Reflection-Tuning: Data Recycling Improves LLM Instruction-Tuning

Ming Li[1], **Lichang Chen**[1], **Jiuhai Chen**[1], **Shwai He**[1], **Heng Huang**[1], **Jiuxiang Gu**[2], **Tianyi Zhou**[1]
[1]University of Maryland [2]Adobe Research
{minglii, bobchen, tianyi}@umd.edu

Abstract

Recent advancements in Large Language Models (LLMs) have expanded the horizons of natural language understanding and generation. Notably, the output control and alignment with the input of LLMs can be refined through instruction tuning. However, as highlighted in several studies, low-quality data in the training set are usually detrimental to instruction tuning, resulting in inconsistent or even misleading LLM outputs. We propose a novel method, termed "reflection-tuning," which addresses the problem by self-improvement and judging capabilities of LLMs. This approach utilizes an oracle LLM to recycle the original training data by introspecting and enhancing the quality of instructions and responses in the data. Extensive experiments on widely used evaluation benchmarks show that LLMs trained with our recycled data outperform those trained with existing datasets in various benchmarks. Codes, data, and models are available in `https://github.com/tianyi-lab/Reflection_Tuning`.

1 Introduction

Recently, the emergence and rapid advancement of Large Language Models (LLMs) [38, 39, 30, 33] have pushed the boundaries of natural language understanding and generation. These models have been applied to a variety of applications [54, 49], from content generation to answering complex questions. A salient feature of LLMs is their potential to follow instructions given to them, a characteristic that has been harnessed to fine-tune and control their outputs. This process, commonly referred to as instruction tuning [43, 25, 5, 26, 8, 53], holds immense promise for customizing LLMs to specific tasks or preferences.

However, instruction tuning is susceptible to the quality of training data. Introducing suboptimal data into the training process can have a cascade of adverse effects. Within the ambit of natural language generation, empirical research delineates that both the integrity and the homogeneity of training data critically modulate the fluency, pertinence, and precision of the generated linguistic content [3, 12, 15]. Datasets exhibiting inconsistencies or subpar quality can precipitate models to engender erratic, prejudiced, or even specious outputs, thereby attenuating their dependability and applicability. Analogous issues permeate instruction-tuning environments. Recent research [48, 34] underscores that even a minuscule fraction of skewed virtual prompts can severely impinge upon a model's operational efficacy, manifesting the susceptibility of large language models (LLMs) to inferior data. On the other hand, ALPAGASUS [5] and Cherry LLM [21] demonstrate that LLMs can achieve enhanced performance metrics by leveraging a select subset of high-quality data.

To address this identified challenge, we introduce a novel method engineered to enhance the quality of extant instruction-tuning datasets autonomously. Drawing inspiration from the evaluative proficiencies of LLMs [55, 7, 23] and contemporary paradigms in self-enhancement [17, 29], our approach hinges on employing an oracle model to introspectively assess and improve the current dataset against specific criteria. This process of data refinement, which we term "reflection-tuning", constitutes a

Workshop on Instruction Tuning and Instruction Following at NeurIPS 2023.

potent and efficacious mechanism to bolster the quality of instruction-tuning data. Crucially, this approach obviates the need for supplementary model training and boasts universal adaptability to diverse instruction-response pair architectures. While analogous methodologies have been broached in recent self-alignment literature [17, 6, 2] – typified by their application of the model for its own enhancement or in aligning model outputs with preconceived critiques – our contribution is pioneering in integrating the reflection and modification paradigm to both instruction and response dimensions, thereby facilitating the genesis of superior instruction-tuning datasets.

Our extensive experiments include comprehensive evaluations of the models trained with reflection-tuning, including the instruction-following evaluations, e.g., Alpaca-Eval, some human-instruction test sets, and benchmarks. Since GPT-4 demonstrates higher agreement with human preferences than agreements between humans [55], we utilize it as our judge for our main instruction-following evaluations. In the comparison with the models trained with the original datasets, e.g., Alpaca [36], WizardLM [46], our reflection-tuned models achieve much better performance. Specifically, our recycled WizardLM 7B model achieves the highest win rate among other open-source 7B models in the Alpaca-Eval leaderboard. Moreover, Our recycled Alpaca achieves a win rate of 88.75% and our recycled WizardLM achieves a win rate of 81.25% on the Vicuna [7] test set with the same number of training data and model size.

2 Related Work

Instruction Tuning of LLMs. The overarching goal of our work is to enhance the model's instruction-following capability, which is consistent with the previous works [8, 25, 27]. It is discovered that the cross-task generalization ability of LLMs could be enhanced by fine-tuning on NLP datasets which are structured with instruction-response pairs [26, 44]. More recent works [28, 1] have expanded instruction tuning to include open-ended generation tasks, which exhibit enhanced handling of complex human instructions.

High-quality data generation. Our method also targets generating better instruction tuning data [42, 31, 46], but it is orthogonal to the previous work since any kind of instruction-response pairs can be further reflected and improved by our method. Recent works either curate the instruction tuning datasets by human labors, e.g., Dolly [10], Longpre [25] or distill the responses from SOTA LLMs like GPT4 [27], e.g., Alpaca [36], Alpaca-GPT4 [31], Vicuna [7], Koala [40]. There is also some exploration of making the instructions more difficult through the evolution [46], which achieves incredible performance on Alpaca-Eval [23]. Different from them, our method could be treated as a useful posthoc tool, which can further enhance the quality of the instruction tuning data.

LLM self-alignment. Our study contributes to the expanding body of self-alignment [35, 17], i.e., it proves the self-check and self-refine ability of the LLMs. Constitutional-AI [2] first introduces the idea of using the feedback of the AI itself as the preference data to optimize the objectives of helpfulness and harmlessness. Recent works [6, 22, 20] show that LLMs can generate useful signals for debugging, filtering, and finetuning with RL. These works inspire our study prompting the ChatGPT to self-reflect its own generated responses and then self-revise.

3 Methodology

3.1 Preliminaries

Initially, we elucidate and formalize extant methodologies that leverage large language models for instruction-tuning. Let f_θ denote the pre-trained LLM, e.g., Llama, with parameters θ and g the oracle LLM, e.g., ChatGPT. We use other lowercase letters $x, y, z, c, ..$ to 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 $f_\theta(x) = \prod_{i=1}^{n} f(x[i]|x[1, ..., i])$.

In the instruction-following 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

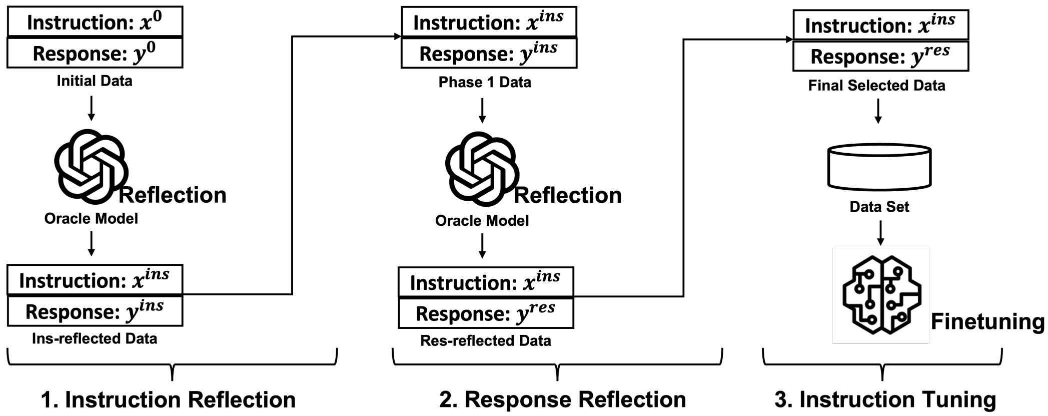

Figure 1: The overall framework of our method.

directly notate this process as $y \sim f(y|x)$. And the loss function for instruction-tuning can be denoted as $L = -\frac{1}{n}\sum_{i=1}^{n} \log f_\theta(y|x)$ where n is the length of response y.

3.2 Reflection-Tuning

As shown in Figure 1, there are two main phases in our method, instruction reflection and response reflection, before the final finetuning. Based on the intuition that students who reflect on the answers usually get higher scores since they can find the errors and make some reasonable changes through the reflection process, and astonished by the self-improvement [17, 29] and judging [55, 7, 23] capability of LLMs, we propose a reflection method for improving the quality of instruction-response pairs. Given the initial base dataset, we are motivated to generate a high-quality version of each data point with an oracle model, ChatGPT for instance. However, a common problem with using LLMs as judges is the failure to obtain diverse results. To overcome this potential problem, inspired by Chain-of-Thought and Tree-of-Thought prompting [45, 50], we further define several specific criteria $\{c_1, ..., c_k\}$ for the oracle model to follow, and respond to those specific criteria with critical responses $\{z_1, ..., z_k\}$, respectively. Then the responses to these criteria can bridge the generation of new instruction-response pairs.

3.2.1 Reflection on Instruction

Specifically, in the instruction reflection phase, the oracle model g is required to reflect on the given instruction-response pair (x^0, y^0) from the original dataset D^0 with some specific criteria $\{c_1^{ins}, ..., c_k^{ins}\}$ and then generate a better instruction-response pair (x^{ins}, y^{ins}) according to its reflection results. With the criteria given, the oracle model g is able to generate critical responses:

$$[z_1^{ins}, ..., z_k^{ins}] \sim g(z_1^{ins}, ..., z_k^{ins}|x^0, y^0, c_1^{ins}, ..., c_k^{ins}) \tag{1}$$

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) for the generation of the new instruction and response pair:

$$[x^{ins}, y^{ins}] \sim g(x^{ins}, y^{ins}|x^0, y^0, c_1^{ins}, ..., c_k^{ins}, z_1^{ins}, ..., z_k^{ins}) \tag{2}$$

where in practice the above process is sampled as a continuous language sequence, and the critical responses would not be decomposed from the whole outputs. The criteria used for instruction are "the Complexity of the Topic", "the Level of Detail Required for response", "Knowledge Required for response", "the Ambiguity of the Instruction" and whether "Logical Reasoning or Problem-Solving Involved".

3.2.2 Reflection on Response

Although both instruction and response are modified, the corresponding response y^{ins} for a given modified instruction x^{ins} is not 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 $\{c_1^{res}, ..., c_m^{res}\}$. The overall process can be noted as:

$$y^{res} \sim g(y^{res}|x^{ins}, y^{ins}, c_1^{res}, ..., c_m^{res}, z_1^{res}, ..., z_m^{res}) \tag{3}$$

where z_i^{res} represents the critical response of ith response criteria c_i^{res}. After the above process, the instruction and response pair $(x^{ins}, y^{res}))$ is regarded as the recycled data pair which will be used for instruction-tuning of model f_θ. The criteria used for instruction are "Helpfulness", "Relevance", "Accuracy", and "Level of Details".

We name the whole above process as a recycling process, which greatly improves the quality of the previous dataset. Then the raw model f_θ will be trained on the newly generated recycled dataset, and the newly generated models are notated as "Recycled Models", eg. Recycled Alpaca.

4 Experimental Setup

4.1 Base Datasets

The Alpaca dataset [36], sourced from Stanford University, offers $52,002$ instruction-following samples. Developed via the self-instruct paradigm [42], it leveraged the capabilities of the text-davinci-003 model. This dataset, while a pioneering attempt in instruction tuning for the LLaMA model, raised concerns about data quality owing to its reliance on the text-davinci-003 model.

On the other hand, the WizardLM dataset [46], which employs the sophisticated Evol-Instruct algorithm, is a refined collection encompassing a total of $250,000$ instruction samples. Two primary evolutionary trajectories, namely "In-depth Evolving" and "In-breadth Evolving", are introduced within this dataset. These trajectories are specifically designed to allow a base instruction to progress either in terms of intricate details or in its overall scope. To enhance data fidelity, ChatGPT 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.

4.2 Implementation Details

Rooted in the Llama2-7b pre-trained model [39], we utilize the prompt and code base from Vicuna and flash attention while the overall training arguments are aligned with protocols from Alpaca and WizardLM datasets. The Adam optimizer [18], with a 2×10^{-5} learning rate and a batch size of 128, steers the training across three epochs with a max length of 2048. The warmup rate is set to 0.03.

4.3 Evaluation Metric

4.3.1 Pari-wise comparison

The task of quantitatively evaluating the instruction-adherence efficacy of LLMs presents considerable challenges. Despite a wealth of research endeavoring to design automated evaluation metrics for LLMs [4], the gold standard remains subjective human evaluation. However, such manual assessments are not only resource-intensive but are also susceptible to inherent human biases.

Incorporating methodologies from cutting-edge LLM evaluations [55, 7, 23], we operationalize GPT4 and ChatGPT as evaluation benchmarks. As delineated in [5], models subjected to evaluation are prompted to generate outputs for each instruction in the test corpus. Subsequent to this, an API-driven model, be it GPT4 or ChatGPT, allocates a score to each response. A model's superiority on this dataset hinges on its endorsement by the adjudicating model.

The adjudication phase entails rating each model-generated response on a scale spanning from 1 to 10, with scores encapsulating facets such as pertinence and precision. To mitigate the positional bias elaborated upon in [19, 41], model-generated outputs are presented to the adjudicating entity 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. This adjudication paradigm underpins our experimental findings.

4.4 Benchmarks

Two prominent benchmarking platforms for LLMs are highlighted: the Huggingface Open LLM Leaderboard[1] and the AlpacaEval Leaderboard[2]. The Huggingface Open LLM Leaderboard employs the evaluation methodology from [14], providing a cohesive framework for assessing generative language model capabilities across a spectrum of evaluation tasks. It focuses on 4 pivotal benchmarks: ARC [9], HellaSwag [52], MMLU [16], and TruthfulQA [24]. Specifically, ARC is a specialized dataset curated for assessing the proficiency of models in answering science questions tailored for grade-school levels. The challenge employs a 25-shot learning paradigm, implying that models are exposed to 25 examples prior to evaluation. HellaSwag is Specifically designed to probe models on their commonsense inference capabilities, which utilizes a 10-shot learning setup, meaning models are trained on 10 sample instances before being tested. MMLU is a comprehensive evaluation suite designed to gauge a model's multitasking learning capability across a diverse range of 57 tasks. These tasks span a myriad of domains including but not limited to elementary mathematics, US history, computer science, and jurisprudence. TruthfulQA is constructed to appraise a model's susceptibility to perpetuating misinformation or falsehoods, which are ubiquitously found online.

On the other hand, the AlpacaEval Leaderboard offers an LLM-centric automatic assessment utilizing the AlpacaFarm [13] 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. These benchmarks are subsequently auto-annotated by either GPT-4, Claude, or ChatGPT, leading to the determination of the aforementioned win rates. Empirical evidence suggests that AlpacaEval's alignment with ground truth annotations sourced from human experts is notably high. Furthermore, model rankings on the AlpacaEval leaderboard exhibit a strong correlation with rankings derived from human annotators.

5 Experimental Results

5.1 Pair-wise Comparison

As depicted in Figure 2, a juxtaposition between our recycled models and other distinguished models is presented. Remarkably, our models exhibit superior performance across the board, with GPT4 being the sole exception, underscoring the efficacy of our methodology. Notably, SelFee [51] aligns with our motivation in leveraging an oracle model to refine dataset responses while using much more data for training including the Alpaca dataset, the ShareGPT dataset, the FLAN dataset, and extra math and code collections. However, even with much more data used, they overlook the criticality of enhancing the instruction set and neglect the deployment of granular criteria for self-enhancement. This negligence results in their suboptimal performance despite a voluminous training dataset. Importantly, our models, equipped solely with instruction tuning on the Alpaca dataset, surpass several counterparts that employ additional RLHF techniques.

5.2 Alpaca Eval Leaderboard

Table 1 delineates the outcomes on the AlpacaEval Leaderboard. Within this evaluation framework, GPT4 is harnessed as the adjudicating entity, contrasting the responses of the test models against the benchmark set by Davinci003. This comparison provides a direct quantification of a model's capacity for instruction adherence and the intrinsic quality of its output. Notably, our models eclipse the performance of all extant 7B open-source counterparts, with the sole exception being Xwin-LM [37] whose training data is unknown and extra RLHF is implemented. Remarkably, our models even surpass some of the models with a larger parameter count. The eminent positioning of our models on this leaderboard underscores the superior caliber of the responses they generate.

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

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

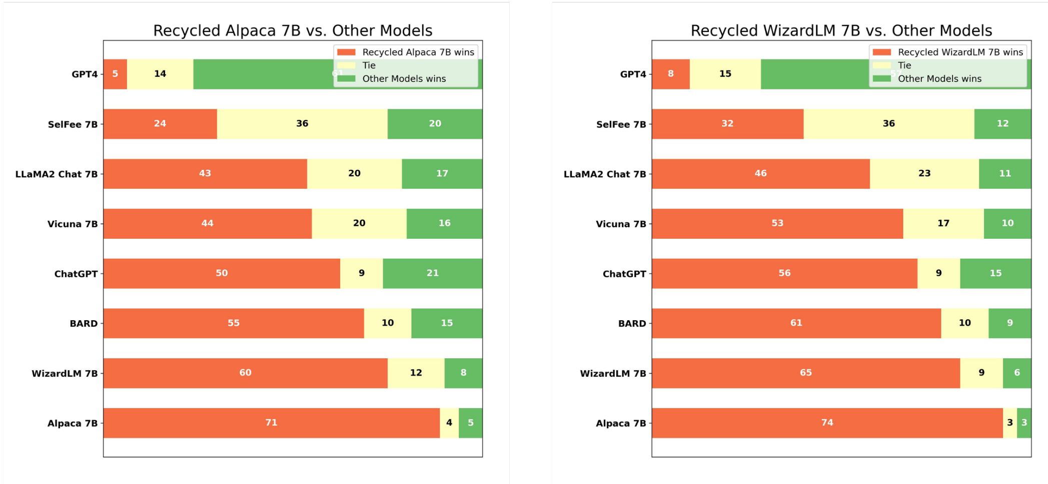

Figure 2: Comparing our recycled models with other renowned models on the Vicuna evaluation set. On the left list the models that are compared. Each bar represents a comparison between our recycled model and the other model. The red parts represent the number of wins and the green parts represent the number of loses. GPT4 is utilized as the judge.

Model	Win Rate	Standard Error	Wins	Draws	Avg Length
GPT4 [27]	95.28	0.72	761	12	1365
Claude 2	91.36	0.99	734	1	1069
ChatGPT	89.37	1.08	716	5	827
XwinLM 7b V0.1 [37]	87.83	-	-	-	1894
Recycled WizardLM 7B (ours)	78.88	1.44	635	0	1494
Recycled Alpaca 7B (ours)	76.99	1.49	619	0	1397
Vicuna 7B v1.3 [7]	76.84	1.49	614	3	1110
WizardLM 13B [46]	75.31	1.51	601	9	985
airoboros 65B	73.91	1.53	587	16	1512
Guanaco 65B [11]	71.80	1.59	578	0	1249
LLaMA2 Chat 7B [39]	71.37	1.59	574	1	1479
Baize-v2 13B [47]	66.96	1.66	538	2	930
Guanaco 33B [11]	65.96	1.67	531	0	1311
Vicuna 7B [7]	64.41	1.69	517	3	1044
Davinci003	50.00	0.00	0	805	307
Guanaco 7B [11]	46.58	1.76	374	2	1364
Alpaca 7B [36]	26.46	1.54	205	16	396

Table 1: The comparison of performance on AlpacaEval Leaderboard.

5.3 Open LLM Leaderboard

Table 2 showcases the performance comparison on the Huggingface Open LLM Leaderboard with some related models. With our Recycle mechanism, our models achieve better average performances across these four representative benchmarks and our results are comparable to llama-2-7b-chat, which is elaborately fine-tuned with extra RLHF.

6 Discussion

6.1 Statistic Analysis

In the ensuing discourse, we delve into a quantitative juxtaposition of the instruction-response data, pre- and post-application of our recycling methodology, as delineated in Table 3. 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

	Huggingface Open LLM Leaderboard				
	Average	ARC	HellaSwag	MMLU	TruthfulQA
Alpaca 7B [36]	50.21	42.65	76.91	41.73	39.55
WizardLM 7B [46]	54.18	51.60	77.70	42.70	44.70
Vicuna 7B v1.3 [7]	55.63	50.43	76.92	48.14	47.01
LLaMA2 Chat 7B [39]	56.34	52.90	78.55	48.32	45.57
Recycled Alpaca 7B (ours)	56.18	53.92	77.68	47.55	45.55
Recycled WizardLM 7B (ours)	56.21	53.92	77.05	48.35	45.21

Table 2: The comparison of performance on Huggingface Open LLM Leaderboard.

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 [32], 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 Instruction-Following Difficulty (IFD) score, as posited by Cherry LLM [21], executed on the nascent pre-trained language model. This score gauges the efficacy of instructions in bolstering response predictions. The consistent ascension in IFD scores lucidly illustrates our instruction's progressive evolution.

	Comparison of Different Models						
	Ins. len	Res. len	Ins. ppl	Res. ppl 1	Res. ppl 2	Coherent	IFD score
Original Alpaca 7B	20.7	65.5	34.3	82.6	49.2	0.53	0.72
Recycled Alpaca 7B	37.9	377.2	13.6	4.5	2.9	0.67	0.83
Original WizardLM 7B	123.0	348.5	12.3	17.0	7.5	0.65	0.66
Recycled WizardLM 7B	66.9	518.7	10.0	3.2	2.5	0.73	0.81

Table 3: The comparison of performance for various models with different metrics. "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. "Coherent" represents the coherent score calculated by SentenceBert. "IFD score" represents the instruction-following difficulty score proposed by Cherry LLM [21].

6.2 Performances on 13B Models

We further train a Recycled Alpaca in the 13B version to further validate the efficacy of our method. With only 52k recycled alpaca data being used for instruction-tuning, our Recycled Alpaca 13B reaches the win rate of 83.42% in the Alpaca Eval leaderboard and reaches an average score of 58.93% on Huggingface Open LLM leaderboard. Considering the small amount of data we used compared with other models, the results are intriguing and satisfactory. We will soon apply our recycled WizardLM data to the 13B model.

7 Conclusion

The evolution of Large Language Models has brought forth unparalleled capacities in natural language processing, especially in the domain of instruction tuning. However, the quality of training data remains a pivotal determinant of model performance. In this work, we introduced the reflection-tuning method, an innovative approach to autonomously improve and recycle the quality of instruction-tuning datasets by leveraging the inherent self-improvement capabilities of LLMs. Our method emphasizes a unique reflect-and-recycle mechanism, a first in the domain, applied comprehensively to both instructions and responses. Experimental results affirm the efficacy of reflection-tuning, with models trained using this method consistently outperforming those trained with traditional datasets. This paves the way for more reliable, consistent, and high-performing LLMs in the future, underscoring the importance of high-quality data recycling and innovative methods in the realm of natural language generation.

References

[1] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[2] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[3] Emily M. Bender and Batya Friedman. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604, 2018.

[4] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models, 2023.

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

[6] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.

[7] 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. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[8] 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. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416, 2022.

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

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

[11] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[12] Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. 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 2185–2194, Hong Kong, China, November 2019. Association for Computational Linguistics.

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

[14] 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. A framework for few-shot language model evaluation, September 2021.

[15] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III au2, and Kate Crawford. Datasheets for datasets, 2021.

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

[17] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[19] Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 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, November 2020. Association for Computational Linguistics.

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

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

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

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

[24] Stephanie Lin, Jacob Hilton, and Owain Evans. 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, May 2022. Association for Computational Linguistics.

[25] 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. The flan collection: Designing data and methods for effective instruction tuning. *ArXiv*, abs/2301.13688, 2023.

[26] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*, 2021.

[27] OpenAI. Gpt-4 technical report, 2023.

[28] 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. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022.

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

[30] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023.

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

[32] Nils Reimers and Iryna Gurevych. 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, November 2019. Association for Computational Linguistics.

[33] Teven Le Scao, Angela Fan, Christopher Akiki, Elizabeth-Jane Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, Franccois Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Rose Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurenccon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa Etxabe, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris C. Emezue, Christopher Klamm, Colin Leong, Daniel Alexander van Strien, David Ifeoluwa Adelani, Dragomir R. Radev, Eduardo Gonz'alez Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady ElSahar, Hamza Benyamina, Hieu Trung Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jorg Frohberg, Josephine L. Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro von Werra, Leon Weber, Long Phan, Loubna Ben Allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, Mar'ia Grandury, Mario vSavsko, Max Huang, Maximin Coavoux, and Mayank Singh. Bloom: A 176b-parameter open-access multilingual language model. *ArXiv*, abs/2211.05100, 2022.

[34] Manli Shu, Jiongxiao Wang, Chen Zhu, Jonas Geiping, Chaowei Xiao, and Tom Goldstein. On the exploitability of instruction tuning, 2023.

[35] Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. Principle-driven self-alignment of language models from scratch with minimal human supervision. *arXiv preprint arXiv:2305.03047*, 2023.

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

[37] Xwin-LM Team. Xwin-lm, 9 2023.

[38] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[39] 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. Llama 2: Open foundation and fine-tuned chat models, 2023.

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

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

[42] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 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, July 2023. Association for Computational Linguistics.

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

[44] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

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

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

[47] Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.

[48] Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. Virtual prompt injection for instruction-tuned large language models, 2023.

[49] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. Harnessing the power of llms in practice: A survey on chatgpt and beyond, 2023.

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

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

[52] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 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, July 2019. Association for Computational Linguistics.

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

[54] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2023.

[55] 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. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

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 3: 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 4: The prompt we used to modify the existing response.