

Improving GPT-2 Fine-Tuning with Direct Preference Optimization for Sonnet Generation and Paraphrase Detection

Stanford CS224N Default Project

Nicolas Bejar

Department of Computer Science
Stanford University
nbejar@stanford.edu

Anna Gutowska

Department of Computer Science
Stanford University
gutowska@stanford.edu

Cristian Budianu

Department of Computer Science
Stanford University
cbudianu@stanford.edu

Abstract

Aligning language models with human preferences is an open challenge. The dominant approach, Reinforcement Learning from Human Feedback (RLHF), is complex and unstable in practice, particularly for preference-style tasks. Direct Preference Optimization (DPO) serves as a simpler alternative, replacing the RL pipeline with a direct preference objective. Yet it remains unclear whether DPO's advantages transfer to creative and semantic tasks. Our primary research question is: *Can Direct Preference Optimization improve a GPT-2's performance on sonnet generation and paraphrase detection compared to standard supervised fine-tuning?* Starting from the default project baseline, we first complete supervised fine-tuning (SFT) of GPT-2 for both tasks, then apply DPO and MADPO as a second alignment stage. We also experiment with techniques such as symmetric data augmentation, LoRA, architectural modifications, ensemble averaging, top-k sampling with repetition penalty. These improvements substantially strengthen the underlying SFT models; our strongest SFT configurations achieve paraphrase detection accuracies of 0.901 (dev) and 0.900 (test) and for sonnet generation chrF scores of 43.17 (dev) and 42.71 (test) on the leaderboard. Our findings show that the improved SFT sonnet model with DPO ($\beta = 0$) achieves a dev chrF of 41.88 and a test chrF of 41.638, up from 41.208 for the baseline SFT, while combining symmetric augmentation with DPO yields the best paraphrase detection results: 0.902 dev accuracy and 0.896 F1, with an identical 0.902 accuracy on the test leaderboard. A systematic sweep over the DPO regularization coefficient β reveals that small values ($\beta = 0.25$) offer the best trade-off, suggesting that mild preference alignment on top of a strong SFT models is sufficient for both tasks.

1 Key Information to include

- Mentor: Wei Liu
- External Collaborators: None
- Sharing project: No
- Late Days:

Team member	Late days remaining
Anna	6
Nicolas	6
Cristian	6
Total pooled	18
Team size	3
Late days applied for the report	3

2 Introduction

Large language models (LLMs) trained on massive text corpora acquire broad knowledge and strong reasoning abilities, but controlling their behavior and aligning it with human preferences remains a challenge. The standard approach, Reinforcement Learning from Human Feedback (RLHF) [1, 2], works by first fitting a reward model to a dataset of human preferences over pairs of model responses and then using reinforcement learning (RL), typically PPO [3], to fine-tune the language model to maximize the learned reward while staying close to a reference policy via a KL-divergence constraint. While this pipeline has produced impressive results in systems like ChatGPT, it is complex and often unstable: it requires training multiple models, sampling from the policy during training, and careful hyperparameter tuning. These limitations make RLHF difficult to implement and scale reliably. Motivated by the challenges of PPO, Rafailov et al. propose DPO (Direct Policy Optimization) as a simpler way to align language models with human preferences that avoids the overhead of RL.[4]

3 Related Work

Prior work by Rafailov et al. shows that Direct Preference Optimization (DPO) can match or exceed PPO-based RLHF on tasks like sentiment generation, summarization (TL;DR), and dialogue (Anthropic HH), without requiring a reinforcement learning loop or a reward model[4]. However, their evaluations are limited to English, models no larger than 6B parameters, and tasks with naturally occurring preference data. One limitation of DPO is that it applies a fixed temperature parameter β uniformly, treating all preference pairs equally. MADPO (Margin-Adaptive DPO) addresses this by estimating per-example preference margins to reweight the loss dynamically [5], making it particularly promising for creative tasks such as poetry, where some preferences are easier to satisfy than others. Additionally, full fine-tuning can be computationally expensive and prone to overfitting on small preference datasets; Low-Rank Adaptation (LoRA) mitigates this by freezing pretrained weights and inserting trainable low-rank matrices in each Transformer layer [6].

4 Approach

The main approach of our project is using Direct Preference Optimization (DPO), introduced by Rafailov et al. [4], to improve performance of the GPT-2 model beyond standard supervised fine-tuning for the paraphrase detection and sonnet generation tasks. Our project builds on the default final project codebase, which provides a GPT-2 model [7] with pre-trained weights and infrastructure for three tasks: sentiment analysis, paraphrase detection, and sonnet generation. We describe each component of our approach below.

Baseline SFT. The starter code provides two baselines: a last-linear-layer approach that freezes GPT-2 and only trains the task-specific classification head, and a full-model fine-tuning approach. For sentiment analysis, the model is trained on the Stanford Sentiment Treebank (SST) [8] and CFIMDB datasets. For paraphrase detection, a classifier head takes the concatenated hidden states from sentence pairs drawn from the Quora Question Pairs dataset and predicts whether they are paraphrases. For sonnet generation, the model is fine-tuned with a standard language modeling objective on a corpus of Shakespearean sonnets, then generates new sonnets autoregressively.

SFT improvements.

To improve supervised fine-tuning (SFT) performance on **paraphrase detection**, we implemented three techniques. First, we exploited the symmetry of the task: the result should be the same regardless of the order of the two sentences. We augmented the dataset by adding reversed pairs—for each

original pair (A, B), we included (B, A)—doubling the number of examples. Second, we replaced the standard classification head with a two-layer feedforward head, enabling the model to capture more complex patterns. Third, we applied ensemble averaging: we selected two checkpoints from different epochs that achieved identical performance on the development set and averaged their weights. These techniques were applied cumulatively, with gains stacking across methods.

We applied three modifications to the baseline **sonnet generation** pipeline. First, we froze the bottom layers of GPT-2 during fine-tuning (keeping only the last four transformer blocks and the language model head trainable), which acts as a regularizer and speeds up training. Second, we replaced greedy decoding with top-k sampling ($k = 50$) combined with a repetition penalty ($\alpha = 1.2$) to reduce degenerate repetition in generated sonnets. Third, we added early stopping with a patience of 3 epochs based on development set chrF, which prevents overfitting.

As a final effort to improve **sonnet generation**, we trained an "improved v2" model to generate a single line of the sonnet given the previous 3 lines. Full sonnets were generated iteratively using a for loop, starting with the first 3 lines. In this way, using the sonnets given, we created 11 times (14 lines minus 3 first lines) more examples (but smaller). We also experimented with explicit prompts instructing the model to write the next line and attempted to impose a rhyme scheme using line numbers, though the latter did not yield successful results.

LoRA We implemented LoRA [6] from scratch and applied it to paraphrase generation model.

DPO. After the SFT stage, we apply Direct Preference Optimization [4] as a second alignment step. DPO removes the need for training a separate reward model (as required by PPO-based RLHF [3, 1]) by directly optimizing the policy on preference pairs. The DPO objective is:

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

where y_w and y_l are the preferred and dispreferred outputs, π_{ref} is the frozen SFT model, and β controls the deviation from the reference policy.

For sonnet generation, we construct preference pairs by generating multiple completions from the SFT model for each training sonnet prefix (first 3 lines) and ranking them by chrF against the ground truth continuation. The highest-scoring completion becomes y_w and the lowest becomes y_l . For paraphrase detection, we generate predictions from the SFT model on the training set and construct pairs where the correct classification is y_w and the incorrect one is y_l . We train DPO for 3 epochs with $\beta = 0.1$ and a learning rate of 1×10^{-5} .

MADPO. To build on top of the DPO implementation, we implemented Margin Adaptive Direct Preference Optimization (MADPO).[5] This DPO variant enhances DPO through an adaptive weight function that reduces the importance of training samples the model is already confident about (large margins) while giving greater emphasis to ambiguous preference pairs (small margins). The coefficient function, $c : \mathbb{R} \rightarrow [c_{\min}, c_{\max}]$ is applied to the explicit reward margin h_ϕ to a scalar:

$$c(h_\phi) = c_{\min} + \frac{c_{\max} - c_{\min}}{1 + \left(\frac{c_{\max} - 1}{1 - c_{\min}} \right) \exp(\lambda(h_\phi - \tau))}$$

where $c_{\min}$ is the dampener for high-margin pairs, $c_{\max}$ amplifies low-margin pairs, τ defines the threshold for a high-margin preference pair, and λ controls the sharpness of the transition around the threshold. Training samples with large margins are downweighted (less than 1) and pairs with small margins are assigned a larger weight (equal to 1). By capping samples with very large margins to a weight of 1, the algorithm prevents explosion and diversion in training.

This coefficient is then used to determine the adaptive weight, $w(h_\phi)$.

$$w(h_\phi) = \begin{cases} \frac{\sigma(c(|h_\phi|) * h_\phi)}{\sigma(h_\phi)}, & \text{if } h_\phi > -\tau \\ 1, & \text{if } h_\phi \leq -\tau \end{cases}$$

The MADPO loss, $\mathcal{L}_{\text{MADPO}}$, is $\mathcal{L}_{\text{DPO}}$ re-weighted by the margin-dependent weight function.[5]

$$\mathcal{L}_{\text{MADPO}}(\theta, \phi; x, y_w, y_l) = w(h_\phi(x, y_w, y_l))\mathcal{L}_{\text{DPO}}(\theta; x, y_w, y_l)$$

Our MADPO hyperparameters are $c_{\min} = 0.1$, $c_{\max} = 2.0$, $\tau = 1.0$, and $\lambda = 1.0$. We train MADPO for 3 epochs with the SFT (improved) model checkpoints and a learning rate of 1×10^{-5} .

Symmetric Augmentation + DPO for Paraphrase Detection. To further boost paraphrase detection performance, we combined symmetric data augmentation with DPO. Starting from an SFT model already trained with symmetric augmentation, we ran DPO on top of it also with symmetric augmentation enabled during preference pair construction. Finally, we investigated the effect of the DPO regularization coefficient β to study its influence on model alignment.

5 Experiments

5.1 Data

We use the datasets provided by or constructed from the default project:

- **Stanford Sentiment Treebank (SST):** 11,855 sentences extracted from movie reviews.[9] Our split: 8,544 train / 1,101 dev / 2,210 test examples (for the base sentiment classification task).
- **CFIMDB:** 2,434 positive and negative movie reviews. Our split: 1,701 train / 245 dev / 488 test examples (for the base sentiment task).
- **Quora dataset:** 400,000 question pairs with labels depicting whether the sentences are paraphrases of one another. Our split: 141,506 training / 20,215 dev / 40,431 test examples with binary labels (for the paraphrase detection task).
- **Shakespeare Sonnets:** 154 of Shakespeare’s sonnets. Our split: 143 training / 12 test sonnets (for the sonnet generation task).
- **Pairwise preference data:** For DPO, we construct pairwise preference data automatically from the SFT model’s outputs, as described in the Approach section, so no additional data collection is required.

5.2 Evaluation method

For sentiment classification, we report accuracy on the dev set. For paraphrase detection, we report both accuracy and macro-averaged F1 on the dev set and the accuracy on the final test set. For sonnet generation, we report the chrF score [10] on both the dev sets and final test set, which measures character n-gram overlap between generated and reference sonnets. We also perform qualitative evaluation, examining whether DPO and MADPO-trained sonnets better adhere to the ABAB CDCD EFEF GG rhyme scheme and produce more Shakespeare-like language. For paraphrase detection, qualitative evaluation includes error analysis over true and false positives and negatives. 5

5.3 Experimental details

All experiments run on a single NVIDIA L4 GPU.

Sentiment Classification: GPT-2 models were trained using the default code parameters with minor adjustments. The last linear layer and full model used learning rates of 1e-3 and 1e-5 respectively. SST and CFIMDB datasets used batch sizes of 64 and 8, respectively.

Paraphrase Detection : Baseline SFT used GPT-2 with a linear classification head, with different learning rates, as above. **Symmetric augmentation** doubled the training data by adding reversed pairs. **LoRA** [6] (implemented from scratch) was configured with rank 8 and applied to reduce trainable parameters to 0.26%. The per-epoch training time decreased to 60% of full fine-tuning, but the accuracy had a slight drop. Still using symmetric augmentation, we replaced the standard linear classification head with a **two-layer feedforward head**. Hidden states are projected from $d \rightarrow d/2$ with GELU activation and 0.1 dropout, then mapped to the two output classes. **Ensemble averaging** was applied by averaging the weights of two checkpoints from the same training run that achieved identical performance on the development set at different epochs.

DPO was applied on top of the symmetrically-augmented SFT checkpoint for 3 epochs with learning rate 1×10^{-6} , batch size 8, 4 samples per prompt for rejected completions (temperature 1.2, top- p 0.9), and a subsample of 10,000 preference pairs. For both sonnet generation and paraphrase detection, $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$ was swept and the best checkpoint selected based on development metrics. DPO was also applied to the ensemble-averaged SFT model under the same settings except $\beta = 0.1$.

Sonnet Generation : For **SFT**, we train for 10 epochs with a learning rate of 1×10^{-5} , batch size of 8, and use the AdamW optimizer. The **improved SFT** model uses the same hyperparameters but freezes the first 8 of 12 transformer layers and adds early stopping with patience 3. **DPO training for sonnet generation** uses 3 epochs, $\beta = 0.1$, and a learning rate of 1×10^{-5} . Similarly, **MADPO training for sonnet generation** uses 3 epochs with the SFT (improved) model checkpoints and a learning rate of 1×10^{-5} . Across the **sonnet generation** experiments, we used top-k sampling with $k = 50$ and repetition penalty $\alpha = 1.2$. As mentioned above, we swept $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$ and selected the best checkpoint by dev accuracy.

The **"improved v2" SFT** model was trained to generate a single sonnet line given previous three lines. A prompt asking the model explicitly to generate the next line of the sonnet was used. We unfroze all transformer layers and set the learning rate to 1×10^{-4} . The best performance on development set was achieved at epoch 8.

5.4 Results

Table 1 shows sentiment analysis results. Table 2 reports paraphrase detection performance with different settings, Table 3 reports test leaderboard scores, and Table 4 summarizes sonnet generation results. Figure 1 shows the results of a systematic search over the DPO regularization coefficient β for sonnet generation, and Figure 2 shows the corresponding sweep for paraphrase detection.

Table 1: Sentiment analysis dev accuracy.

Model	SST Dev Acc	CFIMDB Dev Acc
Last linear layer	0.462	0.861
Full model	0.513	0.976

Table 2: Paraphrase detection experiments, results on dev set. For LoRA the results were obtained with Symmetric Augmentation enabled, but we trained only 10 epochs. The "+" sign indicates stacking with the previous method and it is recursive. [†]DPO rows use symmetric augmentation for both SFT and DPO stages.

Model	Rel. Time/Epoch	Epochs for Best Model	Acc	F1
Baseline	$1 \times$	10	0.895	0.887
DPO ($\beta = 0.1$)	$1.05 \times$	10	0.898	0.891
LoRA	$0.6 \times$	10*	0.880	0.872
Symm. Augm.	$2 \times$	10	0.899	0.893
2 Layer Head+	$2 \times$	5, 8	0.900	0.894
Ensemble averaging+	n/a	n/a	0.901	0.895
DPO + Ensemble averaging+	$2.1 \times$	8+3	0.902	0.896
Symm. Augm. + DPO ($\beta = 0$) [†]	$2.1 \times$	10+3	0.902	0.896
Symm. Augm. + DPO ($\beta = 0.25$) [†]	$2.1 \times$	10+3	0.902	0.896
Symm. Augm. + DPO ($\beta = 0.5$) [†]	$2.1 \times$	10+3	0.902	0.897
Symm. Augm. + DPO ($\beta = 0.75$) [†]	$2.1 \times$	10+3	0.902	0.896
Symm. Augm. + DPO ($\beta = 1$) [†]	$2.1 \times$	10+3	0.902	0.896

We also attempted other experiments that did not work as intended. For example, we trained the sonnet generation model with special tokens indicating line position, but the model repeatedly generated the special token instead of meaningful lines.

Table 3: **Test leaderboard** scores for best performing models. Baseline test scores were based on full fine-tuning and provided by course staff.

Task	Best Model	Test Score	Baseline	Metric
Paraphrase Detection	Symm. Augm. + DPO ($\beta = 0.25$)	0.902	0.856	Accuracy
Sonnet Generation	SFT Improved + DPO ($\beta = 0$)	41.638	41.208	chrF
Sonnet Generation	SFT "improved v2"	42.712	41.208	chrF

Table 4: Sonnet generation chrF scores on dev set.

Model	chrF Score
SFT (baseline)	41.25
SFT (improved)	38.72
DPO (SFT baseline; $\beta = 0.1$)	41.25
DPO (SFT improved; $\beta = 0.1$)	40.42
MADPO (SFT baseline; $\beta = 0.1$)	40.39
MADPO (SFT improved; $\beta = 0.1$)	40.72
DPO (SFT improved; $\beta = 0$)	41.88
DPO (SFT improved; $\beta = 0.25$)	41.29
DPO (SFT improved; $\beta = 0.50$)	40.75
DPO (SFT improved; $\beta = 0.75$)	41.25
DPO (SFT improved; $\beta = 1$)	41.06
SFT "improved v2"	43.17

6 Analysis

SFT. The results obtained by the improved SFT models were expected. Overall data augmentation proved to be a strong technique for both tasks. For paraphrase detection both the two layer head and ensemble averaging provided small gains that stacked.

DPO. After exploring $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$ for DPO using the SFT (improved) model for sonnet generation, $\beta = 0$ resulted in the strongest dev score of 41.88 and a 41.638 chrF score on the test leaderboards. This suggests that the model benefits from minimal constraint and that, for creative generation, staying close to the original SFT reference policy may actively hurt output quality.

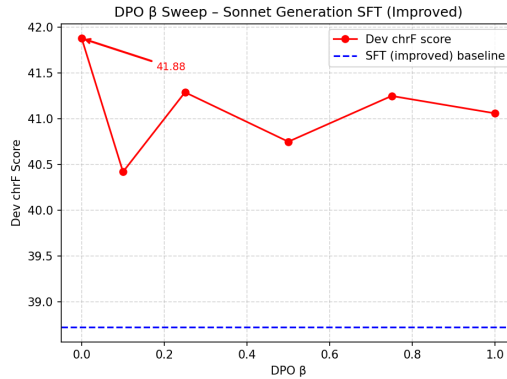


Figure 1: DPO β sweep for sonnet generation (SFT improved). Dev chrF scores are shown across $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$, which $\beta = 0$ achieving the highest chrF (41.88).

The same systematic search over β for DPO on the symmetrically-augmented paraphrase model (Figure 2) reveals a strikingly flat landscape: all five β values produce dev accuracies within ± 0.0003 of each other (range 0.9021–0.9024). The best accuracy is achieved at $\beta = 0.25$ (0.9024) and the best F1 at $\beta = 0.5$ (0.8965). This near-constant performance across β suggests that the benefit

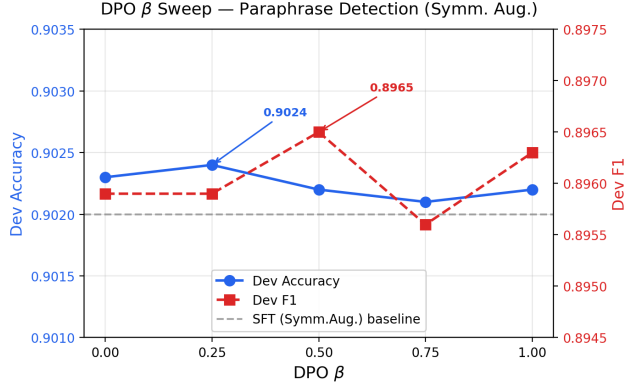


Figure 2: DPO β sweep for paraphrase detection (Symm. Augm. + DPO). Dev accuracy and F1 are shown across $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$. All configurations yield comparable performance, with $\beta = 0.25$ achieving the highest accuracy (0.9024) and $\beta = 0.5$ the highest F1 (0.8965).

of DPO for paraphrase detection comes primarily from the continued training on preference pairs rather than from the KL-regularization strength. The small but consistent improvement over the SFT+Symm. Augm. baseline (from 0.899 to 0.902 dev accuracy) indicates that even a single round of preference optimization on correctly/incorrectly classified pairs can tighten decision boundaries. The test leaderboard confirms this: our best model achieves 0.902 accuracy on the hidden test set, matching dev performance and demonstrating that the gains generalize.

Paraphrase detection. To understand where our best paraphrase model (Symm. Augm. + DPO, $\beta = 0.25$) succeeds and fails, we examined dev-set predictions qualitatively. The model handles clear paraphrases (e.g., “How do I get over depression?” vs. “How can I fight depression?”) and clear non-paraphrases reliably. Errors tend to cluster in two categories: (1) *lexical overlap traps*, where surface similarity masks semantic divergence (e.g., “What is the best way to learn Python?” vs. “What is the best way to learn Java?”), and (2) *subtle paraphrases* with low lexical overlap that require deeper reasoning. Symmetric augmentation helps with category (2) by ensuring the model sees both orderings, reducing positional bias. DPO further refines the decision boundary by up-weighting examples the SFT model found ambiguous. Despite these improvements, the gains over the SFT+Symm. Augm. baseline are modest ($\sim 0.3\%$ accuracy), suggesting that GPT-2’s representational capacity is the primary bottleneck for this task. To supplement this analysis, the confusion matrix in Table 5 reveals DPO’s dev gains are asymmetric. DPO improves recall by correctly identifying 430 additional paraphrase pairs, at the cost of 122 additional false positives, yielding a net gain in both accuracy and F1. This pattern makes intuitive sense for DPO. By training explicitly on preferred vs. rejected pairs, the model becomes more confident in labeling paraphrases, shifting the decision boundary in a way that recovers false negatives at a small cost to specificity.

Table 5: SFT vs. Best DPO Model: Paraphrase Detection Performance. TP, TN, FP, FN reported as raw counts over the dev set.

	SFT	Symm. Augm. + DPO ($\beta = 0.25$)	Change
True Positive	12,773	13,203	+430
True Negative	23,402	23,280	-122
False Positive	2,135	2,257	+122
False Negative	2,119	1,689	-430

Sonnet generation. We evaluated the generated sonnets of the best DPO model (SFT improved; $\beta = 0$) using human-defined qualitative metrics for rhyme scheme, stylistic resemblance to Shakespeare, and overall coherence. Each of these metrics was scored on a scale of 1 to 5 and averaged for each sonnet. Firstly, rhyme adherence measures how closely the output follows the ABAB CDCD EFEF GG structure. This metric is assessed by manually checking the rhymes at each expected position. Secondly, stylistic resemblance captures the syntax and thematic content relative to authentic Shakespeare sonnets. Lastly, coherence measures whether the poem maintains a clear meaning

throughout. In the dev set predictions, the ABAB CDCD EFEF GG structure is never fully achieved. In most instances, the model produces at most one partial rhyming pair before drifting the scheme entirely. In nearly every sonnet, coherence drops sharply after line 4 or 5, suggesting the model struggles to maintain consistency. Our analysis also revealed that several generated sonnets violate the 14-line constraint significantly by either being too short or too long, while others end abruptly mid-sentence. Across the dev set, individual metric averages were 1.4 for rhyme adherence, 2.8 for stylistic resemblance, and 2.3 for coherence, resulting in an overall average of 2.2 out of 5. Overall, the sonnets show surface-level similarity with Shakespeare’s style but consistently fall short at the structural and semantic level. These findings suggest the model captures local stylistic patterns but struggles with long-range structural planning. Some potential improvements may be explicit conditioning on rhyme structure or constrained decoding. Larger models such as GPT-2 Medium or Large, which can track context across longer sequences may also help address these limitations.

7 Conclusion

In this project we investigated whether Direct Preference Optimization can improve the performance of GPT-2 beyond standard supervised fine-tuning for sonnet generation and paraphrase detection. Starting from the provided baseline, we first explored a series of improvements to SFT, including data augmentation, architectural modifications to the classification head, ensemble averaging and improved decoding strategies. These improvements significantly strengthened the underlying SFT models. For paraphrase detection the stack of three techniques produced substantial gains over the baseline to accuracy of 0.901 (dev) and 0.900 (test). For sonnet generation, our strongest SFT configuration achieved a chrF score of 43.17 (dev) and 42.71 (test).

Applying DPO on top of SFT models further improved performance relative to the corresponding baselines. In particular, for paraphrase detection, combining symmetric data augmentation with DPO yielded the best accuracy overall, of 0.902 on both dev and test sets, a consistent improvement over the SFT baseline (0.895) and SFT+augmentation (0.899). For sonnet generation, applying DPO with low regularization improves performance relative to the corresponding SFT baseline, suggesting that weaker constraints on the reference policy may be beneficial for creative generation tasks. A β sweep on paraphrase detection reveals a near-flat landscape, suggesting that for discriminative tasks the main benefit of DPO lies in continued exposure to preference pairs rather than in the specific KL penalty strength.

Limitations include the modest absolute gain on paraphrase detection (constrained by GPT-2’s capacity), the small sonnet corpus limiting DPO’s full potential, and the lack of human evaluation at scale. Future work could explore MADPO for paraphrase detection, larger models (GPT-2 Medium/Large), and richer preference data from human annotators.

Team contributions

Nicolas implemented the DPO training pipeline for paraphrase detection (`dpo_training.py`), the symmetric augmentation extension for DPO stages, designed and ran the β sweep for paraphrase detection, generated paraphrase dev/test predictions, and contributed to the sonnet generation DPO pipeline.

Anna implemented the GPT-2 model (`gpt2.py`, `attention.py`, `gpt2_layer.py`) and trained it for sentiment classification for both the SST and the CFIMDB datasets. Anna also implemented the MADPO extension, generated paraphrase dev/test predictions for DPO and MADPO, and ran the β sweep for sonnet generation.

Cristian implemented the AdamW optimizer and was involved in the initial experiments on the IMDB database. He also implemented LoRA (Low-Rank Adaptation) from scratch. For paraphrase detection he introduced three improvements that stacked (symmetric data augmentation, two layer classification head and ensemble averaging). For sonnet generation, using the available code, he implemented the "improved v2" version.

References

- [1] 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. Training language models to follow instructions with human feedback, 2022.
- [2] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.
- [3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [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] Hyung Gyu Rho. Margin adaptive DPO: Leveraging reward model for granular control in preference optimization, 2025.
- [6] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: low-rank adaptation of large language models, 2021.
- [7] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019. Accessed: 2024-11-15.
- [8] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.
- [9] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.
- [10] Maja Popovic. chrF: character n-gram f-score for automatic mt evaluation. In *WMT@EMNLP*, 2015.