Training Large Language Models: *Practices* and *Research Questions*

Danqi Chen

Princeton Language and Intelligence

Princeton University

September 5th, 2024

Before I start

- **My background**: NLP research → understanding and training LMs
- Focusing on **academic research**

  Q. What research questions in LLM pipeline can academics answer?

- Most content is drawn from **public knowledge** and our experiences in academic research projects
  - Many details still remain opaque, and need rigorous experiments to verify
  - Many findings are still "practices", and lack scientific understanding

How do we learn the (best) practices?

- **Llama 3.1 technical report** (arXiv 2407.21783) 2024/7/23
- **Gemma 2 technical report** (arXiv 2408.00118) 2024/7/31
- **Qwen2 technical report** (arXiv 2407.10671) 2024/7/15
- **Apple Intelligence technical report** (arXiv 2407.21075) 2024/7/29
- **Phi-3 paper** (arXiv 2404.14219) 2024/4/24
- **OLMo paper** (arXiv 2402.00838) 2024/2/1
- **Gemini paper** (arXiv 2312.11805) 2023/12/19
- **Mistral 7B** (arXiv 2310.06825) 2023/10/10

+ many, many scientific research papers (smaller scale, more rigorous, more controlled, clear gap from SOTA)

Scope of this tutorial

This tutorial will focus on **training pipeline** of cutting-edge LLMs

Part I. **Pre-training**

“the model is trained at **massive scale** using straightforward tasks such as **next-word prediction**”

Part II. **Post-training**

“the model is tuned to **follow instructions**, align with **human preferences**, and **improve specific capabilities** (e.g., coding and reasoning)”

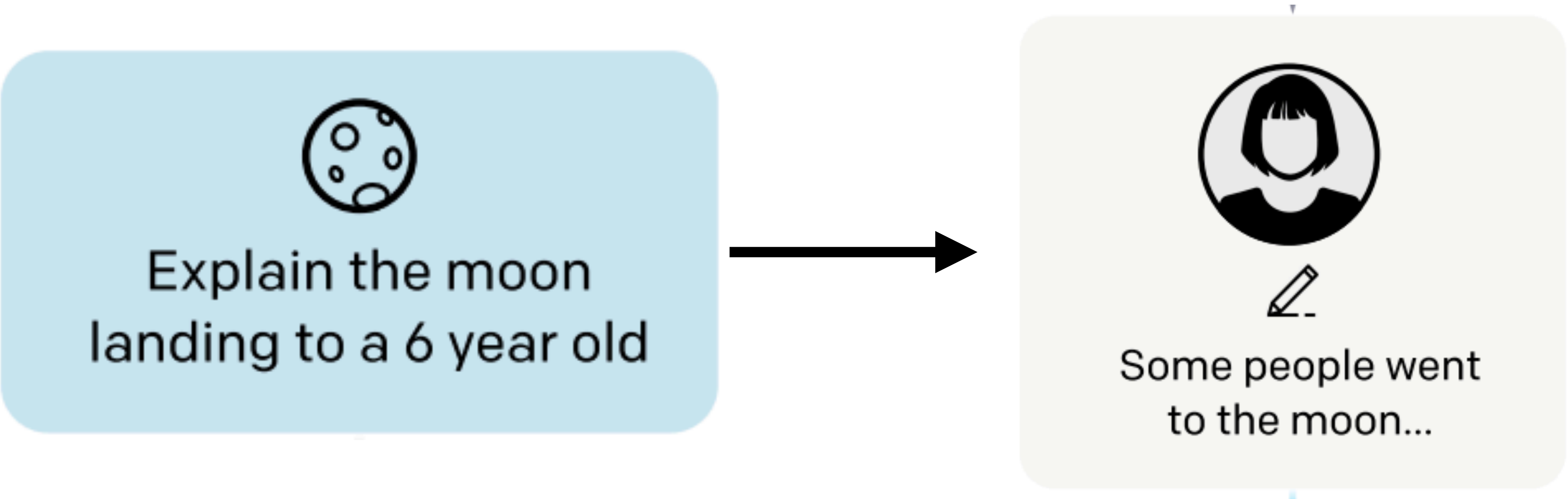
Data vs Algorithms vs ~~Model architectures~~

Part II. **Post-training**

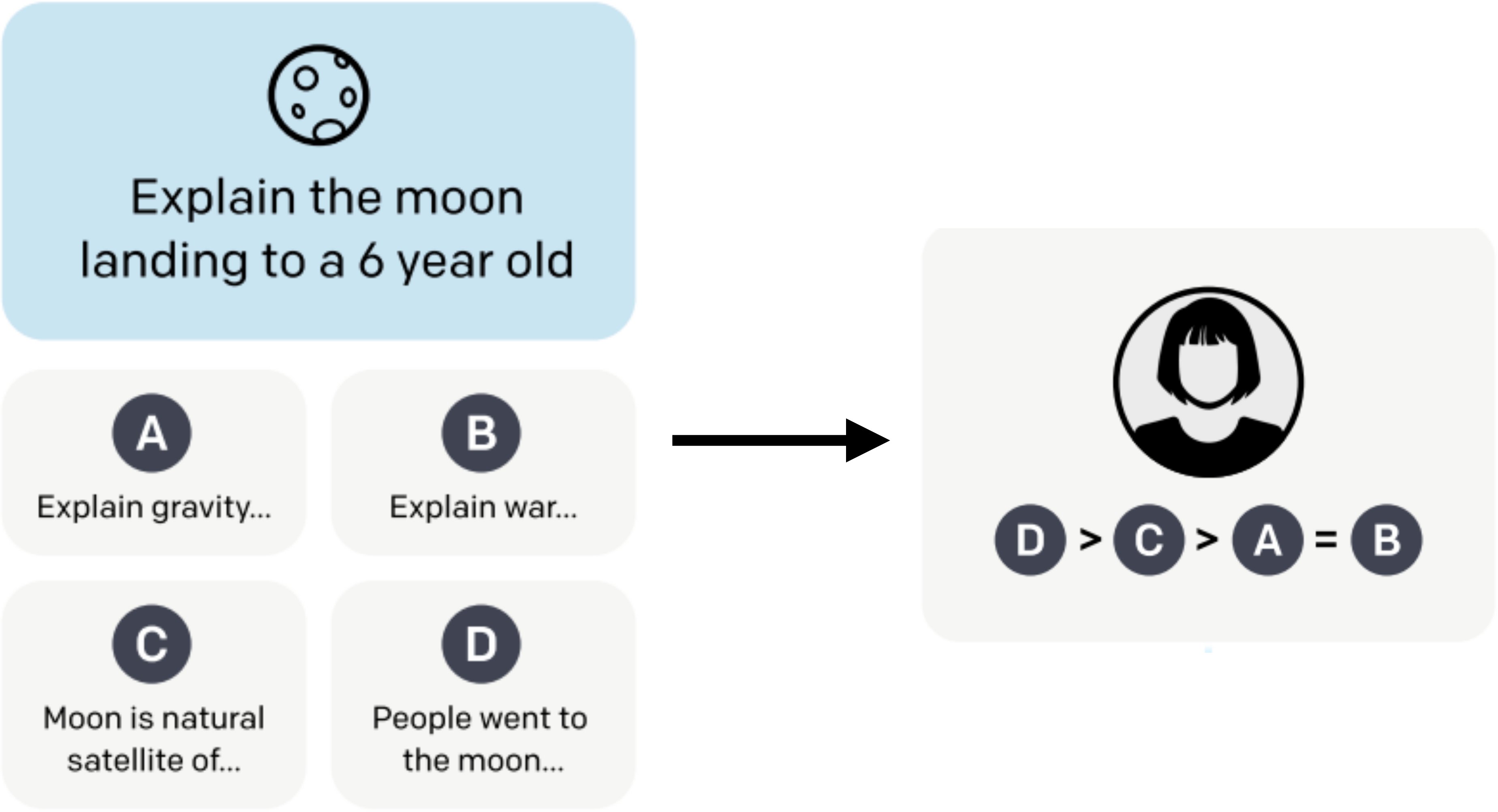
"the model is tuned to **follow instructions**, align with **human preferences**, and **improve specific capabilities** (e.g., coding and reasoning)"

Post-training: Two (simplified) stages

- **Supervised fine-tuning** (SFT)
  - aka. Instruction fine-tuning
  - **Data**: (prompt, response)
  - **Learning**: next-token prediction



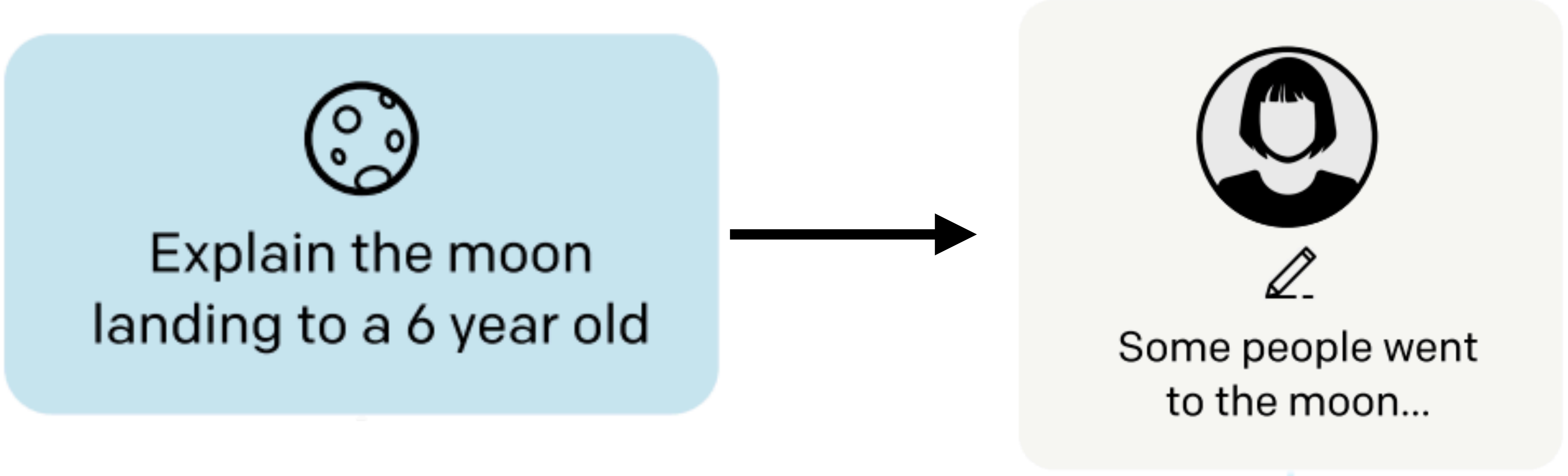
- **Preference learning**
  - aka. Reinforcement learning from human preferences
  - **Data**: (prompt, winning response, losing response)
  - **Learning:** RL (PPO) vs offline preference optimization (DPO)



Training language models to follow instructions with human feedback (2022)

Supervised fine-tuning (SFT)

- **Data**: (**prompt**, **response**)
- **Learning**: next-token prediction



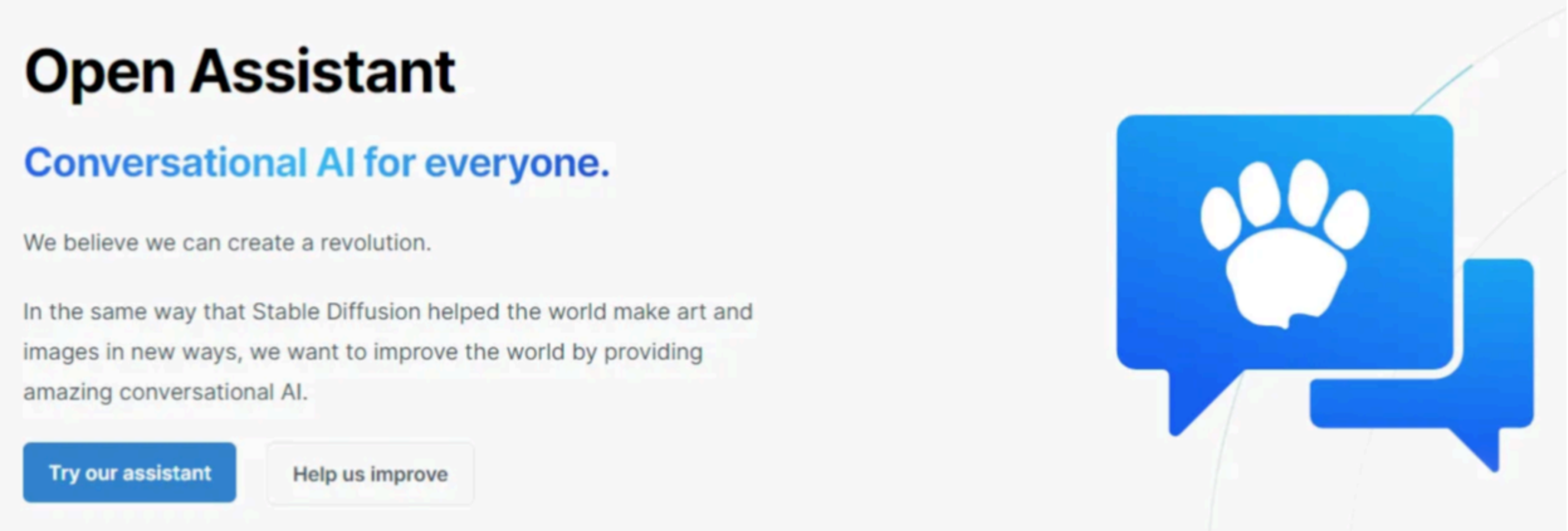
Practices and research questions:

- How to get **prompts**?
- How to get **responses**? Do responses include chain-of-thought?
- How to ***combine*** *and* ***select*** these datasets for instruction tuning?

Supervised fine-tuning datasets

- **Repurposed from existing datasets** (w/ human-written instructions and CoT)
    - Examples: Super-NaturalInstructions, Flan V2
- **Human-written from scratch**
    - Examples: Dolly, Open Assistant

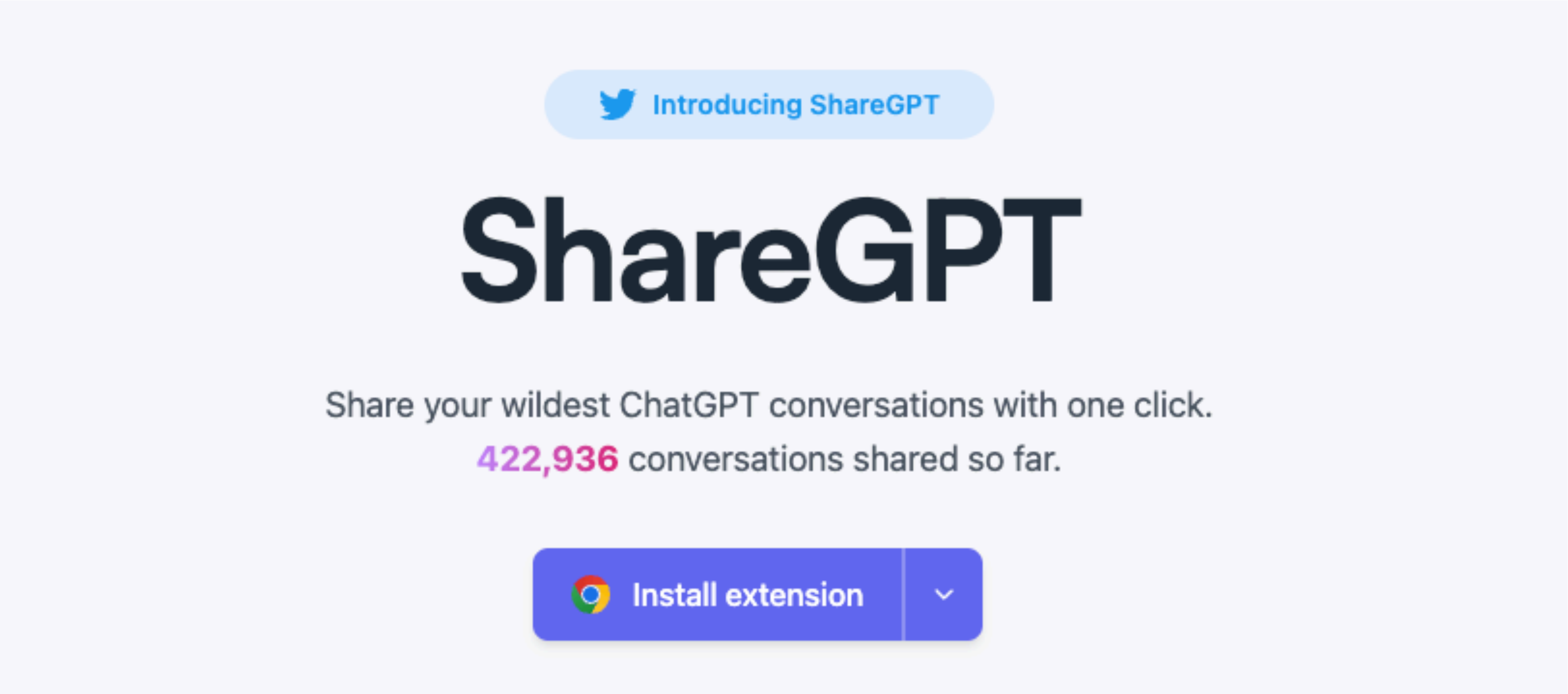
Super-NaturalInstructions (Wang et al., 2022)

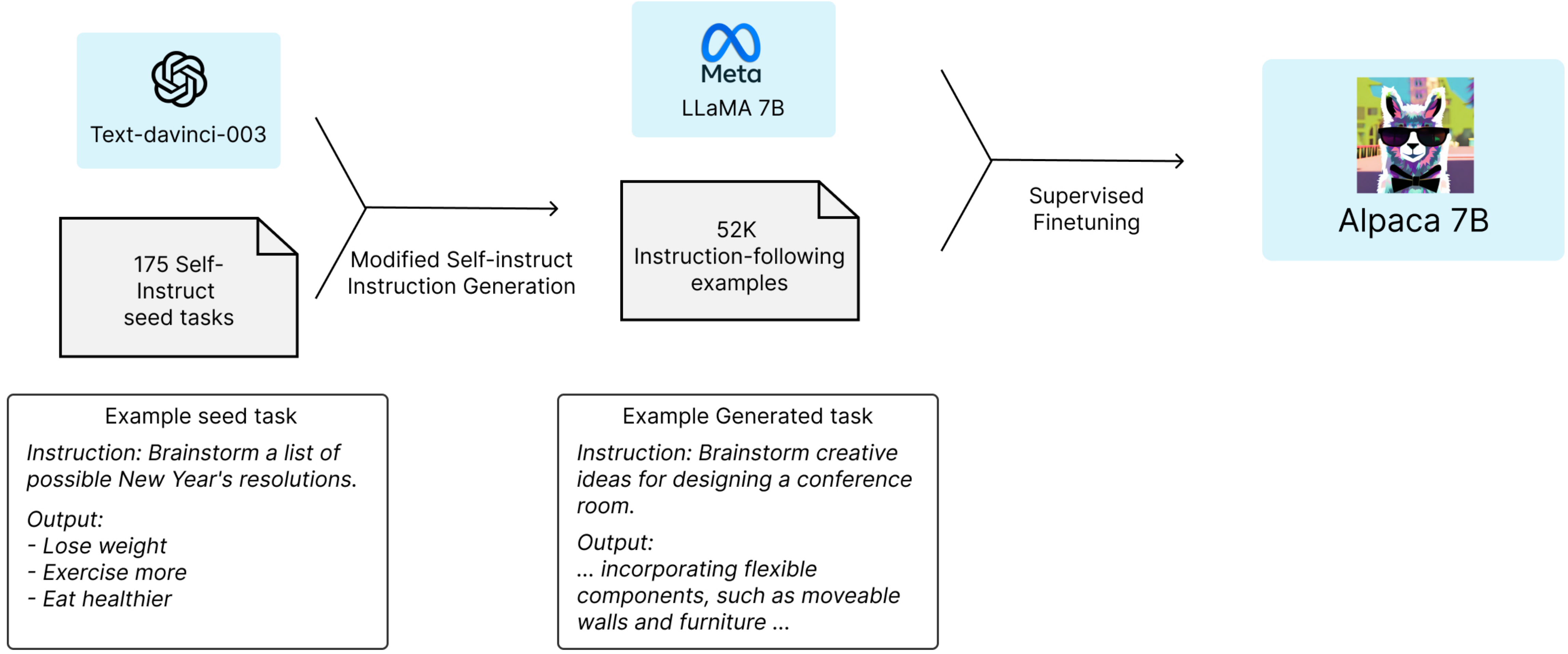


(Köpf et al., 2023)

Supervised fine-tuning datasets

- **Responses generated from LLMs**
  - Example: ShareGPT, UltraChat
- **The instructions can be generated from LLMs too!**
  - Example: Alpaca





Stanford Alpaca

Data mixture of instruction tuning

TÜLU v2

- **FLAN** [Chung et al., 2022]: We use 50,000 examples sampled from FLAN v2.
- **CoT**: To emphasize chain-of-thought (CoT) reasoning, we sample another 50,000 examples from the CoT subset of the FLAN v2 mixture.
- **Open Assistant 1** [Köpf et al., 2023]: We isolate the highest-scoring paths in each conversation tree and use these samples, resulting in 7,708 examples. Scores are taken from the quality labels provided by the original annotators of Open Assistant 1.
- **ShareGPT**[2]: We use all 114,046 examples from our processed ShareGPT dataset, as we found including the ShareGPT dataset resulted in strong performance in prior work.
- **GPT4-Alpaca** [Peng et al., 2023]: We sample 20,000 samples from GPT-4 Alpaca to further include distilled GPT-4 data.
- **Code-Alpaca** [Chaudhary, 2023]: We use all 20,022 examples from Code Alpaca, following our prior V1 mixture, in order to improve model coding abilities.
- ***LIMA** [Zhou et al., 2023]: We use 1,030 examples from LIMA as a source of carefully curated data.
- ***WizardLM Evol-Instruct V2** [Xu et al., 2023]: We sample 30,000 examples from WizardLM, which contains distilled data of increasing diversity and complexity.
- ***Open-Orca** [Lian et al., 2023]: We sample 30,000 examples generated by GPT-4 from OpenOrca, a reproduction of Orca [Mukherjee et al., 2023], which augments FLAN data with additional model-generated explanations.
- ***Science literature**: We include 7,544 examples from a mixture of scientific document understanding tasks— including question answering, fact-checking, summarization, and information extraction. A breakdown of tasks is given in Appendix C.
- ***Hardcoded**: We include a collection of 140 samples using prompts such as 'Tell me about yourself' manually written by the authors, such that the model generates correct outputs given inquiries about its name or developers.

Size	Data	Average -
7B	ShareGPT	47.0
	V1 mix.	47.8
	V2 mix.	**54.2**
13B	V1 mix.	56.0
	V2 mix.	**60.8**
70B	V1 mix.	71.5
	V2 mix.	**72.4**

- What is the notion of **"high-quality" data** in instruction tuning?

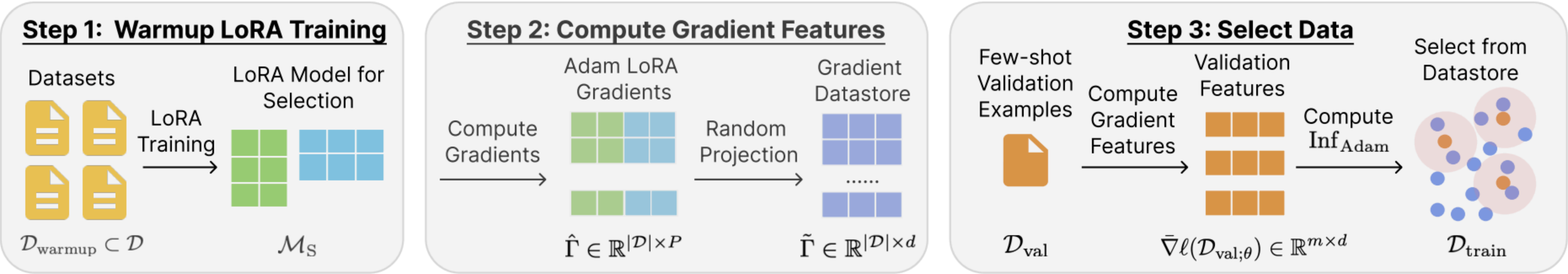
- How to decide **data mixture** or which **examples** to use?

Camels in a Changing Climate: Enhancing LM Adaptation with TÜLU 2 (2023)

LESS: Selecting Influential Data for Targeted Instruction Tuning

- Key idea: use **influence formulation** to estimate how training examples influence models' predictions on target tasks and use it as proxy for data selection

$$\mathrm{Inf}_{\mathrm{Adam}}(x, z) = \sum_{i=1}^{N} \bar{\eta}_i \cos(\nabla l(z; \theta_i), \Gamma(x; \theta_i))$$



LESS: Selecting Influential Data for Targeted Instruction Tuning (2024)

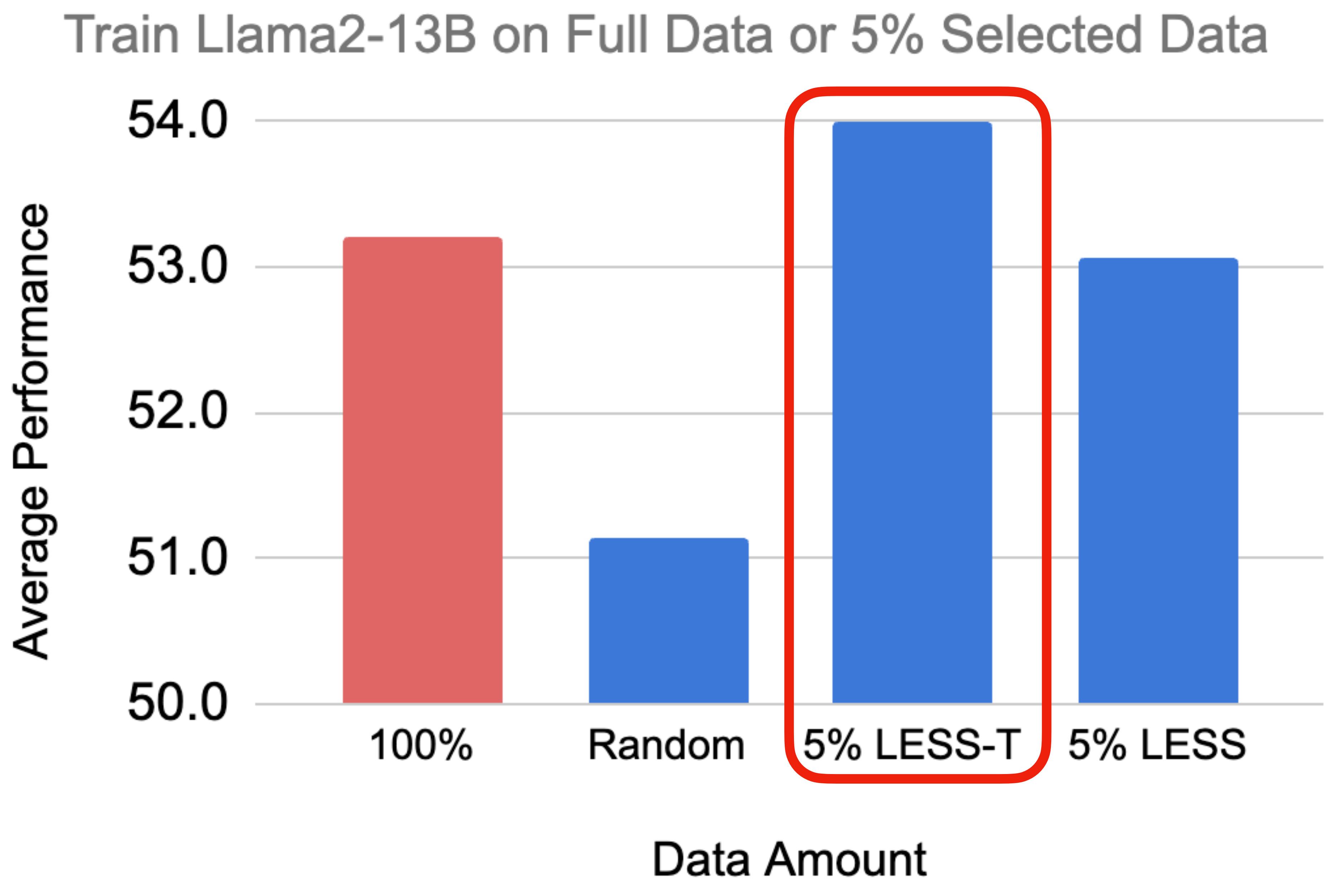
LESS: Selecting Influential Data for Targeted Instruction Tuning

Technical enhancements making it work for:

- Adam optimizer
- Instruction tuning datasets (varied length)
- Large models - efficiency is the key!

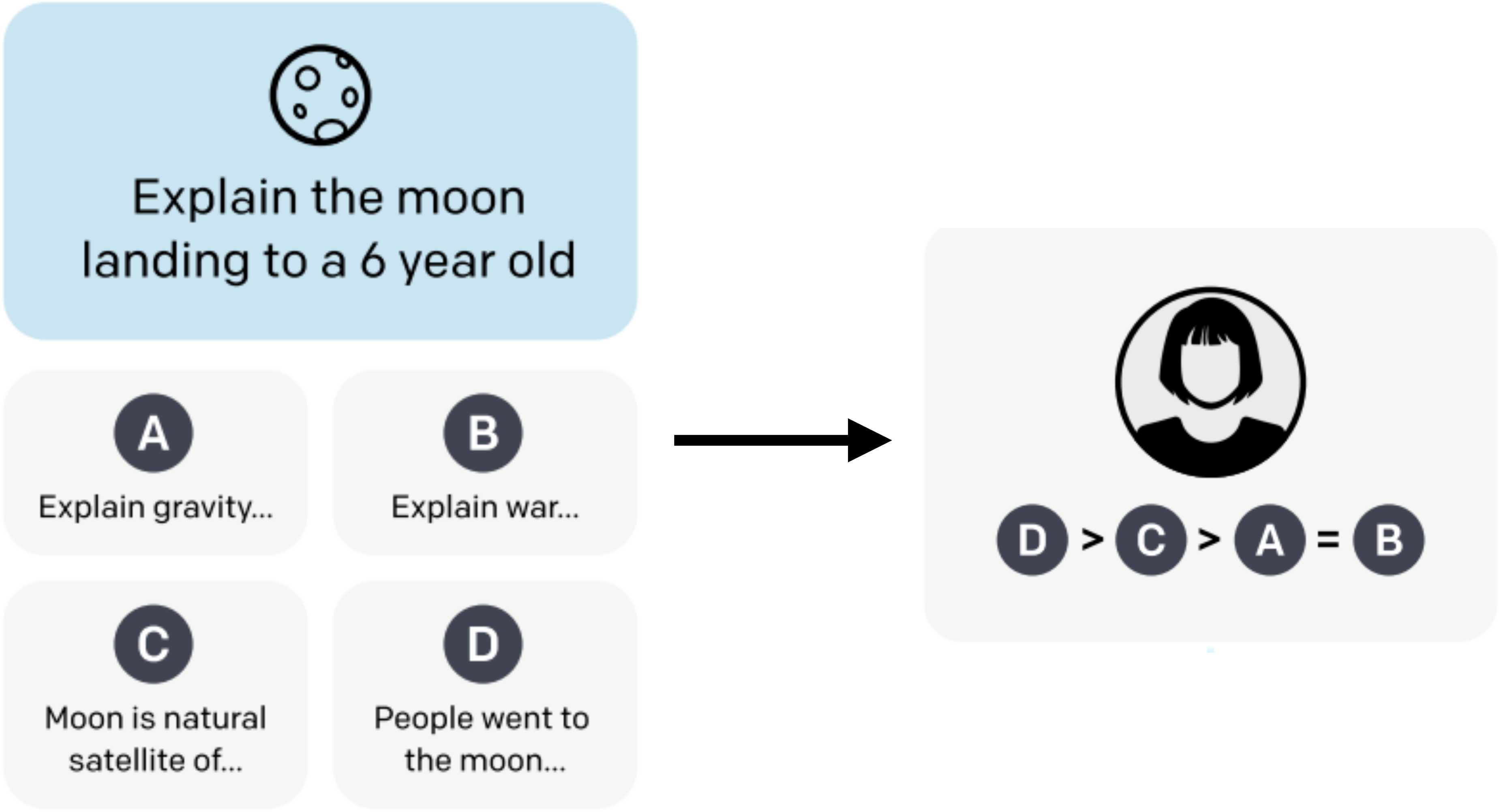
Less-T: “transfer” setting

Instruction tuning examples selected based on LLama-2-7B can be used to instruct fine-tune Mistral-7B and LLama-2-13B!



Preference learning

- **Data**: (**prompt**, **winning response**, **losing response**)
- **Learning:** RL (PPO) vs offline PO (DPO)



- How to get **prompts**?
- How to get **winning responses** and **losing responses**?
  - *Who decides which is winning and which is losing?*

- RL vs offline preference optimization algorithms?

Next: I will first discuss algorithms, present some experimental findings, and come back to the discussion of data

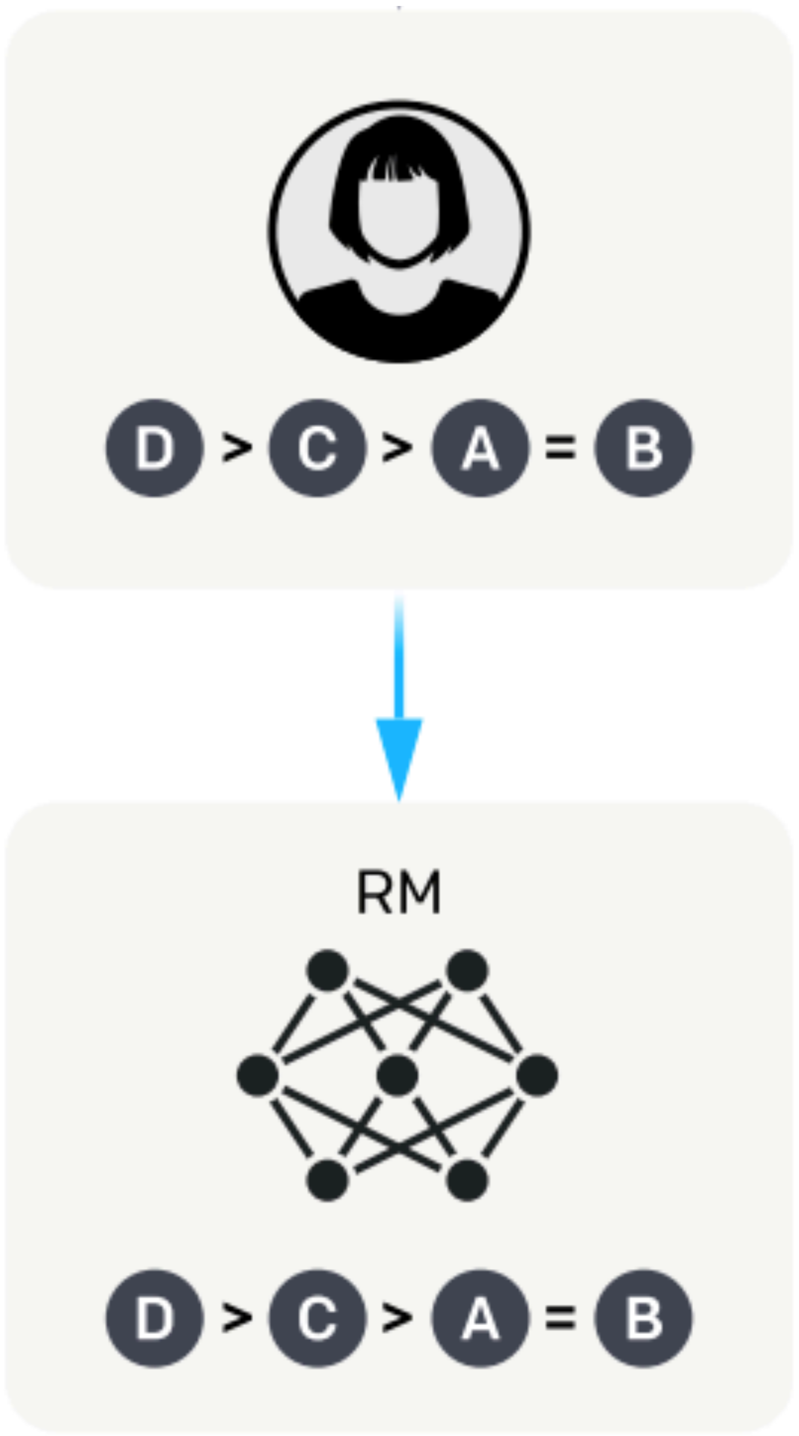
Reinforcement learning from human feedback

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$

- **Step 1:** Train a **reward model (RM)** using the preference data

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

$r_\theta(x, y)$ measures how good response y is following prompt x



Training language models to follow instructions with human feedback (2022)

Reinforcement learning from human feedback

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$

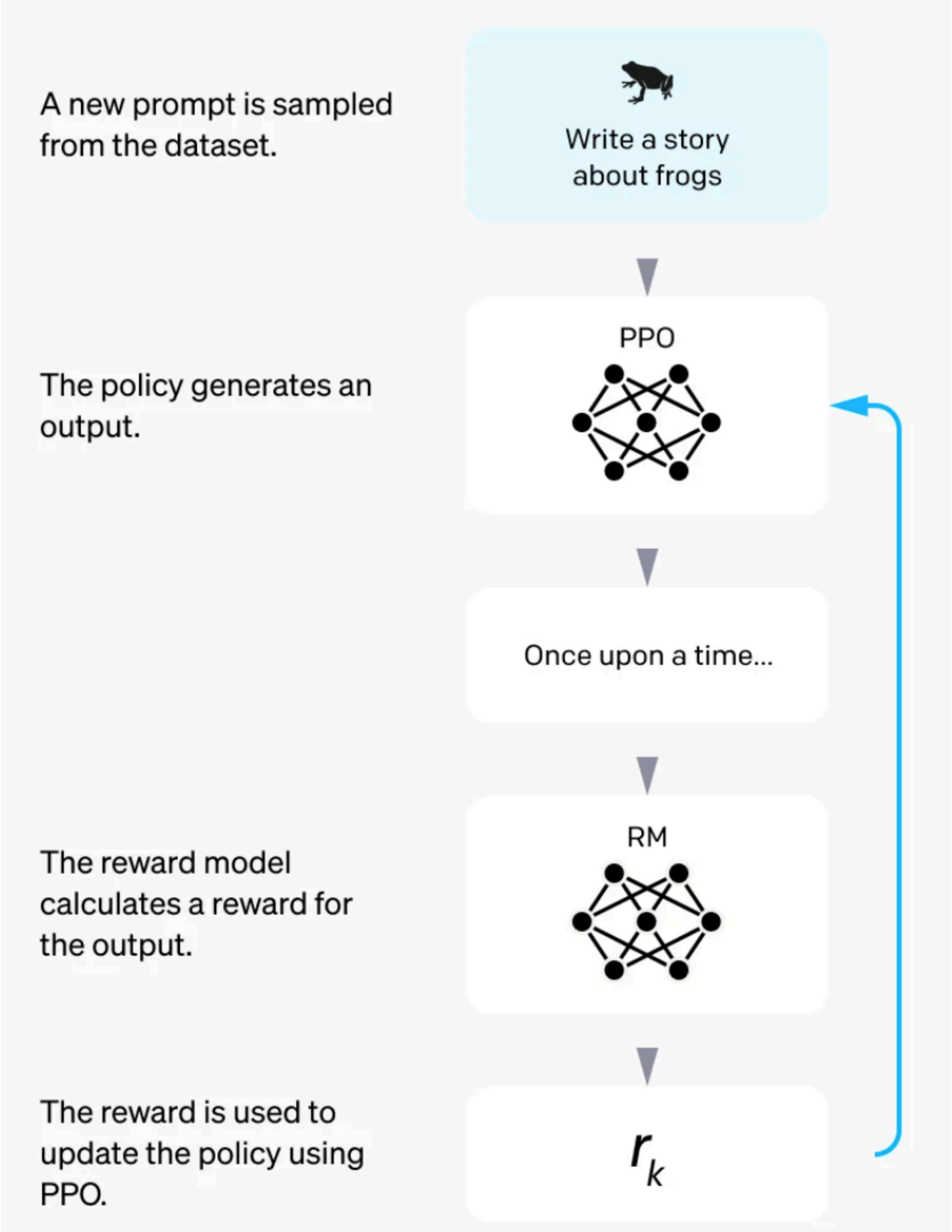
- **Step 1:** Train a **reward model (RM)** using the preference data

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

$r_\theta(x, y)$ measures how good response y is following prompt x

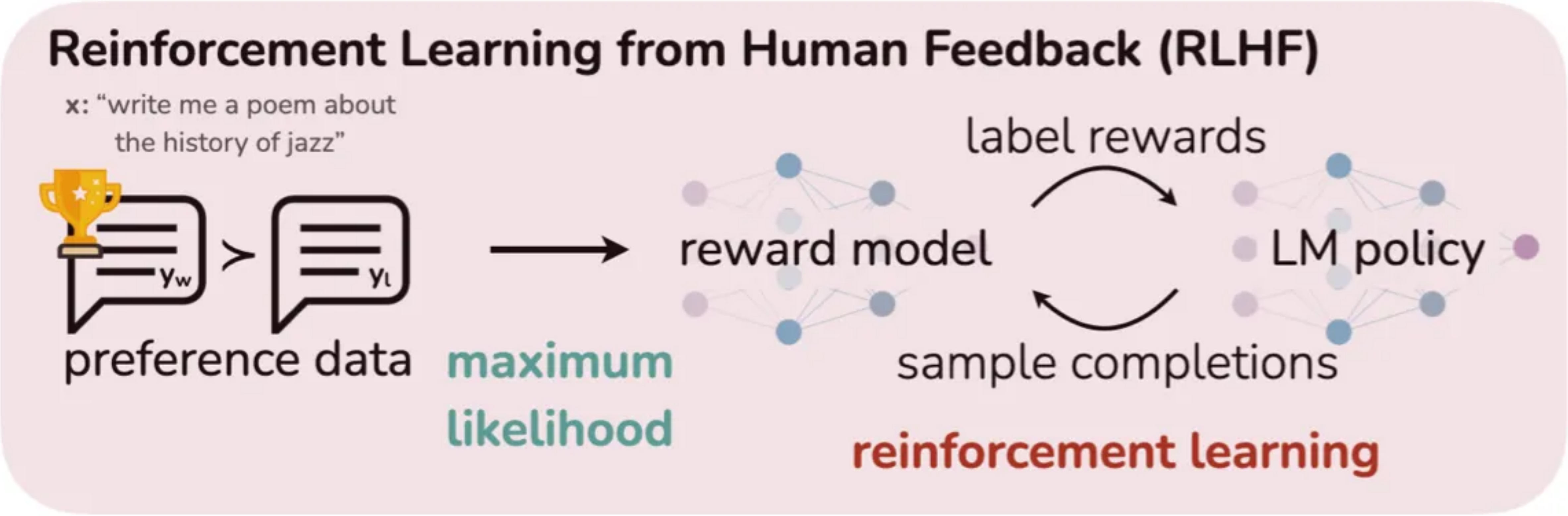
- **Step 2:** Start from the **SFT model** as the **policy model**, sample a response and update the policy with the **RM model**

Additionally, add a per-token KL penalty to make sure the policy model doesn't deviate too much from the SFT model



Reinforcement learning from human feedback

Preference data: (prompt, **winning response**, **losing response**) $(x, y_w, y_l) \sim D$



Drawbacks:

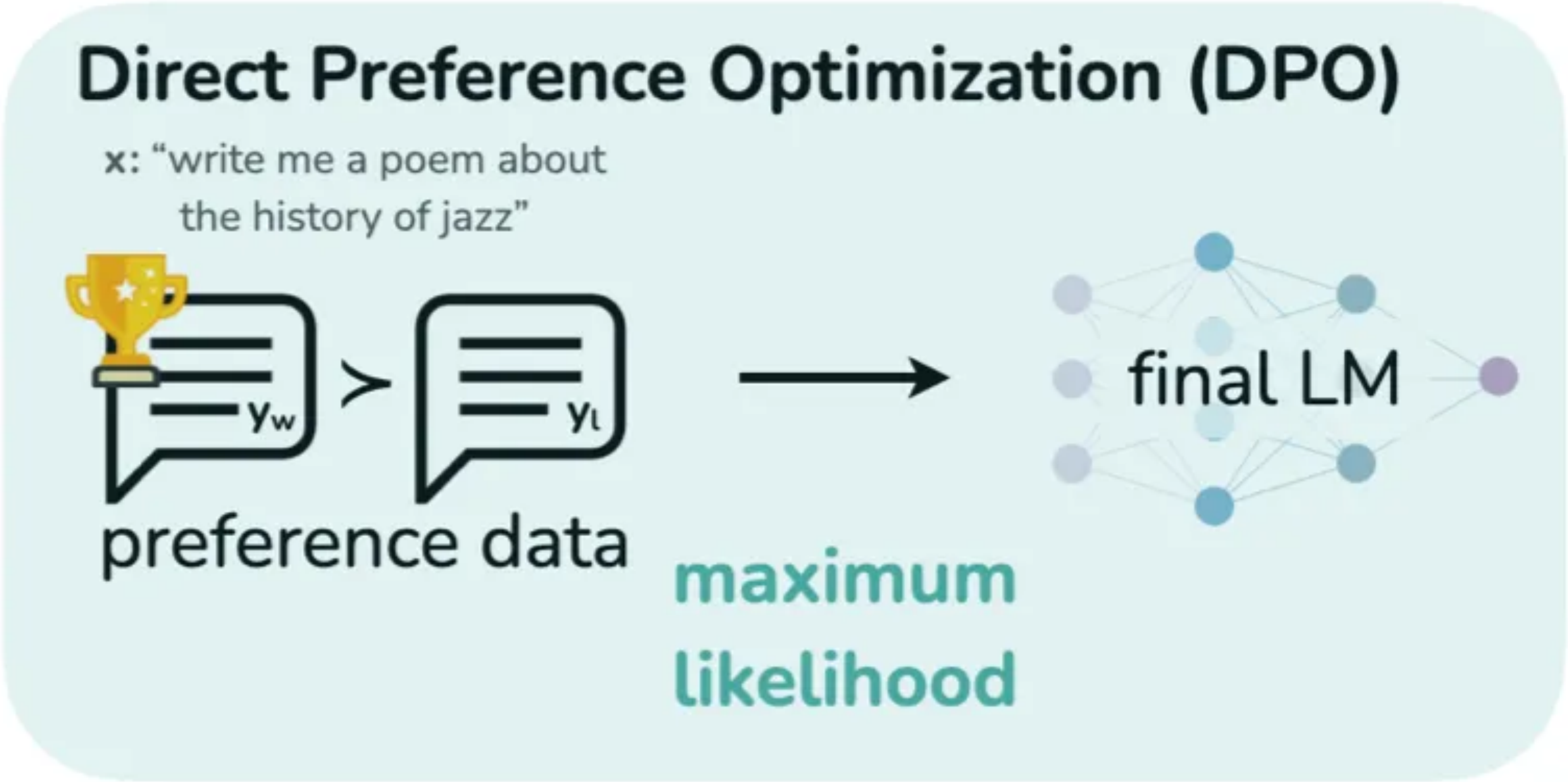
- Involve multiple models SFT, RM, policy models
- Involve multiple stages of training
- Complex, hard to get it right!

1. Optimize **reward model** over **preference data**
2. Optimize **policy model** according to the **reward model**

Next: Why not directly learn the **policy model** from **preference data**?

Direct preference optimization (DPO)

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$



- DPO starts from a very similar RL objective to PPO:

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

- Under a general reward function r_ϕ, the optimal policy can be written as:

$$\pi_r(y \mid x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)$$

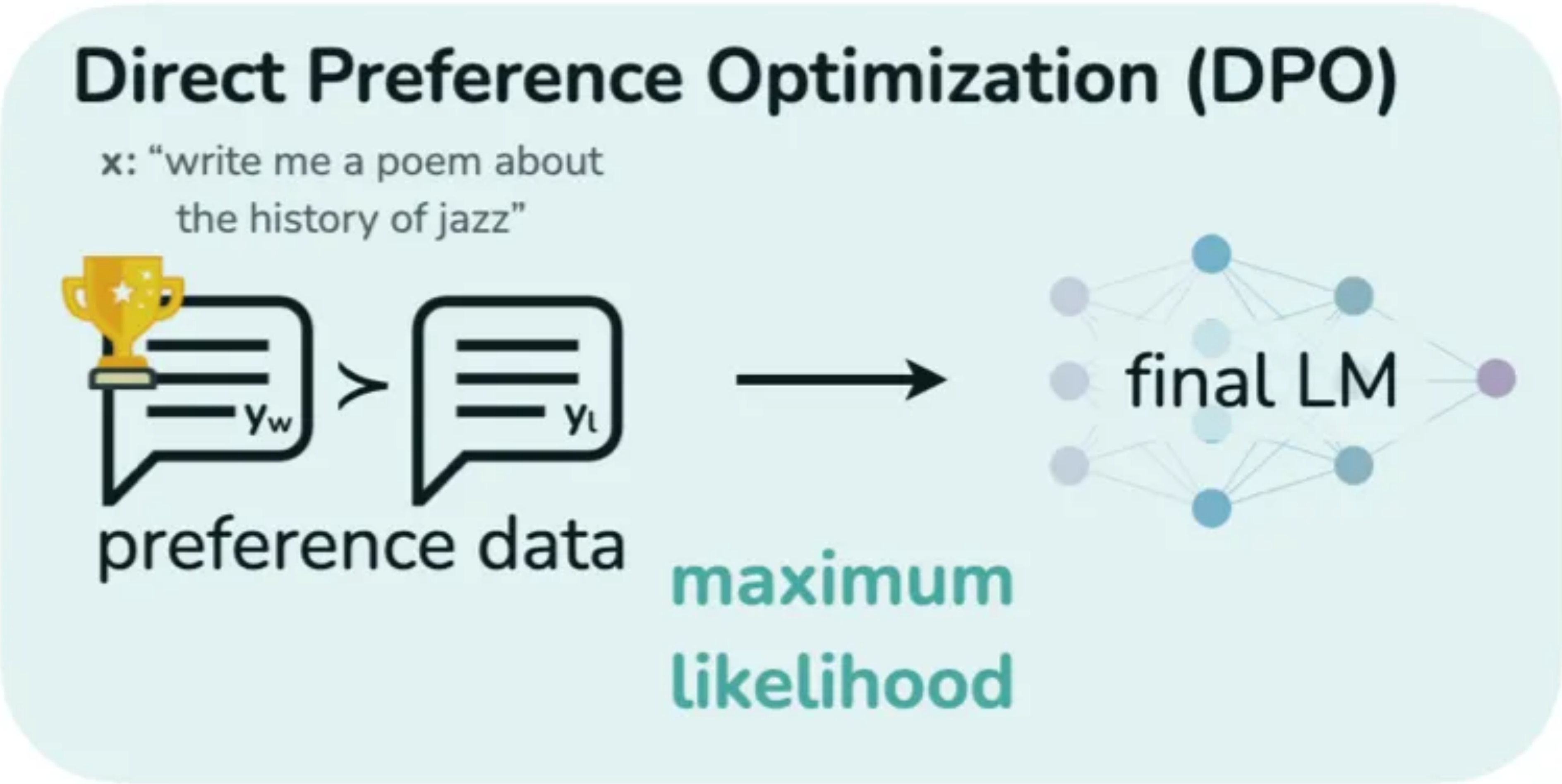
$$\downarrow$$

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

Direct Preference Optimization: Your Language Model is Secretly a Reward Model (2023)

Direct preference optimization (DPO)

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$



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

Reward modeling (Bradley-Terry ranking):

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

DPO objective:

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

Direct Preference Optimization: Your Language Model is Secretly a Reward Model (2023)

Offline preference optimization methods

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$

There are many objectives that you can design for directly learning from preference data!

Method	Objective
RRHF [84]	$\max\left(0, -\frac{1}{\lvert y_w \rvert}\log \pi_\theta(y_w \lvert x) + \frac{1}{\lvert y_l \rvert}\log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
SLiC-HF [88]	$\max\left(0, \delta - \log \pi_\theta(y_w \lvert x) + \log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
DPO [62]	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)}\right)$
IPO [6]	$\left(\log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)} - \frac{1}{2\tau}\right)^2$
CPO [81]	$-\log \sigma\left(\beta \log \pi_\theta(y_w \lvert x) - \beta \log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
KTO [25]	$-\lambda_w \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - z_{\text{ref}}\right) + \lambda_l \sigma\left(z_{\text{ref}} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)}\right)$, where $z_{\text{ref}} = \mathbb{E}_{(x,y)\sim D}\left[\beta \mathrm{KL}\left(\pi_\theta(y \lvert x) \Vert \pi_{\text{ref}}(y \lvert x)\right)\right]$
ORPO [38]	$-\log p_\theta(y_w \lvert x) - \lambda \log \sigma\left(\log \frac{p_\theta(y_w \lvert x)}{1-p_\theta(y_w \lvert x)} - \log \frac{p_\theta(y_l \lvert x)}{1-p_\theta(y_l \lvert x)}\right)$, where $p_\theta(y \lvert x) = \exp\left(\frac{1}{\lvert y \rvert}\log \pi_\theta(y \lvert x)\right)$
R-DPO [60]	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)} - (\alpha \lvert y_w \rvert - \alpha \lvert y_l \rvert)\right)$

Method	LLama-3-instruct (8B)		
	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)
SFT	26.0	25.3	22.3
RRHF [84]	37.9	31.6	28.8
SLiC-HF [88]	33.9	32.5	29.3
DPO [62]	48.2	**47.5**	35.2
IPO [6]	46.8	42.4	**36.6**
CPO [81]	34.1	36.4	30.9
KTO [25]	34.1	32.1	27.3
ORPO [38]	38.1	33.8	28.2
R-DPO [60]	48.0	45.8	35.1

WR: winning rate, LC: length-controlled WR

SimPO: Simple preference optimization

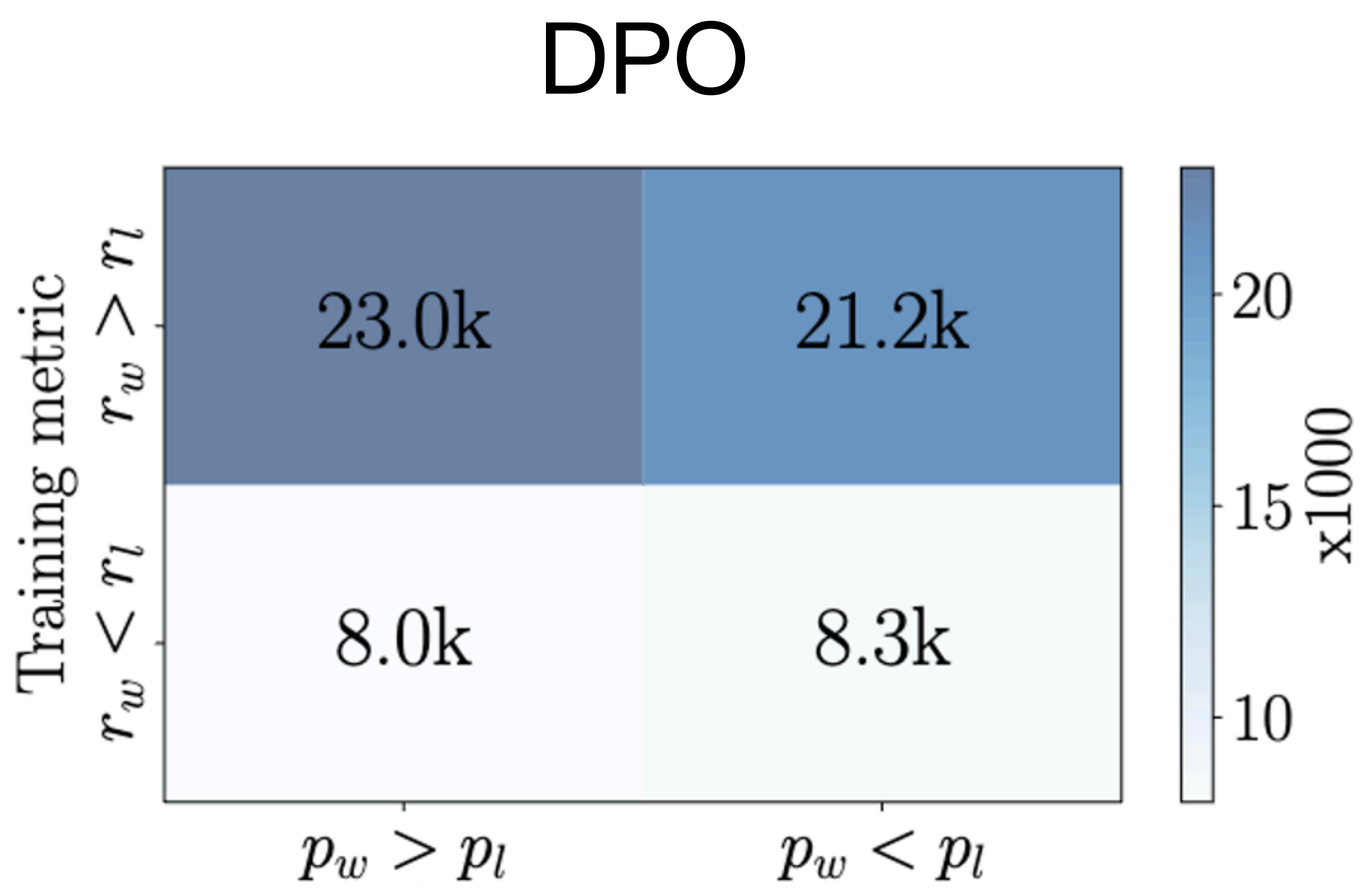
DPO Training: $\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$

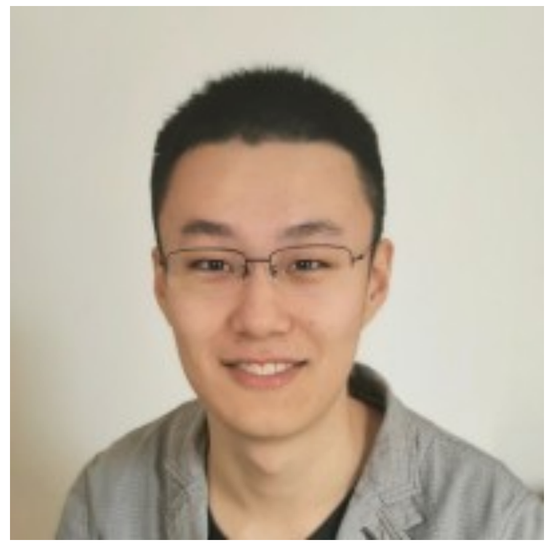
$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)}$$

Inference: We take $\pi_r(y \mid x)$, and start from x, and generate *y*!

- Use greedy, beam search, or sampling
- We don't use π_{ref} at all during inference

What is the role of reference model at all?

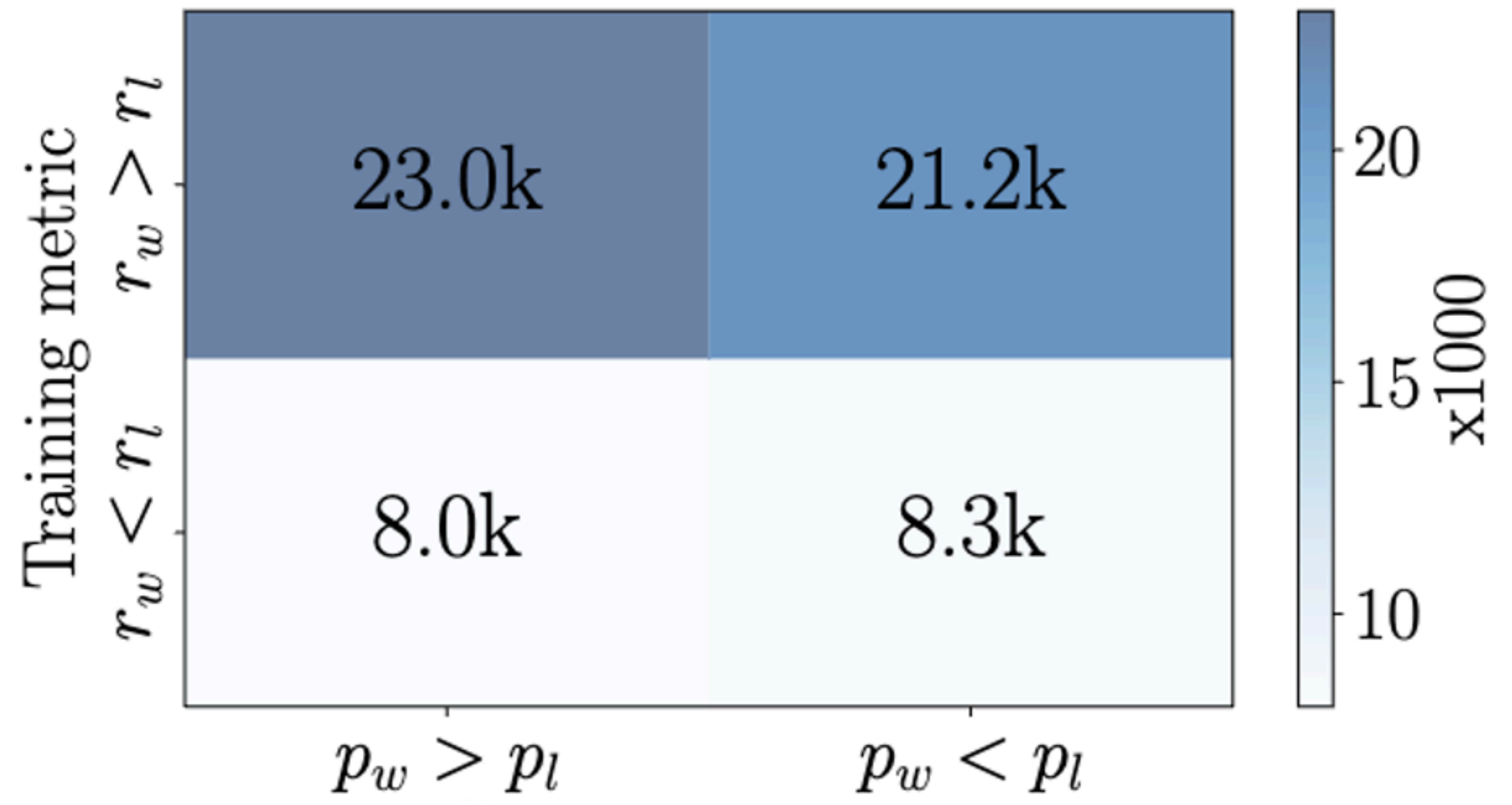




SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

SimPO: Simple preference optimization

DPO



Concurrent work:

Preference Learning Algorithms Do Not Learn Preference Rankings

Angelica Chen
New York University
ac5968@nyu.edu

Sadhika Malladi
Princeton University
smalladi@princeton.edu

Lily H. Zhang
New York University
lily.h.zhang@nyu.edu

Xinyi Chen
Google DeepMind; Princeton University
xinyic@google.com

Qiuyi Zhang
Google DeepMind
qiuyiz@google.com

Rajesh Ranganath
New York University
rajeshr@cims.nyu.edu

Kyunghyun Cho
New York University; Genentech; CIFAR LMB
kyunghyun.cho@nyu.edu

There is a discrepancy between reward function and decoding metrics

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

The SimPO objective

- We simply use the **average** log-likelihood as the reward function:

$$r_{\text{SimPO}}(x, y) = \frac{\beta}{|y|} \log \pi_\theta(y \mid x) = \frac{\beta}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i})$$

  - Why length normalized?

    - Shorter sequences tend to have larger log-likelihood
    - This metric is commonly used to rank options for multiple-choice questions

- We introduce target reward margin in Bradley-Terry ranking objective:

$$p(y_w \succ y_l \mid x) = \sigma\left(r(x, y_w) - r(x, y_l) - \boxed{\gamma}\right)$$

Encourages a large margin between winning and losing rewards

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

The SimPO objective

SimPO Objective

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

$$\mathcal{L}_{\mathbf{SimPO}}(\pi_\theta) = -\mathbb{E}\left[\log \sigma\left(\frac{\beta}{|y_w|} \log \pi_\theta(y_w \mid x) - \frac{\beta}{|y_l|} \log \pi_\theta(y_l \mid x) - \gamma\right)\right]$$

$$r_{\mathrm{SimPO}}(x, y) = \frac{\beta}{|y|} \log \pi_\theta(y \mid x)$$

$$\Downarrow$$

$$p(y_w \succ y_l \mid x) = \sigma\left(r(x, y_w) - r(x, y_l) - \gamma\right)$$

$$\Downarrow$$

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

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

A close look at evaluation

- Two model families: **LLama-3-8B** and **Mistral-7B**
  - We consider two models from each family: **base** and **instruct**
  - **Base**: pre-trained checkpoints
  - **instruct**: instruction-tuned models / procedure and training data are opaque

- The **Base** setting - similar to Zephyr (Tunstall et al., 2023)
  - First fine-tune on **UltraChat** (200k examples; Ding et al., 2023) for instruction tuning
  - Then train on **UltraFeedback** (64k prompts; Cui et al., 2023) for preference optimization

UltraChat is a multi-turn conversational dataset generated by GPT-3.5-turbo covering 30 topics and different types of texts

UltraFeedback takes prompts from diverse sources (e.g., UltraChat, FLAN), and generates responses from 4 different LLMs. Use GPT-4 to score instruction-following, truthfulness, honesty and helpfulness.
winning=highest score, losing=random of remaining 3

A close look at evaluation

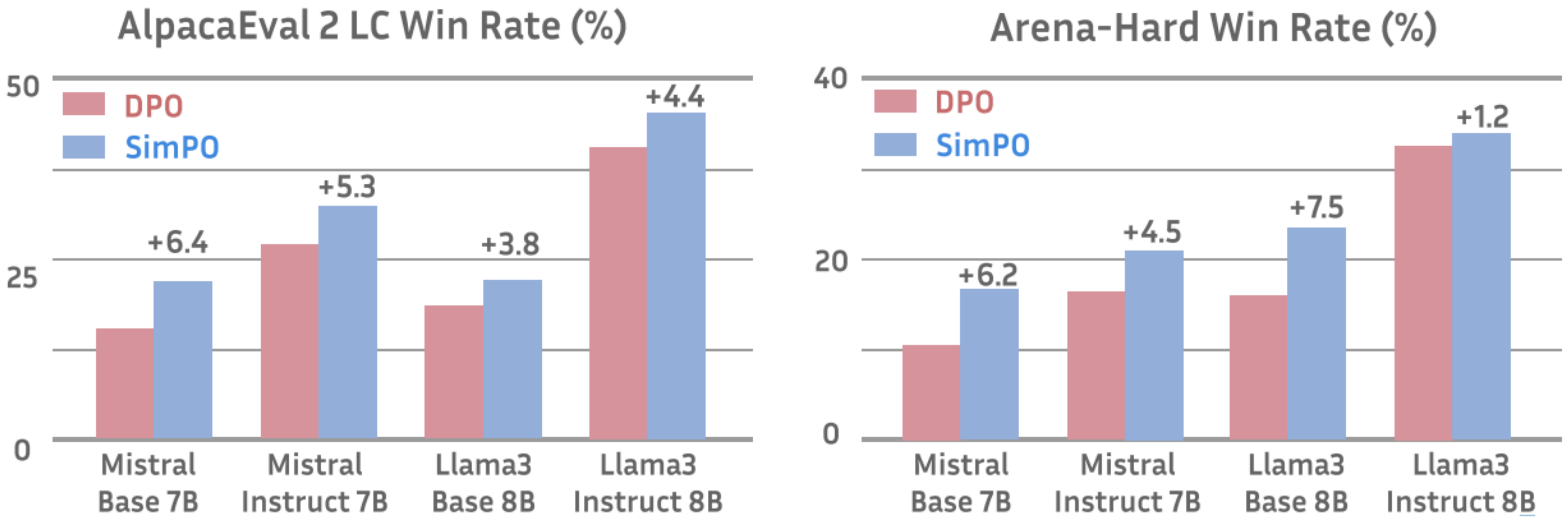
- The **Instruct** setting
  - We take this instruction-tuned model as the SFT model
  - We use it to regenerate 5 responses for each of **UltraFeedback** prompts, using an **off-the-shelf reward model PairRM** (Jiang et al., 2023) to pick the highest score one as **winning response**, and lowest score as **losing response**
    - The preference data is generated by the SFT model (on-policy)!
    - There is one extra **reward model** introduced (DeBERTa-v3-large)

- **Evaluation**

	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
AlpacaEval 2	805	GPT-4 Turbo	GPT-4 Turbo	Pairwise comparison	LC & raw win rate
Arena-Hard	500	GPT-4-0314	GPT-4 Turbo	Pairwise comparison	Win rate
MT-Bench	80	-	GPT-4/GPT-4 Turbo	Single-answer grading	Rating of 1-10

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

Main results: SimPO vs DPO



- **SimPO** vs **DPO**: Consistent and significant gains (have results of other *PO methods in the paper)

- You can build a quite strong chat model by using open-sourced datasets
  - Alpaca Eval 2 LC WR: **Llama3-8B-base+SimPO 22.0%** vs Llama-3-8B-instruct 26.0%

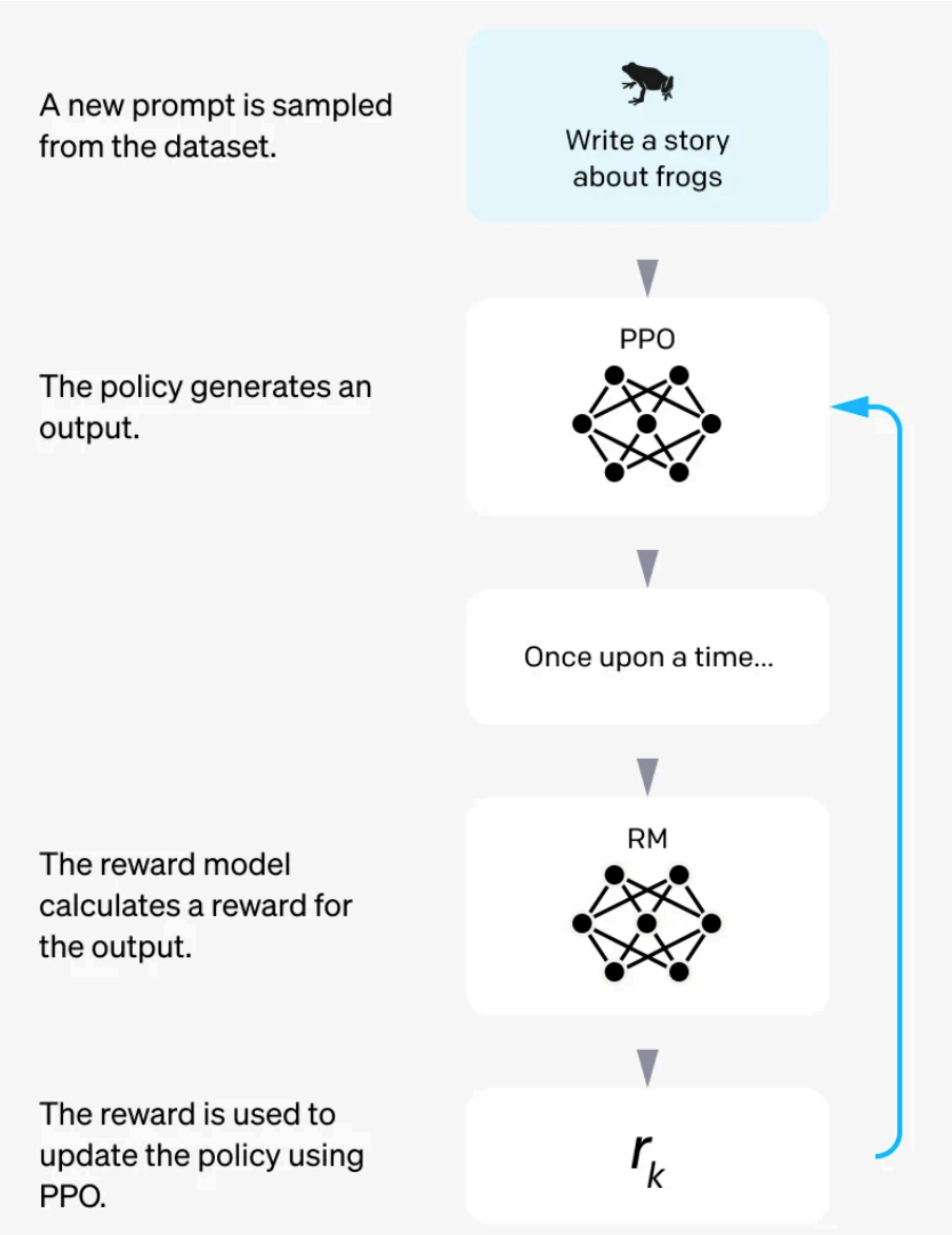
- You can turn **a strong instruction-tuned model** into a much stronger one by generating on-policy data!
  - Alpaca Eval 2 LC WR: Llama-3-8B-instruct 26.0% vs **Llama-3-8B-instruct+SimPO 44.7%**

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

SimPO: additional results

The only change is the **reward model** that helps generate preference data: RLHFlow/ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024)

Model	LC (%)	WR (%)	Len.
GPT-4 Turbo (04/09)	55.0	46.1	1802
Llama3-Instruct-8B-SimPO-v0.2	53.7	47.5	1777
GPT-4 Turbo (11/06)	50.0	50.0	2049
Llama3-Instruct-8B-SimPO	44.7	40.5	1825
Claude 3 Opus	40.5	29.1	1388
Llama3-Instruct-8B-DPO	40.3	37.9	1837
Llama3-Instruct-70B	34.4	33.2	1919
Llama3-Instruct-8B	26.0	25.3	1899
GPT-3.5 Turbo (06/13)	22.7	14.1	1328



SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

SimPO: additional results

SFT model: gemma2-9b-it

Benchmark	Performance	Ranking	<10b Ranking
AlpacaEval 2	72.4	1	1
Arena-Hard	59.1	14	1
WildBench	1166.6	21	1
ZeroEval GSM	88.0	--	--
ZeroEval MMLU	72.2	--	--

https://pli.princeton.edu/blog/2024/what-we-have-learned-simpo

How does preference learning impact general capacity of models (e.g., math, reasoning)?

What makes llama3-8b vs gemma2-9b behave differently?

SimPO: Simple Preference Optimization with a Reference-Free Reward (2024)

The rivalry between PPO and DPO

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

Shusheng Xu [1] **Wei Fu** [1] **Jiaxuan Gao** [1] **Wenjie Ye** [2] **Weilin Liu** [2]
Zhiyu Mei [1] **Guangju Wang** [2] **Chao Yu** [*1] **Yi Wu** [*123]

Unpacking DPO and PPO: Disentangling Best Practices for Learning from Preference Feedback

Hamish Ivison[♣♠] **Yizhong Wang**[♣♠] **Jiacheng Liu**[♣♠]
Zeqiu Wu[♠] **Valentina Pyatkin**[♣♠] **Nathan Lambert**[♣]
Noah A. Smith[♣♠] **Yejin Choi**[♣♠] **Hannaneh Hajishirzi**[♣♠]

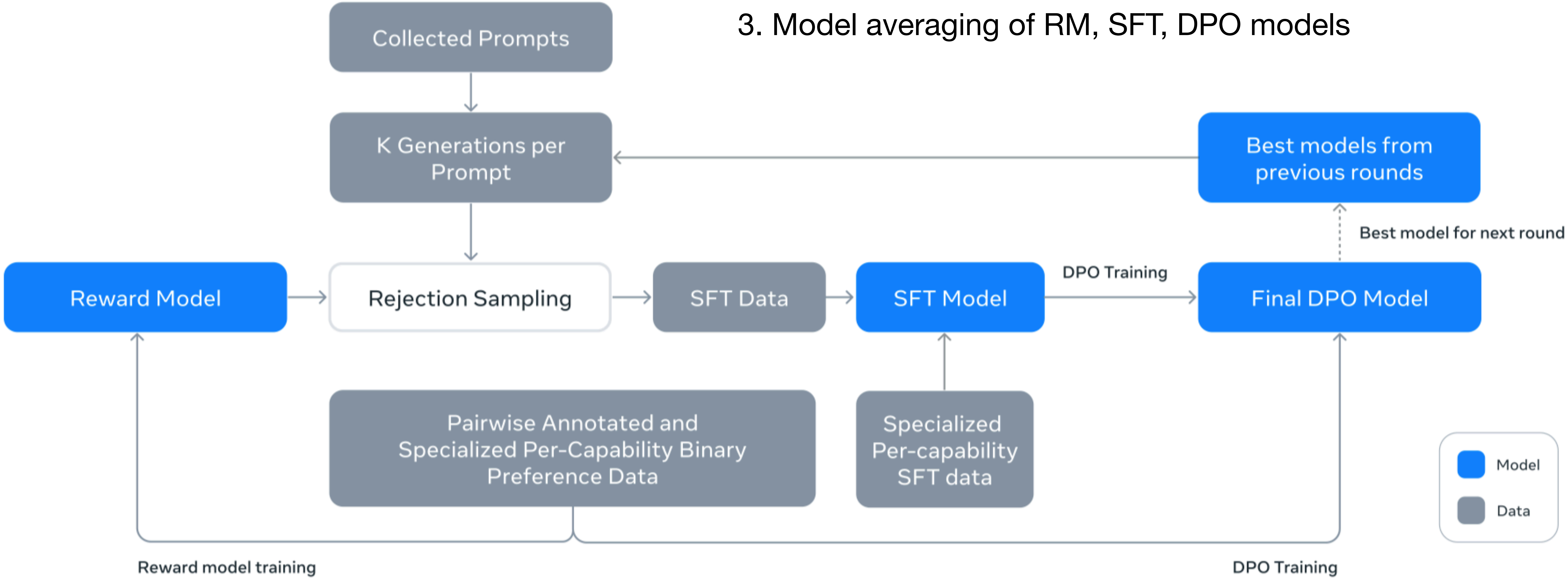
[♣]Allen Institute for AI [♠]University of Washington
hamishiv@cs.washington.edu

Importance in ranked order:

1. **preference data quality**
2. **algorithm choice**
3. **reward model quality**
4. targeted policy training prompts

Post-training pipeline of Llama3

1. Multiple rounds of data generation and model training
2. Use on-policy data than static pre-generated data
3. Model averaging of RM, SFT, DPO models



The Llama 3 Herd of Models (2024)

Part I. **Pre-training**

“the model is trained at **massive scale** using straightforward tasks such as **next-word prediction**”

Pre-training

- Step 1. Prepare a high-quality, tokenized **pre-training corpus** (internet scale)
- Step 2. Decide (Transformer) **model architecture** and **context window size**
- Step 3. Fit the model on the pre-training corpus to maximize log-likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

Llama-3: "We pre-train a model with **405B** parameters on **15.6T** tokens using a context window of **8K** tokens. This standard pre-training stage is followed by a continued pre-training stage that increases the supported context window to **128K** tokens."

Open research questions

- How to curate and filter **high-quality pre-training data**?
- What is a good **data mixture**?
- What is a good **training recipe**? How many stages of training? What data to use?
- **Scaling laws** for determining model sizes and data mix?
- How and when to use **synthetic data**?

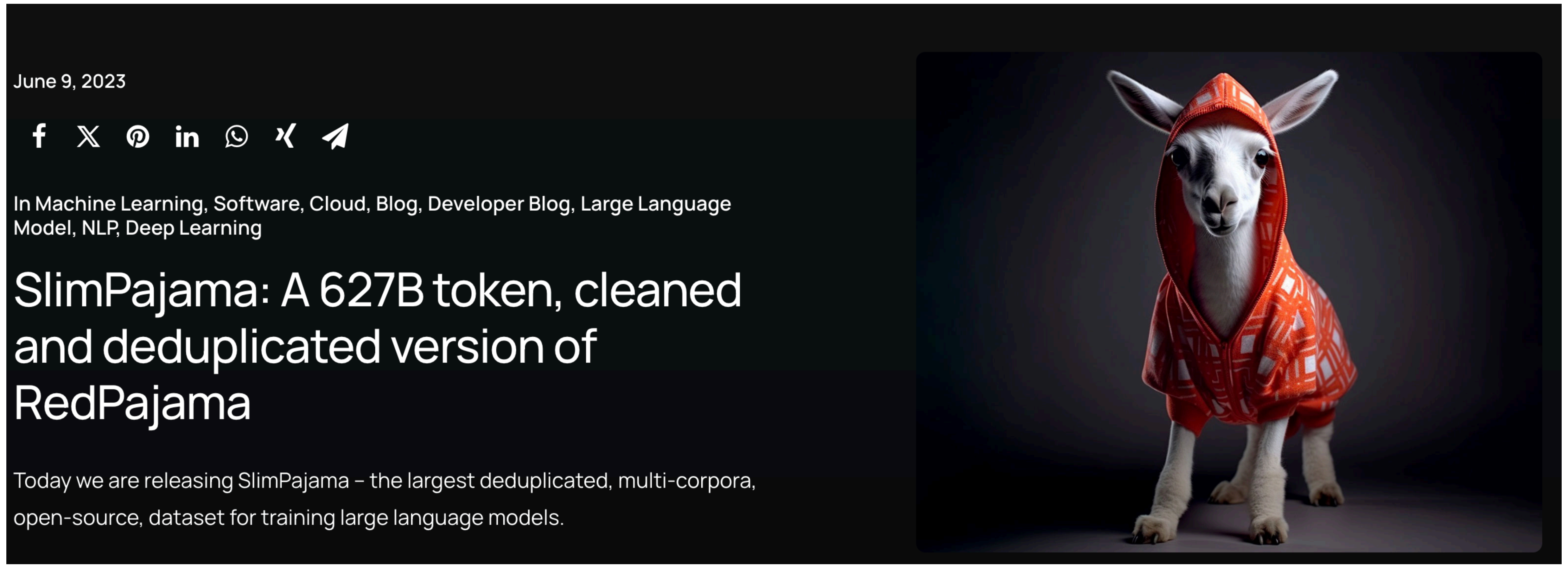
Pre-training corpora in the open

RedPajama

2023-04-17

Dataset	Token Count
Commoncrawl	878 Billion
C4	175 Billion
GitHub	59 Billion
Books	26 Billion
ArXiv	28 Billion
Wikipedia	24 Billion
StackExchange	20 Billion
Total	1.2 Trillion

RedPajama (1.2T) ⟶ SlimPajama (627B)



June 9, 2023

In Machine Learning, Software, Cloud, Blog, Developer Blog, Large Language Model, NLP, Deep Learning

SlimPajama: A 627B token, cleaned and deduplicated version of RedPajama

Today we are releasing SlimPajama – the largest deduplicated, multi-corpora, open-source, dataset for training large language models.

Pre-training corpora in the open

OLMo Data

2023-08-18

Source	Doc Type	Llama tokens *(billions)*
Common Crawl	web pages	2,281
The Stack	code	411
C4	web pages	198
Reddit	social media	89
PeS2o	STEM papers	70
Project Gutenberg	books	6.0
Wikipedia, Wikibooks	encyclopedic	4.3

Total = 3T tokens

Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research (2023)

Pre-training corpora in the open

2023-06-01

[Submitted on 1 Jun 2023]

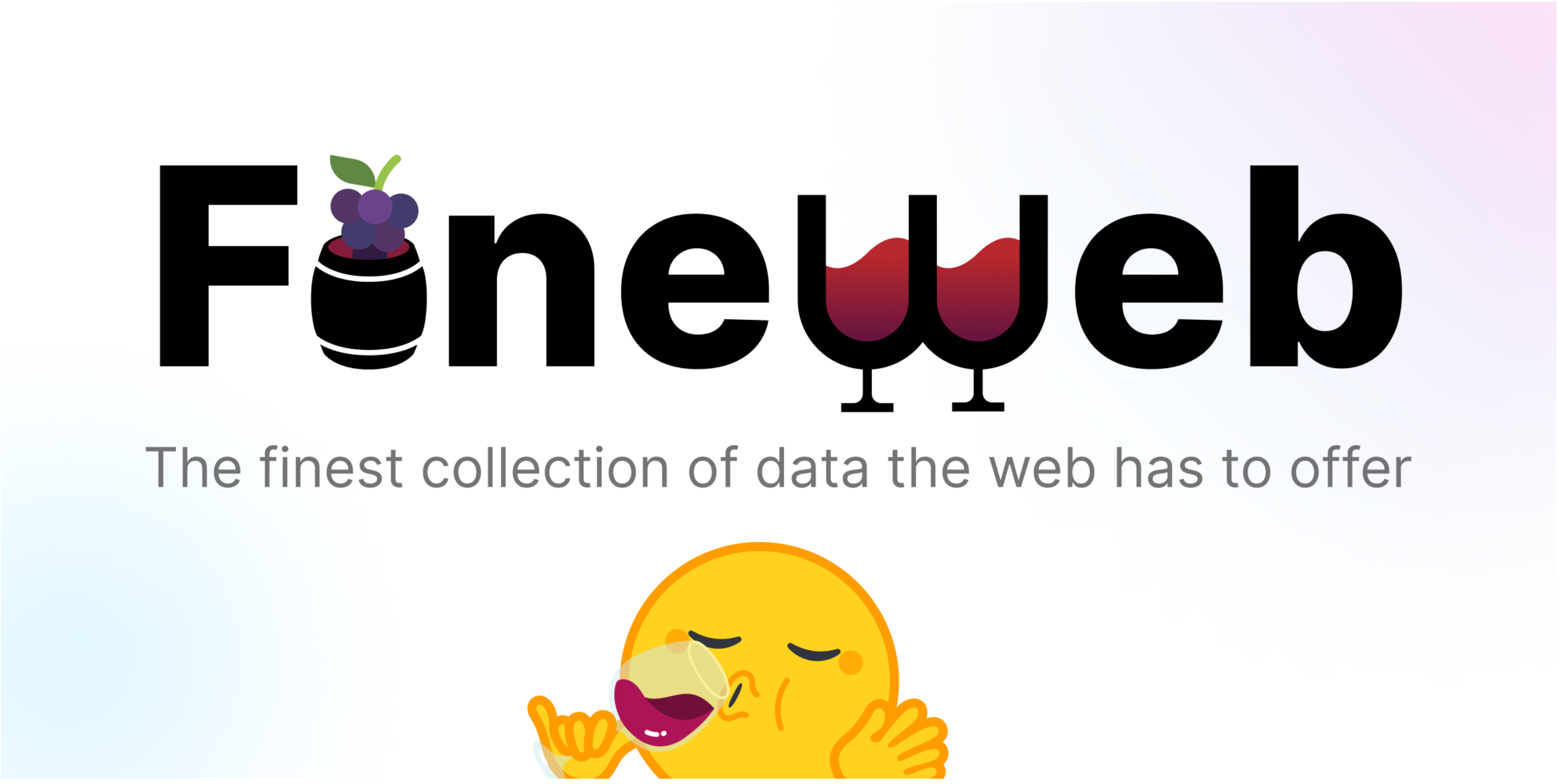
The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, Julien Launay

Large language models are commonly trained on a mixture of filtered web data and curated high-quality corpora, such as social media conversations, books, or technical papers. This curation process is believed to be necessary to produce performant models with broad zero-shot generalization abilities. However, as larger models requiring pretraining on trillions of tokens are considered, it is unclear how scalable is curation and whether we will run out of unique high-quality data soon. At variance with previous beliefs, we show that properly filtered and deduplicated web data alone can lead to powerful models; even significantly outperforming models from the state-of-the-art trained on The Pile. Despite extensive filtering, the high-quality data we extract from the web is still plentiful, and we are able to obtain five trillion tokens from CommonCrawl. We publicly release an extract of 600 billion tokens from our RefinedWeb dataset, and 1.3/7.5B parameters language models trained on it.

Total = 600B tokens (only from Common Crawl)

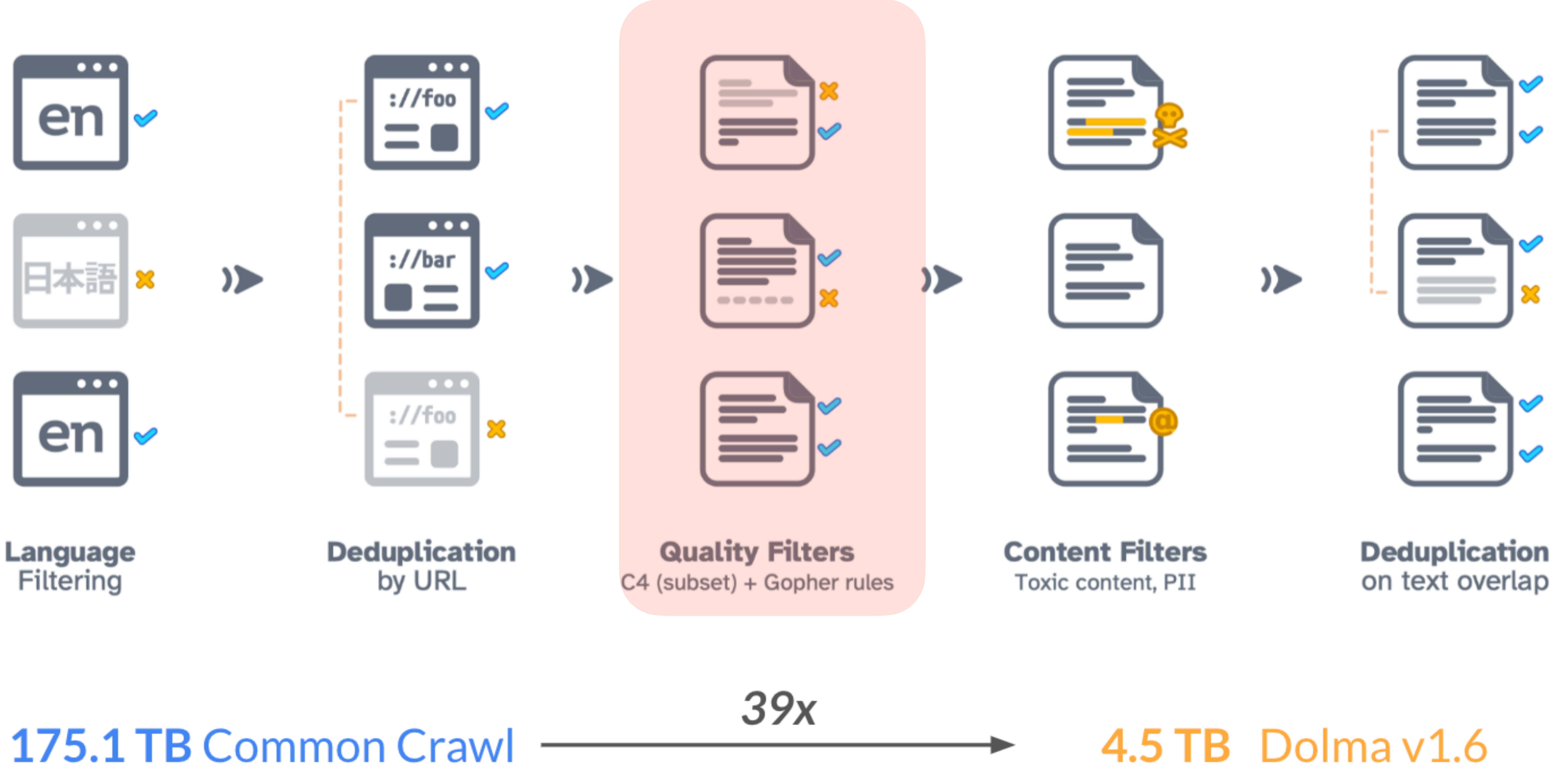
2024-05-31



Total = 15T tokens

Also: FineWeb-edu (1.3T and 5.4T)

Data processing pipeline: example



Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research

Data processing pipeline: example

Quality filters - mostly heuristics based

C4 rules (Raffel et al., 2020)

- We only retained lines that ended in a terminal punctuation mark (i.e. a period, exclamation mark, question mark, or end quotation mark).
- We discarded any page with fewer than 5 sentences and only retained lines that contained at least 3 words.
- We removed any page that contained any word on the "List of Dirty, Naughty, Obscene or Otherwise Bad Words".[6]
- Many of the scraped pages contained warnings stating that Javascript should be enabled so we removed any line with the word Javascript.
- Some pages had placeholder "lorem ipsum" text; we removed any page where the phrase "lorem ipsum" appeared.
- Some pages inadvertently contained code. Since the curly bracket "{" appears in many programming languages (such as Javascript, widely used on the web) but not in natural text, we removed any pages that contained a curly bracket.

New direction: ***model-based quality filters***

Gopher Rules (Rae et al., 2021)

```python
def gopher_rules_pass(sample) -> bool:
    """ function returns True if the sample complies with Gopher rules """
    signals = json.loads(sample["quality_signals"])

    # rule 1: number of words between 50 and 10'000
    word_count = signals["rps_doc_word_count"][0][2]
    if word_count < 50 or word_count > 10_000:
        return False

    # rule 2: mean word length between 3 and 10
    mean_word_length = signals["rps_doc_mean_word_length"][0][2]
    if mean_word_length < 3 or mean_word_length > 10:
        return False

    # rule 2: symbol to word ratio below 0.1
    symbol_word_ratio = signals["rps_doc_symbol_to_word_ratio"][0][2]
    if  symbol_word_ratio > 0.1:
        return False

    # rule 3: 90% of lines need to start without a bullet point
    n_lines = signals["ccnet_nlines"][0][2]
    n_lines_bulletpoint_start = sum(map(lambda ln: ln[2], signals["rps_lines_start_w
    if n_lines_bulletpoint_start / n_lines > 0.9:
        return False

    # rule 4: the ratio between characters in the most frequent 2-gram and the total
    # of characters must be below 0.2
    top_2_gram_frac = signals["rps_doc_frac_chars_top_2gram"][0][2]
    if top_2_gram_frac > 0.2:
        return False

    # rule 5: ...
```

QuRating: Selecting high-quality data with LM signals

- What is the notion of high-quality data in pre-training corpora?
- Can we detect such high-quality data effectively and efficiently?

Choices of quality criteria:

1. are applicable to **a wide variety** of texts
2. require a **deeper understanding** of the content of a text
3. have many subtle **gradations** in quality
4. are **complementary** to each other

Writing style, **facts & trivia**, **educational value**, **required expertise**

QuRating: Selecting High-Quality Data for Training Language Models (2024)

QuRating: Selecting high-quality data with LM signals

Part I
measure quality

Part II
utilize quality

QuRating: Selecting High-Quality Data for Training Language Models (2024)

Pairwise comparisons ⟹ Quality ratings

- We validate on 80 documents with clear differences in quality. GPT-3.5-turbo achieves 92-99% agreement ✅
- The criteria are only weakly correlated (correlation coeff. 0.29-0.55) ✅

QuRating: Selecting High-Quality Data for Training Language Models (2024)

Training the QuRater model

- For each criterion, we collect 250K pairwise judgments from GPT-3.5-turbo for documents from SlimPajama

$$\mathcal{J} = \{(t_i, t_j, p_{i \succ j})\}$$

- We use the Bradley-Terry model to obtain scalar quality ratings
- Fine-tune a *1.3B QuRater* model to predict these quality ratings

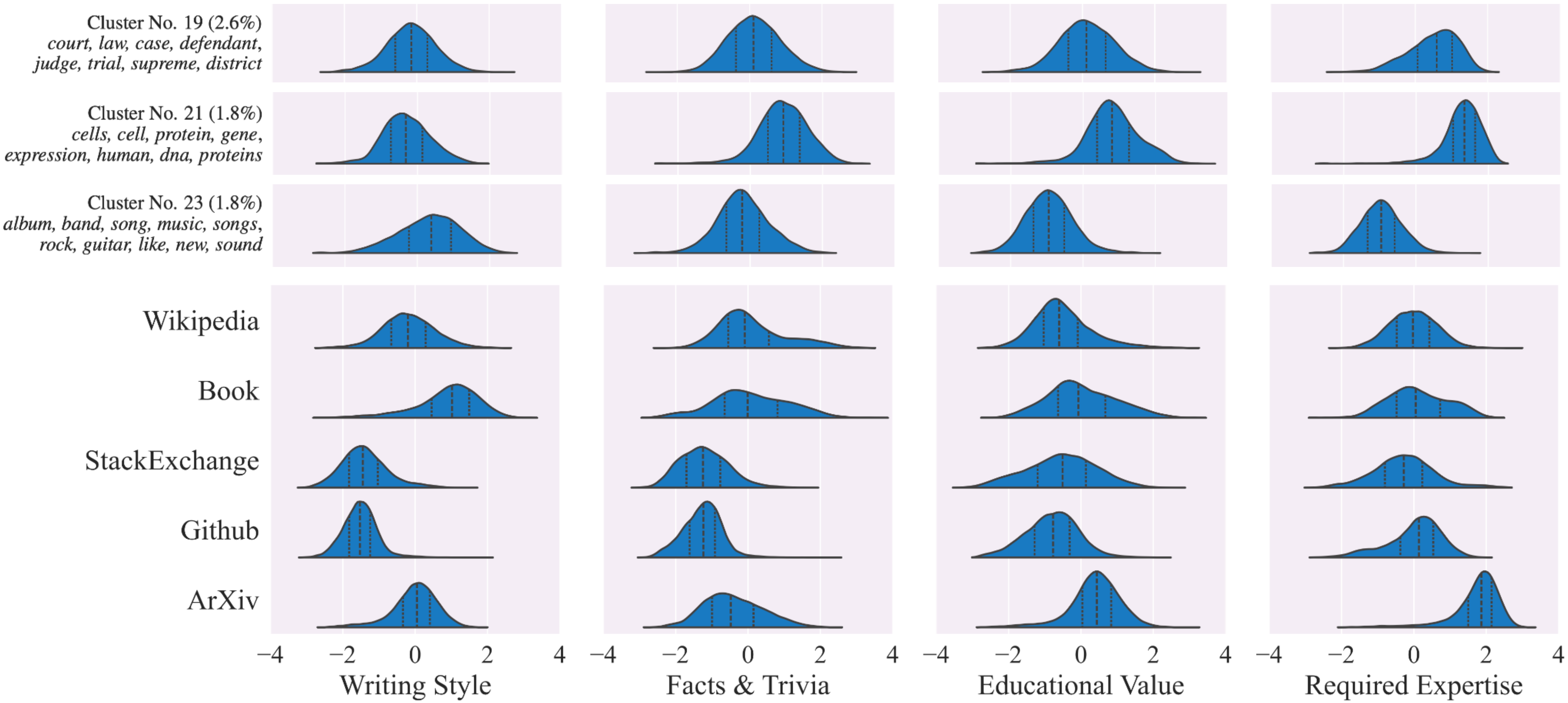
$$p_{B \succ A} = \sigma\left(s_B - s_A\right)$$

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{(t_A, t_B, p_{B \succ A}) \in \mathcal{J}} \left[-p_{B \succ A} \log \sigma\left(s_\theta(t_B) - s_\theta(t_A)\right) - (1 - p_{B \succ A}) \log \sigma\left(s_\theta(t_A) - s_\theta(t_B)\right) \right]$$

- Fine-tuned model achieves >93% validation accuracy ✅
- Use QuRater to annotate 260B token corpus based on SlimPajama
  ⇒ *QuRatedPajama*

QuRating: Selecting High-Quality Data for Training Language Models (2024)

A quick dive into QuRatedPajama



Sampling with quality signals improves performance

Quality vs **diversity**:

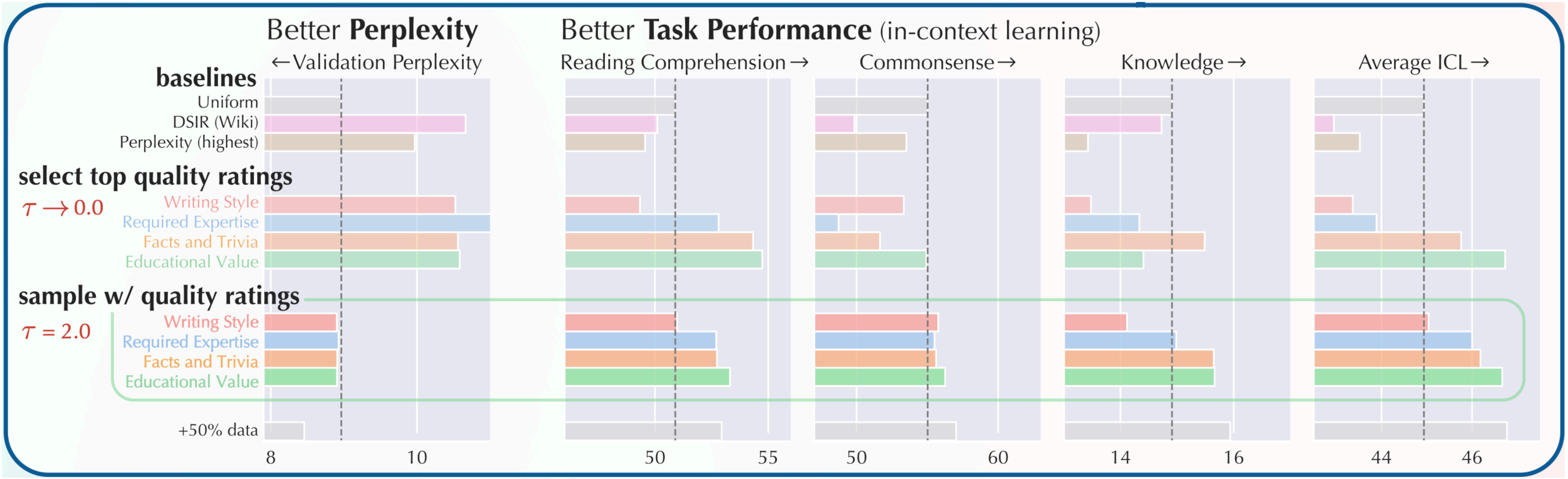
Sample documents *without replacement* from

$$p(\text{document}) \propto \exp(\text{quality rating}/\tau)$$

Temperature τ balances **quality** and **diversity**

$\tau \rightarrow 0.0$: top-k selection / $\tau \rightarrow \infty$: uniform sampling

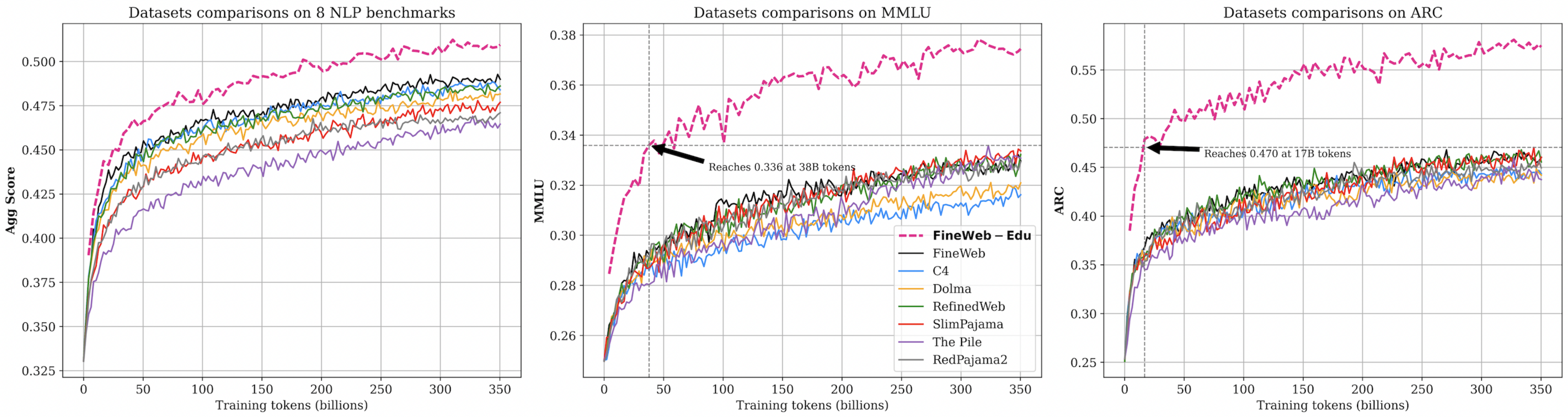
- Selecting **30B** out of **260B** tokens
- Training **1.3B models** from scratch



QuRating: Selecting High-Quality Data for Training Language Models (2024)

FineWeb-edu

- Using synthetic data to develop classifiers for identifying **educational content**
- **450k annotations** generated by **LLama3-70B-instruct** for web samples from FineWeb dataset



https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu

FineWeb-edu

Below is an extract from a web page. Evaluate whether the page has a high educational value and could be useful in an educational setting for teaching from primary school to grade school levels using the additive 5-point scoring system described below. Points are accumulated based on the satisfaction of each criterion:

- Add 1 point if the extract provides some basic information relevant to educational topics, even if it includes some irrelevant or non-academic content like advertisements and promotional material.
- Add another point if the extract addresses certain elements pertinent to education but does not align closely with educational standards. It might mix educational content with non-educational material, offering a superficial overview of potentially useful topics, or presenting information in a disorganized manner and incoherent writing style.
- Award a third point if the extract is appropriate for educational use and introduces key concepts relevant to school curricula. It is coherent though it may not be comprehensive or could include some extraneous information. It may resemble an introductory section of a textbook or a basic tutorial that is suitable for learning but has notable limitations like treating concepts that are too complex for grade school students.
- Grant a fourth point if the extract highly relevant and beneficial for educational purposes for a level not higher than grade school, exhibiting a clear and consistent writing style. It could be similar to a chapter from a textbook or a tutorial, offering substantial educational content, including exercises and solutions, with minimal irrelevant information, and the concepts aren't too advanced for grade school students. The content is coherent, focused, and valuable for structured learning.
- Bestow a fifth point if the extract is outstanding in its educational value, perfectly suited for teaching either at primary school or grade school. It follows detailed reasoning, the writing style is easy to follow and offers profound and thorough insights into the subject matter, devoid of any non-educational or complex content.

The extract: <extract>.

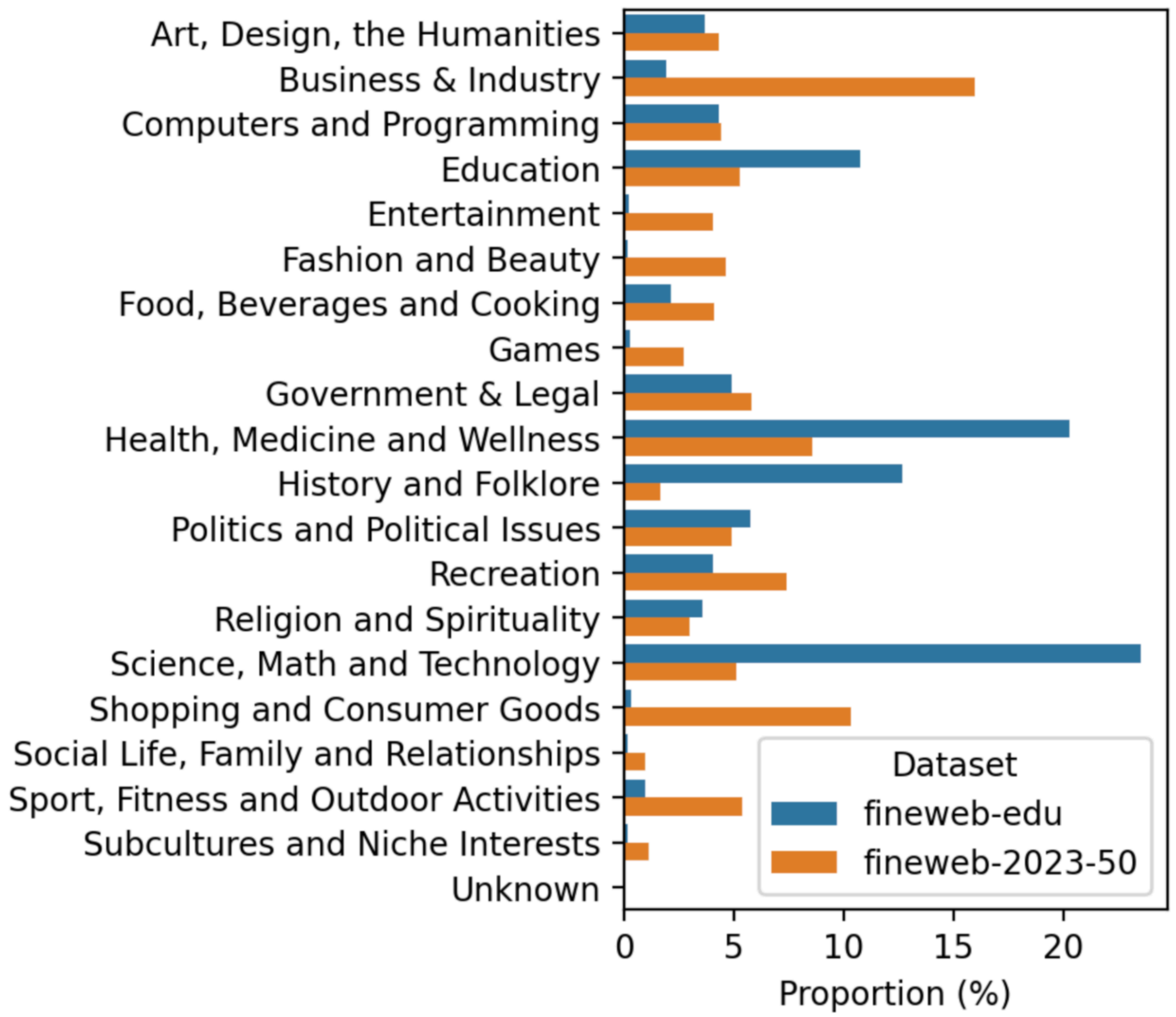
After examining the extract:

- Briefly justify your total score, up to 100 words.
- Conclude with the score using the format: "Educational score: <total points>"

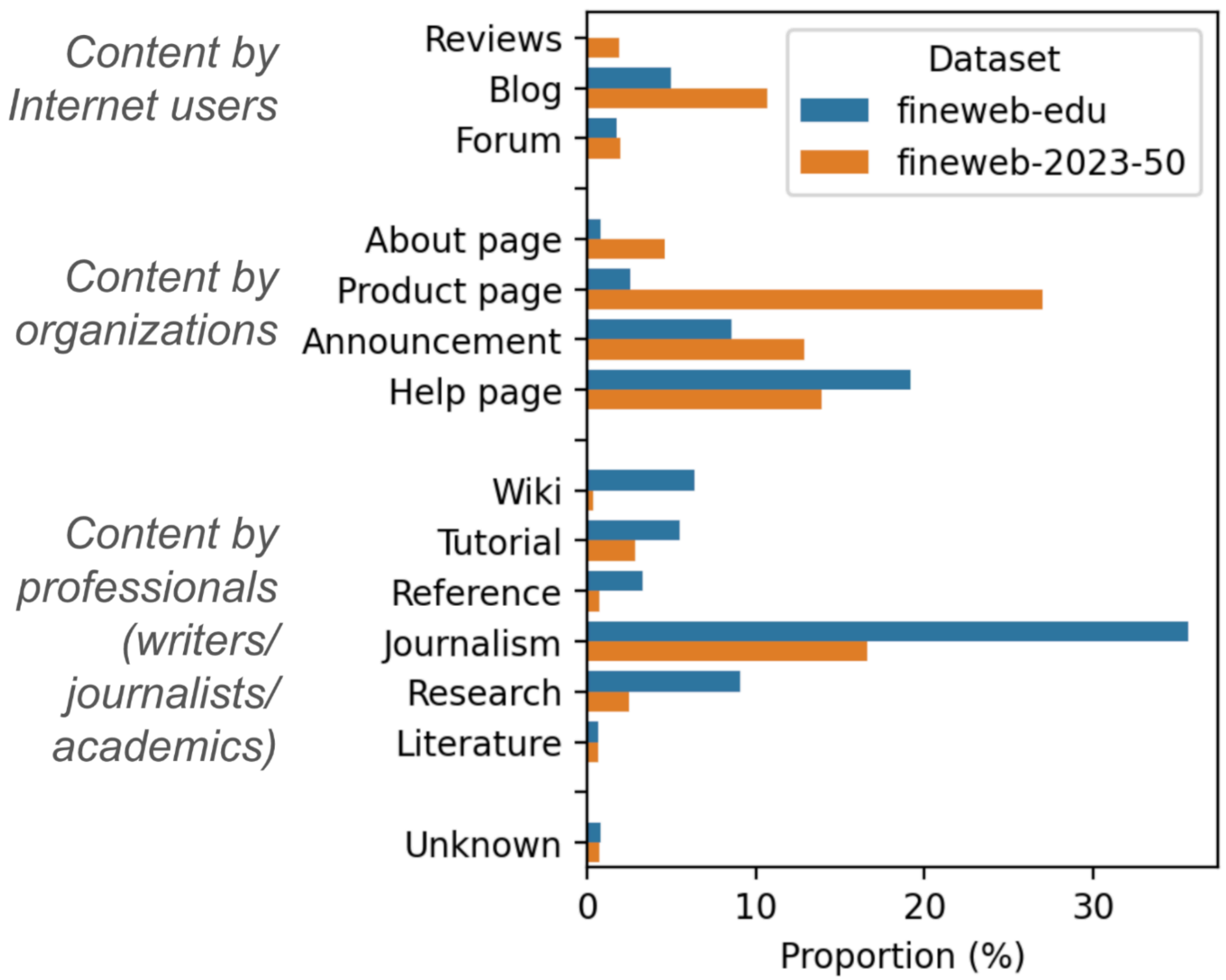
https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu

FineWeb-edu

20 topics



14 formats



The DCLM competition

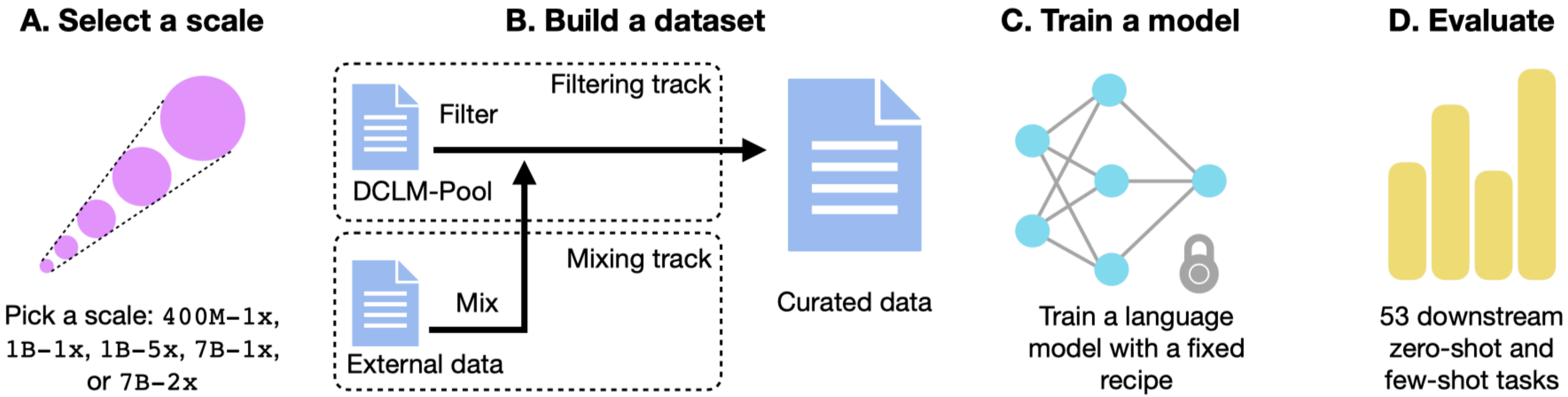
DataComp-LM: In search of the next generation of training sets for language models

Jeffrey Li*[1,2] Alex Fang*[1,2] Georgios Smyrnis*[4] Maor Ivgi*[5]
Matt Jordan[4] Samir Gadre[3,6] Hritik Bansal[8] Etash Guha[1,15] Sedrick Keh[3] Kushal Arora[3]
Saurabh Garg[13] Rui Xin[1] Niklas Muennighoff[22] Reinhard Heckel[12] Jean Mercat[3] Mayee Chen[7] Suchin Gururangan[1] Mitchell Wortsman[1] Alon Albalak[19,20] Yonatan Bitton[14]
Marianna Nezhurina[9,10] Amro Abbas[23] Cheng-Yu Hsieh[1] Dhruba Ghosh[1] Josh Gardner[1]
Maciej Kilian[17] Hanlin Zhang[18] Rulin Shao[1] Sarah Pratt[1] Sunny Sanyal[4] Gabriel Ilharco[1]
Giannis Daras[4] Kalyani Marathe[1] Aaron Gokaslan[16] Jieyu Zhang[1] Khyathi Chandu[11]
Thao Nguyen[1] Igor Vasiljevic[3] Sham Kakade[18] Shuran Song[6,7] Sujay Sanghavi[4] Fartash Faghri[2] Sewoong Oh[1] Luke Zettlemoyer[1] Kyle Lo[11] Alaaeldin El-Nouby[2] Hadi Pouransari[2] Alexander Toshev[2] Stephanie Wang[1] Dirk Groeneveld[11] Luca Soldaini[11]
Pang Wei Koh[1] Jenia Jitsev[9,10] Thomas Kollar[3] Alexandros G. Dimakis[4,21]
Yair Carmon[5] Achal Dave†[3] Ludwig Schmidt†[1,7] Vaishaal Shankar†[2]

[1]University of Washington, [2]Apple, [3]Toyota Research Institute, [4]UT Austin, [5]Tel Aviv University, [6]Columbia University, [7]Stanford, [8]UCLA, [9]JSC, [10]LAION, [11]AI2, [12]TUM, [13]CMU, [14]Hebrew University, [15]SambaNova, [16]Cornell, [17]USC, [18]Harvard, [19]UCSB, [20]SynthLabs, [21]Bespokelabs.AI, [22]Contextual AI, [23]DatologyAI

DataComp-LM: In search of the next generation of training sets for language models (2024)

The DCLM competition

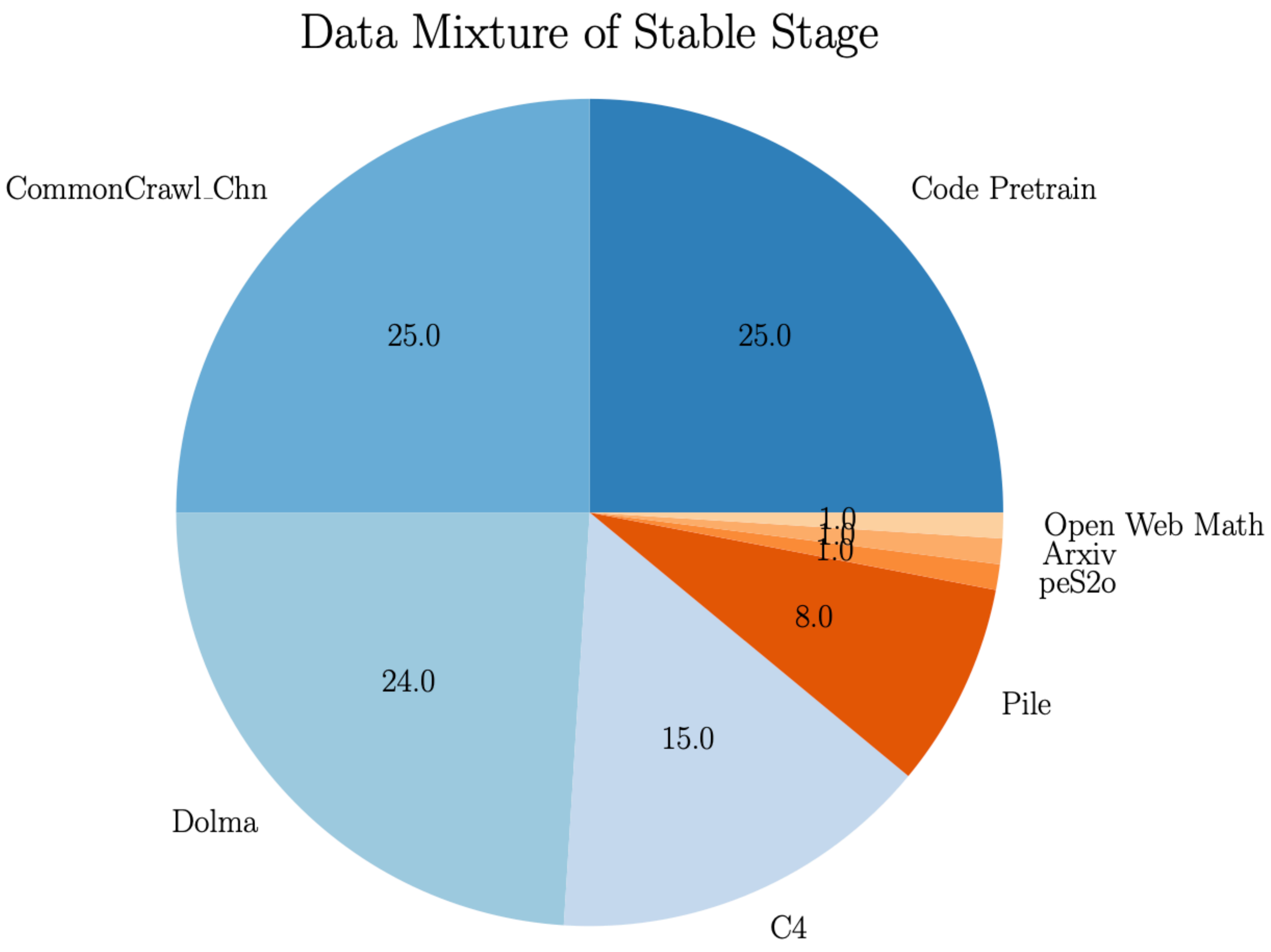


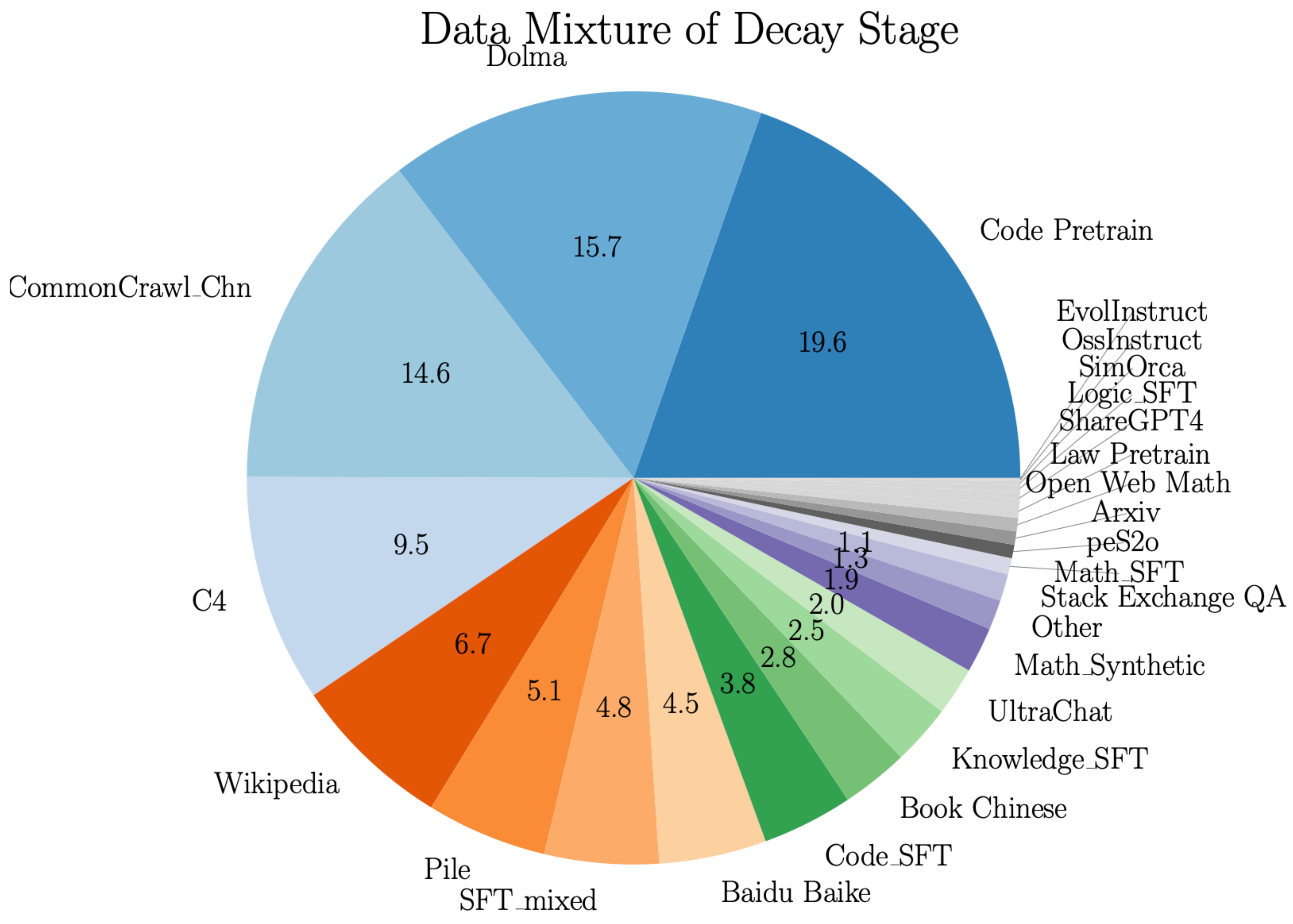
"As a baseline for DCLM, we conduct extensive experiments and find that **model-based filtering** is key to assembling a high-quality training set."

DataComp-LM: In search of the next generation of training sets for language models (2024)

Domain mixture and multi-staged training

Domains: Common Crawl, CC, Github, Wikipedia, Books, arXiv, …





MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies (2024)

Domain mixture and multi-staged training

Data mixture in 2nd stage:

Category	Dataset	Percentage
NL pretraining data	Refinedweb Pile_Wikipedia Pile_StackExchange Pile_arXiv Pile_remaining Dolma_peS2o	39.8% 6.7% 4.8% 1.0% 5.1% 1.0%
NL SFT data	xP3x, OpenAssistant, OpenHermes UltraChat, Oasst-octopack	7.3%
Textbook	UltraTextbooks	4.8%
Code pretraining data	Starcoder Github	19.6%
Code SFT data	Magicoder-OSS, Magicoder-Evol Code-290k-ShareGPT, CommitPackFT Evol-Code Alpaca	3.8%
Math data	Open-web-math, algebraic-stack TemplateGSM, StackMathQA	5.8%

JetMOE (Shen et al., 2024)

- Increase high-quality data in the later stage(s) of pre-training
- The boundary between pre-training and SFT has blurred

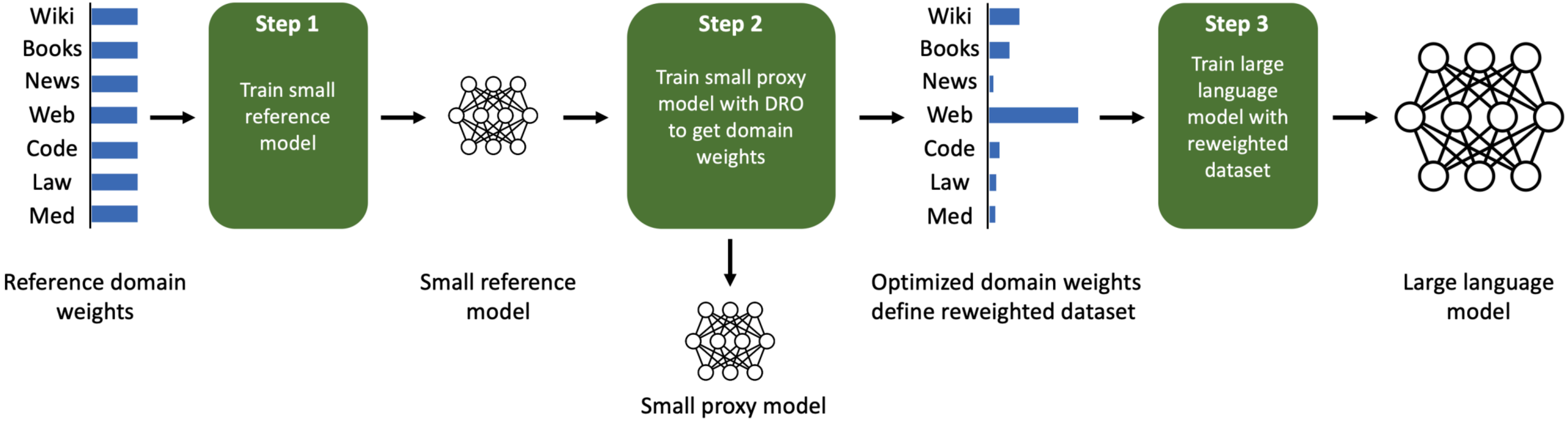
[Submitted on 5 Jun 2024]

Does your data spark joy? Performance gains from domain upsampling at the end of training

Cody Blakeney, Mansheej Paul, Brett W. Larsen, Sean Owen, Jonathan Frankle

"Upsampling domain-specific datasets in relative to CC at the end of training"

How to decide a good domain mixture?



DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining (2023)

How to decide a good domain mixture?

Dynamic batch loading: Load more data for domains where the loss reduction is slow

$$\Delta_t[i] \leftarrow \max\{\ell_t[i] - \ell_{\text{ref}}[i], 0\}$$

i: domain index

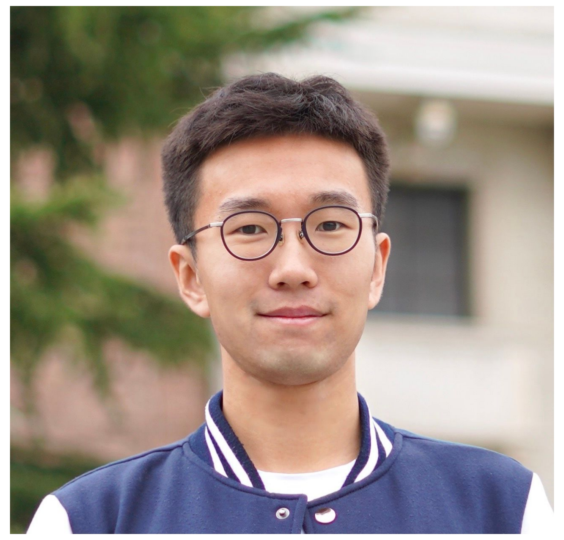
$$\alpha_t = \log(w_{t-m}) + \Delta_t$$

adjust domain weights after m steps

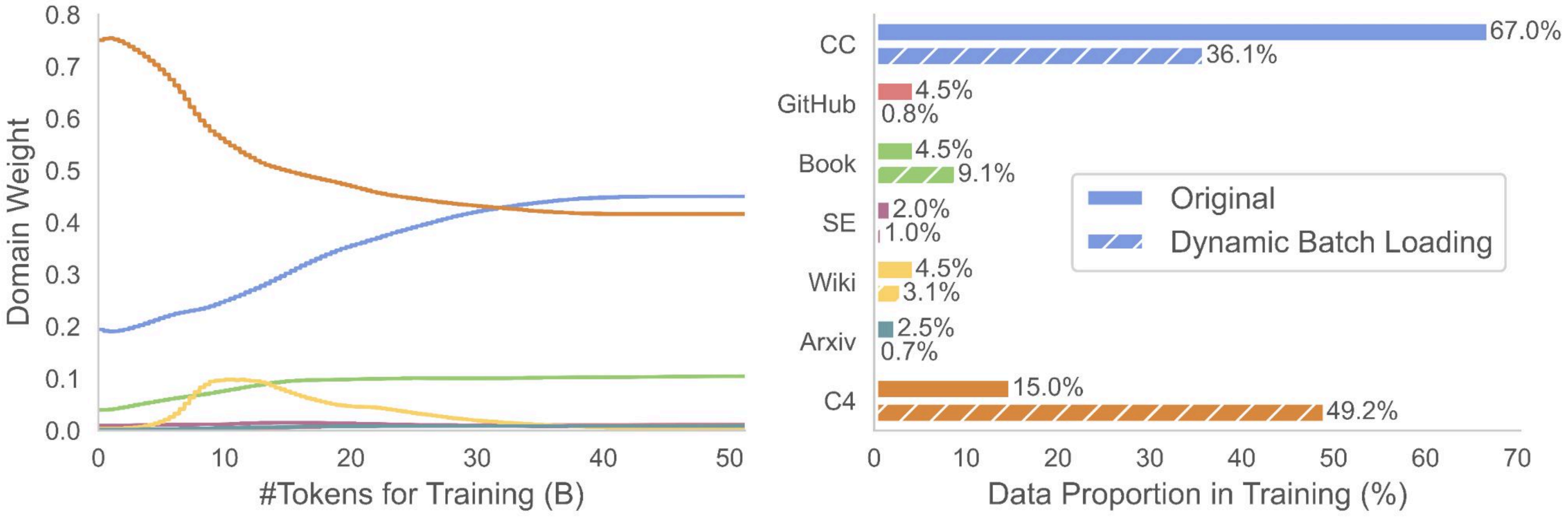
$$w_t = \frac{\exp(\alpha_t)}{\sum_i \exp(\alpha_t[i])}.$$

- No proxy models
- We estimated l_{ref} [I] using either LLaMa-2 checkpoints, or larger models

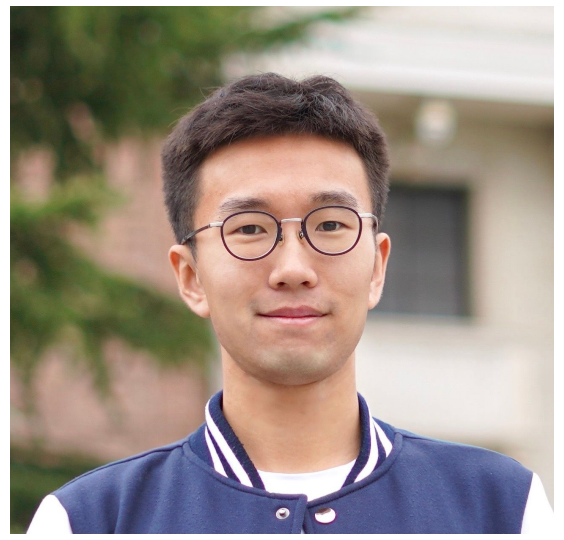
Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning (2024)



How to decide a good domain mixture?



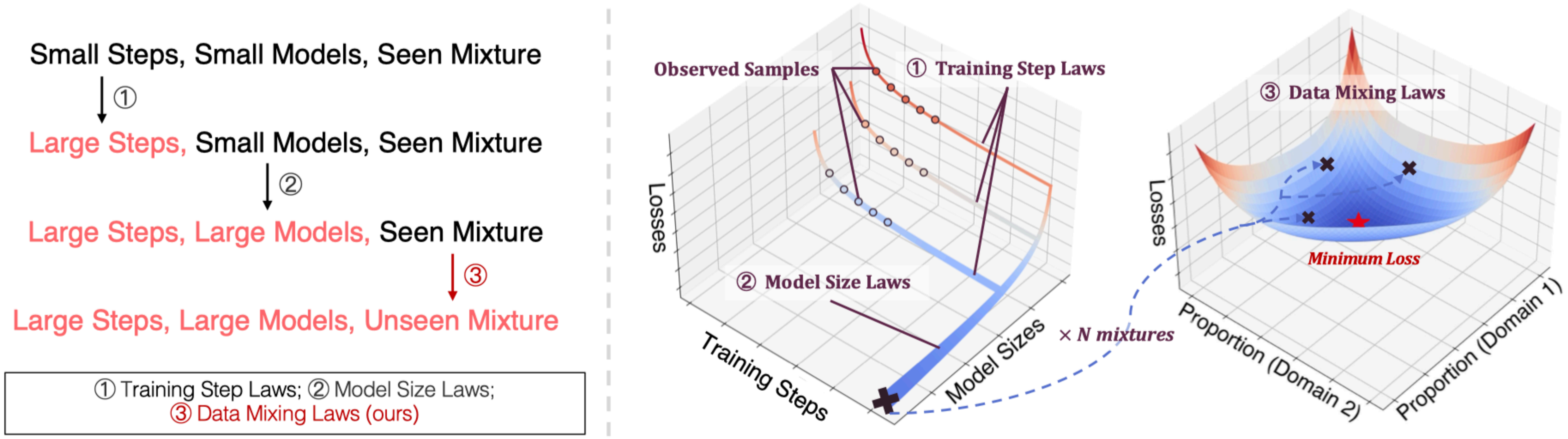
We used more data in C4 and Book, and less in any other domains!



Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning (2024)

Scaling laws for domain mixture

Llama 3.1: “To determine the best data mix, we perform scaling law experiments in which we **train several small models on a data mix** and use that to predict the performance of a large model on that mix. **We repeat this process multiple times for different data mixes to select a new data mix candidate.** Subsequently, we train a larger model on this candidate data mix and evaluate the performance of that model on several key benchmarks.”



Data Mixing Laws: Optimizing Data Mixtures by Predicting Language Modeling Performance (2024)

Active research topics in pre-training

- **Long-context pre-training**
  - Usually in a continued pre-training stage
  - We are running out of long-context data - how to mix short-context data and long-context data effectively? How to recover performance on short-context tasks?

- **The use of synthetic or semi-synthetic data in pre-training**
  - Rephrasing the Web (Maini et al., 2024)
  - Phi-3: “heavily filtered publicly available web data and synthetic LLM-generated data”

- **Model architectures beyond Transformers** (e.g., MoEs, Mamba, Jamba)

- **Training objectives beyond next-token prediction** (e.g., “fill in the middle”)

Questions?

- Thank you!
- Papers and models at https://www.cs.princeton.edu/~danqic/