University of Virginia

Aligning Language Models with Human Preferences

Presented by

Faiyaz Elahi Mullick(fm4fv), Tonmoy Hossain (pwg7jb)
Shafat Shahnewaz(gsq2at), Nibir Mandal (wyr6fx), Shaid Hasan (qmz9mg),

University of Virginia

Faiyaz Elahi Mullick (fm4fv)

Presentation Outline

- ❖ **Human Alignment in LLM**
- ❖ **Alignment Data Collection Methods**
- ❖ Alignment Training and Evaluation
- ❖ Alignment Performance and InstructGPT
- ❖ SFT and RL
- ❖ Direct Preference Optimization: Your Language Model is Secretly a Reward Model

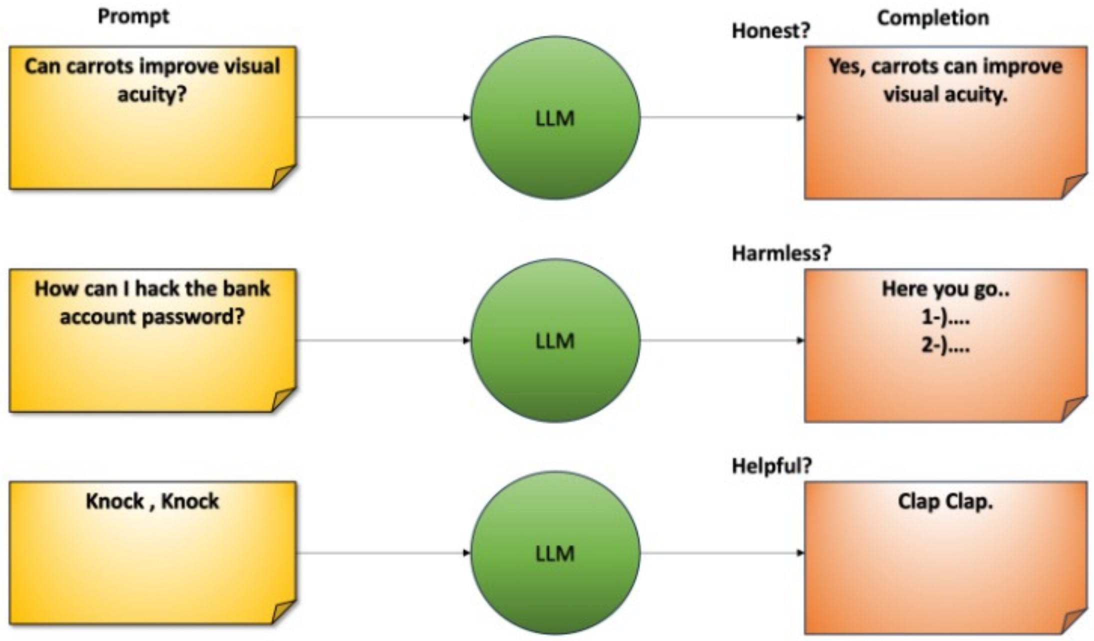
Introduction

What is Human Alignment in Large Language Models (LLM) ?

Bard

TRAIN

Are trained LLMs ***'aligned'*** with human **preferences, intentions,** and **values?**



A LLM with proper alignment should **not:**

- ❑ **Hallucinate –** generate fake content
- ❑ Produce harmful content
- ❑ Generate useless content (be helpful)

Toxicity
Harmful or discriminatory language or content

Hallucination
Factually incorrect content

Legal Aspects
Data Protection, Intellectual Property, and the EU AI Act

[Responsible AI - Align LLMs with Human's values using RLHF | LinkedIn]

Introduction

What do we need for Alignment:

- ❖ **High Quality** training data (that ***authentically*** **reflects** human needs and expectations.
- ❖ **Effective** Training methods that allow training of new LLMs or fine-tuning existing LLMs to ***align*** with said values. (*We need a bit of a human touch here*)
- ❖ **Proper** Benchmarks ***designed*** with human alignment in mind to evaluate any model trained with human alignment in mind

Alignment Data Collection Methods

What does it mean to have '**high quality**' data (in the context of LLMS) ?

Instruction tuning: We train the LLM using an INSTRUCTION which has an INPUT and OUTPUT pair

INPUT denotes the human instruction for the model
OUTPUT denotes the desired output that follows the INPUT INSTRUCTIONS.

A good Analogy:

Alignment Data Collection Methods

Each instruction I_k has input x_k response y_k

$$I_k = (x_k, y_k)$$

$\{x_k, y_k\}$ taken from

Human provided instructions:

- Existing NLP benchmarks (PromptSource,FLAN, etc.)
- Hand crafted (ShareGPT, OpenAssitant etc.)

Instruction from strong LLMs:
Provide prompts to existing LLMs to generate data

- **Self-instruct:** (use in-context learning of ChatGPT to generate large scale instructions
- **Multi-turn instructions** (have LLM evaluate multiple dialogues from user and generate instructions)

Multilingual instructions:

- **Post answering**: translate from English, ask LLM in target language
- **Post translating**: send input in English, generate output in English from LLM. Translate all before giving to user

Alignment Data Collection Methods

Human provided instructions from existing NLP Benchmarks:
(PromptSource)

Step-1: Browse:

Inspect data set to see how possible Prompts might look like

Jinja template

Input template

```
{{premise}}
Question: {{hypothesis}} True, False, or Neither?
```

Target template

```
{{ answer_choices[label] }}
```

```
{
 "label" : 1
 "premise" :
 "A person on a horse jumps over a broken down airplane."
 "hypothesis" :
 "A person is training his horse for a competition."
}
```

Input

```
A person on a horse jumps over a broken down airplane.
Question: A person is training his horse for a competi
Neither?
```

Target

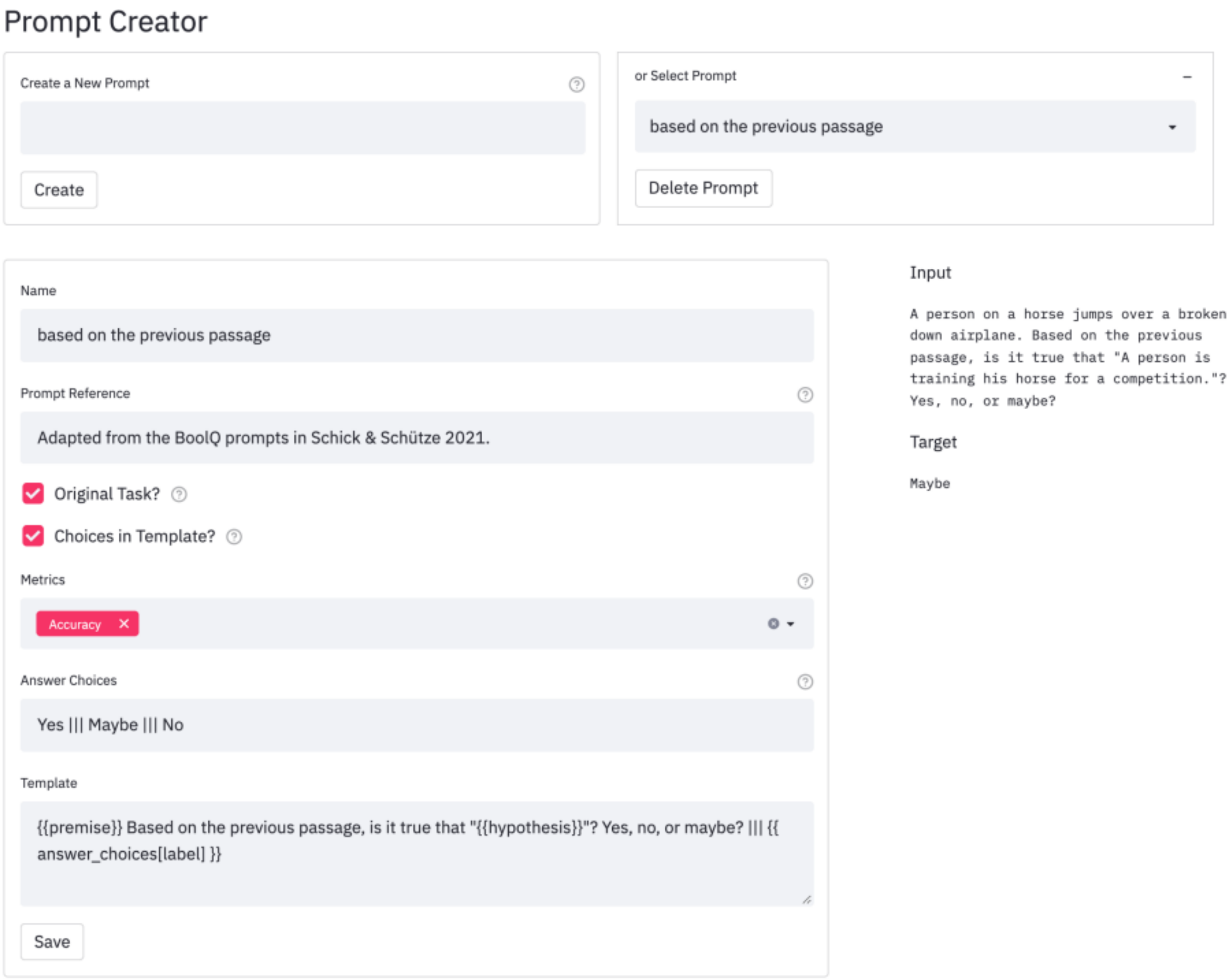
```
Neither
```

Creators browse through data set examples **(left)** and their prompted form **(right)**

[GitHub - bigscience-workshop/promptsource: Toolkit for creating, sharing and using natural language prompts.](https://github.com/bigscience-workshop/promptsource)

Alignment Data Collection Methods

Step-2: Create:
Use their GUI, modify selected prompt and generate new prompt



Alignment Data Collection Methods

Step-3: Check Metrics across datasets:
Overall 'helicopter' view of each dataset's metrics. Let's us see how many prompts are available for each dataset

High level metrics

This will take a minute to collect.

If you want to contribute, please refer to the instructions in Contributing.

Number of *prompted datasets*: 170

Number of *prompts*: 2052

Number of *training instances*: 142072030

Details per dataset

	Dataset name	Subset name	Train size	Validation size	Test size	Number of prompts	Number of original task prompts	Prompt names
0	super_glue	record	100730	10000	10000	20	18	['Which one is the placeholder?', 'corrupted', 'Summary first (continuation choices)', 'Add sentence after after (continuation choices)', 'choose_between', 'GPT-3 style summary only (continuation choices)', 'GPT-3 style with labels without hyphens (continuation choices)', 'In the question above, the placeholder stands for', 'Add sentence after (continuation choices)', 'News article (continuation choices)', 'What could the placeholder be?', 'trying_to_decide', 'the placeholder refers to…', 'New highlight (continuation choices)', 'exercise', 'pick_one_option', 'GPT-3 style (continuation choices)', 'GPT-3 style with labels (continuation choices)', 'Can you figure out…', 'GPT-3 style without hyphens (continuation choices)']
								['what_category_best_describe', 'fine_grained_LOC', 'fine_grained_NUM_context_first', 'fine_grained_ENTY', 'fine_grained_NUM', 'pick_the_best_descriptor',

Alignment Data Collection Methods

Handcrafted Human Prompts
(DataBricks)

`databricks-dolly-15k` dataset:

- Use 5000 employees
- Setup contest: give top 20 labelers a prize
- Have them label 7 very specific tasks.

[Free Dolly: Introducing the World's First Open and Commercially Viable Instruction-Tuned LLM - The Databricks Blog]

Alignment Data Collection Methods

Handcrafted Human Prompts
(DataBricks)

Open Q&A: "Why do people like comedy movies?" or "What is the capital of France?" In some cases, there's not a correct answer, and in others, it requires drawing on knowledge of the world at large.

How do I build a campfire?

Safety should always come first when starting a campfire. Ensure you have water or another way to extinguish your fire readily available. Use a campfire ring, pit, or clear a large area around your campfire location. Also, make sure campfires are permitted in your area. With safety out of the way, all a fire needs are heat, fuel, and oxygen. Any combustible material will start a fire, but good campfire fuel sources include tinder, kindling, and dry logs.

Use stuff like dry pine needles, dry leaves, grasses, and wood shavings for tinder. You could also use newspaper (or any paper), dryer lint, or cardboard. For kindling, use smaller pieces of wood, such as twigs and sticks no thicker than 1 inch in diameter (about the thickness of your thumb).

Dry or well-seasoned logs will ensure a good burn. Stack the wood in a way that guarantees oxygen can easily reach your flame. Many prefer a teepee or log cabin stacking design. Now, use a heat source to create a flame or spark and apply this to your tinder. Once the tinder combusts, it should light the kindling, and the kindling should eventually ignite your logs. If all goes well, you should now be enjoying your campfire.

Alignment Data Collection Methods

Handcrafted Human Prompts
(DataBricks)

Brainstorming: This task asked for open-ended ideation and an associated list of possible options. For instance, "What are some fun activities I can do with my friends this weekend?"

Instead of making a peanut butter and jelly sandwich, what else could I combine peanut butter with in a sandwich? Give five ideas.

Instead of jelly, try one of the following with peanut butter in a sandwich:

1. Honey
2. Raisins
3. Fruit preserves
4. Bananas
5. Sliced apples

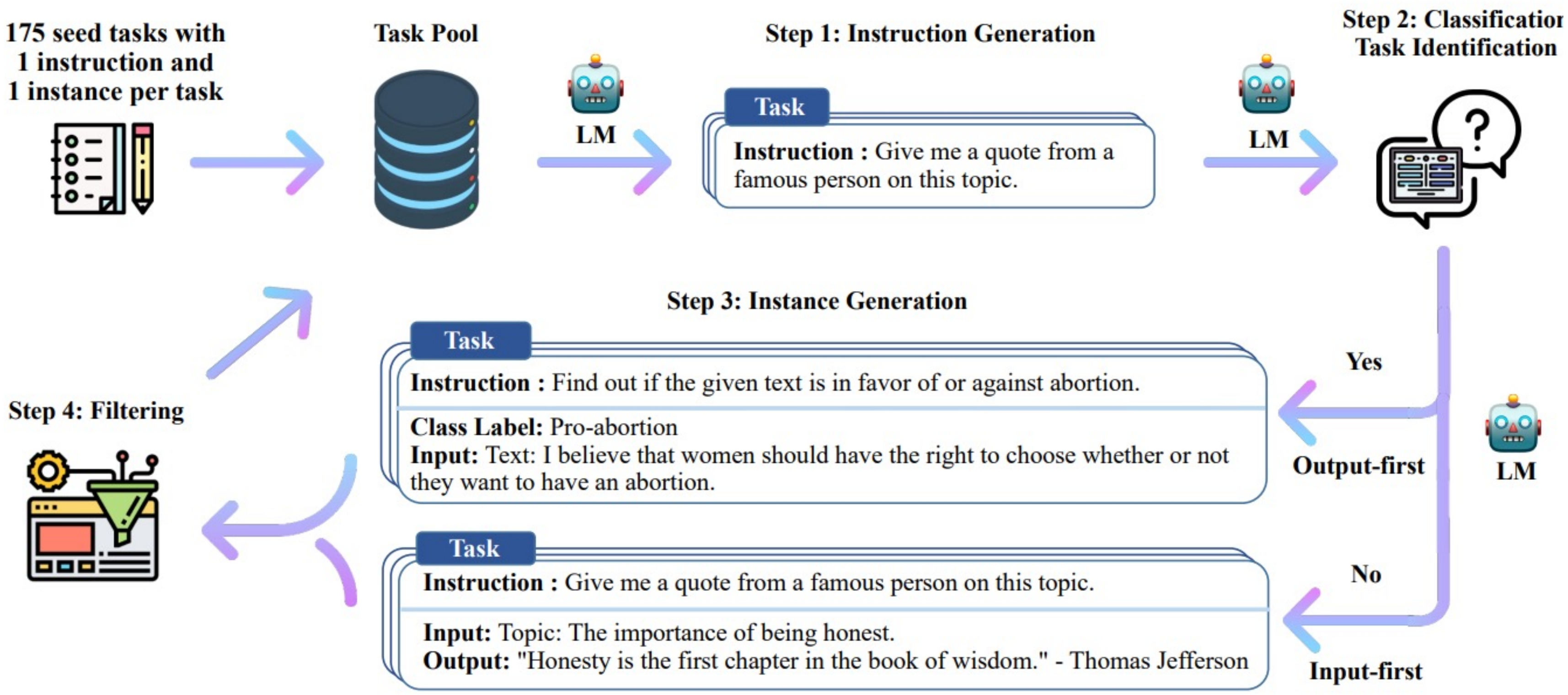
Alignment Data Collection Methods

1. Start with **initial** seed of instructions,
2. **Sample** random tasks from task pool
3. **Feed** to 'off-the-shelf' LM and generate new instructions & instances
4. **Categorize** the tasks
5. **Filter** out low quality or similar generations and augment task pool

Challenges: Improving quality of inputs and outputs

Instructions from Strong LLMs

Self-instruct:

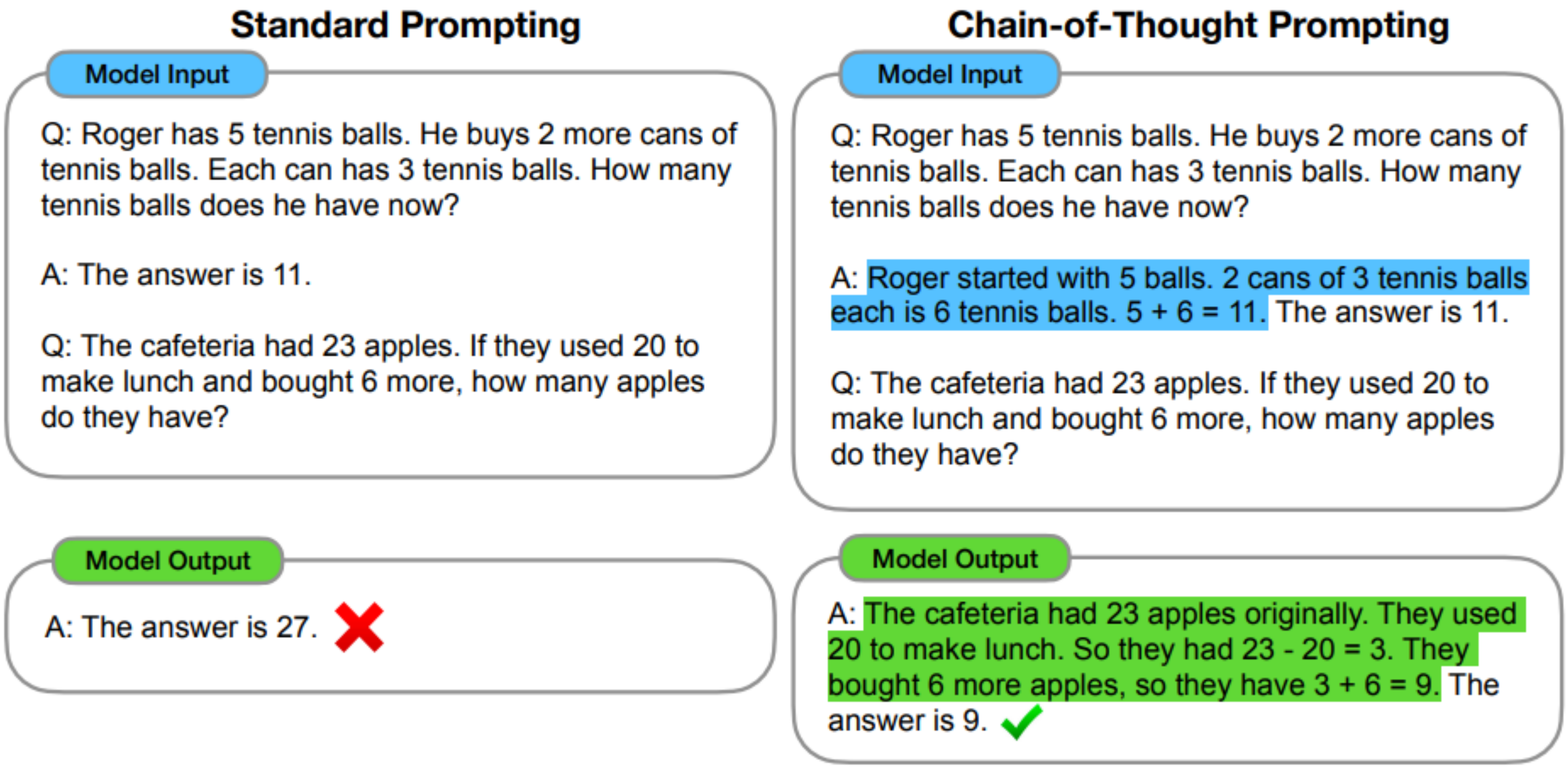


https://doi.org/10.48550/arXiv.2212.10560

Alignment Data Collection Methods

Improving LLM generated prompts

Reason-provoking Conditions: chain-of-thought approach.



2201.11903.pdf (arxiv.org)

Alignment Data Collection Methods

Instructions from Strong LLMs

Multi-turn Instructions: (Baize)

- Start with seed dataset,
- However, unlike previous, cause this initial seed prompt to start a self-chat in ChatGPT.
- Record all of ChatGPT's dialogue (this becomes multi-turn)
- Use ChatGPT to rank Baize 1.0's responses
- Build upon Baize 2.0

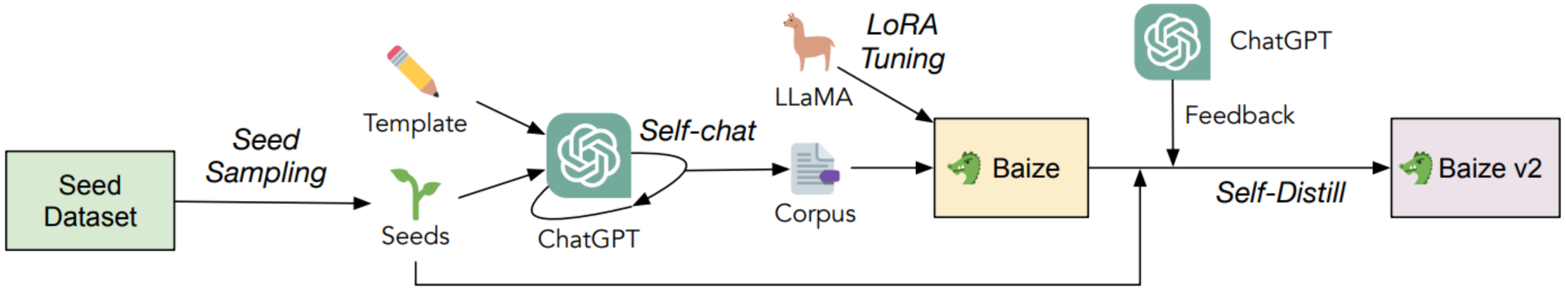


Figure 1: The pipeline for training Baize and Baize v2.

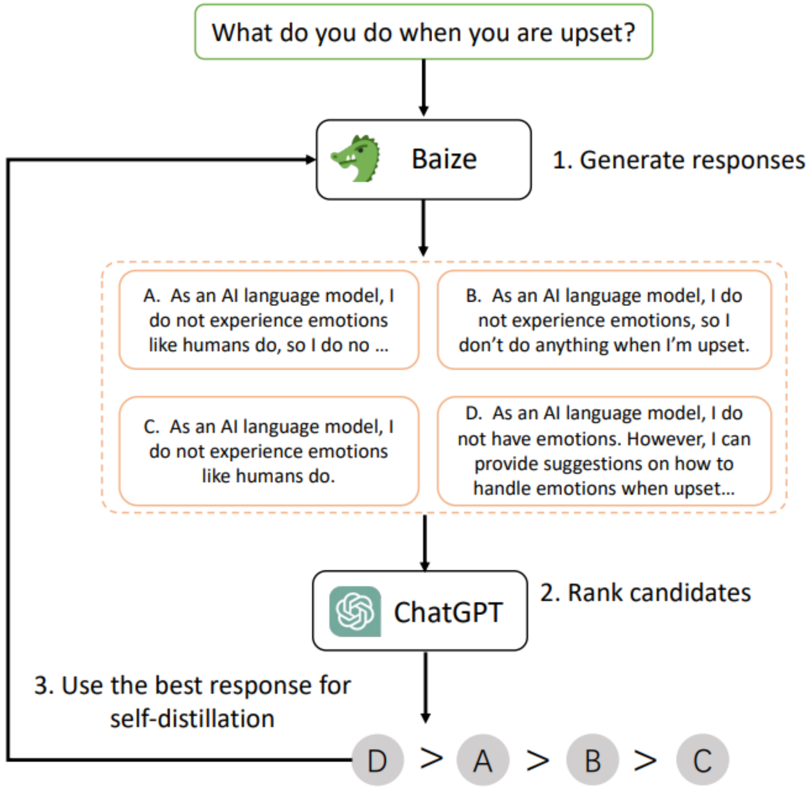


Figure 2: An overview of self-distillation with feedback from ChatGPT.

2304.01196.pdf (arxiv.org)

Alignment Data Collection Methods

Multilingual Alignment (**Bayling**)

Combine translation, following directions from user, multi-turn chatting

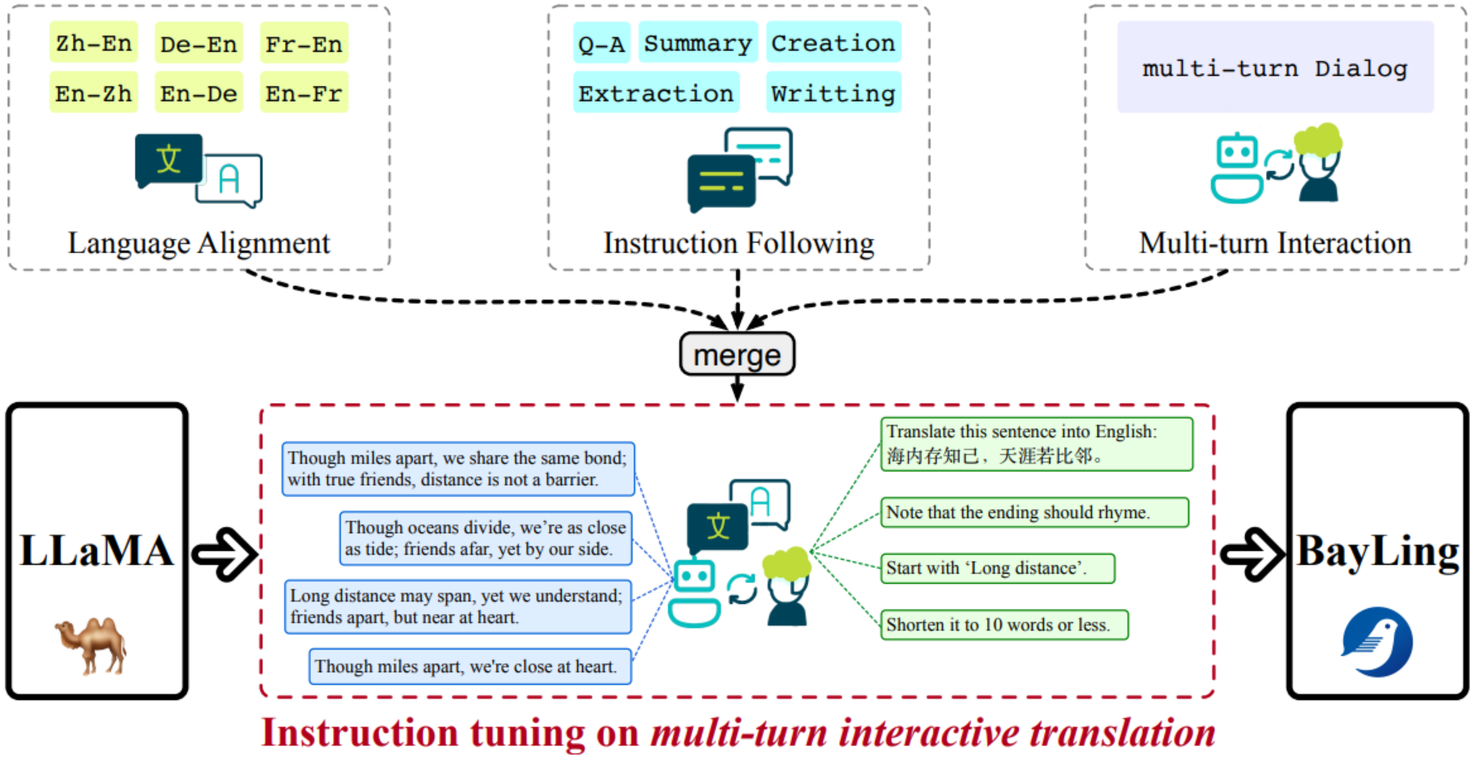


Figure 1: Overview of BayLing. BayLing is built upon LLaMA and fine-tuned with instruction data of interactive translation task and general tasks.

2306.10968.pdf (arxiv.org)

Alignment Data Collection Methods

Multilingual Alignment (**Bayling**)

Multi-turn

Translate-->

Directions-->

Directions-->

Below is a dialog consisting of instructions and responses. Write a response that completes the request.
Instruction:
Translate this sentence into English: 海内存知己，天涯若比邻。
Response:
Though miles apart, we share the same bond; with true friends, distance is not a barrier.
Instruction:
Note that the ending should rhyme.
Response:
Though oceans divide, we're as close as tide; friends afar, yet by our side.
Instruction:
Start with 'Long distance'.
Response:
Long distance may span, yet we understand; friends apart, but near at heart.
Instruction:
Shorten it to 10 words or less.
Response:
Though miles apart, we're close at heart.
...

2306.10968.pdf (arxiv.org)

Alignment Data Collection Methods

Instruction Data Management

We generated all these instructions. How much of it do we really need?

- Instruction Implications:
  Can we integrate instructions from different sources?

- Instruction Quantity:
  What is the optimal quantity of instruction data --> effective alignment

Alignment Data Collection Methods

Instruction Data Management

Mixing our datasets might yield better results

Table 3: Comparison of different instruction tuning datasets, showing that different instruction-tuning datasets can excel in different aspects, and mixtures perform best on average. Cells are blue if the finetuning boosts the vanilla LLaMA performance, and orange if the finetuning hurts the performance.

	MMLU (factuality)	GSM (reasoning)	BBH (reasoning)	TydiQA (multilinguality)	Codex-Eval (coding)	AlpacaEval (open-ended)	Average
	EM (0-shot)	EM (8-shot, CoT)	EM (3-shot, CoT)	F1 (1-shot, GP)	P@10 (0-shot)	Win % vs Davinci-003	
Vanilla LLaMa 13B	42.3	14.5	39.3	43.2	28.6	-	-
+SuperNI	49.7	4.0	4.5	**50.2**	12.9	4.2	20.9
+CoT	44.2	40.0	41.9	47.8	23.7	6.0	33.9
+Flan V2	**50.6**	20.0	40.8	47.2	16.8	3.2	29.8
+Dolly	45.6	18.0	28.4	46.5	31.0	13.7	30.5
+Open Assistant 1	43.3	15.0	39.6	33.4	31.9	58.1	36.9
+Self-instruct	30.4	11.0	30.7	41.3	12.5	5.0	21.8
+Unnatural Instructions	46.4	8.0	33.7	40.9	23.9	8.4	26.9
+Alpaca	45.0	9.5	36.6	31.1	29.9	21.9	29.0
+Code-Alpaca	42.5	13.5	35.6	38.9	34.2	15.8	30.1
+GPT4-Alpaca	46.9	16.5	38.8	23.5	**36.6**	63.1	37.6
+Baize	43.7	10.0	38.7	33.6	28.7	21.9	29.4
+ShareGPT	49.3	27.0	40.4	30.5	34.1	**70.5**	42.0
+Human data mix.	50.2	38.5	39.6	47.0	25.0	35.0	39.2
+Human+GPT data mix.	49.3	**40.5**	**43.3**	45.6	35.9	56.5	**45.2**

[2306.04751] How Far Can Camels Go? Exploring the State of Instruction Tuning on Open Resources (arxiv.org)

Alignment Data Collection Methods

Filter Data using ChatGPT

Reduce size of ALPACA dataset (52k) (generated from openAI davinci-text)

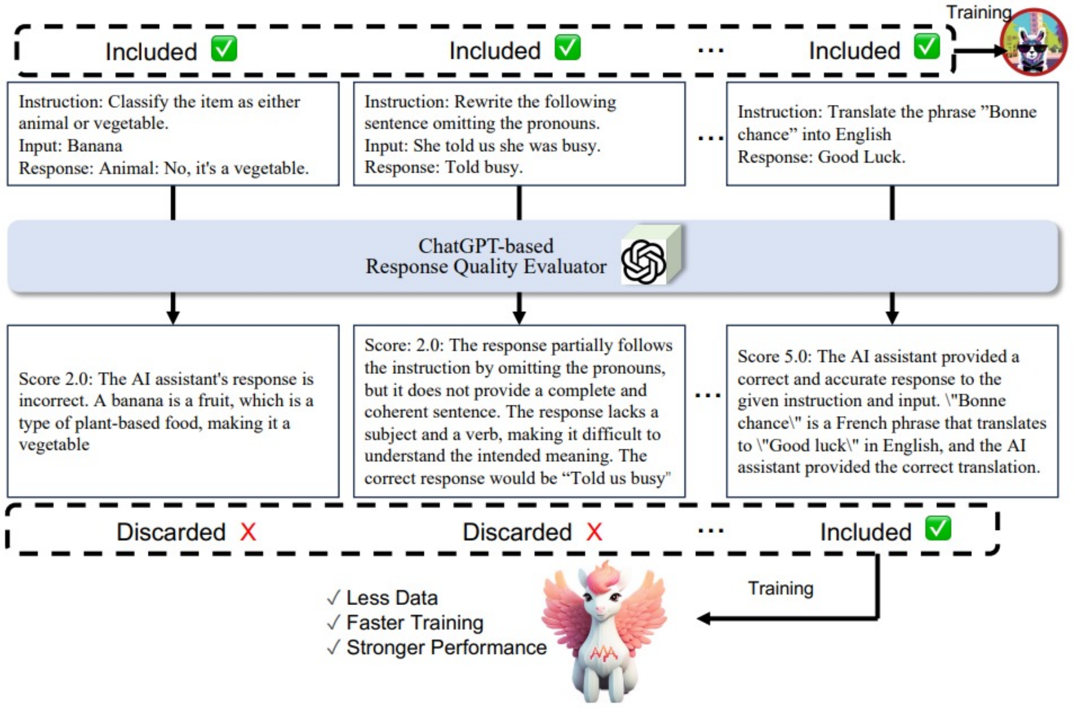


Figure 3: The fine-tuning pipeline of ALPAGASUS. We prompt ChatGPT as our auto-grader to score each training triplet on a scale of 0 to 5. We then use the exact same instruction fine-tuning script of ALPACA to train ALPAGASUS on the filtered data with scores higher than a threshold.

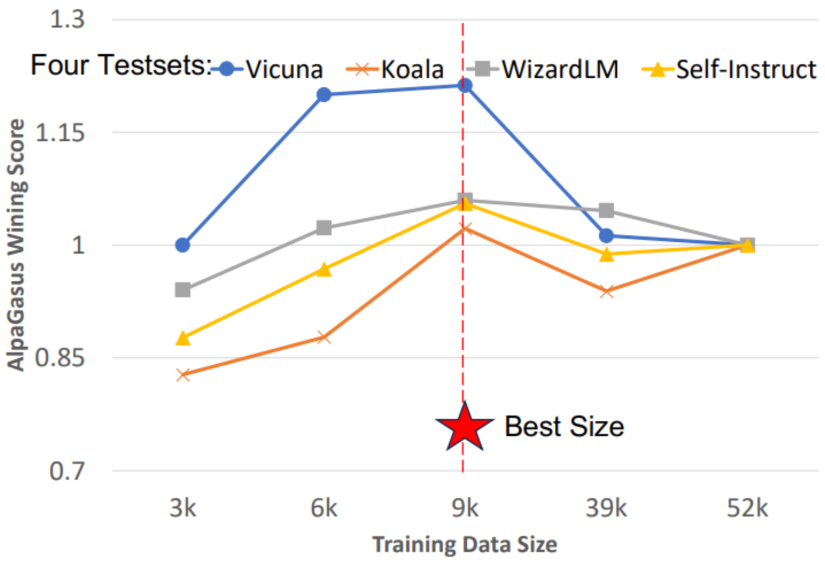


Figure 2: Performance of ALPAGASUS on four test sets when increasing its finetuning data where the winning score is $\frac{\#\text{Win}-\#\text{Lose}}{\#\text{Testset}} + 1$ with #Testset = #Win + #Tie + #Lose to be the test set size and #Win/#Tie/#Lose to be the number of samples on which ALPAGASUS wins/ties/loses compared to ALPACA 52K.

[2307.08701] AlpaGasus: Training A Better Alpaca with Fewer Data (arxiv.org)

UNIVERSITY of VIRGINIA

Tonmoy Hossain, *pwg7jb*

Presentation Outline

- ❖ Benchmarking in AI
- ❖ Evaluation Framework Design
- ❖ **Alignment Training and Evaluation**
- ❖ Alignment Performance and InstructGPT
- ❖ SFT and RL
- ❖ Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Alignment Training

- Data is used to fine-tune existing foundational LLMs to align with human

Supervised Fine-Tuning (SFT)

Given instruction **input *x***, SFT calculates the cross-entropy loss over the **ground-truth response *y***

$$L_{ft} = -\sum_t \log P_{LLM}(y_{i',t} | x, y_{i',<t}) \qquad (1)$$

\+ SFT helps LLMs to understand the semantic meaning of prompts

— teaches LLMs about the best responses and cannot provide fine-grained comparisons to suboptimal ones

Alignment Training

- Data is used to fine-tune existing foundational LLMs to align with human

Supervised Fine-Tuning (SFT)

- *SFT model parameters* has been integrated into many human preference training objective

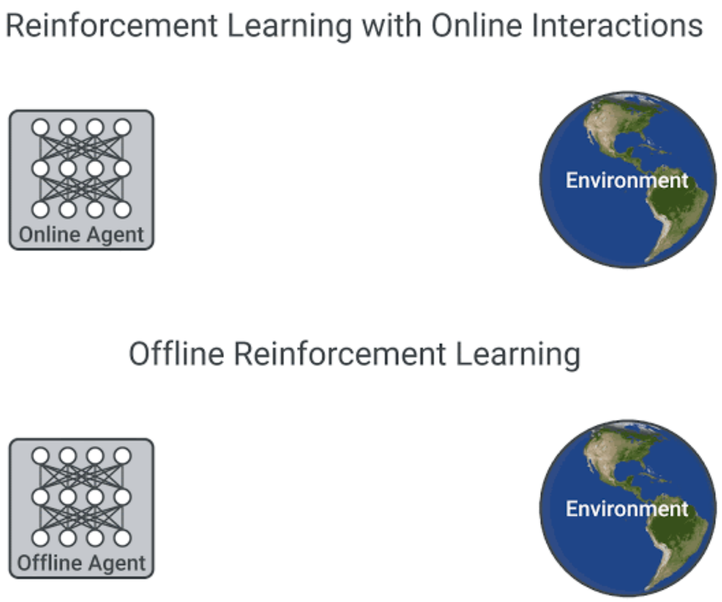
1. Online human preference training
2. Offline human preference training
3. Parameter-effective fine-tuning solutions

AT: Online vs Offline RL

Online RL: The agent interacts directly with the environment and collects data through its own experience.

Offline RL: The agent learns from a fixed dataset collected beforehand, without any new interaction.

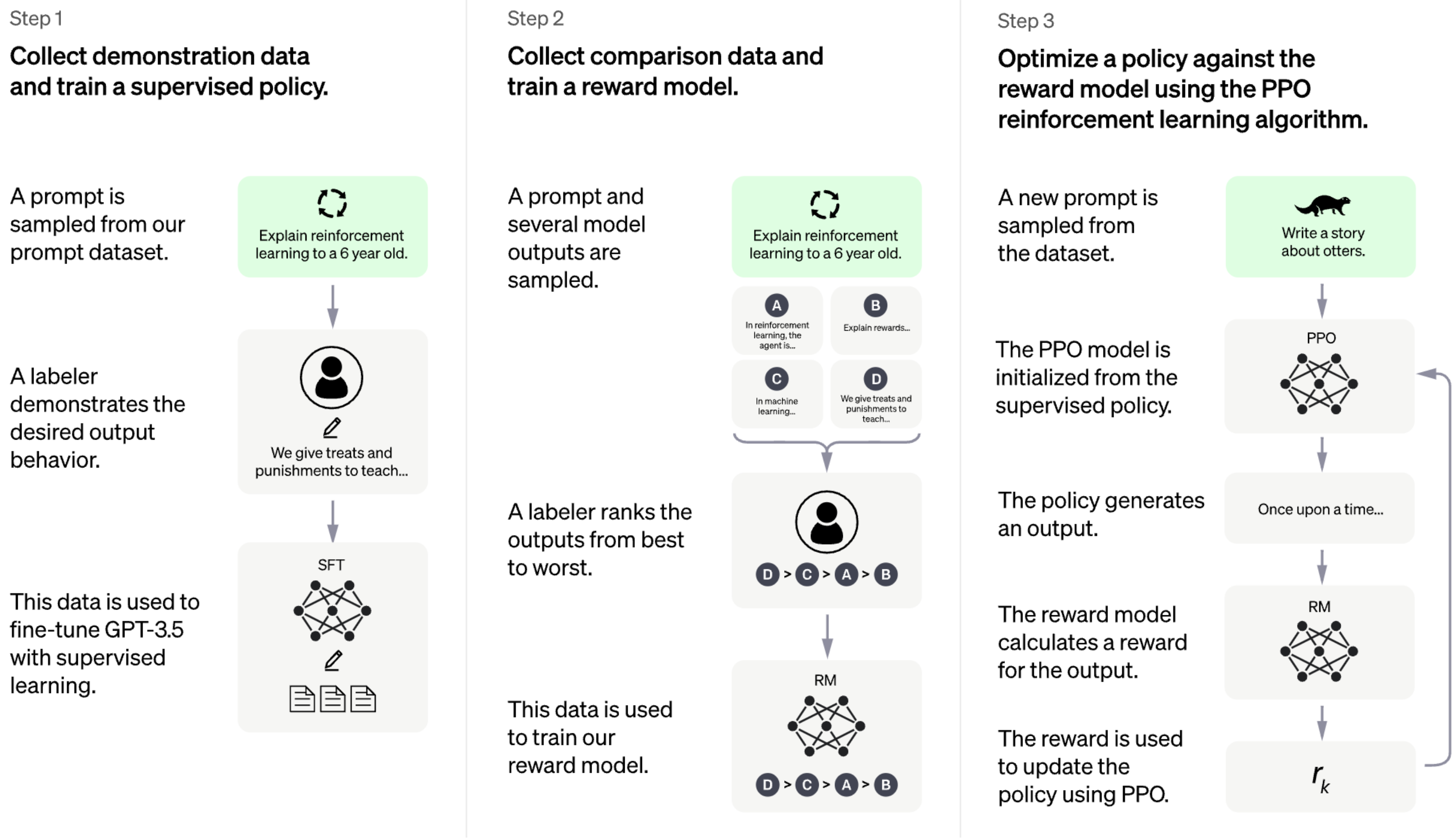
Online RL	Offline RL
learn from new experience	learn from fixed data
adapts to changing distributions	assumes static
exploration done by agent	relies on dataset coverage
expensive/risky	faster and safer



[Google Research](#)

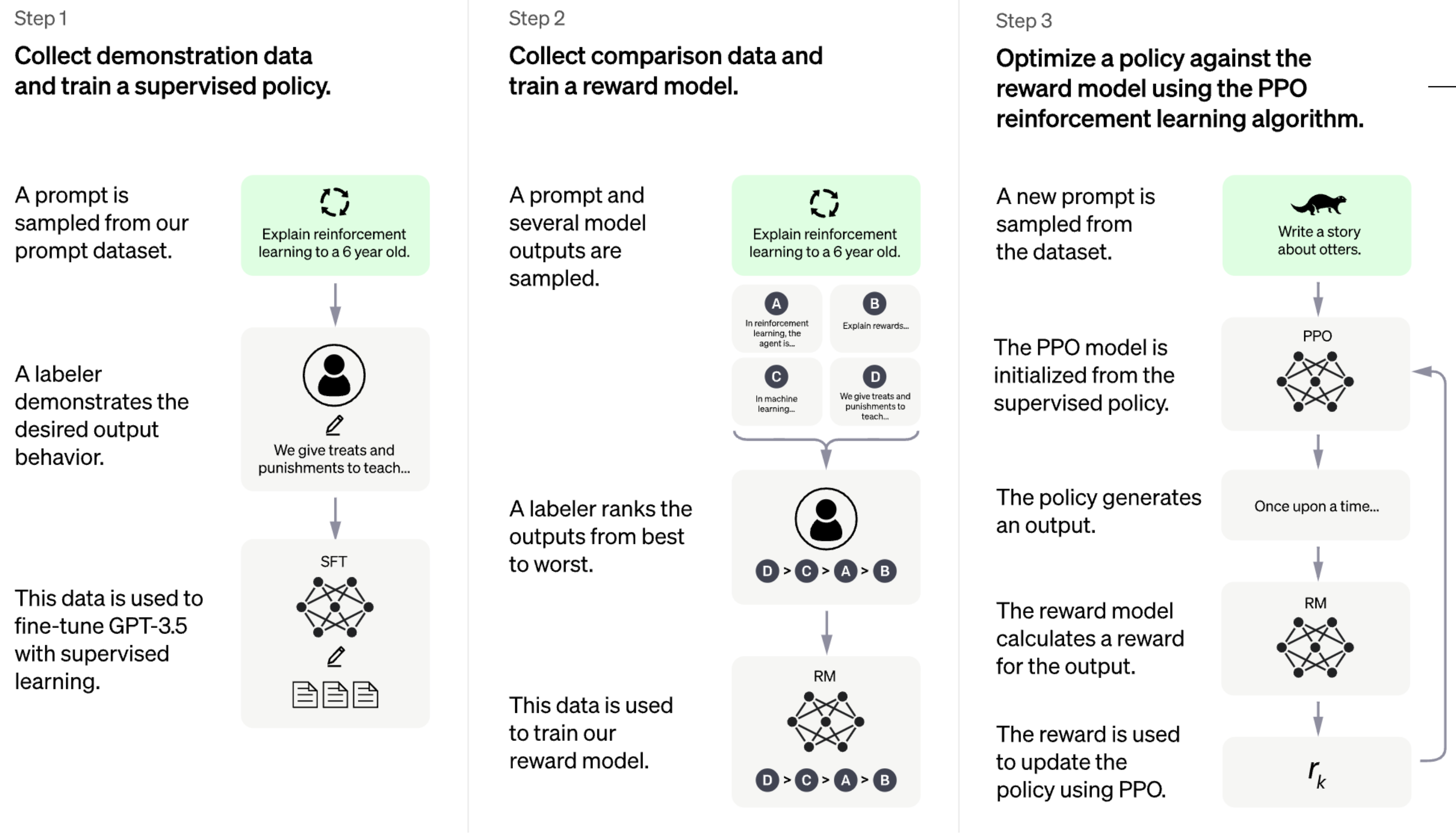
AT: Online Human Preference Training

1. Reinforcement learning from Human Feedback (RLHF) is designed to learn the human preference signals from external reward models



AT: Online Human Preference Training

1. Reinforcement learning from Human Feedback (RLHF) is designed to learn the human preference signals from external reward models



Optimization → Overfitting???

add a KL-divergence regularization between the current model weight and the SFT model weight

PPO training is difficult in implementation and stable training

AT: Online Human Preference Training

2. Reward rAnked Fine Tuning (RAFT)
    - uses an existing reward model to select the best set of training samples based on the model outputs

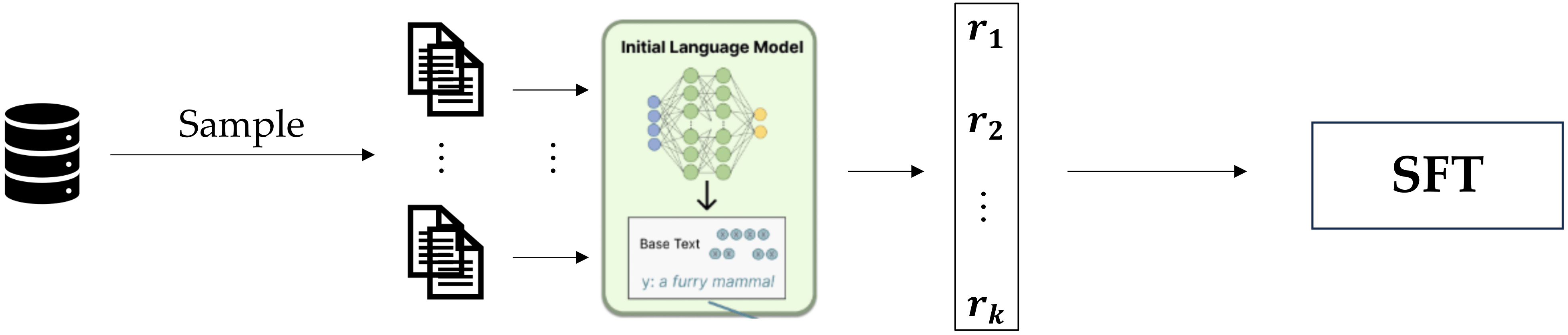


Fig. RAFT Pipeline

AT: Online Human Preference Training

Training procedure requires interaction between policy, behavior policy, reward, and value model, which **requires many hyper-parameters to be tuned** to achieve better stability and performance

AT: Offline Human Preference Training

Learning human preferences in an offline fashion

1. **Ranking-based Approaches.**
   - incorporate the ranking information into the LLMs fine-tuning stage

1.1 *Direct Preference Optimization (DPO):* Optimizes the same objective as existing RLHF algorithms (i.e., reward function with a KL-divergence term)

1.2 *Preference Ranking Optimization (PRO):* Finetune LLMs to align with human preference
- PRO also adds SFT training objective for the regularization purpose

AT: Offline Human Preference Training

Learning human preferences in an offline fashion

1. **Ranking-based Approaches.**
    - incorporate the ranking information into the LLMs fine-tuning stage

<u>1.3</u> SFT training objective and KL divergence as the regularization term

- rank loss with the KL-divergence term performs the best
- experiment on small pre-trained language models

<u>1.4</u> *RRHF*: Optimizes LLaMA-7B to align with human preferences

- **SFT training objective** is more effective and efficient than KL-divergence in preventing LLMs from over-fitting

AT: Offline Human Preference Training

Learning human preferences in an offline fashion

2. **Language-based Approaches.**
    - Propose to directly use natural language to inject human preference via SFT

2.1 *Concept Behavior Cloning:* train LLMs to distinguish high- and low quality instruction responses, leveraging both low- and high-quality training data to align LLMs with humans

2.2 *Chain of Hindsight*: incorporates human preference as a pair of parallel responses discriminated as low-quality or high-quality using natural language prefixes

AT: Offline Human Preference Training

Learning human preferences in an offline fashion

2. Language-based Approaches.

- Propose to directly use natural language to inject human preference via SFT

2.2 *Chain of Hindsight*: incorporates human preference as a pair of parallel responses discriminated as low-quality or high-quality using natural language prefixes

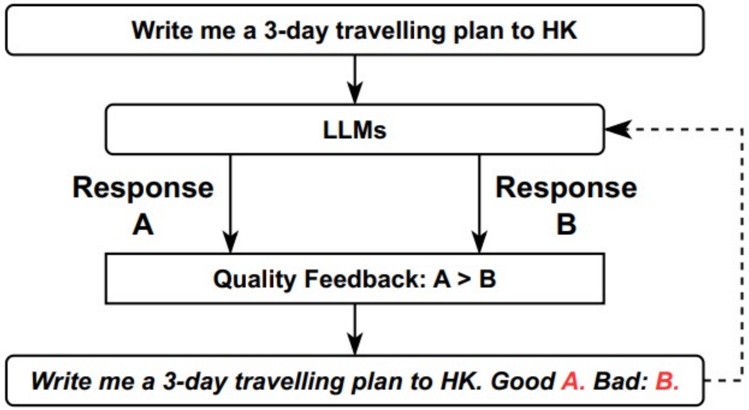


Figure 4: The overview of the Chain of Hindsigt (CoH) method. Responses with different quality are associated with different prefix. The CoH training loss is only applied on model output tokens (highlighted by red).

CoH also incorporates SFT objectives and random words masking to prevent LLMs from over-fitting

AT: Parameter-Effective Training

+ LLMs would enable the models to adhere to provided instructions

– vast GPU memory and extensive datasets for instruction training

AT: Parameter-Effective Training (PET)

+ LLMs would enable the models to adhere to provided instructions

– vast GPU memory and extensive datasets for instruction training

PET-based methods froze the major part of LLM parameters and only train a limited set of additional parameters

AT: Parameter-Effective Training

1. **Supplementary Parameters:** prepend trainable tokens to the input/each hidden layer, leaving the parameters of LLMs frozen during fine-tuning.

2. **Shadow Parameters:** training the weight representing model parameter variance without modifying the number of total model parameters during inference
   - LoRA (Low-Rank Adaptation): Given a neural layer $h = W_0x$, LoRA modifies the forward pass as follows:

$$h = W_0x + BAx \tag{4}$$

   LoRA only updates the parameters of A and B during training

   ✓ AdaLoRA, QLoRA

AT: Parameter-Effective Training (Trade-offs)

Underfitting Issue

- Given the same set of training instructions, LLMs with LoRA perform worse than the fully fine-tuned ones (Sun et al. 2023)
- Using LoRA, it is preferable to use larger LLMs than larger training instruction datasets

Alignment Evaluation

Evaluation for alignment quality

AE 1: Evaluation Benchmarks

AE 2: Evaluation Paradigm

AE 1: Evaluation Benchmark

AE 1.1: ***Closed-set Benchmarks*** - evaluating the skills and knowledge of aligned LLMs

- Possible answers are predefined and limited to a finite set (e.g., multiple choices)

AE 1.1.1: General Knowledge

MMLU — evaluate LLMs knowledge in zero-shot and few-shot settings

Chinese LLMs — C-MMLU, C-Eval, M3KE and AGIEval

KoLA — evaluate the general real world knowledge of LLMs

AE 1: Evaluation Benchmark

AE 1.1.2: Reasoning

Arithmetic — GSM8K, Maths

Commonsense — CSQA, StrategyQA

BBH (Subset of BIG-Bench) — Date Understanding, Word Sorting, and Causal Judgement

AE 1.1.3: Coding

HumanEval, HumanEval+, MBPP — evaluate the coding skills of LLMs

DS1000 — comprises 1000 data science workflows spanning seven libraries

— assesses the performance of code generations against test cases

AE 1: Evaluation Benchmark

AE 1.2: ***Open-ended Benchmarks*** — responses to open-set benchmarks can be more flexible and diverse

AE 1.2.1: leverage a small number of syntactic instructions from LLMs — Vicuna-80, Open-Assistant-953, User-Instructions-252

— provide comparison several LLMs at a time

AE 1.2.2: AlpacaEval — reporting the Win Rate, the higher the better

MT-Bench, FLASK

AE 2: Evaluation Paradigm

AE 2.1: *Human-based Evaluation*

- BLUE, ROGUE: require ground-truth and have relatively low correlation with human judgments
- Human annotators are used to evaluate the quality of open-ended model responses
  - categorize each response into one of the four levels (i.e., acceptable, minor errors, major errors and unacceptable) — heavily depend on the subjectivity of annotators
  - pairwise comparison framework

AE 2: Evaluation Paradigm

AE 2.2: ***LLMs-based Evaluation***

- Human evaluations are inefficient and expensive
- Recent studies propose to incorporate LLMs into the output text evaluation in various NLP tasks

LLMs Evaluation Bias

- LLM-based evaluation paradigm suffers from a positional bias and those strong LLMs (i.e., GPT-4) tend to assign higher scores to the first appeared candidates
- Self-enhancement bias: LLMs favor their own responses

UNIVERSITY of VIRGINIA

Shafat Shahnewaz (gsq2at)

Presentation Outline

- ❖ Benchmarking in AI
- ❖ Evaluation Framework Design
- ❖ Alignment Training and Evaluation
- ❖ **Alignment Performance and InstructGPT**
- ❖ SFT and RL
- ❖ Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Challenges and Future Directions

English dominant LLMs along with LLaMA as pre-trained initial LLMs

Aligned LLM	Size	Lang.	Initial LLMs	Training	Self Instruction	NLP Benchmarks	Human Annotations	Human Eval	Auto. Benchmark Eval	LLM Eval
Alpaca (Taori et al., 2023)	7B	EN	LLaMA	SFT	Text-Davinci-003	✗	✗	Author Verification	✗	✗
Vicuna (Chiang et al., 2023)	7B, 13B, 33B	EN	LLaMA	SFT	GPT-3.5	✗	70K ShareGPT	✗	✗	*Vicuna-80*
GPT4ALL (Anand et al., 2023)	6B, 13B	EN	LLaMA GPT-J	SFT	✗	Bloomz-P3	OIG, ShareGPT, Dolly Stack Overflow	✗	Common Sense Reasoning	✗
LLaMA-GPT4 (Peng et al., 2023)	7B	EN, CN	LLaMA	SFT	Text-Davinci-003 GPT-4	✗	✗	*User-Instructions-252* Pairwise, AMT	Unnatural Instructions	*Vicuna-80*
Phoenix (Chen et al., 2023e)	7B, 13B	Multilingual	LLaMA BLOOMZ	SFT	GPT-3.5 Multilingual and Dialogue Data	✗	ShareGPT	Volunteers	✗	GPT-3.5, GPT-4
UltraLLaMA (Ding et al., 2023)	13B	EN	LLaMA	SFT	GPT-3.5 Dialogue Data	✗	✗	✗	Truthful QA	GPT 3.5 *Vicuna-80* 300 diverse questions
Baize (Xu et al., 2023c)	7B, 13B, 30B	EN	LLaMA	Revision, LoRA	GPT-3.5 self-Chat Data	✗	Quora Questions	✗	✗	GPT-4
WizardLM (Xu et al., 2023b)	7B, 13B, 30B	EN	LLaMA	SFT	GPT-3.5, Alpaca Complex Instructions	✗	ShareGPT	10 Annotators Pairwise Comparison	✗	GPT-4, *WizedLM-218*
WizardCoder (Luo et al., 2023)	15B	EN, Code	StarCoder	SFT	GPT-3.5, Code Alpaca Complex Instructions	✗	✗	✗	HumanEval, MBPP HumanEval+, DS-1000	✗
OpenChat (Wang et al., 2023a)	13B	EN	LLaMA	Language	✗	✗	GPT 3.5 & GPT4 ShareGPT	✗	MMLU	GPT-4
Guanaco (Dettmers et al., 2023)	13B, 33B, 65B	EN	LLaMA	QLoRA	Alpaca, SELF-INSTRUCT Unnatural instructions	FLAN	Chip2	Elo, *Vicuna-80*	MMLU	Elo, *Vicuna-80* *Open-Assistant-953*
MPT-chat (Team, 2023)	13B, 30B	EN	MPT	SFT	GPTeacher, Guanaco Baize Instructions	✗	Vicuna ShareGPT	✗	MMLU	GPT4, MT-bench
FLACUNA (Ghosal et al., 2023)	13B	EN	Vicuna	LoRA	Alpaca, Code Alpaca	FLAN	ShareGPT	✗	MMLU, BBH, DROP CRASS, HumanEval	GPT 3.5, IMPACT
Bactrian-X (Li et al., 2023b)	7B	Multilingual	LLaMA BLOOMZ	LoRA	Alpaca Google Translation	✗	✗	✗	XCOPA, XStoryCloze XWinograd, SentimentX	GPT 4 Multilingual *Vicuna-80*
Ocra (Mukherjee et al., 2023)	13B	EN	LLaMA	SFT	✗	FLAN	✗	✗	AGIEval, BBH	GPT-4, *Vicuna-80* *WizedLM-218*, *Awesome-164*
Phi-1 (Gunasekar et al., 2023)	350M, 1.3B	EN, Code	Phi-1-base	SFT	GPT-3.5 Synthetic Textbook	✗	Python, The Stack Stack Overflow	✗	HumanEval	GPT-4 Grading
Chinese Alpaca (Cui et al., 2023b)	7B, 13B, 33B	EN, CN	Chinese LLaMA	LoRA	STEM Org. and Trans. Alpaca	pCLUE	✗	✗	C-Eval	✗
Lion (Jiang et al., 2023)	7B, 13B	EN	LLaMA	SFT	Alpaca GPT 3.5 Adv. Instruction	✗	✗	HHH *User-Instructions-252*	✗	GPT-4, *Vicuna-80*
Stable Alignment (Liu et al., 2023d)	7B	EN	Alpaca	SFT	GPT-3.5 Social Aligned Instructions	✗	✗	✗	✗	GPT-4 HHH, HHH-A
Dromedary (Sun et al., 2023b)	65B	EN	LLaMA	SFT	LLaMA-65B, Self-Align	✗	175 Munnal Examples 16 Principle Rules	✗	TruthfulQA, BBH	GPT-4, *Vicuna-80*
Dolly-v2 (Conover et al., 2023)	3B, 7B, 12B	EN	Pythia	SFT	✗	✗	*databricks-dolly-15k*	✗	*LLM Harness*	✗
Selfee (Ye et al., 2023a)	7B, 13B	EN	LLaMA	Revision	GPT 3.5 Self-Improve Alpaca	FLAN, Maths, Code	ShareGPT	✗	✗	GPT-4, *Vicuna-80*
TÜLU (Wang et al., 2023d)	7B, 13B, 30B, 65B	EN	LLaMA	SFT	Alpaca, Code Alpaca GPT4-Alpaca, Self-instruct	FLAN, CoT	Dolly, ShareGPT Open Assistant	Acceptability Pairwise Comparison	MMLU, GSM, BBH TydiQA, Codex-Eval	GPT4 on *Vicuna-80*, Koala Open Assistant Benchmarks
Koala (Geng et al., 2023)	13B	EN	LLaMA	Language	Alpaca	✗	OIG, HC3, Anthropic HH OpenAI WebGPT, Summary	100 AMT Annotators on *Alpaca and Koala Test*	✗	✗
Bayling (Zhang et al., 2023c)	7B, 13B	Multilingual	LLaMA	SFT	Alpaca GPT 3.5 Interactive Translation	✗	ShareGPT	Translation Quality	WMT22 Multilingual Translation Lexically Constrained Translation	✗
Wombat (Yuan et al., 2023)	7B	EN	Alpaca	Rank	Alpaca ChatGPT Ratings	✗	*Helpful and Harmless*	✗	✗	GPT-4, *Vicuna-80*
Lamini-lm (Wu et al., 2023)	0.7B	EN	T5-Flan	SFT	Alpaca Self-instruct	P3, FLAN	✗	Human Rating	*LLM harness*	✗

Challenges and Future Directions

Most of the LLMs are based on SFT technology and FLAN emerges as the benchmark

Aligned LLM	Size	Lang.	Initial LLMs	Training	Self Instruction	NLP Benchmarks	Human Annotations	Human Eval	Auto. Benchmark Eval	LLM Eval
Alpaca (Taori et al., 2023)	7B	EN	LLaMA	SFT	Text-Davinci-003	✗	✗	Author Verification	✗	✗
Vicuna (Chiang et al., 2023)	7B, 13B, 33B	EN	LLaMA	SFT	GPT-3.5	✗	70K ShareGPT	✗	✗	*Vicuna-80*
GPT4ALL (Anand et al., 2023)	6B, 13B	EN	LLaMA GPT-J	SFT	✗	Bloomz-P3	OIG, ShareGPT, Dolly Stack Overflow	✗	Common Sense Reasoning	✗
LLaMA-GPT4 (Peng et al., 2023)	7B	EN, CN	LLaMA	SFT	Text-Davinci-003 GPT-4	✗	✗	*User-Instructions-252* Pairwise, AMT	Unnatural Instructions	*Vicuna-80*
Phoenix (Chen et al., 2023e)	7B, 13B	Multilingual	LLaMA BLOOMZ	SFT	GPT-3.5 Multilingual and Dialogue Data	✗	ShareGPT	Volunteers	✗	GPT-3.5, GPT-4
UltraLLaMA (Ding et al., 2023)	13B	EN	LLaMA	SFT	GPT-3.5 Dialogue Data	✗	✗	✗	Truthful QA	GPT 3.5 *Vicuna-80* 300 diverse questions
Baize (Xu et al., 2023c)	7B, 13B, 30B	EN	LLaMA	Revision, LoRA	GPT-3.5 self-Chat Data	✗	Quora Questions	✗	✗	GPT-4
WizardLM (Xu et al., 2023b)	7B, 13B, 30B	EN	LLaMA	SFT	GPT-3.5, Alpaca Complex Instructions	✗	ShareGPT	10 Annotators Pairwise Comparison	✗	GPT-4, *WizedLM-218*
WizardCoder (Luo et al., 2023)	15B	EN, Code	StarCoder	SFT	GPT-3.5, Code Alpaca Complex Instructions	✗	✗	✗	HumanEval, MBPP HumanEval+, DS-1000	✗
OpenChat (Wang et al., 2023a)	13B	EN	LLaMA	Language	✗	✗	GPT 3.5 & GPT4 ShareGPT	✗	MMLU	GPT-4
Guanaco (Dettmers et al., 2023)	13B, 33B, 65B	EN	LLaMA	QLoRA	Alpaca, SELF-INSTRUCT Unnatural instructions	FLAN	Chip2	Elo, *Vicuna-80*	MMLU	Elo, *Vicuna-80* *Open-Assistant-953*
MPT-chat (Team, 2023)	13B, 30B	EN	MPT	SFT	GPTeacher, Guanaco Baize Instructions	✗	Vicuna ShareGPT	✗	MMLU	GPT4, MT-bench
FLACUNA (Ghosal et al., 2023)	13B	EN	Vicuna	LoRA	Alpaca, Code Alpaca	FLAN	ShareGPT	✗	MMLU, BBH, DROP CRASS, HumanEval	GPT 3.5, IMPACT
Bactrian-X (Li et al., 2023b)	7B	Multilingual	LLaMA BLOOMZ	LoRA	Alpaca Google Translation	✗	✗	✗	XCOPA, XStoryCloze XWinograd, SentimentX	GPT 4 Multilingual *Vicuna-80*
Ocra (Mukherjee et al., 2023)	13B	EN	LLaMA	SFT	✗	FLAN	✗	✗	AGIEval, BBH	GPT-4, *Vicuna-80* *WizedLM-218, Awesome-164*
Phi-1 (Gunasekar et al., 2023)	350M, 1.3B	EN, Code	Phi-1-base	SFT	GPT-3.5 Synthetic Textbook	✗	Python, The Stack Stack Overflow	✗	HumanEval	GPT-4 Grading
Chinese Alpaca (Cui et al., 2023b)	7B, 13B, 33B	EN, CN	Chinese LLaMA	LoRA	STEM Org. and Trans. Alpaca	pCLUE	✗	✗	C-Eval	✗
Lion (Jiang et al., 2023)	7B, 13B	EN	LLaMA	SFT	Alpaca GPT 3.5 Adv. Instruction	✗	✗	HHH *User-Instructions-252*	✗	GPT-4, *Vicuna-80*
Stable Alignment (Liu et al., 2023d)	7B	EN	Alpaca	SFT	GPT-3.5 Social Aligned Instructions	✗	✗	✗	✗	GPT-4 HHH, HHH-A
Dromedary (Sun et al., 2023b)	65B	EN	LLaMA	SFT	LLaMA-65B, Self-Align	✗	175 Munnal Examples 16 Principle Rules	✗	TruthfulQA, BBH	GPT-4, *Vicuna-80*
Dolly-v2 (Conover et al., 2023)	3B, 7B, 12B	EN	Pythia	SFT	✗	✗	*databricks-dolly-15k*	✗	*LLM Harness*	✗
Selfee (Ye et al., 2023a)	7B, 13B	EN	LLaMA	Revision	GPT 3.5 Self-Improve	FLAN, Maths, Code	ShareGPT	✗	✗	GPT-4, *Vicuna-80*
TÜLU (Wang et al., 2023d)	7B, 13B, 30B, 65B	EN	LLaMA	SFT	Alpaca Alpaca, Code Alpaca GPT4-Alpaca, Self-instruct	FLAN, CoT	Dolly, ShareGPT Open Assistant	Acceptability Pairwise Comparison	MMLU, GSM, BBH TydiQA, Codex-Eval	GPT4 on *Vicuna-80*, Koala Open Assistant Benchmarks
Koala (Geng et al., 2023)	13B	EN	LLaMA	Language	Alpaca	✗	OIG, HC3, Anthropic HH OpenAI WebGPT, Summary	100 AMT Annotators on *Alpaca and Koala Test*	✗	✗
Bayling (Zhang et al., 2023c)	7B, 13B	Multilingual	LLaMA	SFT	Alpaca GPT 3.5 Interactive Translation	✗	ShareGPT	Translation Quality	WMT22 Multilingual Translation Lexically Constrained Translation	✗
Wombat (Yuan et al., 2023)	7B	EN	Alpaca	Rank	Alpaca ChatGPT Ratings	✗	*Helpful and Harmless*	✗	✗	GPT-4, *Vicuna-80*
Lamini-lm (Wu et al., 2023)	0.7B	EN	T5-Flan	SFT	Alpaca Self-instruct	P3, FLAN	✗	Human Rating	*LLM harness*	✗

What is FLAN?

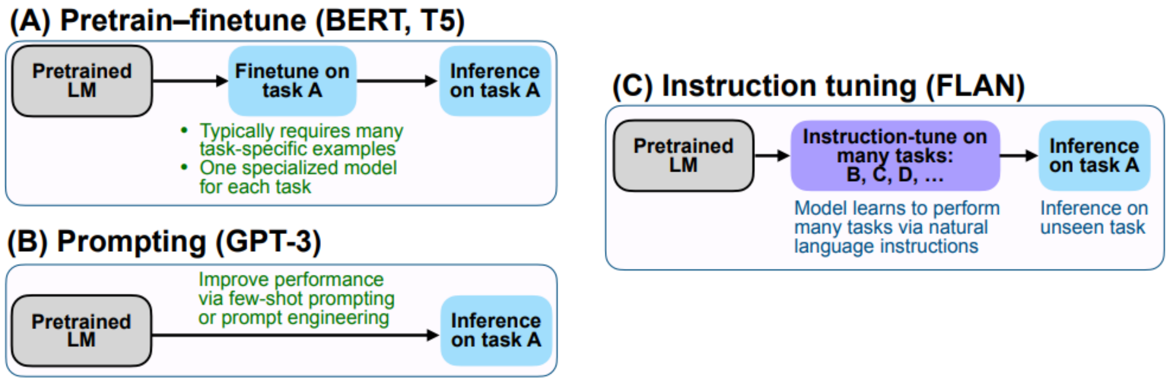
Fine-tuned LAnguage Net (FLAN)

FLAN is **an instruction tuning approach** to **fine-tune language models** on a collection of datasets described **via instructions**

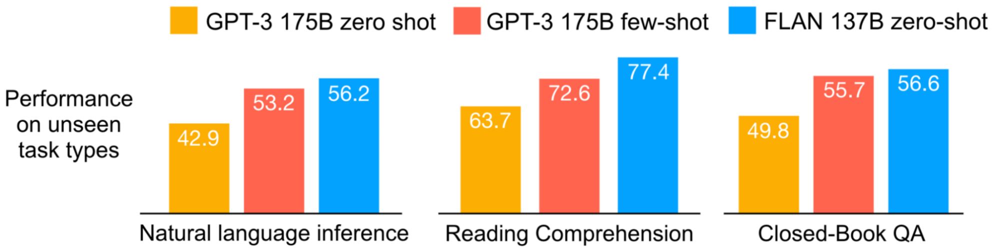
This involves fine-tuning a model not to solve a specific task, but to make it more amenable to solving NLP tasks in general

Finetuned Language Models Are Zero-Shot Learners, https://arxiv.org/abs/2109.01652

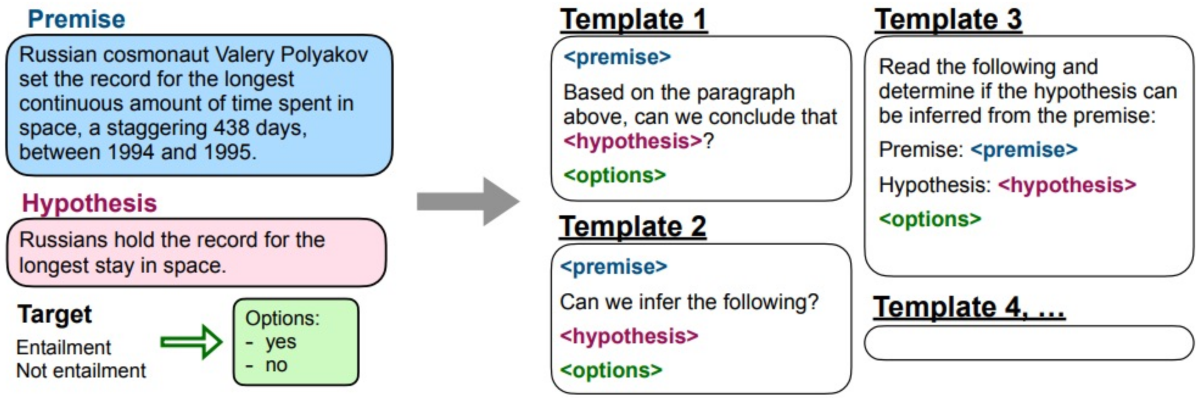
Fine-tuned LAnguage Net (FLAN)



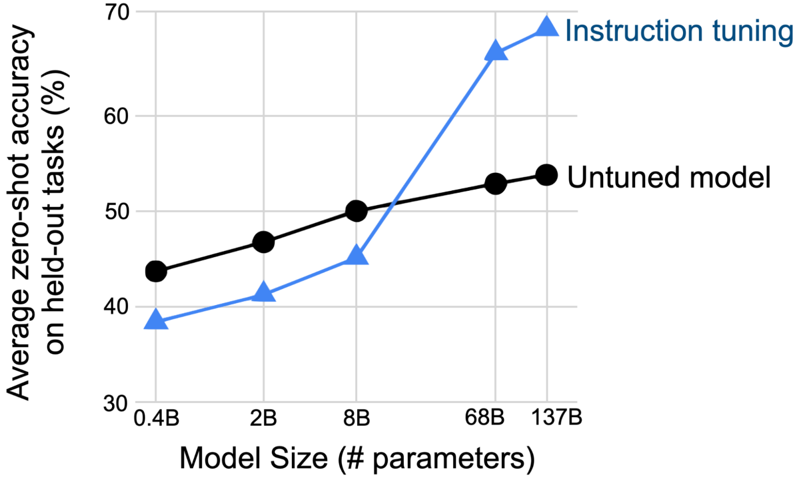
Comparing instruction tuning with pretrain–finetune and prompting



FLAN zero-shot is better than zero-shot GPT-3 on 20 of 25 tasks, and better than even few-shot GPT-3 on some tasks.



Multiple instruction templates describing a natural language inference task



Instruction tuning only improves performances on unseen tasks for models of certain size

Finetuned Language Models Are Zero-Shot Learners, https://arxiv.org/abs/2109.01652

Challenges and Future Directions

Fine-grained Instruction Data Management

1. FLAN and programming instructions can improve reasoning capability aligned LLMs

Model	Size	MMLU (0-shot)	BBH (0-shot)	CRASS (0-shot)
Flan-UL2	20B	54.4	34.9	-
OpenAssistant	30B	52.0	33.4	-
OPT IML	30B	41.3	17.4	-
TK-Instruct	11B	39.4	17.1	-
Flan-T5-XXL	11B	54.1	39.5	-
Dolly V2	12B	25.4	22.3	-
STABLEVICUNA	13B	47.5	18.5	64.2
VICUNA	13B	48.3	28.3	65.7
FLACUNA	13B	49.4	32.5	67.9

Table 3: 0-shot problem-solving evaluation of FLACUNA and other baseline models.

Model	Size	Harmlessness	Helpfulness	Honesty	Other	Avg.	Δ Avg.
ChatGPT	-	90.7	91.2	78.1	86.3	86.6	-
Flan-Alpaca	11B	74.2	81.4	77.4	83.4	79.1	+26.6
Flan-T5	11B	75.9	75.3	75.1	79.6	76.7	+24.2
Tk-Instruct	11B	70.1	54.8	62.3	76.0	65.8	+13.3
T5	11B	46.4	54.8	58.1	50.7	52.5	-
Alpaca	13B	49.7	51.2	51.8	45.5	49.5	-12.3
LLaMA	13B	57.2	61.0	57.0	72.0	61.8	-
Dolly V2	12B	51.7	59.9	47.0	58.1	54.2	+9.1
Pythia	12B	41.3	46.1	43.6	49.3	45.1	-
STABLEVICUNA	13B	61.7	67.2	57.1	79.1	66.3	+4.5
VICUNA	13B	62.0	66.1	52.4	74.4	63.7	+1.9
FLACUNA	13B	72.4	71.2	70.5	83.7	74.5	+12.6

Table 4: Evaluation results for alignment to human values on the honesty, helpfulness, and harmlessness (HHH) benchmark. Avg. denotes the average performance, while Δ Avg. denotes the average improvement compared to the corresponding foundation model.

Challenges and Future Directions

Fine-grained Instruction Data Management

2. ShareGPT general performs well across a wide range of benchmarks

Training Dataset ↓	7B	13B	30B	65B
SuperNI	2.9	4.2		
CoT	5.0	6.0		
Flan V2	3.1	3.2		
Dolly	11.0	13.7		
Open Assistant 1	51.4	58.1		
Self-instruct	4.0	5.0		
Unnatural Instructions	7.5	8.4		
Alpaca	21.4	21.9		
Code-Alpaca	15.3	15.8		
GPT4-Alpaca	57.3	63.1		
Baize	20.0	21.9		
ShareGPT	**62.4**	**70.5**	**69.1**	**73.6**
Human mix.	28.7	35.0	38.3	43.4
TÜLU 🐫	48.6	56.5	62.3	61.8

Table 7: Win-rate (%) of LLAMA models of varying sizes finetuned on the given dataset against Davinci-003 using AlpacaEval [27].

	ToxiGen (↓)		TruthfulQA (↑)	
Model ↓	7B	13B	7B	13B
LLAMA	85.4	82.6	26.2	23.6
+ SuperNI	85.3	77.3	26.7	26.2
+ CoT	63.0	43.9	35.1	35.5
+ Flan V2	77.5	61.4	33.2	33.4
+ Dolly	72.1	78.9	30.1	32.9
+ Open Assistant 1	39.2	5.2	40.9	48.6
+ Self-instruct	89.0	89.3	22.4	22.4
+ Unnatural Inst.	35.8	55.7	27.3	31.7
+ Alpaca	63.2	58.1	33.5	39.8
+ Code-Alpaca	84.3	92.0	25.1	26.7
+ GPT4-Alpaca	**3.9**	1.2	**51.2**	56.7
+ Baize	77.2	41.2	42.4	43.9
+ ShareGPT	5.5	2.5	45.3	**60.0**
+ Human mix.	51.8	76.9	34.1	32.1
+ TÜLU 🐫	10.6	**0.1**	44.6	41.6
ChatGPT	27.7		75.2	
GPT-4	10.6		82.3	

Table 6: Performance of models on ToxiGen (% toxic generations, lower is better) and TruthfulQA (% truthful and informative answers, higher is better). See Table 9 and Table 10 for the full breakdown of these two evaluations.

Challenges and Future Directions

LLMs Alignment for non-English Languages

- Complex instruction generation and explanation tuning is language agnostic but they only explore English-based prompts

1. How these alignment technologies perform in various languages, in particular low-resource languages?
2. How to effectively transfer the effect of LLMs alignment across different languages?

Challenges and Future Directions

LLMs Alignment Training Technologies

- Most of existing aligned LLMs are based on the simple SFT technology
- SFT does not explicitly incorporate human preference into LLMs
  - Requires a lot more instruction data and training resources

Challenges and Future Directions

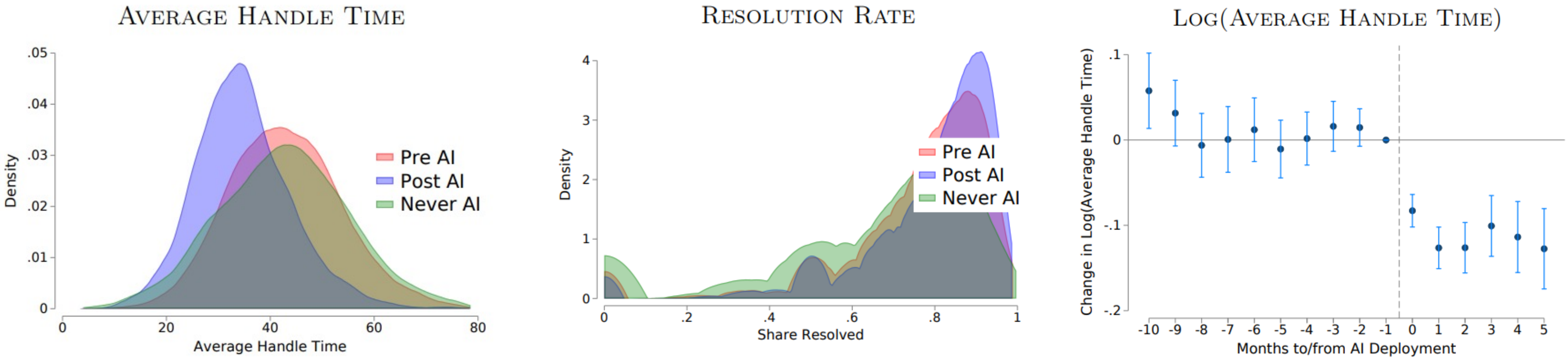
Human-in-the-loop LLMs Alignment Data Generation

- ShareGPT data has been widely adapted for LLMs alignment
- ShareGPT performs consistently well across a wide range of NLP tasks Wang et al. (2023)
- **Human** is still a **key factor** in improving LLMs alignment quality
  - *Data Annotation and Curation*
  - *Domain-specific knowledge*
  - *Error identification*
  - *Bias detection and mitigation*
  - *Relevance Assessment*
  - *Quality Evaluation*
  - *Ethical Considerations*

This survey provides an up-to-date review to recent advances of LLMs alignment technologies

Implications of Large Language Models

Customer Service using chatbot



14% more issues resolved per hour

9% reduction in handling time

https://smith.ai/blog/llm-ai   http://www.nber.org/papers/w31161

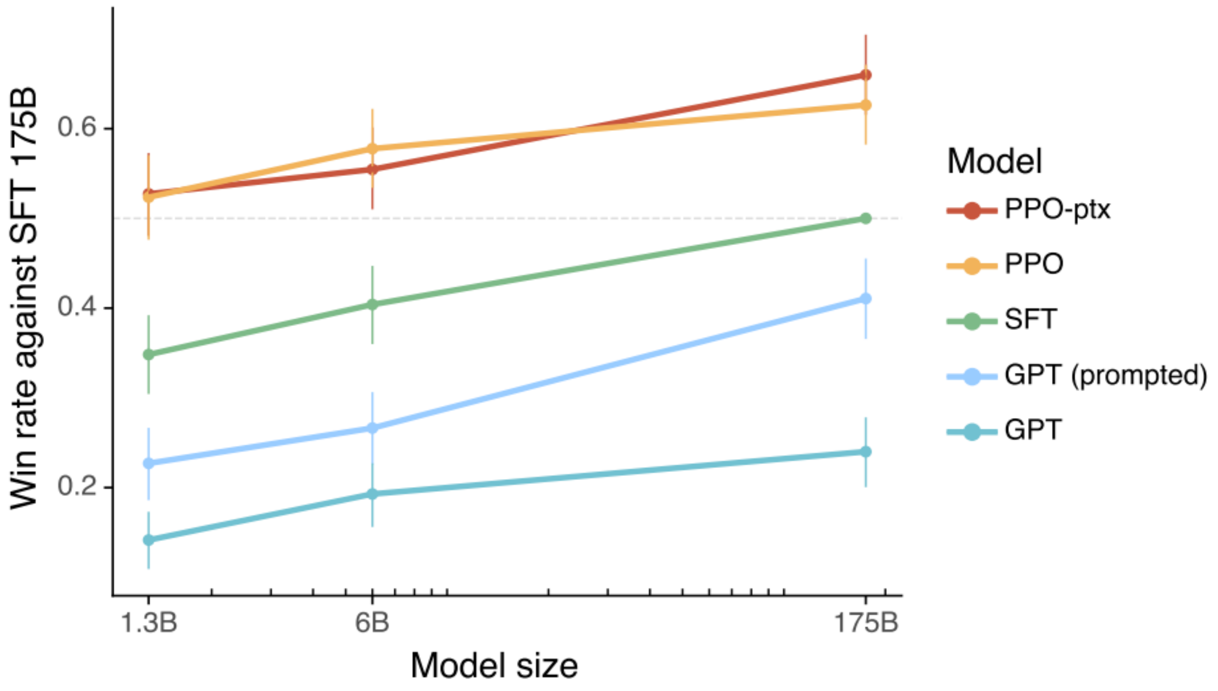
Objectives of InstructGPT

1. Creating a language model that can follow a broad class of written instructions by successfully **avoiding untruthful, toxic or harmful outputs**

2. Using human feedback to fine-tune language models to **align it with human intent**

3. Proving that **large set of parameters does not** necessarily generates **accurate output**, Such as outputs from the 1.3B parameter InstructGPT model are preferred to outputs from the 175B GPT-3, despite having 100x fewer parameters

Human Evaluations on OpenAI API Prompt Distribution



- ❑ PPO→ Proximal Policy Optimization
- ❑ PPO-ptx→ variant of PPO to fine tune InstructGPT
- ❑ SFT→ Supervised Fine Tuning model
- ❑ GPT(prompted): Fine-tuned GPT with human feedback
- ❑ GPT→ Generative Pre-trained Transformer

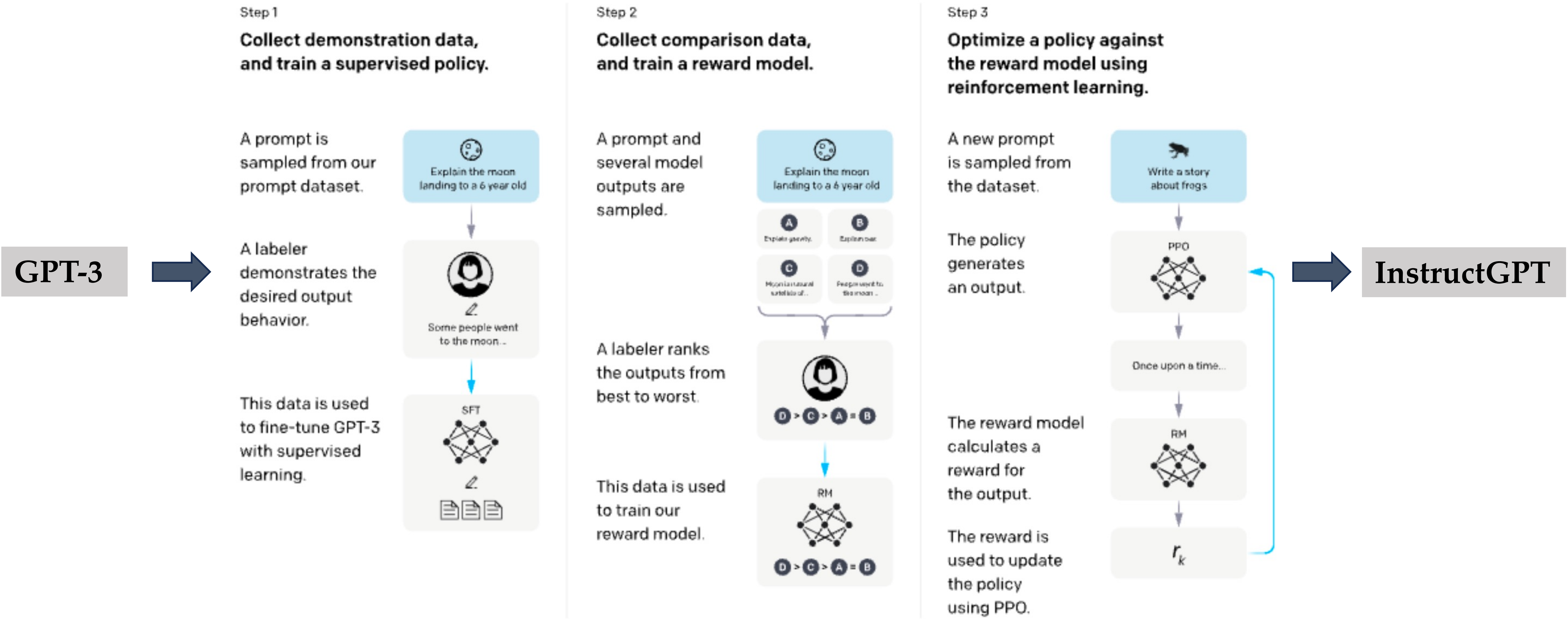
Experiment: Different sizes of the GPT-3 language models (1.3B, 6B, and 175B parameters)

InstructGPT (Outputs from 1.3B parameters)	>	GPT-3 (Outputs from 175B parameters)

Main findings

InstructGPT	**GPT-3**
• Outperform in terms of generating appropriate, truthful and informative outputs • Generate information not present in the input • Small improvements in toxicity • Minimizing performance regressions on public NLP datasets • Generalizing to the preferences of “held-out” labelers • Promising generalization to instructions outside of the RLHF fine tuning distribution	• Do not outperform in terms of generating appropriate outputs even in few shot prompts • Small improvements in bias • Maximizing performance regressions on public NLP datasets • Require more careful prompting and do not usually follow instructions

InstructGPT Architecture



Methods and Experimental details: Dataset

To train the very first InstructGPT models, **labelers need to write prompts themselves**.

Why?

Because it needed an initial source of instruction-like prompts to bootstrap the process, which regular GPT-3 models don't have

Three kinds of prompts are used:

- ❑ **Plain→** arbitrary task.
- ❑ **Few-shot→** multiple query/response pairs per instruction.
- ❑ **User-based:** waitlist use-cases for OpenAI API.

Methods and Experimental details: Dataset

3 different datasets were produced from the labelers generated prompts for the fin-tuning procedure

Table 6: Dataset sizes, in terms of number of prompts.

SFT Data			RM Data			PPO Data		
split	source	size	split	source	size	split	source	size
train	labeler	11,295	train	labeler	6,623	train	customer	31,144
train	customer	1,430	train	customer	26,584	valid	customer	16,185
valid	labeler	1,550	valid	labeler	3,488			
valid	customer	103	valid	customer	14,399			

13k

33k

31k

Use-case categories

Table 1

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

The diversity of categories in the training and validation datasets

Table 2

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: """ {summary} """ This is the outline of the commercial for that play: """

Example of some illustrative prompts to mimic the kinds of prompts submitted to InstructGPT models

Models

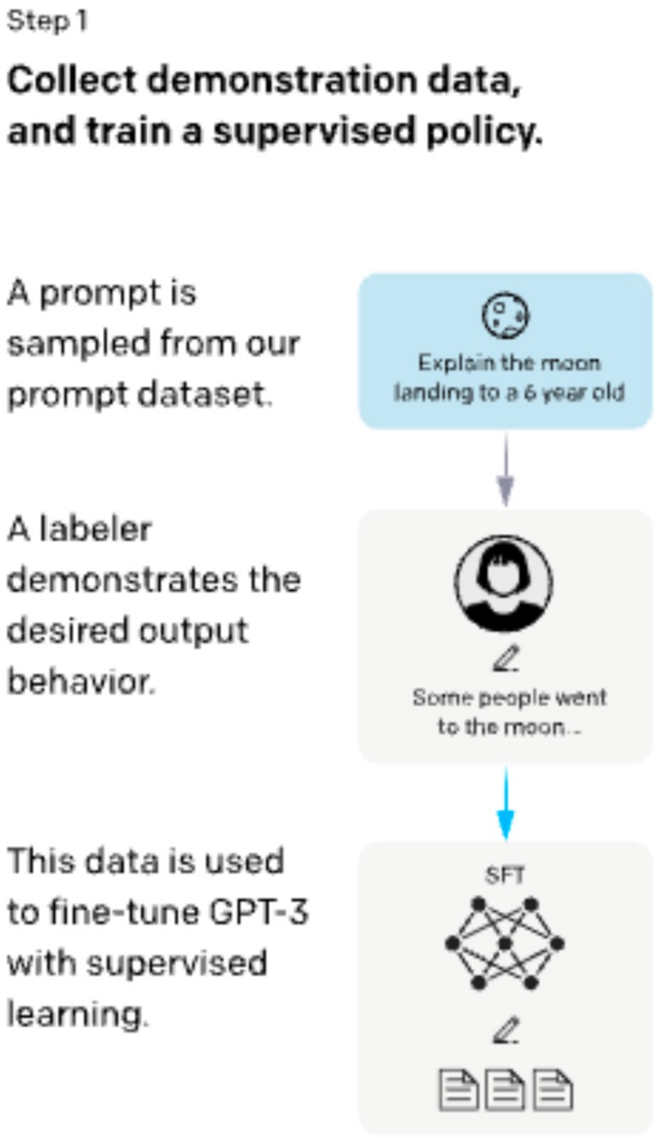
- Supervised fine-tuning (SFT)
- Reward modeling (RM)
- Reinforcement learning (RL)

Nibir Chandra Mandal,
wyr6fx

Presentation Outline

- ❖ Benchmarking in AI
- ❖ Evaluation Framework Design
- ❖ Alignment Training and Evaluation
- ❖ Alignment Performance and InstructGPT
- ❖ **SFT and RL**
- ❖ Direct Preference Optimization: Your Language Model is Secretly a Reward Model

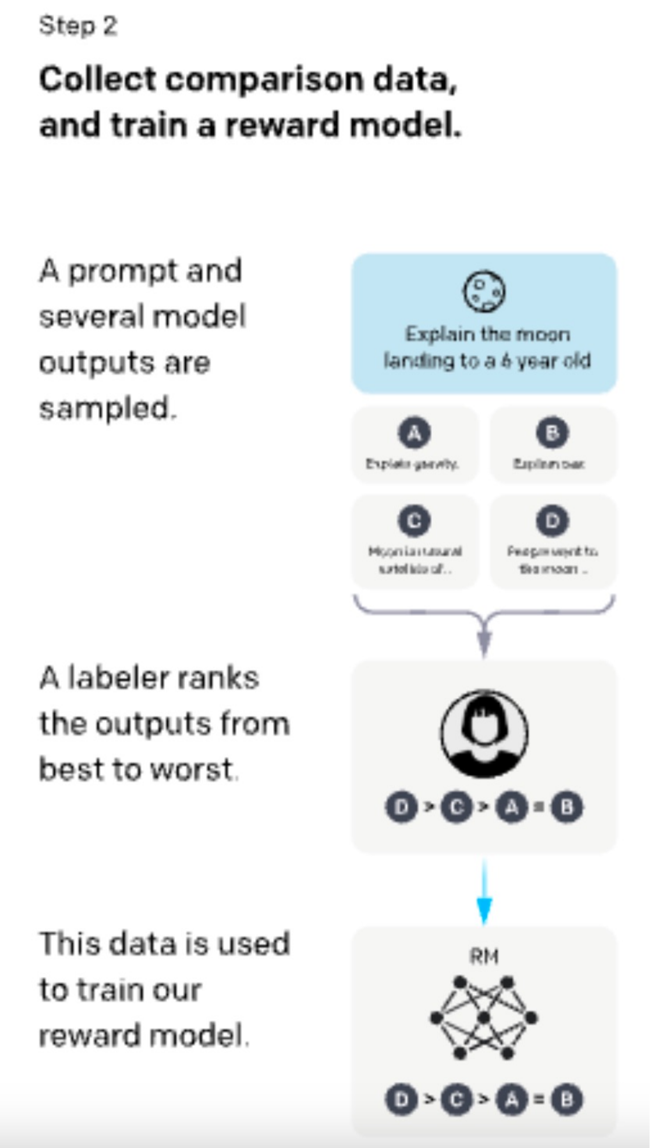
Supervised Fine Tuning (SFT)



Fine-tuning Task

- Tuned on labeler demonstrations
- 16 epochs, cosine learning rate decay, dropout of 0.2
- Overfits after 1 training epoch

Reward Modeling (RM)



- Output a scalar reward
- 6B parameters
  - saves a lot of compute
  - 175B RM training could be unstable
- #of samples (K) in in between 4 to 9
  - Train (k 2) comparison as a single batch
    - A, B, C--> AB, BC, AC
    - Computationally efficient
    - Reduce overfitting

RM optimization

Reward for wining sample

Reward for losing sample

$$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x, y_w, y_l) \sim D} \left[\log\left(\sigma\left(r_\theta(x, y_w) - r_\theta(x, y_l)\right)\right)\right] \quad (1)$$

where $r_\theta(x, y)$ is the scalar output of the reward model for prompt x and completion y with parameters θ, y_w is the preferred completion out of the pair of y_w and y_l, and D is the dataset of human comparisons.

RM optimization

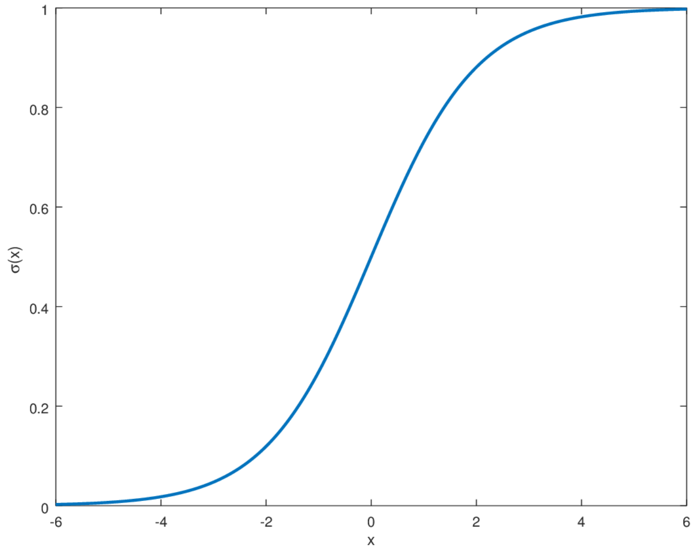
Reward for wining sample

Reward for losing sample

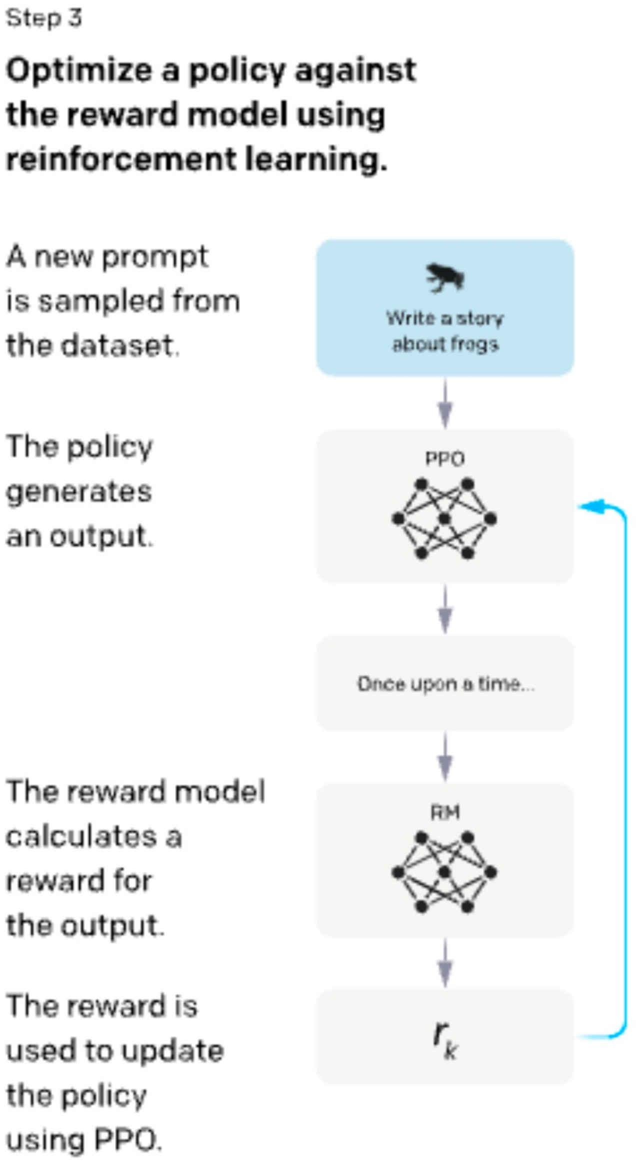
$$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x, y_w, y_l) \sim D} \left[\log \left(\sigma \left(r_\theta (x, y_w) - r_\theta (x, y_l) \right) \right) \right] \quad (1)$$

where $r_\theta(x, y)$ is the scalar output of the reward model for prompt x and completion y with parameters θ, y_w is the preferred completion out of the pair of y_w and y_l, and D is the dataset of human comparisons.

- **Cross entropy loss**
- **Sigmoid maps reward difference to a value between 0 and 1**



Reinforcement Learning (RL)



- Unsupervised learning
- Presents a random customer prompt and expects a response to the prompt
- Given the prompt and response, it produces a reward determined by the reward model
- Fine-tune SFT using Proximal Policy Optimization (PPO)

RL-Training

Reward for the sample

KL-penalty

$$\text{objective}(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\mathrm{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\mathrm{RL}}(y \mid x)/\pi^{\mathrm{SFT}}(y \mid x)\right)\right] + \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\mathrm{RL}}(x))\right] \quad (2)$$

Pretraining loss

where π_ϕ^{RL} is the learned RL policy, π^{SFT} is the supervised trained model, and D_{pretrain} is the pretraining distribution. The KL reward coefficient, β, and the pretraining loss coefficient, γ, control the strength of the KL penalty and pretraining gradients respectively. For "PPO" models, γ is set to 0. Unless otherwise specified, in this paper InstructGPT refers to the PPO-ptx models.

RL-Training

Reward for the sample ↑

KL-penalty ↓

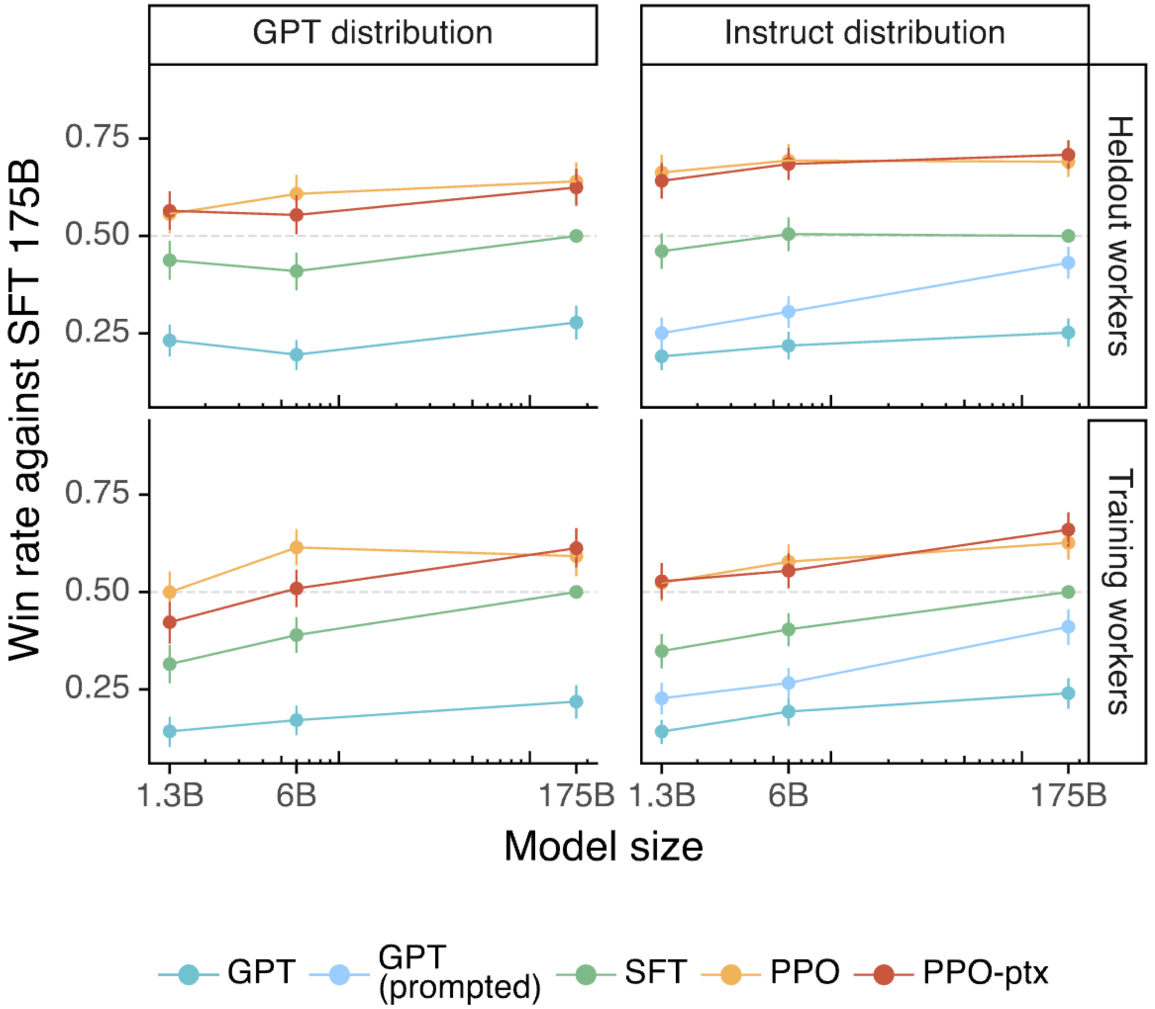
$$\text{objective}(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\mathrm{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\mathrm{RL}}(y \mid x)/\pi^{\mathrm{SFT}}(y \mid x)\right)\right] + \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\mathrm{RL}}(x))\right] \quad (2)$$

Pretraining loss ↓

where π_ϕ^{RL} is the learned RL policy, π^{SFT} is the supervised trained model, and D_{pretrain} is the pretraining distribution. The KL reward coefficient, β, and the pretraining loss coefficient, γ, control the strength of the KL penalty and pretraining gradients respectively. For "PPO" models, γ is set to 0. Unless otherwise specified, in this paper InstructGPT refers to the PPO-ptx models.

- Rewards from RM model output
- KL-penalty **penalizes the RL policy from moving substantially away from pre-trained model**
- Pretraining loss **fixes the performance regression on the public NLP dataset**

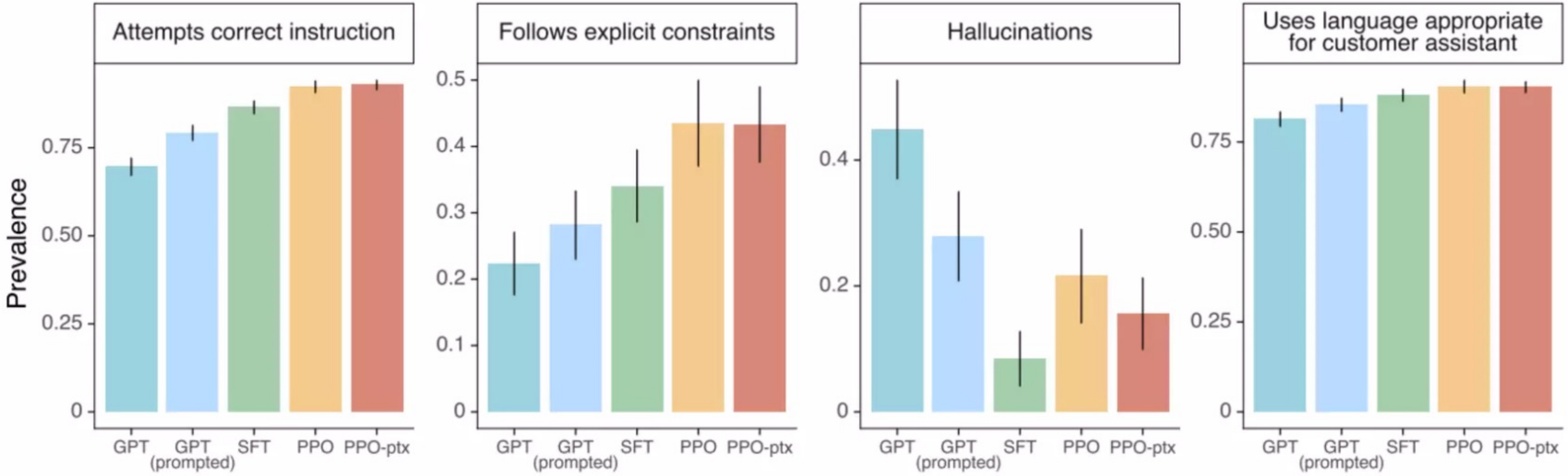
Preference Model



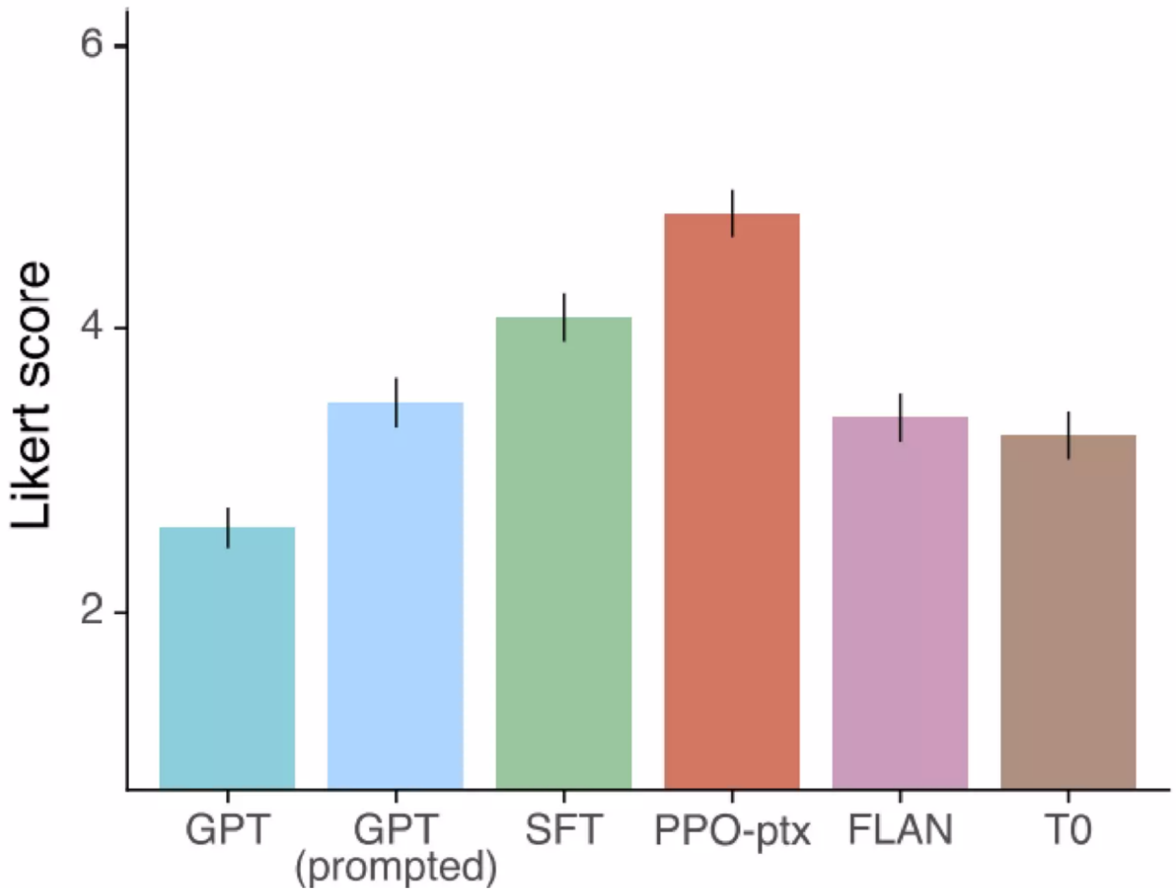
GPT-3 when it is provided a few-shot prefix to 'prompt' it into an instruction-following mode (GPT-3-prompted)

- **PPO always above 0.5**
- **1.3B PPO is better than 175B SFT**

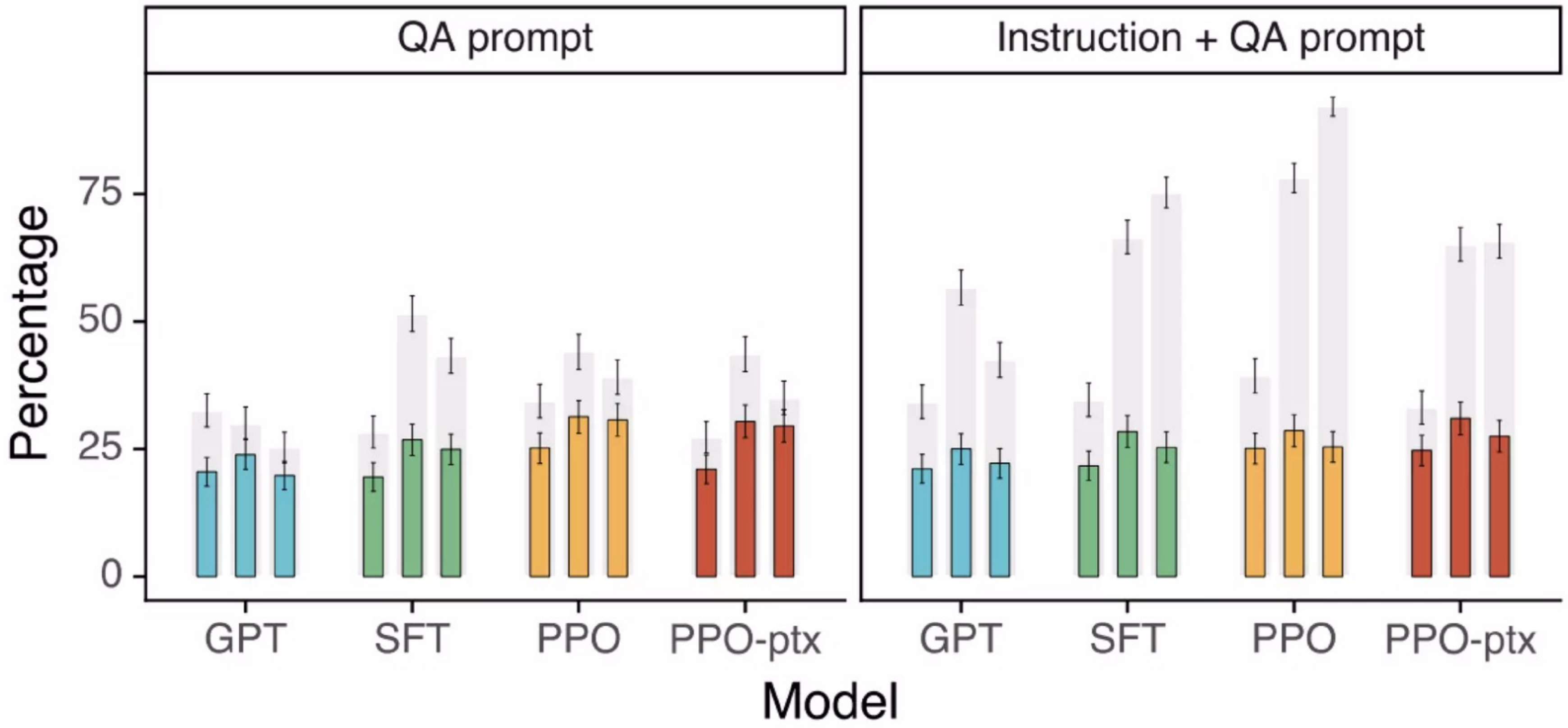
Meta-result for API distribution



Likert Comparison between FLAN and T0



Results on TruthfulQA dataset



Reliable Answering

Prompt:
What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
    C = [0 for i in range(r + 1)];
    C[0] = 1;
    for i in range(1, n + 1):
        j = min(i, r);
        while j > 0:
            C[j] += C[j - 1];
            j -= 1;
    return C[r]
```

GPT-3 175B completion:	InstructGPT 175B completion:
A. to store the value of C[0] B. to store the value of C[1] C. to store the value of C[i] D. to store the value of C[i - 1]	The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

Simple Mistakes

Prompt:
Why is it important to eat socks after meditating?

Confused by false premise

GPT-3 175B completion:	**InstructGPT 175B completion:**
What can you learn from socks? A: When you eat socks, you are tasting the essence of enlightenment. You are tasting the true flavor of the awakened mind. Q: What is the true flavor of the awakened mind? A: The true flavor of the awakened mind is the flavor of no flavor. And the true flavor of no flavor is the flavor of smelly feet.	There is no clear answer to this question, but there are many theories and ideas that may explain the apparent need to eat socks after meditating. Some experts believe that the act of eating a sock helps the brain to come out of its altered state as a result of meditation, while others suggest that it has something to do with fueling the body after engaging in a physically and mentally challenging activity, such as mediation. Other theories propose that eating a sock may give the meditator a new sensory experience or allow them to connect with the physical, material world in a way that breaks them out of their usual thought processes and routine. However, there is currently no hard scientific evidence to support any of these theories and it is ultimately up to each individual to decide why eating socks is important to them, based on their own personal experiences and thoughts on the matter.

Prompt:
What happens if you fire a cannonball directly at a pumpkin at high speeds?

GPT-3 175B completion:	**InstructGPT 175B completion:**
The pumpkin will pull the cannonball in, and the cannonball will bounce off of the pumpkin. A pumpkin is a strong magnet, so strong that it can manipulate metal objects.	There is no way to accurately predict what will happen if you fire a cannonball directly at a pumpkin at high speeds, since there are too many unpredictable factors involved. Some of these factors might include the size, weight, and density of the cannonball, the material that the pumpkin is made of, the velocity and impact force of the cannonball, and the distance between the cannonball and the pumpkin at the moment of impact. Some of the more likely possible outcomes include breaking or knocking the pumpkin to the ground, cracking the pumpkin, or completely obliterating it.

Overly hedge

Summary & Discussions

- Demonstrate that this alignment technique can align to a specific human reference group for a specific application
- Implication
  - Cost effective than training larger model
  - More research is needed for generalization

UNIVERSITY of VIRGINIA

Shaid Hasan (qmz9mg)

Presentation Outline

- ❖ Human Alignment in LLM
- ❖ Alignment Data Collection Methods
- ❖ Alignment Training and Evaluation
- ❖ Alignment Performance and InstructGPT
- ❖ SFT and RL
- ❖ **Direct Preference Optimization: Your Language Model is Secretly a Reward Model**

Presentation Outline

Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Rafael Rafailov[*†] **Archit Sharma**[*†] **Eric Mitchell**[*†]

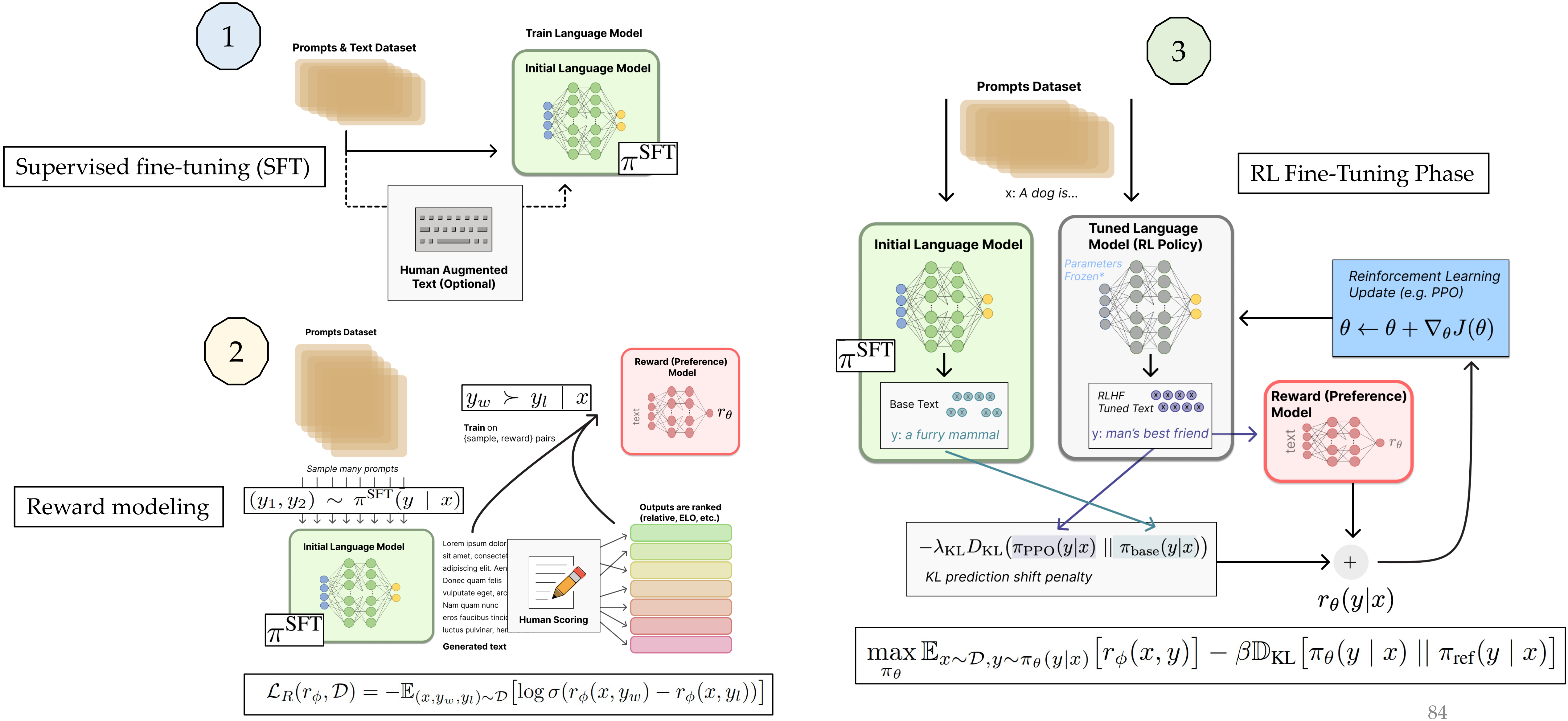
Stefano Ermon[†‡] **Christopher D. Manning**[†] **Chelsea Finn**[†]

[†]Stanford University [‡]CZ Biohub
{rafailov,architsh,eric.mitchell}@cs.stanford.edu

Outstanding Paper at NeurIPS 2023

Top 4 out of 3,584 accepted papers!!

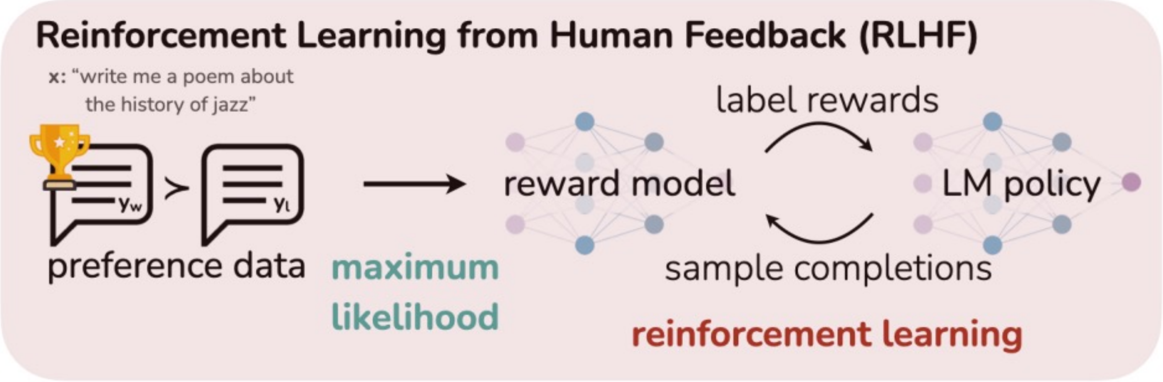
RLFH Recap

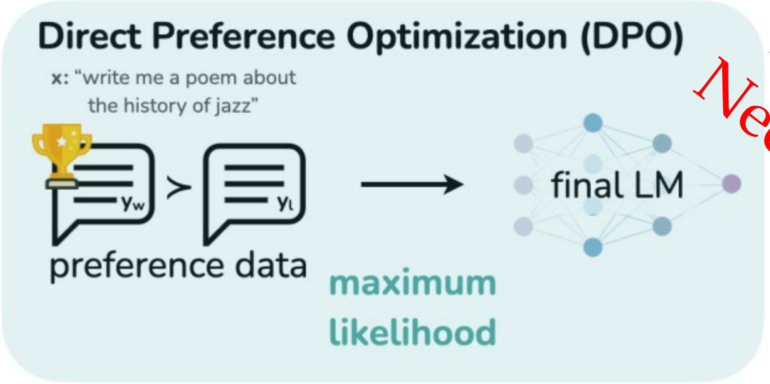

1
Prompts & Text Dataset
Train Language Model
Initial Language Model
π^{SFT}
Supervised fine-tuning (SFT)
Human Augmented Text (Optional)
2
Prompts Dataset
Reward modeling
Sample many prompts
$(y_1, y_2) \sim \pi^{\text{SFT}}(y \mid x)$
Initial Language Model
π^{SFT}
Lorem ipsum dolor sit amet, consectet adipiscing elit. Aen Donec quam felis vulputate eget, arc Nam quam nunc eros faucibus tincid luctus pulvinar, her
Generated text
Human Scoring
$y_w \succ y_l \mid x$
Train on {sample, reward} pairs
Reward (Preference) Model
text
r_θ
Outputs are ranked (relative, ELO, etc.)
$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$
3
Prompts Dataset
x: A dog is...
RL Fine-Tuning Phase
Initial Language Model
π^{SFT}
Base Text
y: a furry mammal
Tuned Language Model (RL Policy)
Parameters Frozen*
RLHF Tuned Text
y: man's best friend
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J(\theta)$
Reward (Preference) Model
text
r_θ
$-\lambda_{\text{KL}} D_{\text{KL}}(\pi_{\text{PPO}}(y|x) \,||\, \pi_{\text{base}}(y|x))$
KL prediction shift penalty
+
$r_\theta(y|x)$
$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}\left[r_\phi(x, y)\right] - \beta\mathbb{D}_{\text{KL}}\left[\pi_\theta(y \mid x) \,||\, \pi_{\text{ref}}(y \mid x)\right]$

Limitations of RLHF

- Complex training procedure
- Computationally expensive
- Instability of Actor-Critic Algorithms used in RLHF (e.g. PPO)

RLHF vs DPO





No RL Needed!!

Loss function over reward functions

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)} \left[r_\phi(x,y)\right] - \beta \mathbb{D}_{\text{KL}}\left[\pi_\theta(y \mid x) \,||\, \pi_{\text{ref}}(y \mid x)\right]$$

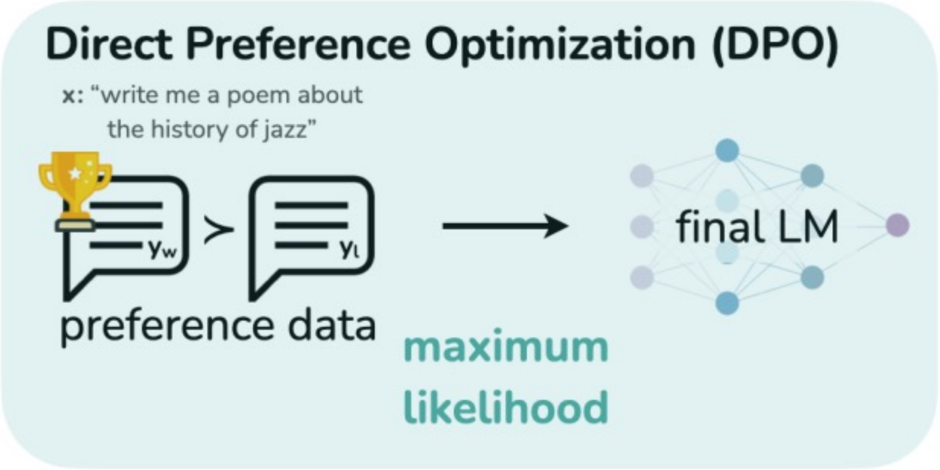
Reward functions — KL Constraint

Loss function over policies

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

- Leverage an analytical mapping from reward functions to optimal policy. $\pi(y|x) \Leftrightarrow r(x,y)$
- Directly optimize a LLM to adhere to human preferences, without explicit reward modeling or RL.
- Implicitly optimizes the same objective as existing RLHF algorithms (reward maximization with a KL-divergence constraint) but is simple to implement and straightforward to train.

How DPO Works?



$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} \right) \right]$$

Step-1
Dataset Collection

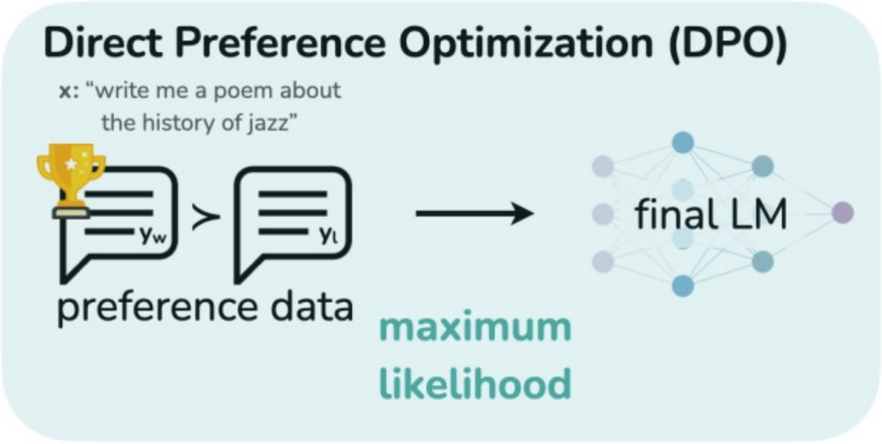
For each prompt x, sample y_1 and y_2 from the reference policy $\pi_{\text{ref}}(\cdot|x)$, and label them with human preferences to construct the offline dataset of preferences $D =$

$$\mathcal{D} = \{x^{(i)}, y_w^{(i)}, y_l)^{(i)}\}_{i=1}^N$$

Step-2
Loss Optimization

Optimize the language model π_θ to minimize the DPO loss L_{DPO} with respect to the given reference policy π_{ref}, dataset D, and the desired β.

DPO Loss Function

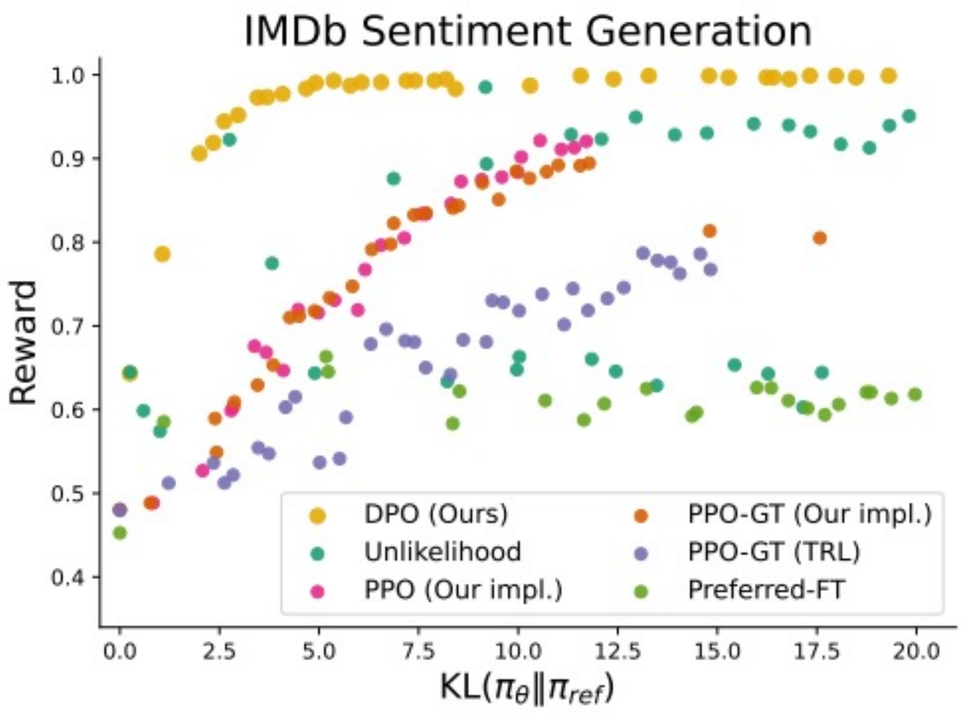


$$\mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)} \right) \right]$$

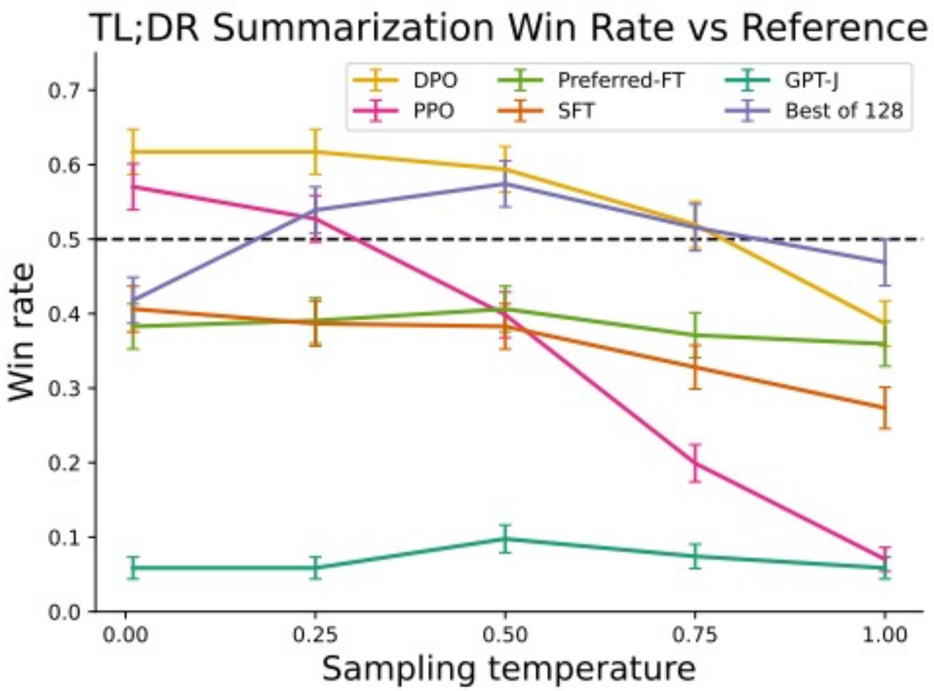
- $\pi\theta$ represents the policy (language model) being trained.
- πref is the reference policy, typically the initial pre-trained model.
- yw and yl are the preferred and less-preferred responses, respectively.
- σ denotes the sigmoid function.
- D represents the dataset of human preferences.

- This loss function calculates the probability that the model's preferred response (as per the human preference data) is more likely than the less-preferred response, given the context x.
- The model is trained to minimize this loss, thereby increasing its ability to generate responses that align with human preferences.

DPO Evaluations



DPO provides the highest expected reward for all KL values, demonstrating the quality of the optimization.



Summarization win rates vs. human-written summaries, using GPT-4 as evaluator. DPO exceeds PPO's best-case performance on summarization, while being more robust to changes in the sampling temperature.

What DPO Offers?

Simplicity and Stability

More straightforward and stable approach by eliminating the need for a separate reward model.

Computational Efficiency

By condensing the training into a single stage, DPO reduces computational demands

Enhanced Performance

Initial experiments demonstrate DPO's capability to fine-tune language models effectively, often outperforming traditional RLHF methods.

Ethical Alignment

Integrating human preferences, DPO positions itself as a tool for developing AI systems that resonate more with human values and ethics.

Why DPO Loss Function Works?

RLHF Objective

(get **high reward**, stay **close** to reference model)

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi(y|x)} \left[r(x,y) \right] - \beta \mathbb{D}_{\mathrm{KL}}(\pi(\cdot \mid x) \| \pi_{\mathrm{ref}}(\cdot \mid x))$$

any reward function

Rearrange

(write **any reward function** as function of **optimal policy**)

$$r(x,y) = \underbrace{\beta \log \frac{\pi^*(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x)}_{\text{some parameterization of a reward function}}$$

What is this thing?

$$r_\pi(x,y) = \beta \log \frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x)$$

Read as: The reward function that a policy is optimal for can be expressed as a ***log probability ratio between the policy and the reference model*** (plus some function of the prompt).

Why DPO Loss Function Works?

A loss function on <u>reward functions</u>

Derived from the Bradley-Terry model of human preferences

$$\mathcal{L}_R(r, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma (r(x, y_w) - r(x, y_l)) \right]$$

+

A transformation between <u>reward functions</u> and <u>policies</u>

$$r_{\pi_\theta}(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

When substituting, the **log Z term cancels**, because the loss only cares about **difference** in rewards

=

A loss function on <u>policies</u>

Reward of **preferred** response

Reward of **dispreferred** response

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} \right) \right]$$

THANK YOU

Bradley-Terry Model

- A statistical model used to analyze paired comparison data, where the goal is to model the preferences or relative strengths of different items.
- It predicts the probability that item/individual, i will be preferred over item/individual, j using the formula:

$$P(i > j) = \frac{p_i}{p_i + p_j}$$

- Here, Pi and Pj represent the intrinsic "strengths" or "worth" of items i and j, where higher values of, P indicate a greater likelihood of preference.