods (**middle**) can suffer from the inference cost and quality issues of synthetic responses. SELECTLLM (**bottom**) does not suffer from these issues by estimating the effectiveness of instructions via prompting LLMs.

As it is relatively easy to access unlabelled instructions or texts from various sources, one promising solution to address these challenges is developing a way to select important unlabelled instructions for annotation, similar to active learning (Settles, 2009); from this, one can create a high-quality small dataset if unlabelled instructions

[1]We will release the code and data upon acceptance.

are well-selected and annotated accordingly. However, existing selection methods present limitations. Active learning-based algorithms focusing on *density* in the embedding space (Sener and Savarese, 2018) often overlook the complexity of instruction samples. Alternatively, output-based approaches that assess *uncertainty* in model predictions (Kung et al., 2023) grapple with high computational costs and the challenges posed by synthetic label quality (Figure 1). In light of these challenges, a few recent works show the capability of LLMs for measuring the quality or relevance of multiple texts (Sun et al., 2023; Liu et al., 2023). Inspired by these, we investigate the potential of LLMs *to select effective unlabelled instructions, leveraging their vast knowledge base to discern the complexity and utility of each instruction without generating inferences.*

Contribution. In this work, we introduce SELECTLLM, a method that selects an effective subset of unlabelled instructions by prompting LLMs. At a high level, our method uses LLMs to estimate the usefulness and impact of each instruction *without the corresponding labels*; through the initial experiments presented in Figure 3, we first verify that LLMs do possess such a capability. To improve instruction selection via LLMs, SELECTLLM first divides the entire dataset into several small subsets to construct the input queries for prompting; specifically, we use equal-size K-means clustering to create each subset with diverse unlabelled instructions while preserving the overall dataset structure. Then, SELECTLLM constructs an input query for each subset using a carefully designed input prompt for selecting with LLMs. Finally, we forward these constructed queries into the LLM, and SELECTLLM selects a few instructions that are expected to be the most helpful in fine-tuning models.

We demonstrate the effectiveness of SELECTLLM compared to various state-of-the-art selection methods on two popular benchmarks for instruction tuning, Dolly (Conover et al., 2023) and Cleaned Alpaca (Taori et al., 2023). Our experiments show that SELECTLLM consistently outperforms the baselines across the varied selection sizes. For instance, when selecting samples from Dolly, SELECTLLM exhibited nearly 2.5% - 3% relative improvements compared to the strongest baseline on the Rouge-L F1 score and cosine similarity, respectively across all sample sizes (1k/2k/3k). In addition, SELECTLLM shows better cross-task generalizations, an important characteristic of instruction-tuned LMs. More interestingly, we observe that the generated responses by SELECTLLM are preferred to the responses from other baselines when evaluated with GPT-4.

Additionally, the input prompt in SELECTLLM is primarily designed to select instructions that enhance the overall performance of LLMs. However, its flexible nature allows for easy customization to meet specific user needs (*e.g.,* reducing toxicity), by simply adjusting SELECTLLM's input prompt during sample selection. This adaptability of our framework paves the way for using LLMs to select samples with various desired characteristics, tailored to suit distinct use cases, which is not possible with previous selection methods.

2 Related Work

Instruction tuning for LMs. Instruction tuning (Wei et al., 2022), a form of fine-tuning LLMs, has emerged as a prominent methodology to align pre-trained LMs for various tasks by describing the tasks in the common form of instructions. Due to its ease of implementation and remarkable generalization capabilities for unseen tasks (Wei et al., 2022; Chung et al., 2022; Jang et al., 2023), it has gained substantial popularity recently. Constructing these instruction datasets with human annotations is a standard way (Conover et al., 2023; Sanh et al., 2021; Wang et al., 2022b), but this method faces challenges in terms of variety of instructions and the total number of instances due to labeling cost (Wang et al., 2022a); one promising solution for this limitation is to synthesize existing datasets and create diverse, multi-task datasets with the help of LLMs such as Alpaca (Taori et al., 2023) and Self-instruct (Wang et al., 2022a). However, using LLM-created data increases the risk of including low-quality examples, and it is known that removing such noise from the dataset is critical for the effective instruction tuning of LLMs (Zhou et al., 2023; Cao et al., 2023); therefore, in this work, we explore an alternative way to use LLMs to construct a high-quality instruction dataset, by using LLMs to select unlabeled instructions.

Sample selection for efficient instruction tuning. Expanding instruction datasets has its own set of challenges, including the need for extensive resources, time, human annotation, and the prevalence of redundant data. A common solution involves human intervention, such as the approach by

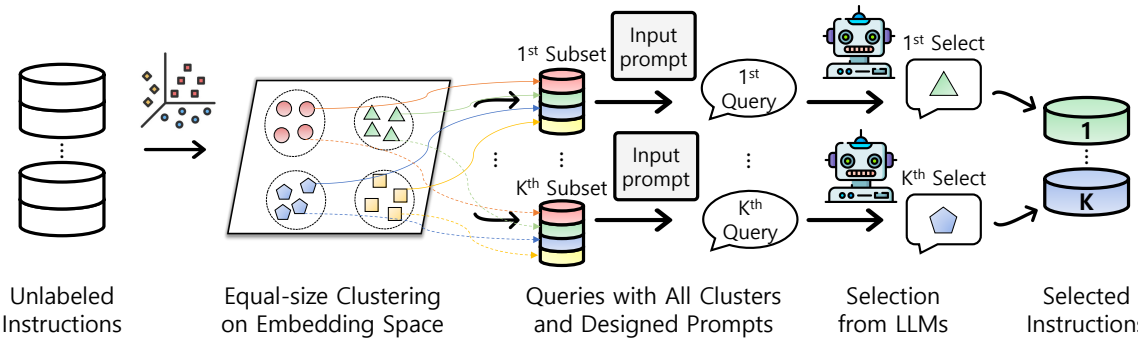

Figure 2: Illustration of the proposed SELECTLLM.

Zhou et al. (2023), which manually annotates high-quality instructions after filtering out low-quality data. However, there are huge costs associated with such procedures done by human labor. An effective solution is to gather a large pool of unlabeled instructions and then selectively annotate the most useful ones, similar to the popular task of active learning (Settles, 2009; Sener and Savarese, 2018). Here, the most important thing is *selection criteria*.

One line of work focuses on *density* to include *diverse* instructions; for example, Chen et al. (2023a) map the unlabelled instructions into an embedding space and use K-means clustering (Hartigan and Wong, 1979) and the K-center greedy algorithm (Sener and Savarese, 2018) for the selection. Another line of work focuses on *uncertainty* (or *difficulty*) of instructions measured with LLMs' outputs; Kung et al. (2023) measure uncertainty by observing how LLM-generated responses vary with changes in the input instructions. Unlike these approaches, we directly prompt LLMs with unlabelled instructions, to select the few examples expected to help train the model; from LLM's capability of reasoning and generating useful responses, we assume that they could infer the impactfulness of the unlabelled instructions. Meanwhile, a recent study by (Chen et al., 2023b) also prompts LLMs to construct a high-quality instruction dataset. However, our work differs as we focus on selecting unlabelled instructions while (Chen et al., 2023b) focuses on filtering out low-quality labeled instructions.

3 SELECTLLM: Select Important Unlabelled Instructions Using LLMs

3.1 Preliminary

We first describe the problem setup of our interest. Let $\mathcal{X} = \{x_i\}_{i=1}^{M}$ denote the given unlabeled dataset, where x_i represents ith unlabeled instruction and M is the total number of instructions. Then, our goal is to select N most effective instructions from $\mathcal{X}$ which will be labeled by human annotators, to fine-tune a target large language model (LLM) f_θ, *e.g.*, LLaMA-2 (Touvron et al., 2023), and make it be generalized for various instructions. Formally, we select N instructions from $\mathcal{X}$ under a selection criteria $s(j)$:

$$\mathcal{X}(S) = \{x_{s(j)}\}_{j=1}^{N}, \quad \text{where } S = \{s(j)|s(j) \in [1, M]\}_{j=1}^{N} \qquad (1)$$

where S denotes the indices of selected instructions. Then, the selected instruction $x \in \mathcal{X}(S)$ is labeled with a corresponding label y by human annotators, and it results in the annotated instruction dataset $\mathcal{D}$, where $\mathcal{D} = \{(x_{s(j)}, y_{s(j)})\}_{j=1}^{N}$. Therefore, the performance of LLM f_θ fine-tuned on $\mathcal{X}(S)$ significantly varies depending on which selection criteria $s(j)$ is considered. While various selection criteria have been explored under tasks like active learning (see Section 2), this direction is less explored under the paradigm of instruction tuning.

3.2 Selection via prompting LLMs

For the selection criteria $s(j)$, SELECTLLM proposes to use LLMs with a properly designed prompt without using ground truths or generated labels. Our high-level intuition is that LLMs can infer the potential impact of each instruction by only reading the instruction; as shown in Figure 3, we observed that the recent LLM, *e.g.*, ChatGPT, could estimate the effectiveness of each instruction for model training (*e.g.*, LLaMA-2), even without the corresponding labels. To further improve the effectiveness of selection via LLMs, we carefully designed the input prompt to incorporate several important perspectives for instruction tuning, and it is presented in Figure 4. Formally, this process could be described as follows: we first assume that

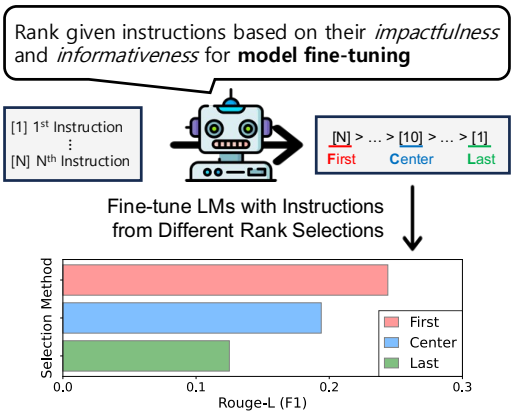

Figure 3: Experiments to verify LLMs' capability to infer the importance of *unlabelled* instructions. We prompt ChatGPT to sort the instructions based on their effectiveness for model training; then, we compare the performance of three fine-tuned LMs (LLaMA-2) on instructions with different ranks (First, center, and last). A full prompt is presented in Appendix C.

the dataset $\mathcal{X}$ is divided into K non-overlapped subsets, *i.e.*, $\mathcal{X} = \bigcup_{k=1}^{K} \mathcal{X}_k$. Then, we construct input query q_k using the designed prompt $p_{\texttt{sel}}$ and $\mathcal{X}_k$, and forward it to LLM to select $\widetilde{N} = \lfloor N/K \rfloor$ examples:

$$S_k = \texttt{LLM}\big(p_{\texttt{sel}}(q_k, \widetilde{N})\big) \tag{2}$$

where $S_k = \{s_k(j)\}_{j=1}^{\widetilde{N}}$, $s_k(j) \in [1, N]$.

3.3 Composing query of LLMs via clustering

To further improve the effectiveness of using LLMs for selection, we carefully design how to divide the entire dataset into several subsets which would be used to construct input queries, based on the equal-size clustering method. Here, our high-level idea is composing the subsets that maximize the diversity among the instructions while maintaining the global structure of the dataset. Specifically, we first extract the embeddings of the instructions in $\mathcal{X}$, using the pre-trained sentence encoder g_ϕ such as Sentence-BERT (Reimers and Gurevych, 2019a). Then, we conduct K-means clustering (Hartigan and Wong, 1979) on these embeddings, and calculate $D \in \mathbb{R}^{N \times K}$, the distance of all instances in $\mathcal{X}$ to K cluster centers $c_1, \cdots, c_K$. Based on the distances, we assigned each instance x among $[1, K]$, by iteratively taking the one with the shorted distance to the cluster center among the remaining instances, to guarantee equal sizes for each k.

Overall, the selection procedure of SELECTLLM is as follows: (1) construct input queries by separating the entire dataset into multiple subsets of diverse instructions. Then, (2) feed these queries into a LLM and get the selected indices. The formal presentation of these procedures is presented in Algorithms 1 and 2 in Appendix.

```
The following are {N} candidate instructions
that describe a task, each indicated by a
number identifier [].

[1]
### Instruction: {Example #1 Instruction}
### Input: {Example #1 Input}
.
.
.
[N]
### Instruction: {Example #N Instruction}
### Input: {Example #N Input}

Examine the provided list of {N} instructions
, each uniquely identified by a number in
brackets [].

Your task is to select {num} instructions
that will be annotated by human annotators
for model fine-tuning.

Look for instructions that are clear and
relevant, exhibit a high level of complexity
and detail, represent a diverse range of
scenarios and contexts, offer significant
instructional value and potential learning
gain, and present unique challenges and
specificity.

These selected instructions should ideally be
 the most beneficial for model fine-tuning
after being annotated by human annotators.

Present your selections using the format [].
e.g., [1,2] or [2,3].

The most impactful {num} instructions (only
identifiers) are:
```

Figure 4: Designed input prompt of SELECTLLM.

4 Experiments

In this section, we design our experiments to investigate the following questions:

- Does SELECTLLM outperform the previous state-of-the-art selection methods for instruction tuning LLMs? (Tables 1, 2, 3) If so, why? (Table 7)
- Do LLMs tuned with SELECTLLM provide good responses qualitatively? (Tables 4, 6)
- What is the effect of each component in SELECTLLM? (Table 5)

4.1 Setups

Datasets and metrics. We use labeled datasets without using their responses to test our hypothesis. Further, we utilize one human-generated dataset

Table 1: Experimental results on Dolly (Conover et al., 2023). Rouge-L (F1) and Cosine similarity of generated responses from fine-tuned LLaMA-2 by different numbers of examples are compared. The best and second best scores are highlighted in **bold** and underline, respectively.

Methods	Rouge-L (F1)			Cosine Similarity			Avg Across Sizes	
	$1k$	$2k$	$3k$	$1k$	$2k$	$3k$	Avg Rouge	Avg Cosine
$\text{Length}_{\text{short}}$	0.073	0.109	0.130	0.192	0.265	0.336	0.104	0.264
Perplexity	0.158	0.183	0.192	0.402	0.433	0.453	0.178	0.429
$\text{CBS}_{\text{sbert}}$	0.147	0.200	0.216	0.359	0.473	0.512	0.188	0.448
$\text{Length}_{\text{long}}$	0.256	0.247	0.238	0.641	0.626	0.611	0.247	0.626
$\text{CBS}_{\text{instr}}$	0.258	0.255	0.255	0.617	0.638	0.632	0.256	0.629
Random	0.239	0.264	0.278	0.589	0.644	0.650	0.260	0.628
Diversity	0.237	0.275	0.282	0.582	0.650	0.666	0.265	0.633
OpenEnd	0.258	0.271	0.282	0.627	0.641	0.669	0.270	0.646
Coreset	0.271	0.281	0.279	0.649	0.662	0.659	0.277	0.657
Ours	**0.278**	**0.288**	**0.289**	**0.668**	**0.680**	**0.686**	**0.285**	**0.678**

and one machine-generated dataset. For the former, we use (Conover et al., 2023) which is a combined effort of several Databricks employees, and as for the latter, we use Cleaned Alpaca which is based on (Taori et al., 2023) but is cleaned up to fix any errors in the input prompts of the original dataset and has responses generated by GPT-4. For evaluating the performance, we assess the similarity between inferred and actual texts using Rouge scores and cosine similarity. Additionally, we perform a GPT-based analysis to gauge the effectiveness of the generated inferences.

Baselines. We consider several baselines for comparison with our algorithm as follows: *(1) Random*: selecting instances from the unlabeled dataset purely randomly. *(2) Length*: Considers the length of input instruction, focusing on both longer and shorter ones to evaluate their impact ($\text{Length}_{\text{long}}$ and $\text{Length}_{\text{short}}$). *(3) Cluster-Based Selection (CBS)* (Chen et al., 2023a): transforming instructions into embedding space, clustering them with HDBSCAN (Campello et al., 2013), and selecting samples using the K-Center-Greedy algorithm. We consider two different embedding spaces with Sentence-BERT (Reimers and Gurevych, 2019a) and InstructOR (Su et al., 2023), and denote them as $\text{CBS}_{\text{sbert}}$ and $\text{CBS}_{\text{instr}}$, respectively. *(4) Perplexity* (Marion et al., 2023): selecting samples based on low per-token perplexity, indicating high model certainty and fluency. *(5) Diversity* (Wang et al., 2022a): for each instruction in the dataset, Rouge score is computed against a randomly selected subset comprising n samples ($n \ll M$). Then, we select k samples that exhibit the minimum Rouge scores. *(6) Open-Endedness (OpenEnd)* (Li et al., 2023): generating three inferences per prompt, counting unique bigrams, and selecting samples with the greatest variety of bigrams. *(7) Coreset* (Sener and Savarese, 2018): Similar to CBS, transforming instructions into embedding space with Sentence-bert, then selecting samples with K-Center-Greedy algorithm (Sener and Savarese, 2018).

Implementation Details. We utilize the Dolly and Cleaned Alpaca datasets for training and evaluation. Due to the absence of predefined train-test splits in these datasets, we allocated 1k samples from each for testing, leaving 14k and 51k samples in the Dolly and Cleaned Alpaca datasets, respectively, for training. We employed sampling algorithms to select subsets of 1k to 3k samples from each train set. The LLaMA-2 (7B) model was fine-tuned using QLoRa (Touvron et al., 2023) to optimize memory usage during training. For instruction selection with SELECTLLM, ChatGPT (`gpt-3.5-turbo-0613`) was utilized. To ensure robustness in our results, we conducted experiments using three different random seeds. The models' evaluation scores were averaged to derive a final score for each method. Further details regarding the training process are provided in the Appendix.

4.2 Main Results

In this section, we present our main experimental results. Our comprehensive evaluation, incorporating both Rouge scores and Cosine Similarity metrics, provides a detailed insight into the performance variations across different sample sizes (1k, 2k, and 3k), as well as the average perfor-

Table 2: Experimental results on Cleaned Alpaca (Conover et al., 2023). Rouge-L (F1) and Cosine similarity of generated responses from fine-tuned LLaMA-2 by different numbers of examples are compared. The best and second best scores are highlighted in **bold** and underline, respectively.

Methods	Rouge-L (F1)			Cosine Similarity			Avg Across Sizes	
	$1k$	$2k$	$3k$	$1k$	$2k$	$3k$	Avg Rouge	Avg Cosine
$\text{Length}_{\text{short}}$	0.219	0.263	0.261	0.519	0.632	0.625	0.248	0.592
Perplexity	0.264	0.278	0.272	0.628	0.646	0.640	0.271	0.638
$\text{CBS}_{\text{sbert}}$	0.254	0.264	0.288	0.598	0.618	0.665	0.269	0.627
$\text{Length}_{\text{long}}$	0.228	0.266	0.297	0.550	0.618	0.683	0.264	0.617
$\text{CBS}_{\text{instr}}$	0.257	0.280	0.292	0.610	0.655	0.673	0.276	0.646
Random	**0.281**	0.281	0.271	0.653	0.662	0.656	0.278	0.657
Diversity	0.268	**0.286**	0.297	0.650	**0.673**	0.690	0.284	0.671
OpenEnd	0.250	0.247	0.276	0.616	0.601	0.645	0.258	0.621
Coreset	0.277	0.267	0.299	0.650	0.635	0.695	0.281	0.660
Ours	0.276	0.277	**0.301**	**0.661**	0.667	**0.707**	**0.285**	**0.678**

mance across these sizes. The results for Dolly are detailed in Table 1 and for Alpaca in Table 2, respectively. To be specific, our analysis leads to several nuanced observations:

Dominant performance of SELECTLLM. *SelectLLM* consistently outperforms other methods in the Dolly dataset, maintaining a lead with an average improvement of 2.6% in Rouge Score and 3% in Cosine Similarity across all sample sizes. This highlights SELECTLLM's adaptability and effectiveness in processing human-generated data. In the Cleaned Alpaca dataset, SELECTLLM shows its strength particularly at the 1k and 3k sample sizes, outperforming others on the cosine similarity metric. While its performance at the 2k size is slightly lower, the overall trend underscores its reliability across various data volumes.

Consistent effectiveness across datasets. SELECTLLM exhibits unparalleled consistency in both human and synthetic datasets. This uniformity across sample sizes sets it apart from other baselines and demonstrates its broad applicability. In contrast, other methods like Coreset, Diversity, and $\text{Length}_{\text{long}}$ show fluctuating performances depending on the dataset and sample size. For example, Coreset varies notably with sample size, while Diversity and $\text{Length}_{\text{long}}$ excel in the Cleaned Alpaca dataset but falter in the Dolly dataset. OpenEndedness performs better in Dolly but shows decreased effectiveness in Cleaned Alpaca. This further highlights the robust and adaptable nature of SELECTLLM.

Cross-dataset generalization. We analyze how well models trained on dolly samples using various sampling techniques generalize to cleaned alpaca data, and vice versa. Our results are presented in Table 3 for both datasets. We made an intriguing observation with regards to models trained on dolly data - the model trained using our approach remarkably performed better than all the baselines by 10% on the cleaned alpaca test set. On the Cleaned Alpaca dataset, Coreset shows comparable performance to SELECTLLM in terms of Cosine Similarity and slightly better performance in Rouge scores. Further, Cleaned Alpaca appears to be a better dataset for cross-evaluation generalization when observing the performance of all baselines trained on it.

Table 3: Cross-dataset generalization for 3k sample size. Rouge-L (F1) / Cosine similarity of generated responses from LLaMA-2. The best scores are highlighted in **bold**, and column names indicate the dataset trained on.

Methods	Dolly	Cleaned Alpaca
Random	0.205 / 0.589	0.260 / 0.669
OpenEnd	0.208 / 0.627	0.244 / 0.640
Coreset	0.208 / 0.651	**0.271 / 0.684**
SELECTLLM	**0.229 / 0.668**	0.263 / 0.683

Evaluation with GPT-4. We evaluate the quality of generated responses between LLaMA-2 fine-tuned with SELECTLLM and other methods on a 1k size instruction dataset, randomly sampled from the Dolly dataset. We ask GPT-4 to choose the better response to a given instruction, similar to (Liu et al., 2023). As shown in Table 4, SELECTLLM wins in 52% of the cases when compared to Random sampling, and 44% cases when compared to Coreset sampling, further showcasing better infer-

Table 4: Win-Tie-Draw from GPT-4 evaluation (%) on SELECTLLM against Random and Coreset with Dolly.

Compared Methods	Win	Tie	Lose
SELECTLLM vs Random	**52.1**	18.4	29.5
SELECTLLM vs Coreset	**44.2**	25.4	30.4

Table 5: Ablation study. Rouge and cosine similarity of generated responses from fine-tuned LLaMA-2 by selecting 1k examples on Dolly with different methods are compared. The best scores are highlighted in **bold**.

Division	Selection	Rouge-L (F1)	Cosine Sim
Random	Random	0.239	0.589
Random	OpenEnd	0.258	0.627
Random	LLM	0.274	0.651
$\text{Sim}_{\texttt{KMeans}}$	LLM	0.264	0.625
$\text{Div}_{\texttt{KMeans}}$	Random	0.236	0.585
$\text{Div}_{\texttt{KMeans}}$	OpenEnd	0.251	0.617
$\text{Div}_{\texttt{KMeans}}$	LLM	**0.278**	**0.668**

ence quality of SELECTLLM. The prompt for the evaluation is provided in the appendix (Figure 6).

4.3 More Analyses with SELECTLLM

4.3.1 Ablation study

To show the effectiveness of each component of SELECTLLM, we ablate against different combinations of local selection prompting methods (Sec 3.2) and global division algorithms (Sec 3.3) on 1k samples from Dolly. Table 5 shows the results.

Local selection. We first compare our prompting-based selection method in section 3.2 with different non-prompting techniques, under the same global division method such as Random and SELECTLLM's method (called $\text{Div}_{\texttt{Kmeans}}$). We observe that other selection methods, Random and OpenEndedness, are not as effective in comparison, highlighting the superiority of LLM-based selection in selecting higher-quality instructions for training other LLMs, without true labels.

Global division. Next, we compare the global division method, $\text{Div}_{\texttt{Kmeans}}$, with $\text{Sim}_{\texttt{Kmeans}}$ and random sampling. $\text{Sim}_{\texttt{Kmeans}}$ is a method that clusters instruction embeddings and constructs input queries with similar instructions, rather than diverse ones. We observe that $\text{Sim}_{\texttt{Kmeans}}$ performs the worst, indicating that having diverse instructions to choose from helps LLMs perform a better local selection. This is also highlighted by the result that Random division performs better than $\text{Sim}_{\texttt{Kmeans}}$, but worse than $\text{Div}_{\texttt{Kmeans}}$.

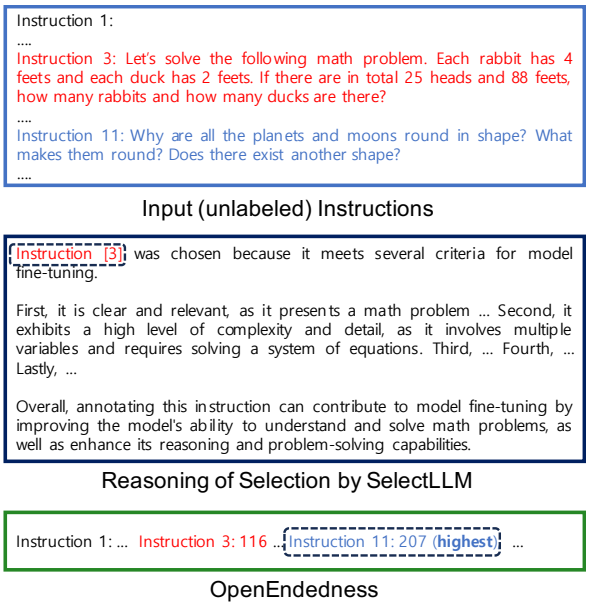

Figure 5: Qualitative example of selection with a given query composed of 14 instructions on Dolly.

4.3.2 Qualitative results

Comparison of outputs from fine-tuned LLMs. We evaluate responses from LLMs fine-tuned with SELECTLLM against Random and Coreset baselines, focusing on their handling of closed and open QA formats. Key examples are in Table 6, with more in the appendix (Table 9).

In the first example in Table 6, SELECTLLM demonstrates superior precision and adherence to the instruction compared to the other methods. While the Coreset and Random methods provide correct responses, they tend to replicate the input text without much specificity. In contrast, SELECTLLM generates a response that is not only precise and direct but also effectively extracts and succinctly presents the relevant information.

The second example, based on an Open-ended task further underscores SELECTLLM's nuanced understanding and response capability. While Random and Coreset give basic, concise answers, SELECTLLM adds personalized, empathetic elements (highlighted in blue), showing deeper instruction comprehension. This reflects the advanced response capabilities essential in instruction-tuned models.

Analysis of chosen instructions. We provide additional experiments to examine *why* SELECTLLM *could be effective compared to other selection methods.* To this end, we first conduct statistical analysis for the instructions selected by Random, Coreset, OpenEnd, and SELECTLLM (Table 7). The results show that (1) SELECTLLM selects high quality

Table 6: Examples of generated sentences from fine-tuned LLMs with different selection methods.

Instruction (#1): Where was Sarah Schechter born?
Input: Schechter was born in Boston and grew up in Brookline, Massachusetts and New York City. Her father, Danny Schechter, a grandson of Russian-Jewish immigrant socialists, was a human rights activist, author, filmmaker and television producer.
Ground Truth: Sarah Schechter was born in Boston, Massachusetts.

Random	**Coreset**	**Ours**
Sarah Schechter was born in Boston and grew up in Brookline, Massachusetts and New York City. Her father, Danny Schechter, a grandson of Russian-Jewish immigrant socialists, was a human rights activist, author, filmmaker and television producer. *### Instruction: ··· ### Input: ··· ### Response: ··· ### Instruction:*	*Sarah Schechter was born in Boston and grew up in Brookline, Massachusetts and New York City. Her father, Danny Schechter, a grandson of Russian-Jewish immigrant socialists, was a human rights activist, author, filmmaker and television producer.*	*Sarah Schechter was born in Boston, Massachusetts.*

Instruction (#2): Write an email to the head of academic coaching at Services for Students with Disabilities sharing that I've applied to the coaching position. I have experience volunteering with children on the autism spectrum.
Ground Truth:
Hi,
I hope you are having a wonderful day. My name is _, and I just applied for the Program Staff position at the Services for Students with Disabilities to empower students who have disabilities to reach their academic dreams. My background working in higher education and volunteering with kids on the Autism spectrum combine to bring an empathetic and impactful perspective. Please let me know if there is anything else I can do at this time to be considered for this position.
With gratitude,

Random	**Coreset**	**Ours**
Dear Head of Academic Coaching, *I am writing to share that I have applied to the coaching position. I have experience volunteering with children on the autism spectrum.* *Thank you for your time.* *Sincerely,* *Katie*	*I'm writing to share that I've applied to the coaching position. I have experience volunteering with children on the autism spectrum.*	*Dear Head of Academic Coaching,* *I am writing to share that I have applied to the coaching position. I have experience volunteering with children on the autism spectrum. I am passionate about helping children with disabilities and I believe that I would be a great asset to your team. I am confident that I can make a positive impact on the lives of the students that I work with.*

Table 7: Analysis of 1k selected instructions with different approaches on Dolly. Diversity is measured using kNN distance (Carbonera and Abel, 2015) with $k = 1$. Perplexity is measured with GPT2-large (Radford et al., 2019), and Length is the number of characters.

Methods	Diversity (↑)	Perplexity (↓)	Length
Random	0.721	89	460
Coreset	**0.931**	47	847
OpenEnd	0.710	71	646
SELECTLLM	0.796	**30**	**1417**

(*i.e.*, lower perplexity) instructions with more details (*i.e.*, longer length), and (2) the selected instructions are considerably diverse; it demonstrates the effectiveness of selection by LLMs and composing diverse query via clustering, respectively. Next, to further explore the advantage of selection via LLMs over existing approaches, we conduct an additional comparison between SELECTLLM and the method that uses OpenEnd for local selection and $\text{Div}_{\texttt{Kmeans}}$ for global division, which is presented in 6th row in Table 5. Similar to the experiments in Table 4, we provide two different indices selected by these methods, and ask ChatGPT which selection is better (Full prompt is presented in Figure 8). Here, we find that the selection by SELECTLLM is more preferred (31.8% vs 28.0%), and it indicates that the selection itself is beneficial from LLMs not only during fine-tuning. Lastly, we present specific examples of the selections with two methods, along with the rationales for the selection with SELECTLLM generated via zero-shot chain-of-thought with ChatGPT (Kojima et al., 2022). As shown in Figure 5, we observe the several underlying rationales considered by LLM in making its selection. More examples are in Appendix D.

5 Conclusion

We introduce SELECTLLM, a new approach that uses LLMs to choose an efficient subset from a set of unlabelled instructions. Our experiments on two popular benchmarks show that SELECTLLM is more effective than previous selection methods. This demonstrates how LLMs can improve the efficiency of instruction tuning for language models.

Limitations

Despite the impressive performance of SELECTLLM, it is not without its limitations. A primary concern is the expense associated with utilizing LLMs like ChatGPT for data selection, which can be substantial. Additionally, the scalability of SELECTLLM when dealing with exceptionally large datasets, or in scenarios requiring real-time data annotation, remains an area that needs further exploration. This aspect is particularly crucial given the ever-increasing size of datasets and the imperative for efficient processing in a wide range of practical applications. Hopefully, these limitations will be addressed in the future upon our work.

Broader Impact and Ethical Implications

The findings from our research not only establish the proficiency of LLMs in autonomously selecting high-quality data for training but also open new paths for future investigation in this domain. The successful application of LLMs in data-constrained environments is demonstrated by the exceptional ability of SELECTLLM. This study, therefore, marks a significant stride in the field of instruction tuning for LLMs, paving the way for more efficient and effective training methodologies and expanding the scope of autonomous capabilities of LLMs. In terms of ethical implications, the potential for any risk is limited to the application of LLMs in our framework, and the general risks associated with them such as LLMs showing bias in selecting certain instructions according to what it believes to be an impactful instruction. Further, bias can also be introduced based on how the prompt is designed by the user, when querying the LLMs in our framework.

References

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

Joel Luis Carbonera and Mara Abel. 2015. A density-based approach for instance selection. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 768–774. IEEE.

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. *arXiv preprint arXiv:2305.09246*.

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*.

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

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

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the benefits of training expert language models over instruction tuning.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Po-Nien Kung, Fan Yin, Di Wu, Kai-Wei Chang, and Nanyun Peng. 2023. Active instruction tuning: Improving cross-task generalization by training on prompt sensitive tasks.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pretraining llms at scale.

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.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach.

Burr Settles. 2009. Active learning literature survey.(2009).

Hongjin Su, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, Tao Yu, et al. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agent. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language model with self generated instructions. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Super-naturalinstructions:generalization via declarative instructions on 1600+ tasks. In *EMNLP*.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners.

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 More Details about Experiments

A.1 Datasets

1) Dolly (Conover et al., 2023): The Dolly dataset, developed by Databricks, Inc., is a comprehensive collection of over 15,000 human-generated instruction-following records. It encompasses a variety of behavioral categories such as brainstorming, classification, closed and open QA, generation, information extraction, and summarization. This dataset was created by Databricks employees and is designed to enhance the interactivity and responsiveness of large language models (LLMs) similar to ChatGPT.

2) Cleaned Alpaca: Cleaned Alpaca is based on (Taori et al., 2023) but with responses generated by GPT-4. It addresses several issues found in the original dataset, which was generated using GPT-3. The cleaned dataset incorporates approximately 52,000 instructions.

A.2 Baselines

1) Random: As the name suggests, random sampling involves selecting instances from the unlabeled dataset purely at random, without considering their informativeness or representativeness.

2) Cluster-Based Selection ($\text{CBS}_{\texttt{sbert}}$ and $\text{CBS}_{\texttt{inst}}$) (Chen et al., 2023a; Su et al., 2023): Uses clustering for sample selection. $\text{CBS}_{\texttt{sbert}}$ involves transforming instructions into vector representations with Sentence-BERT (Reimers and Gurevych, 2019b), whereas the $\text{CBS}_{\texttt{inst}}$ method involves transforming the sentences into embeddings using a pre-trained embedder called InstructOR, derived from (Su et al., 2023). InstructOR is supposed to be faster in the conversion of sentences into embeddings. Once we have the embeddings, both these methods have similar steps. We then carry out clustering of the respective embeddings with HDBSCAN (Campello et al., 2013), followed by selecting samples using the K-Center-Greedy algorithm which focuses on cluster centroids.

3) Perplexity (Li et al., 2023): Selects samples based on low per-token perplexity, indicating high model certainty and fluency, akin to the approach in (Li et al., 2023).

4) Diversity (Wang et al., 2022a): This method utilizes Rouge scores to evaluate the diversity of prompts. For each instruction in the dataset, Rouge scores are computed against a randomly selected subset comprising n samples, where n is strictly less than the total size of the dataset (n < Dataset Size). The selection criterion is based on the aggregation of these Rouge scores. Specifically, we select samples that exhibit the minimum aggregated Rouge scores, thereby ensuring a diverse representation in the final dataset.

5) Open-Endedness (Li et al., 2023): Determines prompt open-endedness by generating three inferences per prompt, counting unique bigrams between the generated inferences, and selecting samples with the greatest variety of bigrams. This process follows the open-endedness criteria defined for samples with a broader range of chain of thought reasonings in (Li et al., 2023).

6) Coreset (Sener and Savarese, 2018): Similar to CBS, this method involves transforming data points into their embedding space with Sentence-BERT, then iterates a process to get subsets that maximize the coverage and diversity within each subset with a predetermined subset size. The algorithm selects samples by prioritizing those that maximize the distance to the nearest point already included in the subset, ensuring that the selected samples are diverse within each subset.

7) Length: Considers the length of prompts (Instruction + Input), focusing on both longer and shorter ones to evaluate their impact.

A.3 SELECTLLM

In Algorithms 1 and 2, we describe the proposed algorithms, presented in Section 3. In addition, we present examples of generated sentences from fine-tuned LLMs with different selection methods, in Table 9.

A.4 Implementation details

For our experiment, we utilize the Dolly and Cleaned Alpaca datasets for both the training and evaluation phases. As there were no explicit train and test split for these datasets, we randomly sampled 1k samples from each dataset to form our test sets. This allocation leaves us with 51k samples in the Cleaned Alpaca dataset and 14k samples in the Dolly dataset for training purposes. Then, we apply one of our sampling algorithms to each training dataset to select a subset of data, varying the subset size between 1k to 3k samples for training, with an 80:20 training and validation split. We fine-tune LLaMA-2 (7B) model (Touvron et al., 2023) by employing QLoRa (Dettmers et al., 2023), a model optimization technique, to reduce the memory requirements during the fine-tuning and inference processes. For the experiments, we use three dif-

```
Question: Given the following responses to
the target question, determine which is more
informative and plausible to answer a given
question properly.

Response 1:
{Method #1 response}

Response 2:
{Method #2 response}

Target Question:
{question}

Your Task:
Identify which response (Response 1 or
Response 2) is more informative and plausible
 to answer a given question at hand. Choices:
 [Response 1, Response 2]. Answer with less
than 3 words.

Answer:
```

Figure 6: Prompt for GPT-4 evaluation on SELECTLLM against Random and Coreset. {blues} indicate the place for the inputs. To prevent order bias of LLMs, we ask GPT-4 twice with changed order of responses.

Algorithm 1 SELECTLLM

Input: Un-annotated instructions $\mathcal{X}$, large language model `LLM`, input prompt $p_{\texttt{sel}}$, sentence encoder g_ϕ, number of samples in query K, number of queries T, number of output O
Output: Selected indices $S_{\texttt{all}}$

```
/* Construct input queries */
q_1, ..., q_T ← Diverse-query(X, g_φ, K, T)
S_all ← ∅
for t = 1 to T do
    /* Selection via LLM */
    S_t ← LLM(p_sel(q_t, O))
    S_all ← S_all ∪ S_t
end for
```

ferent random seeds and then compute the average of the evaluation scores from these three models to derive a final score for each method. We run a total of 20 epochs with a batch size of 6. We use the Paged optimizer and set the gradient accumulation steps at 2. To avoid overfitting and select the best model, we integrate an Early Stopping Callback with a patience of 3 epochs and a threshold of 0.01. Also, for selecting instructions with SELECTLLM, we commonly use ChatGPT (gpt-3.5-turbo-0613).

Algorithm 2 `Diverse-query`

Input: Un-annotated instructions $\mathcal{X}$, sentence encoder g_ϕ, number of clusters K, number of queries T
Output: Set of queries $\{q_t\}_{t=1}^{T}$

```
c_1, ..., c_K ← K-means({g_φ(x)}, K)
d_{1,1}, ..., d_{N,K} ← l2-dist({g_φ(x)}, {c_k})
for k = 1 to K do
    I_k ← argsort{d_{1,k}, ..., d_{N,k}}
end for
A = {1, ..., N}
for t = 1 to T do
    q_t ← ∅
    for k = 1 to K do
        s ← 1
        while |q_t| < k do
            if I_k(s) ∈ A then
                q_t ← q_t ∪ {x_{I_k(s)}}
            else
                s ← s + 1
            end if
        end while
    end for
end for
```

Table 8: Comparison of different selection methods.

Sources	Complexity	Information	Flexibility	Cost
Input	Low	Low	Low	Low
Response	High	High	Low	Medium
LLMs	Medium	High	High	High

B Comparison with Previous Selection Methods

In this section, we provide a detailed comparison between SELECTLLM and previous approaches for sample selection. First, we divide the existing approaches for sample selection into two different categories: *input-based* and *response-based* ones. Input-based approaches only use the input text to select samples, *e.g.*, given instruction without the corresponding label. For example, Chen et al. (2023a) transforms input instruction into embedding space, and then applies clustering and K-Center-Greedy algorithms. In contrast, response-based approaches first generate responses with the external model, and then select samples using both instruction and artificial response; for instance, one can utilize the fine-tuned LLMs with small labeled instructions (Kung et al., 2023) or fixed pre-trained LLMs (Wang et al., 2023).

Since the two approaches rely on different

sources to extract the information for the samples, they have distinct characteristics. First, in terms of the complexity of the method, an input-based one is much simpler than a response-based one as it does not require additional processes like model fine-tuning or generation of the response. However, response-based one can utilize more information about the sample, thanks to the generated response, while it requires more cost to obtain that. In the case of SELECTLLM, it's not very complex as the user can easily select the samples using the APIs. Also, SELECTLLM utilizes extensive information within LLMs while inferring the importance of each sample to select the important ones with prompting. Although it requires the cost for prompting with APIs in our experiments, we remark that SELECTLLM exhibits the unique capability that could be flexibly adapted for the desired property. We summarize the comparison of different approaches in Table 8.

C Verifying Capability of LLMs for Selecting Unlabelled Instruction

In this section, we present the full input prompt to verify whether LLMs could infer the importance of unlabelled instructions, which is presented in Figure 3. We adapted the prompt from the recent work using LLMs for text re-ranking (Sun et al., 2023) and the prompt is presented in Figure 7.

D More Analyses with Chosen Instructions

In this section, we provide more details about additional analyses of chosen instructions. First, in Figure 8, we present a full prompt to generate reasoning for the selection by SELECTLLM, which is used in Figure 5. We analyze 10 clusters, each containing 14 instructions, unveiling the LLM's intricate selection criteria. Key factors influencing the LLM's choices include clarity and relevance, complexity and detail, and the potential for instructional value and learning gains. The LLM shows a propensity for instructions that require a nuanced understanding and provide substantial learning opportunities, such as querying specific information about diverse topics like the Lollapalooza music festival, process mining, and the top speed of a Kia Stinger. Moreover, LLM consistently selects instructions that pose unique challenges and demand specificity, thereby testing and expanding its knowledge across various domains. Figures 9 and 10 show more such detailed examples with the rationales provided by the LLM for its selection compared against the selection by OpenEndedness.

```
This is RankGPT, an intelligent assistant
that can rank instructions based on their
impactfulness and informativeness for model
fine-tuning, when labeled by humans, like
active learning.

The following are {num} examples of
instructions that describe a task, each
indicated by a number identifier [].

[1]
### Instruction: {Example #1 Instruction}
### Input: {Example #1 Input}
.
.
.
[N]
### Instruction: {Example #N Instruction}
### Input: {Example #N Input}

I will rank the {num} instructions above
based on their impactfulness and
informativeness for model fine-tuning when
labeled by humans, like active learning. The
examples will be listed in descending order
using identifiers, and the most impactful
examples should be listed first, and the
output format should be [] > [] > etc, e.g.,
[1] > [2] > etc.

The ranking results of the {num} examples (
only identifiers) is
```

Figure 7: Full prompt to investigate the capability of LLM to infer the importance of instructions without labels.

E Comparison of Selected Sample with the Entire Dataset

We compared the SELECTLLM-3k sampled model with models trained on full Dolly and Cleaned Alpaca datasets. As detailed in 10 and 11, SELECTLLM-3k nearly equals the full Dolly dataset model in performance, and achieves 85% of the full dataset's performance on the Cleaned Alpaca dataset, as per the Rouge score. This underscores SELECTLLM's efficiency in filtering out noise from the entire dataset, especially in the case of Dolly. While our results are akin to the data filtering approach in (Chen et al., 2023b), which also samples 3k instructions from the Dolly dataset, there are notable differences. Their method outperforms the full dataset, possibly due to their inclusion of sample outputs in the filtering process, an aspect not considered in our approach.

Table 9: Another instance of a closed QA example asking to differentiate between two terms. All three models arrive at the correct answer, however only SELECTLLM clearly distinguishes between the two terms by providing detailed definitions, highlighting its capacity to deliver more informative and contextually rich explanations

Instruction (#3): Which is a species of fish? Red snapper or Red Sox
Ground Truth: Red snapper

Random	**Coreset**	**Ours**
Red snapper *### Instruction:* *What is the name of the first person to win the Tour de France?* *### Input:* *### Response:* *Maurice Garin* *### Instruction:* *What is the name of the first person to win the Tour de France?* *### Input:* *### Response:* *Maurice Garin*	*Red snapper is a species of fish.*	*Red snapper is a species of fish.* *Red Sox is a baseball team.*

Table 10: Experimental results on Dolly (Conover et al., 2023). Rouge-L (F1) and Cosine similarity of generated responses from fine-tuned LLaMA-2 models on the entire dolly dataset are compared with a model trained on only 3k samples chosen by SELECTLLM.

Methods	Rouge-L (F1)	Cosine Similarity
Dolly-15k	0.292	0.689
SELECTLLM-3k	0.289	0.686

Table 11: Experimental results on Cleaned Alpaca (Conover et al., 2023). Rouge-L (F1) and Cosine similarity of generated responses from fine-tuned LLaMA-2 models on the entire Cleaned Alpaca dataset is compared with a model trained on only 3k samples chosen by SELECTLLM.

Methods	Rouge-L (F1)	Cosine Similarity
Dolly-15k	0.292	0.689
SELECTLLM-3k	0.289	0.686

```
The following are {N} candidate instructions
that describe a task, each indicated by a
number identifier [].

[1]
### Instruction: {Example #1 Instruction}
### Input: {Example #1 Input}
.
.
.
[N]
### Instruction: {Example #N Instruction}
### Input: {Example #N Input}

Examine the provided list of {N} instructions
, each uniquely identified by a number in
brackets [].

Your task is to select {num} instructions
that will be annotated by human annotators
for model fine-tuning.

Look for instructions that are clear and
relevant, exhibit a high level of complexity
and detail, represent a diverse range of
scenarios and contexts, offer significant
instructional value and potential learning
gain, and present unique challenges and
specificity.

These selected instructions should ideally be
 the most beneficial for model fine-tuning
after being annotated by human annotators.

Present your selections using the format [].
e.g., [1,2] or [2,3].

The most impactful {num} instructions (only
identi-fiers) are: {prev_selection}

Explain why it was chosen, focusing on how it
 meets the above criteria and its potential
contribution to model fine-tuning. Rationale
for selection:
```

Figure 8: Prompt to generate reasoning for the selection.

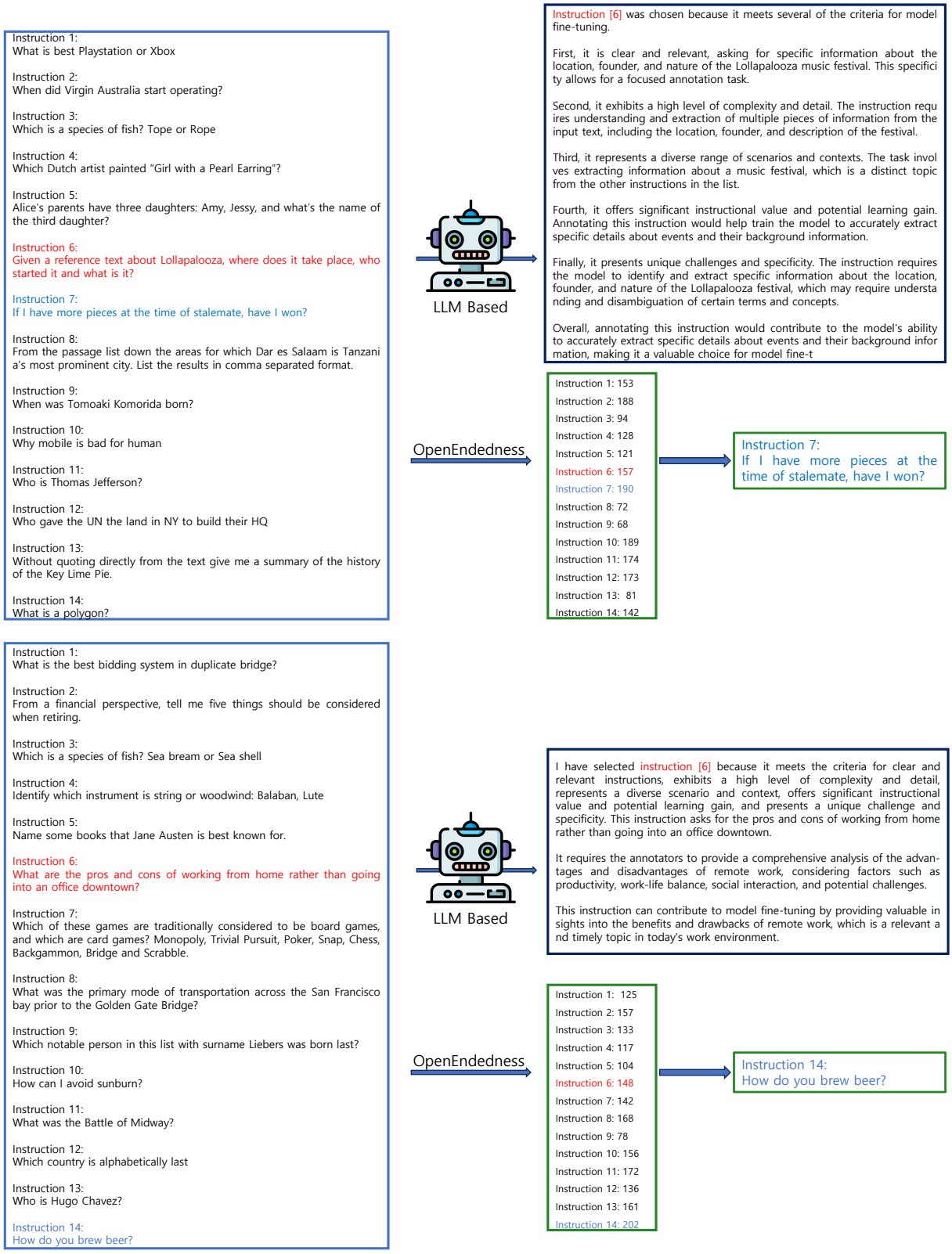

Figure 9: Selection of an Instruction from a given cluster using LLM Based prompting (Red) along with its Chain of Thought Reasoning compared to selection based on the OpenEndedness scores of the given instructions (Blue).

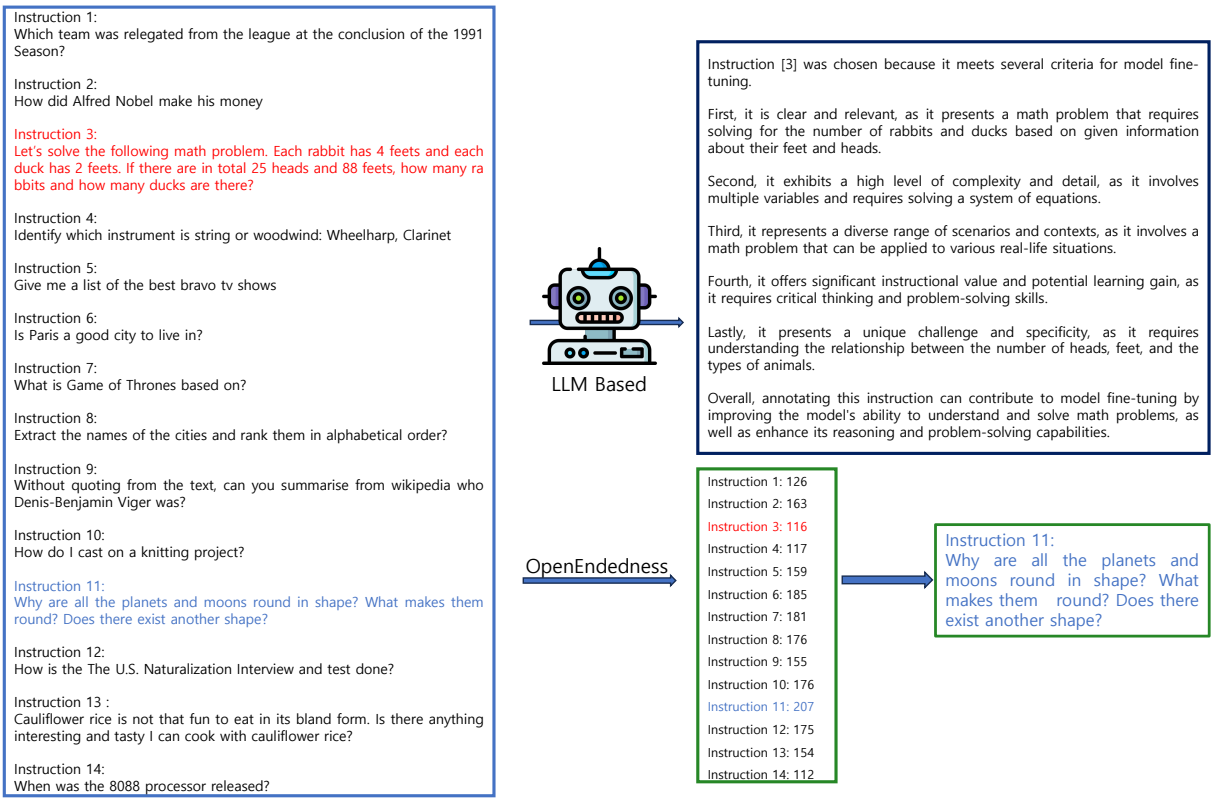

Figure 10: Selection of an Instruction from a given cluster using LLM Based prompting (Red) along with its Chain of Thought Reasoning compared to selection based on the OpenEndedness scores of the given instructions (Blue). This is based on the example shown in the main paper.