Selective Reflection-Tuning:
Student-Selected Data Recycling for LLM Instruction-Tuning

Ming Li[1], Lichang Chen[1], Jiuhai Chen[1], Shwai He[1], Jiuxiang Gu[2], Tianyi Zhou[1]

[1]University of Maryland, College Park   [2]Adobe Research

{minglii, bobchen, tianyi}@umd.edu

Project: https://github.com/tianyi-lab/Reflection_Tuning

Abstract

Instruction tuning is critical to large language models (LLMs) for achieving better instruction following and task adaptation capabilities but its success heavily relies on the training data quality. Many recent methods focus on improving the data quality but often overlook the compatibility of the data with the student model being finetuned. This paper introduces Selective Reflection-Tuning, a novel paradigm that synergizes a teacher LLM's reflection and introspection for improving existing data quality with the data selection capability of the student LLM, to automatically refine existing instruction-tuning data. This teacher-student collaboration produces high-quality and student-compatible instruction-response pairs, resulting in sample-efficient instruction tuning and LLMs of superior performance. Selective Reflection-Tuning is a data augmentation and synthesis that generally improves LLM finetuning and self-improvement without collecting brand-new data. We apply our method to Alpaca and WizardLM data and achieve much stronger and top-tier 7B and 13B LLMs.

1 Introduction

The quality of instruction tuning (Wei et al., 2022; Chen et al., 2023a; Mishra et al., 2022; Chung et al., 2022; Zhang et al., 2023; Liu et al., 2023a) data is paramount to the LLM being fine-tuned, i.e., the student model. There is a growing trend and demand for the community to automatically improve the quality of instruction tuning data. Previous works either curate datasets by human experts (Conover et al., 2023; Longpre et al., 2023; Zhou et al., 2023) or distill the responses of well-trained LLMs (Taori et al., 2023; Peng et al., 2023; Chiang et al., 2023; Vu et al., 2023; Xu et al., 2023a; Li et al., 2023b, 2024a; Xu et al., 2024). The self-improvement (Bai et al., 2022b; Huang et al., 2023; Pan et al., 2023) ability of LLMs has also been explored to improve the instruction or response of a training sample.

However, these existing methods of data enhancement (Huang et al., 2023; Ye et al., 2023; Li et al., 2023b; Mitra et al., 2023) do not take a critical criterion into account: **Is the teacher-refined data compatible to the needs of the student model?** These approaches typically do not account for the inherent randomness and potential degradation associated with the generative models' output, leading to an oversight in how the student model responds to these "improved" data samples. Thus a mechanism for the student model to selectively integrate these enhancements has been notably absent. To bridge this gap, our work introduces an **teacher-student collaboration pipeline** wherein a teacher generative model engages in a reflection process to enhance both the instruction and response of a data sample. The student model then evaluates whether to incorporate these improvements based on its unique statistical attributes. This pipeline is versatile and can be adapted to various contexts where data enhancement is needed.

Then, another pivotal question arises: **How does the student model decide which enhanced data are needed and critical to its training?** This question underpins the challenge of autonomously evaluating the quality of instructions and responses. Common practices involve utilizing sophisticated models like GPT-4 for assessment purposes (Zheng et al., 2023; Li et al., 2023e; Liu et al., 2023c; Chiang and Lee, 2023) or employing a secondary judge model equipped with evaluative capabilities (Wang et al., 2023c; Li et al., 2023a). These methods, however, present limitations: they fail to address the discrepancies between the evaluating model and the actual student model undergoing training. Particularly in the latter approach, even though the judge model and the student model might share the same structural framework, their weight distributions diverge once endowed with the evaluative functions.

Consequently, the preferences of the judge model may not align with the real student model's requirements. To circumvent these issues, we adopt a statistical method, utilizing the Instruction-Following Difficulty (IFD) score proposed by Li et al. (2023c, 2024b). This score is derived directly from the raw student model, thereby mitigating potential domain shifts and ensuring that the evaluation is better aligned with the student model's learning context.

In our approach, the IFD score serves as a crucial metric that measures how much help the instruction can provide to the likelihood of the response if added as an extra condition, representing the **Difficulty** of the sample. However, though effective, the IFD score mainly assesses the instructions. Motivated by Humpback (Li et al., 2023d) which requires LLMs to generate potential instruction based on responses, we further introduce a reversed version of IFD named reversed-IFD (r-IFD). This metric evaluates how much the response contributes to predicting the corresponding instruction. A lower r-IFD score suggests the student can easily deduce the corresponding instruction given the response, indicating this sample is feasible for the student to learn, representing the **Feasibility** of the sample. This dual approach, employing both **IFD scores for Difficulty** and **r-IFD scores for Feasibility**, enables a comprehensive and nuanced assessment of the instruction-tuning process, ensuring the refined data aligns well with the student model's capabilities and objectives.

We name our overall method Selective Reflection-Tuning, which contains the selective instruction reflection phase and the selective response reflection phase. In the first phase, a teacher model is utilized to reflect on the instruction of the given sample based on some criteria and generate a new sample. Then the student model makes the decision of whether to accept the improvement based on difficulty (IFD). In the second phase, the teacher model reflects and generates a sample with a new response and the student model decides whether to accept based on feasibility (r-IFD). With our interactive pipeline, we obtain a dataset with supreme quality, with only instruction tuning on a relatively small amount of data, our model outperforms most existing open-source models with even larger model sizes. Our contributions include:

- We propose a teacher-student collaboration pipeline where the teacher model and student model cooperate to build a more coherent and model-compatible instruction tuning dataset, which can be further adapted into other self-improvement scenarios.
- We present a nuanced evaluation schema reversed-IFD, quantifying the relevance of instruction-response pairs, and representing the feasibility of the sample for the student.
- With only instruction tuning on a few thousand of automatically generated data, our models achieve top-tier performances, indicating the supreme quality of our data.

2 Preliminaries

Let f_θ denote the pre-trained student model, e.g., LLaMA, with parameters θ and g the teacher model, e.g., ChatGPT. Let lowercase letters $x, y, z, c, ..$ denote the text segments, which could be phrases or sentences, and each token in x is denoted as $x[i]$. We use uppercase letters $D, ..$ to denote the collection of language sequences or datasets, and D_0 represents the initial base dataset. Since both f_θ and g are in auto-regressive manners, a sequence $x = (x[1], ..., x[n])$ can be further denoted as:

$$f_\theta(x) = \prod_{i=1}^{n} f(x[i]|x[1, ..., i-1]) \quad (1)$$

In the instruction tuning setting, there will be a mapping function that turns the original raw instruction x into the desirable format and requests models for a response y. For simplicity, we directly notate this process as $y \sim f(y|x)$. And the loss function for instruction-tuning can be denoted as:

$$L_\theta(y|x) = -\frac{1}{n} \sum_{i=1}^{n} \log f_\theta(y|x) \quad (2)$$

where n is the length of response y.

Motivated by Cherry LLM (Li et al., 2023c) which proposes the IFD score to measure the difficulty of instruction in the given instruction-response pairs. We utilize the perplexity of the IFD score (Li et al., 2024b), which is formulated as:

$$\text{IFD}_\theta(y|x) = \frac{\text{ppl}(y|x)}{\text{ppl}(y)} = \exp(L_\theta(y|x) - L_\theta(y)) \quad (3)$$

where $\text{ppl}(y|x)$ represents the perplexity of model f_θ to fit the response y given the instruction x as the context, and $\text{ppl}(y)$ represents the perplexity
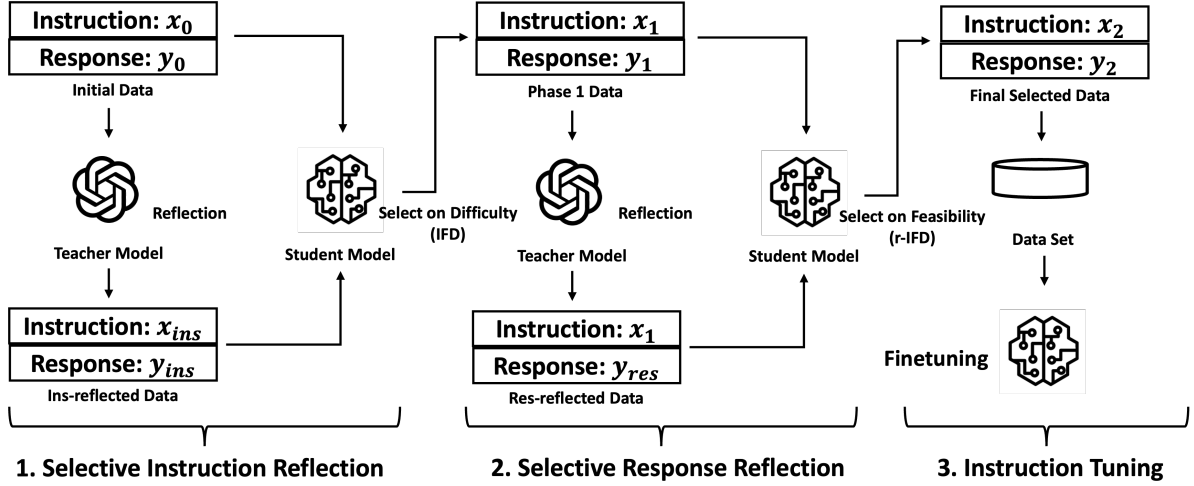
Figure 1: The overall pipeline of our method. The first Selective Instruction Reflection phase aims to obtain a better instruction for a data sample and the second Selective Response Reflection phase aims to obtain a better response for the sample. The reflection process is conducted by the well-trained teacher model and the selection process is conducted by the student model.

of model f_θ to directly fit the response y without any context given. This value represents how the given instruction x affects the generation of corresponding response y for given model f_θ, which has been shown as an effective metric for evaluating the given instruction-following data pairs (Li et al., 2024b). A higher IFD score indicates that the instruction is more challenging for the student model to generate the response, suggesting the instruction's difficulty for the student model.

3 Methodology

As shown in Figure 1, there are two main phases in our method, Selective Instruction Reflection and Selective Response Reflection phase. In each phase, the teacher model generates the updated version of instructions or responses based on some given specific criteria $\{c_{ins,1}, ..., c_{ins,k}\}$ [1], then the student model judges if the updates are beneficial to it based on difficulty (IFD) or feasibility (reverse-IFD). Finally, these selectively improved samples can be used for the final instruction tuning.

3.1 Selective Reflection on Instruction

Reflection on Instruction

Given the instruction-response pair (x_0, y_0) from the original dataset D_0 with some specific criteria $\{c_{ins,1}, ..., c_{ins,k}\}$, the teacher model g is required to reflect on this sample and generate a better instruction-response pair (x_{ins}, y_{ins}) according to its reflection. With the criteria given, the teacher

---
[1] Prompt for reflection can be found in Appendix B

model g is able to generate critical responses:

$$[z_{ins,1}, ...] \sim g(z, ...|x_0, y_0, c_{ins,1}, ...) \quad (4)$$

where both original instruction and response are wrapped into the prompt rather than original instruction alone. These critical responses further serve as the guidance (chain of thought) (Wei et al., 2023; Yao et al., 2023) for the generation of the new instruction and response pair:

$$[x_{ins}, y_{ins}] \sim g(x, y|x_0, y_0, c_{ins,1}, ..., z_{ins,1}, ...) \quad (5)$$

where the above process is sampled as a continuous language sequence, and the critical responses would not be decomposed from the whole output.

Selection on Instruction

Though the given sample pair is updated by the teacher model, it remains uncertain whether this updated version is truly better for the student model. While most existing work evaluates the quality of a data sample by directly prompting existing generative models, they inevitably suffer from the misalignment problem. Thus we utilize the IFD score (Li et al., 2023c) calculated based on the specific base student model, which measures how the instruction benefits the generation of corresponding responses for the model, representing the difficulty of the sample.

After obtaining the updated instruction-response pair, the base model f_θ is required to compare the IFD score of the original pair (x_0, y_0) and updated pair (x_{ins}, y_{ins}) and the sample with higher IFD

scores will be chosen:

$$(x_1, y_1) = \underset{(x,y)}{\operatorname{argmax}}(\text{IFD}_\theta(y|x)) \qquad (6)$$

where $(x, y) \in \{(x_0, y_0), (x_{ins}, y_{ins})\}$. Then the chosen data pair (x_1, y_1) with a higher IFD score will be sent to the next phase.

3.2 Selective Reflection on Response

Reflection on Response

After the first phase, although the instruction x_1 is guaranteed to be difficult for the student model, the corresponding response y_1 is still sub-optimal. Thus another reflection on the response process is further proposed. Similar to the above procedure, a new set of criteria for reflection on response is defined as $\{c_{res,1}, ..., c_{res,m}\}$. The overall process can be noted as:

$$y_{res} \sim g(y|x_1, y_1, c_{res,1}, ..., c_{res,m}, z_{res,1}, ..., z_{res,m}) \qquad (7)$$

where $z_{res,i}$ represents the critical response of ith response criteria $c_{res,i}$. In the process, the instruction and response pair (x_1, y_{res}) is fully improved.

Selection on Response

Our pipeline aims to improve both the instruction and response in an instruction-tuning sample. IFD score measures the difficulty of the sample. We take a step further by adding another dimension which we call reversed IFD (r-IFD) representing the feasibility for the student to generate the instruction given the response. A lower r-IFD score suggests the student can easily deduce the corresponding instruction given the response, indicating this sample is feasible for the student to learn, which measures the model-specific matching degree of the existing data pair. [2]

The high-level idea of r-IFD is in line with the success of Humpback (Li et al., 2023d), which utilizes LLM to predict the corresponding instruction from given texts (responses), and hypothesizes that "we can predict instructions for these candidate gold answers that can be used as high-quality example pairs". In our paper, we further hypothesize that a response is more informative for training if it is feasible for the LLM to predict the corresponding instruction from the response. This hypothesis is naturally proved by the Humpback, which generates instructions that can be handled by LLMs, while those difficult ones are naturally discarded.

Under this circumstance, the reversed IFD score should be small since the smaller value represents that it is easier for the model to generate the corresponding instruction given the response. Specifically, the r-IFD score is calculated as:

$$\text{r-IFD}_\theta(x|y) = \frac{\text{ppl}(x|y')}{\text{ppl}(x)} = \exp(L_\theta(x|y') - L_\theta(x)) \qquad (8)$$

where y' represents the text segment generated by mapping the original y into a query to guess the corresponding potential instructions.

For the given original sample pair (x_1, y_1) from the first phase and reflected sample pair (x_1, y_{res}), the selection process can be formulated as:

$$(x_2, y_2) = \underset{(x,y)}{\operatorname{argmin}}(\text{r-IFD}_\theta(x|y)) \qquad (9)$$

where $(x, y) \in \{(x_1, y_1), (x_1, y_{res})\}$.

After the above phases, there will be a corresponding data pair (x_2, y_2) for each original (x_0, y_0), which is represented as our selective reflected data. Then we discard all the samples which is not response-reflected for the consistency of response distribution. We name the whole above process as a selective recycling process, which greatly improves the quality of the previous dataset [3]. The student model f_θ will be trained on the newly generated data and the new models are notated as "sRecycled Models", eg. sRecycled Alpaca.

4 Experimental Setup

4.1 Base Datasets

The Alpaca dataset (Taori et al., 2023), sourced from Stanford University, offers $52,002$ instruction samples. Developed via the self-instruct paradigm (Wang et al., 2023d), it leveraged the capabilities of the text-davinci-003 model. The WizardLM dataset (Xu et al., 2023a) is a refined collection encompassing a total of $250,000$ instruction samples. To enhance data fidelity, gpt-3.5-turbo-0613 has been meticulously integrated during the refinement process. From this extensive dataset, we predominantly focused on the WizardLM-7b subset, comprising $70,000$ samples. We test our method on both of these two datasets to verify the effectiveness of our method and name the corresponding models as "sRecycled Alpaca" and "sRecycled WizardLM".

---

[2]Two examples with low or high r-IFD scores can be found in Appendix H for better illustration.

[3]Some statistic analysis can be found in Appendix E

| | Alpaca Eval Leaderboard | | | | | | |
	Win Rate	Standard Error	Wins	Draws	Avg Length	Data	RLHF/AIF
GPT4 (OpenAI, 2023)	95.28	0.72	761	12	1365	/	/
Claude 2	91.36	0.99	734	1	1069	/	/
Zephyr 7B Beta (Tunstall et al., 2023)	90.60	1.03	727	1	1444	774,000	✓
ChatGPT	89.37	1.08	716	5	827	/	/
Evo v2 7B	89.35	1.08	715	5	1754	/	/
XwinLM 7b V0.1 (Team, 2023)	87.83	1.15	703	1	1894	/	✓
sRecycled WizardLM 13B (ours)	**85.96**	**1.23**	**692**	**0**	**1523**	**46,064**	✗
Zephyr 7B Alpha (Tunstall et al., 2023)	85.76	1.23	688	3	1302	774,000	✓
OpenChat V2 13B (Wang et al., 2023a)	84.97	1.26	683	2	1564	82,600	✗
Humpback LLaMa 65B (Li et al., 2023d)	83.71	1.31	672	2	1269	502,133	✗
UltraLM 13B V2.0 (Ding et al., 2023)	80.64	1.31	673	0	1399	774,000	✗
sRecycled WizardLM 7B (ours)	**83.48**	**1.31**	**672**	**0**	**1583**	**46,325**	✗
Vicuna 13B v1.3 (Chiang et al., 2023)	82.11	1.35	660	2	1132	125,000	✗
GPT-3.5	81.71	1.33	642	25	1018	/	/
LLaMA2 Chat 13B (Touvron et al., 2023)	81.09	1.38	652	0	1513	27,750	✓
UltraLM 13B (Ding et al., 2023)	80.64	1.40	647	1	1087	774,000	✗
sRecycled Alpaca 7B (ours)	**79.58**	**1.42**	**639**	**0**	**1353**	**37,114**	✗
Claude2 Alpaca 13B (Chen et al., 2023b)	78.93	1.44	633	0	1127	52,002	✗
Recycled WizardLM 7B	78.88	1.44	635	0	1494	70,000	✗
Recycled Alpaca 7B	76.99	1.49	619	0	1397	52,002	✗
Vicuna 7B v1.3 (Chiang et al., 2023)	76.84	1.49	614	3	1110	125,000	✗
WizardLM 13B (Xu et al., 2023a)	75.31	1.51	601	9	985	250,000	✗
Guanaco 65B (Dettmers et al., 2023)	71.80	1.59	578	0	1249	9,850	✗
LLaMA2 Chat 7B (Touvron et al., 2023)	71.37	1.59	574	1	1479	27,750	✓
Vicuna 7B (Chiang et al., 2023)	64.41	1.69	517	3	1044	70,000	✗
Davinci003	50.00	0.00	0	805	307	/	/
LIMA 7B (Zhou et al., 2023)	41.29	1.74	332	0	1624	1,000	✗
Alpaca 7B (Taori et al., 2023)	26.46	1.54	205	16	396	52,002	✗

Table 1: The comparison of performance on AlpacaEval Leaderboard. "Data" represents the number of data used for fine-tuning. "RLHF/AIF" represents whether the model utilize an additional RLHF or RLAIF process.

4.2 Evaluation Metric

To evaluate the effectiveness of our method, we utilize 4 commonly used automatic evaluation metrics, including (1) **Pair-wise Comparison**, (2) **Alpaca Eval**, (3) **Open LLM Leaderboard**, and (4) **MT-Bench**. Besides, additional (5) **Human Study** is also conveyed for the evaluation. [4]

5 Experimental Results

5.1 Main Results

For **Pair-wise Comparison**, we compare our sRecycled WizardLM 7B with other classic open-source models by using GPT4 as the judge as shown in Figure 2. Notably, our model outperforms most models by a large margin, regardless of whether they are 7B or 13B, ("LLaMA2 Chat 13B", "Vicuna 13B v1.3"), or whether extra RLHF/AIF is utilized ("LLaMA2 Chat 7B", "Zephyr 7B Alpha"), or whether other data improvement methods are utilized ("Recycled Wiz 7B", "WizardLM Orca 7B"[5], "Orca 2 7B"(Mitra et al., 2023)).

---

[4]Detailed description can be found in Appendix C.
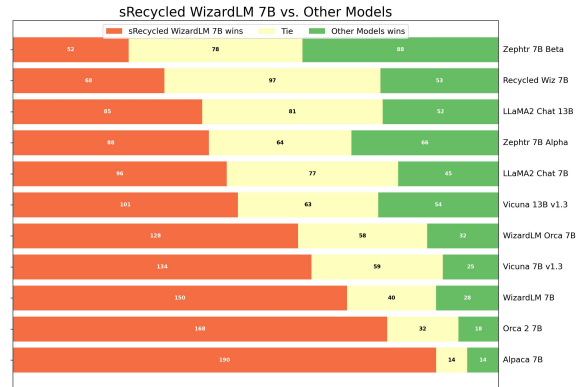[5]https://huggingface.co/datasets/pankajmathur/WizardLM_Orca

Figure 2: The pair-wise comparison between our model with other classic open-source models by using GPT4 as the judge. From the comparison, our model outperforms most of them by a large margin, regardless of their model size and whether extra RLHF/AIF is utilized.

Table 1 delineates the outcomes on the **AlpacaEval Leaderboard** in which our models stand out for delivering promising results with a streamlined approach. This comparison provides a direct quantification of a model's capacity for instruction adherence and the intrinsic quality of its output. Remark-

	Huggingface Open LLM Leaderboard						
	Average	ARC	HellaSwag	MMLU	TruthfulQA	Data	RLHF/AIF
Alpaca 7B (Taori et al., 2023)	50.21	42.65	76.91	41.73	39.55	52,002	✗
WizardLM 7B (Xu et al., 2023a)	54.18	51.60	77.70	42.70	44.70	70,000	✗
Vicuna 7B v1.3 (Chiang et al., 2023)	55.63	50.43	76.92	48.14	47.01	125,000	✗
sRecycled Alpaca 7B (ours)	56.05	54.01	78.07	46.69	45.41	37,114	✗
LLaMA2 Chat 7B (Touvron et al., 2023)	56.34	52.90	78.55	48.32	45.57	27,750	✓
sRecycled WizardLM 7B (ours)	56.79	54.78	77.86	45.63	48.91	46,325	✗
Vicuna 13B v1.1 (Chiang et al., 2023)	59.21	52.73	80.14	51.90	52.08	125,000	✗
LLaMA2 Chat 13B (Touvron et al., 2023)	59.94	59.04	81.94	54.64	44.12	27,750	✓
Vicuna 13B v1.3 (Chiang et al., 2023)	60.01	54.61	80.41	52.88	52.14	125,000	✗
sRecycled WizardLM 13B (ours)	60.22	59.73	80.15	55.64	45.37	46,064	✗
WizardLM 13B 1.0 (Xu et al., 2023a)	60.25	57.20	81.00	52.30	50.50	250,000	✗

Table 2: The comparison of performance on Huggingface Open LLM Leaderboard. "Data" represents the number of data used for fine-tuning. "RLHF/AIF" represents whether the model utilizes an additional RLHF or RLAIF process.
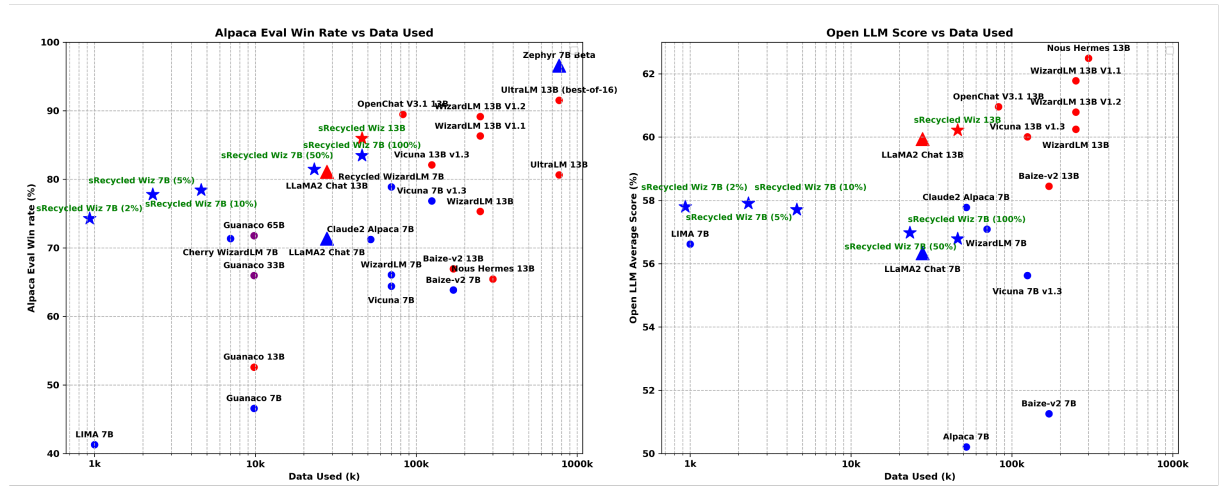

Figure 3: Comparison between model performances and data used for fine-tuning on the Alapca Eval benchmark and the open LLM leaderboard. We utilize star markers to represent our models, dot markers to represent other instruction-tuned models and triangle markers to represent RLHF/AIF models. Blue markers represent 7B models, red markers represent 13B models and purple markers represent models with larger weights.

	Huggingface Open LLM Leaderboard					AlpacaEval
	Average	ARC	HellaSwag	MMLU	TruthfulQA	AlpacaEval
sRecycled WizardLM 7B (2%) (926)	57.80	54.69	78.80	47.00	50.70	74.29
sRecycled WizardLM 7B (5%) (2,316)	57.91	54.86	79.83	46.69	50.23	77.78
sRecycled WizardLM 7B (10%) (4,632)	57.71	55.46	79.56	46.83	48.98	78.43
sRecycled WizardLM 7B (30%) (13,897)	56.89	54.61	79.25	44.67	49.05	82.48
sRecycled WizardLM 7B (100%) (46,325)	56.79	54.78	77.86	45.63	48.91	83.21

Table 3: The comparison of performance on Huggingface Open LLM Leaderboard and AlpacaEval Leaderboard by using different amounts of selective recycled WizardLM data. In the first parentheses are the percentage of data used for tuning and in the second parentheses are the specific amount of number used.

ably, with a win rate that competes closely with heavyweight counterparts, our models achieve this with only instruction tuning on a small amount of our high-quality data. Furthermore, our approach does not rely on additional processes such as RLHF (Ouyang et al., 2022; Bai et al., 2022a) or RLAIF (Bai et al., 2022b; Lee et al., 2023), which demand

a significant overhead. This reduction in complexity represents a significant advancement in model efficiency, making it a cost-effective and agile solution for real-world applications. The ingenuity of our model lies in its simplicity and effectiveness, proving that with intelligent design less is more.

Table 2 showcases the performance comparison

on the **Huggingface Open LLM Leaderboard** with some related models. Similarly, with only instruction tuning on a small amount of data, our models surpass plenty of the models on the average performances across representative benchmarks. These benchmarks do not directly measure the instruction-following ability or the quality of responses generated by LLMs, but a relatively higher performance on these benchmarks still shows the non-degradation quality of our method.

For the **human evaluation**, we compare the responses to given testing instructions between our sRecycled WizardLM 7B model with the original WizardLM 7B model by human evaluators, there are $57/108$ wins for our model, $23/108$ ties, and $28/108$ losses. These results further prove the efficacy of our method in improving the quality of the original data.

5.2 Fewer Data Scenario

To better illustrate the supreme quality of our sRecycled dataset, we further conduct experiments where only part of the data samples are utilized. Following Li et al. (2023c), we calculate the IFD score of each data sample and select the top k-percent of the data for the instruction tuning. Their performances on the Open LLM Leaderboard and the Alpaca Eval Leaderboard are shown in Table 3 [6]. Since selecting data by IFD score is an effective method to find a better instruction tuning subset from the overall data set, this consistent decrease in performance on Alpaca Eval indicates the difficulty in finding a subset with higher performances, which further verifies the overall high quality of our selective recycled data.

Figure 3 draws the scatters comparing the data used and corresponding performance. It illustrates a striking balance of efficiency and performance achieved by our models. Despite using markedly less data, our models—represented by the distinctive star markers—consistently occupy the upper echelons of the performance spectrum on both the Alpaca Eval benchmark and the open LLM leaderboard. Furthermore, the plots reveal that our models achieve these results without scaling up to the larger data requirements that other models seem to necessitate, as indicated by their position further to the right along the x-axis. The results not only signal superior data quality but also suggest a potential reduction in the computational resources

---

[6]Detailed table and ablation can be found in Appendix F

and time required for training, which is crucial for sustainable and scalable AI development.

Furthermore, it is astonishing that with less than $1,000$ selective recycled data, our "sRecycled WizardLM 7B (2%) (926)" outperforms most existing 7B models, including LIMA, which is trained with manually curated data samples. This not only verifies LIMA's (Zhou et al., 2023) hypothesis but also pushes it further forward: In addition to human-carefully-crafted instruction tuning data, less than $1,000$ totally automatically generated data can also yield substantial benefits in model alignment and performance.

6 Ablation Study

6.1 Ablation on Reflection

Extensive experiments are conducted on several 7B models as shown in Table 4. We utilize the pair-wise comparison with GPT4 as the judge to measure the performance of different models.

Compared with the original WizardLM model, our performance is dramatically better, which directly showcases the supreme capability of our method to increase the data quality. "Reflect on Ins." and "Reflect on Res." represent models that are trained with data reflected merely on instruction or response and no selection process is utilized. Through these comparisons, it can be found that reflection on instruction only improves the data quality a little, while reflection on response improves the data quality more. This phenomenon is reasonable due to the similarity in response distribution between original WizardLM data and WizardLM data reflected on instruction. On the contrary, when the response is reflected, it directly affects the target that LLM needs to fit on, thus directly showing an improvement in the response quality. "Reflect on Ins. + Res." represents the model trained by using reflection-tuning ("Recycled WizardLM 7B") without the selection process, though already having the good capability to follow instructions, our model still outperforms it with less data.

6.2 Ablation on Selection

Moreover, to further verify the effectiveness of our selection mechanism, experiments with different selection methods are conducted shown in Table 4.

"Select by Randomness" represents the student model randomly choosing whether to accept improved data. Not only does this model underperform our final model largely, but it also underper-

	Win	Tie	Lose	Win Rate
vs. Original WizardLM	150	40	28	1.560
vs. Reflect on Ins.	143	51	24	1.546
vs. Reflect on Res.	72	93	53	1.087
vs. Reflect on Ins. + Res.	68	97	53	1.069
vs. Select by Randomness	81	94	43	1.174
vs. Select by Coherence	75	96	47	1.128
vs. Select by Perplexity	64	99	55	1.041
vs. Select by IFD only	58	107	53	1.023
vs. Select by r-IFD only	74	96	48	1.119

Table 4: The pair-wise comparison between our sRecycled WizardLM 7B with other models. The "win", "Tie" and "Lose" represent the number of wins or losses of sRecycled WizardLM 7B. The win rate is calculated as (Num(Win) − Num(Lose)) / Num(All) +1.

forms both "Reflect on Res." and "Reflect on Ins. + Res.". This baseline result indicates that without a proper selection method, the blind mixture of data might harm the model's performance.

"Select by Coherence" represents the data selected based on the coherence between instruction and response, which is calculated by cosine similarity of the Sentence-BERT (Reimers and Gurevych, 2019) embeddings. In this setting, the data pairs, whose instruction and response are more related, are more likely to be selected. The performance of this model is slightly better than the random selection model, and still worse than both "Reflect on Res." and "Reflect on Ins. + Res.", indicating the ineffectiveness of this selection method.

"Select by Perplexity" represents the student model choosing whether to accept the improved data by whether the perplexity is improved, which is the closest to ours. The performance of this model surpasses both "Reflect on Res." and "Reflect on Ins. + Res.", showing that a selection process can definitely further improve the model's performance, verifying our motivation for adding the selection mechanism. However, this model still underperforms our model, indicating the efficacy of our selection strategy.

"Select by IFD only" and "Select by r-IFD only" represent situations where we only utilize IFD or r-IFD scores for student side selection. Utilizing only IFD results in a model that is close to our main model, indicating the usefulness of the IFD score. However, its performance is still lower, indicating the effect of the r-IFD.

7   Comparison with Related Work

Earlier works on instruction tuning focus on creating large, high-quality datasets curated by human experts (Khashabi et al., 2020; Ye et al., 2021; Wei et al., 2022; Wang et al., 2022; Du et al., 2022), time-consuming and labor-intensive. Thus a number of works try to construct instruction-tuning datasets automatically. Self-Instruct (Wang et al., 2023d) utilizes the in-context learning capability of GPT-3 to expand tasks to many diverse instruction-response pairs. WizardLM (Xu et al., 2023a) applies an evolution methodology to refine and diversify the original instruction data. LaMini-LM (Wu et al., 2024) introduces to generate Top-Fuided instructions based on Wiki data. Peng et al. (2023) utilize GPT4 to generate responses for existing datasets. UltraChat (Ding et al., 2023), establishes various scopes and systematically generates a multitude of instructions within each designated area. Orca (Mitra et al., 2023) directly apply GPT4 to generate reasoning steps for given instructions. SelFee (Ye et al., 2023) utilizes ChatGPT to enhance the response quality. Reflection-Tuning (Li et al., 2023b) improves both the instruction and response sequentially by reflecting on specific criteria. DEITA (Liu et al., 2023b) utilizes ChatGPT to diversify and then select the data. LIFT (Xu et al., 2023b) also tries to utilize ChatGPT/GPT4 to expand and compress the data.

All the above works are related to ours by involving a teacher model to improve the instruction data, however, all of them are **teacher-dominating**: Both the generation and selection are all decided by the teacher model and without involving the student. We are the first to introduce the **teacher-student collaboration pipeline** and it works fine.

8   Conclusion

Selective Reflection-Tuning, as proposed in this paper, marks a significant advancement in data improvement for instruction tuning of Large Language Models. By integrating an interactive pipeline between a teacher model and a student model, and utilizing the novel metrics of IFD and reversed-IFD, this approach has demonstrated a marked improvement in the quality and relevance of instruction-tuning datasets. The resulting enhancement in model performance across various benchmarks not only attests to the efficacy of our method but also suggests its potential applicability in broader machine learning contexts.

Limitations

The involvement of the student model makes it possible to build high-quality and student-compatible instruction-response data. However, the main limitation of this method is that the data samples selected by different student models are different, thus the statistics (IFD scores and r-IFD scores) need to be calculated again for different student models. We believe the use of model-specific data samples is more reasonable due to the distinct characteristics of different models, and utilizing the statistics-based method is much more efficient than other generation-based methods, the necessity of re-calculation for new models is still not efficient enough.

Acknowledgement

References

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional ai: Harmlessness from ai feedback.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin.

2023a. Alpagasus: Training a better alpaca with fewer data.

Lichang Chen, Khalid Saifullah, Ming Li, Tianyi Zhou, and Heng Huang. 2023b. Claude2-alpaca: Instruction tuning datasets distilled from claude. https://github.com/Lichang-Chen/claude2-alpaca.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

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

Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge.

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.

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

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

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

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

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

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

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 2020. Look at the first sentence: Position bias in question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1109–1121, Online. Association for Computational Linguistics.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback.

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

Ming Li, Jiuhai Chen, Lichang Chen, and Tianyi Zhou. 2024a. Can llms speak for diverse people? tuning llms via debate to generate controllable controversial statements. *ArXiv*, abs/2402.10614.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, and Tianyi Zhou. 2023b. Reflection-tuning: Recycling data for better instruction-tuning. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024b. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. *ArXiv*, abs/2402.00530.

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

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023d. Self-alignment with instruction backtranslation.

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

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. 2023a. Mmc: Advancing multimodal chart understanding with large-scale instruction tuning.

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

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023c. G-eval: Nlg evaluation using gpt-4 with better human alignment.

S. Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *ArXiv*, abs/2301.13688.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland. Association for Computational Linguistics.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agrawal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, Hamid Palangi, Guoqing Zheng, Corby Rosset, Hamed Khanpour, and Ahmed Awadallah. 2023. Orca 2: Teaching small language models how to reason.

OpenAI. 2023. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

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.

Xwin-LM Team. 2023. Xwin-lm.

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.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment.

Thuy-Trang Vu, Xuanli He, Gholamreza Haffari, and Ehsan Shareghi. 2023. Koala: An index for quantifying overlaps with pre-training corpora.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2023a. Openchat: Advancing open-source language models with mixed-quality data.

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

Tianlu Wang, Ping Yu, Xiaoqing Ellen Tan, Sean O'Brien, Ramakanth Pasunuru, Jane Dwivedi-Yu, Olga Golovneva, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023c. Shepherd: A critic for language model generation.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023d. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

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

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2024. Lamini-lm: A diverse herd of distilled models from large-scale instructions.

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

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models.

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

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. 2023. Selfee: Iterative self-revising llm empowered by self-feedback generation. Blog post.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

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

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

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 Prompt for Evaluation

We provide the detailed prompt we used for the pair-wise comparison in Figure 4.

Prompt for Performance Evaluation

System Prompt
You are a helpful and precise assistant for checking the quality of the answer.

User Prompt
[Question]
Question
[The Start of Assistant 2's Answer]
Answer 2
[The End of Assistant 2's Answer]
[The Start of Assistant 2's Answer]
Answer 2
[The End of Assistant 2's Answer]

We would like to request your feedback on the performance of two AI assistants in response to the user question displayed above.
Please rate the helpfulness, relevance, accuracy, level of details of their responses. Each assistant receives an overall score on a scale of 1 to 10, where a higher score indicates better overall performance. Please first output a single line containing only two values indicating the scores for Assistant 1 and 2, respectively. The two scores are separated by a space. In the subsequent line, please provide a comprehensive explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment.

Figure 4: The prompt we used to request ChatGPT or GPT4 to evaluate the responses.

B Prompt for Reflection

The prompts for the reflection are shown in Figure 5 and Figure 6.

Prompt for Reflecting Instruction

System Prompt
You are a helpful, precise but picky assistant for checking the quality of a given instruction.

User Prompt
[Instruction]
Instruction
[The Start of Answer]
Answer
[The End of Answer]

We would like you to answer several questions related to the quality of a given instruction.
1. Why this instruction is not good? First analyze the instruction based on the Complexity of the Topic, Level of Detail Required, Knowledge Required, Ambiguity of the Instruction and Logical Reasoning or Problem-Solving Involved. Then analyze why this answer is not good for the given instruction based on the Helpfulness, Relevance, Accuracy and Level of Details. Finally, analyze why this bad instruction leads to a bad answer.
2. Based on the reason you provided, generate a new and complete instruction that is complex and difficult to answer directly. Make sure the new instruction is relevant but independent to the original instruction, which can be answered without knowing the original instruction, put the new instruction in the format of [New Instruction] your instruction [End]
3. Answer the newly generated instruction as detailed as possible, in the format of [New Answer] your answer [End]

Figure 5: The prompt we used to modify the existing instruction.

Prompt for Reflecting Response

System Prompt
You are a helpful, precise but picky assistant for checking the quality of the answer to a given instruction.

User Prompt
[Instruction]
Instruction
[The Start of Answer]
Answer
[The End of Answer]

We would like you to answer several questions related to the quality of the answer to the given instruction.
1. Why this answer is not good for the given instruction? Analyze based on the Helpfulness, Relevance, Accuracy, and Level of Details.
2. Based on the reason you provided, generate a better answer, new and complete, as detailed as possible, in the format of [Better Answer] your answer [End]

Figure 6: The prompt we used to modify the existing response.

C   Evaluation Metric

C.1   Pair-wise comparison

Evaluation of the responses generated by LLMs is an open problem that plenty of researchers are still working on, due to the lack of real ground truth for the open-domain questions, most of the previous methods can not be directly implemented for judging the instruction-following ability of LLMs. However, using LLM as a judge, e.g., GPT4, for evaluation is recently a widely accepted and common practice (Touvron et al., 2023; Chiang et al., 2023; Dettmers et al., 2023; Liu et al., 2023c; Chiang and Lee, 2023). Previous studies (Zheng et al., 2023; Li et al., 2023e) have shown that GPT4's evaluations are consistent with human evaluations. We utilized the testing instruction set from WizardLM (Xu et al., 2023a) which contains 218 diverse human-curated instructions, which are categorized into specific sub-categories.

Specifically, we directly follow the evaluation method from Chen et al. (2023a); Li et al. (2023c), which contains rating each model-generated response on a scale spanning from 1 to 10, with scores encapsulating several aspects such as accuracy and relevance. To further mitigate the positional bias elaborated upon in (Ko et al., 2020; Wang et al., 2023b), model-generated outputs are presented to the LLM judge in two distinct sequences and subsequently scored. Hence, a model's dominance is ratified under the following conditions: **Wins:** Exhibits superiority in both sequences or prevails in one while maintaining parity in the alternate sequence. **Tie:** Demonstrates parity across both sequences or prevails in one while faltering in the alternate. **Loses:** Underperforms in both sequences or maintains parity in one while being eclipsed in the alternate.

C.2   Alapca Eval Leaderboard

AlpacaEval Leaderboard offers an LLM-centric automatic assessment utilizing the AlpacaFarm (Dubois et al., 2023) evaluation dataset. It is an automated evaluation mechanism for LLMs that offers efficiency, cost-effectiveness, and reliability. Operating on the AlpacaFarm evaluation dataset, it gauges models' proficiency in adhering to generic user instructions. The generated outputs are juxtaposed against benchmark responses from Davinci003. Empirical evidence suggests that AlpacaEval's alignment with ground truth annotations sourced from human experts is notably high.

C.3   Open LLM Leaderboard

The Huggingface Open LLM Leaderboard employs the evaluation methodology from (Gao et al., 2021), providing a cohesive framework for assessing generative language model capabilities across a spectrum of evaluation tasks. It focuses on 4 pivotal benchmarks: ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), and TruthfulQA (Lin et al., 2022).

C.4   MT-Bench

We also provide the performances of our sRecycled Models on MT-bench, as shown in Table 5. Since our training focused on 1-turn instructions and did not include any multi-turn data, the 1-turn score on the MT bench is promising and comparable to LLaMA2-13B-chat, while the 2-turn score is not that satisfactory. However, the Vicuna dataset Chiang et al. (2023) can introduce multi-turn dialog data to the model training. Hence, we tried training with our data based on the existing Vicuna 7B v1.5 model, whose result is reported in the last row as "sRecycled Wiz + Vicuna 7B". Compared with the original Vicuna model, the 1-turn, 2-turn, and overall scores are improved dramatically and the overall score is similar to the performance of Vicuna-13B.

	1-turn	2-turn	Overall
sRecycled Alpaca 7B	6.653	2.888	4.891
sRecycled Wiz 7B	6.538	4.588	5.563
Vicuna 7B v1.5	6.569	5.588	6.078
sRecycled Wiz + Vicuna 7B	7.063	5.975	6.519

Table 5: The MT-Bench results of our models, including 1-turn, 2-turn, and Overall Scores.

C.5   Human Study

To further validate the superiority of our method, we conducted a further human study to further evaluate the effectiveness of our method. In the test set, there are 27 sub-categories that have 4 or more testing instructions, thus we randomly sampled 4 instructions from each sub-category to form a set containing 108 instructions. Then 3 human participants are given the task of comparing the responses generated by the comparing models with the criteria same as the previous pair-wise evaluation. For each comparison, 3 options are given (Win, Tie, and Loss) and the final results are determined by the majority voting of the participants.

D   Implementation Details

For the Llama2 pre-trained model (Touvron et al., 2023), we utilize the prompt and code base from Vicuna (Chiang et al., 2023) and flash attention (Dao et al., 2022) while the overall training arguments are aligned with protocols from Alpaca and WizardLM datasets. The Adam optimizer (Kingma and Ba, 2017), with a 2×10^{-5} learning rate for the 7b model and a 1×10^{-5} learning rate for the 13b model, and a batch size of 128, steer the training across three epochs with a max length of 2048. The warmup rate is set to 0.03.

E   Statistic Analysis

E.1   Basic Data Statistics

In this section, we delve into a quantitative analysis of the instruction-response data, pre- and post-application of our methodology, as delineated in Table 6. We first compare both "Recycled Data" and "sRecycled Data" to the original data.

Observationally, there's an increase in the average token length of instructions within the Alpaca dataset, whereas a decrement manifests for the WizardLM dataset, epitomizing the method's adept adaptability. The succinctness and elementary nature of the Alpaca dataset's instructions warrant an enhancement in intricacy through our method, thereby elongating their length. Conversely, the pre-existing complexity and intricacy in WizardLM's instructions render our algorithm inclined towards succinctness. Pertaining to the response section, there's a marked propensity of our approach to engender detail-rich textual content, leading to relatively long responses.

Moreover, leveraging Sentence-BERT (Reimers and Gurevych, 2019), we quantify the coherence metric between instructions and their affiliated responses. It's discernible that our technique invariably fabricates samples with better coherence, signifying a superior alignment between modulated instructions and consequent responses. Additionally, to elucidate the metamorphosis in instructional difficulty, we employ the IFD score, executed on the pre-trained llama2-7b language model to check the the difficulties of instructions. The increase in IFD scores represents the increase in the overall difficulty of instructions. Moreover, r-IFD is also calculated, and the decrease in r-IFD scores represents the instruction response pair is more related.

E.2   Data Component Distribution

In our selective reflection-tuning, there are four different outcomes for each original data sample: both instruction and response are modified, only instruction is modified, only response is modified, and none of instruction and response are modified. Thus to provide a better view of the data conponents, we provide the pie chart for our sRecycled Alpaca 7B and sRecycled Wizardlm 7B data as shown in Figure 7.

	Comparison of Different Models							
	Ins. len	Res. len	Ins. ppl	Res. ppl 1	Res. ppl 2	Coherent	IFD	r-IFD
Original Alpaca Data	20.7	65.5	34.3	82.6	49.2	0.53	0.71	0.48
Recycled Alpaca Data	37.9	377.2	13.6	4.5	2.9	0.67	0.84	0.50
sRecycled Alpaca Data	31.4	345.9	19.8	4.2	2.8	0.65	0.84	0.36
Original WizardLM Data	123.0	348.5	12.3	17.0	7.5	0.65	0.71	0.52
Recycled WizardLM Data	66.9	518.7	10.0	3.2	2.5	0.73	0.83	0.41
sRecycled WizardLM Data	70.7	519.6	12.0	3.1	2.4	0.72	0.83	0.39

Table 6: The comparison of some basic statistics. "Ins. len" and "Res. len" represent the average token length of the instructions and responses. "Ins. ppl" represents the average perplexity of instructions. "Res. ppl 1" and "Res. ppl 2" represent response perplexities without or with the context of corresponding instructions. All the perplexity is calculated upon our initial pre-trained model llama2-7b. "Coherent" represents the coherent score calculated by SentenceBert. "IFD" represents the instruction-following difficulty score proposed by Cherry LLM (Li et al., 2023c) and "r-IFD" represents the reversed instruction-following difficulty score proposed by us.
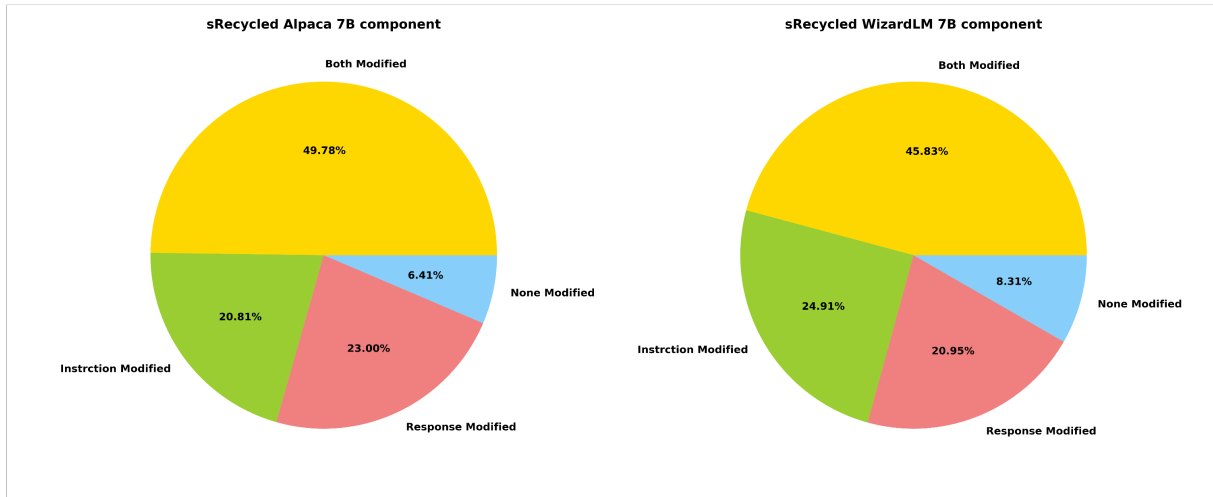

Figure 7: The component distribution of the sRecycled Alpaca 7B and sRecycled Wizardlm 7B data.

F Detailed Few Data Scenario

The detailed performances in the few data scenarios are shown in TABLE 7 and comparisons with the randomly selected method are shown in TABLE 8.

	Huggingface Open LLM Leaderboard					AlpacaEval
	Average	ARC	HellaSwag	MMLU	TruthfulQA	AlpacaEval
sRecycled WizardLM 7B (1%) (463)	57.31	54.86	78.40	46.17	49.79	67.79
sRecycled WizardLM 7B (2%) (926)	57.80	54.69	78.80	47.00	50.70	74.29
sRecycled WizardLM 7B (3%) (1,390)	57.34	55.12	78.80	42.68	49.16	74.50
sRecycled WizardLM 7B (5%) (2,316)	57.91	54.86	79.83	46.69	50.23	77.78
sRecycled WizardLM 7B (10%) (4,632)	57.71	55.46	79.56	46.83	48.98	78.43
sRecycled WizardLM 7B (30%) (13,897)	56.89	54.61	79.25	44.67	49.05	82.48
sRecycled WizardLM 7B (50%) (23,163)	56.98	55.11	78.87	45.31	48.63	81.47
sRecycled WizardLM 7B (70%) (32,428)	56.63	54.95	78.55	46.31	46.71	81.47
sRecycled WizardLM 7B (100%) (46,325)	56.79	54.78	77.86	45.63	48.91	83.21

Table 7: The comparison of performance on Huggingface Open LLM Leaderboard and AlpacaEval Leaderboard by using different amounts of selective recycled WizardLM data. In the first parentheses are the percentage of data used for tuning and in the second parentheses are the specific amount of number used.

	Huggingface Open LLM Leaderboard					AlpacaEval
	Average	ARC	HellaSwag	MMLU	TruthfulQA	AlpacaEval
sRecycled Wiz 7B (2%) (926) (IFD)	57.80	54.69	78.80	47.00	50.70	74.29
sRecycled Wiz 7B (2%) (926) (Random)	56.13	54.77	78.98	43.15	47.64	72.13
sRecycled Wiz 7B (5%) (2,316) (IFD)	57.91	54.86	79.83	46.69	50.23	77.78
sRecycled Wiz 7B (5%) (2,316) (Random)	57.07	54.10	78.97	46.54	48.67	76.40
sRecycled Wiz 7B (10%) (4,632) (IFD)	57.71	55.46	79.56	46.83	48.98	78.43
sRecycled Wiz 7B (10%) (4,632) (Random)	57.06	54.86	78.09	46.82	48.46	77.11
sRecycled Wiz 7B (30%) (13,897) (IFD)	56.89	54.61	79.25	44.67	49.05	82.48
sRecycled Wiz 7B (30%) (13,897) (Random)	56.80	54.95	78.07	47.39	46.81	79.73

Table 8: The comparison of performance on Huggingface Open LLM Leaderboard and AlpacaEval Leaderboard by using different strategies in the few data scenarios.

G Ablation on Larger Evaluate Set

The evaluation set used on the main page in Table 4 is the WizardLM test set, which contains 218 human-written instructions, and is currently one of the most widely used test sets. Another widely used test set is the Vicuna test set, which is used in MT-Bench, but it contains only 80 instructions and the results are presented in Appendix C. Thus the test set we used for ablation is almost three times the Vicuna set. Moreover, in our evaluation, every comparison will be processed twice to eliminate the potential position bias. Thus we don't think it would be regarded as a really small test set.

However, to further validate the effectiveness of our method, we further combine the Vicuna (Chiang et al., 2023) test set (80), Koala (Vu et al., 2023) test set (180), WizardLM (Xu et al., 2023a) test set (218), Self-instruct (Wang et al., 2023d) test set (252), and LIMA (Zhou et al., 2023) test set (300) into a huge evaluation set of 1030 instructions for the ablation study as shown in Table 9. The results on this huge test set share similar trends compared with using the WizardLM test set alone, indicating the effectiveness of our method.

	Win	Tie	Lose	Win Rate
vs. Original WizardLM	738	126	166	1.556
vs. Reflect on Ins.	688	196	146	1.548
vs. Reflect on Res.	360	388	282	1.078
vs. Reflect on Ins. + Res.	329	434	267	1.044
vs. Select by Randomness	401	416	213	1.149
vs. Select by Coherence	360	444	226	1.108
vs. Select by Perplexity	307	438	285	1.050

Table 9: The pair-wise comparison between our sRecycled WizardLM 7B with other models. The "win", "Tie" and "Lose" represent the number of wins or losses of sRecycled WizardLM 7B. The win rate is calculated as (Num(Win) − Num(Lose)) / Num(All) +1.

H Examples for r-IFD Illustrtaion

Example 1:
(r-IFD=0.148, Low, Prefered)
Instruction:
Reverse a linked list.
Response:

```
def reverse(head):
    prev = None
    cur = head

    while cur:
        next = cur.next
        cur.next = prev
        prev = cur
        cur = next
    head = prev

    return head
```

Example 2:
(r-IFD=0.921, High, Not Prefered)
Instruction:
Identify the type of sentence "I drove to the store yesterday".
Response:
Declarative sentence.

In the first example, after reading through the given code, LLM can easily understand the task and guess what this code is for, indicating sufficient information in the response and its good match to the instruction. However, in the second example, the response is not able to provide enough information to derive the instructions and is vague in various aspects. It indicates that the response might not be feasible to be reasoned by the model and thus needs to be improved.