Building and Training: Instruction Following for LLMs

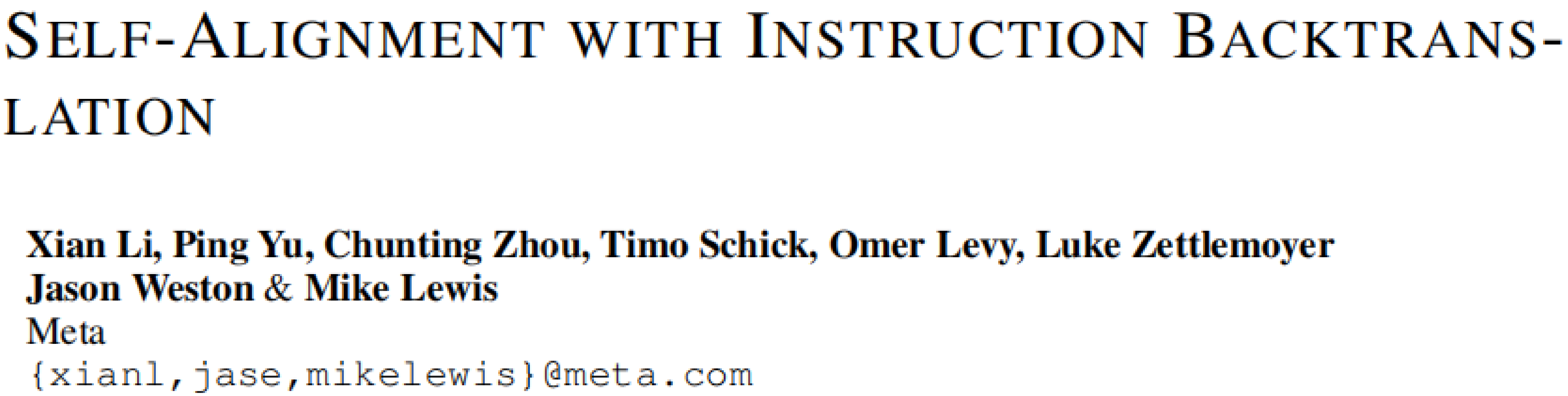
Qingyu Ren   2024/11/7

✓How to construct multi-constraint dataset

- Manually annotating high-quality instructional data is both expensive and time-consuming.
- In the context of long-form text generation, obtaining human annotationfor long texts is difficult.

✓How to use the dataset to train model

- Direct Preference Optimization (DPO), become infeasible in the case of long-form text generation.
- In the constraint-following scenario, the more constraints there are, the harder it becomes, just like how the difficulty of a course increases over time.



Self-Alignment with Instruction Backtranslation

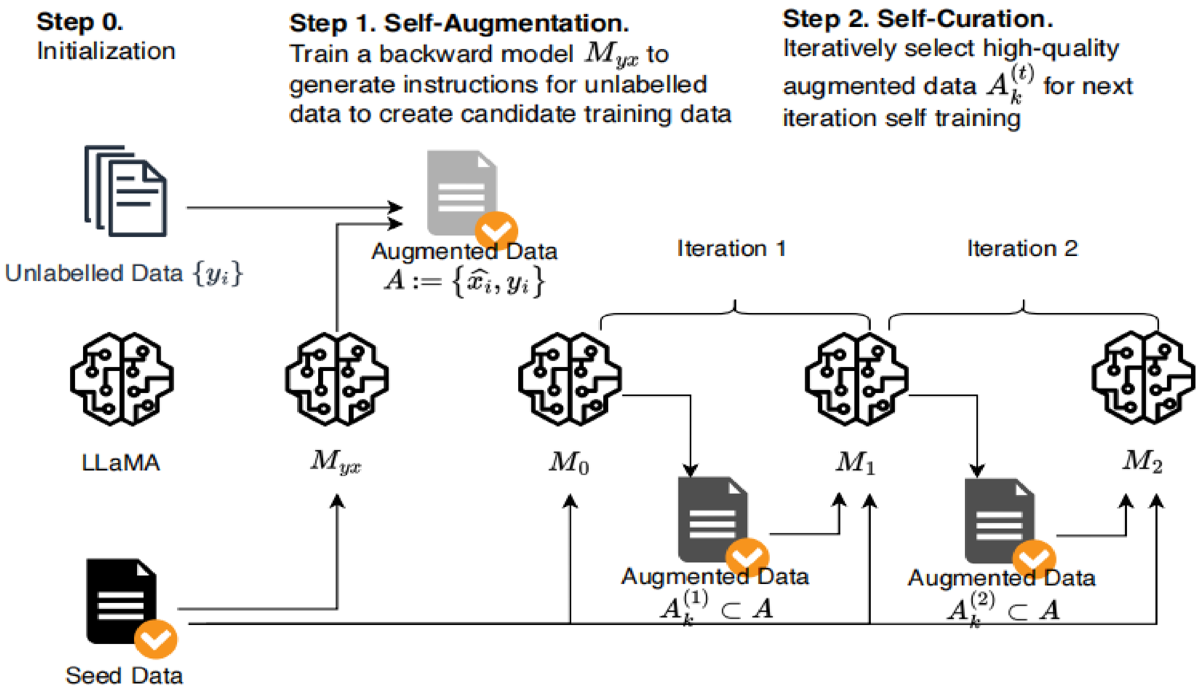
Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer
Jason Weston & Mike Lewis
Meta
{xianl,jase,mikelewis}@meta.com

Motivation

1. Manually annotating high-quality instructional data is both expensive and time-consuming, requiring specialized domain knowledge.

2. There is a large amount of text data available on the internet that is not paired with specific instructions. By effectively leveraging this unannotated data, the amount of data available for training language models can be significantly increased, while also reducing costs.

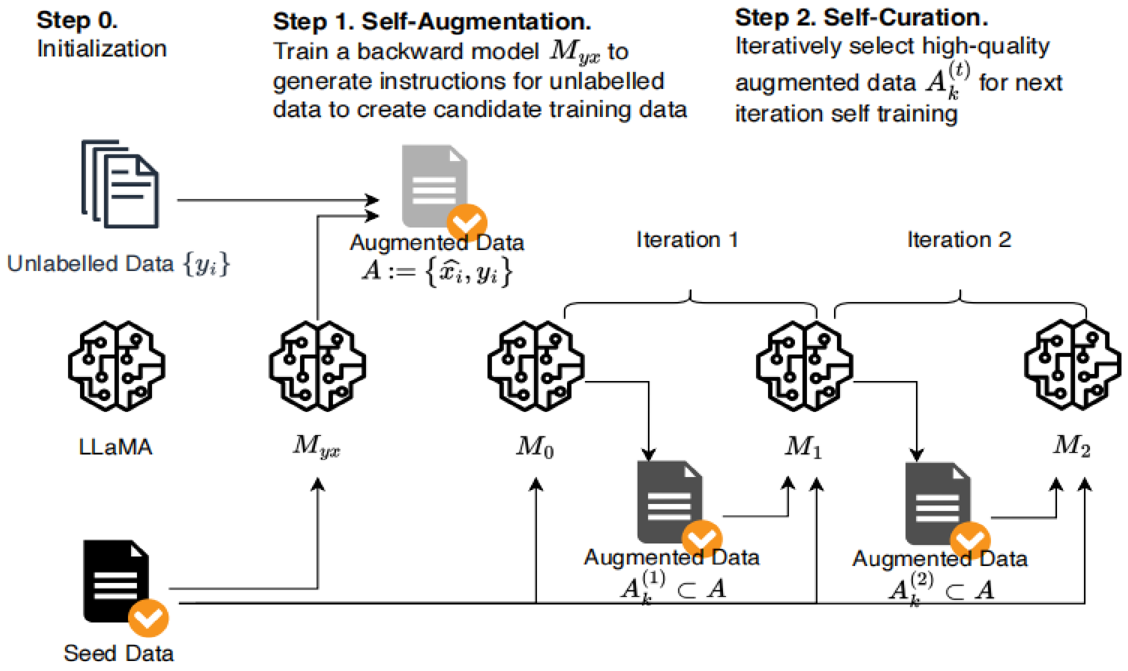
Method


Step 0.
Initialization
Step 1. Self-Augmentation.
Train a backward model M_{yx} to generate instructions for unlabelled data to create candidate training data
Step 2. Self-Curation.
Iteratively select high-quality augmented data $A_k^{(t)}$ for next iteration self training
Unlabelled Data $\{y_i\}$
Augmented Data
$A := \{\hat{x}_i, y_i\}$
Iteration 1
Iteration 2
LLaMA
M_{yx}
M_0
M_1
M_2
Augmented Data
$A_k^{(1)} \subset A$
Augmented Data
$A_k^{(2)} \subset A$
Seed Data

Step 0: Initialization

Start with a seed model and prepare a small amount of seed data. Use these seed data (output-instruction pairs) to perform an initial fine-tuning of the model so that it can predict instructions given an output.

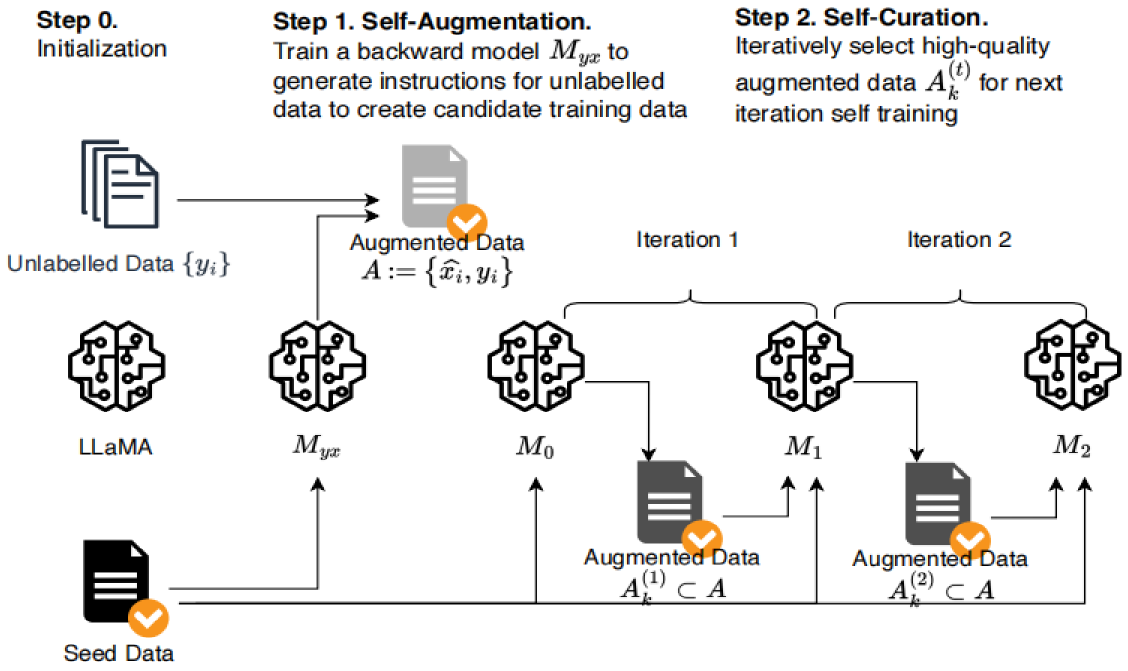
Method


Step 0.
Initialization
Step 1. Self-Augmentation.
Train a backward model M_{yx} to generate instructions for unlabelled data to create candidate training data
Step 2. Self-Curation.
Iteratively select high-quality augmented data $A_k^{(t)}$ for next iteration self training
Unlabelled Data $\{y_i\}$
Augmented Data
$A := \{\hat{x}_i, y_i\}$
Iteration 1
Iteration 2
LLaMA
M_{yx}
M_0
M_1
M_2
Augmented Data
$A_k^{(1)} \subset A$
Augmented Data
$A_k^{(2)} \subset A$
Seed Data

Step 1: Self-Augmentation

Using the initialized fine-tuned base model (denoted as Myx), generate instruction prompts for unlabeled web documents to create a large set of (instruction, output) pairs as augmented training data A.

Method


Step 0.
Initialization
Step 1. Self-Augmentation.
Train a backward model M_{yx} to generate instructions for unlabelled data to create candidate training data
Step 2. Self-Curation.
Iteratively select high-quality augmented data $A_k^{(t)}$ for next iteration self training
Unlabelled Data $\{y_i\}$
Augmented Data
$A := \{\hat{x}_i, y_i\}$
Iteration 1
Iteration 2
LLaMA
M_{yx}
M_0
M_1
M_2
Augmented Data
$A_k^{(1)} \subset A$
Augmented Data
$A_k^{(2)} \subset A$
Seed Data

Combine the Augmented Data and Seed Data as the training dataset, using a system prompt to distinguish between the two data sources.

Seed Data:
Answer in the style of an AI Assistant.

Augmented Data:
Answer with knowledge from web search.

Step 2: Self-Curation

Using the same seed model to evaluate the quality of the candidate instruction-output pairs generated in the self-augmentation step. The model assigns a quality score to each candidate pair, and pairs that score above a certain threshold are selected as high-quality data. This curated dataset is then used for further fine-tuning, resulting in an improved model. This process is iterated, continuously generating higher-quality training data and a progressively better model.

Method

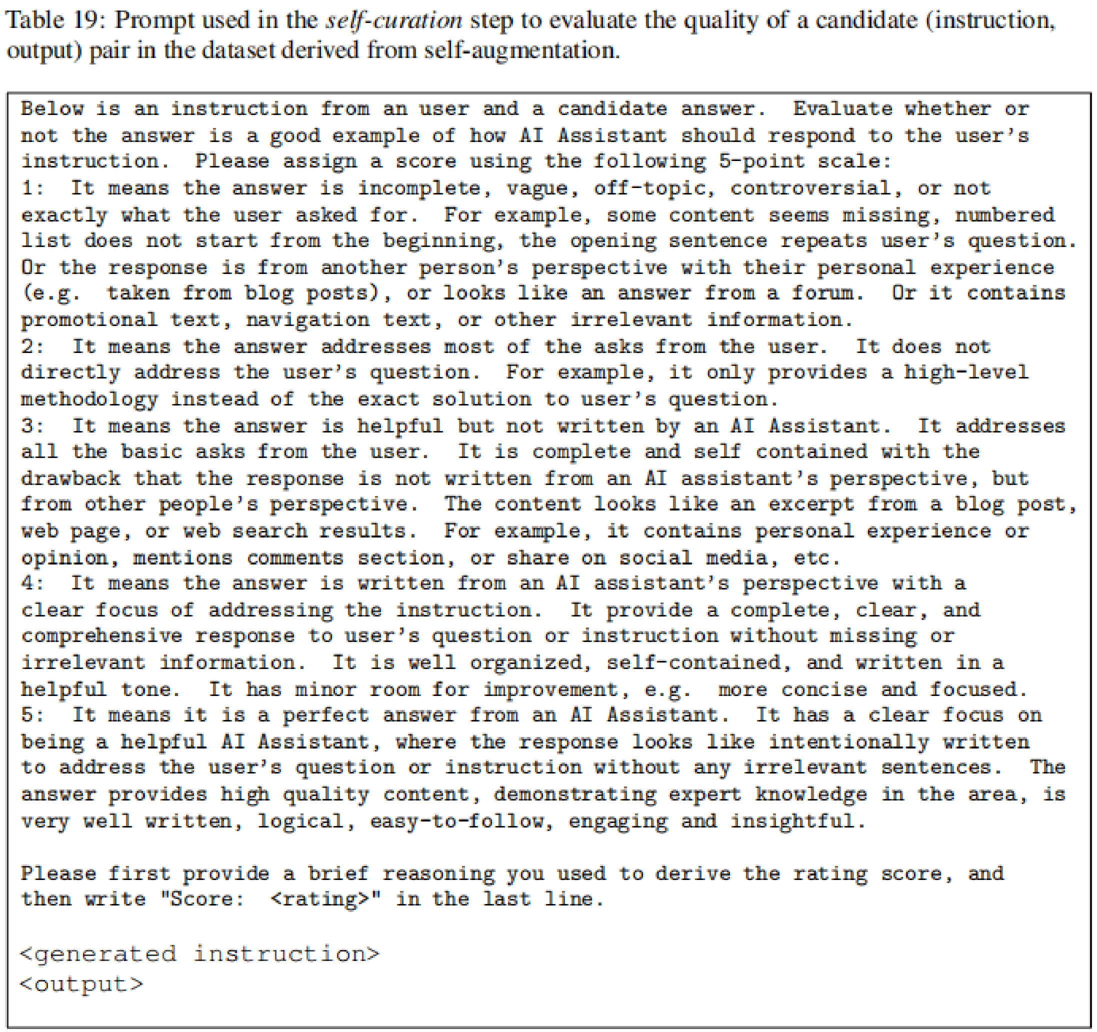


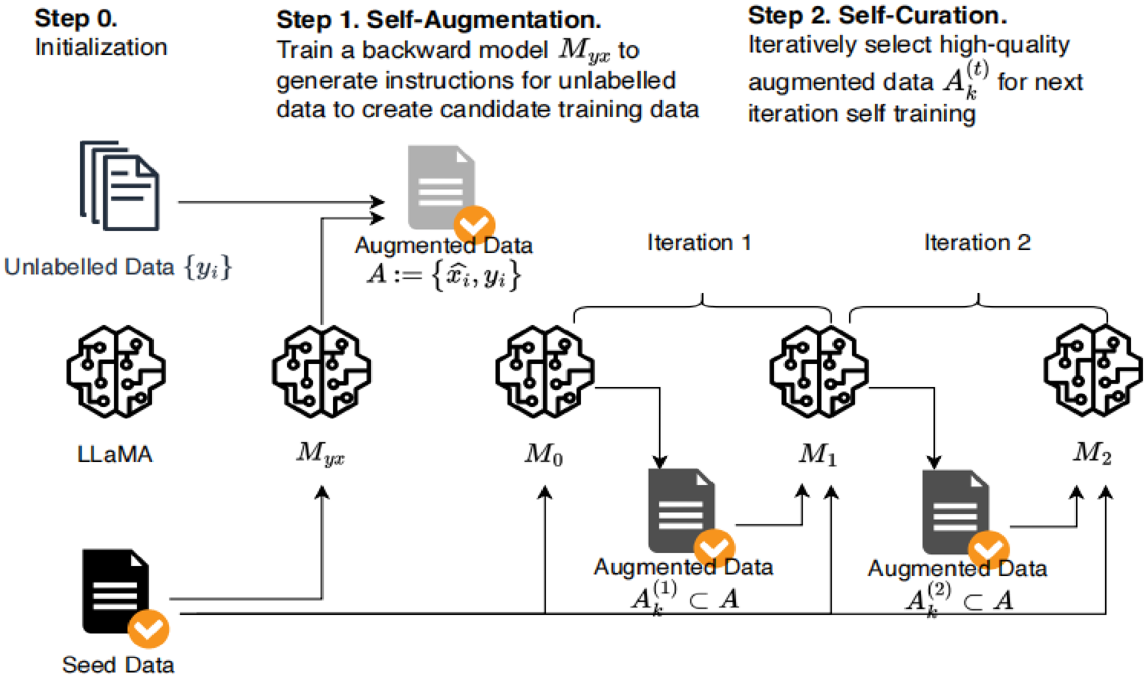
Table 19: Prompt used in the *self-curation* step to evaluate the quality of a candidate (instruction, output) pair in the dataset derived from self-augmentation.

```
Below is an instruction from an user and a candidate answer.  Evaluate whether or
not the answer is a good example of how AI Assistant should respond to the user's
instruction.  Please assign a score using the following 5-point scale:
1:  It means the answer is incomplete, vague, off-topic, controversial, or not
exactly what the user asked for.  For example, some content seems missing, numbered
list does not start from the beginning, the opening sentence repeats user's question.
Or the response is from another person's perspective with their personal experience
(e.g.  taken from blog posts), or looks like an answer from a forum.  Or it contains
promotional text, navigation text, or other irrelevant information.
2:  It means the answer addresses most of the asks from the user.  It does not
directly address the user's question.  For example, it only provides a high-level
methodology instead of the exact solution to user's question.
3:  It means the answer is helpful but not written by an AI Assistant.  It addresses
all the basic asks from the user.  It is complete and self contained with the
drawback that the response is not written from an AI assistant's perspective, but
from other people's perspective.  The content looks like an excerpt from a blog post,
web page, or web search results.  For example, it contains personal experience or
opinion, mentions comments section, or share on social media, etc.
4:  It means the answer is written from an AI assistant's perspective with a
clear focus of addressing the instruction.  It provide a complete, clear, and
comprehensive response to user's question or instruction without missing or
irrelevant information.  It is well organized, self-contained, and written in a
helpful tone.  It has minor room for improvement, e.g.  more concise and focused.
5:  It means it is a perfect answer from an AI Assistant.  It has a clear focus on
being a helpful AI Assistant, where the response looks like intentionally written
to address the user's question or instruction without any irrelevant sentences.  The
answer provides high quality content, demonstrating expert knowledge in the area, is
very well written, logical, easy-to-follow, engaging and insightful.

Please first provide a brief reasoning you used to derive the rating score, and
then write "Score:  <rating>" in the last line.

<generated instruction>
<output>
```

Method



Step 0.
Initialization
Step 1. Self-Augmentation.
Train a backward model M_{yx} to generate instructions for unlabelled data to create candidate training data
Step 2. Self-Curation.
Iteratively select high-quality augmented data $A_k^{(t)}$ for next iteration self training
Unlabelled Data $\{y_i\}$
Augmented Data
$A := \{\hat{x}_i, y_i\}$
Iteration 1
Iteration 2
LLaMA
M_{yx}
M_0
M_1
M_2
Augmented Data
$A_k^{(1)} \subset A$
Augmented Data
$A_k^{(2)} \subset A$
Seed Data

How to improve the model's ability to follow instructions?

Through an iterative process, a stronger model continuously learns on better data to follow instructions more effectively, generating higher-quality outputs.

Experiment

Seed data: 3200 examples from the Open Assistant dataset(chosen from the first turn of the conversation tree)

Base model:LLaMA

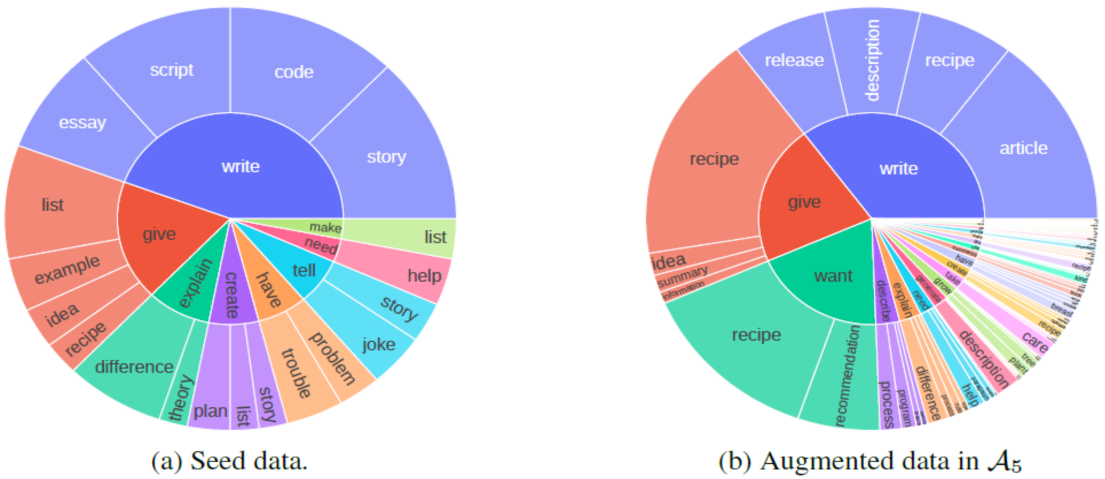
Unlabelled data: English portion of the Clueweb corpus

Baseline: text-davinci-003, LIMA, Guanaco

Evaluation: automatic evaluation using AlpacaEval , human preference evaluation

Experiment

- task diversity analysis of the seed data and augmented data



(a) Seed data.

(b) Augmented data in $\mathcal{A}_5$

Figure 6: Instruction diversity of seed data and augmented data. The inner circle shows common root verbs with the corresponding common noun objects in the outer circle, based on 8% of seed data and 13% of augmented data since not all instructions have the parsed verb-noun structure. The augmentation data appears to possess diversity especially in the long tail, and to be complementary to the existing human-annotated seed data.

in the long tail: uncommon data types

Complementary to seed data

Experiment

- SCALING ANALYSIS
- Data quality vs. data quantity

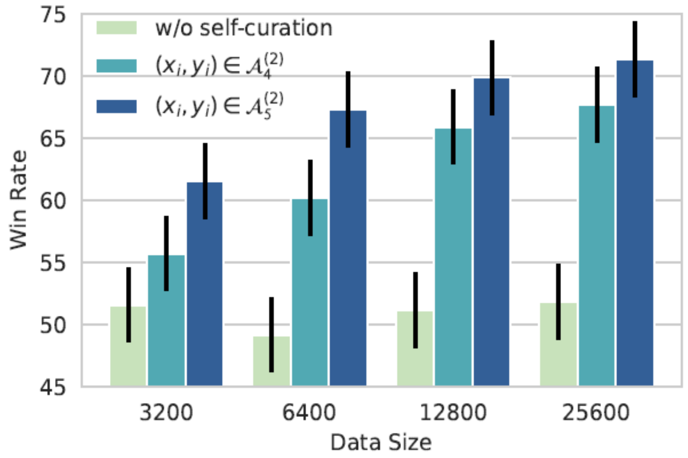


Figure 2: Evaluating self-augmented data of different data size and quality using self-curation. ′

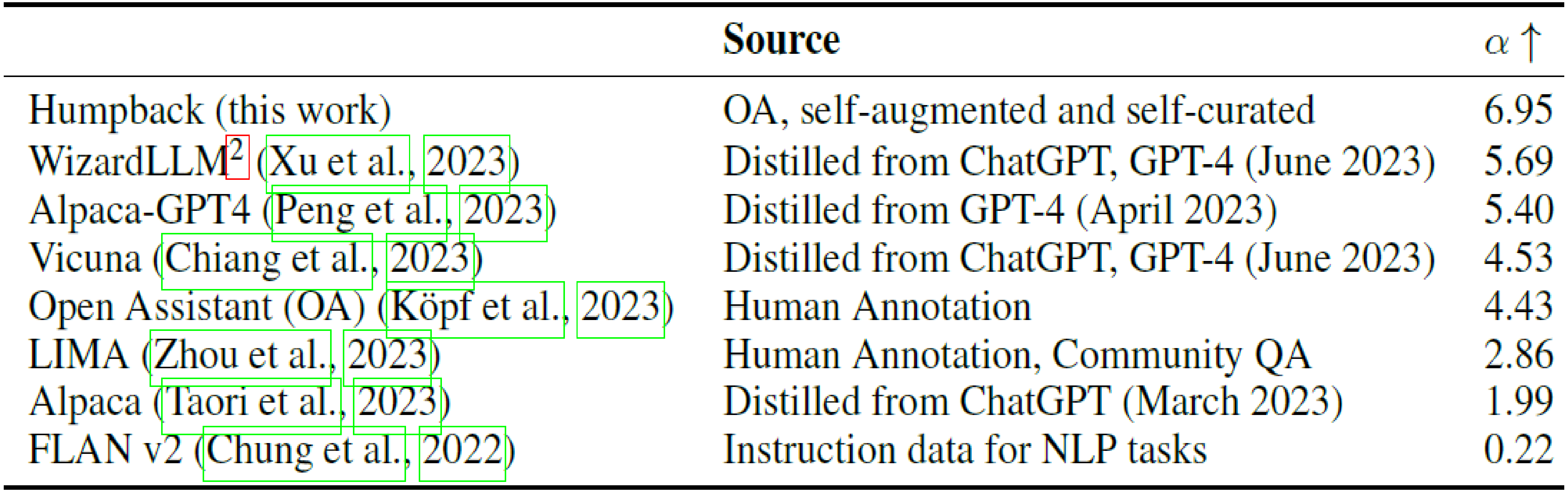
“superficial alignment hypothesis“:

only a few thousands of high-quality instruction following examples are sufficient for aligning a pretrained base model to follow instructions

increasing the quantity of high-quality data provides further gains (whereas increased quantities of low-quality data does not). Contrast to the superficial alignment hypothesis.

Experiment

- Data scaling efficiency Humpback (this work)
- measure the win rate of each model against text-davinci-003 when finetuning 7B LLaMa with the given finetune dataset(k=5)
- an estimate of this efficiency using the data scaling coefficient α, which is calculated by fitting empirical data with w = α logN + C, where w is the win rate measuring generation quality of the model finetuned on N examples.



	Source	$\alpha \uparrow$
Humpback (this work)	OA, self-augmented and self-curated	6.95
WizardLLM[2] (Xu et al., 2023)	Distilled from ChatGPT, GPT-4 (June 2023)	5.69
Alpaca-GPT4 (Peng et al., 2023)	Distilled from GPT-4 (April 2023)	5.40
Vicuna (Chiang et al., 2023)	Distilled from ChatGPT, GPT-4 (June 2023)	4.53
Open Assistant (OA) (Köpf et al., 2023)	Human Annotation	4.43
LIMA (Zhou et al., 2023)	Human Annotation, Community QA	2.86
Alpaca (Taori et al., 2023)	Distilled from ChatGPT (March 2023)	1.99
FLAN v2 (Chung et al., 2022)	Instruction data for NLP tasks	0.22

Experiment

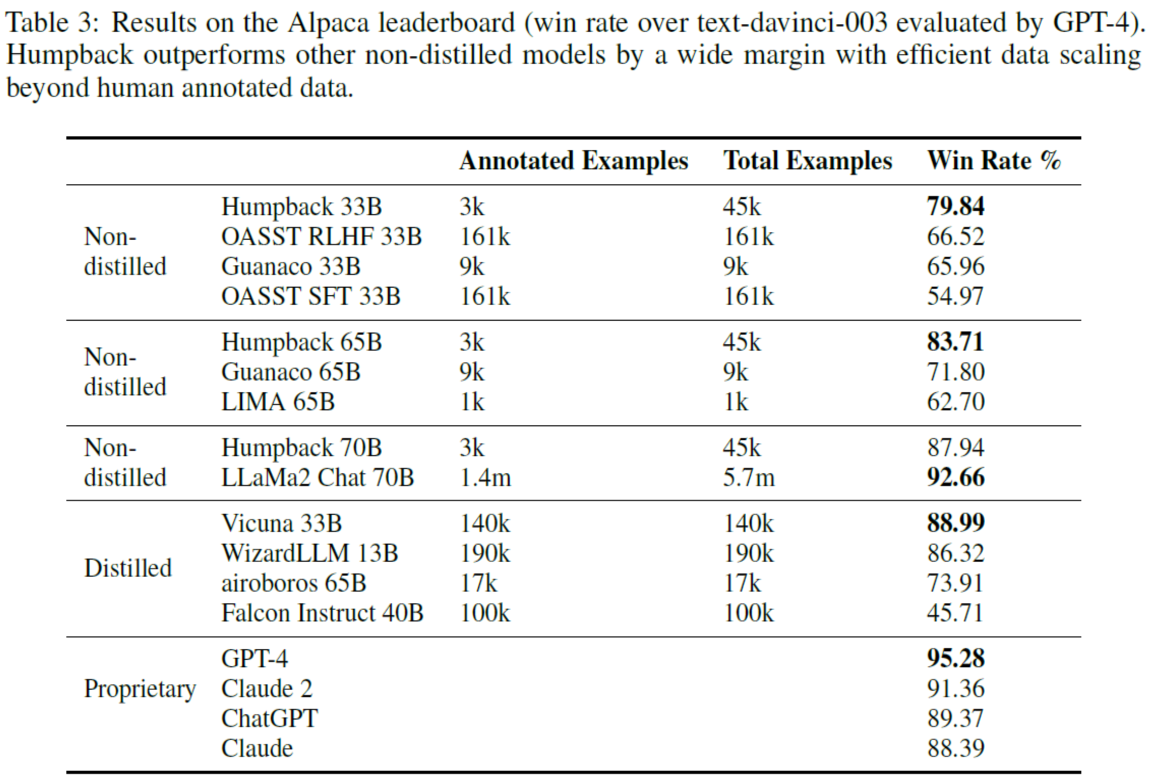
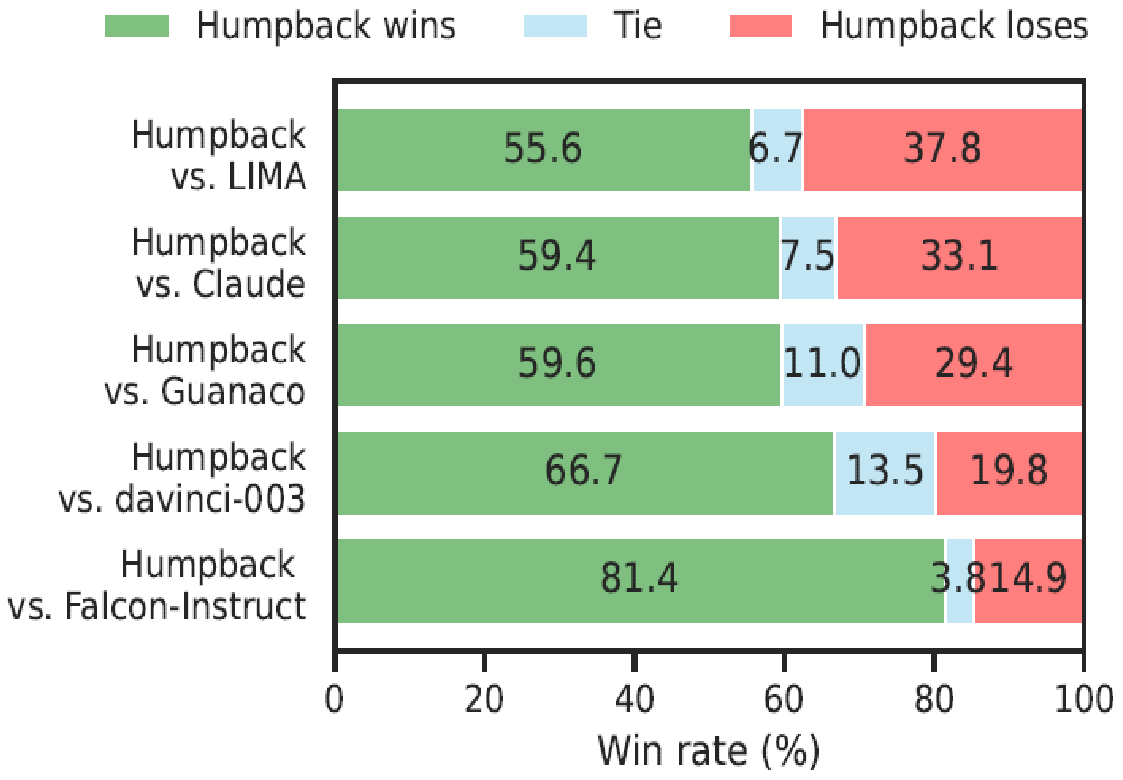


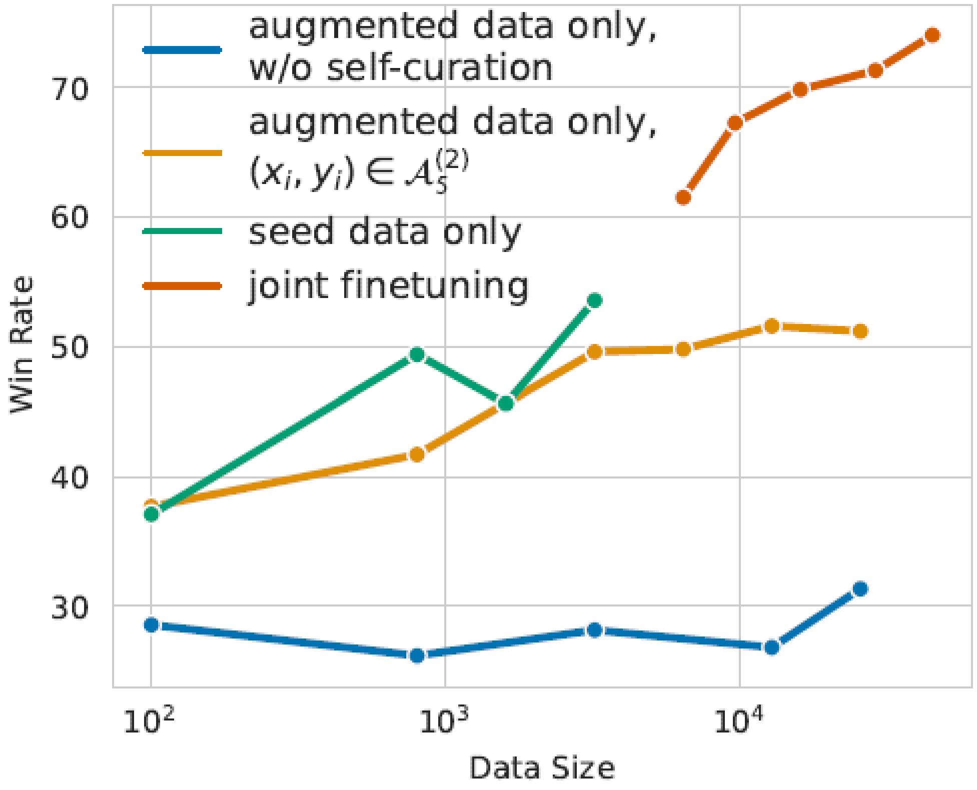
Table 3: Results on the Alpaca leaderboard (win rate over text-davinci-003 evaluated by GPT-4). Humpback outperforms other non-distilled models by a wide margin with efficient data scaling beyond human annotated data.

		Annotated Examples	Total Examples	Win Rate %
Non-distilled	Humpback 33B	3k	45k	**79.84**
	OASST RLHF 33B	161k	161k	66.52
	Guanaco 33B	9k	9k	65.96
	OASST SFT 33B	161k	161k	54.97
Non-distilled	Humpback 65B	3k	45k	**83.71**
	Guanaco 65B	9k	9k	71.80
	LIMA 65B	1k	1k	62.70
Non-distilled	Humpback 70B	3k	45k	87.94
	LLaMa2 Chat 70B	1.4m	5.7m	**92.66**
Distilled	Vicuna 33B	140k	140k	**88.99**
	WizardLLM 13B	190k	190k	86.32
	airoboros 65B	17k	17k	73.91
	Falcon Instruct 40B	100k	100k	45.71
Proprietary	GPT-4			**95.28**
	Claude 2			91.36
	ChatGPT			89.37
	Claude			88.39



Experiment

- Ablations



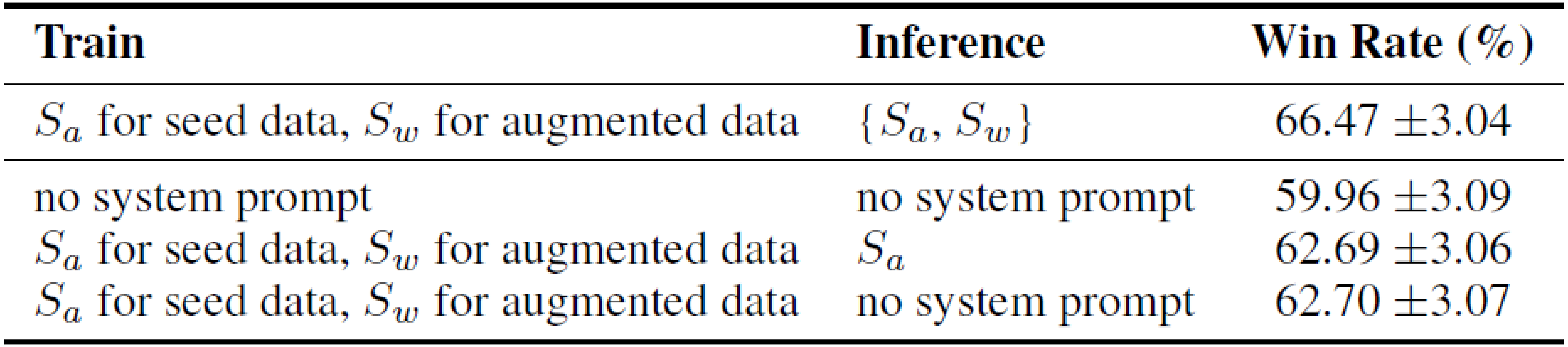
1.without self-curation, the quality of instruction following does not improve, or even deteriorates with more data.(bad data)

2. training on the higher quality self-curated data brings improvements as training set size increases.

3. This indicates that seed data and augmented data are complimentary,where the seed data has the same distribution as the target domain (AI assistant response), while the data from web corpus may enlarge the diversity of the instructions and outputs.

Experiment

- System prompts



Train	Inference	Win Rate (%)
S_a for seed data, S_w for augmented data	$\{S_a, S_w\}$	66.47 ±3.04
no system prompt	no system prompt	59.96 ±3.09
S_a for seed data, S_w for augmented data	S_a	62.69 ±3.06
S_a for seed data, S_w for augmented data	no system prompt	62.70 ±3.07

We found adding system prompts to distinguish augmented data from seed data is helpful.

Suri: Multi-constraint Instruction Following for Long-form Text Generation

Chau Minh Pham Simeng Sun[*] Mohit Iyyer

University of Massachusetts Amherst

{ctpham,simengsun,miyyer}@cs.umass.edu

Motivation

1. Challenges in Long-Text Generation:

- Existing Large Language Models (LLMs) perform well when generating short texts, but often struggle to maintain coherence and fidelity to instructions when generating long-form texts. Long-form texts, such as technical reports or novel chapters, typically require the model to understand and follow more complex instructions.

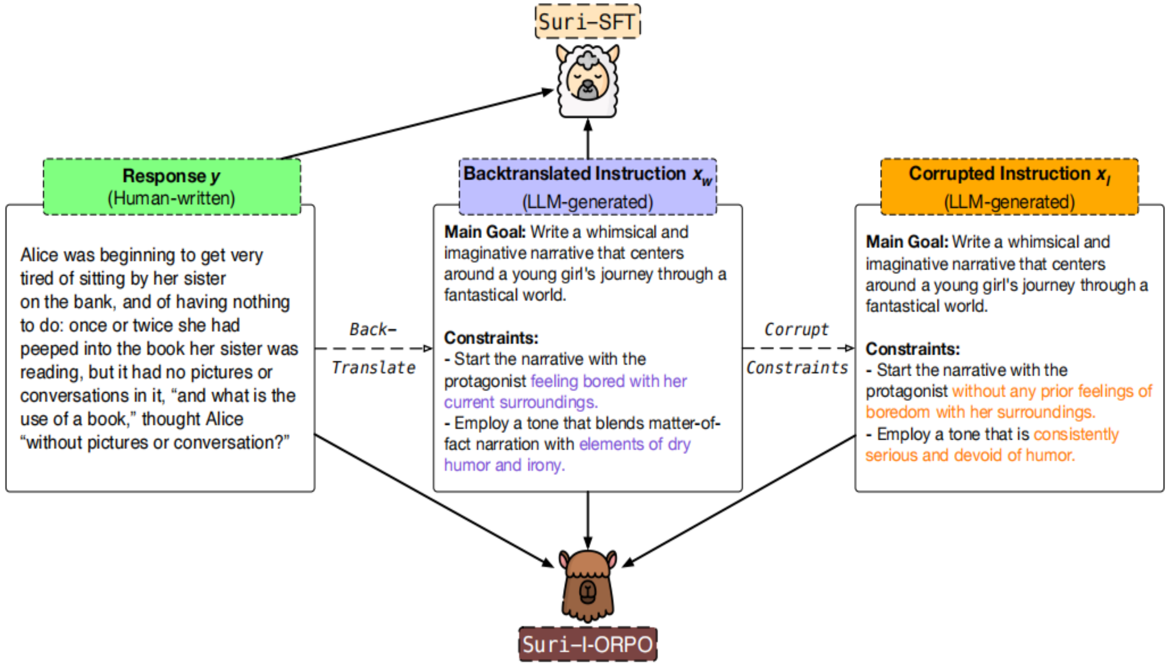
2. Infeasibility of Preference Adjustment Algorithms:

- In the context of long-form text generation, obtaining human preference judgments for long texts is costly and impractical. Therefore, traditional preference-based adjustment algorithms, such as Direct Preference Optimization (DPO), become infeasible in this setting.

Method

1. The Suri Dataset (x_{w}, x_{l}, y)

- Sample long-form human-written texts from existing datasets (such as novel chapters) as gold responses(y).
- Use llm to perform instruction backtranslation on these texts to generate instructions (backtranslated instruction x_w) that could have been followed to create the text. These instructions include a main goal and approximately 10 constraints.
- Make minimal edits to each constraint in x_w to generate corrupted instruction x_l that violate the constraints.



Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, “and what is the use of a book,” thought Alice “without pictures or conversation?”

Main Goal: Write a whimsical and imaginative narrative that centers around a young girl's journey through a fantastical world.

Constraints:
- Start the narrative with the protagonist feeling bored with her current surroundings.
- Employ a tone that blends matter-of-fact narration with elements of dry humor and irony.

Main Goal: Write a whimsical and imaginative narrative that centers around a young girl's journey through a fantastical world.

Constraints:
- Start the narrative with the protagonist without any prior feelings of boredom with her surroundings.
- Employ a tone that is consistently serious and devoid of humor.

Method

2. Aligning language models with Suri

- Suri-I-ORPO
- The original algorithm learns from preference judgments, requiring access to chosen and rejected responses in the (x, y_w, y_l) format. Since our dataset contains gold and corrupted instructions instead, we modify ORPO so that the algorithm accepts (x_w, x_l, y).

$$\mathcal{L}_{\text{I-ORPO}} = \mathbb{E}_{(x_w, x_l, y)} \left[\mathcal{L}_{\text{SFT}} + \lambda \cdot \mathcal{L}_{\text{I-OR}} \right]$$

where

$$\mathcal{L}_{\text{I-OR}} = -\log \sigma \left(\log \frac{\textbf{odds}_\theta(y|x_w)}{\textbf{odds}_\theta(y|x_l)} \right)$$

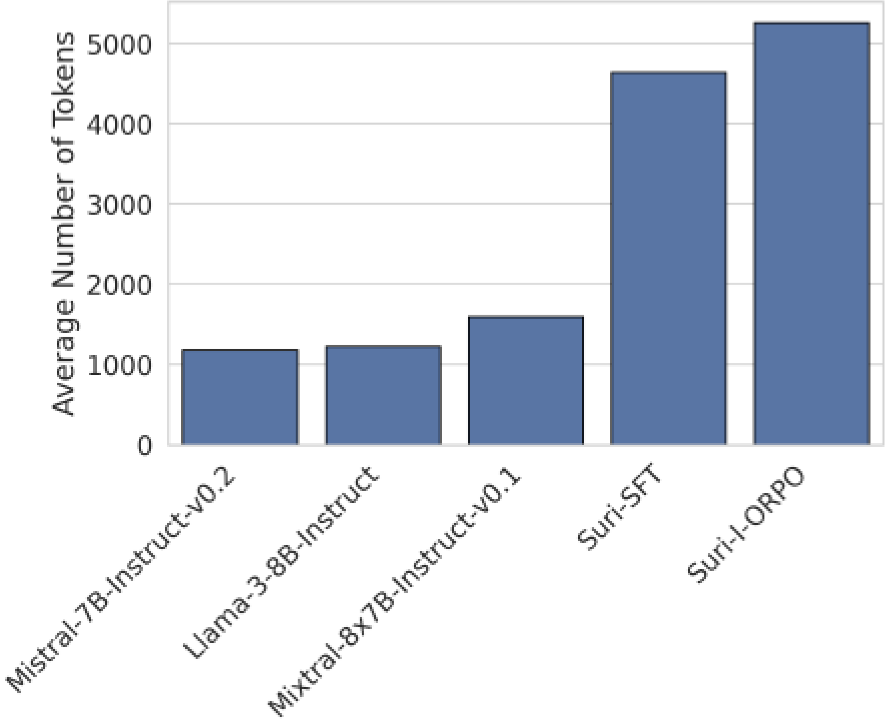
$$\textbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

LI-ORPO makes the model more inclined to generate responses that meet the requirements when faced with the correct instructions, and less inclined to generate the same responses when faced with corrupted instructions.

Experiment

1.Suri-I-ORPO and Suri-SFT generate substantially longer text.

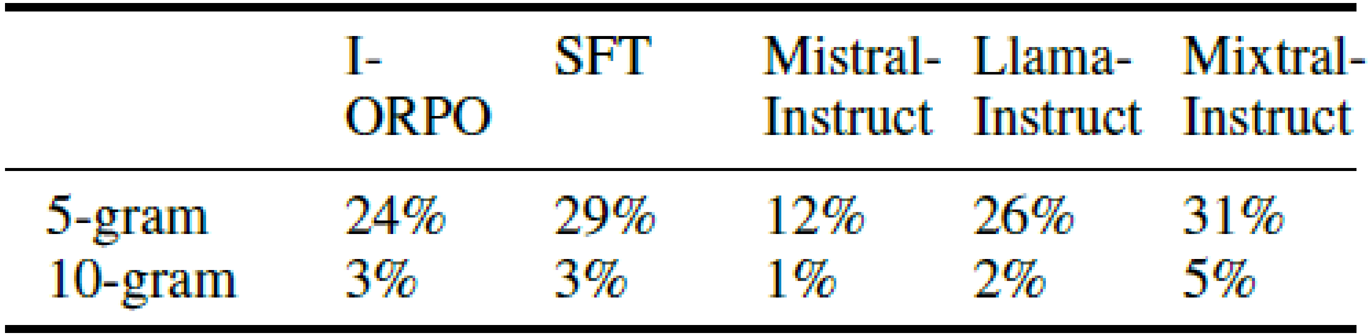
- Proprietary models like GPT-4 and Claude are excluded due to their maximum generation output limit, whereas open-weight models allow for outputs of arbitrary maximum length.
- Base model:Mistral-7B-Instruct-v0.2



Experiment

2. Suri-I-ORPO and Suri-SFT do not degenerate into repetitions at longer sequences

- Degenerate into repetitions: In long-text generation tasks, a common issue is that models may produce **repetitive or redundant** content.



	I-ORPO	SFT	Mistral-Instruct	Llama-Instruct	Mixtral-Instruct
5-gram	24%	29%	12%	26%	31%
10-gram	3%	3%	1%	2%	5%

Experiment

3. I-ORPO improves **ranking accuracy**

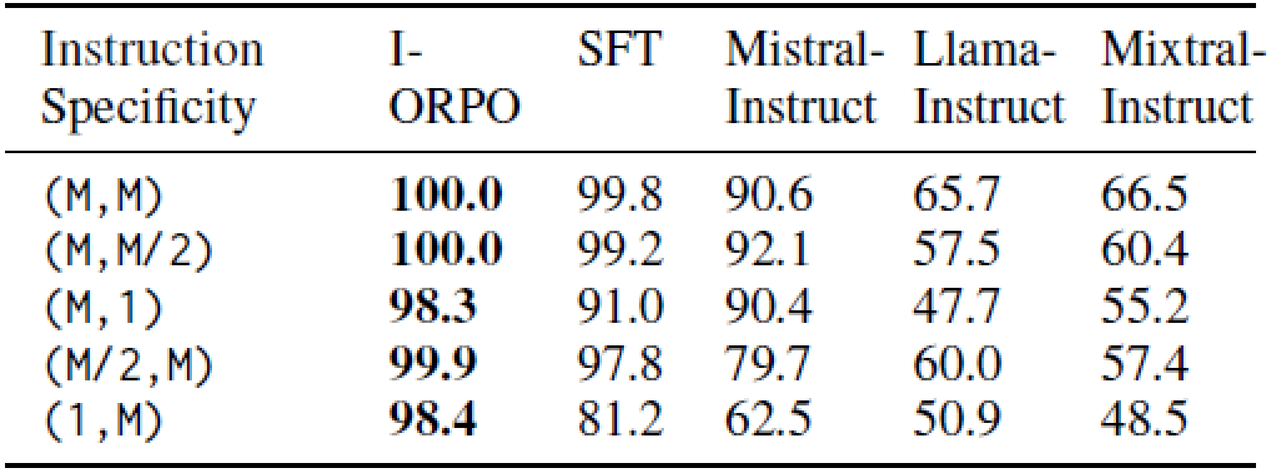
- Ranking accuracy :the extent to which the model can identify and prioritize the correct instructions to generate responses./The effectiveness validation of the I-ORPO method.

- We calculate the **sum of token log probabilities** in the response given the previous tokens, denoted by logps(y|x), and determine accuracy based on the **proportion of times** when logps(y|x_w) > logps(y|x_l).

Experiment

3. I-ORPO improves **ranking accuracy**

- Across five different settings, which are defined by the number of all constraints included (M constraints in total) and the number of those included constraints that are corrupted: (M,M), (M,M/2), (M,1), (M/2,M),(1,M).



Instruction Specificity	I-ORPO	SFT	Mistral-Instruct	Llama-Instruct	Mixtral-Instruct
(M,M)	**100.0**	99.8	90.6	65.7	66.5
(M,M/2)	**100.0**	99.2	92.1	57.5	60.4
(M,1)	**98.3**	91.0	90.4	47.7	55.2
(M/2,M)	**99.9**	97.8	79.7	60.0	57.4
(1,M)	**98.4**	81.2	62.5	50.9	48.5

Conifer: Improving Complex Constrained Instruction-Following Ability of Large Language Models

Haoran Sun*, Lixin Liu*, Junjie Li, Fengyu Wang,
Baohua Dong, Ran Lin, Ruohui Huang
Alibaba Group, Beijing, China
`sunhaoran0402@gmail.com, llx271805@alibaba-inc.com`

Motivation

1. The Importance of Complex Constraints

- In real-world applications, users' instructions to LLMs often contain complex constraints,such as specific formats, content limitations, or style preferences (including both soft and hard constraints).

2. The Need for a Progressive Learning Approach (Curriculum Learning)

- Given the difficulty of complex instructions, traditional training methods may fall short in enabling the model to learn effectively. Therefore, a new training approach is needed to help the model progressively develop skills for handling complex instructions, learning step-by-step from easier to more challenging tasks, and improving through process feedback.

Method

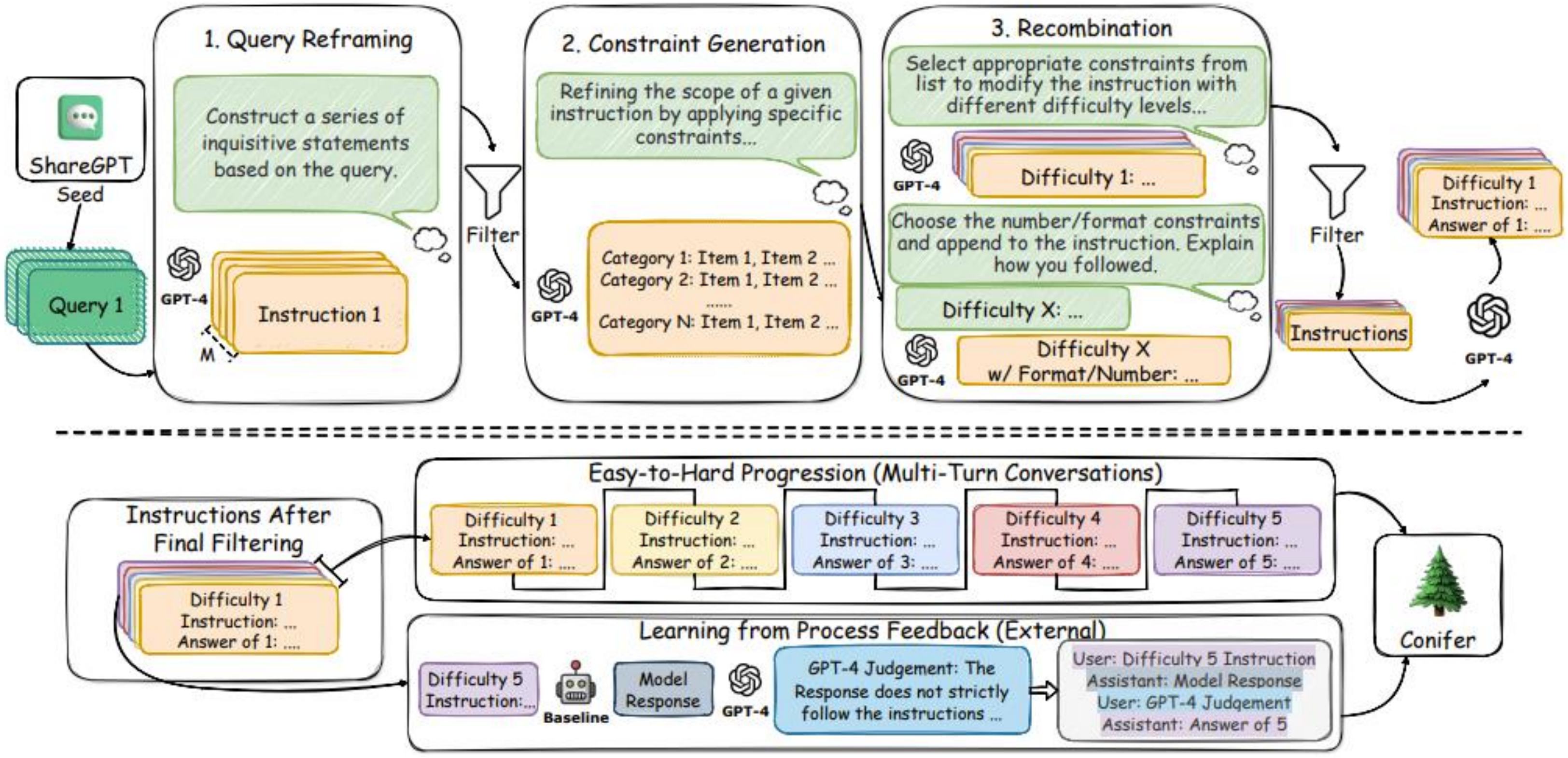

ShareGPT
Seed
Query 1
1. Query Reframing
Construct a series of inquisitive statements based on the query.
GPT-4
Instruction 1
M
Filter
2. Constraint Generation
Refining the scope of a given instruction by applying specific constraints...
GPT-4
Category 1: Item 1, Item 2 ...
Category 2: Item 1, Item 2 ...
......
Category N: Item 1, Item 2 ...
3. Recombination
Select appropriate constraints from list to modify the instruction with different difficulty levels...
GPT-4
Difficulty 1: ...
Choose the number/format constraints and append to the instruction. Explain how you followed.
Difficulty X: ...
GPT-4
Difficulty X w/ Format/Number: ...
Filter
Instructions
GPT-4
Difficulty 1
Instruction: ...
Answer of 1: ....
Instructions After Final Filtering
Difficulty 1
Instruction: ...
Answer of 1: ....
Easy-to-Hard Progression (Multi-Turn Conversations)
Difficulty 1
Instruction: ...
Answer of 1: ....
Difficulty 2
Instruction: ...
Answer of 2: ....
Difficulty 3
Instruction: ...
Answer of 3: ....
Difficulty 4
Instruction: ...
Answer of 4: ....
Difficulty 5
Instruction: ...
Answer of 5: ....
Learning from Process Feedback (External)
Difficulty 5
Instruction:...
Baseline
Model Response
GPT-4
GPT-4 Judgement: The Response does not strictly follow the instructions ...
User: Difficulty 5 Instruction
Assistant: Model Response
User: GPT-4 Judgement
Assistant: Answer of 5
Conifer

Figure 2: Paradigm of the production of the proposed Conifer dataset. The upper portion depicts the instruction collection phase (Section 3.1), while the lower portion outlines the progressive learning scheme (Section 3.2).

Method

The Conifer Dataset

- GPT-4 consistently struggles in generating instructions that contain multiple complex constraints. we have decomposed this challenging task into smaller, more manageable tasks for GPT-4.

1.Query Reframing

(1) Reformulate the original user query to provide different perspectives and expressions while keeping the core entities of the query unchanged.

(2) Use the GPT-4 model to reframe each query into at least three diverse forms.

Method

- 2. Constraint Generation

（1）GPT-4 struggles to produce appropriate constraints consistently.

（2）Consequently, we have adopted in-context learning techniques, providing manually crafted examples that pair instructions with their corresponding constraints.

（3）These constraints are categorized into two levels: broader categories and specific items. For instance, in naming-related instructions, ‘Cultural Background’ could serve as a category, with ‘Chinese’, ‘Continental’ as example items within this category.

Method

- 3. Recombination

a) GPT-4 is directed to select specific and suitable constraints from the generated constraint list and incorporate them into the instruction.

b) We instruct GPT-4 to develop instructions with varying 5 levels of difficulty. Each instruction of increasing difficulty should integrate 1-2 categories and 2-3 items, thus presenting a more complex challenge than its simpler counterpart.

c) GPT-4 seems to struggle with incorporating format or numerical constraints with sentences.

d) We randomly select 1,000 instructions after the recombination and prompt GPT-4 to augment them with additional format or numerical constraints. These constraints are synthesized by GPT-4 using three seed constraints.

Method

4. Two-Stage Filtering

a) Post-Query Reframing Filtering

Objective: The removal of instructions that lack necessary context for meaningful responses.

b) Post-Recombination Filtering

Objective: Identify and resolve conflicts within instructions that may result from the recombination of various constraints.

Method

Progressive Learning Scheme

1.Easy-to-Hard Progression

- A learning strategy inspired by curriculum learning, starting with simple examples and gradually progressing to more complex ones.
- We aggregate multi-level instructions and answers generated from the same seed instruction into a single sample using the multi-turn conversational format. The sequence is organized to introduce the simpler tasks first, which then progressively lead towards more challenging ones.

Method

2.Learning from Process Feedback

a) Internally, when prompted to respond to instructions that contain format and numerical constraints, GPT-4 is instructed to illustrate how it adheres to the specified constraints explicitly within its responses.The instructions of this part are randomly sampled from all the instructions of our Conifer.

b) Externally,GPT-4 is tasked to identify constraints where the baseline modle fails to adhere in the most challenging constraints, and this judgement is employed to construct a new multi-turn conversation which take the form of: {Difficult Instruction, Model Response, GPT-4 Judgement, GPT-4 Response}. The instructions of this part are selectively sampled from the highest difficulty level, level 5, within the Conifer dataset.

Method

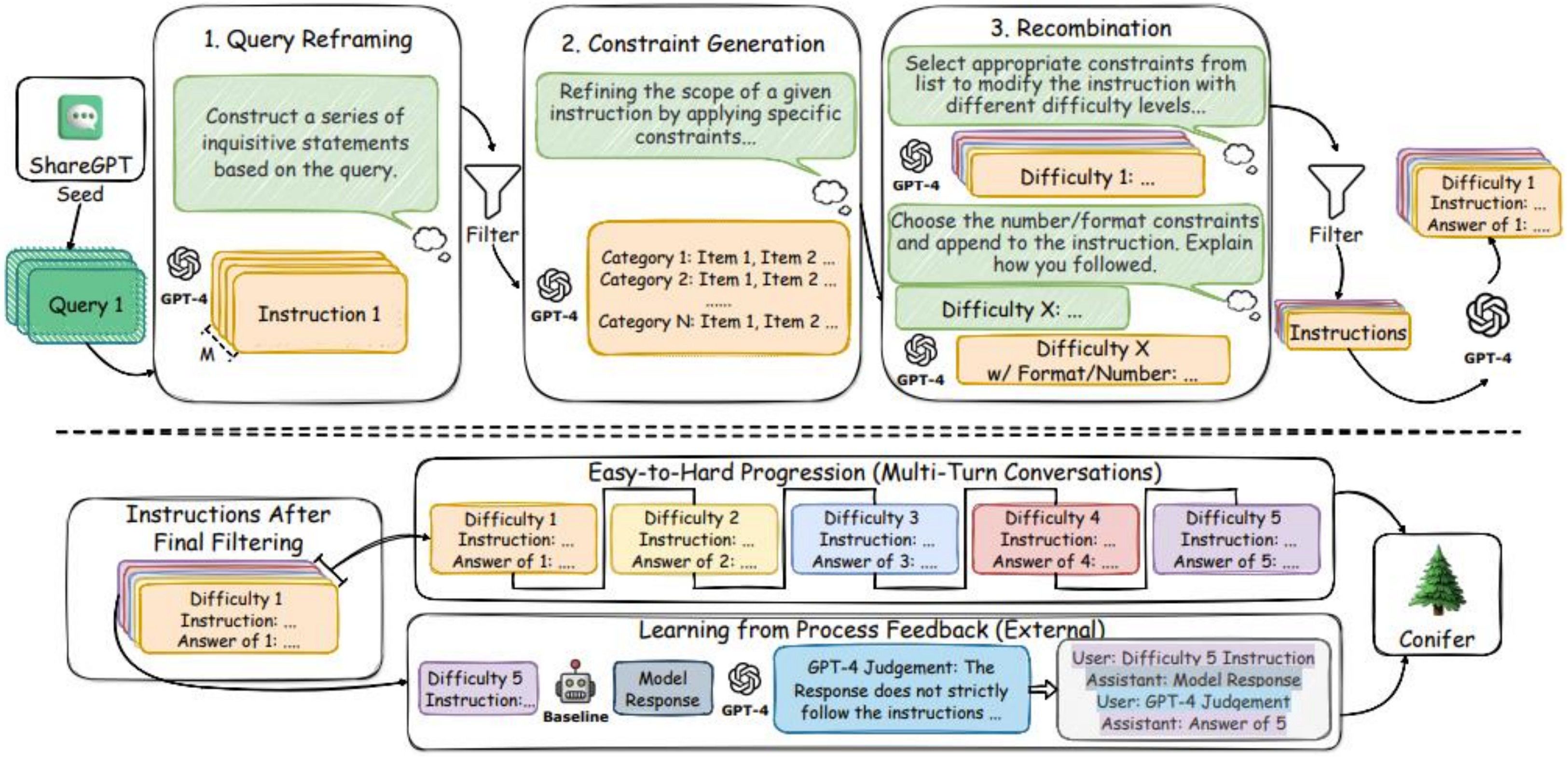

1. Query Reframing
ShareGPT
Seed
Query 1
Construct a series of inquisitive statements based on the query.
GPT-4
Instruction 1
M
Filter
2. Constraint Generation
Refining the scope of a given instruction by applying specific constraints...
GPT-4
Category 1: Item 1, Item 2 ...
Category 2: Item 1, Item 2 ...
......
Category N: Item 1, Item 2 ...
3. Recombination
Select appropriate constraints from list to modify the instruction with different difficulty levels...
GPT-4
Difficulty 1: ...
Choose the number/format constraints and append to the instruction. Explain how you followed.
Difficulty X: ...
GPT-4
Difficulty X w/ Format/Number: ...
Filter
Instructions
GPT-4
Difficulty 1
Instruction: ...
Answer of 1: ....
Instructions After Final Filtering
Difficulty 1
Instruction: ...
Answer of 1: ....
Easy-to-Hard Progression (Multi-Turn Conversations)
Difficulty 1
Instruction: ...
Answer of 1: ....
Difficulty 2
Instruction: ...
Answer of 2: ....
Difficulty 3
Instruction: ...
Answer of 3: ....
Difficulty 4
Instruction: ...
Answer of 4: ....
Difficulty 5
Instruction: ...
Answer of 5: ....
Learning from Process Feedback (External)
Difficulty 5
Instruction:...
Baseline
Model Response
GPT-4
GPT-4 Judgement: The Response does not strictly follow the instructions ...
User: Difficulty 5 Instruction
Assistant: Model Response
User: GPT-4 Judgement
Assistant: Answer of 5
Conifer

Figure 2: Paradigm of the production of the proposed Conifer dataset. The upper portion depicts the instruction collection phase (Section 3.1), while the lower portion outlines the progressive learning scheme (Section 3.2).

Experiment

1. Models and Datasets

- Models: Mistral-7B and LLaMA-2-13B
- Datasets: merge the collected 13k Conifer dataset with the initial 53k ShareGPT to form the final 66k SFT dataset

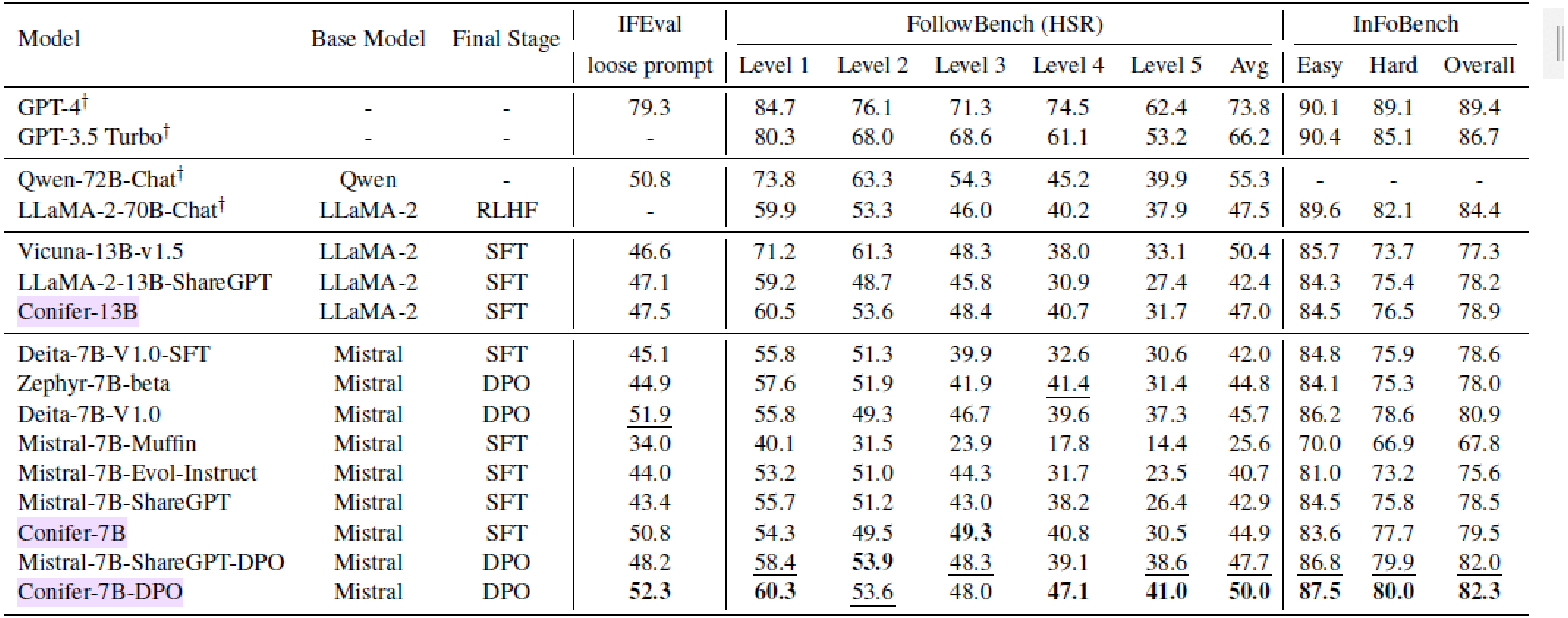
2.Evaluation

- instruction-following benchmarks:IFEval,FollowBench,InFoBench
- instruction-following benchmarks on LLMs’ general ability to align with human preferences:

(1)AlpacaEval:an LLM-based automatic evaluation by GPT-4 Turbo. The score is the win-rate against the reference model (GPT-4 Turbo) on the dataset.

(2)MT-Bench: a multi-turn question set. The responses are rated by GPT-4 on scores scaled from 1-10.

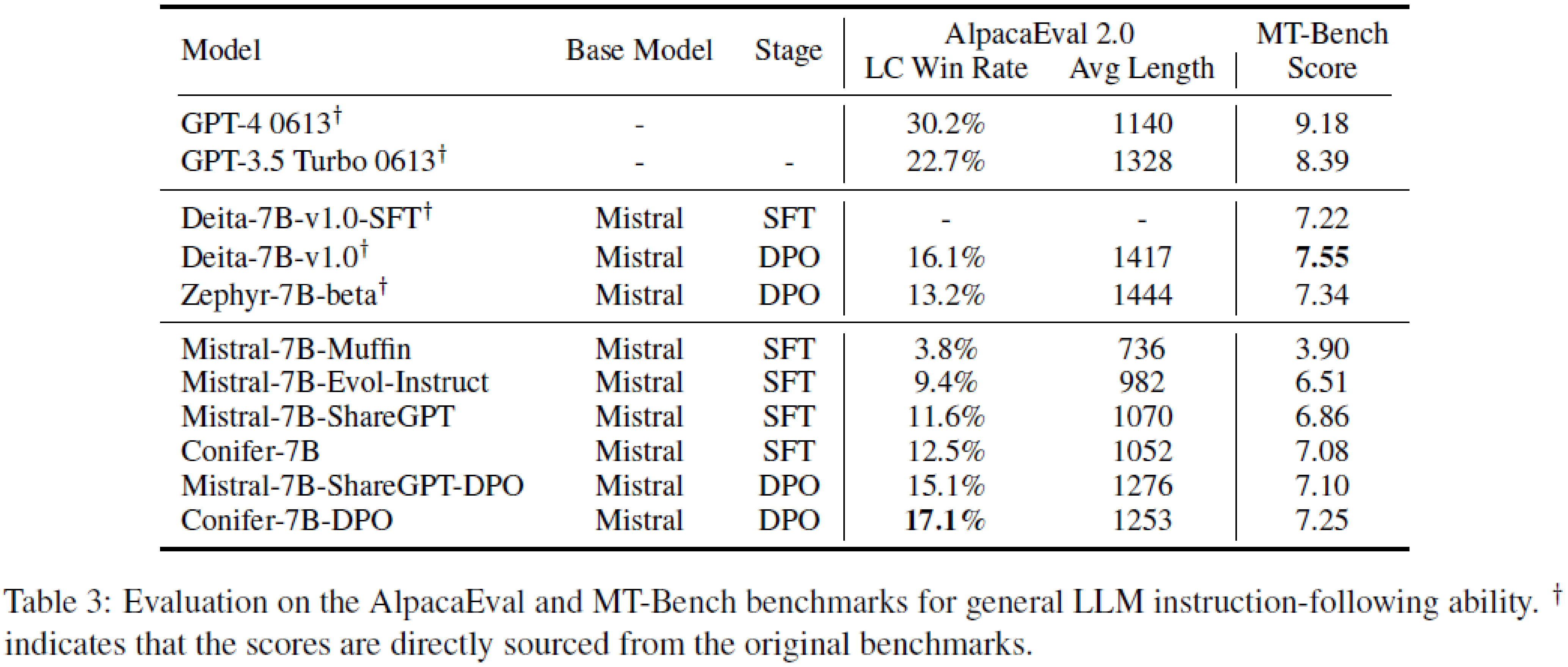
Experiment(in domain)



Model	Base Model	Final Stage	IFEval	FollowBench (HSR)						InFoBench		
			loose prompt	Level 1	Level 2	Level 3	Level 4	Level 5	Avg	Easy	Hard	Overall
GPT-4†	-	-	79.3	84.7	76.1	71.3	74.5	62.4	73.8	90.1	89.1	89.4
GPT-3.5 Turbo†	-	-	-	80.3	68.0	68.6	61.1	53.2	66.2	90.4	85.1	86.7
Qwen-72B-Chat†	Qwen	-	50.8	73.8	63.3	54.3	45.2	39.9	55.3	-	-	-
LLaMA-2-70B-Chat†	LLaMA-2	RLHF	-	59.9	53.3	46.0	40.2	37.9	47.5	89.6	82.1	84.4
Vicuna-13B-v1.5	LLaMA-2	SFT	46.6	71.2	61.3	48.3	38.0	33.1	50.4	85.7	73.7	77.3
LLaMA-2-13B-ShareGPT	LLaMA-2	SFT	47.1	59.2	48.7	45.8	30.9	27.4	42.4	84.3	75.4	78.2
Conifer-13B	LLaMA-2	SFT	47.5	60.5	53.6	48.4	40.7	31.7	47.0	84.5	76.5	78.9
Deita-7B-V1.0-SFT	Mistral	SFT	45.1	55.8	51.3	39.9	32.6	30.6	42.0	84.8	75.9	78.6
Zephyr-7B-beta	Mistral	DPO	44.9	57.6	51.9	41.9	41.4	31.4	44.8	84.1	75.3	78.0
Deita-7B-V1.0	Mistral	DPO	51.9	55.8	49.3	46.7	39.6	37.3	45.7	86.2	78.6	80.9
Mistral-7B-Muffin	Mistral	SFT	34.0	40.1	31.5	23.9	17.8	14.4	25.6	70.0	66.9	67.8
Mistral-7B-Evol-Instruct	Mistral	SFT	44.0	53.2	51.0	44.3	31.7	23.5	40.7	81.0	73.2	75.6
Mistral-7B-ShareGPT	Mistral	SFT	43.4	55.7	51.2	43.0	38.2	26.4	42.9	84.5	75.8	78.5
Conifer-7B	Mistral	SFT	50.8	54.3	49.5	**49.3**	40.8	30.5	44.9	83.6	77.7	79.5
Mistral-7B-ShareGPT-DPO	Mistral	DPO	48.2	58.4	**53.9**	48.3	39.1	38.6	47.7	86.8	79.9	82.0
Conifer-7B-DPO	Mistral	DPO	**52.3**	**60.3**	53.6	48.0	**47.1**	**41.0**	**50.0**	**87.5**	**80.0**	**82.3**

Table 2: Main results on three instruction-following benchmarks: IFEval, FollowBench, and InFoBench. We use boldface for the best results and underline for the second-best results among the 7B models. † indicates that the results are directly sourced from the original benchmarks.

Experiment(out of domain)

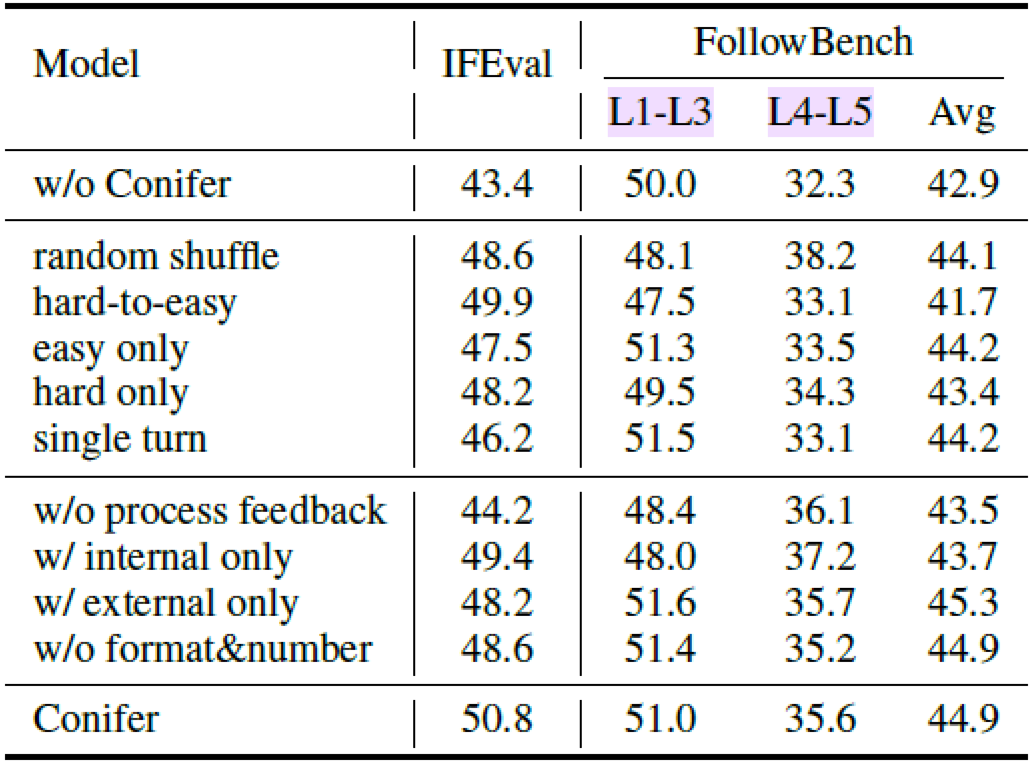


Model	Base Model	Stage	AlpacaEval 2.0		MT-Bench
			LC Win Rate	Avg Length	Score
GPT-4 0613[†]	-		30.2%	1140	9.18
GPT-3.5 Turbo 0613[†]	-	-	22.7%	1328	8.39
Deita-7B-v1.0-SFT[†]	Mistral	SFT	-	-	7.22
Deita-7B-v1.0[†]	Mistral	DPO	16.1%	1417	**7.55**
Zephyr-7B-beta[†]	Mistral	DPO	13.2%	1444	7.34
Mistral-7B-Muffin	Mistral	SFT	3.8%	736	3.90
Mistral-7B-Evol-Instruct	Mistral	SFT	9.4%	982	6.51
Mistral-7B-ShareGPT	Mistral	SFT	11.6%	1070	6.86
Conifer-7B	Mistral	SFT	12.5%	1052	7.08
Mistral-7B-ShareGPT-DPO	Mistral	DPO	15.1%	1276	7.10
Conifer-7B-DPO	Mistral	DPO	**17.1%**	1253	7.25

Table 3: Evaluation on the AlpacaEval and MT-Bench benchmarks for general LLM instruction-following ability. [†] indicates that the scores are directly sourced from the original benchmarks.

Experiment

3. Ablation Studies



Model	IFEval	FollowBench		
		L1-L3	L4-L5	Avg
w/o Conifer	43.4	50.0	32.3	42.9
random shuffle	48.6	48.1	38.2	44.1
hard-to-easy	49.9	47.5	33.1	41.7
easy only	47.5	51.3	33.5	44.2
hard only	48.2	49.5	34.3	43.4
single turn	46.2	51.5	33.1	44.2
w/o process feedback	44.2	48.4	36.1	43.5
w/ internal only	49.4	48.0	37.2	43.7
w/ external only	48.2	51.6	35.7	45.3
w/o format&number	48.6	51.4	35.2	44.9
Conifer	50.8	51.0	35.6	44.9

Table 4: Ablation studies results when applying different progressive learning scheme in Section 3.2.

1)varying the sequence of difficulty within the easyto-hard progression, such as through random shuffling or reversing the order to hard-to-easy, results in reduced overall performance. In particular, randomizing the sequence results in a notable decrease in performance.

2) Excluding certain difficulty levels, such as including only easy or only hard instructions, leads to decreased performance on the L4-L5 or L1-L3 of FollowBench, respectively.

3) Note that relying solely on internal feedback results in a notable decrease in FollowBench, while the IFEval score slightly improves, which is in line with expectations since internal feedback only addresses format and numerical constraints where IFEval is particularly focused.

Experiment

4. Data Contamination

- When a model is trained on the training set, an improvement in performance is often observed because the model is able to "memorize" specific samples and answers rather than truly learning the ability to solve the problems.

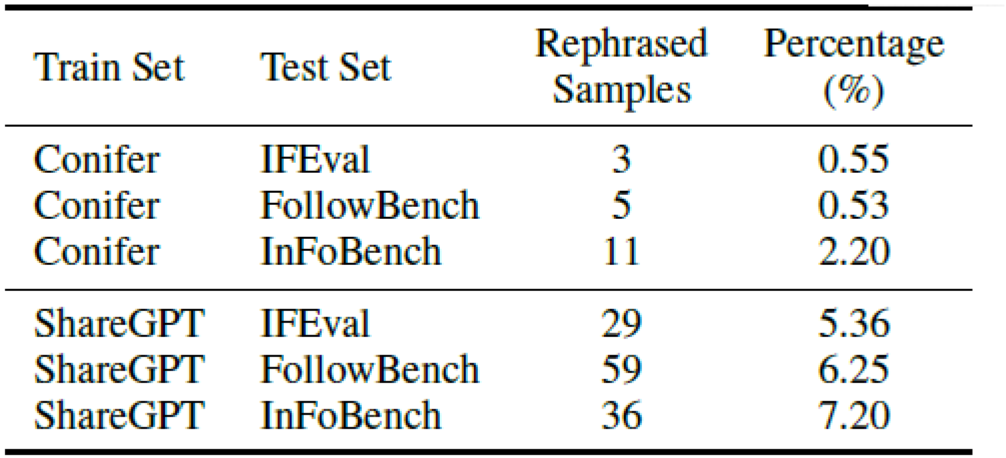
- To assess whether Conifer dataset exhibits data contamination with three instruction-following benchmarks, we conduct analysis using:

  (1) cosine similarity between Conifer and test samples

  (2) using GPT-4 to detect rephrased samples across training and testing sets

- Rephrased samples refer to a question or instruction in the test set that is expressed in a slightly different way but is the same as or similar to a sample in the training set.

Experiment



Train Set	Test Set	Rephrased Samples	Percentage (%)
Conifer	IFEval	3	0.55
Conifer	FollowBench	5	0.53
Conifer	InFoBench	11	2.20
ShareGPT	IFEval	29	5.36
ShareGPT	FollowBench	59	6.25
ShareGPT	InFoBench	36	7.20

Table 5: Rephrased samples assessment between Conifer and ShareGPT datasets with the benchmarks, as evaluated by GPT-4 Turbo. The values indicate the proportion of rephrased samples within each test set.

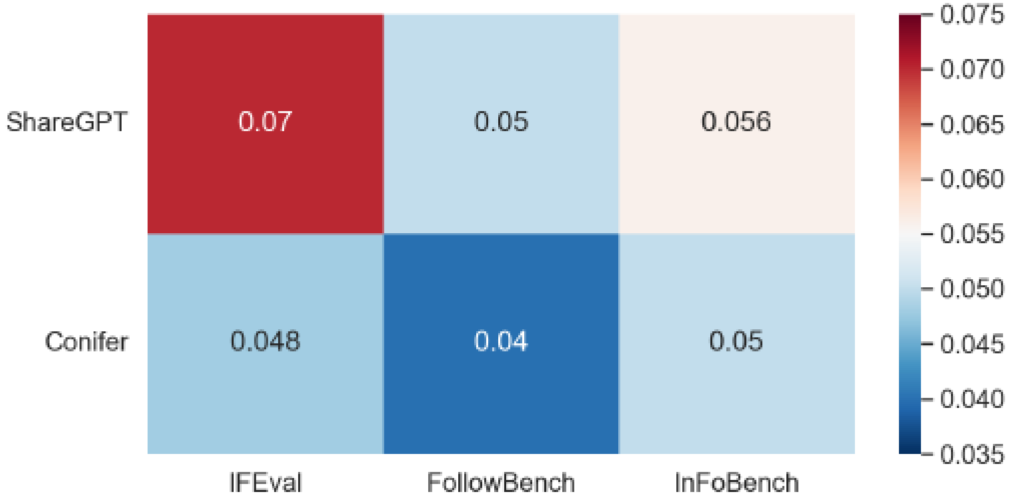


Figure 3: Cosine similarities between sentences from training and testing sets. Values near 0 indicate little overlap and no data contamination.

Summary

How to construct multi-constraint dataset

- Instruction backtranslation
- Instruction backtranslation + Corrupting Instructions
- Easy-to-Hard

How to use the dataset to train model

- I-ORPO
- Progressive Learning Approach (Curriculum Learning)