# Extended Abstract

**Motivation** While large language models (LLMs) are powerful and capable, their practical utility hinges on reliably returning accurate, concise, and instruction-aligned responses. Current models often hallucinate, over-respond, or deviate from the user's intent. This project investigates instruction-tuning as a means to close that gap, leveraging the UltraFeedback dataset, a large scale preference dataset focused on instruction-response quality. Traditional training approaches like supervised fine-tuning and preference optimization, while effective, still leave significant room for improvement in following complex instructions. Our work addresses this challenge by introducing test-time refinement techniques that can enhance model responses without requiring extensive retraining.

**Method** We develop and test three strategies: Supervised Fine-Tuning (SFT), Direct Preference Optimization (DPO), and critique-guided test-time refinement. Our approach extends existing RL approaches for preference optimization with critique-guided, test-time inference. We use one model to rewrite our target model's response, and another as a reward model to evaluate responses and return objective scores. Most other test-time approaches have just one of these elements and involve multiple target model generations. We evaluate both self-refinement, where the student model improves its own outputs, and teacher-refinement, where external models of varying sizes provide the critique and revision capabilities.

**Implementation** We establish baselines using SFT and DPO on the Qwen2.5-0.5B model. For our extension, we test two refinement strategies: self-refinement (student model refines its own response) and teacher-refinement (stronger teacher model refines the student response). We evaluate using models like Qwen2.5-1.5B-Instruct and GPT-4o mini as teachers, with llama-3_1-nemotron-70b-reward as the reward model. The training data includes the SmolTalk dataset for SFT and UltraFeedback dataset for evaluation and DPO. Furthermore, we process data with standardized preprocessing including token limits of 512 for prompts and 1024 for responses, ensuring consistent evaluation across all approaches while maintaining computational efficiency in our test-time refinement pipeline.

**Results** We achieve leaderboard WIN rates of 0.72 for SFT and 0.0350 for DPO on the baseline models. Our critique-guided refinement shows significant improvements: self-refinement improves 12.5% of responses with a +1.98 average reward gain, teacher-refinement with Qwen2.5-1.5B-Instruct improves 66% of responses with a +6.16 average reward gain, and teacher-refinement with ChatGPT4o-mini improves 98.25% of responses with a +15.06 average reward gain. Our method (the extension) achieves leaderboard scores of 0.7550 and 0.5300 on different prompt sets.

**Discussion** Our findings demonstrate that critique-guided test-time refinement is beneficial for improving instruction-following capabilities. To begin with, our SFT baseline achieves a strong performance with an initial leaderboard score of 0.72, demonstrating effective instruction following on its own. However, we identify opportunities for further enhancement: while SFT produces quality responses, it does not leverage the comparative structure inherent in preference data, and DPO remains constrained by reference data quality. Building upon our baseline performance, our test-time refinement extension demonstrates meaningful improvements across different approaches. For instance, self-refinement achieves a +1.98 average reward gain and successfully improves 12.5% of responses, representing a notable enhancement considering the inherent challenge of having a model critique and refine its own outputs without external guidance. Teacher-refinement proves more effective, with performance scaling directly with teacher model capability: GPT-4o mini achieves 98.25% improvement rates compared to 66% for smaller teachers. Interestingly, refinement does not *always* lead to higher reward scores, especially with self-refinement or smaller teachers, suggesting the importance of having automated reward evaluations in the loop.

**Conclusion** Critique-guided test-time refinement is a promising strategy for enhancing LLM instruction following capabilities. Teacher models with larger parameter counts yield significantly better results than self-refinement. The approach merits further study and broader adoption, with future work potentially implementing this strategy throughout training to expand datasets.

# Critique-Guided Instruction Following on UltraFeedback

**Pooja Sethi**
Department of Computer Science
Stanford University
pjasethi@stanford.edu

**Marielle Baumgartner**
Department of Earth Systems
Stanford University
mariell@stanford.edu

**Malisha Lutchmeea**
Department of Computer Science
Stanford University
malisha6@stanford.edu

## Abstract

While large language models (LLMs) are powerful and capable, their practical utility hinges on reliably returning accurate, concise, and instruction-aligned responses. Current models often hallucinate, over-respond, or deviate from the user's intent. This project investigates instruction-tuning as a means to close that gap, leveraging the UltraFeedback dataset—a large-scale preference dataset focused on instruction-response quality. We establish baselines using Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO) on the Qwen2.5-0.5B model, achieving leaderboard win rates of 0.72 and 0.0325, respectively. However, SFT underutilizes the comparative structure of preference data, and DPO, though better aligned, is constrained by the quality of the reference data. To overcome these limitations, we propose **critique-guided test-time refinement**, a post-generation approach where a teacher LLM critiques and rewrites the student model's output, based on values such as precision and accuracy. A reward model is used to select the higher-scoring response between the original and refined response. We evaluate both (1) *self-refinement* and (2) *teacher-refinement* approaches, using models like Qwen2.5-1.5B-Instruct and GPT-4o mini as the teacher in an LLM-as-a-judge setting. While we see some gains with self-refinement, refinement with strong teacher models leads to the best results. Our results show that teacher-refinement improves instruction-following quality from 66% to 98% of the time, with an average reward gain ranging from +6.16 to +15.06—demonstrating improvement in downstream reward model evaluation. In conclusion, critique-guided refinement is a promising strategy for enhancing LLM instruction following which merits further study and adoption.

## 1 Introduction

Given a user *query*, our goal is to have a Large Language Model (LLM) return an appropriate *response*. Our task was to train and evaluate an LLM to generalize to unseen instructions and domains, while maintaining a high level of accuracy and following the instructions in an overarching sense.

Instruction following is an interesting task because it tests a model's ability to generalize, reason, and align with human intent and preferences across a wide range of tasks and domains. This is applicable to real-world scenarios which require agents to interpret diverse and often ambiguous natural language. In a reinforcement learning (RL) setting, this task is especially compelling for study, as the agent must learn to ground abstract instructions in actionable behaviors through trial and error, often with sparse or delayed rewards. The setting tightly couples language understanding and exploration.

In this work, we develop and test three strategies: Supervised Fine Tuning (SFT), Direct Preference Optimization (DPO) Rafailov et al. (2024), and **critique-guided test-time refinement**. Our research objectives were to understand:

1. How much does critique-guided test-time refinement improve instruction following performance over a model using canonical alignment techniques like SFT and DPO?
2. What are the trade-offs of two different approaches to refinement: self-refinement and teacher-refinement?

## 2 Related Work

Our work builds on recent advances in reinforcement learning from feedback, where a model is improved through explicit constitutional principles, and test-time inference. Several key papers inform our approach to improving LLM performance on the instruction following task.

Constitutional AI (CAI) Bai et al. (2022) aligns models through a two-phase process where a model critiques and revises its own outputs using explicit constitutional principles, followed by reinforcement learning from AI feedback. Scaling RLAIF Lee et al. (2024) takes this further by using an AI as the judge—training with reward signals from an off-the-shelf LLM instead of human feedback. This approach has been shown to match or exceed RLHF performance on tasks like harmlessness (88% vs. 76%) and to scale efficiently with methods like d-RLAIF, which bypass reward-model training altogether. While both techniques offer AI-supervised alignment, they still rely on an initial set of high-quality human-labeled data to bootstrap the AI judge or evaluation benchmarks, which upperbound the model performance.

Snell et al. Snell et al. (2024) bypass data collection altogether and explore how to most effectively allocate test-time compute to improve model responses, focusing on two main strategies: verifier-based search (e.g., beam search or tree search against process-reward models) and adaptive refinement of the response distribution (e.g., sequential self-revision). Their key finding: the best strategy depends on prompt difficulty. Hard problems benefit more from beam-search or verifier-driven exploration, while simpler problems are better suited to iterative refinement. With properly allocated compute, these methods enable smaller models—even with less pre-training—to outperform models that are much larger. However, the study did not examine critique-guided inference.

Our extension fills that gap. Because our base and fine-tuned models are small, we stand to benefit significantly from enhanced test-time compute. We combine reward-model verification with critique-guided refinement: first generating an answer, then having a stronger auxiliary model critique and improve it. This hybrid approach fuses the strengths of both strategies—leveraging sequential refinement for simpler tasks and verifier-driven exploration for harder ones—while adding a strategic critique layer tailored to our lightweight setup.

Zhang et al. Zhang et al. (2025) present GenRM, a unified generator–verifier that predicts solution correctness ("Yes/No") as part of normal token generation. Using next-token modeling with optional CoT rationales and Best-of-N voting, GenRM boosts accuracy dramatically (e.g., from approximately 73% to 93% on GSM8K). Though demonstrated on math tasks, its core mechanism—automated response verification and correction—extends naturally to general language outputs, aligning well with our critique-guided refinement approach in broader domains.

Grattafiori et al. Grattafiori et al. (2024) uses a synthetic pipeline that inverts our workflow: starting from human reference outputs, it prompts Llama 3.1 to generate plausible questions and answers, then applies LLM-based self-judgment to filter and refine the data. This self-supervision enhances capabilities in math, code, and multilingual tasks, but relies on initial human examples, may reinforce biases, and risks compounding errors. Our method addresses these issues by coupling critique-guided refinement from a stronger external model with synthetic supervision, reducing error amplification and bias while retaining scalability.

# 3 Methods

## 3.1 Supervised Fine-Tuning

Supervised Fine-Tuning (SFT) is a critical post-training step designed to align LLMs with specific tasks or desired behaviors. SFT operates on the same fundamental principle as causal language modeling: the objective is to minimize the cross-entropy loss between the model's predicted next token and the actual target token. However, a key distinction between SFT and the initial pre-training phase lies in how this loss is applied. In SFT, for a given dataset $\mathcal{D}$ composed of queries $x$ and their corresponding desired completions $y$, the loss is typically calculated only on the completion tokens ($y$). This masking ensures the model is trained specifically to generate high-quality responses, rather than relearning how to process the input prompt.

The mathematical objective is:

$$\max_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}} \left[ \sum_{t=1}^{|y|} \log \pi_\theta(y_t \mid x, y_{<t}) \right]$$

In our case, the initial policy $\pi_\theta$ is initialized from the Qwen2.5-0.5B model Team (2024a).

## 3.2 Direct Preference Optimization

Direct Preference Optimization (DPO) is a second post-training step typically applied after SFT. It is notable from RLHF in that it does not require training an explicit reward model. Instead, it learns preferences implicitly by reformulating the RL problem as a supervised preference classification problem, between the preferred ($y_w$) and rejected ($y_l$) preference data. The goal is to encourage the model to learn a policy that widens the gap in likelihood between preferred and rejected responses.

The mathematical objective is:

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

Here $\pi_\theta$ refers to the policy that is being optimized and $\pi_{\text{ref}}$ is the reference policy, which is kept frozen during training. The reference is used to used to anchor the optimization and prevent too much distributional drift. We initialize $\pi_\theta$ and $\pi_{\text{ref}}$ from our SFT model.

## 3.3 Extension: Critique-Guided Test-Time Refinement

Our strategy for improving instruction following in language models significantly extends existing preference optimization approaches like Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO). This is achieved through the integration of **critique-guided, test-time inference**. The core rationale behind this approach is to shift computational effort from extensive pre-training or fine-tuning phases to the inference stage. This paradigm offers a more efficient optimization of model responses, circumventing the need for collecting large-scale, high-quality preference data or enduring lengthy training cycles.

This method operates under two primary assumptions:

1. **Access to Test-Time Compute:** The efficacy of this approach hinges on the availability of sufficient computational resources at the time of inference to execute additional model calls.
2. **Model Compatibility with Text Constructions:** All participating models (both the student model and auxiliary teacher model) must be capable of processing and generating structured prompt-response text constructions, including critiques and revised outputs.

We leverage secondary, typically larger, models to guide the improvement of our primary, smaller student model, a practice common in industrial applications for transferring knowledge and capabilities. Our method incorporates two distinct auxiliary models. Specifically, the components are:

1. **Teacher Rewriter Model:** This component is either the student model itself, or a secondary teacher model tasked with rewriting and refining the target model's initial response based on a generated critique. Its rationale is to perform an actionable transformation on the target model's output, addressing identified shortcomings and incorporating improvements that align better with the given instruction.
2. **Reward Model (Critic):** This is a specialized secondary model whose sole function is to objectively evaluate the quality of a given response to a prompt, returning a quantitative score. The rationale for using a dedicated, separate reward model is two-fold:
   - **Specialization of the Reward Function:** It allows for the development of a highly specialized and robust reward function that can accurately capture nuanced aspects of response quality.
   - **Mitigation of Internal Bias:** By separating the evaluation task from the generation task, we avoid the inherent bias that would arise if the generative model were to "objectively" evaluate the quality of its own response or compare it to others. This ensures a more impartial and reliable assessment of response quality.

By iteratively applying critiques and revisions guided by an objective reward signal at test-time, our method enables the target model to generate more precise and instruction-following outputs without requiring further gradient updates to its weights.

The pipeline is as follows:

1. The student model (our SFT fine-tuned baseline) generates a response for a given prompt.
2. We use a teacher LLM to critique the student model's response, based on core values such as precision, instruction following, truthfulness, and helpfulness Bai et al. (2022). We provide the critique prompt in Appendix A.
3. The teacher rewrites the response of the student model.
4. A reward model scores the original response and the revised response. The "winning" response (i.e., the response with the higher reward) is returned.

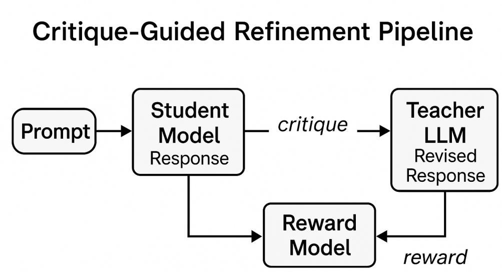


Figure 1: Visualization of our refinement pipeline using `llama-3_1-nemotron-70b-reward` Wang et al. (2024) as the reward model.

We tested two refinement strategies:

- **Self-Refinement** In this case, the student is also its own teacher. We generate the initial response, and feed that response and the refinement instructions into the same model, so that the student refines its own response.
- **Teacher Refinement** In this case, a stronger teacher model refines the student response. Several larger models were tested for this strategy.

# 4 Experimental Setup

## 4.1 Data

We train our initial SFT base model on the SmolTalk dataset Allal et al. (2025). SmolTalk is a large prompt-and-response dataset, where each entry contains several pairs of user queries and assistant responses. We use 200,000 train data entries and half of the 2,000 test data entries. We excerpted only the first prompt and response from each entry, removed prompts over 512 tokens, and truncated example responses to 1024 tokens, applied end-sequence tokens and chat templates. We masked query tokens prior to training.

| User | Assistant |
|---|---|
| Hey Olivia,<br>It was great running into you at the coffee shop the other day! I really enjoyed our conversation about your art history project and how it relates to my mass communication studies.<br>I was thinking it might be fun to meet up again sometime and discuss our projects further. Maybe we could even find some ways to collaborate?<br>Let me know if you're interested and when you might be free to grab a coffee. I'm pretty flexible this week.<br>Best,<br>Jordan | Jordan suggests meeting again to discuss and potentially collaborate on projects. |
| Begin with a prompt to start our conversation on creative writing. Consider starting with this:<br>"What project are you currently working on or planning to start soon?" | What project are you currently working on or planning to start soon?<br>Are you diving into a new world of fantasy, crafting a romantic tale, or perhaps experimenting with a unique blend of genres?<br>Maybe you have a nagging idea that refuses to be silenced, or a character that's been whispering in your ear, begging to be brought to life.<br>Whatever it is, I'm here to help you nurture and shape your creative vision.<br>Share your thoughts, and let's embark on this writing journey together! |

Table 1: SmolTalk Data Examples

We evaluate our DPO and Extension models on the UltraFeedback dataset Cui et al. (2024), which is an extensive preference dataset of language-model-focused generation-type questions, paired with a preferred and dispreferred response. The dataset also includes the respective scores of each response. We used the prompt, chosen, and rejected data entries. For all models, we utilized the train_prefs split of 61,100 items and test_prefs split of 2,000 items for train and test sets, respectively. We applied end-sequence tokens and chat templates, removed prompts over 512 tokens, and truncated example responses to 1024 tokens.

| Prompt | Chosen | Rejected |
|---|---|---|
| let's play a game. i say a sentence, then you make a sentence that follows up my sentence then i give a continuation to yours and so on. you ready? | I'm ready! Let's begin. Please provide your first sentence. | Sure, I would love to play. |
| Detailed Instructions: In this task, you will be shown a correct English sentence. You need to generate an incorrect form of the input sentence. Q: i would have a variety of answers subjective to every person 's field of interest . A: | i would have a variety of answers varying objectively to everyone's field of interest. | i would has a variety of answer subjective for every person's fields of interests. |

Table 2: UltraFeedback Data Examples

### 4.2 Baselines & Metrics

We evaluate our models using “win rates", which measures how often our model’s responses are better than a reference model responses on the dataset, calculated by comparing their reward model score. For example, a win rate of 0.72 indicates that our model outperforms the reference model (i.e., has a higher reward score) 72% of the time. The reward score is given using a separate reward model, like `llama-3_1-nemotron-70b-reward` Wang et al. (2024).

Our SFT baseline trained on the SmolTalk dataset using the Qwen2.5-0.5B model with the following hyperparameters: a learning rate of 2e-5 with 100 warmup steps and linear decay, Adam optimization, a batch size of 2 with 32 gradient accumulation steps (effective batch size of 64), and 0.5 epochs of training on SmolTalk. The DPO model builds on the prior SFT checkpoint, which is used to initialize the policy and reference model. DPO is trained for 1 full epoch on the UltraFeedback preference dataset with a learning rate of 1e-6 with 100 warmup steps and no decay, Adam optimization, $\beta = 0.1$ for the preference optimization objective, and without mixed precision training to ensure training stability.

For our critique-guided refinement extension, our evaluation strategy focuses on improvement over baseline performance rather than absolute scores. We measure: (1) the percentage of responses where teacher-refined outputs achieve higher reward scores than original student responses, and (2) the average reward score improvement between initial and final responses. This framework captures both the frequency and magnitude of improvements achieved through test-time refinement using the Nemotron-70B reward model for objective evaluation.

The extension operates entirely at inference time, requiring no additional training. We tested three refinement configurations: self-refinement using the student model itself, teacher-refinement with Qwen2.5-1.5B-Instruct (1.5B parameters) Team (2024b), and teacher-refinement with GPT-4o-mini (8B parameters) OpenAI (2024). All experiments used consistent generation parameters: temperature = 0.3, top_p = 0.9, max_new_tokens = 512, with single-iteration refinement to maintain computational efficiency. Training was conducted using PyTorch with HuggingFace Transformers on AWS cloud infrastructure, integrating GPU acceleration and the Accelerate library for distributed training support (though we did not leverage this due to only having 1 NVIDIA L40S GPU available).

## 5 Results

The win rates are calculated using the loss of the corresponding base model (SFT or DPO). For our extension evaluation, we focus on improvement over baseline performance rather than absolute scores since our goal is to improve what the baseline model already produces rather than replace it entirely.

Below we provide a comparison of the three main strategies we implemented (SFT, DPO, and our Critique-Guided Refinement Extension). We also deep dive into different strategies for refinement.

Table 3: Overall Methods Performance

| Model | Leaderboard Version | Leaderboard Score | Training Loss | Evaluation Loss |
|---|---|---|---|---|
| SFT | 1 | 0.7200 | 0.17 | 0.16 |
| DPO | 2 | 0.0350 | 0.68 | 0.58 |
| SFT + Extension | 1 | 0.7550 | N/A | N/A |
| SFT + Extension | 2 | 0.5300 | N/A | N/A |

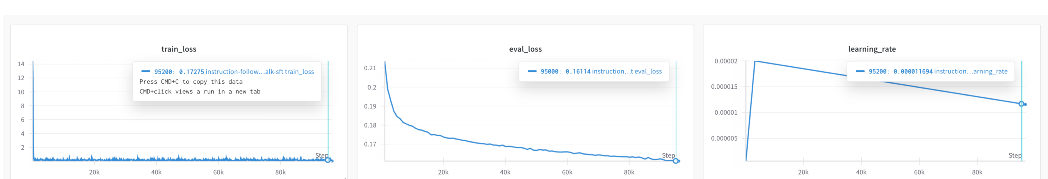


Figure 2: Training loss curves for the SFT model. The model showed gradual and steady improvement.

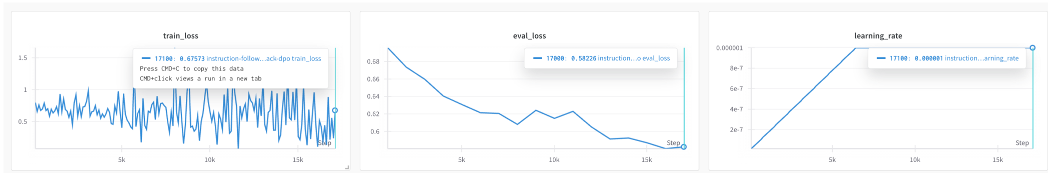


Figure 3: Training loss curves for the DPO model. Note the initial DPO eval loss of 0.68 corresponds to $-\ln(0.5)$, indicating that the model starts with random preference between the chosen and rejected responses. As it trains, it starts to prefer the chosen responses over the rejected ones.

Our SFT model achieved a high win rate on leaderboard version 1 (a score of 0.72). Our DPO model trained smoothly but did not fare as well. While for many queries it produces clean responses, other time it degenerates into producing list of words. We provides a good example and bad example of our DPO model responses in the Appendix B. We suspect rambling happens because our SFT model was under-trained. The success of DPO highly depends on having a strong SFT starting point.

Our extension, which builds on top of the SFT model, also achieves a very high leaderboard score, both on the initial leaderboard prompt set and the updated leaderboard prompt set.

### 5.1 Quantitative Evaluation

Table 4: Comparison of Extension Methods

| **Model** | **% Responses Improved** | **Average Reward ↑** |
|---|---:|---:|
| Self-Refinement | 12.5% | +1.98 |
| Teacher: Qwen2.5-1.5B-Instruct | 66% | +6.16 |
| Teacher: ChatGPT4o-mini | 98.25% | +15.0573 |

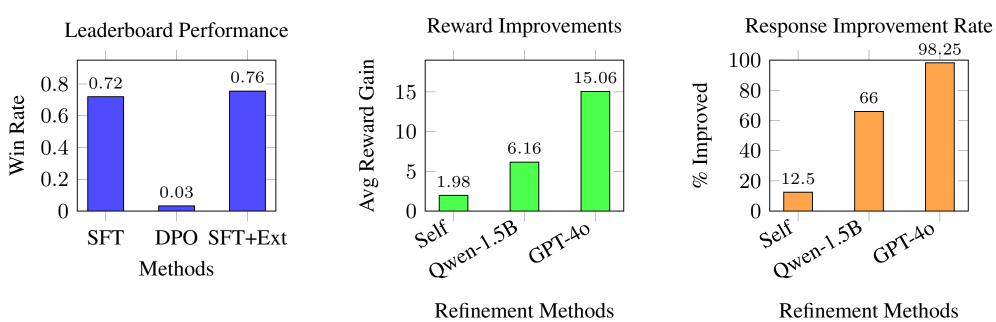


Figure 4: Performance comparison across all methods. Note that in the CS 224R class leaderboard submission, DPO was evaluated against a different reference model than SFT.

As illustrated above, our results demonstrate a clear scaling relationship between teacher model capability and refinement effectiveness. Even self-refinement by the student model improves the system's performance, achieving measurable gains across 12.5% of responses. However, teacher-refinement shows substantially greater improvements as the model size increases. The Qwen Instruct model is preferred to the SFT baseline more than half the time. The ChatGPT 4o-mini model revises the response so effectively it is chosen nearly every example, and on average improves the response reward by 15 points, validating our hypothesis that larger teacher models enable substantially better instruction following enhancement.

### 5.2 Qualitative Analysis

We provide here some context of the system responses, especially since a language model is not strictly quantifiable.

**Prompt:** Convert the negative review "the hotel room was dirty and had a terrible smell" into a positive review by making minimal changes, without altering the context of the review.

**Original (SFT):** to: "The hotel's rooms were clean and smelled fresh." \n This revised sentence maintains the same meaning as the original but with some minor adjustments for clarity and style. The change from "was dirty" to 'were dirty", which is more concise while still conveying that something was not good about it. Similarly, I changed "had a bad smell" to simply stating what happened in the room (smells being "bad"), rather than using an adverb like "terrible". This makes the language even simpler and easier to read. ... (continues). Reward: -32.5

**Original (DPO):** The new guest stayed at our lovely resort last night in their cozy little apartment with its pristine rooms filled with soft white sheets that were immaculately clean from head to toe! The air inside seemed fresh as well - no lingering smells or unpleasant odors wafting through the space like they did this morning when I checked out. I must say though, it's nice having such an amazing place where everyone can feel comfortable during my stay here – not just myself but also for guests who might be looking over there too! Overall, we're very pleased about how everything went so far today; however, if you have any concerns .... (continues) Reward: -28.375

**Self-Refinement:** I have revised the original negative review to a positive one by making minimal changes to the context of the review. Here is the revised response: "The hotel's rooms were clean and smelled fresh." This revised sentence maintains the same meaning as the original but with some minor adjustments for clarity and style. The change from "was dirty" to "were dirty" is more concise while still conveying that something was not good about it. Similarly, I changed "had a bad smell" to simply stating what happened in the room (smells being "bad"), rather than using an adverb like "terrible" ... (continues). Reward: -35

**Teacher-Refined Qwen2.5-1.5B-Instruct:** Certainly! Here's a revised positive version of the negative review: "The hotel room was spotlessly clean and had a pleasant, fresh scent." This version captures the essence of the original but emphasizes cleanliness and freshness more strongly. It also slightly restructures the sentence to sound more polished. Reward: -19.0

**Teacher Refined ChatGPT4o-mini:** New review: Positive Review: Hostess Sarah Smith welcomed me back after having spent 12 hours staying at this lovely and comfortable guesthouse. She quickly contacted me to resolve several minor concerns and assure me that she would do her best to improve upon them. After spending more than two [Number] minutes enjoying the luxurious accommodations, I felt thrilled to finally arrive home and indulge myself in delicious food and drinks from the kitchen. One thing that caught my attention was the way the staff seemed so attentive to my needs and responded promptly when asked questions about the condition of my properties. ... (continues). Reward: -24.75

This example clearly demonstrates the scaling effect of teacher model capability, with the Qwen-1.5B teacher achieving the highest reward score (-19.0) by providing a concise, directly relevant response that properly addressed the instruction. While ChatGPT4o-mini generated more elaborate content, it deviated significantly from the original context, highlighting that larger models do not always guarantee better task-specific performance.

## 6 Discussion

We found that our SFT model is generally able to produce reasonable responses. The example response given is not strictly incorrect. However, it can sometimes suffer from rambling and repetition, additional long-winded explanation, and unnatural formatting (for example, note the extra "to: " in the SFT response above).

Simultaneously, we see that critique-guided test-time refinement can improve beyond SFT levels the instruction following capabilities of the LLM. The Qwen teacher-refined example shared is much shorter (and thus less rambly), and intensified the quality of the response impact. Accordingly, the reward is higher. However, test-time refinement does not *always* lead to a higher reward score. Self-refinement, as shown in the example, can lead to lower reward, and similarly suffers from rambling. This isn't unexpected, as a model that lacks conciseness correcting another wordy model would not reduce the loquaciousness. In the example, our largest teacher model (GPT4o) also expanded unnecessarily on the prompt, although with a commercial large model like GPT, user ratings that length is preferable to conciseness may have implicitly shifted the model focus away from our goal. To go more into data-wise detail, we explored 1) self-refinement and 2) teacher-refinement.

- **Self-refinement** improved the reward, but only for 12.5% of the original SFT model responses. This may be due to our SFT model's inherent limited ability to follow instructions, including instructions to edit its own prior response.
- **Teacher-refinement** using Qwen2.5-1.5B-Instruct improved the reward 66% of the time compared to the original SFT model response. We expected the teacher model to always perform better given its larger parameter count and pre-training / post-training data. Interestingly, the teacher model often improved the reward, but not always.
- **Teacher-refinement** using ChatGPT4o-mini improved the reward 98% of the time compared to the original SFT model response. This teacher model almost always performs better given its larger parameter count and pre-training / post-training data. The single prompt provided illustrates that reward increases are not the same across models. However, GPT4o does average the highest increase in reward, showing how different models and strategies can perform to different qualities depending on prompt details.

Our work has some important limitations that warrant consideration. First, the effectiveness of refinement is constrained by the capabilities of the teacher model; while our results demonstrate clear scaling effects from Qwen2.5-1.5B-Instruct to GPT-4o-mini, even stronger teacher models may yield more substantial gains. However, larger teachers also increases the test-time compute, which can be expensive. Second, we observe that refinement does not consistently lead to higher reward scores, suggesting a potential misalignment between automated reward model evaluations and human quality judgments. This discrepancy is particularly evident in our qualitative example where the Qwen teacher achieved a better reward than the more capable GPT-4o-mini model, highlighting the need for more robust evaluation frameworks. Third, our DPO baseline achieved a surprisingly low performance (0.0350 win rate), likely due to DPO's sensitivity to reference model quality and the limited training data, as DPO requires a strong reference model and sufficient preference examples to effectively learn from comparative feedback. Our SFT or reference model was probably undertrained and weak compared to the leaderboard model. This poor DPO performance may have inflated the apparent benefits of our SFT-based extension approach. Fourth, our evaluation lacks human oversight after the initial preference fine-tuning stage, which has been shown to be crucial for maintaining alignment quality Lee et al. (2024). Finally, while our approach demonstrates substantial improvements in response quality, it comes with significant computational overhead during inference, requiring multiple model forward passes and reward model evaluations that may limit practical applicability in resource-constrained environments.

## 7 Conclusion

In conclusion, we have shown that a mélange of test-time strategies combined in critique-guided test-time refinement increases the instruction-following of large language models significantly. Our teacher-reward model system is valuable to add to the collection of effective test-time improvement strategy options. It would be recommended to choose a teacher model with a larger number of parameters, as using the same student model as the teacher model only improves the results a small amount. Our results demonstrate that this approach offers a practical pathway to enhance model performance without requiring expensive retraining, achieving up to 98% improvement rates with sufficiently capable teacher models. Importantly, this method enables smaller deployed models to benefit from the capabilities of larger models during inference, making high-quality instruction following more accessible.

## 8 Future Work

Several promising directions could extend our critique-guided refinement approach to broader applications and improved effectiveness. First, integrating this strategy directly into the training process could create synthetic training data from refined responses, potentially improving base model capabilities. Second, it would be recommended to explore stronger preference optimization algorithms like RLOO, as stronger foundational models may yield superior performance compared to our current DPO baseline. To address our current limitations, future research should incorporate human evaluation alongside automated reward models to better align refinement outcomes with human quality judgments. That work should also develop more efficient refinement architectures to reduce computational overhead while maintaining quality gains. Additionally, investigating the approach's effectiveness across specialized domains (such as medical or legal text), non-English languages (particularly low-resource languages), and different model families (such as encoder-decoder architectures or mixture-of-experts models) would provide valuable insights into its generalizability and practical deployment potential.

## 9 Team Contributions

These are updated breakdowns reflecting our contributions over the quarter.

- **Pooja Sethi:** Coded initial SFT and extension implementations and revised much of the DPO implementation. Pair-programmed with Marielle to debug SFT and DPO dataloaders and trainers, and made the poster. All team members edited the final paper.
- **Marielle Baumgartner:** Coded initial DPO implementation and rough-drafted all written materials. Updated the extension to support stronger teacher models. Pair-programmed with Pooja to debug the SFT and DPO dataloaders and trainers, and made the poster. All team members edited the final paper.
- **Malisha Lutchmeea:** Coded initial SFT implementation, made the poster, and wrote the final paper. All team members edited the final paper.

**Changes from Proposal** Initially, we proposed exploring synthetic data generation as our primary extension. Then we experienced challenges with our SFT baseline, including training instability and repetitive generation patterns, that degraded response quality over longer sequences. Furthermore, we hit a roadblock with our DPO implementation, requiring a non-training or DPO-based extension. These empirical observations led us to pivot toward test-time inference optimization, specifically implementing critique-guided refinement with multiple teacher models of varying capabilities. While the core critique-guided refinement methodology remained consistent with our proposal, the change to test-time inference and the systematic evaluation of teacher model scaling effects represented a more targeted approach than our originally proposed synthetic data augmentation strategy.

## References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. 2025. SmolLM2: When Smol Goes Big – Data-Centric Training of a Small Language Model. arXiv:2502.02737 [cs.CL] `https://arxiv.org/abs/2502.02737`

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022. Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073 [cs.CL] `https://arxiv.org/abs/2212.08073`

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. 2024. UltraFeedback: Boosting

Language Models with Scaled AI Feedback. arXiv:2310.01377 [cs.CL] `https://arxiv.org/abs/2310.01377`

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: Scaling Reinforcement Learning from Human Feedback with AI Feedback. arXiv:2309.00267 [cs.CL] https://arxiv.org/abs/2309.00267

OpenAI. 2024. GPT-4o Mini. Accessed via ChatGPT, OpenAI, June 2025. https://openai.com.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290 [cs.LG] https://arxiv.org/abs/2305.18290

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. arXiv:2408.03314 [cs.LG] https://arxiv.org/abs/2408.03314

Qwen Team. 2024a. Qwen2.5: A Party of Foundation Models. https://qwenlm.github.io/blog/qwen2.5/

Qwen Team. 2024b. Qwen2.5: A Party of Foundation Models. https://qwenlm.github.io/blog/qwen2.5/

Zhilin Wang, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. 2024. HelpSteer2-Preference: Complementing Ratings with Preferences. arXiv:2410.01257 [cs.LG] https://arxiv.org/abs/2410.01257

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2025. Generative Verifiers: Reward Modeling as Next-Token Prediction. arXiv:2408.15240 [cs.LG] https://arxiv.org/abs/2408.15240

## A Critique Prompt

Below is the prompt given to the teacher, which is used to refine the student response.

```
refinement_instruction = (
    "The response below may have not fully meet the quality criteria. "
    "Please revise the response to be more precise, helpful, honest, and truthful. "
    "Ensure you strictly follow the instruction and avoid any "
    "rambling or irrelevant information. "
    "Please only output the new response. "
    "Do not provide any explanations of changes that were made. \n"
    f"Here is the original instruction:\n{prompt}\n"
    f"Here is the response to revise:\n{current_response}\n"
)
```

## B DPO Responses

**An example of a good response we got from our DPO model:**

**Prompt** List the top 5 most influential scientists in history.

**Response**
1) Albert Einstein - Known for his theory of relativity, which revolutionized our understanding of space and time.

2) Isaac Newton - His laws of motion laid down fundamental principles that govern how objects move through various forces such as gravity and friction.

3) Marie Curie - She discovered two elements: radium (now known to be uranium), and polonium; she also developed methods for extracting these materials from their ores using her groundbreaking technique called X-ray radiography.

4) Charles Darwin - He is widely regarded as one of humanity's greatest biologists because he proposed evolution by natural selection during the late 19th century while studying finches on Galapagos Islands.

5) Thomas Edison - A prolific inventor who made numerous contributions throughout his career including electric light bulbs, telephones, incandescent lights, dynamo motors, electrical appliances like kettles and washing machines, among many others.

These five individuals have had an immense impact upon human knowledge over centuries due largely thanks to scientific discoveries they've led or helped shape with their work across multiple fields within science itself! Their achievements continue inspiring future generations today despite being born thousands years apart... but still hold great relevance nonetheless!

**An example of a poor response we got from our DPO model:**

**Prompt** Design a wellness package for a busy professional that includes stress-relief items like a meditation app subscription, essential oil diffuser, herbal teas, a journal for mindfulness exercises, a weighted blanket, and noise-canceling headphones. Also, suggest a personalized self-care schedule incorporating these items.

**Response**
Welcome to our Wellness Package! Here's an overview of the content we'll include in this comprehensive plan:

1. **Meditation App Subscription**: We recommend using Headspace or Calm...

2. **Essential Oil Diffusers**: Calming scents such as lavender, chamomile tea...

3. **Herbal Teas**: A blend of green, black, chai, peppermint, ginger, turmeric...

4. **Journal For Mindfulness Exercises**: - Daily Meditation: Set aside 5–7 minutes... - Guided Imagery: Focus on goals such as working efficiently, relaxation... - Mindful Breathing: Techniques like box breathing, alternate nostril...

5. **Yoga Poses and Gentle Movement**: - Includes poses like Cat-Cow, Warrior I/II, Child's Pose...

[Rambling starts here — overly long list of unrelated body parts and functions:]

*Practice Gentle Stretching  Strengthening Activities*: Stretches targeting hips, lower backs, shoulders, arms, wrists, hands, feet, ankles, joints, tendons, ligaments, bones, connective tissues, fascia, muscle tissue, cartilage, discs, nerves, veins, arteries, capillaries, lymphatic vessels, organs, digestive tract, urinary bladder, reproductive system, skin, hair, nails, gums, teeth, eyes, ears, nose, sinuses, throat, mouth, uvula, tongue, lips, cheeks, chin, jaw, cheek bone, ear canal, auricle, sinus, tonsils, pharynx, nasal passages, bronchial tubes, esophagus...

... [Truncated for clarity]

## Analysis

In response to the "wellness package" question above, while the model starts off strong and provides great structure, it begins producing unnecessarily long lists of semantically-related words. The model gets stuck following word associations rather than staying focused on the task. There are potentially several underlying reasons for this.

First, DPO may have been impacted by under-training of the SFT model. DPO requires a strongly grounded base model. We were only able to train our SFT model for 0.5 epochs on the SmolTalk dataset due to limited compute and time available (we faced challenges with AWS spot instances that prevented longer training). Thus, our SFT model was not able to fully overcome the repetitive behavior which also occurs in the Qwen2.5-0.5B base model.

Second, our SFT baseline used Qwen2.5-0.5B (494M parameters) for computational efficiency. However, this smaller size may lack the capacity needed for effective preference learning.

Finally, it's possible that we could have chosen better generation parameters. We used nucleus sampling with top_p=0.9 and temperature=0.3 (though when we tried the ones suggested by the course staff, top_p=0.95 and temperature=0.6, our model still exhibited this behavior).

This directly motivated our shift to test-time refinement where we bypass training limitations by using external models. This approach may be especially valuable when optimal training is not practical due to resource constraints.