THE DELTA LEARNING HYPOTHESIS: PREFERENCE TUNING ON WEAK DATA CAN YIELD STRONG GAINS

Anonymous authors
Paper under double-blind review

ABSTRACT

Preference tuning has greatly improved large language models (LLMs), yet obtaining preference data remains challenging, often requiring expensive human annotation or strong LLM judges to assess response quality. We explore the feasibility of synthetically generating preference pairs *without* optimizing the preferred response quality to train LLMs that surpass the preferred responses. We formulate the **delta learning hypothesis**, which posits that models can improve beyond the quality of their training data by learning solely from the relative quality difference—-rather than the absolute quality—of paired responses. To validate this hypothesis, we conduct controlled experiments across diverse domains: a toy stylistic task (bold section generation), a math reasoning task (GSM8K), and real-world instruction-following. We show that preference tuning via Direct Preference Optimization (DPO) can enable models to extrapolate improvements from suboptimal data, whereas directly imitating weak data through supervised fine-tuning (SFT) can degrade performance. Armed with these insights, we build a simple weak-to-strong setup that achieves consistent gains over Llama-3.1-8B-Instruct, as well as a SOTA-competitive preference dataset—all without any strong judge.

1 INTRODUCTION

Preference tuning has become a crucial step in the LLM development pipeline (Lambert et al., 2024; Touvron et al., 2023; Ivison et al., 2024). Modern recipes for generating preference data typically involve sampling many responses to a set of prompts and then distinguishing the best responses as the chosen (preferred) response using either human annotation or a strong LLM judge (Lambert et al., 2024; Touvron et al., 2023). Such practices implicitly assume that optimizing the quality and correctness of the chosen response is crucial for downstream training. In this work, we challenge this assumption and introduce the **delta learning hypothesis**, which posits that the absolute quality of the chosen response is not necessary to drive learning. Instead, it suffices to ensure that a meaningful *delta* exists between the chosen and rejected (dispreferred) responses. Moreover, we hypothesize that such deltas can enable improvement beyond the absolute quality of the chosen response.

In Section 2, we present an intriguing empirical finding that motivates our hypothesis. Section 3 formalizes the hypothesis and explores it in two controlled settings, where we can explicitly manipulate the quality of both chosen and rejected responses. Section 4 extends our findings to real-world applications. Notably, we (1) achieve consistent self-improvement gains over Llama-3.1-8B-Instruct and (2) construct a preference dataset that rivals a SOTA dataset built using a GPT-4o judge (Tulu 3 data, Lambert et al. (2024))—all without relying on any LLM judge or humans for supervision.

2 ULTRAFEEDBACK: A MOTIVATING CASE STUDY

We begin our study with an intriguing empirical result that motivates our central hypothesis. We find that training on paired preference data generated by weak models can improve a stronger model's performance, even when finetuning directly on the weaker models' outputs hurts.

Data. The ULTRAFEEDBACK preference dataset (Cui et al., 2023) is constructed by prompting a set of large language models (LLMs) with a diverse set of prompts and then scoring the resulting responses using a strong judge model (GPT-4). For each prompt x, we form preference pairs (x, y_c, y_r) by selecting one high-scoring response y_c and one lower-scoring response y_r. Because ULTRAFEEDBACK was constructed in 2023, the models underlying these responses (e.g., Llama 2) are generally weaker than modern LLMs (e.g., Llama 3). To ensure a clear performance gap

between (1) the weaker data generators and (2) the stronger models we later train, we exclude any responses from models that have an LMSYS Chatbot Arena ELO score close to or above Llama-3.2-3B-Instruct. We call the resulting filtered dataset ULTRAFEEDBACK-WEAK. See Appendix A.1 for full filtering details.

Training and evaluation. We finetune Llama-3.2-3B-Instruct and Llama-3.1-8B-Instruct (Touvron et al., 2023) on ULTRAFEEDBACK-WEAK in two ways. One, we use (1) Direct Preference Optimization (DPO) to train on the preference pairs (x, y_c, y_r), updating the model to prefer higher-scoring responses y_c to lower-scoring responses y_r (Rafailov et al., 2024). Crucially, although the chosen response y_c derives from a model that is weaker than the models we train, it is still higher-quality relative to the rejected response y_r. We compare DPO to (2) Supervised Finetuning (SFT) directly on the chosen responses (x, y_c). To ablate the effect of the regularization used in DPO, we also consider **SFT + regularization**, where we apply LoRA with very low rank ($r = 8$) to constrain the optimized model to be close to the base model (Biderman et al., 2024; Hu et al., 2021). See Appendix A.1 for more training details and hyperparemters.

We evaluate the trained models on a suite of standard downstream tasks measuring instruction following (IFEval (Zhou et al., 2023), AlpacaEval 2 (Dubois et al., 2024)), math reasoning (MATH (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021)), truthfulness (TruthfulQA (Lin et al., 2021)), and general reasoning (BigBenchHard (Suzgun et al., 2022)).

Model	MATH	GSM8K	BBH	TruthfulQA	AlpacaEval2	IFEval	Avg.
LLAMA-3.2-3B-INSTRUCT	36.5	69.0	51.7	53.9	19.4	66.7	49.5
+ UF-WEAK SFT	24.0	63.5	50.6	46.0	5.5	40.8	38.4
+ UF-WEAK SFT (reg.)	26.5	66.5	50.7	46.6	4.7	41.4	39.4
+ UF-WEAK DPO	**37.5**	**71.0**	**53.2**	**59.2**	**22.0**	**67.3**	**51.7**
LLAMA-3.1-8B-INSTRUCT	42.5	81.3	66.7	60.1	20.9	71.0	57.1
+ UF-WEAK SFT	31.5	69.0	62.3	44.9	6.6	50.6	44.2
+ UF-WEAK SFT (reg.)	39.2	74.5	**68.2**	45.0	9.3	65.4	50.3
+ UF-WEAK DPO	**44.0**	**86.0**	62.6	**63.0**	**25.6**	**72.5**	**59.0**

Table 1: We tune Llama-3 Instruct models on ULTRAFEEDBACK-WEAK (UF-WEAK), which comprises of preference data generated by weaker models. Training with DPO to prefer "weak over weaker" yields gains, while SFT (with/without regularization) on the weak responses directly hurts.

Results. We present results in Table 1. Both standard SFT and SFT + regularization substantially *hurt* performance—likely because the models are finetuned to imitate responses from weaker models. In contrast, DPO on the same weak data consistently improves performance, indicating that there is valuable signal in the pairwise contrast between chosen and preferred outputs.

3 THE DELTA LEARNING HYPOTHESIS

Why does preference learning with weak data help when SFT on the same data hurts? We hypothesize that training on paired responses (x, y_c, y_r) allows the model to learn from the relative quality difference between y_c and y_r—the delta—rather than their absolute quality. Even if both responses y_c, y_r have low absolute quality compared to the model we aim to improve, as long as y_c is meaningfully better than y_r, the model can learn from this delta to improve.

Formally, let $\mu(x, y)$ be the "true" utility of a response y to some prompt x. The **delta learning hypothesis** posits that as long as $\mu(x, y_c) > \mu(x, y_r)$, even if both are suboptimal, the paired-preference signal suffices to drive improvement *beyond* the quality $\mu(x, y_c)$ of the preferred response y_c. In contrast, supervised fine-tuning encourages the model to simply imitate y_c, which can hinder improvement and even be detrimental when $\mu(x, y_c)$ is low. In the remainder of this section, we will present experiments in controlled settings where we explicitly impose definitions of μ and construct responses of varying utility to explore this hypothesis.

3.1 CONTROLLED SETTING: NUMBER OF BOLD SECTIONS

To study our hypothesis, we define $\mu(x, y)$ to be "the number of Markdown-denoted bold section headers in y" (e.g., ****My Header****). Suppose we have a model M that produces 6 such sections

on average, so that $\mu\big(x, M(x)\big) = 6$. Our hypothesis predicts that if we collect pairs (x, y_c, y_r) where y_c contains, say, 3 sections and y_r contains 2, then training M to prefer y_c over y_r should teach the model to produce more sections overall—even though 3 sections (the "better" response) is still less than the model's current capability. As we show below, this is indeed what we observe.

Data. We construct a dataset of prompts x with responses $y_{k_1} \dots y_{k_n}$ containing varying numbers k_i of bolded sections. To construct each prompt x, we start with prompts from the Tulu 3 SFT dataset (Lambert et al., 2024) and append the instruction "Include bolded sections in your response." To obtain the associated responses y_{k_i}, we modify the appended instruction into a hard constraint: "Include exactly k_i bolded sections in your response." We use Llama-3.2-3B-Instruct to generate responses adhering to this constraint; we filter for correct adherence with regular expressions.

Training and evaluation. We tune Llama-3.2-3B-Instruct with DPO on preference pairs (x, y_{k_i}, y_{k_j}) formed by selecting a response y_{k_i} with more sections ($k_i > k_j$) as the preferred response. To isolate the effect of the positive delta from any potential confounding effects associated with DPO training, we also evaluate two control settings: (1) reversing the preference pairs ($k_i < k_j$) and (2) creating preference pairs where both preferred and rejected responses contain equal numbers of sections ($k_i = k_j$). Finally, we compare against SFT directly on the prompts and chosen responses (x, y_{k_i}) of each preference pair. We use a validation set to pick training hyperparameters. See Appendix A.2 for training details. We evaluate by measuring the average number of bolded sections generated before and after training in response to a set of held-out test prompts.

Model/Algorithm	Chosen Res.	Rejected Res.	Section Delta	**# Sections Generated**
Llama-3.2-3B-Instruct (Baseline)	—	—	—	5.9
+ DPO	3 sections	2 sections	**+1**	81.1 (**+ 75.2**)
+ DPO	2 sections	3 sections	**−1**	1.1 (**− 4.8**)
+ DPO	3 sections	3 sections	0	6.1 (**+ 0.2**)
+ SFT	3 sections	—	—	4.4 (**− 1.5**)
+ SFT	2 sections	—	—	2.9 (**− 3.0**)

Table 2: We train Llama-3.2-3B-Instruct with DPO on chosen and rejected responses with varying numbers of bold sections, and compare to SFT on the chosen response directly. When the responses contain less sections than the model's baseline behavior, the sections generated decreases after SFT. In contrast, training with DPO enables the model to learn from the *delta* between responses, improving the model's number of sections generated even when each response individually is suboptimal.

Results. We present results in Table 2 and qualitative examples of model generations in Figure 1. Results strongly support our hypothesis: SFT on responses with fewer sections than the original model's generations reduces the number of generated sections. However, even when responses are individually suboptimal, pairing them together with a positive delta massively boosts section generation, extrapolating beyond the number of sections contained in the chosen response[1]. Moreoever, DPO with a negative delta hurts; DPO with zero delta minimally changes the model. This suggests that the positive delta is essential for driving the observed gains.

3.2 Controlled setting: GSM8K with inserted errors

To test whether the delta learning hypothesis extends beyond stylistic features to semantic quality, we controllably introduce arithmetic errors into GSM8K solutions. Our hypothesis predicts that training to prefer "fewer mistakes" over "more mistakes" should enhance mathematical reasoning, even when direct SFT on error-ridden solutions is detrimental.

Data. We use a strong model (Llama-3.1-70B-Instruct, which has near-perfect accuracy on GSM8K) to introduce either 2 or 3 arithmetic errors into ground truth solutions from the GSM8K train set. Each GSM8K problem x is thus associated with two responses, y_2 (2 errors) and y_3 (3 errors). See Figure 2 for qualitative examples of the corrupted solutions.

[1]The model learns to make every single word a new section header! See Figure 1.

Training and evaluation. We tune from OLMo-1.7-7B-SFT, a model that is not saturated on GSM8K. We train using DPO on GSM8K problems with solution pairs (x, y_2, y_3) as well as flipped preferences (x, y_3, y_2). We compare against SFT directly on corrupted solutions (x, y_2). We evaluate via standard 8-shot chain-of-thought prompting on a 919 question subset of the GSM8K test set. We use the remaining 400 test set questions as a validation set to select hyperparameters.

Model/Algorithm	Chosen response	Rejected response	Error Delta	GSM8K Acc.
OLMo-1.7-7B-SFT (Baseline)	—	—	—	22.7
+ DPO	Solution w/ 2 errors	Solution w/ 3 errors	**− 1**	24.2 (**+ 1.5**)
+ DPO	Solution w/ 3 errors	Solution w/ 2 errors	**+ 1**	21.8 (**− 0.9**)
+ SFT	Solution w/ 2 errors	—	—	20.6 (**− 2.1**)

Table 3: We train OLMo-1.7-7B-SFT using DPO and SFT on GSM8K solutions containing varying amounts of synthetically-inserted arithmetic errors. DPO training yields gains when the chosen response has less errors than the rejected response, even though both are error-ridden.

Results. See Table 3. The results are consistent with our hypothesis and the findings from Section 3.1. Even though SFT on GSM8K solutions with 2 errors hurts, as long as we pair it against a solution with *more* errors (i.e., 3 errors), training with DPO to prefer "less errors" yields gains.

4 Pushing towards SOTA

Now that we've validated our hypothesis in our controlled settings, how might we leverage the insight in practice? In this section, we present two gains that that we achieve by constructing paired data with an explicit gap, even when the chosen response's quality is not optimized. We evaluate on the benchmark suite detailed in Section 2. See Appendix A.4 for full training details.

Model	MATH	GSM8K	BBH	TruthfulQA	AlpacaEval2	IFEval	Avg.
LLAMA-3.1-8B-INSTRUCT	42.5	81.3	**66.7**	60.1	20.9	71.0	57.1
+ SFT (self-generated responses)	36.6	80.0	62.2	57.7	22.1	67.8	54.4
+ DPO (self vs. weaker responses)	**43.0**	**83.7**	66.6	**63.0**	**26.0**	**71.5**	**59.0**
TULU-3-8B-SFT	31.5	76.2	**69.7**	48.0	12.4	64.7	50.4
+ DPO (Tulu 3 data, GPT-4o judge)	44.0	84.3	68.7	72.9	33.5	**76.3**	63.3
+ DPO (ours, Tulu prompts no judge)	**44.9**	**85.5**	62.3	**85.8**	**36.8**	74.3	**64.9**

Table 4: Our hypothesis implies that we can create useful preference data for DPO training *without* optimizing the quality of the chosen responses. Leveraging this insight, we achieve gains using only self- and weak-supervision by pairing self-generated responses with weaker responses (top half). We also create SOTA-comparable preference data without using any LLM judge (bottom half).

Improving a SOTA LLM with only self- and weak-supervision. Starting from the Tulu 3 SFT prompts x, we greedy decode with Llama-3.1-8B-Instruct to obtain responses y. By construction, these responses reflect the model's current capability level, and so we would not expect SFT on (x, y) to yield gains. However, our hypothesis implies that so long as we can pair response y with a response y' that is worse, then DPO training can extract supervision from the delta between y, y'. To obtain such a y', we simply use greedy responses from a smaller model, in this case Llama-3.2-3B-Instruct. We train on the resulting pairs (x, y, y') with DPO and SFT. We present results in Table 4 (top half); our DPO setup improves over Llama-3.1-8B-Instruct without any stronger supervision.

Constructing SOTA preference data without strong models. Following a similar setup as above, we start with the Tulu 3 DPO prompts and greedy decode chosen and rejected responses y, y' with Qwen-2.5-3B-Instruct and Qwen-2.5-1.5B-Instruct (Qwen Team, 2024), respectively. Crucially, we do not use a strong LLM judge to assess response quality in forming these pairs; following our hypothesis, we rely solely on the delta that implicitly exists between the chosen and rejected response. We compare DPO training from the same base model (Tulu3-8B-SFT) with (a) our preference data to (b) the Tulu 3 DPO preference data, a SOTA dataset constructed with a GPT-4o judge. As shown in Table 4 (bottom half), our preference data rivals and often outperforms SOTA.

5 Related Work

Language model preference tuning. Preference tuning (Ziegler et al., 2019) has become an integral part of the LLM development pipeline. It is now widely used to improve the safety (Dai et al., 2023), helpfulness and harmlessness (Ouyang et al., 2022), as well as general capabilities Lambert et al. (2024) of LLMs. Initial approaches for preference tuning used human-annotated preference data to train a reward model (Ziegler et al., 2019; Ouyang et al., 2022), and then optimized against the reward model using reinforcement learning algorithms such as PPO (Schulman et al., 2017). Recent work has sought to simplify this pipeline by (1) approximating human annotations with a strong LLM judge (Cui et al., 2023) and (2) removing the need for an explicit external reward model by moving towards direct optimization algorithms like DPO (Rafailov et al., 2024). In our work, we seek to understand whether the chosen response truly needs to be high quality for preference tuning to succeed—we seek to explore the feasibility of preference tuning on suboptimal data and on pairs constructed without strong judge supervision.

Weak-to-strong generalization. As language models continually improve, a natural question that arises is how we might improve them beyond the frontier of human capability. The weak-to-strong generalization problem thus seeks to study the extent to which weak supervision can elicit strong behaviors in trained models (Burns et al., 2023; Hase et al., 2024). Consistent with our motivating case study, recent work has observed that preference tuning on wrong answers only can yield improvements, so long as the chosen response is "less wrong" than the rejected (Yao et al., 2024). Our work seeks to deeply understand and push the limits of this phenomenon. We formalize a concrete hypothesis that we empirically verify with controlled experiments, and show that the implications can lead to SOTA-competitive gains.

6 Limitations and future work

Our study has several limitations that leave open questions for future work.

First, due to compute limitations, we use LoRA tuning with very high rank for the experiments in Section 4 to approximate full finetuning (Biderman et al., 2024). See Appendix A.4 for more details. We hypothesize that our results hold in the full finetuning regime, but we were not able to validate this empirically.

Second, while we have empirically demonstrated that preference tuning can extract supervision from the delta in quality between chosen and rejected responses, what specific properties of the delta matter remain unclear. Does a larger delta result in larger gains for preference tuning? What is more critical—the absolute quality of the chosen response or the delta in quality between chosen and rejected responses? Furthermore, are different deltas composable, and if so, under what conditions? We are optimistic that understanding which tasks benefit most from delta-based learning and understanding how to better synthesize deltas is a promising direction.

Third, we have not characterized how learning from deltas scales. Do our findings generalize to significantly larger models, such as LLMs with 70B parameters? Additionally, how does having suboptimal chosen responses relative to traditional preference data impact dataset size scaling?

Finally, the underlying factors driving our observation that comparing Qwen-2.5-3B-Instruct responses to Qwen-2.5-1.5B-Instruct responses rivals the Tulu 3 preference dataset remain unclear. Does this trend hold broadly when pairing responses from a larger and a smaller model, or is it due to some specific effect from the Qwen models? Moreover, how does the reward signal derived from this heuristic compare to the traditional reward signal from a strong LLM judge? Are these signals complementary, or do they capture fundamentally different aspects of preference tuning?

We leave these questions for future work.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic,

Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. Falcon-40B: an open large language model with state-of-the-art performance. 2023.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. Lora learns less and forgets less. *arXiv preprint arXiv:2405.09673*, 2024.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

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. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

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

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

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

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

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

Peter Hase, Mohit Bansal, Peter Clark, and Sarah Wiegreffe. The unreasonable effectiveness of easy training data for hard tasks. *arXiv preprint arXiv:2401.06751*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

Hamish Ivison, Yizhong Wang, Jiacheng Liu, Zeqiu Wu, Valentina Pyatkin, Nathan Lambert, Noah A. Smith, Yejin Choi, and Hannaneh Hajishirzi. Unpacking dpo and ppo: Disentangling best practices for learning from preference feedback, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. Unintentional unalignment: Likelihood displacement in direct preference optimization. *arXiv preprint arXiv:2410.08847*, 2024.

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

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

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

MosaicML NLP Team. Introducing mpt-30b: Raising the bar for open-source foundation models, 2023. URL www.mosaicml.com/blog/mpt-30b. Accessed: 2023-06-22.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Lewis Tunstall, Nathan Lambert, Nazneen Rajani, Edward Beeching, Teven Le Scao, Leandro von Werra, Sheon Han, Philipp Schmid, and Alexander Rush. Creating a coding assistant with starcoder. *Hugging Face Blog*, 2023. https://huggingface.co/blog/starchat.

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

Jihan Yao, Wenxuan Ding, Shangbin Feng, Lucy Lu Wang, and Yulia Tsvetkov. Varying shades of wrong: Aligning llms with wrong answers only. *arXiv preprint arXiv:2410.11055*, 2024.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Experimental details

We use code from the OpenRLHF Github repository (Hu et al., 2024) to train all of our models, and evaluate with the OpenInstruct repository (Ivison et al., 2024). We use the default evaluation settings from OpenInstruct and report strict prompt accuracy for IFEval, truth-info accuracy for TruthfulQA, length-controlled winrate for AlpacaEval, 4-shot CoT accuracy for MATH, 8-shot CoT accuracy for GSM8K, and CoT accuracy for BBH.

A.1 ULTRAFEEDBACK EXPERIMENT

Filtering. As of February 7 2025, Llama-3.2-3B-Instruct achieves a LMSYS Chatbot Arena ELO score of 1103 and Llama-3.1-8B-Instruct achieves 1176 ELO. We filter out all responses from the original ULTRAFEEDBACK dataset that were generated by models with higher ELO than 1100. This excludes GPT-4-0613 (1163 ELO), GPT-3.5-Turbo (1106 ELO), and WizardLM-70B (1106 ELO). The best remaining model is Vicuna-33B (1091 ELO); see Table 5 for a full list of remaining models.

Hyperparmeters. We generally use hyperparameters close to defaults suggested by recent work (Lambert et al., 2024; Hu et al., 2024), and sweep a reasonable range for each method to avoid overoptimizing any one particular setup. We use length normalization in the DPO loss, which Lambert et al. (2024) suggests to generally work better. For both SFT and DPO, we use a cosine annealing LR schedule with a warmup ratio of 0.03. For DPO, we use batch size 32 and train for one epoch. We sweep LR in {1e-7, 3e-7, 5e-7, 7e-7} as well as DPO $\beta \in \{5, 10\}$. For SFT, we use batch size 256, and sweep epochs in $\{1, 2\}$ and learning rate in {1e-5, 5e-5, 1e-6}. When constraining SFT with low-rank LoRA, we use $r = 8$ and $\alpha = 2r = 16$.

Model	Reference
Alpaca-7B	Taori et al. (2023)
Bard	https://bard.google.com/
Falcon-40B-Instruct	Almazrouei et al. (2023)
Llama-2-13B-Chat	Touvron et al. (2023)
Llama-2-70B-Chat	Touvron et al. (2023)
Llama-2-7B-Chat	Touvron et al. (2023)
MPT-30B-Chat	Team (2023)
Pythia-12B	Biderman et al. (2023)
StarChat	Tunstall et al. (2023)
UltraLM-13B	Ding et al. (2023)
UltraLM-65B	Ding et al. (2023)
Vicuna-33B	Chiang et al. (2023)
WizardLM-7B	Xu et al. (2023)
WizardLM-13B	Xu et al. (2023)

Table 5: Models used to generate the responses in our ULTRAFEEDBACK-WEAK dataset, constructed by filtering out all responses generated by Llama-3.2-3B-Instruct level models from the original ULTRAFEEDBACK dataset.

A.2 NUMBER OF BOLDED SECTIONS EXPERIMENT

We use the same hyperparameters and sweep the same ranges as in Appendix A.1. We train on exactly 16384 data points for each setting.

A.3 GSM8K WITH ERRORS

We use largely the same hyperparameters and sweep the same ranges as in Appendix A.1. Because the chosen and rejected responses are highly similar and differ (by construction) by only a few tokens, vanilla DPO training may drastically reduce the likliehood of the chosen response (Razin et al., 2024). To mitigate this, we follow existing practice and also consider adding a NLL regularization term to the DPO objective, which we sweep as a hyperparameter. We select best hyperparmeters on a held-out validation set. We train on exactly 7473 samples (the size of the full GSM8K train set) for each setting.

A.4 REAL-WORLD PREFERENCE TUNING EXPERIMENTS

We use same hyperparmeters as in Appendix A.1, except we sweep DPO hyperparameters less extensively due to compute limitations: we only consider LR in {1e-7, 5e-7}. We also train models with LoRA as opposed to full-finetuning; to limit the error introduced by this approximation, we

use a very high rank of $r = 256$ ($\alpha = 2r = 512$), which recent work suggests can generally match full finetuning performance (Biderman et al., 2024). This is a limitation of our study that we hope to address in future work.

B QUALITATIVE EXAMPLES

B.1 NUMBER OF BOLDED SECTIONS

Examples of model generations before and after DPO training are presented in Figure 1.

B.2 TRAINING DATA FOR GSM8K WITH ARITHMETIC ERRORS

Examples of the corrupted training data are presented in Figure 2.

Llama-3.2-3B-Instruct (no training baseline)

User: ...assuming the graph is Eulerian, determine the number of possible closed trails of length 7 (i.e., the number of Eulerian circuits of length 7) that the guide can choose from, if possible. (Note: A graph is Eulerian if it has an Eulerian circuit, and it must satisfy certain conditions). If the tour guide decides to add an additional attraction to the itinerary, represented by adding a new node and connecting it with edges to 3 existing nodes, analyze how this change affects the Eulerian properties of the graph and determine the new possible number of closed trails of length 7, if any exist.

Include bolded sections in your response.

Assistant: ****Problem Analysis****

The problem asks for the number of possible closed trails (Eulerian circuits) in a graph with 50 nodes and 100 edges, and then analyzing the effect of adding a new node and edges on the Eulerian properties of the graph.

****Initial Graph Properties****

⟨text ommitted without sections for space⟩

****Initial Number of Eulerian Circuits****

⟨text without sections ommitted for space⟩

****Adding a New Node and Edges****

⟨text without sections ommitted for space⟩

****Conclusion****

The addition of a new node and edges to the graph does not allow for any Eulerian circuits of length 7, since the new node introduces a constraint that cannot be satisfied. Therefore, the number of possible closed trails of length 7 is 0.

After DPO training (3 sections chosen response / 2 sections rejected response)

User: ...assuming the graph is Eulerian, determine the number of possible closed trails of length 7 (i.e., the number of Eulerian circuits of length 7) that the guide can choose from, if possible. (Note: A graph is Eulerian if it has an Eulerian circuit, and it must satisfy certain conditions). If the tour guide decides to add an additional attraction to the itinerary, represented by adding a new node and connecting it with edges to 3 existing nodes, analyze how this change affects the Eulerian properties of the graph and determine the new possible number of closed trails of length 7, if any exist.

Include bolded sections in your response.

Assistant: **Initial Conditions **The graph **has 50 nodes** and **100 edges**, **satisfying the conditions** of an Eulerian graph **with a degree** of at least **2** at **all** nodes **except** possibly **one**.

Eulerian Circuit Conditions

For a graph to have an Eulerian circuit **(i.e.,** **an Eulerian path** **that** **loops** **back** **to** **the** **starting** **node** **exactly** **once** **),** **all** **nodes** **must** **have** **even** **degrees** **or** **the** **number** **of** **edges** **going** **into** **a** **node** **must** **be** **equal** **to** **the** **number** **of** **edges** **leaving** **it** **exactly** **once** **for** **each** **node** **except** **the** **starting** **and** **ending** **node** **(**in** **the** **case** **of** **a** **closed** **trail** **or** **path** **of** **length** **greater** **than** **2** **)** **.

New Graph Conditions

After **adding** **a** **new** **node** **and** **edges** **to** **3** **existing** **nodes** **the** **graph** **will** **no** **longer** **be** **Eulerian** **since** **not** **all** **nodes** **will** **have** **even** **degrees** **any** **more** **the** **new** **node** **will** **have** **degree** **3** **and** **two** **existing** **nodes** **will** **have** **degree** **4** **and** **the** **remaining** **46** **nodes** **will** **have** **degree** **2** **or** **more** **than** **2** **.

Conclusion

The **initial** **graph** **had** **the** **potential** **for** **Eulerian** **circuits** **of** **length** **7** **but** **the** **new** **graph** **does** **not** **satisfy** **the** **conditions** **for** **an** **Eulerian** **graph** **any** **more** **the** **number** **of** **closed** **trails** **of** **length** **7** **is** **** **0** **.**

Figure 1: DPO training massively increases the number of sections generated by the model (from 5 to 89 in this example). Most notably, the increase extrapolates beyond the number of sections (i.e., absolute quality) of the chosen response (3 sections).

Ground truth solution (uncorrupted)

User: Teresa scored 70 marks in science, 80 in music, 85 in social studies, and the physics exam, which was the hardest test that Teresa had ever taken in her entire life, she scored half as many marks as she scored in music. Calculate the total marks Teresa scored in all the subjects.

Assistant: The total marks of science and music are 70+80 = <<70+80=150>>150

When you add the marks she scored in social studies, the full marks become 150+85 = <<150+85=235>>235.

She scored half as many marks in physics as she scored music, totaling 1/2*80 = 40 marks.

When you add the marks she scored in physics to the total marks of the other subjects, the full marks become 235+40 = <<235+40=275>>275

275

Solution with 2 errors added

User: Teresa scored 70 marks in science, 80 in music, 85 in social studies, and the physics exam, which was the hardest test that Teresa had ever taken in her entire life, she scored half as many marks as she scored in music. Calculate the total marks Teresa scored in all the subjects.

Assistant: The total marks of science and music are 70+80 = <<70+80=**151**>>**151**

When you add the marks she scored in social studies, the full marks become 151+85 = <<151+85=236>>236.

She scored half as many marks in physics as she scored music, totaling 1/2*80 = **41 marks**.

When you add the marks she scored in physics to the total marks of the other subjects, the full marks become 236+41 = <<236+41=277>>277

277

Solution with 3 errors added

User: Teresa scored 70 marks in science, 80 in music, 85 in social studies, and the physics exam, which was the hardest test that Teresa had ever taken in her entire life, she scored half as many marks as she scored in music. Calculate the total marks Teresa scored in all the subjects.

Assistant: The total marks of science and music are 70+80 = <<70+80=**151**>>**151**

When you add the marks she scored in social studies, the full marks become 151+85 = <<151+85=236>>236.

She scored half as many marks in physics as she scored music, totaling 1/2*80 = **39 marks**.

When you add the marks she scored in physics to the total marks of the other subjects, the full marks become 236+39 = <<236+39=**274**>>**274**

274

Figure 2: We prompt a strong langauge model to explicitly introduce either 2 or 3 errors into the ground truth solutions from the GSM8K training set. The resulting errors are **bolded**.