Extended Abstract

Motivation LLM-powered chatbots have enabled users to access a large wealth of information in a way that feels both conversational and fundamentally human. The process for doing so, however, is quite opaque. By implementing Extension 2.1 in Ouyang et al. (2022), our project seeks to identify the factors that contribute to strong model training for Instruction Following tasks. We seek to determine whether learning from human preferences, synthetically augmented training data; or perhaps, even a mix of the two methods provides the best method to train LLMs. In doing so, we hope to make our models just a little bit better at understanding the human experience.

Method Using a 0.5 B-parameter Qwen-2.5 decoder, we first conducted Supervised Fine-Tuning (SFT) on the LLM to improve model performance using a $97\%/3\%$ Train-Test Split on Allal et al. (2025)'s Smol-Smoltalk dataset. We then performed Direct Preference Optimization (DPO) with and without Synthetic Data Augmentation (SDA) to see how, if at all, these methods improve performance over the SFT-trained model. For these models we used a $98\%/2\%$ Train-Test Split on Dubois et al. (2024)'s UltraFeedback dataset, augmented with synthetic queries for the SDA training. Using the Nemotron-70B Reward Model, we compared all three of our trained models against the Qwen2.5-0.5B-Instruct for 500 responses with a context window of 512 tokens. We limited our models to this token threshold due to compute constraints. Collectively, these methods allowed us to measure performance of these training methods against a model explicitly design for human interaction, providing a solid baseline for performance.

Implementation We trained all models using an A100-80GB on the Google Colab platform. Tokens were label-masked and processed in bfloat16 with Flash-Attention-2 for the both SFT model and DPO model training processes. Both processes used the Adam Optimizer, but had differences in hyperparameters due to relative instability. SFT used a learning rate of $1e$-4 with other hyperparameters chosen by sweep. For DPO, we encountered substantially more instability causing us to lower our learning rate to $5e$-6. We also included regularization methods including clamp of 20 and a conservative $\beta = 0.015$. To properly evaluate our SDA model, we used the same hyperparameters and training as we did within the DPO model.

Results Across our three methods, we saw improvement in our win rate against the instruct model. The SFT-only model obtained a win rate of 62.7%, the DPO-model obtaining a win rate of 64.9%, and the Synthetic Data Augmented DPO-model won by 65.5%. Additionally, we observed steady training and evaluation loss throughout training in all three methods. Finally, qualitative analysis of the model responses show that all three models perform well on general queries; however, our SFT model often gave no answer on complex queries. That being said, our DPO and SDA models were able to answer such queries to moderate success, potentially leading to their superior benchmark performance.

Discussion The margin of victory and successive gains in win rate suggest that all three methods were effective in improving performance. Our qualitative analysis further supports this claim beyond the raw numerical evaluations of our algorithm. All three methods had strong performance on general queries with clear and direct responses to the questions, which conforms to our expectation of beating the instruct model baseline. Those extra 2-3% may come from the DPO model and SDA-enhanced model's ability to answer those difficult queries. Given the compute limits faced, small models, and limitation of training to single-turn datasets, however, it would be difficult to extend these results to the larger LLMs that power AI chat bots. Still, they provide evidence for potentially effective methods of training.

Conclusion These results show that both learning from human preferences and diversifying the dataset are effective methods in training LLMs in Instruction Following tasks. It's important to note that our experiments were conducted on smaller models; and thus, we cannot conclude that these methods will be as effective for the larger models that power popular LLM applications. Still, they provide an exciting path for what might be to come for LLM training and how we may combine methods to align LLMs with ideal instruction following behavior. Hopefully, making them have just a bit better human understanding along the way.

Data-Augmented DPO: Comparing Enhancements of SFT-Trained LLMs

Austin Bennett
Department of Computer Science
Stanford University
gaustinb@stanford.edu

Rishi Padmanabhan
Department of Computer Science
Stanford University
riship1@stanford.edu

Jared Weissberg
Department of Computer Science
Stanford University
jared1@stanford.edu

Abstract

This paper explores a survey of methods for improving LLM performance on Instruction Following. Using a 0.5B-parameter Qwen-2.5 decoder as our base model, we compare three training approaches: Supervised-Fine Tuning (SFT), Direct Preference Optimization (DPO)), and Synthetic Data Augmentation (SDA). These experiments are conducted in succession to gauge relative impact. Through this pipeline and hyperparameter tuning, we were able to achieve strong results against the fine-tuned Qwen2.5-0.5B-Instruct, winning most queries on the Ultra-Feedback dataset. Our augmented dataset had marginal improvements on top of DPO, showing that it may be an effective training method for Instruction Following, but that further research is warranted to confirm this result.

1 Introduction

Ever since the release of the Vaswani et al. (2017)'s paper "Attention is All You Need", research on Large Language Models (LLMs) has been thrust into the limelight – and rightfully so. After years of foundational research, LLM-powered applications have provided a watershed moment for Artificial Intelligence. While the field was previously relegated to complex back-end algorithms, AI chat bots have allowed users to leverage a large corpus of information using nothing more than an everyday conversation. Though how are we able to convert rigid model predictions to the multi-turn interactions we have LLMs? The answer lies in how the models are trained.

Many of these methods rely on a field of Artificial Intelligence known as Reinforcement Learning, where an agent learns to reason about an environment by optimizing a reward model that represents the problem. In the context of LLM training, Reinforcement Learning can be used to supplement prior training on difficult to follow instructions. Most commonly, this consists of running Direct Preference Optimization (DPO) after conducting Supervised-Fine Tuning (SFT) on a base model, but further improvements may still be possible.

Direct Preference Optimization consists of optimizing model weights for a preferred model answer within pairs in response to a user query; however, this method inherently relies on some arbiter to determine this be it another LLM or a human grader. In doing so, we may limit the diversity of responses that the model sees; and therefore, restrict its usefulness on less common responses. Thus, our paper explores Extension 2.1: Synthetic Data Augmentation to improve our dataset and observe changes to model performance after re-running DPO with the synthetic dataset.

Stanford CS224R 2025 Final Report

By training with the additional prompts generated with Synthetic Data Augmentation (SDA), we're able to learn from a more diverse dataset; and in doing so, can make our model more adaptable. Though as we stray further from our core dataset, we may see instability and misaligned performance. By pushing this boundary, however, we get to answer a fundamental question about improving model performance: is it better to learn form rigid preferences or by expanding that which we're evaluating? Our results show that perhaps a mix of both is best.

2 Related Work

Synthetic Data Augmentation is a popular process for increasing the diversity of a dataset; however, one must be careful to ensure that responses are still high quality. Thus, we began by examining a survey of methods for SDA including Bai et al. (2022)'s approach of RLAIF with Constitutional AI. Here, the authors were able to generate synthetic data by feeding a list of rules and principles to an AI model, which in turn supplemented the dataset. They found that this method to improve the performance of the model, suggesting that human labeling of data may not be as necessary, as we can control model training with other LLMs.

We see further evidence of this claim in Lee et al. (2024), where the authors use direct-RLAIF which avoids using a reward model by obtaining rewards from an off-the-shelf LLM model. In doing so, they achieved superior results to canonical RLAIF and claim that direct-RLAIF has performance on-par with using human feedback. Beyond replicating human performance, we see that diversifying the dataset for specific tasks can also achieve higher performance. Indeed, Dong and Ma (2025) describes an iterative synthetic data generation process in which a conjecture model is iteratively trained on statements proven by a prover model; thereby, building up knowledge over time between the two models with increasingly challenging conjectures to prove. For our setting, however, we decided that the method used in Lee et al. (2024) was most appropriate, albeit with some minor modifications for our Synthetic Data Analysis.

3 Method

3.1 Overview

To compare performance between SFT, DPO, and SDA-augmented DPO, we propose a three-stage training process. Using a 0.5 B-parameter Qwen-2.5 decoder, we begin by conducting Supervised Fine-Tuning before moving into the Direct Preference Optimization on our SFT model. Then, we conduct Synthetic Data Augmentation to generate for additional prompts for a dataset on which we conduct DPO training with the same parameters. Comparing these two models, we can determine whether SDA improves or hinders the training process.

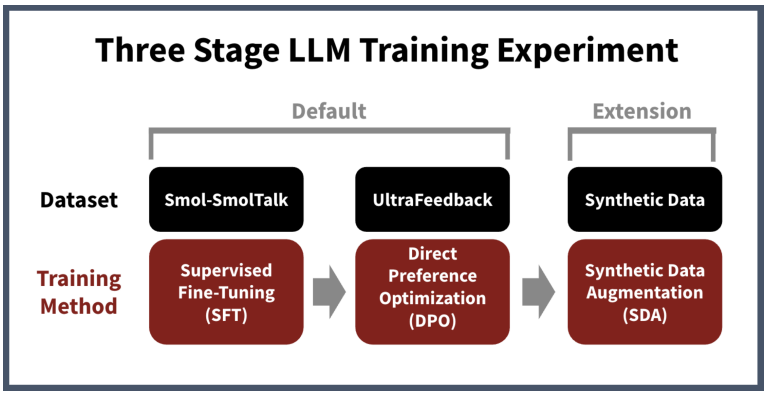

Figure 1: High-Level Overview of Method and Extension

Considering that the standard process for LLM training involves conducting SFT prior to DPO, we elected to keep that order flow within our experiments. Furthermore, since we're interested in seeing how SDA can potentially add to DPO, this process needs to be run last. While alternate experiments

may seek to compare running another method with SDA after SFT in comparison to DPO, we're solely concerned with how this can affect DPO. Thus, we follow this three stage model.

3.2 Supervised Fine Tuning

We begin with an pretrained 0.5 B-parameter Qwen-2.5 decoder. In order to run SFT, we use Allal et al. (2025)'s Smol-SmolTalk as a training dataset. It contains nearly half a million high-quality chat pairs generated form GPT-4o with both a paired user query and assistant response regarding a sample model instruction. For the initial model alignment we seek for SFT, this style was ideal. After filtering to $446,531$ pairs, we employed a $97\%/3\%$ Training-Evaluation split in order to maximize the size of our training set, but still have a sufficiently large evaluation set. To obtain strong accuracy, we wanted our models to learn from as much data as possible, however, this cannot come at the expense of reporting accurate metrics about the model.

Using this data, we label-masked our user tokens and processed in bfloat16 with Flash-Attention 2 when present. We used the Adam Optimizer with a learning rate of $1e$-4; and due to compute constraints, limited tokens to 512. Using these parameters, we then conducted SFT according to the following objective:

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

3.3 Direct Preference Optimization

Using our SFT-trained model as a base model, we subsequently conducted DPO to refine performance. We used a subset of Dubois et al. (2024)'s UltraFeedback as the training dataset with a $98\%/2\%$ Training-Evaluation split. Since our SFT model was trained on 512 tokens, we did the same for DPO. The updated leaderboard, however, required 1024 tokens. Between loading the SFT model and running DPO, we encountered frequent memory issues. Thus, we ran DPO with 512 tokens, but generated responses with 1024 tokens. From there, we similarly label-masked our user tokens and processed in bfloat16 with Flash-Attention-2 when present. We then optimized our model parameters according to the reward parameterization seen in Rafailov et al. (2023). This yielded the following loss function for DPO:

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

Given persistent problems with model instability and overfitting, we used a conservative set of hyperparameters. First, we set $\beta = 0.015$ in the above loss function to ensure that samples did not have an overwhelming affect on model weights. We also used the Adam Optimizer with a learning rate $5e$-6 with cosine scheduling to ensure that the learning rate decreased overtime. Additionally, we employed a clamp value of 20 on our DPO margin to help regularize the training and prevent large spikes. Finally, to ensure safe training we instituted a warmup period of 100 iterations with a linear increase in β. Collectively, these efforts allowed for stable DPO training with a consistent gradual decrease in evaluation loss before an eventual leveling off in results after about 1000 iterations. We let training run until around 2000 to ensure that we'd optimized as much as possible, but stopped afterwards to prevent overfitting.

3.4 Synthetic Data Augmentation

In preparing our synthetic dataset, our team considered many different approaches; however, since we wanted to see exactly how much impact SDA could have on top of DPO, we elected to keep the synthetic data as similar to our training set (Dubois et al. (2024)'s UltraFeedback). That being said, increasing the diversity of the dataset was still of utmost importance. Thus, we took a very similar approach to the methods presented in Lee et al. (2024), which aligned very well with this goal.

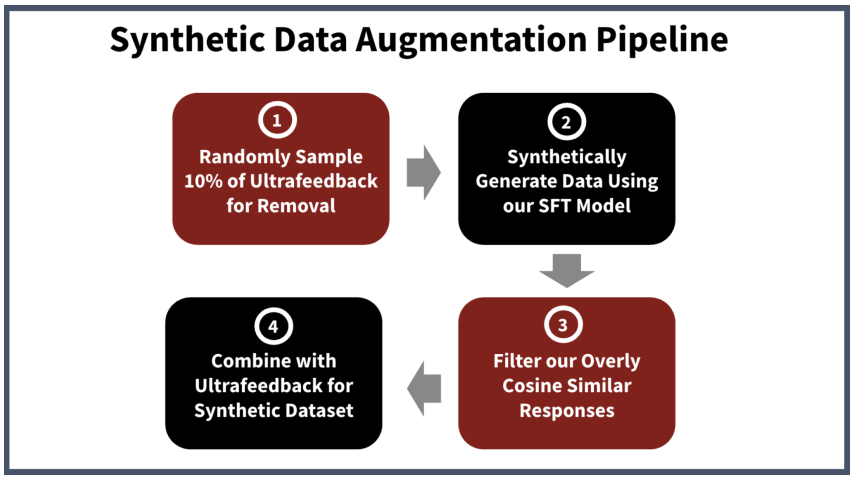

Figure 2: Synthetic Data Augmentation

Our Synthetic Data Augmentation pipeline works in a few key stages. One key divergence from the methods in Lee et al. (2024) is that we use our own SFT model to create synthetic data to add into our hybrid dataset with Ultrafeedback. We considered using a top-of-the-line LLM to generate these queries as the authors in Lee et al. (2024) did, however, we decided to use the SFT model to make the pipeline as self-contained as possible. It's entirely possible that gains from training on the augmented dataset could have just been the model inheriting the style of the better trained LLM. Thus, we avoided this approach for our experiments.

Naturally, this process could lead to a lack of diversity among the prompts, so we filtered our most cosine similar responses leaving only the most different responses. We then combined it with our original dataset to obtain our synthetically generated dataset and conducted DPO on our synthetically augmented dataset with the same parameters in order to provide a direct comparison to the DPO run without SDA.

4 Experimental Setup

Our goal is to measure perfomance on Instruction Following tasks, thus, a natural experiment to conduct is to compare our trained models to well-trained LLMs. More specifically, we compare our models against the Qwen2.5-0.5B-Instruct baseline, which is version of our the Qwen-0.5B-model explicitly trained for conversation. Given the fact that our models are aligned for instruction following, this provides a strong reference point for us to compare our model results against.

Prior to model training, we held out 500 queries from the Dubois et al. (2024)'s UltraFeedback dataset for testing. This was done to ensure that we weren't training our models on responses used for future testing. We generated responses to the queries for both our models and the instruct model, using LLM-as-a-judge with NVIDIA's Nemotron-70B-Reward model serving as an evaluation model to pick a winner on each query. Given our persistent memory issues, we ran 512 token queries for our model to unlock best performance. While margin of victory is potentially a consideration in future tests, for our purposes we were solely interested in seeing how often these models can produce better results than the instruct model. Given our limited resources, it's more likely that one of our experiments could lead to an outlier response compared to the instruct model; and therefore, we thought this would unnecessarily bias our metrics against our models. Still, weighting win rate by margin of victory is a consideration for our future experiments.

5 Results

5.1 Overview

Overall, we observed that all three of our methods beat out the Instruct model. They also progressively yielded a stronger win rate against its predecessor using the Nemotron evaluator, though the

differences were small. More specifically, we obtained the following win rates on the 500 held-out prompts from Dubois et al. (2024)'s Ultrafeedback:

Table 1: Win Rate on Nemotron

Method	SFT	DPO	SDA
Instruct Model	0.627	0.648	0.656

5.2 Quantitative Evaluation

5.2.1 Supervised Fine Tuning

Upon running Supervised Fine-Tuning, we observed a spike in training loss before a gradual decline consistent with the usual supervised learning process. At the recommendation of our teaching assistant, Fengyu Li, we stopped our SFT process earlier on in the process, since our initial runs of SFT had overfit the training model. Our evaluation loss followed a very similar trend suggesting that overfitting was not a concern at this stage.

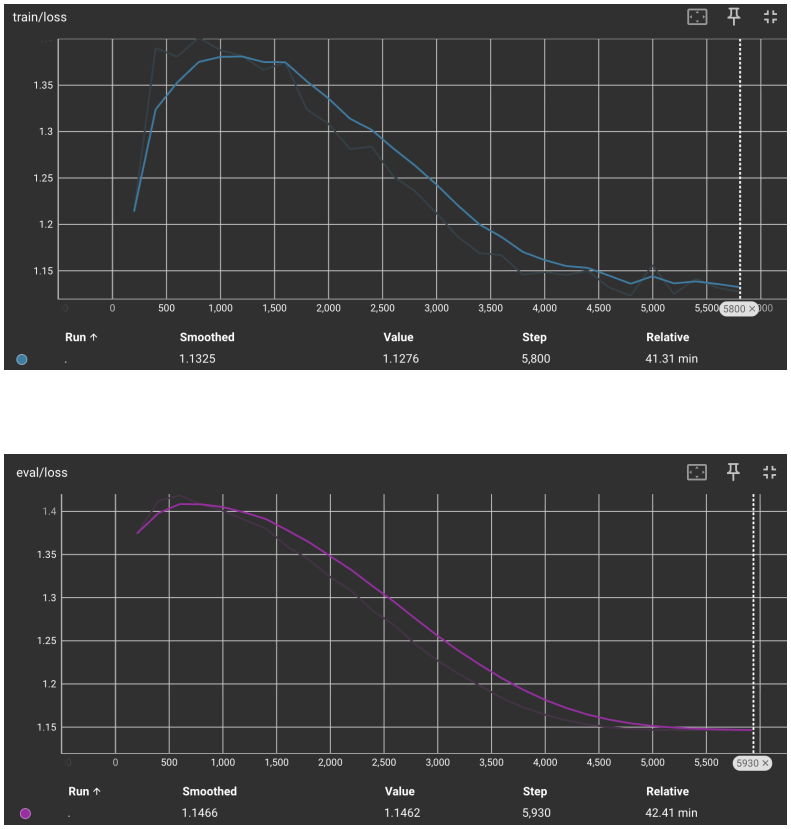

Figure 3: SFT Training Loss (Top) and Evaluation Loss (Bottom)

Upon evaluation with the Nemotron reward model. our SFT model was able to win 62.7% of runs against the Instruct model suggesting solid performance, but still some room for improvement. Prompts on the most recent Instruction Following leaderboard testset, however, caused our SFT model to give an empty response in substantial portion of the dataset. Thus, it's possible that these non-answers could have impacted our Nemotron score on Dubois et al. (2024)'s Ultrafeedback dataset. Still, the general score is high enough to suggest that SFT is able to align our base model with a more conversational style.

5.2.2 Direct Preference Optimization

Upon running Direct Preference Optimization, we observed a very smooth gradual decline in evaluation with a much noisier training loss than we observed in SFT. This is to be expected as DPO training can be quite unstable; however, the numerous regularization methods we employed appeared to yield a smooth training process. Similar to SFT, we terminated the run at the plateau of the evaluation loss to avoid overfitting and obtained the following graphs of training loss and evaluation loss.

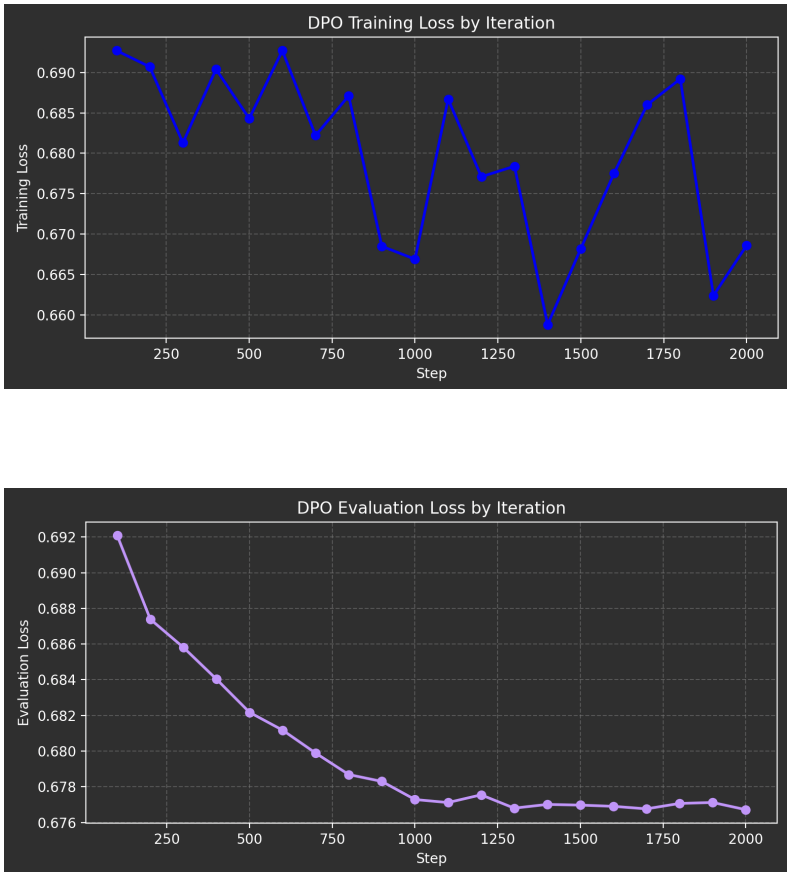

Figure 4: DPO Training Loss (Top) and Evaluation Loss (Bottom)

Our DPO model was able to win 64.9% of runs against the Instruct model suggesting a slightly improved performance over the SFT model. Additionally, the margin of victory appears larger on average (though this was not an evaluation metric) and our DPO model is now able to respond logical to many of the held out prompts for the Instruction Following leaderboard that the SFT model could not. These suggest that the win rate may underrate the difference between the two models.

5.2.3 Synthetic Data Analysis

Finally, we used our SDA pipeline to generate a series of synthetic pairs using Dubois et al. (2024)'s UltraFeedback as the basis. We randomly selected 10% of the dataset for generation. In total, our pipeline chose to keep 657 of these pairs. Combined with the remaining entries in Dubois et al. (2024)'s Ultrafeedback, this formed our synthetically augmented dataset. We then fine-tuned our SFT model with DPO again, but this time using the SDA dataset as opposed to Ultrafeedback. The training loss and evaluation loss were highly similar; however, there was a modest gain in the win rate using Nemotron evaluation to 65.6%. Beyond this, the two DPO models were quite similar with the SDA-enhanced DPO model achieving the following loss curves:

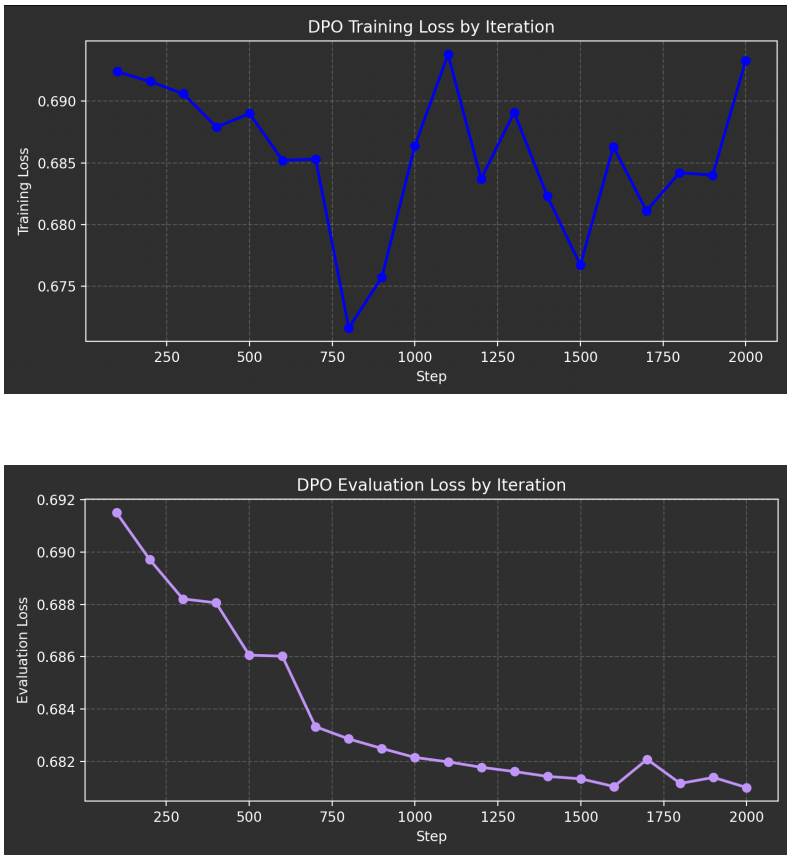

Figure 5: SDA-enhanced DPO Training Loss (Top) and Evaluation Loss (Bottom)

5.3 Qualitative Analysis

5.3.1 General Queries

As suggested by the modest gains in win rate against the instruct model as determined by the Nemotron reward model, all three models demonstrated a strong ability to produce clean readable text in response to a general query. Generally, our DPO and SDA-enchanced DPO models tended to be a bit more direct and provide scaffolding to their answers than the SFT model; however, sometimes the latter approach was preferable. Below are a few selected examples:

Instruction Following Held-Out Set Prompt #2

Prompt: How can virtual reality technology be integrated with psychological therapies to create accessible, self-managed interventions for individuals experiencing chronic stress or anxiety?
SFT Response: Virtual reality (VR) technology has the potential to revolutionize the way we interact with virtual reality (VR) technology, enabling the creation of immersive and interactive experiences that are designed to simulate real-world scenarios. [Truncated for brevity]
DPO Response: Virtual reality technology can be integrated with psychological therapies to create accessible, self-managed interventions for individuals experiencing chronic stress or anxiety by providing a unique and immersive environment that allows for personalized exploration, relaxation, and relaxation of the mind. [Truncated for brevity]

5.3.2 Complex Queries

Despite the uniform success on general queries, performance more complex queries (i.e. those involving logic, languages, calculation) differed sharply between the models. The SFT model was

generally biased against answering complex questions, such as those seen on the held-out Instruction Following leaderboard dataset and would often output no response. Our DPO models, however, answered these prompts. Below are a few selected examples demonstrating this difference:

Instruction Following Held-Out Set Prompt #3

Prompt: How can I convert Blender's Eevee materials to Cycles materials using Python scripting?
SFT Response: No Output.
DPO Response: You can use the following Python script to convert Blender's Eevee materials to Cycles materials [Code snippet follows, truncated for brevity].

5.3.3 Translation Queries

Our SFT model usually did not respond to translation queries, but interestingly our DPO models exhibited a minor ability to translate. In particular, one especially interesting response from the held-out Instruction Following leaderboard consisted of the model hallucinating a single word in Italian – "Leiaggi" – but the rest of the sentence is similar.

Instruction Following Held-Out Set Prompt #288

Prompt: Given a sentence in German, provide an equivalent paraphrased translation in Italian that retains the same meaning both through the translation and the paraphrase. Q: Diese Bücher gehören zu den klassischen Werken der deutschen Literatur. A:
SFT Response: No Output.
DPO Response: Leiaggi sono parte di lettere classiche italiane.

This suggests that even though we have increased capabilities with our DPO-trained model, it still can fall victim to the well-documented issue of hallucinations with LLMs.

6 Discussion

6.1 Experimental Inferences

From our experiments, we observed that SFT, DPO, and SDA are all effective tools for training a model on Instruction Following tasks. This suggests diverse datasets can work with the more classic SFT and DPO to effectively tune LLMs. Still, our gains from SDA were fairly marginal and likely the result of (1) a small generated dataset, and (2) being drawn from the underlying dataset itself. Thus, generating for SDA the same pipeline may not be the best method for this process, but it did have some benefit in our case.

6.2 Limitations

Our underlying base models were quite small (2.5B) and not the typical LLMs backing AI chat bots. Thus, one should not assume that the exact same gains can be had for larger LLMs. Additionally, our lack of compute inherently restricted the number of tokens that we could process; and just as responses can get weaker with larger context windows, we may see the worse experimental results when extrapolated to a larger context window. Finally, we used single query and response datasets. This could be inaccurate for multi-turn conversations as we often see in LLMs, especially if further prompts clarify a previous one.

6.3 Future Work

Considering our limitations on computation, future work should be done to investigating large scale generation of data generated from an SFT model as it could potentially lead to a larger gain that what we observed. Additionally, further work should be done to explore diversifying the dataset beyond the SFT model as it may lead to a better set of query response pairs to learn from with DPO. One such method would be to use ChatGPT or another strong LLM to generate responses. Considering its

efficacy in Lee et al. (2024) and the consistent improvement in these models, it may end up being a more effective method for SDA.

7 Conclusion

These results are certainly exciting, but the ultimate test rigid human evaluation. While we don't directly impart this into our synthetic data set or modify training values as they arise, humans are the ones who use these models and truly understand what it means to be aligned with their preferences. For this reason, training cannot only involve methods to diversify data or better select for pre-labeled preferences, it must actively include as many humans and as many different experiences as possible. If we do, general artificial intelligence may be just around the corner; but if we don't, we risk relegating our models to a subset of the human experience, denying this tool to everyone who happened to be forgotten by the process.

8 Team Contributions

- **Austin Bennett:** Project Proposal, Project Milestone, SFT Model (Assist), Experimental Assistant (Assist), DPO (Lead), Poster (Lead), Project Report (Lead)
- **Rishi Padmanabhan:** DPO (Lead), SDA, Poster (Assist), Project Report (Assist)
- **Jared Weissberg:** Data Processing, Experimental Design (Lead), SFT Model (Lead), DPO (Assist), Project Report (Assist)

Changes from Proposal Originally, we had sought to pursue Extension 2.7 to use tools for fantasy football; however, given the lack of real-time data that we'd sought to use, we decided to pivot to our next choice: Extension 2.1: Synthetic Data Augmentation (SDA).

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. (2022). https://arxiv.org/abs/2212.08073

Kefan Dong and Tengyu Ma. 2025. STP: Self-play LLM Theorem Provers with Iterative Conjecturing and Proving. *arXiv preprint arXiv:2502.00212* (2025). https://arxiv.org/abs/2502.00212

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2024. AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. https://doi.org/10.48550/arXiv.2305.14387 arXiv:2305.14387 [cs.LG]

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. (2024). https://arxiv.org/abs/2309.00267

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model Is Secretly a Reward Model. *arXiv preprint arXiv:2305.18290* (2023).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, Vol. 30. 5998–6008. https://arxiv.org/abs/1706.03762