MORE LLM PREFERENCE ALIGNMENT

Team 6
Fengyu Gao, Shunqiang Feng, Wei Shen, Zihan Zhao

Is DPO Superior to PPO for LLM Alignment? A Comprehensive Study

Fengyu Gao (wan6jj)

Introduction

- RLHF is a crucial step for LLM alignment.
- DPO, as a simplified RLHF method, is often preferred and reported to have strong performances.
- This paper:
  - **Is DPO truly superior to PPO in the RLHF domain?**
  - **Can the performance of PPO be improved in RLHF benchmarks?**

PPO Formulation

- Step 1: Train a reward model

$$\mathcal{L}_R(r_\phi) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}}\left[\log\sigma(r_\phi(\mathbf{x},\mathbf{y}_w) - r_\phi(\mathbf{x},\mathbf{y}_l))\right]$$

  - Maximize rewards on accepted answers
  - Minimize rewards on rejected answers
- Step 2: Reinforcement Learning
  - Maximize KL-regularized reward

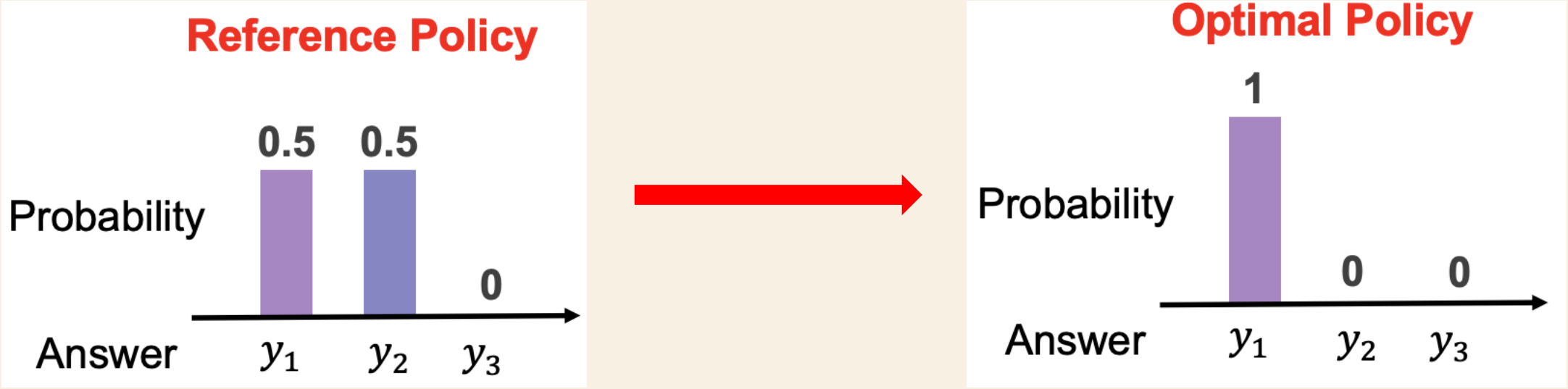
$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta\log\frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]$$

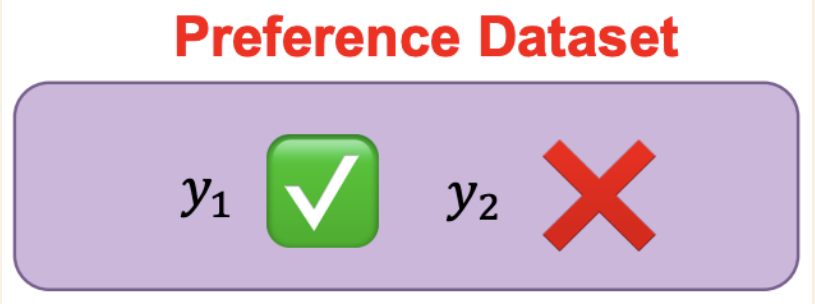
DPO Formulation

- Maximize log-likelihood on accepted answers
- Minimize log-likelihood on rejected answers

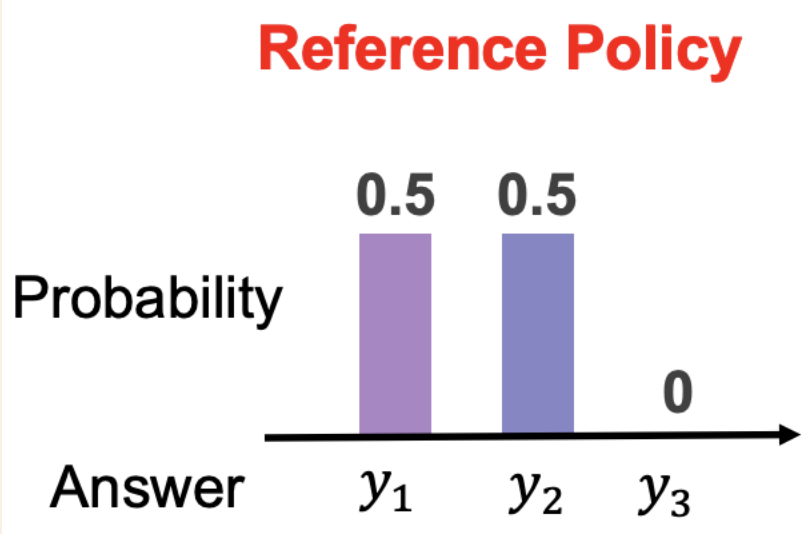
$$\mathcal{L}_{\text{DPO}}(\pi_\theta) = -\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \left(\log \frac{\pi_\theta(\mathbf{y}_w \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w \mid \mathbf{x})} - \log \frac{\pi_\theta(\mathbf{y}_l \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l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

Understanding the Limitation of DPO

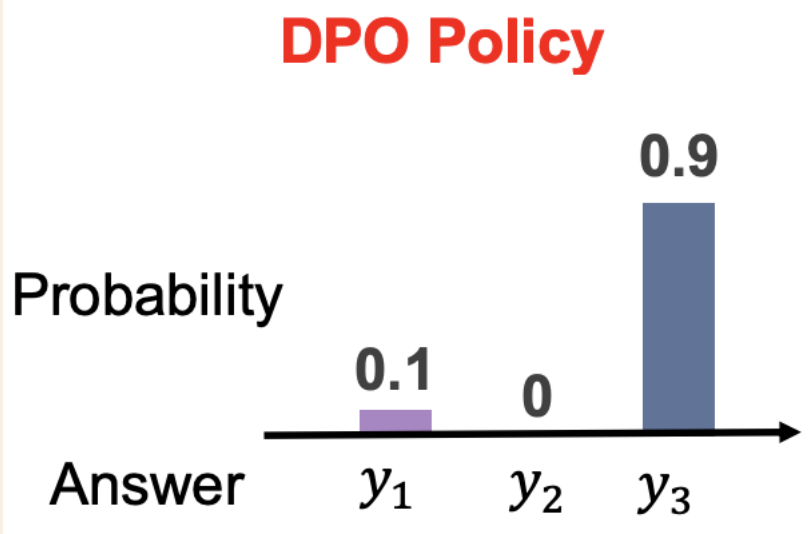


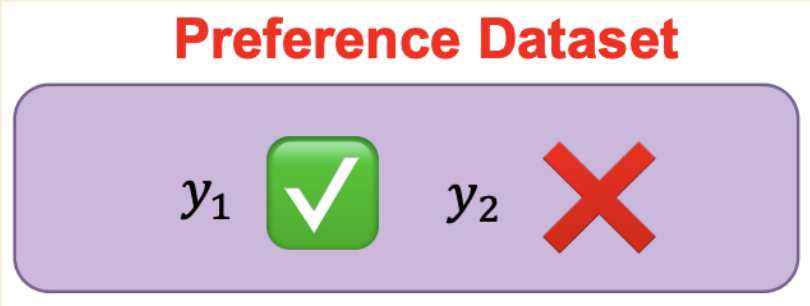


Understanding the Limitation of DPO



DPO

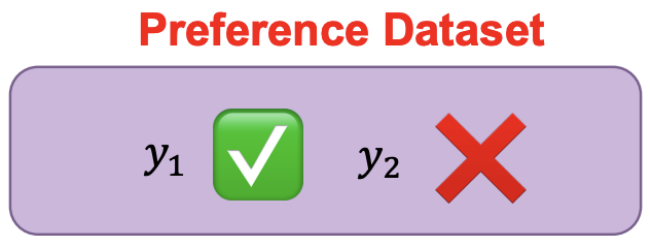


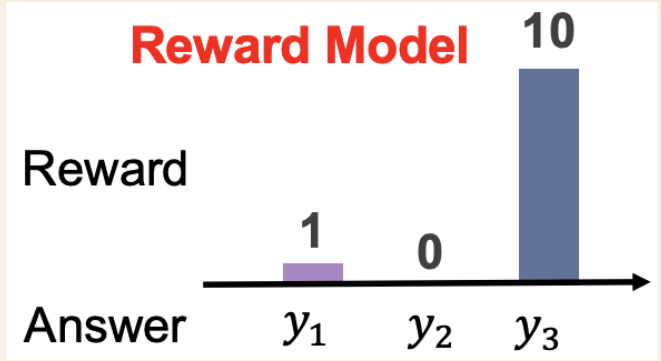


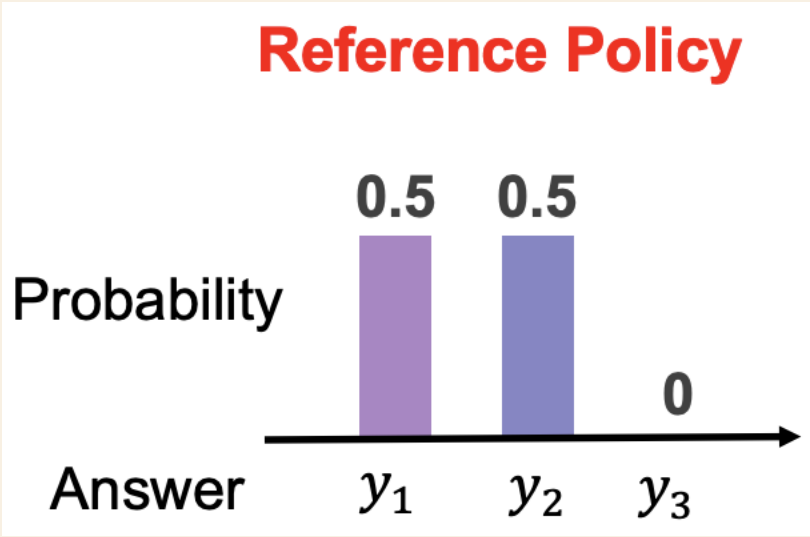
$$\mathcal{L}_{\mathrm{DPO}}(\pi_\theta) = \log(1 + \left(\frac{\pi_\theta(y_2 \mid x)}{\pi_\theta(y_1 \mid x)}\right)^{\beta})$$

DPO fails to find the optimal policy because there is an Out-of-Distribution answer!

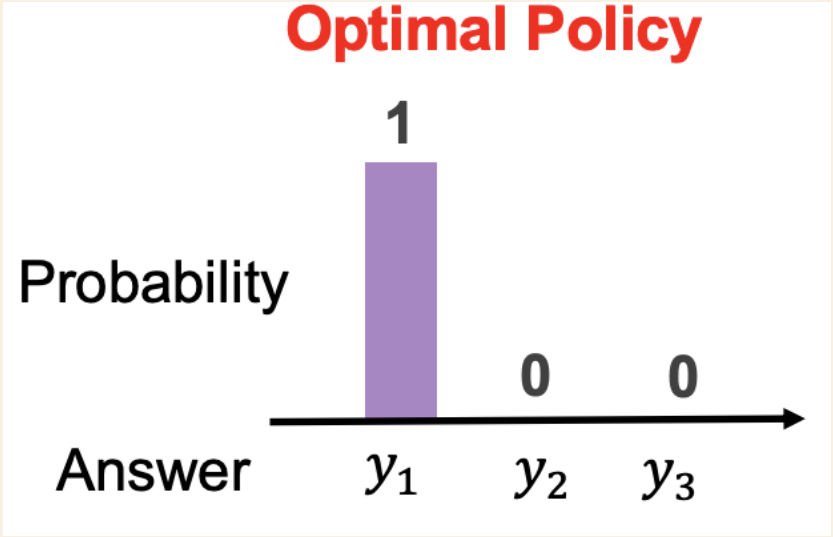
Understanding the Limitation of DPO







PPO



$$J_r(\pi_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\mathbf{y}\sim\pi_\theta}\left[r(\mathbf{x},\mathbf{y}) - \beta \log \frac{\pi_\theta(\mathbf{y}\mid\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}\mid\mathbf{x})}\right]$$

Experiments on SafeRLHF

	ΔHelp. ↑	Harm. ↓	S.R. ↑
SFT (Alpaca)	-2.62	1.50	41.6%
PPO	1.69	-12.08	99.5%
+ SFT (Safe)	4.47	-12.33	99.6%
DPO	-4.19	-0.97	55.4%
+ SFT (Safe)	-1.62	-3.50	71.8%
+ filter dual-unsafe	2.46	-4.88	80.8%
+ filter dual-safe	-2.86	-6.82	95.8%
DPO Iter.1	-3.22	-5.23	86.7%
DPO Iter.2	-3.27	-8.83	99.7%
DPO Iter.3	-3.26	-10.21	99.9%
DPO Iter.4	-2.96	-11.07	99.9%

Solution 1: Additional SFT on the preference dataset

Experiments on SafeRLHF

- Generate new responses with SFT (Safe)
- Use a learned reward model for preference labeling
- Repeat this process and iteratively set the reference model as the latest DPO model

	ΔHelp. ↑	Harm. ↓	S.R. ↑
SFT (Alpaca)	-2.62	1.50	41.6%
PPO	1.69	-12.08	99.5%
+ SFT (Safe)	4.47	-12.33	99.6%
DPO	-4.19	-0.97	55.4%
+ SFT (Safe)	-1.62	-3.50	71.8%
+ filter dual-unsafe	2.46	-4.88	80.8%
+ filter dual-safe	-2.86	-6.82	95.8%
DPO Iter.1	-3.22	-5.23	86.7%
DPO Iter.2	-3.27	-8.83	99.7%
DPO Iter.3	-3.26	-10.21	99.9%
DPO Iter.4	-2.96	-11.07	99.9%

Solution 2: Online generation and scoring with a trained reward model.

Experiments on SafeRLHF

	ΔHelp. ↑	Harm. ↓	S.R. ↑
SFT (Alpaca)	-2.62	1.50	41.6%
PPO	1.69	-12.08	99.5%
+ SFT (Safe)	4.47	-12.33	99.6%
DPO	-4.19	-0.97	55.4%
+ SFT (Safe)	-1.62	-3.50	71.8%
+ filter dual-unsafe	2.46	-4.88	80.8%
+ filter dual-safe	-2.86	-6.82	95.8%
DPO Iter.1	-3.22	-5.23	86.7%
DPO Iter.2	-3.27	-8.83	99.7%
DPO Iter.3	-3.26	-10.21	99.9%
DPO Iter.4	-2.96	-11.07	99.9%

Solution 3: Filter out controversy and noisy preference pairs

Key Factors to PPO for RLHF

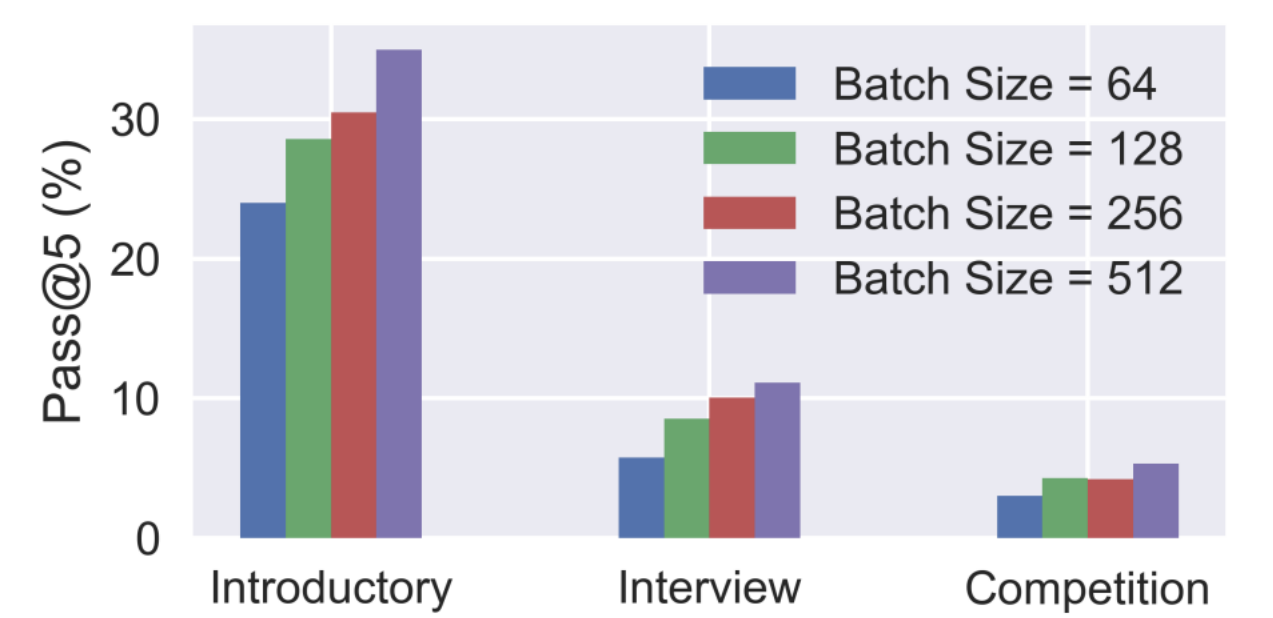


Figure 2. Performance of PPO on APPS dataset under different batch sizes. The base LLM is CodeLlma-13B. "Introductory", "Interview" and "Competition" represent three levels of difficulty.

Insight 1: A LARGE Batch Size

Key Factors to PPO for RLHF

Task	HH-RLHF	APPS			CodeContest		
Metric	OpenAssaint Reward	Intro. pass@5	Inter. pass@5	Comp. pass@5	pass@10	pass@100	pass@1k
SFT	0.532	38.6%	10.1%	3.9%	0.9%	4.3%	12.0%
baseline PPO	0.706	18.0%	2.4%	1.1%	4.3%	6.0%	7.7%
+ Adv.Norm.	0.716	38.1%	11.4%	4.6%	6.8%	9.4%	15.4%
+ Large.Batch.	0.716	42.3%	14.6%	7.5%	5.1%	12.8%	19.6%
+ Ref.EMA	**0.718**	44.4%	18.0%	9.1%	6.8%	13.7%	21.4%

Insight 2: Add Advantage Normalization

Key Factors to PPO for RLHF

Task	HH-RLHF	APPS			CodeContest		
Metric	OpenAssaint Reward	Intro. pass@5	Inter. pass@5	Comp. pass@5	pass@10	pass@100	pass@1k
SFT	0.532	38.6%	10.1%	3.9%	0.9%	4.3%	12.0%
baseline PPO	0.706	18.0%	2.4%	1.1%	4.3%	6.0%	7.7%
+ Adv.Norm.	0.716	38.1%	11.4%	4.6%	6.8%	9.4%	15.4%
+ Large.Batch.	0.716	42.3%	14.6%	7.5%	5.1%	12.8%	19.6%
+ Ref.EMA	**0.718**	44.4%	18.0%	9.1%	6.8%	13.7%	21.4%

Insight 3: Update the reference model with exponential moving average during training

Wei Shen (zyy5hb)

Tülu 3: Pushing Frontiers in Open Language Model Post-Training

Background

- Post-training – the collection of techniques including instruction tuning, reinforcement learning from human feedback, and other types of finetuning – has become a **crucial step** in building frontier language models.
- Fully **open source** counterparts (e.g., Tülu 2 (Ivison et al., 2023) and Zephyr-β (Tunstall et al., 2023)) often rely on simpler-to-implement and cheaper pipelines and **have become outdated** on many metrics.
- To close the gap between open and closed post training, this paper introduces Tülu 3, a family of **open state-of-the-art post-trained models**, alongside all of the data, training recipes, code, infrastructure, and evaluation framework.
- Best performing recipe yields Tülu 3 models that **outperform** the state-of-the-art post-trained open-weight models of the same size such as Llama 3.1 Instruct (Dubey et al., 2024) or Mistral-Instruct (Mistral AI,2024), and at the large 70B size Tülu matches the offerings of closed providers such as Claude 3.5 Haiku and GPT-4o mini. Furthermore, at 405B size our model performs competitively against DeepSeek v3 (DeepSeek-AI et al., 2024) and GPT 4o (11-24).

Tülu 3 Overview

- Key components of Tülu 3:
  - Tülu 3 Data: new permissively licensed training datasets targeting core skills
  - Tülu 3 Eval: an evaluation suite and tools to establish clear performance goals and guide improvement through training stages
  - Tülu 3 Recipe: an advanced multi-stage training pipeline incorporating
    - new algorithmic advancements in reinforcement learning,
    - cutting-edge infrastructure,
    - rigorous experimentation to optimize data mixes, methods, and parameters across various training stages.

Tülu 3 Overview

- The Tülu 3 training recipe involves multiple stages, with each stage building upon the previous model and focusing on different types of data – namely, prompt-completion instances for **supervised finetuning**, preferences for **preference tuning**, or verifiable rewards for **reinforcement learning**.

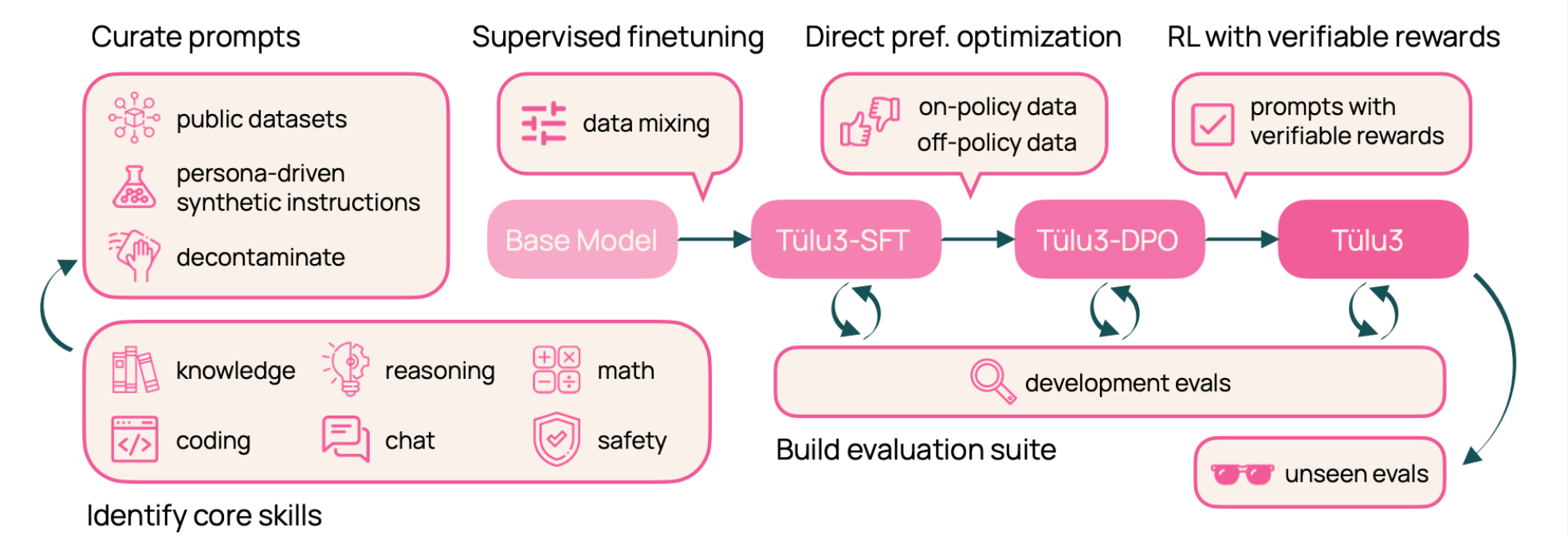


Figure 1 An overview of the TÜLU 3 recipe. This includes: data curation targeting general and target capabilities, training strategies and a standardized evaluation suite for development and final evaluation stage.

Tülu 3 Data

- Focusing on core skills of knowledge recall, reasoning, mathematics, coding, instruction following, general chat, and safety.
- New datasets released with Tülu 3 are **color-coded** for emphasis.

Category	Prompt Dataset	Count	# Prompts used in SFT	# Prompts used in DPO	Reference
General	Tülu 3 Hardcoded[↑]	24	240	–	–
	OpenAssistant[1,2,↓]	88,838	7,132	7,132	Köpf et al. (2024)
	No Robots	9,500	9,500	9,500	Rajani et al. (2023)
	WildChat (GPT-4 subset)[↓]	241,307	100,000	100,000	Zhao et al. (2024)
	UltraFeedback[α,2]	41,635	–	41,635	Cui et al. (2023)
Knowledge Recall	FLAN v2[1,2,↓]	89,982	89,982	12,141	Longpre et al. (2023)
	SciRIFF[↓]	35,357	10,000	17,590	Wadden et al. (2024)
	TableGPT[↓]	13,222	5,000	6,049	Zha et al. (2023)
Math Reasoning	Tülu 3 Persona MATH	149,960	149,960	–	–
	Tülu 3 Persona GSM	49,980	49,980	–	–
	Tülu 3 Persona Algebra	20,000	20,000	–	–
	OpenMathInstruct 2[↓]	21,972,791	50,000	26,356	Toshniwal et al. (2024)
	NuminaMath-TIR[α]	64,312	64,312	8,677	Beeching et al. (2024)
Coding	Tülu 3 Persona Python	34,999	34,999	–	–
	Evol CodeAlpaca[α]	107,276	107,276	14,200	Luo et al. (2023)
Safety & Non-Compliance	Tülu 3 CoCoNot	10,983	10,983	10,983	Brahman et al. (2024)
	Tülu 3 WildJailbreak[α,↓]	50,000	50,000	26,356	Jiang et al. (2024)
	Tülu 3 WildGuardMix[α,↓]	50,000	50,000	26,356	Han et al. (2024)
Multilingual	Aya[↓]	202,285	100,000	32,210	Singh et al. (2024b)
Precise IF	Tülu 3 Persona IF	29,980	29,980	19,890	–
	Tülu 3 IF-augmented	65,530	–	65,530	–
Total		23,327,961	939,344	425,145[γ]	

Supervised Finetuning

- From Prompts to SFT Data:
  - For prompts with existing responses: keep the original response if it was written by a human or a frontier model, like GPT-4o.
  - If a set of prompts did not have responses: generate new responses using GPT-4o. Or hand-wrote responses to hardcoded prompts.
- The Tülu 3 SFT Mix
  - To develop our SFT mix, we first identified the skills that were lagging behind state of the art models using Llama 3.1 trained on Tülu 2 as our baseline.
  - To design our final SFT mix, we **first built skill-specific data mixtures and models**, keeping the mixtures that led to the **best performance on individual skills**, ignoring other evaluations. This was done to approximate the upper bound for each evaluation given our setup.
  - We then **combined these mixtures** to create our initial Tülu 3 preview mix. We then continued to **iterate on the mixture** by adding or removing datasets to improve lagging skills, decontaminating against our evaluations and downsampling particularly large datasets.

Supervised Finetuning

- The Tülu 3 SFT Mix

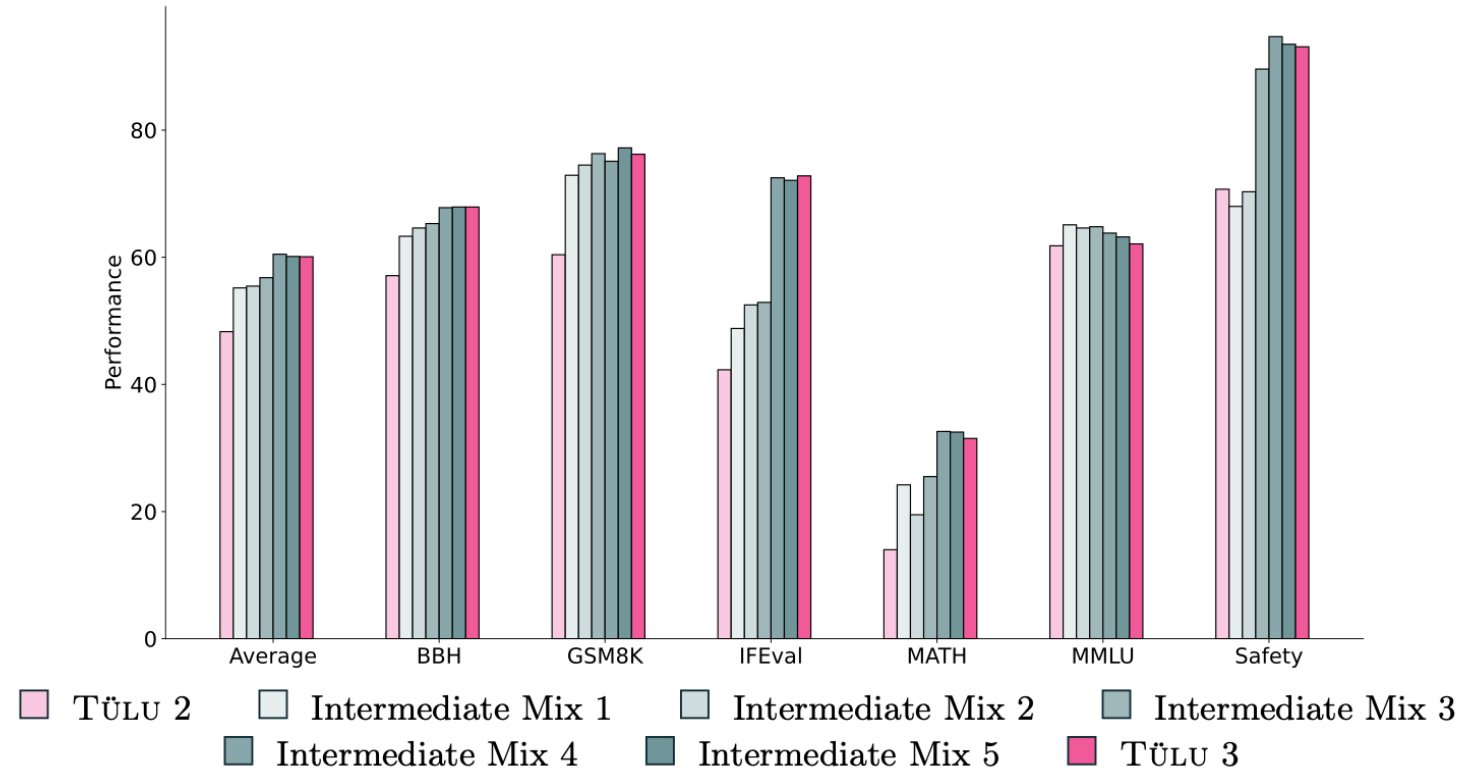


Figure 3 Average and selected skill-specific performance from training Llama 3.1 8B on our initial TÜLU 2 SFT mix, and our intermediate and final TÜLU 3 SFT mixes. Intermediate mixes 1, 2, and 3 were the result of adding new datasets to improve performance. Intermediate mixes 4 and 5 were the result of running multiple rounds of decontamination, causing small drops in performance.

Preference Finetuning

- From Prompts to Preference Data:
  - For a given prompt, we randomly sample four models from a model pool to generate responses.
  - Use an LLM-as-a-judge (GPT-4o-2024-0806), to rate each response from 1 to 5 across four different aspects

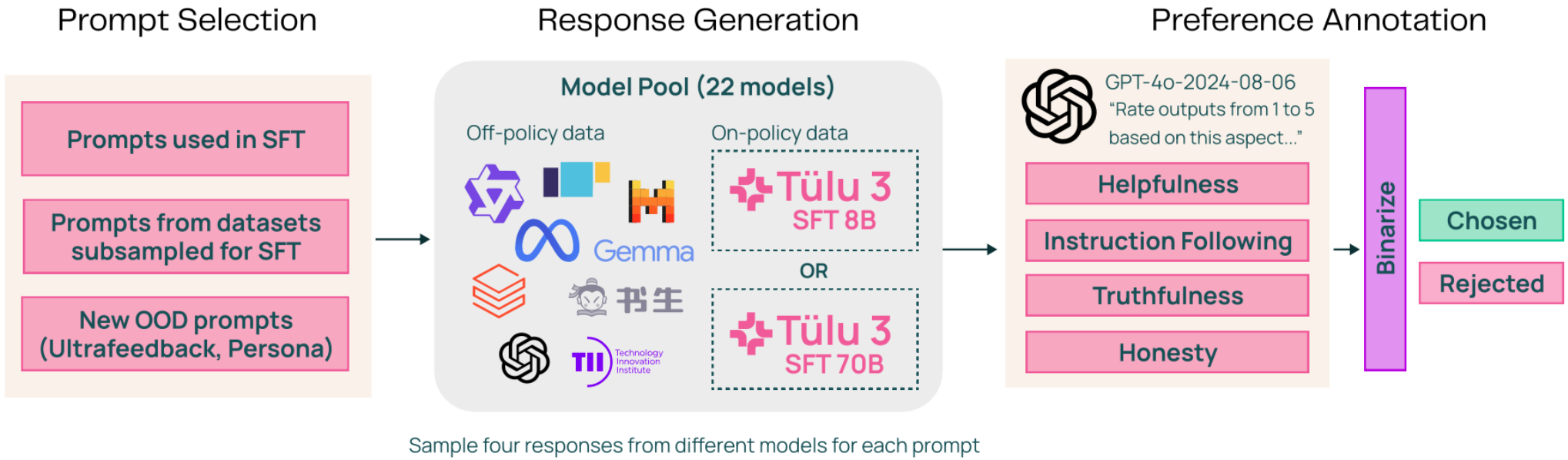


Figure 7 Pipeline for generating and scaling preference data that is based from Ultrafeedback (Cui et al., 2023).

Preference Finetuning

- Comparison Between PPO and DPO:
  - **PPO Gets Similar Average Scores with DPO in this Non-Tuned Setup.** Overall, we found that PPO could reach a comparable level of performance to DPO (albeit slightly lower) in this controlled setup.
  - **PPO is More Computationally Expensive.** The PPO runtime is roughly 28 hours using two nodes, whereas the DPO runtime is about 4 hours using a single node.

Algorithm	LR	$\gamma - \beta$ ratio	β	Epochs	Batch Size	Average Score
SFT Base	-	-	-	-	-	55.7
SimPO	5.00E-07	0.5	2	1	128	51.8
SimPO	5.00E-07	0.3	10	1	128	52.9
DPO	5.00E-07	-	0.1	3	32	55.2
PPO	1.00E-06	-	0.0325	1	64	54.5
PPO	1.00E-06	-	0.05	1	64	55.5
DPO-norm	1.00E-07	-	5	3	32	56.1
DPO-norm	5.00E-07	-	10	3	32	55.2
DPO-norm	5.00E-07	-	15	3	32	55.7
DPO-norm	5.00E-07	-	2	3	32	46.8
DPO-norm	5.00E-07	-	5	3	32	53.4
DPO-norm	5.00E-07	-	5	1	32	**57.3**

Reinforcement Learning with Verifiable Rewards

- RLVR: a novel method for training language models on tasks with **verifiable outcomes** such as mathematical problem-solving and instruction following.

RLVR: $\max_{\pi_\theta} \mathbb{E}_{y\sim\pi_\theta(x)}\left[R_{\text{RLVR}}(x,y)\right] = \left[v(x,y) - \beta\text{KL}[\pi_\theta(y|x)||\pi_{\text{ref}}(y|x)]\right]$

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

$$v(x,y) = \begin{cases} \alpha & \text{if correct,} \\ 0 & \text{otherwise.} \end{cases}$$

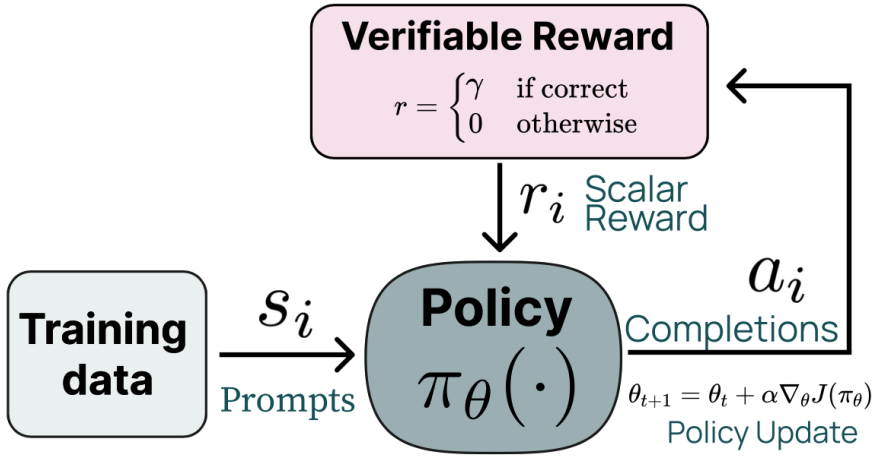


Figure 18 An overview of how Reinforcement Learning with Verifiable Rewards (RLVR) works. We sample completions from a policy model given a set of prompts, and verify their correctness using a deterministic function. If the answer is verifiably correct, we provide reward of α, otherwise 0. We then train against this reward using PPO.

Reinforcement Learning with Verifiable Rewards

- RLVR Data: focus on two domains (mathematics, exact instruction following) and three evaluations (GSM8K, MATH, IFEval) with relatively straightfoward methods for verification
- Key Findings:
  - RLVR Can Improve Performance in Targeted Domains: In all cases, we achieve models that outperform the initial model in that particular evaluation. We also see that the verifiable rewards (i.e., correctness on the train set) improves consistently for all three settings.
  - Starting from a Weaker Model Can Converge to the Same Verifiable Rewards: starting from both SFT and DPO can lead to the same level of verifiable rewards. However, we find that starting from a stronger model usually results in better test set performance.

Reinforcement Learning with Verifiable Rewards

- They ran their final RLVR runs using the combined verifiable prompt set, and used the best DPO models from the prior section as starting points.

Model Size		8B			70B		
Category	Benchmark$_{\text{(Eval Setting)}}$	Llama 3.1 Inst.	Tülu 3 DPO	Tülu 3 RLVR	Llama 3.1 Inst.	Tülu 3 DPO	Tülu 3 RLVR
Avg.		62.2	64.4	**64.8**	73.4	75.9	**76.0**
Knowledge	MMLU$_{\text{(0 shot, CoT)}}$	**71.2**	68.7	68.2	**85.3**	83.3	83.1
	PopQA$_{\text{(15 shot)}}$	20.2	**29.3**	29.1	46.4	46.3	**46.5**
	TruthfulQA$_{\text{(6 shot)}}$	55.1	**56.1**	55.0	66.8	**67.9**	67.6
Reasoning	BigBenchHard$_{\text{(3 shot, CoT)}}$	62.8	65.8	**66.0**	73.8	81.8	**82.0**
	DROP$_{\text{(3 shot)}}$	61.5	62.5	**62.6**	**77.0**	74.1	74.3
Math	MATH$_{\text{(4 shot CoT, Flex)}}$	42.5	42.0	**43.7**	56.4	62.3	**63.0**
	GSM8K$_{\text{(8 shot, CoT)}}$	83.4	84.3	**87.6**	**93.7**	93.5	93.5
Code	HumanEval$_{\text{(pass@10)}}$	**86.3**	83.9	83.9	**93.6**	92.4	92.4
	HumanEval+$_{\text{(pass@10)}}$	**82.9**	78.6	79.2	**89.5**	88.4	88.0
IF & Chat	IFEval$_{\text{(Strict)}}$	80.6	81.1	**82.4**	**88.0**	82.6	83.2
	AlpacaEval 2$_{\text{(LC \% win)}}$	24.2	33.5	**34.5**	33.4	49.6	**49.8**
Safety	Safety$_{\text{6 task avg.}}$	75.2	**87.2**	85.5	76.5	**89.0**	88.3

Table 23 Final performance of RLVR-trained TÜLU 3 models compared to Llama 3.1 and DPO starting points. The best-performing model on each benchmark (i.e., in each row) and of each size is **bolded**.

Tülu 3 Evaluation Framework

They split their evaluation suite into a **development set** and an **unseen set**, the former used for developing models, and the latter only for evaluating final models.

	Category	Benchmark	CoT	# Shots	Chat	Multiturn ICL	Metric
Development	Knowledge Recall	MMLU	✓	0	✓	✗	EM
		PopQA	✗	15	✓	✓	EM
		TruthfulQA	✗	6	✓	✗	MC2
	Reasoning	BigBenchHard	✓	3	✓	✓	EM
		DROP	✗	3	✗	N/A	F1
	Math	GSM8K	✓	8	✓	✓	EM
		MATH	✓	4	✓	✓	Flex EM
	Coding	HumanEval	✗	0	✓	N/A	Pass@10
		HumanEval+	✗	0	✓	N/A	Pass@10
	Instruction Following	IFEval	✗	0	✓	N/A	Pass@1 (prompt; loose)
		AlpacaEval 2	✗	0	✓	N/A	LC Winrate
	Safety	Tülu 3 Safety	✗	0	✓	N/A	Average*
Unseen	Knowledge Recall	MMLU-Pro	✓	0	✓	N/A	EM
		GPQA	✓	0	✓	N/A	EM
	Reasoning	AGIEval English	✓	0	✓	✓	EM
	Math	Deepmind Mathematics	✓	0	✓	✓	EM (Sympy)
	Coding	BigCodeBench	✗	0	✓	N/A	Pass@10
	Instruction Following	IFEval-OOD	✗	0	✓	N/A	Pass@1 (prompt; loose)
		HREF	✗	0	✓	N/A	Winrate

Table 24 The Tülu 3 Evaluation Regime: settings for development (**top**) and unseen (**bottom**) portions of the evaluation suite. **CoT** are evaluations run with chain of thought prompting (Wei et al., 2022b). **#Shots** is the number of in-context examples in the evaluation template. **Chat** refers to whether we use a chat template while prompting the model. **Multiturn ICL** refers to a setting where we present each in-context example as a separate turn in a conversation (applicable only when a chat template is used and # Shots is not 0). *Average over multiple sub-evaluations – full details of the safety evaluation are included in the Appendix.

Future Work

- **Long Context and Multi-turn.** Currently, the data collected for Tülu 3 is relatively short and does not contain long multi-turn data (the average number of turns in our mixture is 2.4 turns and majority of samples are under 2,048 tokens in length). However, long-context has been popular area of focus in recent work, as improving the context window of LMs enables new use-cases and more in-context examples, potentially improving performance. Relatedly, improving multi-turn capabilities can better improve end-user experience, with a non-trivial number of real-world user conversations with LMs going over 2 turns
- **Multilinguality.** They specifically focus on English data and evaluations for Tülu 3.
- **Tool Use and Agents.** While we evaluate Tülu 3 on its own, LMs are being increasingly deployed as parts of larger systems, in which they have access to tools or are themselves part of a larger 'agent' framework. Furthermore, training models to use tools is a natural way to dramatically improve their reasoning and mathematical skills, rather than trying to accomplish everything 'in the weights.'

Thank you!