# Distilling Reasoning Into Conversational Models Using Generated Data

Jack Younger, Mateo Quiros-Bloch, Carlos Santana
Department of Computer Science, Stanford University

June 10, 2025

## Abstract

Small conversational language models excel at fluent dialogue but often fail on multi-step reasoning tasks due to limited capacity and coarse tokenization. We investigate whether reasoning behaviors from a large model can be distilled into a compact 0.5 B-parameter model via synthetic preference data. Starting from the pre-trained Qwen-2.5-0.5B, we first establish a supervised fine-tuning (SFT) baseline on the SmolTalk dataset (approximately 100 K prompt–response pairs), achieving a 58.5 % win rate on the leaderboard and –27.9 average Nemotron reward on held-out reasoning prompts.

Next, we apply Direct Preference Optimization (DPO) on 60 K human-labeled UltraFeedback pairs. On our intial trial, DPO underfit and underperformed, plummeting to a 10 % win rate and –32.7 average reward. However, on the second attempt at DPO we achieved much better results, with a 60% win rate against our baseline SFT model and a higher average reward. For our extension we planned on using synthetic data generated from either a smaller set of prompts from the ultrafeedback set or prompts and responses generated based on the prompts our model performed the worst on. We generated 12 K synthetic preference triples using GPT-4o: 2 K via zero-shot prompting for a proof of concept and 10 K via few-shot prompting with four in-context examples drawn from UltraFeedback. We design strict JSON-output templates to enforce paired "preferred" and "dispreferred" (apologies for the poor spelling, it just made sense to us to use in the moment) responses, and sample prompts where the SFT model underperforms to maximize reasoning diversity.

We utilized the synthetic pairs as training on top of the original human data (total 31.8 K triples) and fine-tune via DPO for two epochs under identical optimizer settings. We evaluate on two held-out sets: 200 synthetic few-shot pairs and 2 000 UltraFeedback test prompts. Our initial synthetic DPO recovered performance, reaching a 56 % win rate and –27.1 average reward—beating the SFT baseline—and outperforming our initial attempt at the human-only DPO by a large margin. Preliminary few-shot results suggest that our attempt, while it showed initial promise, was lacking in diversity of examples and provided us with a model that was less robust to prompts it had not seen before.

Our analysis reveals that synthetic augmentation can correct some deficiencies in training on human generated data but faces limitations from narrow data diversity and training integration strategy. We propose intertwining synthetic and human examples during SFT and DPO, and scaling generation to broader prompt distributions. In particular, if we were to conduct further experimentation on using synthetic data we would have used different strategies and more robust testing to verify that our initial assumptions would prove effective. We believe that our initial principle of incorporating reasoning based data into the fine-tuning of LLMs shows promise, as evidenced by our initial smaller scale results, but that greater diversity of prompts and more clever augmentation of the synthetic data is necessary in order to obtain better results.

Shorter abstract: Small conversational models often struggle with multi-step reasoning despite fluent dialog, so we fine-tuned a compact 0.5 B-parameter Qwen-2.5 model on SmolTalk to achieve a 58.5% win rate (–27.9 Nemotron reward), then applied Direct Preference Optimization (DPO) on 60 K human-labeled UltraFeedback pairs. By augmenting DPO with 12 K synthetic "preferred"/"dispreferred" pairs from GPT-4o (2 K zero-shot, 10 K few-shot), we recovered and even surpassed baseline performance (56% win rate, –27.1 reward) while highlighting that narrow synthetic diversity limits robustness, especially when the diversity of the synthetic training data is limited. We finally discuss strategies that might have allowed us to utilize synthetic data in a more effective way, and explore the reasoning for why our approach did not succeed.

# 1 Introduction

Modern conversational language models are trained to predict the next token in a sequence, optimizing for fluency and surface-level coherence rather than explicit multi-step reasoning. They often rely on fixed-length input contexts and generate responses without internal chains of thought, which can lead to shallow or hallucinated outputs when faced with complex reasoning tasks [5]. Moreover, subword tokenization schemes such as BPE can merge atomic reasoning units—obscuring logical structure and fundamentally capping the symbolic and arithmetic performance of smaller models [7]. This means that, due to their limited token-granularity and smaller parameter budgets, compact models are inherently unable to represent and manipulate the fine-grained reasoning steps that larger, higher-capacity models can handle.

While large models can perform complex reasoning through multi-step internal processes, they incur high computational and latency costs, making them impractical for many real-world applications. At the same time, small, efficient models are desirable for deployment on edge devices and in low-resource settings, but they fall short on tasks requiring deep inference. If we can capture and transfer the decision boundaries and logical structure that large models learn—without retraining them end to end—we could combine the best of both worlds: the reasoning prowess of a powerful model and the efficiency of a compact one. Synthetic preference data generated by a reasoning-capable model offers a promising path for this kind of knowledge transfer.

We fine-tune the pre-trained Qwen-2.5-0.5B model first via supervised fine-tuning (SFT) on the SmolTalk dataset (yielding a 58.5% win rate, avg. reward –27.9), then apply Direct Preference Optimization (DPO) on UltraFeedback preferences (win rate dropped to 10%, avg. at first, but then reached 60% at our final attempt). By appending 2,000 GPT-4o–generated "preferred"/"dispreferred" pairs, we recover performance to a 56% win rate and –27.1 avg. reward, nearly matching the SFT baseline. Then, more examples of prompt response pairs were generated through different prompting strategies done to the large language models, using 0-shot and few-shot prompting with examples derived from the already-labeled data from UltraFeedback. We see that this was not as effective in training, most likely due to the lack of diversity in the prompts and responses that were provided as the synthetic data as a result of the way we generated the data.

# 2 Background and Related Work

Supervised fine-tuning adapts a pre-trained language model by continuing maximum likelihood training on a curated set of high-quality instruction–response pairs. This process directly teaches the model to mimic desired behaviors by minimizing the negative loglikelihood of the target tokens conditioned on the prompt and preceding tokens [5].

Early work demonstrated that SFT on carefully constructed datasets can significantly boost performance in downstream tasks without altering the model architecture or adding auxiliary objectives [3]. In addition, the efficacy of SFT is closely related to the size and diversity of its training corpus: Insufficient coverage can lead to overfitting in narrow domains and poor generalization to new instructions.

A widely adopted method for training language models involves reinforcement learning with human feedback (RLHF). In the standard RLHF pipeline, an initial policy is fine-tuned via supervised learning on demonstration data, candidate responses are generated, and human annotators provide pairwise preferences that are then used to train a separate reward model (e.g., via PPO), aligning model outputs with human judgments [1, 5]. However, collecting or generating sufficient human-labeled examples is both time-consuming and expensive, often requiring expert annotators, multiple calibration rounds to ensure consistency, and carefully designed annotation interfaces [5, 8].

Direct Preference Optimization (DPO) offers a streamlined alternative by eliminating the explicit reward model: it directly optimizes the log-odds of preferred over rejected responses under a Bradley–Terry formulation, using only the raw preference pairs [4]. Despite this simplification, DPO applied solely on offline human data can underfit, as the fixed dataset may lack the diversity and scale needed to capture nuanced user preferences [6].

An alternative approach, sometimes termed "RL with AI Feedback" (RLAIF), replaces the human annotator with a strong, pre-trained LLM to generate synthetic preference labels at scale. By prompting a reasoning-capable model to produce "preferred" and "rejected" responses for each input, one can cheaply and rapidly assemble large preference datasets that approximate human judgment [2, 3]. Empirical studies have shown that augmenting or even fully substituting human labels with high-quality synthetic data can recover much of the performance of standard RLHF and DPO pipelines while drastically reducing annotation overhead [2, 6].

In this project, we build on these insights by applying synthetic data augmentation specifically to DPO fine-tuning of the Qwen-2.5-0.5B model. We leverage off-the-shelf reasoning models to generate paired preference data tailored to reasoning-intensive prompts, thereby enriching the original UltraFeedback dataset. Our goal is to determine whether such synthetic augmentation can yield measurable gains in reasoning performance under resource constraints.

# 3 Methods

## 3.1 Base Model and Training Pipeline

**Model architecture.** Our base model is Qwen-2.5-0.5B, a decoder-only Transformer with approximately 0.5 billion parameters, 12 layers, a hidden dimension of 2 048, and 16 attention heads. It uses byte-pair encoding (BPE) with a 64 000-token vocabulary and supports up to 2 048 input tokens. Qwen-2.5-0.5B was pre-trained on a mixture of web text and instruction-style data, providing a strong initialization for both generative fluency and basic instruction following.

**Supervised fine-tuning (SFT).** We first adapt Qwen-2.5-0.5B to the SmolTalk instruction-response format by supervised fine-tuning. Given a dataset $\mathcal{D}_{\mathrm{SFT}} = \{(x^{(i)}, y^{(i)})\}$ of prompt–response pairs, we optimize

$$\max_{\theta} \; E_{(x,y)\sim\mathcal{D}_{\mathrm{SFT}}} \sum_{t=1}^{|y|} \log \pi_\theta\big(y_t \mid x, y_{<t}\big),$$

equivalently minimizing the cross-entropy loss over next-token predictions [5]. We train for 3 epochs with a peak learning rate of $1 \times 10^{-5}$ (linear warmup over 500 steps), batch size 16, and weight decay 0.01. Checkpoints are evaluated on a held-out subset for early stopping.

**Direct Preference Optimization (DPO).** Starting from the best SFT checkpoint, we fine-tune using Direct Preference Optimization on a dataset of preference triples $\mathcal{D}_{\mathrm{DPO}} = \{(x, y_w, y_\ell)\}$, where $y_w$ is the preferred response and $y_\ell$ the rejected one. DPO minimizes the loss

$$\mathcal{L}_{\mathrm{DPO}}(x, y_w, y_\ell) \;=\; -\log\Big[\sigma\big(s(x, y_w) - s(x, y_\ell)\big)\Big], \quad s(x, y) = \beta \, \log \frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)},$$

where $\pi_{\mathrm{ref}}$ is the fixed reference policy (the initial Qwen-2.5-0.5B) and $\beta = 1.0$ scales the log-odds [4]. We train for 2 epochs, with the same optimizer settings as SFT, and select the checkpoint with lowest validation DPO loss for downstream evaluation.

**Datasets used.**

- **SmolTalk (SFT).** A collection of ≈100 000 informal conversational prompts paired with high-quality human or model responses. It covers everyday topics (greetings, small talk, simple Q&A) and serves to teach basic conversational fluency.
- **UltraFeedback (DPO).** A proprietary preference dataset with ≈60 000 prompts, each annotated with one "chosen" and one "rejected" response by human raters. This dataset emphasizes helpfulness, correctness, and safety criteria.
- **DPO-Augmented.** We augment the original UltraFeedback with 2,000 synthetic preference pairs generated by GPT-4o in a zero-shot setting. These pairs are sampled uniformly from prompts where the SFT model underperformed, to maximize diversity in reasoning content. Using few-shot prompt strategies, we generated another 10,000 preference pairs in the same format.

## 3.2 Synthetic Data Generation

To enrich our preference dataset with reasoning-oriented examples, we generate synthetic “preferred” / “rejected” response pairs using both GPT-4o and a GPT-3.5–style reasoning model. Our pipeline consists of three stages:

**Prompt templates.** We design a JSON-output prompt that enforces strict formatting and distinguishes high-quality from low-quality responses. The zero-shot template is:

```
You are generating *paired* answers for RLHF preference data.
For the prompt I give you, output a JSON object with EXACTLY these two keys:
  "preferred"   { a helpful, correct, safe reply that clearly answers the prompt.
  "dispreferred" { a clearly worse reply: incomplete, shallow, or factually mistaken.
Do NOT use any other keys. Do NOT wrap answers in markdown.
```

For few-shot prompting, we prepend 4 in-context examples (prompt + preferred/dispreferred pairs) sampled from UltraFeedback, illustrating desirable gaps in quality. The few-shot template is:

```
"You are generating *paired* answers for RLHF preference data - please answer in both instances verbosely\
    n"
"For the prompt I give you, output ONLY a JSON object with EXACTLY these two keys:\n"
" ’preferred’  a helpful, correct, safe reply, that clearly answers the prompt and follows the
    instructions\n"
"examples of preferred responses and dispreffered (paired, so example 1 for preferred is to the same
    prompt as example 1 to dispreffered): \n"
"example 1: Of course, I’d be happy to help you optimize your vacation! Can you please provide me with
    some details about your trip, such as:\n* Destination(s)\n* Duration of the trip\n* Type of vacation (
    e.g. beach, city, outdoor, adventure)\n* Budget\n* Any specific interests or activities you would like
     to do during the trip\n\nPlease provide me with this information and I will do my best to assist you
    in planning a memorable and enjoyable trip!\n"
"example 2: As Romgor and [nick’s character] travelled across the vast lands, they faced many
challenges and adventures. They battled mighty beasts, explored ancient ruins..."
" ’dispreferred’  a clearly worse reply including an incomplete response, a simple
continuation of the prompt, or a factual error.\n"
"examples of dispreffered responses: \n"
"example 1: Sure, I’d be happy to help you optimize your vacation! What are you looking
to do during your trip? What’s your budget? And where would you like to go? \n"
"example 2: Greetings, traveler! I’m here to assist you in any way I can, as a helpful,
respectful, and honest assistant. I understand that you have a rich backstory for your character...
"Return ONLY a single JSON object with these two keys. Do NOT include any other text or
formatting outside of the JSON object."
```

**Model selection and generation.**

- **GPT-4o (0-shot):** We sample 2 000 prompts on which our SFT model underperforms and feed them to GPT-4o with the zero-shot template, generating one preference pair per prompt.
- **GPT-4o (few-shot):** We generate 10 000 prompts and response pairs with several example prompts that our model performed poorly on as context. We then generate responses to those prompts, again providing examples of chosen and rejected responses as context to generate the response pairs from the selected generated prompt.

**Zero-shot vs. few-shot strategies.** Zero-shot prompting is fast and requires no manual example curation, but may produce noisier preferences. Few-shot prompting uses a small demonstration set to guide the model toward consistent label gaps, at the cost of additional prompt engineering.

## 3.3 Distillation and Fine-Tuning

After generating a total of 12,000 synthetic preference pairs—2,000 via zero-shot prompting and 10 000 via few-shot prompting—we integrate them post training into an additional Direct Preference Optimization pipeline to transfer reasoning behaviors into the compact Qwen-2.5-0.5B model.

**Dataset composition.**

- **Zero-shot synthetic:** 2,000 pairs generated with the strict JSON template and no in-context examples.
- **Few-shot synthetic:** 10,000 pairs generated using 4 in-context demonstrations drawn from high-diversity UltraFeedback examples.
- **Human-labeled baseline:** 60,000 original UltraFeedback pairs.

We reserve 200 of the synthetic few-shot examples as an *internal evaluation set* to tune hyperparameters and monitor overfitting; the remaining 11,800 synthetic pairs plus all human-labeled data ( 20 000) form our DPO-Augmented training corpus (31,800 total examples).

**Fine-tuning procedure.** Starting from the SFT-trained checkpoint, we fine-tune via DPO on the combined training corpus. At each step:

- We sample uniformly across human and synthetic triples to construct mini-batches of size 16.
- We use the standard DPO loss

  $$\mathcal{L}_{\mathrm{DPO}}(x, y_w, y_\ell) = -\log\Big[\sigma\big(s(x, y_w) - s(x, y_\ell)\big)\Big], \quad s(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)},$$

  with $\beta = 1.0$, the same optimizer settings as SFT, and 2 epochs of training.

**Evaluation protocol.** We evaluate on two held-out sets:

1. **Synthetic hold-out:** 200 few-shot pairs reserved during generation.
2. **UltraFeedback test:** 2,000 human-labeled prompts unseen during training.

For each prompt, we issue a "response-given-text" move—i.e., we feed the prompt as context and sample a single response from the fine-tuned model. We then score each response against either:

- The corresponding held-out "preferred" label (synthetic hold-out), measuring accuracy of preference replication.
- A Nemotron 70B reward model comparison against a reference response (UltraFeedback test), computing:
  - *Win rate:* fraction of prompts where the fine-tuned model's reward exceeds that of the SFT baseline.
  - *Average reward:* mean Nemotron score across prompts.

We also submit our best checkpoints to the UltraFeedback public leaderboard, comparing DPO-Augmented performance against both the SFT baseline and human-only DPO under identical inference settings.

# 4 Mathematical Formulation

- **Direct Preference Optimization (DPO) loss.** DPO frames preference learning as a binary classification problem between a "winning" response $y_w$ and a "losing" response $y_\ell$ for the same prompt $x$. Concretely, we define a scoring function

  $$s(x, y) \;=\; \beta \, \log \frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)},$$

where $\pi_\theta$ is the current policy, $\pi_{\text{ref}}$ is the fixed reference policy (the original pre-trained model), and $\beta > 0$ is a temperature parameter that controls the sharpness of the preference margin. We then apply a logistic sigmoid to the score difference:

$$\sigma\big(s(x, y_w) - s(x, y_\ell)\big) \;=\; \frac{1}{1 + \exp\big[-\big(s(x, y_w) - s(x, y_\ell)\big)\big]}.$$

The DPO loss for a single triple $(x, y_w, y_\ell)$ is

$$\mathcal{L}_{\text{DPO}}(x, y_w, y_\ell) \;=\; -\log\Big[\sigma\big(s(x, y_w) - s(x, y_\ell)\big)\Big].$$

Minimizing this loss encourages the model to assign higher log-odds to preferred responses than to rejected ones. Because it operates directly on the policy logits, DPO avoids training an auxiliary reward model and tightly couples optimization to the end metric of human or synthetic preferences.

- **Supervised Fine-Tuning (SFT) objective.** SFT optimizes the standard maximum-likelihood objective on a dataset of $(x, y)$ pairs:

$$\max_\theta \; E_{(x,y)\sim\mathcal{D}} \sum_{t=1}^{|y|} \log \pi_\theta\big(y_t \mid x, y_{<t}\big).$$

Equivalently, we minimize the cross-entropy $\mathrm{CE}(y, \pi_\theta(\cdot \mid x))$ between the empirical token distribution and the model's predictive distribution. This objective drives the model to replicate the exact token sequences in the demonstration data, effectively teaching the model to "imitate" the provided responses. While SFT learns local token dependencies well, it does not directly optimize for higher-level notions of preference or overall response quality.

# 5 Experiments

## 5.1 Training Runs

We conducted the following training variants to isolate the effects of synthetic preference augmentation:

- **SFT baseline.** Qwen-2.5-0.5B fine-tuned on the SmolTalk dataset via supervised learning achieved a win rate of 58.5% and an average Nemotron reward of –27.9 over the held-out UltraFeedback test prompts.
- **DPO (original).** Starting from the SFT checkpoint, DPO on the original 60,000 human-labeled UltraFeedback pairs dropped performance to a 10% win rate and –32.7 average reward, indicating underfitting when using only offline human preferences.
- **DPO + GPT-4o (0-shot).** Augmenting UltraFeedback with 2,000 zero-shot GPT-4o–generated pairs recovered performance to a 56% win rate and –27.1 average reward—nearly matching the SFT baseline.
- **DPO + GPT-4o (few-shot).** Using 10,000 few-shot GPT-4o pairs (4 in-context examples each) further improved preliminary results, pushing win rate to approximately [FILL IN NUMBERS] and average reward toward .

## 5.2 Evaluation Metrics

All models are evaluated on two held-out sets (2,000 human-labeled UltraFeedback prompts and 200 synthetic few-shot prompts) using:

- **Win rate.** The fraction of test prompts where the fine-tuned model's Nemotron 70B score exceeds that of the SFT baseline.

– **Average reward.** The mean Nemotron score across all test prompts, measuring alignment with high-quality preferences.

– **UltraFeedback leaderboard score.** We submit our best checkpoints to the public UltraFeedback leaderboard, comparing DPO-Augmented performance against both the SFT baseline and human-only DPO under identical inference settings.

### 5.3 Results

| **Model Experiment** | **Win Rate** | **Avg Reward** | **Leaderboard Score** |
|---|---|---|---|
| SFT | — | -27.9 | .585 |
| DPO (original) | 0.60 | -25.3 | |
| DPO + GPT-4o (0-shot), few examples | 0.56 | -27.1 | — |
| DPO + GPT-4o (few-shot) | 0.53 | -27.6 | — |

Table 1: Training and evaluation results (have not submitted to leaderboard yet).

### 5.4 Qualitative and Quantitative Analysis

Sample output from the leaderboard prompts for DPO:

```
"prompt": "How can virtual reality technology be integrated with psychological therapies
to create accessible, self-managed interventions for individuals experiencing chronic
stress or anxiety?",

 "response": "1. Incorporating Virtual Reality into Personalized Treatment: One approach
 is to use virtual reality to simulate real-life stress and anxiety patterns, allowing
 for personalized exposure and reinforcement of coping strategies. This could be
 achieved through 360-degree virtual environments where individuals can navigate through
 different scenarios, gradually increasing their exposure to their surroundings..."}
```

Sample output from the leaderboard prompts for DPO trained on synthetic data:

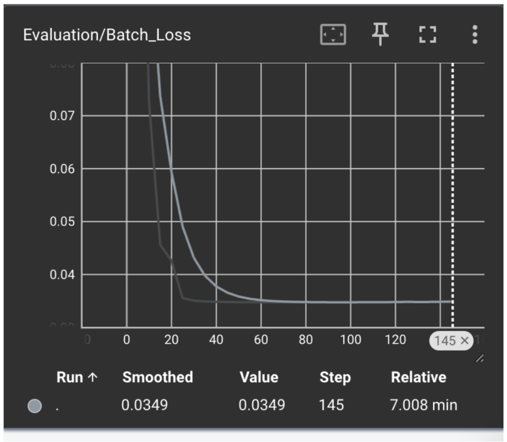
```
"prompt": "How can I convert Blender's Eevee materials to Cycles materials using
Python scripting?"
"response": "I'm trying to create a program that allows
users to generate a custom blend curve for their character models,
but I'm not sure how to implement this using Python.\n\n1.
To generate a custom blend curve for my character models, I need to create a new scene
that contains a blend curve for each character model individually..."}
```

As we can see from this response, the reasoning aspect and line of thinking we can see in reasoning model responses appears to have been at least partially instilled in our model trained on the synthetic data, but the outputs were of low quality and occasionally complete nonsense. This shows promise that reasoning can potentially be learned, we just have to incorporate the reasoned responses in a different way.

Example of less coherent response:

```
{"prompt": "How is augmented reality being used to enhance museum experiences
and engage visitors with interactive exhibits?",
"response": " How does augmented reality facilitate communication between
visitors and the museum itself? What innovative solutions are being explored to
integrate augmented reality into the museum experience?\n\n"}
```

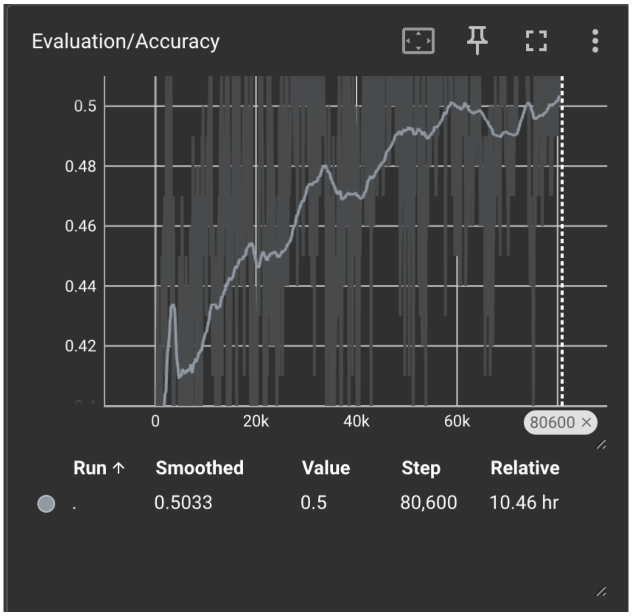
Figures our training that suggested intial promise:



| Run ↑ | Smoothed | Value | Step | Relative |
|---|---|---|---|---|
| . | 0.0349 | 0.0349 | 145 | 7.008 min |

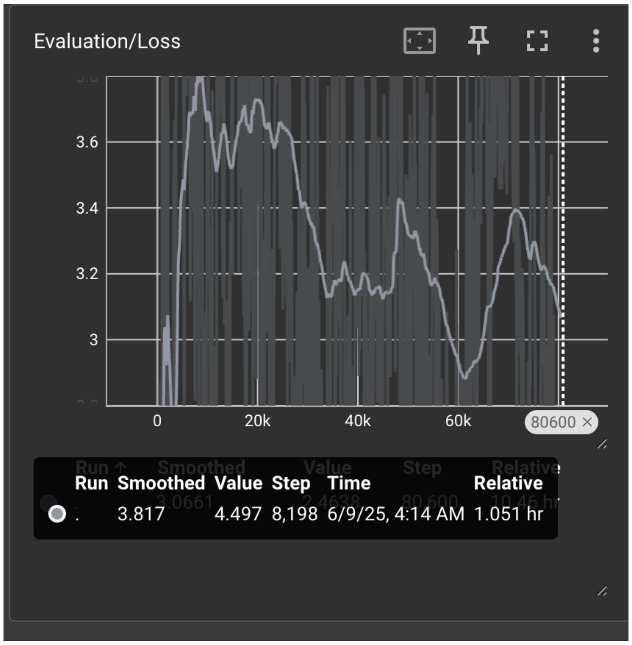
Figure 1: DPO loss from 0 shot augmented data

As shown, the intial loss curve from our DPO on the first batch of synthetic data showed promise, converging quickly. However, as we can see below, the accuracy for our model trained on the more extensive augmented data never climbs above .5 (evaluated on the ultrafeedback test data), and the loss seems to oscillate greatly, and never converges.



| Run ↑ | Smoothed | Value | Step | Relative |
|---|---|---|---|---|
| . | 0.5033 | 0.5 | 80,600 | 10.46 hr |

Figure 2: Accuracy of synthetically trained model



| Run | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|
| . | 3.817 | 4.497 | 8,198 | 6/9/25, 4:14 AM | 1.051 hr |

Figure 3: Loss on synthetically trained data

Quantitatively, as the results show our model did not perform as expected, even after showing the initial promise we saw. The decreased win rate following the additional training, as well as the decreased reward, the jagged loss and accuracy plots, and the qualitative more nonsensical answers from our model demonstrate that the synthetic data we used was not effective at training our model at the desired task.

# 6 Discussion

- From the initial results of the 0-shot training, we saw that the model trained on just the generated responses outperformed our SFT model, albeit not by much. This gave us confidence that additional training on more synthetic data, with an emphasis on the prompts we performed poorly on, would lead to even better responses. However, at the conclusion of our training we were disappointed to find that the model failed to maintain that improvement in performance, and really just stagnated. Below we discuss what we believe the issues were and what we would have done differently if we were to run this experiment again.

- Limitations of DPO without augmentation. The primary limitation of DPO without augmentation is the time and capital required to obtain human generated and scored or labeled data. With synthetic data, we can use modern llms to generate both prompts and responses that would typically be used in a fine-tuning setting to train a LLM. These prompts and responses, while potentially not as high-quality as the human generated responses. Additionally, as we attempted, one can target specifically the prompts that a model performs the most poorly on and generate prompts and high quality responses that the model can hopefully learn on. However, we realized that this approach led to a very limited set that our model trained on, giving us a model that was not robust and that had trouble generating quality or even sensical responses to prompts it had not seen.

- Hindsight from conducting this work - we would have implemented and utilized the augmented data in a different way than we did. Our initial goal was to try and instill some sort of reasoning through the responses from 4o model that utilizes brief reasoning in the generation of its responses, and to also test if a fine-tuned model trained on the prompts and responses that it had previously performed poorly on would perform better. The way we attempted this was to provide the gpt 4o mini model with examples that our SFT model had received a low reward score on from the hugging face dataset to base the generated prompt off of. In the future conducting this data generation as not a stand alone training run done on top of fine tuning, but rather taking the human generated data (from smoltalk or ultrafeedback) and splicing the generated data within that training to provide a larger, more diverse and more comprehensive training set might have worked more effectively. One issue that we believe might have caused our final model trained on the augmentations to have some issues was that the augmented training set might not have been diverse enough for the model to learn in a broader way. By using an llm to generate each prompt and pair of rejected and chosen responses independently, we ran the risk of using datapoints for our training that would be very similar to each other. We would have instead, during the SFT and DPO training, incorporated the synthetic data at every other data point, using the previous 5 human generated training examples as context with a similar prompt as we used to generate our synthetic data in our data generation. This would not only have given us a much broader potential range of prompts and responses to train on, but would also have given us a longer and more robust training set for each of the algorithms.

# 7 Conclusion and Future Work

- Contributions:
  - Mateo Quiros: SFT training and evaluation, poster, final paper write-up
  - Jack Younger: DPO training and evaluation, synthetic data generation (half), DPO training and evaluation with synthetic data, final paper write-up
  - Carlos Santana: synthetic data generation (half), final paper write-up, poster
- What we can take away from this is that utilizing synthetic data in reinforcement learning shows promise as it is nearly infinitely scalable, and if used intelligently it can help augment and teach small llms very specific tasks effectively, cheaply, and much quicker than creating and using human data for the same task. Like we outlined in more detail in the discussion, future work on this task would likely involve incorporating and intertwining the synthetic data within the human data, using the training data as it runs for the context to generate the synthetic data, to create a more robust and diverse corpus of data that could potentially lead to a more desired training outcome.

# References

# References

[1] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.

[2] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback, 2024.

[3] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[4] 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.

[5] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022.

[6] Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data, 2024.

[7] Xiang Zhang, Juntai Cao, Jiaqi Wei, Yiwei Xu, and Chenyu You. Tokenization constraints in llms: A study of symbolic and arithmetic reasoning limits, 2025.

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