

Data-Centric Control of Verbosity for DPO-Based Instruction Alignment

Stanford CS224N {Custom} Project

Will Furlow

Department of Computer Science
Stanford University
wfurlow@stanford.edu

Susan Lee

Department of Computer Science
Stanford University
leesusan@stanford.edu

Abstract

Preference-based alignment methods such as Reinforcement Learning from Human Feedback (RLHF) and Direct Preference Optimization (DPO) are widely used to train large language models (LLMs) to produce helpful, instruction-following responses. However, these methods are known to over-optimize for verbosity, often favoring longer responses even when they do not improve answer quality. While prior work primarily addresses this issue through modifications to the optimization algorithm, we investigate whether verbosity bias in preference datasets can be mitigated through careful data design. To test this hypothesis, we construct preference pairs that contrast concise-correct and verbose-correct responses, ensuring that response length, independent of correctness, does not influence the learned preference. We generate responses at three controlled verbosity levels using a teacher LLM and filter low-quality outputs with an LLM-as-a-judge. Using this dataset, we fine-tune Mistral-7B-Instruct-v0.2 with standard DPO without modifying the training objective. Our verbosity-disentangled model achieves the highest length-controlled win rate (20.6%) among all fine-tuned conditions while producing substantially shorter responses than vanilla DPO (829 vs. 1488 average characters). These results show that carefully designed preference data can decouple verbosity from quality, enabling more concise and human-aligned responses without modifying the optimization algorithm.

1 Key Information

- Mentor: David Anuraha
- No external collaborators, not sharing project
- Late Days:

Team member	Late days remaining
Susan	1
Will	3
Total pooled	4
Team size	2
Late days applied for the report	2

2 Introduction

Preference-based alignment methods such as Reinforcement Learning from Human Feedback (RLHF) and Direct Preference Optimization (DPO) have become the dominant paradigm for aligning large language models (LLMs) with human preferences [1, 2]. However, models trained with these methods often exhibit a tendency to generate unnecessarily verbose responses, favoring longer outputs even

when they do not improve answer quality [3]. This issue is further compounded during evaluation: LLM-based judges have been shown to exhibit verbosity bias, systematically preferring longer responses even when their quality is comparable [4]. This is problematic, as verbose responses have been shown to degrade reasoning quality and instruction-following capabilities [5].

Several mechanisms contribute to verbosity bias. At the algorithmic level, sequence likelihoods are products of per-token probabilities that decrease with length, causing longer preferred responses to receive larger gradient updates during optimization [5]. Park et al. [3] also show that the implicit reward learned by DPO develops strong correlations with response length, with length explaining 30–46% of reward variance. In addition, preference data also exhibits verbosity bias, systematically preferring longer responses [3].

Existing approaches to mitigating verbosity bias are predominantly algorithm-centric. R-DPO [3] introduces a regularization term based on length differences, LD-DPO [5] modifies token likelihoods to reduce length sensitivity, and SimPO [6] uses length-normalized rewards. While these approaches modify the optimization objective, they do not address another potential source of the problem: the structure of the preference data itself.

This raises a question that prior work has largely left open: can dataset construction alone mitigate verbosity bias? If preference pairs explicitly contrast concise-correct responses against verbose-correct ones, the learning algorithm would observe that verbosity does not imply higher quality. In principle, this would allow the model to disentangle response length from correctness without modifying the training objective.

We test this hypothesis by constructing a verbosity-disentangled preference dataset using a teacher LLM (Kimi, moonshot-v1-8k) to generate concise, verbose, and too-short responses for 10,000 prompts sampled from UltraFeedback [7] and HelpSteer2 [8], filtered with an LLM-as-judge (GPT-4o-mini). We train standard DPO on Mistral-7B-Instruct-v0.2 on the resulting preference pairs and evaluate on AlpacaEval 2.0 [9] using length-controlled win rate (LC-WR) as our primary metric. Our verbosity-disentangled model achieves the highest LC-WR among all fine-tuned conditions (20.6%) at substantially shorter average length than vanilla DPO (829 vs. 1488), supporting the hypothesis that dataset design alone is a viable lever for verbosity control.

Our contributions are: (1) a scalable pipeline for constructing verbosity-disentangled preference data using style-conditioned teacher LLM generation and LLM-as-judge filtering; (2) a controlled comparison of data-centric versus algorithm-centric verbosity reduction across seven model conditions; and (3) an empirical analysis of the conciseness–quality tradeoff under a data-driven intervention, including limitations such as the interaction between SFT initialization and verbosity-specific preference optimization.

3 Related Work

Recent work has identified several unintended behavioral biases introduced by preference-based training [3, 10, 4, 11]. In particular, multiple studies have shown that preference optimization methods can implicitly reward longer responses. For example, Park et al. [3] demonstrate that the DPO loss function can favor longer outputs due to the structure of the optimization objective, while Singhal et al. [10] show that reward signals learned in RLHF can strongly correlate with response length. In addition, evaluation methods themselves may reinforce this bias: LLM-based judges have been shown to prefer more verbose responses even when their quality is comparable [4].

Verbosity has also been studied as an explicit evaluation dimension for aligned language models. For example, YapBench [12] introduces a benchmark designed to measure how concisely models respond to prompts, highlighting that many instruction-tuned models produce unnecessarily long outputs even when shorter answers would suffice. However, YapBench focuses specifically on prompts designed to evaluate verbosity behavior rather than general instruction-following performance. In this work, we instead evaluate models using AlpacaEval 2.0, which measures overall response quality while controlling for verbosity through its length-controlled win rate (LC-WR) metric [9].

Prior work has primarily attempted to mitigate verbosity bias through modifications to preference optimization objectives. For example, LD-DPO [5], R-DPO [3], and related methods adjust the DPO objective to reduce the influence of response length during training [13, 14, 15]. Other approaches attempt to normalize token probabilities to remove length dependence. For example, SimPO [6]

proposes a simplified preference optimization method that removes the reference model used in DPO and derives rewards directly from the model’s own token probabilities. SimPO evaluates responses using average token log-probabilities rather than summed probabilities, which reduces the tendency of likelihood-based objectives to favor longer responses.

While these methods modify the optimization objective to mitigate verbosity, relatively little work has examined whether verbosity bias can be addressed through the design of the preference dataset itself, which has been shown to similarly exhibit verbosity bias. In contrast to these algorithm-centric methods, we explore a data-centric approach to verbosity control. Rather than modifying the training objective, we investigate whether the structure of the preference dataset itself can influence verbosity. Specifically, we construct a verbosity-disentangled preference dataset in which candidate responses differ primarily in length while maintaining comparable correctness, and evaluate how training on such data affects model conciseness and answer quality.

4 Approach

Our approach tests a data-centric hypothesis: verbosity bias can be mitigated by constructing preference data that explicitly disentangles length from quality, without modifying the DPO algorithm itself. Rather than altering the training objective as in prior length-aware DPO variants [3, 5], we generate verbosity-controlled preference pairs using a teacher model, pairing concise but complete responses as chosen against verbose or too-short responses as rejected. We train standard DPO on this dataset and evaluate against vanilla DPO trained on UltraFeedback. We additionally experiment with and without an SFT initialization to test whether adapting the model’s output distribution prior to preference optimization improves or interferes with verbosity-specific training. In all conditions, we measure length-controlled win rate on AlpacaEval to assess whether the model produces higher-quality responses without simply becoming shorter.

4.1 Constructing Disentangled Dataset

We construct a verbosity-disentangled preference dataset using 10,000 prompts sampled equally from UltraFeedback [7] and HelpSteer2 [8], split 90/5/5 into train, validation, and test sets (9,000 / 500 / 500 prompts). We sample equally from both sources to balance UltraFeedback’s diversity of instruction types with HelpSteer2’s conversational and practical prompts, improving distributional coverage.

For each prompt, we use style-conditioned prompting with a Kimi teacher model (moonshot-v1-8k) to generate three response styles: concise (correct and complete, but without unnecessary elaboration), verbose (correct but unnecessarily detailed), and too-short (incomplete or insufficient). Crucially, the concise and verbose conditions are both correct, which ensures our design isolates length as the axis of preference rather than conflating verbosity with quality, which is the key distinction from standard preference datasets such as UltraFeedback. We include too-short responses as an additional rejected condition to ensure the model does not over-correct toward minimal responses at the expense of completeness.

Candidates are filtered using an LLM-as-judge (GPT-4o-mini) to remove truncations, refusals, and responses that fail to satisfy their target style. In particular, concise and verbose responses must both be factually correct and substantively complete, differing only in length. Of 26,979 generated training candidates, 8,093 are dropped due to truncation (229), refusals (506), or judge-assessed incorrectness (3,531 concise, 3,827 verbose), leaving 18,886 remaining (5,261 concise, 4,921 verbose, 8,704 too-short). For validation, 1,178 of 1,497 generated candidates remain (361 concise, 331 verbose, 486 too-short), with 319 dropped. From the filtered candidates, we construct preference pairs (concise, verbose) and (concise, too-short), yielding 8,734 training pairs and 640 validation pairs.

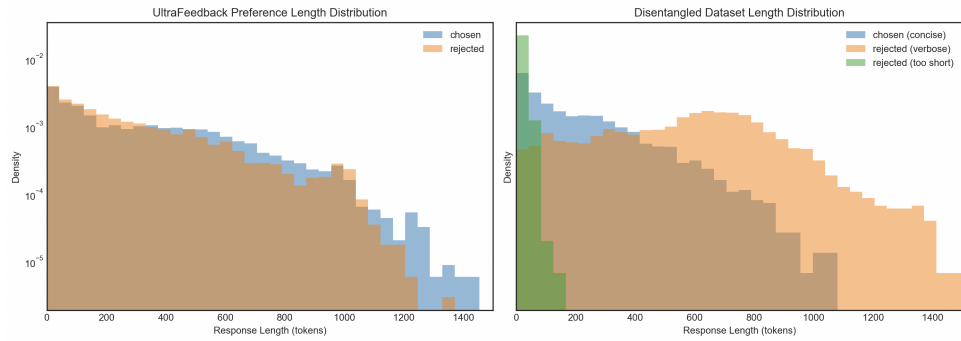


Figure 1: Illustration of chosen vs. rejected responses in UltraFeedback vs. our verbosity-disentangled dataset.

4.2 SFT

To construct the supervised fine-tuning dataset, we extract prompt–response pairs from our disentangled preference dataset by pairing each prompt with its concise chosen response, de-duplicating so that each prompt appears only once. We perform SFT on Mistral-7B-Instruct-v0.2 [16] using this data via QLoRA (4-bit NF4, LoRA rank 16, $\alpha = 32$, learning rate 5×10^{-5}). This produces an SFT adapter that we use as an initialization for a subset of DPO experiments, allowing us to compare DPO with and without prior instruction tuning. Due to compute constraints, we did not perform extensive hyperparameter search and instead used commonly adopted settings from prior QLoRA fine-tuning work.

4.3 DPO

We apply DPO to train the model (with and without SFT) to prefer concise, complete responses over verbose or too-short ones.

Our primary condition uses our verbosity-disentangled preference dataset, where preferred responses are concise and correct, and rejected responses are either verbose or too short. As a comparison, we also train standard DPO on the UltraFeedback binarized dataset [7], which reflects typical alignment practice without verbosity-specific supervision. All DPO runs share identical hyperparameters ($\beta = 0.1$, learning rate 10^{-5} , LoRA rank 16, $\alpha = 32$) to isolate the effect of dataset differences. Similarly, we did not conduct a broad hyperparameter sweep and instead used standard configurations commonly used in prior DPO implementations.

4.4 Baselines

We compare against three baselines. The first is the raw Mistral-7B-Instruct-v0.2 model without any fine-tuning, which reflects the verbosity behavior of a standard instruction-tuned model. The second is DPO trained on the UltraFeedback binarized dataset [7], which reflects standard alignment practice where preference pairs are constructed based on overall quality rather than verbosity-specific supervision. This comparison isolates the effect of our dataset design from the effect of DPO training itself. Finally, we include SimPO [6] as a baseline to compare against a different preference optimization method which uses a length-normalized score that can reduce the accumulation effects that favor longer responses. For SimPO, we use the same data as for our second baseline.

5 Experiments

5.1 Data

Our verbosity-disentangled preference data is constructed by sampling from UltraFeedback and HelpSteer2. We constructed a prompt pool by sampling from UltraFeedback and HelpSteer2, which provide diverse, alignment-relevant instruction prompts across a broad range of tasks. UltraFeedback contributes prompts commonly used in preference-learning research, while HelpSteer2 offers

complementary conversational and practical instructions, improving distributional coverage. We use these datasets solely as prompt sources to generate candidates varying in verbosity and to construct preference pairs that disentangle verbosity from correctness (method described in 4.1).

For our baseline (vanilla dataset), we use the UltraFeedback Binarized dataset [17], a standard preference dataset in which chosen and rejected responses are determined by overall quality ratings, with no control over response length or verbosity. We sample 8,000 training pairs and 640 validation pairs from the `train_prefs` split, shuffled with a fixed seed, matching the scale of our disentangled dataset for a fair comparison.

5.2 Evaluation method

We evaluate all models on AlpacaEval 2.0 [9], a standard benchmark of 805 prompts with pre-generated GPT-4 Turbo reference responses. For each prompt, an automated judge compares our model’s output against the GPT-4 Turbo reference and decides which it prefers. Win rate (WR) is the fraction of prompts where the judge preferred our model over GPT-4 Turbo. We use GPT-4o-mini as the judge rather than the default GPT-4 to reduce API costs. Since we are comparing relative differences between our own model variants rather than submitting to the public leaderboard, judge consistency across runs matters more than absolute calibration with standard AlpacaEval scores.

AlpacaEval 2.0 is particularly well-suited for our setting because it provides both raw win rate and length-controlled win rate (LC-WR), which statistically adjusts for response length to reduce the tendency of LLM judges to favor longer responses regardless of quality. Since our central goal is to reduce verbosity without sacrificing quality, LC-WR serves as our primary metric: a model that simply becomes shorter but loses quality will see its raw win rate drop, while a model that becomes more concise without quality loss will maintain or improve its LC-WR. We additionally report average response length in characters to directly track verbosity across conditions.

5.3 Experimental details

All experiments use Mistral-7B-Instruct-v0.2 [16] as the base model with 4-bit NF4 quantization (double quantization enabled, compute dtype bfloat16). Parameter-efficient fine-tuning uses LoRA with rank $r = 16$, $\alpha = 32$, dropout 0.05, applied to all attention and MLP projection layers (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`).

SFT is trained on our disentangled dataset for 1000 steps with learning rate 5×10^{-5} , effective batch size 16 (per-device batch size 1, gradient accumulation 16), and maximum sequence length 1024. DPO uses identical LoRA settings with $\beta = 0.1$, learning rate 10^{-5} , maximum sequence length 1024, maximum prompt length 512, and 1000 training steps. All training is run on a single NVIDIA H100 GPU via Modal. After training, each condition is evaluated with AlpacaEval.

We evaluate seven conditions:

Instruct (baseline) We use the raw Mistral-7B-Instruct-v0.2 model without fine-tuning to establish the baseline length–quality tradeoff.

Instruct + SimPO We train SimPO on Mistral-7B-Instruct-v0.2 using QLoRA (rank 16, $\alpha = 32$) on 8,000 UltraFeedback Binarized training pairs and 640 validation pairs, with $\beta = 2.5$, $\gamma = 0.25$, and learning rate 5×10^{-7} .

Instruct + SFT We fine-tune on our disentangled dataset for 1000 steps with learning rate 5×10^{-5} , warmup ratio 0.03, effective batch size 16 (per-device batch size 1, gradient accumulation 16), and maximum sequence length 1024. This establishes the effect of additional instruction tuning without any preference optimization.

Instruct + Vanilla DPO We applied DPO directly to Mistral-7B-Instruct-v0.2 with no SFT, trained for 1000 steps on 8,000 UltraFeedback Binarized training pairs and 640 validation pairs ($\beta = 0.1$, learning rate 10^{-5} , warmup ratio 0.03, effective batch size 16, max sequence length 1024, max prompt length 512, bf16 precision). This is the standard alignment baseline against which we measure the effect of our dataset design.

Instruct + Disentangled DPO We used the same DPO setup as above, but trained on our verbosity-disentangled dataset of 8,734 preference pairs, where concise responses are chosen and verbose or too-short responses are rejected.

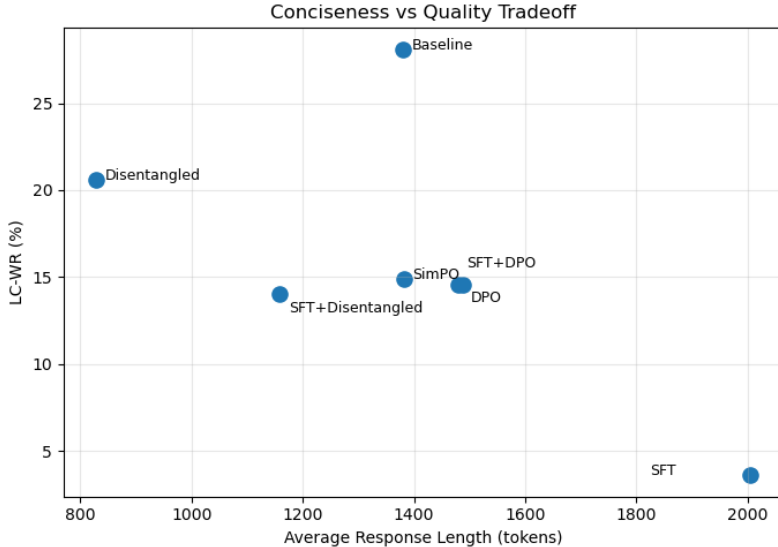
Instruct + SFT + Vanilla DPO The SFT adapter is loaded as initialization before applying vanilla DPO with identical hyperparameters, testing whether an SFT initialization improves standard preference optimization.

Instruct + SFT + Disentangled DPO The SFT adapter is loaded as initialization before applying disentangled DPO with identical hyperparameters, combining instruction tuning with verbosity-specific preference optimization.

5.4 Results

Model	WR (%)	LC-WR (%)	Avg. Length
Mistral-7B-Instruct (baseline)	20.0	28.1	1381
SFT	5.2	3.6	2004
SimPO (Vanilla)	20.77	14.91	1382
Vanilla DPO	11.80	14.54	1488
SFT + Vanilla DPO	11.68	14.55	1480
Disentangled DPO	11.4	20.6	829
SFT + Disentangled DPO	11.4	14	1159

Table 1: AlpacaEval 2.0 results. WR = win rate, LC-WR = length-controlled win rate. All models are based on Mistral-7B-Instruct-v0.2.



5.5 Human Evaluation

To verify that our automated evaluation reflects human preferences, we conduct a small-scale human evaluation on 30 randomly sampled prompts from our AlpacaEval on the Instruct + Disentangled DPO experiment. The goal of this evaluation is to assess agreement between human judgments and the automated judge rather than to benchmark our model against GPT-4. For each prompt, we present the responses from our model and the GPT-4 reference in randomized order and select the response we prefer.

Humans preferred the GPT-4 reference in 80% of cases and our model in 20% of cases. Agreement between human judgments and the automated LLM judge was 76.7%, suggesting that the automated evaluation broadly aligns with human preferences.

Model	Wins	Percent
GPT-4 Reference	24	80.0%
Disentangled DPO (ours)	6	20.0%

Human vs LLM judge agreement: 23 / 30 (76.7%). A sample of our human evaluation can be found in the appendix A.

6 Analysis

We evaluate whether verbosity bias in preference optimization can be mitigated through dataset construction. Specifically, we examine whether training on verbosity-disentangled preference pairs produces more concise responses without sacrificing answer quality.

6.1 Vanilla vs Disentangled DPO

Our comparison between vanilla DPO and our disentangled preference dataset shows that dataset construction has a strong effect on model verbosity. The vanilla DPO model produces responses with an average length of 1488 characters, while the disentangled DPO model produces substantially shorter responses averaging 829 characters, a reduction of roughly 44%. Despite this large reduction in verbosity, the two models achieve nearly identical raw win rates on AlpacaEval (11.8% for vanilla DPO vs. 11.4% for disentangled DPO). When evaluated using length-controlled win rate (LC-WR), the disentangled model achieves a substantially higher score (20.6% vs. 14.54%). These results suggest that training on verbosity-disentangled preference data can significantly reduce response length while maintaining competitive answer quality, supporting the hypothesis that verbosity bias in preference optimization can be mitigated through dataset design rather than modifications to the training objective.

6.2 Effects of SFT

Instruct + SFT + Disentangled DPO underperforms Instruct + Disentangled DPO on both LC-WR (14.0% vs. 20.6%) and average response length (1159 vs. 829). The SFT-only condition provides insight into this behavior: it produces the longest responses among all models (2004 characters) and achieves the lowest win rates (WR 5.2%, LC-WR 3.6%). Although the SFT dataset contains only concise responses, we hypothesize that its relatively small size (5k examples) appears insufficient to override the potential verbosity bias of the base instruction-tuned model. As a result, SFT primarily reinforces general helpfulness and elaboration behaviors rather than enforcing conciseness. When disentangled DPO is applied on top of this initialization, the conciseness preference must first counteract this verbosity bias, reducing its effectiveness. These results suggest that preference optimization is more effective than SFT for enforcing conciseness in this setting.

6.3 Comparison with SimPO

SimPO achieves the highest raw WR (20.77%) but a lower LC-WR (14.91%) than disentangled DPO (20.6%), and maintains an average response length nearly identical to the baseline (1382 vs. 1381). This suggests SimPO preserves overall response quality but does not meaningfully reduce verbosity; its length-normalized reward reduces the optimization bias toward longer responses during training, but does not actively discourage verbosity in the output distribution. By contrast, disentangled DPO produces substantially shorter responses with higher LC-WR, indicating that explicit preference pairs contrasting concise and verbose responses provide a stronger verbosity reduction signal than objective-level length normalization alone.

6.4 Conciseness-quality tradeoff

Taken together, these results suggest that verbosity bias in aligned language models is influenced by both the optimization objective and the structure of the preference data. Prior work has focused almost entirely on the former, yet our results show that intervening at the data level alone produces meaningful verbosity reduction (a 44% decrease in average response length with no raw win rate degradation) without any modification to the training algorithm. At the same time, the gap between

our best fine-tuned model and our baseline on LC-WR suggests that data-centric intervention alone is not fully sufficient at this scale, and that combining verbosity-disentangled data with algorithmic improvements may be necessary to close the quality gap. The poor performance of the SFT-only condition further suggests that the initialization distribution matters: when the model’s initialized distribution favors verbosity, as suggested by the SFT-only results, preference optimization appears to have less room to enforce conciseness.

However, we also acknowledge that all fine-tuned conditions underperform compared to the Mistral-7B-Instruct baseline on LC-WR (28.1%), despite the baseline producing longer responses on average. This suggests that fine-tuning on a smaller dataset might introduce a quality regression that partially offsets the gains in mitigating verbosity bias.

The gap highlights an inherent tension in conciseness alignment: explicitly optimizing for brevity with limited data may come at the cost of the broader instruction-following behaviors that LC-WR rewards. We believe that scaling the disentangled dataset further or mixing verbosity-disentangled pairs with general-quality preference data could be good directions for closing this gap.

7 Conclusion

In this work, we set out to test the hypothesis that verbosity bias in DPO can be addressed through data design rather than algorithmic modification. Our results offer support for this view. Training on verbosity-disentangled preference pairs produces a 44% reduction in average response length with no degradation in raw win rate and a meaningful improvement in LC-WR relative to vanilla DPO. However, all fine-tuned models remain below the untuned baseline on LC-WR, suggesting that the data-centric intervention improves conciseness but does not fully preserve the base model’s overall instruction-following quality. We were also surprised that SFT initialization on concise responses seemed to hurt disentangled DPO rather than helping it, pointing to a tension between instruction-tuning objectives and conciseness-specific preference optimization that could warrant further investigation.

Limitations Our experiments are limited by the relatively small size of the disentangled dataset (~8k preference pairs), which may constrain the strength of the learned preference signal and likely contributes to the quality regression observed relative to the untuned baseline. Scaling the dataset is a natural first step toward closing this gap. In addition, both candidate generation and filtering rely on LLM-based pipelines (a teacher model for candidate generation and GPT-4o-mini as judge), which could introduce noise into the preference labels. Our evaluation relies primarily on AlpacaEval 2.0 with GPT-4o-mini as judge rather than the standard GPT-4 Turbo, which means our absolute scores are not directly comparable to published leaderboard results; we use LC-WR to mitigate length bias, but automated evaluation cannot fully substitute for large-scale human evaluation. Our human evaluation partially addresses this, but is limited by its small scale (30 prompts) and the fact that evaluators are the authors, which may introduce bias. Finally, all experiments use a single model architecture (Mistral-7B-Instruct-v0.2) with limited hyperparameter tuning due to compute constraints, our findings may not generalize to larger models, different base architectures, or different preference optimization algorithms.

Future work Future work could explore larger datasets and stronger filtering methods. More extensive human evaluation would also help better measure verbosity–quality tradeoffs. Finally, combining verbosity-disentangled datasets with algorithmic modifications to preference optimization objectives may further improve the conciseness–quality Pareto frontier.

Team contributions

Susan constructed the verbosity-disentangled dataset. Will implemented the SFT and DPO training pipelines. Both contributed equally to experiment runs and analysis.

References

- [1] Long Ouyang, Jeff 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, 2022.
- [2] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024.
 - [3] Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization, 2024.
 - [4] Itsuki Saito et al. Verbosity bias in preference labeling by large language models, 2023.
 - [5] Wei Liu, Yang Bai, Chengcheng Han, Rongxiang Weng, Jun Xu, Xuezhi Cao, Jingang Wang, and Xunliang Cai. Length desensitization in direct preference optimization, 2024.
 - [6] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward, 2024.
 - [7] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with scaled ai feedback, 2024.
 - [8] Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models, 2024.
 - [9] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators, 2025.
 - [10] Kunal Singhal et al. A long way to go: Investigating length correlations in rlhf, 2023.
 - [11] Shunyu Zhang et al. Format bias in reward modeling, 2024.
 - [12] Xinyi Wang, Tianyi Zhang, Yiming Chen, Yifan Li, and Maosong Sun. Yapbench: A benchmark for measuring response verbosity of large language models. *arXiv preprint arXiv:2406.10250*, 2024.
 - [13] Pan Lu et al. Eliminating biased length reliance of direct preference optimization, 2024.
 - [14] Jiacheng Zhao et al. Bias fitting: Mitigating length bias in reward models, 2025.
 - [15] Seungone Kim et al. Mitigating length bias in rlhf through a causal lens, 2025.
 - [16] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Roux, Arthur Mensch, Pierre-Jean Savary, Chris Bamford, Devendra Singh Chaplot, Diego Hernandez de las Casas, Florian Bressand, Gyorgy Lengyel, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
 - [17] 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.

A Human Evaluation Examples

Below we show several example prompts and responses from our human evaluation.

Example 1

Prompt: What kind of foods do they eat in Thailand

Response A (GPT-4 reference): Thai cuisine is renowned for its vibrant flavors, aromatic spices, and a delicate balance of sour, sweet, salty, and spicy elements. Here is a list of some common Thai dishes and food items that one might encounter in Thailand:

1. ****Pad Thai**** - Stir-fried rice noodles with eggs, fish sauce, tamarind juice, red chili pepper, plus a combination of bean sprouts, shrimp, chicken, or tofu, garnished with crushed peanuts and coriander.

2. ****Tom Yum Goong**** - A hot and sour soup with shrimp that features flavors of lemongrass, lime leaves, galangal, lime juice, fish sauce, and crushed chili peppers.
3. ****Som Tam**** (Green Papaya Salad) - A spicy salad made from shredded unripe papaya, often mixed with tomatoes, beans, chilies, lime juice, fish sauce, and peanuts.
4. ****Green Curry**** - Made from green curry paste, coconut milk, bamboo shoots, chicken, and Thai basil. Variants include other meats like beef or pork, and even fish balls or seafood.
5. ****Massaman Curry**** - A rich, relatively mild Thai curry that contains coconut milk, roasted peanuts, potatoes, bay leaves, sugar, cinnamon, and tamarind sauce with meat like chicken, beef, or lamb.
6. ****Thai Fried Rice**** - A version of fried rice that can include ingredients such as chicken, shrimp, eggs, onions, and tomatoes. It's flavored with Thai basil and fish sauce.
7. ****Mango with Sticky Rice**** (Khao Niew Mamuang) - A popular dessert made with glutinous rice, ripe mangoes, and coconut milk.
8. ****Satay**** - Grilled skewers of marinated meat, typically served with peanut sauce and a small side of cucumber salad.
9. ****Pad Kra Pao**** - Stir-fried meat (often minced pork or chicken) with Thai basil and chilies, often served over rice with a fried egg on top.
10. ****Larb**** - A spicy salad made from minced meat (chicken, pork, beef, or duck), mint, lime juice, fish sauce, and toasted ground rice.

These dishes reflect just a fraction of the diversity and complexity of Thai cuisine, which can vary significantly between regions. Northern Thai food, for example, has strong influences from neighboring Laos and is known for dishes like khao soi (a coconut curry noodle soup), while Southern Thai cuisine often features fresh seafood and more pronounced flavors from tamarind and turmeric.

Response B (Disentangled DPO): Thai cuisine is known for its bold flavors and use of herbs and spices. Some common Thai dishes and ingredients include:

1. Pad Thai: Stir-fried rice noodles with eggs, tofu, beansprouts, peanuts, and scallions.
2. Tom Yum Soup: A hot and sour soup with lemongrass, galangal, kaffir lime leaves, and shrimp or chicken.
3. Tom Kha Gai: A coconut milk-based soup with galangal, lemongrass, chicken, and mushrooms.
4. Green Curry: A spicy curry made with green chilies, coconut milk, and chicken or beef.
5. Massaman Curry: A mild curry with potatoes, peanuts, and beef or chicken.
6. Som Tam: Papaya salad with tomatoes, peanuts, chilies, and lime juice.
7. Pad See Ew: Stir-fried flat rice noodles with soy sauce, garlic, and vegetables.
8. Mango Sticky Rice: Sweet sticky rice served with ripe mango and coconut milk.
9. Fried Rice: Cooked with eggs, vegetables, and meat or seafood.
10. Thai Iced Tea: Sweetened tea with milk and ice.
11. Thai BBQ: Grilled meats, skewered and served with dipping sauces.
12. Fresh Fruit: Mangoes, pineapples, durian, and dragon fruit are popular fruits in Thailand.

Human preference: A

Example 2

Prompt: How far away is Sagittarius A*, the black hole in the center of the milky way galaxy, from Earth and can you please provide that distance in light years and parsecs? Can you please also compare that distance to the distance of the center of the Andromeda galaxy from Earth?

Response A (GPT-4 reference): Sagittarius A* (Sgr A*) is the supermassive black hole located at the center of our Milky Way galaxy. It is approximately 26,000 light-years away from Earth. In parsecs, this distance is about 7,956 parsecs (since 1 parsec is equivalent to 3.262 light-years).

The Andromeda Galaxy (M31) is the nearest spiral galaxy to the Milky Way and is on a collision course with our galaxy, with an expected merger occurring in about 4 billion years. The distance from Earth to the center of the Andromeda Galaxy is roughly 2.537 million light-years, which is about 778,000 parsecs.

To compare the two distances:

- Sagittarius A*: 26,000 light-years or 7,956 parsecs - Andromeda Galaxy: 2.537 million light-years or 778,000 parsecs

Therefore, the center of the Andromeda Galaxy is much farther away from Earth than the center of the Milky Way, by a factor of about 100.

Response B (Disentangled DPO): The distance to Sagittarius A* (Sgr A*) is approximately 26,000 light-years (8.0 kpc) from Earth. Here are the conversions you requested:

- 1 light-year $\approx 9.461 \times 10^{12}$ kilometers - 1 parsec $\approx 3.0857 \times 10^{16}$ meters - 1 parsec ≈ 3.26 light-years

So, - Distance to Sgr A* in light-years: 26,000 ly - Distance to Sgr A* in parsecs: 8.2×10^4 m or 25.8 parsecs

The distance from Earth to the center of the Andromeda Galaxy (M31) is approximately 2.537 million light-years (773 kpc).

- Distance to M31's center in light-years: 2,537,000 ly - Distance to M31's center in parsecs: 773 kpc or 2.537×10^9 m

Example 3

Prompt: Curate a Spotify playlist based on the vibe. Publish this playlist as a song list.

Vibe: coding on weekend

Response A (GPT-4 reference): Creating a playlist for coding on the weekend would typically include a mix of instrumental tracks, electronic music, and perhaps some lo-fi beats to help maintain focus and a relaxed atmosphere. While I can't actually publish a Spotify playlist, I can curate a list of songs that would fit this vibe. You can then search for these tracks on Spotify to create your own playlist.

Here's a list of songs that might resonate with your coding weekend vibe:

1. Tycho - "Awake" 2. Bonobo - "Kiara" 3. Emancipator - "Soon It Will Be Cold Enough to Build Fires" 4. RJD2 - "Ghostwriter" 5. Ratatat - "Loud Pipes" 6. Nujabes - "Feather" 7. ODESZA - "Kusanagi" 8. Four Tet - "Two Thousand and Seventeen" 9. Carbon Based Lifeforms - "Photosynthesis" 10. Jon Hopkins - "Open Eye Signal" 11. The Album Leaf - "Twentytwofourteen" 12. Thievery Corporation - "Lebanese Blonde" 13. Moby - "Porcelain" 14. Boards of Canada - "Dayvan Cowboy" 15. Aphex Twin - "Avril 14th" 16. Ulrich Schnauss - "Knuddelmaus" 17. Amon Tobin - "At the End of the Day" 18. Little People - "Start Shootin'" 19. El Ten Eleven - "My Only Swerving" 20. Explosions in the Sky - "Your Hand in Mine"

This mix offers a balance between ambient and downtempo tracks that can help you get into the zone without being too distracting. You can adjust the list to match your personal taste and the specific mood you're looking to create for your coding session. Remember to check for similar playlists on Spotify as well, as they can offer additional inspiration and a ready-to-go soundtrack for your weekend coding.

Response B (Disentangled DPO): Title: "Code Weekend Vibes"

1. "Work" by Rae Sremmurd 2. "Lose Yourself" by Eminem 3. "The Middle" by Zedd, Maren Morris, Grey 4. "I'm the One" by DJ Khaled ft. Justin Bieber, Quavo, Chance the Rapper, Lil Wayne 5. "Money Trees" by Kendrick Lamar 6. "Working on It" by 6lack 7. "Focus" by Halsey 8. "Gimme Shelter" by The Rolling Stones 9. "Thinking Out Loud" by Ed Sheeran 10. "Eye of the Tiger" by Survivor 11. "Can't Stop" by Red Hot Chili Peppers 12. "Pumped Up Kicks" by Foster the People 13. "I Gotta Feeling" by The Black Eyed Peas 14. "Good Vibes" by Chris Janson 15. "Stronger" by Kanye West.

Human preference: B