LIONS: An Empirically Optimized Approach to Align Language Models

Xiao Yu*, **Qingyang Wu***, **Yu Li, Zhou Yu**
Columbia University
{xy2437, qw2345, yl5016, zy2461}@columbia.edu

Abstract

Alignment is a crucial step to enhance the instruction-following and conversational abilities of language models. Despite many recent work proposing new algorithms, datasets, and training pipelines, there is a lack of comprehensive studies measuring the impact of various design choices throughout the whole training process. We first conduct a rigorous analysis over a three-stage training pipeline consisting of supervised fine-tuning, offline preference learning, and online preference learning. We have found that using techniques like sequence packing, loss masking in SFT, increasing the preference dataset size in DPO, and online DPO training can significantly improve the performance of language models. We then train from Gemma-2b-base and LLama-3-8b-base, and find that our best models exceed the performance of the official instruct models tuned with closed-source data and algorithms. Our code and models can be found at https://github.com/Columbia-NLP-Lab/LionAlignment.

1 Introduction

Large language models (LLMs), pre-trained on datasets of trillion-scale tokens, have shown remarkable performance across a wide range of natural language processing tasks (Brown et al., 2020; OpenAI et al., 2024; Touvron et al., 2023; AI@Meta, 2024). However, these pre-trained models often struggle to follow human instructions and generate responses that are unsafe or inappropriate (Wei et al., 2023; Deshpande et al., 2023). Recent research has increasingly focused on aligning LLMs: this includes many new algorithms based on reinforcement learning (Ouyang et al., 2022a; Rafailov et al., 2023; Meng et al., 2024; Guo et al., 2024a), new datasets to facilitate preference learning (Cui et al., 2023; Banghua et al., 2023), and new training pipelines to improve the overall alignment performance (Tunstall et al., 2023b; Tran et al., 2023; Dong et al., 2024). Although these contributions demonstrate sizable improvements, the training processes, datasets, and hyper-parameters often remain heterogeneous or closed-source (Gemma et al., 2024; AI@Meta, 2024). This makes it difficult to pinpoint the source of improvements and limits the development of more effective or efficient alignment algorithms.

In this work, we replicate modern alignment pipelines (Tunstall et al., 2023b; Xu et al., 2024b) and analyze sources in the training process that could affect performance. In the three-stage training of supervised fine-tuning (SFT), offline preference learning, and online preference learning, we find that: 1) sequence packing and loss masking significantly enhance SFT, 2) scaling offline preference datasets improves overall performance, and 3) online learning greatly benefits chat benchmarks.

Then, we aggregate our findings and fine-tune from Gemma-2b-base (Gemma et al., 2024) and LLaMA-3-8b-base (AI@Meta, 2024) using publicly available datasets and open-source algorithms. We evaluate our models on popular benchmarks such as Arena-Hard (Zheng et al., 2023), AlpacaEval-2 (Li et al., 2023b), MT-Bench (Zheng et al., 2023), and OpenLLM (Beeching et al., 2023). In all benchmarks, our models exceed the performance of the officially instruct models, which rely on closed-source datasets and algorithms. We believe our easily reproducible study offers useful insights for alignment research, and our fine-tuned models are valuable for downstream applications.

Our contributions are:

- We present a rigorous analysis of modern alignment training pipelines, and identify a set of design choices that significantly impact the performance of language models.
- We aggregate our empirical findings into a step-by-step recipe, and show that our

* denotes equal contribution.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 8732–8753
November 12-16, 2024

models outperform the officially released instruct models, which relies on closed-source datasets and algorithms.

- We make our model, training dataset, and code publicly available for future research in language model alignment.

2 Preliminaries

Traditional reinforcement learning for human feedback (RLHF) methods typically starts with supervised fine-tuning, then trains a reward model r mimicking human preferences, and finally optimizes a language model π_θ to maximize the reward (Ziegler et al., 2020; Bai et al., 2022; Ouyang et al., 2022a). However, this process can be complex and difficult to tune. Many recent works (Tran et al., 2023; Xu et al., 2024b) have proposed alternative algorithms. In this work, we examine a three-stage RLHF pipeline: 1) supervised fine-tuning; 2) offline preference learning with DPO (Rafailov et al., 2023); and 3) online preference learning with DPO. Below, we review each training stage.

Supervised Fine-tuning Stage Supervised Fine-tuning (SFT) maximizes the log likelihood of the ground truth response y given a user's query x:

$$p(y|x) = \frac{1}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i|x, y_{<i}).$$

Efficiently optimizing the above objective is a challenging problem. We explore various strategies for SFT, which are detailed in Section 3.2.

Offline Preference Learning Stage In the second phase, the SFT model π^{SFT} is trained offline with a collection of human preference data. This is often done using DPO, which first re-parametrizes the reward r in terms of the optimal policy[1]:

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

where π_{ref} is a reference policy, β controls the strength of the KL-divergence, and $Z(x)$ is the partition function. Then, DPO optimizes the following objective treating the LMs as reward models:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma \left(r_w - r_l\right)\right].$$

[1]Under the formulation of reward maximization under a KL-divergence constraint

where (x, y_w, y_l) are preference pairs consisting of the prompt, the winning response, and the losing response, respectively; and $r_w = r(x_w, y_w)$, $r_l = r(x_l, y_l)$ are rewards given a choice of π_θ and π_{ref}. Despite its simplicity, practical concerns include the choice of reference model π_{ref}, strength of KL-divergence β, scalability of training, and data filtering. We will analyze these in Section 3.3.

Online Preference Learning Stage After offline preference learning, the model can be further fine-tuned with online preference pairs to improve its performance (Xu et al., 2024b; Guo et al., 2024b; Tran et al., 2023). Similar to the traditional RLHF pipeline, the process includes: 1) sampling multiple responses from π_θ; 2) use a reward model or judge to rank the responses; and 3) optimize π_θ using DPO. In Section 3.4, we will examine the effectiveness of further online learning against the simple offline training process.

3 Alignment Procedure Analysis

This section explores and quantifies which choices are important to align language models. We introduce our experimental setups in Section 3.1, and analyze different configurations in each stage of the training process in Section 3.2, Section 3.3, and Section 3.4, respectively. For a controlled study, we fix the model architecture to Gemma-2b (Gemma et al., 2024).

3.1 Experiment Setup

SFT Training Data To ensure the reproducibility, we carefully selected open-source, high-quality datasets for supervised fine-tuning. These include OpenHermes-2.5 (Teknium, 2023), SlimOrca (Lian et al., 2023; Longpre et al., 2023; Mukherjee et al., 2023), MetaMathQA (Yu et al., 2023), UltraChat (Ding et al., 2023), OrcaMath (Mitra et al., 2024), Capybara (Daniele and Suphavadeeprasit, 2023), and Deita-10k (Liu et al., 2024a). Since OpenHermes-2.5 is a collection of many other smaller open-source datasets, we also implemented a deduplication process to remove duplicate samples across these datasets. We use the latest version of those datasets to ensure that there is no contamination of data for our evaluation benchmarks. We summarize the dataset statistics in Table 1.

Offline Preference Learning Data For a controlled study, all models are trained on a fixed pool of pairwise preference dataset. We follow (Dong

Dataset	Samples (%)	Tokens (%)
OpenHermes-2.5	35.79%	28.77%
MetaMathQA	20.96%	10.14%
SlimOrca	19.74%	15.42%
UltraChat	11.29%	27.89%
OrcaMath	10.85%	7.60%
Capybara	0.86%	1.68%
Deita-10k	0.51%	8.51%
Total Count	1.84M	878M

Table 1: SFT dataset statistics

Dataset	Samples (%)	Judge
HH-RLHF	37.83%	Human
TLDR-Preference	27.85%	Human
UltraFeedback	23.04%	GPT-4
Distilabel-Orca	4.83%	GPT-4-Turbo
Py-DPO	3.57%	GPT-4-Turbo
Distilabel-Capybara	2.87%	GPT-4-Turbo
Total Count	264K	-

Table 2: Offline DPO dataset statistics

et al., 2024; Tunstall et al., 2023b) and manually select a mixture of publicly available datasets such as UltraFeedback (Cui et al., 2023), HH-RLHF (Bai et al., 2022), and TLDR-preferences (Stiennon et al., 2020a). These datasets consist of chat responses generated by a variety of LMs, and the winning/losing response is decided by prompting a judge model (e.g., GPT-4) or by asking human raters. We present the dataset statistics in Table 2. For more details on these datasets, please refer to Appendix A.1. Unless otherwise indicated, all dataset *subsets* mentioned in this section are randomly sampled from this 264K mixture.

Online Preference Learning Data We follow prior work (Meng et al., 2024; Xu et al., 2024b) and consider using data from UltraFeedback (Cui et al., 2023) as prompts. We then sample multiple responses from π_θ and use Pair-RM (Jiang et al., 2023b) as a judge to obtain preference pairs. This results in an online collected dataset of 60k in size. Unless otherwise indicated, all dataset *subsets* related to online learning are randomly sampled from this 60k dataset.

Evaluation Benchmarks We assess our models using OpenLLM (Beeching et al., 2023) and Arena-Hard-Auto (Li et al., 2024). The HuggingFace OpenLLM leaderboard evaluates an LM across a diverse set of reasoning, math, and knowledge tasks, and the average score is reported. Arena-Hard-Auto evaluates an LM's instruction-following ability using 500 challenging user queries curated from the live Chatbot Arena leaderboard (Zheng et al., 2023). To quantify the models' performance, it prompts a judge model (GPT-4-turbo) to compare the generated response against a reference response (by GPT-4), and uses the win rate as the final score. Since evaluation with GPT-4-turbo is expensive and using GPT-4's answers provides reference answers that are too strong, we use GPT-4-Omni[2] as the judge model and answers by GPT-3.5-turbo[3] as references for Sections 3.2 to 3.4. We denote this modification as *Arena-Hard-Auto**.

3.2 Supervised Fine-tuning

Supervised fine-tuning (SFT) plays a critical role in aligning Large Language Models (LLMs), often serving as the first step of alignment. However, different techniques, including sequence packing, padding, and loss masking, have been proposed for SFT (Chiang et al., 2023; Tunstall et al., 2023b; Shi et al., 2024). We re-examine the effectiveness of these strategies within the context of alignment.

Packing Packing optimizes the training efficiency by grouping sequences of varying lengths into a single long sequence without requiring any padding. This technique, commonly used in LLM pre-training, is now also utilized in instruction-based supervised fine-tuning, as implemented by models like Zephyr (Tunstall et al., 2023b)[4].

Padding In contrast to packing, padding extends shorter sequences with padding tokens and truncates longer ones to a fixed maximum length. It is often paired with loss masking, and is implemented in training models like Alpaca (Taori et al., 2023) and Vicuna (Chiang et al., 2023)[5].

Loss Masking The standard language model training computes loss across all tokens in a sequence. Loss masking, however, ignores loss computation on tokens that are not output tokens like user instructions. It prevents the model from learning irrelevant information, alleviating catastrophic forgetting and overfitting.

[2]We use version GPT-4o-2024-05-13

[3]We use version GPT-3.5-turbo-0125

[4]https://github.com/huggingface/alignment-handbook

[5]https://github.com/lm-sys/FastChat

Model	OpenLLM	Arena Hard Auto*
Gemma-2b	46.51	-
$\|D\| = 10K$		
Padding	42.49	3.1
+ Loss Mask	43.62	2.4
Packing	47.95	5.1
+ Loss Mask	**48.34**	**5.2**
$\|D\| = 1.6M$		
Packing	47.14	3.9
+ Loss Mask	**53.80**	**8.8**

Table 3: Performance comparison for different SFT strategies on OpenLLM and Arena-Hard-Auto*.

Model	OpenLLM	Arena Hard Auto*
gemma-2b-sft	54.67	8.8
+ default DPO	55.13	11.6
+ sqlen=2048	55.31	12.7
+ π_{ref}=SFT chosen	55.39	**13.1**
+ π_{ref}=DPO	**55.42**	12.5
+ π_{ref}=LLaMA-3-8b	55.31	12.7

Table 4: Effect of training with longer sequences and using different reference models. *sqlen* refers to maximum sequence length. $\pi_{ref} = x$ refers to DPO training with different reference models.

Loss masking can be used in conjunction with both packing and padding strategies. Packing without loss masking and padding with loss masking is widely adopted in SFT, but the combination of packing with loss masking is largely unexplored. We evaluate the performance of these strategies on both small and large datasets. For each dataset size $|D|$, we train all models from Gemma-2b-base over 3 epochs using a batch size of 32, a sequence length of 2,048 tokens, a learning rate of 2e-5. We then repeat this with $|D|$=10K with DEITA-10k (Liu et al., 2024b) and $|D|$=1.6M with Open-Hermes2.5 (Teknium, 2023), MetaMathQA (Yu et al., 2023), and UltraChat (Ding et al., 2023).

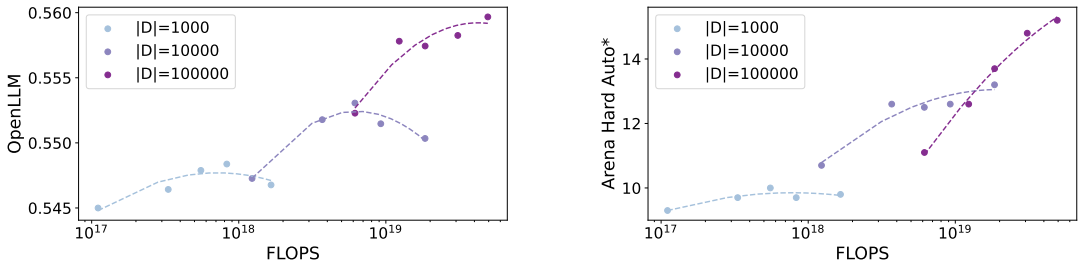
Table 3 summarizes our results. We find that combining packing with loss masking consistently yields the best performance across both dataset scales. We believe this is because other strategies may overfit chat templates: the starting tokens in each batch remain unchanged, leading to poor adaptation to unseen templates used in benchmarks such as OpenLLM. Next, we find increasing dataset size widens the performance gap between packing with and without loss masking. This may be due to the increasing number of user instructions as the dataset size grows, which is unnecessary for the model to learn. Overall, this indicates that π^{SFT} should be trained with packing and loss masking, over a large collection of high-quality datasets as in Table 1.

3.3 Offline Preference Learning

Following prior work, we use DPO (Rafailov et al., 2023) and continue training from the last iteration of SFT from Section 3.2. We selected DPO for our study because it is one of the most widely used algorithms for preference optimization (Tunstall et al., 2023b; Jiang et al., 2024; Yang et al., 2023; Yuan et al., 2024), despite the recent appearance of many alternatives (Meng et al., 2024; Gorbatovski et al., 2024; Ethayarajh et al., 2024). We then compare different training settings such as choosing sequence length/reference model; tuning beta; scaling offline alignment; and filtering preference datasets.

Choosing Sequence Length/Reference Model Popular implementations of DPO (Tunstall et al., 2023a,b) use a sequence length of 1024, and a reference model $\pi_{\text{ref}} = \pi^{\text{SFT}}$. However, many recent work differs in this setting: using either a longer sequence length (Meng et al., 2024), or a different reference model (Rafailov et al., 2023; Gorbatovski et al., 2024). We compare these configurations by training Gemma-2b on a *10k randomly sampled subset* from the mixture dataset and evaluating on OpenLLM and Arena-Hard-Auto*. Specially, we measure the impact of 1) using a longer sequence length of 2048, and 2) using different reference models proposed by prior work. The latter includes using π^{SFT} after further SFT on all chosen responses in the 10k subset (denoted as π_{ref}=*SFT chosen*); π^{SFT} after additional DPO training on the 10k subset (denoted as π_{ref}=*DPO*); and using a stronger model such as LLaMA-3-8b (denoted as π_{ref}=*LLaMA-3-8b*).

Table 4 shows that training with a longer sequence length of 2048 significantly improves performance on both benchmarks. We believe this is because multi-turn chat data are intrinsically long, and that longer responses may contain more complex reasoning compared to shorter answers (Zhao et al., 2024). We also find using different reference models such as π_{ref}=*SFT chosen* or π_{ref}=*DPO* slightly improves performance. However, as these methods require additional training, for simplicity we use $\pi_{\text{ref}} = \pi^{\text{SFT}}$ for the rest of the experiments.



(a) OpenLLM (b) Arena Hard Auto*

Figure 1: Measuring the effect of dataset size ($|D|$) and training steps (FLOPs) on final performance. While performance can quickly saturate given a fixed $|D|$, increasing the dataset size increases the point of saturation. Dotted lines are our interpolation using a degree 2 polynomial.

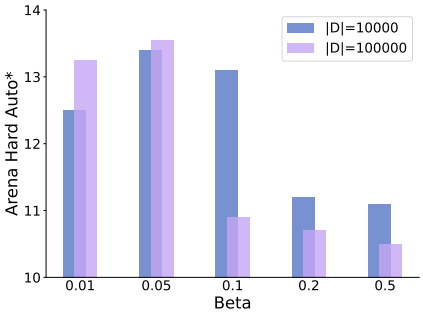


Figure 2: Varying KL-divergence strength (β) under different training data sizes. We find the best β stays relatively consistent across different dataset sizes.

Tuning Beta Beta β is a hyperparameter in DPO that controls the strength of KL-divergence. Besides sequence length and reference model, many prior work (Tunstall et al., 2023b; Gorbatovski et al., 2024) also differs in the choice of β. It is unclear whether β can critically affect performance, and how other factors such as training data size can interact with β.

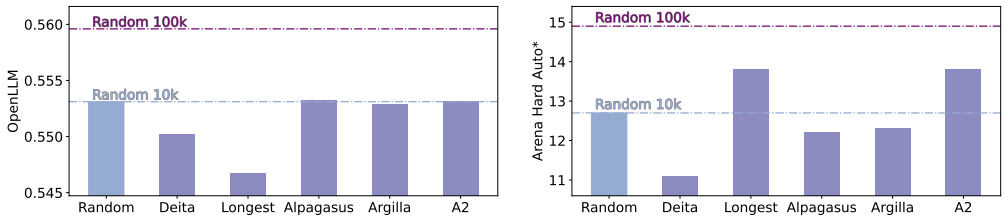
To investigate this, we first fix a dataset size $|D|$, and vary $\beta \in \{0.01, 0.05, 0.1, 0.2, 0.5\}$. We then repeat this process for different dataset sizes. In Figure 2, we find that 1) using a high KL-divergence β significantly harms performance, and 2) the best β stays relatively consistent across different training data sizes. This indicates that β can be tuned using only a small subset of the data[6], which is much more compute-efficient than sweeping using the full dataset.

Scaling Offline Alignment Prior work in SFT shows that scaling high-quality data during pretraining can significantly improve performance (Hoffmann et al., 2022; Kaplan et al., 2020). We investigate whether a similar scaling law exists in offline preference alignment. For a given dataset size $|D|$, we fix all training hyperparameters (e.g., $\beta = 0.1$ and a learning rate of 5e-7) and only vary the number of training steps. We then repeat this process for $|D| = \{1, 10, 100\} \times 10^3$, all randomly sampled from the dataset in Table 2. For other training details, please refer to Appendix A.2. We present the results in Figure 1.

In Figure 1, we find that 1) under a fixed dataset size, performance quickly saturates/over-optimizes as training step increases (Rafailov et al., 2024; Gao et al., 2022); and 2) increasing dataset size raises the point of saturation. We believe this indicates that similar to SFT, scaling law exists in offline preference learning so that scaling both dataset size and training steps can improve performance. We note that this finding contrasts many DPO training configurations in prior work, where either a 2-10 times smaller dataset is used (Tunstall et al., 2023b; Ivison et al., 2023), or 2-4 times fewer training steps are performed (Meng et al., 2024; Gorbatovski et al., 2024).

Filtering Preference Datasets Besides increasing training data, several prior work (Liu et al., 2024b; Zhou et al., 2023; Zhao et al., 2024) have also explored scaling *down* training data. These work finds that training with a small selection of "highest-quality" data can match or outperform training with the full dataset. To measure the effectiveness of these approaches, we considered training with a 10k data budget obtained from different data selection algorithms: DEITA (Liu et al., 2024b), LONGEST (Zhao et al., 2024), ALPAGASUS (Chen et al., 2024), and ARGILLA (Argilla, 2024). These methods select data based on their

[6]However, we note that using a subset too small (e.g., $|D| = 1000$) do not yield meaningful variations across runs.



(a) OpenLLM (b) Arena Hard Auto*

Figure 3: Effect of training on 10k data selected using different filtering algorithms. We find that simply training on a larger dataset (100k) outperforms all methods.

response length, response quality, prompt diversity, or a mixture of them[7]. We also consider A2, our simple heuristic that filters data based on a combination of score difference (Argilla, 2024; Wang et al., 2024) and high rating alike ALPAGASUS. On a high level, our method 1) remove preference pairs that have a score difference of less than 2.0, which is similar to Argilla (2024) that treats pairs with small score differences as noisy training data; and 2) bin preference pairs by their score differences, and uniformly sample preference pairs that has the highest chosen score from each bin, which is similar to Chen et al. (2024) that trains on the highest-quality sequences judged by GPT-3.5-turbo/GPT-4-turbo. For more implementation details on these algorithms, please refer to Appendix A.4.

Figure 3 summarizes the results. We find that 1) simply training on a 10 times larger dataset (100k) outperforms all data filtering methods; and 2) A2 is the only method that is competitive with random sampling in *both* benchmarks. We believe the former result strengthens the importance of data quantity and diversity in offline preference learning. The latter indicates that filtering methods based on attributes about the data itself may be insufficient for DPO, and that "better" data may be model dependent (Xia et al., 2024; Yu et al., 2024).

3.4 Online Preference Learning

Finetuning π_θ with preference data obtained online has proven highly effective in further enhancing model performance (Dong et al., 2024; Guo et al., 2024b). Given the high computational complexity of online training (Schulman et al., 2017a; Tran et al., 2023), we investigate whether it remains "essential" compared to the much more efficient offline alternative (Section 3.3).

[7]Some algorithms such as DEITA are originally designed for SFT datasets and uses a single prompt-response pair (x, y). In these cases, we use (x, y_w) from DPO datasets as (x, y).

Model	OpenLLM	Arena Hard Auto*
gemma-2b-sft	54.67	8.8
+ offline DPO (10k)	55.31	12.7
+ offline DPO (100k)	55.96	14.9
+ ODPO (1k)	55.32	12.7
+ ODPO (5k)	55.31	13.4
+ ODPO (10k)	55.32	14.6

Table 5: Effect of Online DPO (denoted as *ODPO*). We initialize all ODPO runs from the DPO (10k) checkpoint, and investigate the effect of different training data sizes.

We measure the effect of various online training data sizes on the final performance, and compare it against offline DPO. Specifically, we follow Meng et al. (2024) and first sample $n = 5$ responses with a temperature of 0.8 for each prompt from the UltraFeedback dataset. We then use Pair-RM[8] (Jiang et al., 2023b) as a judge and use the best and worst response as y_w and y_l, respectively. Similar to Meng et al. (2024), we perform one iteration of online training using DPO. We train all models from the DPO checkpoint trained with 10k randomly sampled data (denoted as *offline DPO (10k)*).

In Table 5, we first find that online training mainly benefits chat benchmarks (Arena Hard Auto*) but not core capability/knowledgege benchmarks (OpenLLM). We believe this is because online preference pairs are derived from π_θ itself, making it unlikely for π_θ to acquire *new* knowledge or skills. Next, we find that increasing the number of online training samples to 10k reaches comparable performance to offline DPO with 100k data. This indicates that online training remains competitive, and can be much more sample efficient than offline training for chat benchmarks.

[8]We note that results may vary with different reward models (Lambert et al., 2024), which we leave for future work.

4 The LION Series

In the previous section, we empirically analyzed the best strategies to perform supervised finetuning, offline preference learning, and online preference learning. We aggregate these findings to train a series of models, the LION series, and evaluate them on numerous LLM benchmarks.

4.1 Experiment Setup

Training Recipe We aggregate our findings from Section 3 into a single training recipe. During the SFT stage, we use the packing with loss masking strategy. During the DPO stage, we 1) use a sequence length of 2048 and $\pi_{\text{ref}} = \pi^{\text{SFT}}$, 2) sweep for the optimal β using a small (10k) subset, and 3) train our models over a large dataset with a compute budget equivalent to the best model in Figure 1. For online DPO, we follow (Meng et al., 2024) and use Pair-RM (Jiang et al., 2023b) as a judge. We perform one iteration of online DPO and train on the full 60k online preference data.

Training Datasets We use the same datasets from Section 3.2 for SFT. Given the scaling trends for offline preference learning (Section 3.3), we additionally add Nectar (Banghua et al., 2023), HelpSteer (Wang et al., 2023), and PKU-SafeRLHF (Dai et al., 2024). For online learning, we follow (Meng et al., 2024) and use prompts from UltraFeedback. We summarize the datasets used in Appendix B.3.

Evaluation Benchmarks To holistically evaluate our models performance, we follow prior work and consider in total four benchmarks: Arena-Hard-Auto (Li et al., 2024), AlpacaEval-2 (Dubois et al., 2024; Li et al., 2023b), MT-Bench (Zheng et al., 2023), and OpenLLM (Beeching et al., 2023). In addition to the evaluation methods used in Section 3, AlpacaEval-2 uses a length-controlled (LC) metric to evaluate the model's instruction-following ability; and MT-Bench uses GPT-4 as a judge to score the model's response on a diverse set of QA tasks. We use the standard evaluation setting for all benchmarks, such as using GPT-4-turbo as the judge model and GPT-4 as the reference for Arena-Hard-Auto (c.f. Section 3.1).

4.2 Models and Baselines

Models We train all of our models from Gemma-2b-base (Gemma et al., 2024) and LLaMa-3-8B-base (AI@Meta, 2024). These models are pre-trained with trillions of tokens from the web, and highly performant for a wide range of text generation tasks (Beeching et al., 2023). We denote our models trained after each phase as *-lion-sft*, *-lion-dpo*, and *-lion-odpo*, representing the SFT, offline DPO, and online DPO stages, respectively.

Baselines We mainly compare our method against the officially released instruct models, Gemma-2b-it and Llama-3-8b-it. From the base model, Gemma-2b-it is first trained using SFT, and further finetuned using a novel, close-sourced RLHF algorithm (Gemma et al., 2024). LLaMA-3-8b-it is trained using a combination of SFT, rejection sampling, PPO, and DPO (AI@Meta, 2024). Both models are trained using close-sourced data.

4.3 Main Results

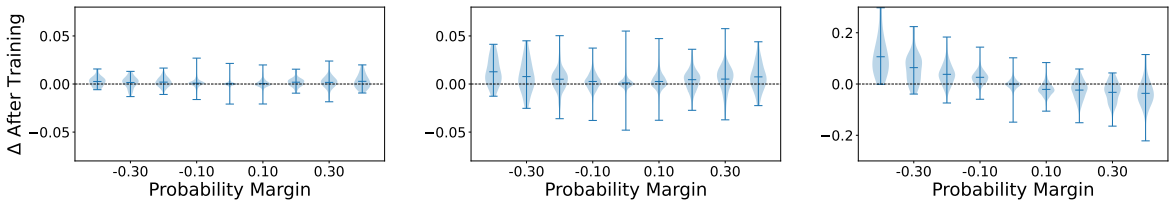
We present the main results in Table 6. We find that after the SFT and offline DPO training phase, our Gemma-2b model already outperforms the Gemma-2b-it on all benchmarks. It also matches or surpasses various popular 7b models, such as LLaMA-2-7b-chat (Touvron et al., 2023) and Vicuna-7b (Chiang et al., 2023). Similarly, our LLaMA-3-8b-lion-dpo shows competitive performance against the LLaMA-3-8b-it, despite only being trained with SFT and DPO. Finally, after online DPO training, our models further improve, surpassing the officially released instruct models in all benchmarks. We believe this result indicates the effectiveness of our training recipe throughout the three stages of alignment training.

4.4 Qualitative Analysis

To provide insights into the opaque training process during preference learning, we additionally measure how the sequence probability for y_w and y_l change after DPO training. We use a fixed unseen test set of (x, y_w, y_l) obtained from public datasets (see Appendix A.1), and compare $\pi_\theta(y_w) - \pi_\theta(y_l)$ for each preference pair before and after training. We find that many of the best-performing models have a parabolic shape as shown in Figure 4(b). This shows that well-trained models learns to improve confidence not only in pairs they could already distinguish correctly before training (i.e., $\pi_\theta(y_w) > \pi_\theta(y_l)$) but also equally in pairs they previously could not (i.e., $\pi_\theta(y_w) < \pi_\theta(y_l)$). Please refer to Appendix B.5 for more details.

Model	Method	Size	Arena-Hard	AlpacaEval-2	MT-bench	OpenLLM
Gemma-2b	-	2B	-	-	-	46.69
Gemma-2b-it	SFT+RLHF	2B	3.4	5.44	5.63	42.75
Gemma-2b-zephyr	SFT+DPO	2B	0.9	2.65	4.13	46.92
LLaMA-2-7b-chat	SFT	7B	4.6	5.35	6.22	53.16
Vicuna-7b-v1.5	SFT	7B	2.5	7.62	6.57	52.06
Gemma-2b-lion-sft (ours)	SFT	2B	2.4	7.79	6.37	54.78
Gemma-2b-lion-dpo (ours)	SFT+DPO	2B	4.6	8.75	6.58	55.35
Gemma-2b-lion-odpo (ours)	SFT+DPO+ODPO	2B	**5.0**	**9.57**	**6.75**	**55.98**
LLaMA-3-8b	-	8B	-	-	-	63.05
LLaMA-3-8b-it	SFT+RS+DPO+PPO	8B	20.6	22.9	8.00	68.28
LLaMA-3-8b-lion-sft (ours)	SFT	8B	11.3	17.9	7.58	68.71
LLaMA-3-8b-lion-dpo (ours)	SFT+DPO	8B	19.1	21.8	8.12	71.28
LLaMA-3-8b-lion-odpo (ours)	SFT+DPO+ODPO	8B	**22.0**	**26.8**	**8.19**	**71.41**
LLaMA-3-70B-it	SFT+RS+DPO+PPO	70B	41.1	34.4	8.95	73.96
GPT-3.5-turbo-0125	-	-	24.8	22.7	8.39	-
GPT-4 Turbo	-	-	82.6	55.0	9.32	-

Table 6: Evaluating the LION series across multiple chat and core knowledge benchmarks. We report the win rate and length-controlled win rate for Arena-Hard-Auto and AlpacaEval-2, respectively.



(a) undertrained (b) best performing (c) overtrained

Figure 4: We track the changes in the probability margin $\pi_\theta(y_w) - \pi_\theta(y_l)$ under various training configurations, and find that the best-performing models exhibit a parabolic pattern. Arena-Hard-Auto* results from left to right are 10.7, 14.8, and 13.2. Test loss from left to right is 0.63, 0.65, and 1.33.

5 Related Work

Many recent studies extensively explored alignment methods to improve LLMs' ability to follow human instructions. Early approaches train LLMs with supervised fine-tuning (SFT) using high-quality human-written demonstrations (Sanh et al., 2021; Wei et al., 2022; Chung et al., 2022; Mishra et al., 2021). This method enjoys various properties such as fast convergence and scaling laws in training data/model sizes (Kaplan et al., 2020; Hoffmann et al., 2022). However, SFT is vulnerable to exposure bias, and can generate outputs that do not align well with human intent. To this end, reinforcement learning from human feedback (Schulman et al., 2017b; Stiennon et al., 2020b; Ouyang et al., 2022b) was proposed. These work typically employs an online RL algorithm (such as PPO, Schulman et al. (2017a)) to optimize LLMs towards a reward model mimicking human preferences. Although effective, PPO can be complex to implement and often suffers from high reward variance, making it challenging to maintain stable performance (Xu et al., 2024c).

To address the limitations of PPO, many recent work focused on offline preference learning algorithms, such as Direct Preference Optimization (Rafailov et al., 2023), KTO (Ethayarajh et al., 2024), CTO (Xu et al., 2024a), and more (Guo et al., 2024a; Hong et al., 2024). These algorithms are much easier to use, and spurred many recent studies to explore: 1) collecting high-quality offline preference data (Cui et al., 2023; Dai et al., 2024; Banghua et al., 2023), and 2) designing better training pipelines such as iterative DPO and online DPO (Guo et al., 2024b; Xu et al., 2024b; Tran et al., 2023; Xu et al., 2024c). Although these contributions demonstrate improvements, the training processes, datasets, and hyper-parameters often remain heterogeneous, and it is difficult to understand the source of improvements. Our goal is to provide a comprehensive analysis of the alignment

pipeline starting from SFT to online preference learning, and to serve as a reference point for better understanding in the alignment process.

6 Conclusion

We present a detailed analysis of modern alignment pipelines, and present a step-by-step recipe to finetune models using only publicly available datasets and open-sourced algorithms. We find that model performance substantially improves by 1) using packing and loss making in SFT, 2) scaling offline preference datasets and training steps in DPO, and 3) training with online preference data to improve chat performance. We then aggregate our findings and train the LION series, and show that they outperform the officially released instruct models as well as models of larger sizes on benchmarks such as Arena-Hard-Auto, AlpacaEval-2, MT-bench, and OpenLLM. These results illustrate the importance of many previously overlooked design choices, and serve as a reference point for future work in alignment research.

7 Limitations

7.1 Sensitivity of Model Backbone

In our analysis, we investigated the effect of various training strategies during each stage *independently*, and aggregate a training recipe using the best results from each stage. However, it is possible that there are combined effects between two or more stages (e.g., modify SFT and offline DPO simultaneously), which could lead to different or better results. Since this would result in an exponentially larger search space for training strategies, we chose to conduct our experiments in a sequential manner. We leave this exploration for future work.

7.2 Sensitivity of Training Data

Unlike SFT, in our prior experiments we find that offline preference datasets can vary significantly in quality and quantity. We therefore manually selected a mixture of high-quality datasets for our experiments in Section 3.3 based on some empirical heuristics (Appendix A.1). Since this choice is empirical, we believe results may vary when, in the future, datasets of higher quality and larger sizes become available. We believe creating new, higher-quality datasets is perpendicular to our work, and we leave this for future work.

7.3 More Model Architectures

Our analysis primarily focuses on the Gemma-2b-base model. This is because 1) Gemma-2b is a light-weight yet performant model used widely in the community, and 2) it requires significantly less compute to conduct analysis as in Section 3 compared to using larger models such as LLaMA-3-8B-base. However, we believe it would be beneficial to extend our analysis to other model architectures such as LLaMA-3-8b (AI@Meta, 2024) and Mistral-7b (Jiang et al., 2023a). We plan to extend our experiments with models of different sizes and architectures in future work.

8 Ethical Considerations

In this work, we focus on the reproduction study and step-by-step recipe to align large language models. Our model was trained on publicly available alignment datasets. Despite our efforts to carefully examine and curate our training data sources, there is a possibility that malicious or harmful content may still be present. To mitigate these risks, we acknowledge the necessity of incorporating more datasets specifically focused on safety, harmfulness, and bias. Furthermore, for future work, we commit to conducting more comprehensive evaluations on safety, harmfulness, and bias to enhance the robustness and ethical standards of language model alignment.

References

AI@Meta. 2024. Llama 3 model card.

Team Argilla. 2024. Argilla: Dpo-mix-7k.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Zhu Banghua, Frick Evan, Wu Tianhao, Zhu Hanlin, and Jiao Jiantao. 2023. Starling-7b: Improving llm helpfulness and harmlessness with rlaif.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar

Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. Alpagasus: Training a better alpaca with fewer data. *Preprint*, arXiv:2307.08701.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *Preprint*, arXiv:2310.01377.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2024. Safe rlhf: Safe reinforcement learning from human feedback. In *The Twelfth International Conference on Learning Representations*.

Luigi Daniele and Suphavadeeprasit. 2023. Amplify-instruct: Synthetically generated diverse multi-turn conversations for efficient llm training. *arXiv preprint arXiv:(coming soon)*.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Ameet Deshpande, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, and Karthik Narasimhan. 2023. Toxicity in chatgpt: Analyzing persona-assigned language models. *Preprint*, arXiv:2304.05335.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *Preprint*, arXiv:2405.07863.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *Preprint*, arXiv:2402.01306.

Leo Gao, John Schulman, and Jacob Hilton. 2022. Scaling laws for reward model overoptimization. *Preprint*, arXiv:2210.10760.

Team Gemma, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas,

Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Alexey Gorbatovski, Boris Shaposhnikov, Alexey Malakhov, Nikita Surnachev, Yaroslav Aksenov, Ian Maksimov, Nikita Balagansky, and Daniil Gavrilov. 2024. Learn your reference model for real good alignment. *Preprint*, arXiv:2404.09656.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. 2024a. Direct language model alignment from online ai feedback. *Preprint*, arXiv:2402.04792.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. 2024b. Direct language model alignment from online ai feedback. *Preprint*, arXiv:2402.04792.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *Preprint*, arXiv:2403.07691.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *Preprint*, arXiv:2311.10702.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023b. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *Preprint*, arXiv:2306.02561.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. *Preprint*, arXiv:2403.13787.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023a. Camel: Communicative agents for "mind" exploration of large scale language model society. *Preprint*, arXiv:2303.17760.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2024. From live data to high-quality benchmarks: The arena-hard pipeline.

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

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Slimorca: An open dataset of gpt-4 augmented flan reasoning traces, with verification.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024a. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024b. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *Preprint*, arXiv:2312.15685.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V.

Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *Preprint*, arXiv:2301.13688.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023. Wizardcoder: Empowering code large language models with evol-instruct.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Preprint*, arXiv:2405.14734.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. In *Annual Meeting of the Association for Computational Linguistics*.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *Preprint*, arXiv:2402.14830.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *Preprint*, arXiv:2306.02707.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, and et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022a. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. 2022b. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, Bradley Knox, Chelsea Finn, and Scott Niekum. 2024. Scaling laws for reward model overoptimization in direct alignment algorithms. *Preprint*, arXiv:2406.02900.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan D. Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng-Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017a. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017b. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

Zhengyan Shi, Adam X. Yang, Bin Wu, Laurence Aitchison, Emine Yilmaz, and Aldo Lipani. 2024. Instruction tuning with loss over instructions. *Preprint*, arXiv:2405.14394.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020a. Learning to summarize from human feedback. In *NeurIPS*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan J. Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020b. Learning to summarize from human feedback. *ArXiv*, abs/2009.01325.

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

Teknium. 2023. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay

Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Hoang Tran, Chris Glaze, and Braden Hancock. 2023. Iterative dpo alignment.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. 2023a. The alignment handbook. `https://github.com/huggingface/alignment-handbook`.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023b. Zephyr: Direct distillation of lm alignment. *Preprint*, arXiv:2310.16944.

Binghai Wang, Rui Zheng, Lu Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, Songyang Gao, Nuo Xu, Yuhao Zhou, Xiaoran Fan, Zhiheng Xi, Jun Zhao, Xiao Wang, Tao Ji, Hang Yan, Lixing Shen, Zhan Chen, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. 2024. Secrets of rlhf in large language models part ii: Reward modeling. *Preprint*, arXiv:2401.06080.

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Polak Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. 2023. Helpsteer: Multi-attribute helpfulness dataset for steerlm. *Preprint*, arXiv:2311.09528.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023. Jailbroken: How does llm safety training fail? *Preprint*, arXiv:2307.02483.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *Preprint*, arXiv:2402.04333.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *ArXiv*, abs/2304.12244.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024a. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *Preprint*, arXiv:2401.08417.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. 2024b. Some things are more cringe than others: Iterative preference optimization with the pairwise cringe loss. *Preprint*, arXiv:2312.16682.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weiling Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. 2024c. Is dpo superior to ppo for llm alignment? a comprehensive study. *ArXiv*, abs/2404.10719.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, Fei Deng, Feng Wang, Feng Liu, Guangwei Ai, Guosheng Dong, Haizhou Zhao, Hang Xu, Haoze Sun, Hongda Zhang, Hui Liu, Jiaming Ji, Jian Xie, JunTao Dai, Kun Fang, Lei Su, Liang Song, Lifeng Liu, Liyun Ru, Luyao Ma, Mang Wang, Mickel Liu, MingAn Lin, Nuolan Nie, Peidong Guo, Ruiyang Sun, Tao Zhang, Tianpeng Li, Tianyu Li, Wei Cheng, Weipeng Chen, Xiangrong Zeng, Xiaochuan Wang, Xiaoxi Chen, Xin Men, Xin Yu, Xuehai Pan, Yanjun Shen, Yiding Wang, Yiyu Li, Youxin Jiang, Yuchen Gao, Yupeng Zhang, Zenan Zhou, and Zhiying Wu. 2023. Baichuan 2: Open large-scale language models. *Preprint*, arXiv:2309.10305.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Xiao Yu, Baolin Peng, Michel Galley, Jianfeng Gao, and Zhou Yu. 2024. Teaching language models to self-improve through interactive demonstrations. *Preprint*, arXiv:2310.13522.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *Preprint*, arXiv:2401.10020.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. *Preprint*, arXiv:2402.04833.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

A More Details on Alignment Procedure Analysis

A.1 Offline Preference Dataset Curation

Gathering a high-quality dataset of sufficient scale is imperative to study various properties of current offline preference learning algorithms. There are many open-source preference labeled datasets available online, including Ultrafeedback (Cui et al., 2023), TLDR-Preferences (Stiennon et al., 2020a), and Nectar (Banghua et al., 2023). However, they show significant differences in 1) the quality and diversity of the prompts, 2) models used to generate the responses, 3) judge models, and 4) the number of preference pairs. It is therefore unclear which dataset is of sufficient quality to train a model on, and how the model's performance changes on downstream benchmarks.

To this end, we first selected a collection of 12 datasets, and empirically measure each dataset's quality by 1) sample upto 10k samples from each dataset, 2) train SFT-finetuned Gemma-2b and record its performance on MT-bench. We present the results in Figure A2 and Figure A3. We then used the top-six datasets according to their overall score in our offline preference learning experiments in Section 3.3. The baseline is our π^{SFT} model.

A.2 Training Hyperparams for Scaling DPO

For all runs, we finetune from the best π^{SFT} obtained from Section 3.2. We use a sequence length of 2048, $\beta = 0.1$, batch size of 128, learning rate of 5e-7, and vary training steps for each run. For $|D|$=100k, we continue training from previous runs instead of starting from scratch to save compute.

In addition to our result in Figure 1 measuring performance in OpenLLM and Arena-Hard-Auto*, we also present other metrics such as evaluation loss, reward margin, and reward accuracy in Table A1. We note that the evaluation loss and reward margin are *inconsistent* with the performance in Arena-Hard-Auto* (or OpenLLM). This indicates that simple metrics such as evaluation loss and reward margin may not be good indicators of model final performance.

A.3 Reward Annotation for Data Filtering

While datasets such as UltraFeedback (Cui et al., 2023) provide ratings in [1,10] for chosen/rejected responses, other datasets such as TLDR (Stiennon et al., 2020a) and HH-RLHF (Bai et al., 2022) does not. This makes methods such as filtering based on score difference (e.g., ARGILLA) not applicable.

To this end, we consider a simple approach to use Nexusflow/Starling-RM-34B (Banghua et al., 2023), the best reward model according reward-bench (Lambert et al., 2024), to provide a score prediction to all of our training data. Specifically, we first used the reward model to compute a real value score for the prompt + chosen response and prompt + rejected response separately. Next, since the predicted score is a real value, we 1) rescale it to $[0, 1]$ to obtain $\tilde{s}$, and then 2) consider a least square solution to find a, b under the function:

$$\hat{s} = \text{clip}(a \cdot \tilde{s} + b, \text{min} = 0, \text{max} = 10)$$

where the least square error between the true score s for data that contains a GPT-4 annotated score and the rescaled $\hat{s}$ is minimized. This results in $a = 11.1745, b = 1.1791$. The average least square error and absolute error is 3.5389 and 1.4278, respectively. Finally, we augment the entire 264K training data with this score $\hat{s}$, and present the score distribution in Figure A6.

Note that this reward model achieves 67.72% accuracy over the entire 264K dataset (both with and without our score transformation). This indicates that ∼30% of the predicted score might not be accurate. Therefore, we use the original score annotation when available, and use the predicted and rescaled score only when necessary.

A.4 More Details on Dataset Filtering Algorithms

We consider data filtering algorithms both from the instruction-tuning domain and from the preference learning domain. This include algorithms such as DEITA (Liu et al., 2024b), LONGEST (Zhao et al., 2024), ALPAGASUS (Chen et al., 2024), and ARGILLA (Argilla, 2024). We also consider A2, which can be seen as a combination of ARGILLA and ALPAGASUS: first removing pairs that has a score difference of less than two, and then sampling 10k data that has the highest chosen score in each bin. We apply each of the algorithms above to select 10k data from the 264K shown in Table 2. We present the selected data distributions for each algorithm (except for A2) in Figure A7, and for A2 in Figure A5.

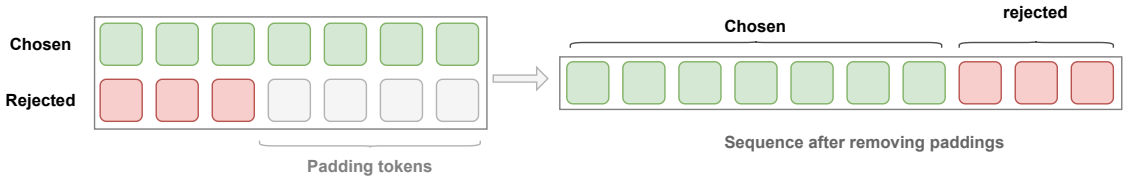


Figure A1: Illustration of efficient DPO implementation. Traditional DPO training requires adding padding tokens to the batch. Our implementation can remove the need of paddding tokens, and thus improving the training efficiency.

FLOPs	$\|D\|$	Arena-Hard-Auto*	Eval Loss	Eval Reward Margin	Eval Reward Accurarcy
3.1e19	100k	**14.8**	0.6507	0.5375	0.6554
1.8e19	10k	13.2	1.3300	**0.7851**	0.5738
6.2e18	100k	11.1	0.6389	0.3970	0.6334
1.2e18	10k	10.7	**0.6333**	0.2082	**0.6922**
1.1e17	1k	9.3	0.6940	-0.0039	0.4858

Table A1: Automatic evaluation metrics such as loss, reward margin, and reward accuracies on test set is inconsistent with final performance on benchmarks such as Arena-Hard-Auto*. All runs used $\beta = 0.1$.

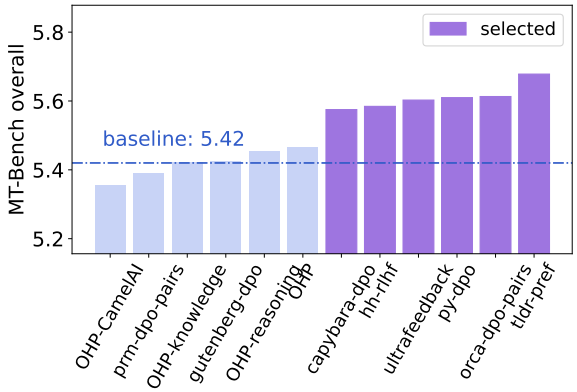


Figure A2: MT-bench score after training Gemma-2b on upto 10k samples from each dataset. Datasets we used in Section 3 is colored in purple. "OHP" stands for Openhermes-Preference.

Configuration	Training Time
LLaMA3-8b DPO	15.72 hours
+ Fast Implementation (ours)	11.40 hours
Improvement	27.48%

Table A2: DPO Training times for different configurations.

B More Details on Training LION Series

B.1 Efficient DPO Implementation

In this work, we introduced an efficient DPO implementation for Transformers. The motivation is to eliminate the computation overhead caused by padding tokens, as in DPO, chosen and rejected samples normally have varied lengths. Our approach involves removing all padding tokens within a batch and concatenating the remaining sequences into a single, continuous sequence. To handle sequence boundaries effectively in the self-attention layers, we utilize FlashAttention (Dao, 2024). This ensures that the removed padding tokens do not interfere with the processing of the valid tokens. An illustration of this process can be found in Figure A1.

We evaluated the training times for LLAMA3-DPO with and without the fast DPO model implementation. The experiments were conducted using the specified offline preference dataset, running on a setup of four A100 80GB GPUs. As shown in Table A2, our fast DPO model implementation achieves a 27.48% speed improvement.

B.2 Training Details

All the training experiments in this paper were conducted on 4×A100 80GB GPUs. We used Deepspeed (Rasley et al., 2020) for all our experiments as we find that storingin model weights in fp32 is essential for DPO's performance as learning rate is small. For other training details, please see Table A3 and Table A4.

B.3 Training Datasets

Following our findings in Section 3, we train the LION series using a combination of datasets from the instruction-tuning domain and the preference learning domain. For SFT, we use the same dataset collection as in Table 1. Given the scaling trends of offline DPO, we add in more preference datasets

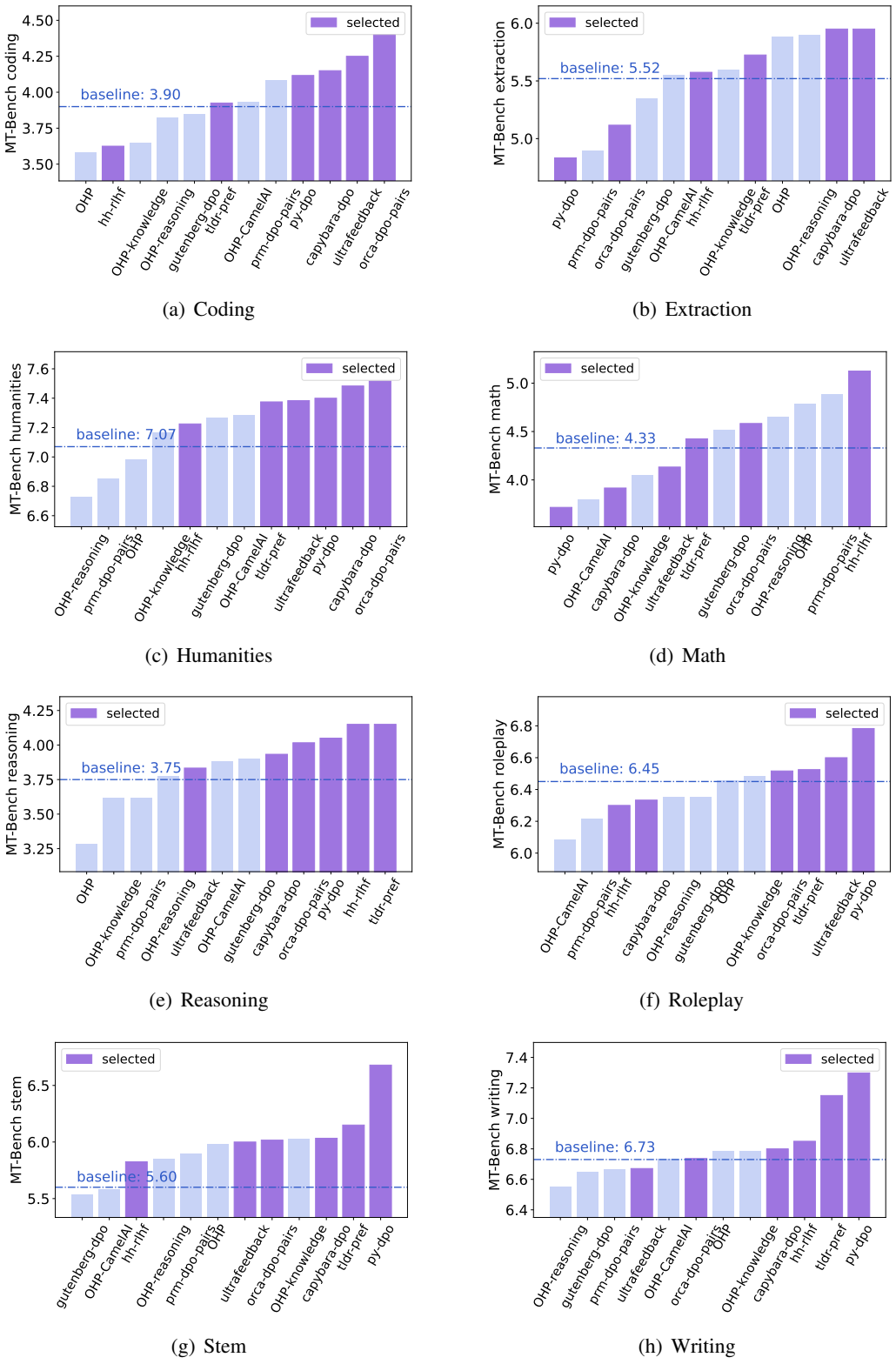
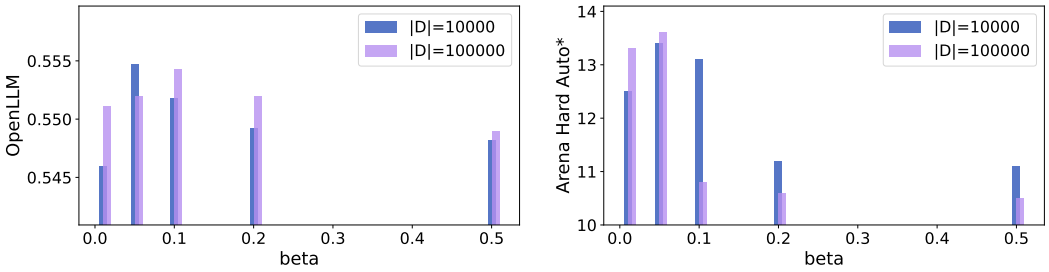


Figure A3: MT-bench performance after training Gemma-2b on upto 10k samples from each dataset. Datasets we used in Section 3.3 are colored in purple.

such as Nectar (Banghua et al., 2023) in addition to Table 2. We summarize the datasets for training the LION below:

Details of Supervised Fine-Tuning Data

- **OpenHermes-2.5** (Teknium, 2023): The OpenHermes-2.5 dataset contains 1 million di-



(a) OpenLLM Performance (b) Arena Hard Auto* Performance

Figure A4: Effect of β on model performance across datasets of different sizes.

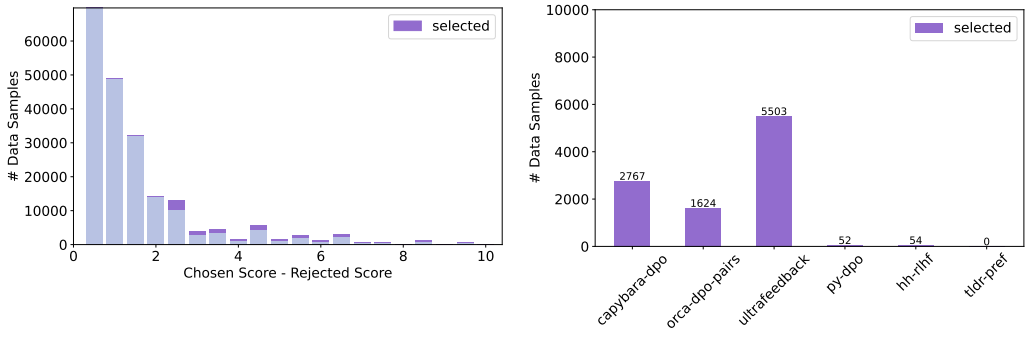
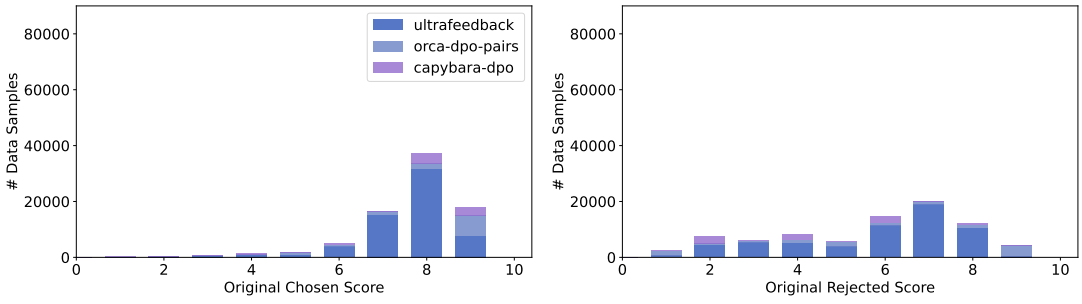
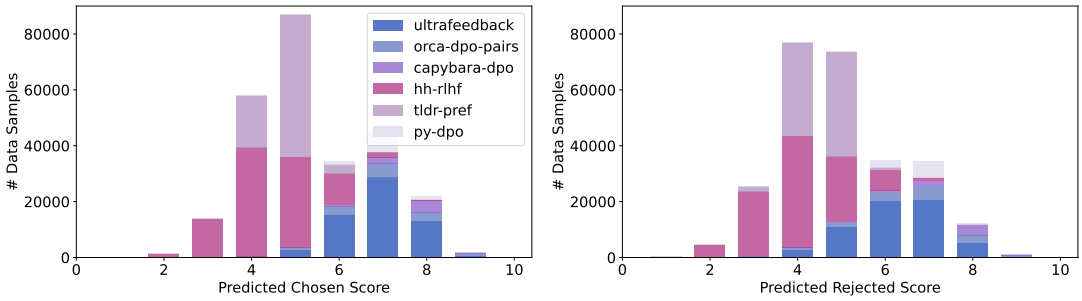


Figure A5: Data distribution for A2, which 1) removes DPO pairs with score difference less than two, and 2) sample 10k data that has the highest chosen score in each score difference bin.



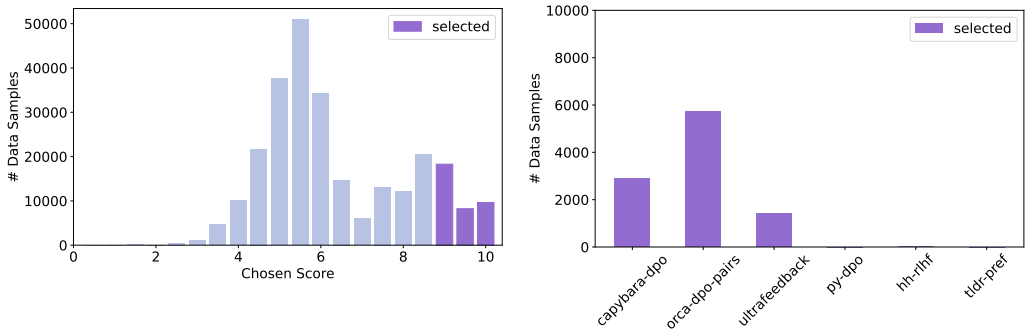
(a) Original



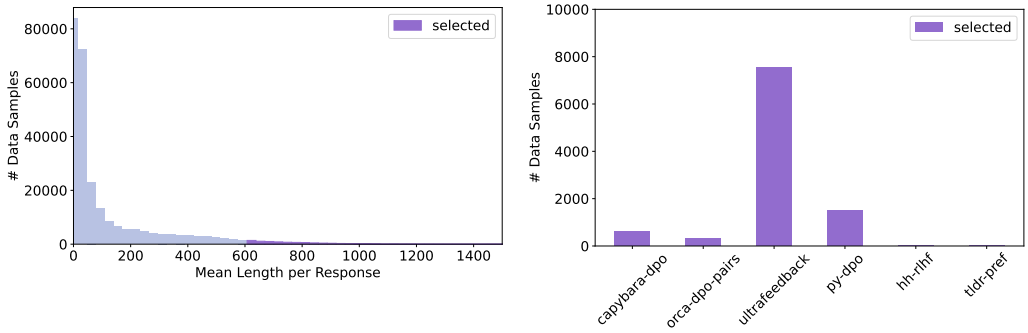
(b) Predicted

Figure A6: Score distribution for the dataset we used in Section 3.3. (a) Responses from Ultrafeedback, Orca-DPO-pairs, and Capybara-DPO already contain scores annotated by GPT-4/GPT-4-turbo. (b) We rescaled the score prediction produced by Nexusflow/Starling-RM-34B.

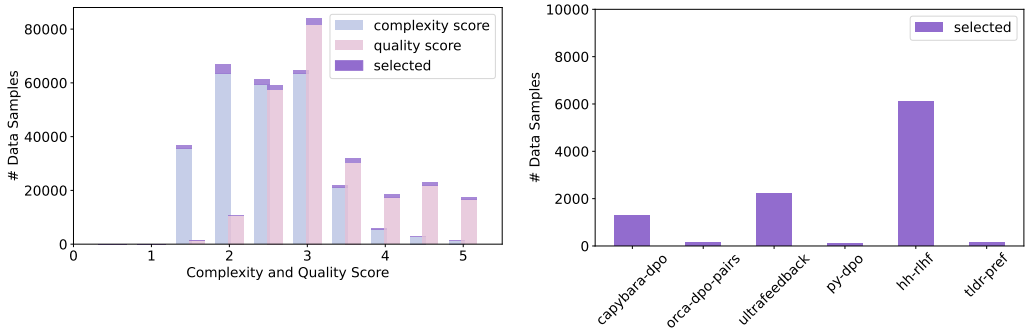
verse, synthetic samples. It includes data from various sources like Airoboros, CamelAI (Li



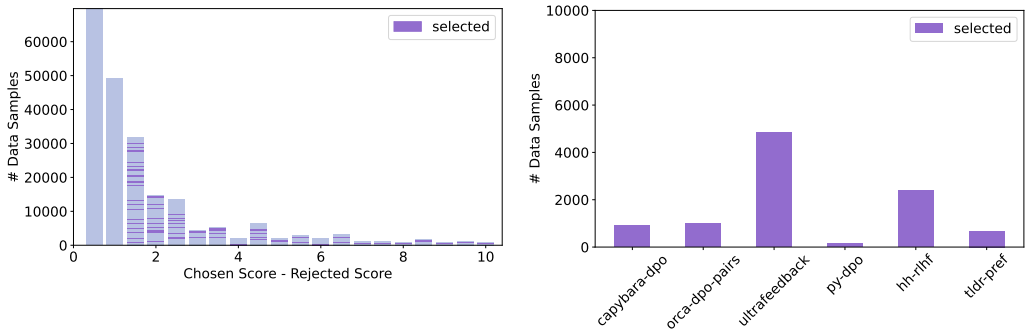
(a) Alpagasus-10k



(b) Longest-10k



(c) DEITA-10k



(d) Argilla-10k

Figure A7: Data distribution after applying the respective filtering algorithms.

et al., 2023a), ChatBot Arena, and several others, each contributing to fields ranging from physics and mathematics to code assistance and medical tasks. Please check the repo for details.

- **MetaMathQA** (Yu et al., 2023): MetaMathQA is created using question bootstrap-

Hyperparam	Gemma-2b	LLaMA-3-8b
Warmup ratio	0.1	0.1
Peak Learning Rate	2e-5	2e-5
Max Sequence Length	8192	8192
Batch Size	16	64
Weight Decay	0.0	0.0
Number Epochs	3	3
Learning Rate Decay	Cosine	Cosine
Max Grad Norm	1.0	1.0
Training Time	25.80 hrs	56.36 hrs

Table A3: SFT Training Details. Training time is measured on 4 GPUs configuration.

Hyperparam	Gemma-2b	LLaMA-3-8b
Beta	0.05	0.01
Warmup ratio	0.1	0.1
Max Sequence Length	2048	2048
Peak Learning Rate	5e-7	5e-7
Batch Size	64	128
Weight Decay	0.0	0.0
Number Epochs	2	1
Learning Rate Decay	Cosine	Cosine
Max Grad Norm	1.0	1.0
Training Time	10.15 hrs	11.40 hrs

Table A4: DPO Training Details. Training time is measured on 4 GPUs configuration.

ping, where mathematical questions are rewritten from GSM (Cobbe et al., 2021) and Math (Hendrycks et al., 2021) dataset. The dataset is further enriched by rephrasing questions and using rejection sampling to select only correctly answered paths, enhancing diversity and reasoning capabilities.

- **SlimOrca** (Lian et al., 2023): The SlimOrca dataset is a curated subset of the OpenOrca (Mukherjee et al., 2023) data, containing about 500,000 GPT-4 completions refined using human annotations from the FLAN (Longpre et al., 2023) dataset to remove incorrect answers.

- **UltraChat** (Ding et al., 2023): UltraChat is large-scale, informative, and diverse multi-round dialogue dataset aimed at improving language model conversational skills. It contains 1.5M samples with a wide range of topics and instructions.

- **OrcaMath** (Mitra et al., 2024): OrcaMath comprises 200,000 synthetic mathematical problems created using a collaborative multi-agent setup with GPT-4.

- **Capybara** (Daniele and Suphavadeeprasit, 2023): Capybara uses the Amplify-Instruct method to create synthetic multi-turn conversations from quality single-turn seeds. It focuses on diverse, logical reasoning across domains, with each conversation exploring deep, diverse topics.

- **Deita-10k** (Liu et al., 2024a): Deita is an open-source dataset aimed at enhancing instruction tuning for Large Language Models (LLMs) through Automatic Data Selection. It incorporates a dataset of 10,000 high-quality, alignment-specific Supervised Fine-Tuning (SFT) data points. This data is primarily selected from larger datasets including 58K entries from ShareGPT (Chiang et al., 2023), 105K from UltraChat (Ding et al., 2023), and a 143K mixture from WizardLM (Xu et al., 2023) data (Luo et al., 2023).

Details of Offline Preference Data

- **TLDR** (Stiennon et al., 2020a): This data is used to train a reward model for summarization. Summaries for the reward model came from the TL;DR dataset. We use the comparisons data, where annotators chose the better of two summaries.

- **PKU-SafeRLHF** (Dai et al., 2024): This dataset contains 83.4K preference entries annotated for harmlessness and helpfulness. Each entry includes two responses to a question, with safety meta-labels and preferences. The responses came from Alpaca-7B, Alpaca2-7B, and Alpaca3-8B models, following SFT performed on Llama2-7B and Llama3-8B with the Alpaca 52K dataset.

- **HelpSteer** (Wang et al., 2023): It is open-source Helpfulness Dataset designed by NVIDIA to improve models' helpfulness, factual accuracy, and coherence, with adjustable response complexity and verbosity. It contains 37,120 samples, each including a prompt, a response, and five human-annotated attributes of the response, rated from 0 to 4: Helpfulness (overall helpfulness), Correctness (pertinence and accuracy of facts), Coherence (consistency and clarity), Complexity (intellectual depth), and Verbosity (amount of detail).

Model	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8k	Average
Gemma-2b	48.38	71.77	41.77	33.08	**66.30**	16.91	46.51
Gemma-2b-it	43.60	62.55	36.95	**45.85**	61.80	10.99	43.62
Gemma-2b-lion-sft (ours)	50.94	70.65	45.04	43.80	64.88	53.37	54.78
Gemma-2b-lion-dpo (ours)	52.30	72.47	45.31	45.06	65.19	51.78	55.35
Gemma-2b-lion-odpo (ours)	**53.75**	**73.04**	**45.52**	45.66	64.40	**53.53**	**55.98**
LLaMA-3-8b-Base	58.02	82.15	65.09	43.92	**77.58**	51.55	63.05
LLaMA-3-8b-Instruct	61.86	78.79	**65.70**	51.64	75.30	75.13	68.07
LLaMA-3-8b-lion-sft (ours)	59.64	80.80	64.21	54.26	76.64	76.72	68.71
LLaMA-3-8b-lion-dpo (ours)	63.91	82.95	63.67	60.01	76.56	**80.59**	71.28
LLaMA-3-8b-lion-odpo (ours)	**63.99**	**83.18**	63.59	**61.12**	76.72	79.91	**71.41**

Table A5: Detailed task evaluation results on OpenLLM.

- **UltraFeedback** (Cui et al., 2023): UltraFeedback is a diverse preference dataset with 64k prompts and 256k responses from various sources, annotated by GPT-4 for instruction-following, truthfulness, honesty, and helpfulness. It includes 380k high-quality feedback entries, allowing the creation of 1 million comparison pairs. Prompts are sourced from datasets like UltraChat, ShareGPT, Evol-Instruct, TruthfulQA, FalseQA, and FLAN, ensuring broad representation and diversity.

- **Nectar** (Banghua et al., 2023): Nectar is a high-quality 7-wise comparison dataset. It features diverse chat prompts from sources like lmsys-chat-1M, ShareGPT, Antropic/hh-rlhf, UltraFeedback, Evol-Instruct, and Flan. Responses from models such as GPT-4, GPT-3.5-turbo, LLama-2-7B-chat, and Mistral-7B-Instruct are ranked by GPT-4, resulting in 3.8M pairwise comparisons.

- **Py-DPO**[9]: The DPO dataset enhances Python coding abilities using the validated Python-Alpaca dataset for "chosen" responses. "Rejected" values, generated with a mix of airoboros-l2-13b-3.1.1 and bagel-7b-v0.1, are assumed to be of lower quality.

- **Distilabel-Capybara**[10]: The Distilabel-Capybara dataset, created by distilabel addresses the lack of multi-turn open datasets for DPO/RLHF by providing multi-turn dialogue preferences on top of Capybara.

- **Distilabel-Orca**[11]: Similar to Distilabel-Capybara, this dataset is created by distilabel to generate preference labels on top of Orca.

Details of Online Preference Data

We used data from UltraFeedback (Cui et al., 2023) as prompts. We sample multiple responses from π_θ and use Pair-RM (Jiang et al., 2023b) as a judge to obtain preference pairs. This results in an online collected dataset of 60k in size.

B.4 Performance Details

Table A5 presents a comprehensive breakdown of the performance of our Gemma-2b and LLaMA-3-8b models across various OpenLLM tasks, highlighting the improvements brought by the lion-sft, lion-dpo, and lion-odpo alignment training methods. The lion-sft model showed substantial improvements across all tasks, with significant gains in GSM8k (53.37) and TruthfulQA (43.80). Building on these improvements, the lion-dpo model particularly enhanced ARC (52.30) and HellaSwag (72.47), while maintaining strong performance in other tasks. The lion-odpo model achieved the highest scores overall, excelling in ARC (53.75) and HellaSwag (73.04), and maintaining superior performance in GSM8k (53.53).

For the LLaMA-3-8b model, the lion-sft variant displayed robust performance across all tasks, with notable scores in GSM8k (76.72) and TruthfulQA (54.26). The lion-dpo model further improved performance, achieving higher scores in ARC (63.91), HellaSwag (82.95), and significantly in GSM8k (80.59) and TruthfulQA (60.01). The lion-odpo model marginally outperformed the lion-dpo model, attaining the highest scores in ARC (63.99),

[9] https://huggingface.co/datasets/jondurbin/py-dpo-v0.1

[10] https://huggingface.co/datasets/argilla/distilabel-capybara-dpo-7k-binarized

[11] https://huggingface.co/datasets/argilla/distilabel-intel-orca-dpo-pairs

HellaSwag (83.18), and TruthfulQA (61.12), while maintaining exceptional performance across all tasks.

Both the Gemma-2b and LLaMA-3-8b models benefit significantly from the alignment training, with each subsequent model (sft, dpo, odpo) showing progressive improvements. The lion-odpo models generally achieve the highest scores, demonstrating the effectiveness of this alignment method in enhancing model performance across diverse tasks.

B.5 More Details on Qualitative Analysis

To provide insights into the opaque training process during preference learning, we additionally record how the sequence probability for y_w and y_l change after DPO training. We use a fixed unseen test set of (x, y_w, y_l) obtained from public datasets (see Appendix A.1), and compute $\pi_\theta(y_w) - \pi_\theta(y_l)$ for each preference pair before and after training. We then qualitatively compare various models from Table 6 and from Section 3. We present the visualizations in Figure 4.

In Figure 4, we find that many of the best-performing models have a parabolic shape as shown in Figure 4(b). This indicates that well-trained models learn to not only increase confidence in pairs they could distinguish correctly before training (i.e., $\pi_\theta(y_w) > \pi_\theta(y_l)$), but also improve on pairs where they previously could not (i.e., $\pi_\theta(y_w) < \pi_\theta(y_l)$). Undertrained models (Figure 4(a)) achieve a similar shape but with a much smaller magnitude. While overtrained models (Figure 4(c)) show a change in probability at an even greater scale (e.g., for $\pi_\theta(y_w) < \pi_\theta(y_l)$), it is often achieved by sacrifising performance on the other side of the plot (e.g., $\pi_\theta(y_w) > \pi_\theta(y_l)$).