

DPO for Structural Sonnet Generation and Paraphrase Detection with GPT-2

Stanford CS224N - Default Final Project

Jason Amsler

Department of Computer Science
Stanford University
jamsler@stanford.edu

Diego Bustamante

Department of Computer Science
Stanford University
diegobus@stanford.edu

Daniel Mottesi

Department of Computer Science
Stanford University
mottesid@stanford.edu

Abstract

Fine-tuning large language models with maximum-likelihood estimation (MLE) optimizes token-level prediction accuracy but does not directly encode structural or preference-based constraints. We investigate whether Direct Preference Optimization (DPO) [1] can improve downstream performance on two CS224N default project tasks—paraphrase detection and sonnet generation—when applied to GPT-2 [2]. We implement the core GPT-2 components within the provided project scaffolding [3], establish competitive MLE baselines (SST 5-class accuracy 0.509, CFIMDB binary accuracy 0.984, paraphrase dev accuracy 0.900, sonnet chrF 42.0), and then apply DPO with synthetically constructed preference pairs. For sonnet generation, we design a novel structural preference signal contrasting continuations that satisfy Shakespearean sonnet form constraints (exactly 11 continuation lines, 5–10 words per line) against structurally flawed outputs. DPO with a line-count-only signal achieves the highest chrF of 43.0 (+1.0 over MLE), while a combined line-count-plus-word-count signal yields the first structurally perfect sonnets (16.7% exact struct rate) and dramatically improves valid line percentage (60.6% $\rightarrow$ 92.7%), at a modest chrF cost. For paraphrase detection, DPO on yes/no token preferences did not improve over the MLE baseline. Our results demonstrate that DPO can encode domain-specific structural constraints that MLE cannot express, but the quality of the preference signal and the choice of β critically determine the tradeoff between fluency and structural adherence.

1 Key Information

- TA mentor: Fang Wu
- External collaborators (if no, indicate “No”): No
- External mentor (if no, indicate “No”): No
- Sharing project (if no, indicate “No”): No
- Late Days (if you are using any):

Team member	Late days remaining
Diego Bustamante	3
Jason Amsler	2
Daniel Mottes	0
Total pooled	5
Team size	3
Late days applied for the report	1

2 Introduction

Large language models (LLMs) pretrained with next-token prediction learn broad linguistic competence from internet-scale corpora, but aligning their outputs to task-specific objectives—such as stylistic fidelity, structural correctness, or classification accuracy—requires additional fine-tuning. Standard maximum-likelihood estimation (MLE) fine-tuning optimizes a token-level cross-entropy objective, which encourages the model to reproduce the statistical distribution of training data. While effective for many tasks, MLE provides no direct mechanism to penalize outputs that violate higher-level structural constraints or to express preferences over complete outputs.

Reinforcement learning from human feedback (RLHF) [4] addresses this gap by training a separate reward model on human preference data and optimizing the policy using PPO [5], but this pipeline is computationally expensive and difficult to stabilize. Direct Preference Optimization (DPO) [1] offers an elegant alternative: by rearranging the RLHF objective, DPO derives a simple binary cross-entropy loss over preference pairs that bypasses the reward model and RL loop entirely, while retaining an implicit KL-divergence constraint that prevents the policy from drifting too far from a reference model.

In this work, we investigate whether replacing MLE fine-tuning with DPO improves performance on the two CS224N default project tasks: sentiment paraphrase detection on Quora Question Pairs [6], and Shakespearean sonnet generation. We first implement the core GPT-2 architecture [2] and establish competitive MLE baselines with careful hyperparameter tuning. We then design task-specific preference pair construction strategies and apply DPO fine-tuning. Our primary contribution is demonstrating that DPO can encode structural constraints for sonnet generation—specifically, the 11-line continuation length and 5–10 words-per-line requirements of Shakespearean sonnet form—that MLE fine-tuning fundamentally cannot express. We conduct ablation studies across two preference signal designs (line-count-only vs. combined line-count-and-word-count) and three values of the DPO temperature parameter $\beta \in \{0.05, 0.1, 0.2\}$, revealing a systematic tradeoff between fluency preservation and structural adherence.

3 Related Work

Direct Preference Optimization. Rafailov et al. [1] showed that the standard KL-constrained RLHF objective admits a closed-form optimal policy, enabling a mathematical rearrangement that eliminates the reward model entirely. The resulting DPO loss is a binary cross-entropy over preference pairs, parameterized by a temperature β that controls the strength of the implicit KL constraint to the reference policy. Plugging the reward reparameterization into the Bradley–Terry preference model [7], the partition function cancels, yielding:

$$\mathcal{L}_{\text{DPO}}(\pi_{\theta}; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_{\theta}(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_{\theta}(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right) \right] \quad (1)$$

where y_w and y_l are the preferred and dispreferred completions. DPO matched or exceeded PPO-based RLHF on controlled sentiment generation, summarization, and dialogue tasks. Critically for our work, the sentiment experiment in [1] constructs preference pairs synthetically using a classifier, demonstrating that DPO can work without human preference annotation.

GPT-2 and autoregressive language models. GPT-2 [2] is a Transformer-based [8] autoregressive language model pretrained on WebText. It uses causal (left-to-right) self-attention, learned positional embeddings, and pre-LayerNorm residual blocks. We use the 124M-parameter gpt2 variant as our base model, loading pretrained weights from HuggingFace and fine-tuning for each downstream task.

Decoding strategies. For sonnet generation, we employ nucleus (top- p) sampling [9] with temperature scaling, which truncates the probability distribution to the smallest set of tokens whose cumulative probability exceeds a threshold p . This avoids the repetitive and degenerate outputs associated with greedy or pure sampling decoding while maintaining diversity.

4 Approach

Our approach proceeds in two stages: (1) establishing MLE baselines for all three tasks, and (2) applying DPO fine-tuning with task-specific synthetic preference pairs.

4.1 GPT-2 Implementation

We implement GPT-2 following the default project specification [3]. Each of the 12 Transformer layers applies pre-LayerNorm, causal multi-head self-attention (12 heads, $d_k = 64$), and a position-wise feed-forward network ($d_{\text{ff}} = 3072$) with GELU activation, residual connections, and dropout. The causal attention mask is an upper-triangular matrix that prevents tokens from attending to future positions:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V \quad (2)$$

where $M_{ij} = -\infty$ for $j > i$. We load pretrained weights from HuggingFace, remapping the fused `c_attn` convolution to separate Q, K, V linear projections. We implement the AdamW optimizer [10, 11] with decoupled weight decay.

4.2 MLE Baselines

Sentiment classification. `GPT2SentimentClassifier` extracts the hidden state at the last non-padding token and applies a linear head mapping to 5 classes (SST) or 2 classes (CFIMDB). We train with cross-entropy loss, using $\text{lr} = 10^{-3}$ for last-linear-layer fine-tuning and $\text{lr} = 10^{-5}$ for full-model fine-tuning, for 10 epochs.

Paraphrase detection. `ParaphraseGPT` uses a cloze-style prompt format:

Is "{s1}" a paraphrase of "{s2}"? Answer "yes" or "no":

Rather than training a separate classification head, we use GPT-2’s weight-tied language modeling head (`hidden_state_to_token`) to produce vocabulary logits at the last token position, then slice to the “yes” (token 8505) and “no” (token 3919) logit dimensions. This formulation enables a natural transition to DPO, which operates on token-level log-probabilities. We apply three key improvements over the starter code: (i) **data augmentation** via sentence-order swapping ($(s_1, s_2) \rightarrow (s_2, s_1)$) for symmetry regularization, (ii) **inverse-frequency class weighting** to address label imbalance, and (iii) a **linear warmup-then-decay learning rate schedule** (warmup ratio 0.06) with gradient clipping (max norm 1.0). We train for 7 epochs with $\text{lr} = 10^{-5}$ and batch size 16, selecting the checkpoint with best dev F1.

Sonnet generation. `SonnetGPT` wraps GPT-2 for autoregressive language modeling on full sonnet sequences. The forward pass returns logits at all positions (not just the last token), and training minimizes cross-entropy on shifted targets with padding masked out. During generation, we use top- p sampling ($p = 0.9$) with temperature $T = 1.2$, respecting GPT-2’s 1024-token context window. We add **early stopping on held-out chrF**: at each epoch, we generate continuations for a development set of sonnets, compute corpus-level chrF [12] against ground-truth continuations, and save the checkpoint only if chrF improves (patience 2, $\delta = 0.0$). This prevents overfitting to the training sonnets and selects the generation-quality optimum rather than the training-loss optimum.

4.3 DPO Extension

DPO loss implementation. We implement the DPO objective (Equation 1) in `dpo_utils.py`. For sequence-level tasks (sonnets), `compute_sequence_logprobs` computes the sum of token-level log-probabilities over the completion span only, using a binary mask that is 1.0 for continuation tokens and 0.0 for prompt and padding tokens. For token-level tasks (paraphrase),

`compute_single_token_logprob` extracts $\log P(\text{target token} \mid \text{context})$ at the last position. The reference model π_{ref} is a frozen copy of the MLE-finetuned checkpoint; the policy π_{θ} is initialized from the same checkpoint and updated via DPO.

Sonnet preference pairs. We construct preference pairs from the MLE-finetuned base model by sampling multiple candidate continuations per training prompt. Preferred structure was defined according to the formal structure of a Shakespearean sonnet, which should contain 14 lines total, adhere to iambic pentameter on each line, and match a specific rhyme scheme per line. While line-count can be trivially enforced, iambic pentameter was approximated as having a line length of 5–10 words, which roughly maps to appropriate syllable count. Due to difference in old english pronunciations, existing methods for rhyme verification were unsuitable to create an appropriate signal for preference pairs and thus rhyme scheme was omitted as a preference signal. Candidate completions were generated with temperatures randomly sampled from the range 0.9–1.2 for increased output diversity and with top- p sampling ($p = 0.9$). We define two ablation conditions:

- **Combined (line count + word count):** *Chosen* candidates must have exactly 11 continuation lines with every line containing 5–10 words, and chrF ≥ 20.0 against the ground truth. *Rejected* candidates exhibit structural flaws (wrong line count or word-count violations), ranked by structural severity (line-count deviation as primary key, fraction of extreme word counts as secondary). This yields 147 pairs from 131 training prompts.
- **Line count only:** *Chosen* candidates have exactly 11 lines (word count unconstrained); *rejected* candidates have incorrect line counts. This yields 113 pairs from 131 training prompts, providing a cleaner but narrower learning signal focused solely on stopping behavior.

We generate up to 10 candidates per batch with up to 10–20 retry batches per prompt, accumulating candidates until at least one valid pair is formed. A truncation heuristic checks whether the first 11 lines of an overlong candidate are individually valid, recovering additional chosen examples.

Paraphrase DPO. For paraphrase detection, we construct preference pairs directly from ground-truth labels: for each prompt, y_w is the correct label token (“yes” or “no”) and y_l is the incorrect token. The DPO model (ParaphraseDPOGPT) uses the same weight-tied language modeling head as the MLE model but is trained with the single-token DPO loss at $\beta = 0.1$.

5 Experiments

5.1 Data

We use the datasets provided by the default project. For paraphrase, we use Quora Question Pairs [6] with 141,506/20,215/40,431 binary paraphrase pairs. For sonnet generation, the training corpus consists of 154 Shakespeare sonnets; a held-out development set of sonnet prompts (first 3 lines) with ground-truth continuations is used for chrF evaluation.

For DPO, we construct synthetic preference pairs as described in Section 4.3. The combined sonnet preference set contains 147 pairs (max 2 per prompt) and the line-count-only set contains 113 pairs.

5.2 Evaluation Method

Paraphrase. Paraphrase detection is evaluated by both accuracy and macro F1 on the Quora dev set, with F1 used for checkpoint selection to account for class imbalance.

Sonnet generation. We use four complementary metrics to capture both fluency and structural quality:

- **chrF** [12]: corpus-level character n -gram F-score against ground-truth continuations, measuring textual similarity and Shakespearean style fidelity.
- **Exact Struct %**: the fraction of generated sonnets that are *structurally perfect*—exactly 11 continuation lines, each with 5–10 words. This is a binary per-sonnet metric.

- **Avg. continuation lines:** the mean number of non-empty continuation lines across generated sonnets. The target is 11.
- **Valid line %:** the fraction of individual continuation lines (across all sonnets) that have 5–10 words, measuring word-count adherence at the line level.

These metrics decompose sonnet quality into orthogonal dimensions: chrF captures fluency/style, while the structural metrics capture form adherence that MLE training cannot directly optimize.

5.3 Experimental Details

All models use the gpt2 (124M parameter) architecture. MLE baselines are trained on a single GPU with AdamW. A critical finding during baseline development was that learning rate must be tuned separately for each fine-tuning mode: $\text{lr} = 10^{-3}$ for last-linear-layer and $\text{lr} = 10^{-5}$ for full-model fine-tuning; the starter code defaults produced substantially lower accuracy.

Paraphrase MLE training uses $\text{lr} = 10^{-5}$, batch size 16, 7 epochs, warmup ratio 0.06, and gradient clipping (max norm 1.0). The final recipe incorporates several substantive changes relative to the starter code: switching from a separate MLP classification head to GPT-2’s weight-tied vocabulary head sliced to “yes”/“no” logits; aligning train and inference prompt formats; symmetry augmentation via sentence-order swapping; linear warmup and decay scheduling; gradient clipping; inverse-frequency class weighting; and checkpoint selection by dev F1 rather than accuracy. Together these changes yield 0.900 dev accuracy (macro F1 0.894), up from 0.896 under the earlier recipe. We did not run controlled ablations isolating the contribution of each change.

For DPO training, all runs use $\text{lr} = 10^{-6}$ (an order of magnitude below MLE to prevent catastrophic drift), batch size 4, and the AdamW optimizer with no weight decay. The reference model is the frozen MLE-finetuned checkpoint. We sweep the DPO temperature $\beta \in \{0.05, 0.1, 0.2\}$ across both preference signal ablations (combined and line-count-only), training for 100 epochs with evaluation every 5 epochs. Sonnet DPO runs use generation parameters $T = 1.2$, $p = 0.9$.

For paraphrase DPO, we train for 10 epochs with $\beta = 0.1$ and $\text{lr} = 10^{-5}$, batch size 16.

5.4 Results

Sentiment baselines. Table 1 shows sentiment dev accuracy. Our results closely match or exceed the handout targets [3], with full fine-tuning slightly exceeding the CFIMDB target (0.984 vs. 0.976). We do not apply DPO to sentiment, as the classification task is well-served by MLE fine-tuning.

Table 1: Sentiment dev accuracy vs. baseline targets.

Fine-tuning	SST (5-class)		CFIMDB (binary)	
	Target	Ours	Target	Ours
Last-linear-layer	0.462	0.460	0.861	0.841
Full model	0.513	0.509	0.976	0.984

Paraphrase baselines and DPO. Our updated training recipe (Section 5.3) yields 0.900 dev accuracy (macro F1 0.894), up from 0.896 under the earlier configuration. DPO fine-tuning with $\beta = 0.1$ on yes/no token preferences did not improve over this MLE baseline. We attribute this to the nature of the task: the paraphrase objective reduces to a binary classification over a two-token output space, which MLE cross-entropy already optimizes directly. DPO’s advantage lies in expressing preferences over *complete*, *open-ended* outputs; for a two-token decision, the MLE and DPO objectives provide essentially the same learning signal. Given this null result and limited compute budget, we did not pursue further DPO ablations for paraphrase.

Sonnet generation: MLE vs. DPO. Table 2 presents our main sonnet results. The MLE baseline improves chrF from 40.1 (pretrained GPT-2) to 42.0 and nearly doubles valid line percentage (33.3% $\rightarrow$ 60.6%), confirming it learns Shakespearean style. However, the MLE model overshoots the target of 11 continuation lines (avg. 13.8 lines) and achieves 0% structural perfection rate—it never produces a single sonnet with exactly 11 lines all having 5–10 words.

Table 2: Sonnet generation results (dev set). Best epoch shown for each DPO run. $\uparrow$ indicates higher is better.

Model	chrF $\uparrow$	Struct % $\uparrow$	Avg. Lines	Valid Line % $\uparrow$
GPT-2 Pretrained	40.1	0.0%	6.0	33.3%
MLE Baseline	42.0	0.0%	13.8	60.6%
Combined $\beta = 0.05$ (best struct)	39.6	16.7%	~ 10	92.7%
Combined $\beta = 0.1$	40.9	8.3%	~ 10	87.1%
Combined $\beta = 0.2$	42.5	0.0%	10.4	72.0%
LineCount $\beta = 0.05$	42.8	0.0%	13.5	42.0%
LineCount $\beta = 0.1$ (best chrF)	43.0	0.0%	12.3	35.1%
LineCount $\beta = 0.2$	42.7	0.0%	12.4	44.3%

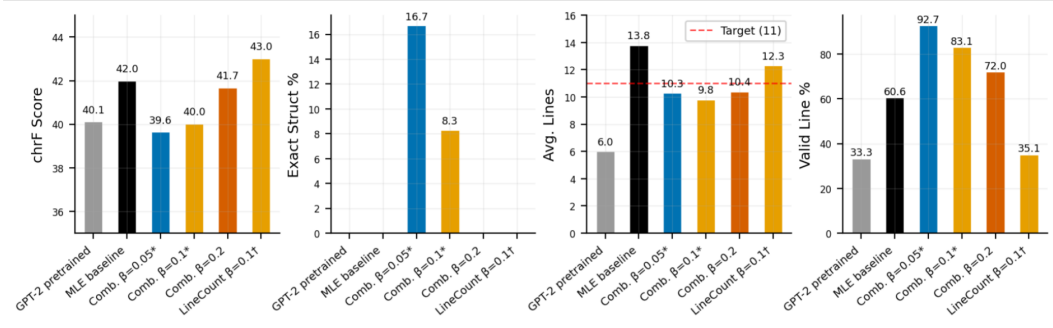


Figure 1: **DPO Improves Sonnet Structure While Preserving Language Quality.** Comparison of pretrained GPT-2, MLE fine-tuned, and DPO models across four evaluation metrics: chrF similarity to reference sonnets, exact structural validity (11 continuation lines with 5–10 words each), average continuation length, and percentage of valid line lengths. MLE improves stylistic similarity (chrF 40.1 to 42.0) but produces overly long continuations (avg. 13.8 lines) and never generates perfectly structured sonnets. DPO trained on structural preference pairs substantially improves structural adherence—achieving up to 16.7% perfectly structured sonnets and increasing valid-line rates to >90%—while maintaining competitive chrF scores.

DPO with the **combined signal** dramatically improves structural metrics: at $\beta = 0.05$, the model achieves 16.7% structural perfection (the only configuration where any sonnets achieve perfect form) and 92.7% valid line percentage, but at a chrF cost (42.0 $\rightarrow$ 39.6). At $\beta = 0.2$, the KL constraint is too conservative—chrF is preserved but structural perfection never exceeds 0%.

DPO with the **line-count-only signal** preserves and even improves chrF: $\beta = 0.1$ achieves the highest chrF of any configuration (43.0), while gently pushing average continuation lines from 13.8 toward 11 (reaching 12.3). However, without word-count supervision, valid line percentage actually decreases and structural perfection remains at 0%.

6 Analysis

MLE learns style, DPO learns structure. Our results reveal a clean separation of what each training objective captures. MLE fine-tuning teaches the model to produce text that *sounds* Shakespearean, improving chrF and word-level patterns. However, it has no mechanism to penalize outputs that overshoot the 11-line target or violate per-line word counts. DPO with curated structural preference pairs can push the model toward form constraints that MLE cannot express: the combined signal at $\beta = 0.05$ produces the first structurally perfect sonnets and pushes valid line percentage from 60.6% to 92.7%.

The preference signal determines the tradeoff. Our ablation between combined and line-count-only preference pairs reveals that the choice of signal critically determines what the model learns. The line-count-only signal provides a clean, low-noise learning target (stop at 11 lines) that preserves and even improves fluency, achieving the highest chrF. The combined signal teaches both line count and word count simultaneously, but the multi-constraint signal is noisier—rejected candidates differ from chosen ones in multiple ways, making it harder for the model to isolate the relevant structural features. This suggests a potential curriculum approach: first train on line-count pairs to learn stopping behavior, then fine-tune on combined pairs to tighten word counts.

β controls the fluency–structure tradeoff. Across both preference signal types, we observe a consistent pattern: lower β allows more aggressive optimization of the preference signal (higher structural metrics) but permits greater drift from the reference model (lower chrF); higher β preserves fluency but limits structural learning. At $\beta = 0.05$ with the combined signal, average continuation lines collapses from 13.8 to 6.6 by epoch 99—the model overshoots from too many lines to too few, suggesting over-optimization. At $\beta = 0.2$, the model barely moves from the MLE baseline.

Training dynamics and off-policy concerns. Our preference pairs are generated once from the MLE base model and reused across all 100 DPO epochs (off-policy). As training progresses, the DPO loss collapses toward zero and the reward margin grows monotonically (reaching ~ 17 at epoch 100), indicating that the model has “solved” the static training pairs. This likely causes overfitting to the specific rejection patterns in the data rather than learning the general structural constraint. Standard RLHF pipelines regenerate rollouts from the current policy each epoch (on-policy) to keep the signal relevant. The β -scaled KL constraint partially mitigates this by limiting how far the policy can drift, and with only 113–147 pairs the signal is sparse enough that on-policy generation might not yield substantially different pairs.

Paraphrase DPO limitations. DPO did not improve paraphrase detection over MLE. We hypothesize this is because the yes/no classification task is already well-suited to cross-entropy optimization: the completion space is binary and the MLE objective directly maximizes the probability of the correct label. DPO’s advantage—encoding preferences over *complete outputs*—is most valuable when the output space is large and the desired structure cannot be captured by token-level likelihood, as in sonnet generation.

7 Conclusion

We investigated whether DPO can improve over MLE fine-tuning for three NLP tasks using GPT-2. Our main finding is that DPO can encode domain-specific structural constraints for sonnet generation that MLE fundamentally cannot express: the combined line-count-and-word-count preference signal produced the first structurally perfect Shakespearean sonnet continuations (16.7% struct rate) and dramatically improved word-count adherence (60.6% $\rightarrow$ 92.7% valid line percentage). A line-count-only signal achieved the highest overall chrF (43.0 vs. 42.0 MLE baseline) while gently correcting the line count toward the target.

However, DPO’s benefits are task-dependent: it did not improve paraphrase detection, where the binary output space is already well-served by MLE. The results highlight that the quality and design of the preference signal—not just the training algorithm—determine what the model learns, and that the β parameter provides a principled knob for trading off fluency against structural adherence.

Limitations and future work. Our DPO training is off-policy: preference pairs are generated once and reused, which may lead to overfitting as the policy diverges from the distribution that generated the pairs. Iterative DPO with on-policy pair regeneration is a natural next step. Additionally, a curriculum approach—first training on line-count pairs, then on combined pairs—could achieve strong structural metrics without the chrF degradation we observe with the combined signal alone. Finally, our structural metrics are defined heuristically; incorporating learned quality metrics or human evaluation would strengthen the evaluation.

Team Contributions

Jason Amsler implemented the Adam optimizer and sentiment classifier, generated last-linear-layer and full-model predictions for CFIMDB and SST, implemented the paraphrase detection pipeline including the cloze-style prompt formulation, fixed paraphrase token IDs, led milestone write-up.

Diego Bustamante implemented the core GPT-2 model, causal self-attention, and transformer layers. Built the sentiment classification baseline. Implemented the DPO loss functions and utilities. Built the sonnet preference pair generation pipeline, designed the structural validity criteria, and ran all sonnet DPO experiments and ablations.

Daniel Mottesi implemented the sonnet generation pipeline including the SonnetGPT model, early stopping on held-out chrF, and the top- p sampling decoder. Added symmetry-based augmentation and optimization stabilizers to Paraphrase, as well as class-balanced loss weighting. Involved in experiment analysis and paper writing.

References

- [1] 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.
- [2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.
- [3] Stanford CS224N. CS 224N default final project: Build GPT-2. Course handout, 2025.
- [4] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 2017.
- [5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [6] Quora. First quora dataset release: Question pairs. <https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs>, 2017.
- [7] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs. *Biometrika*, 39:324–345, 1952.
- [8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.
- [9] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *International Conference on Learning Representations*, 2020.
- [10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [11] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [12] Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, 2015.

A Appendix

A.1 DPO Training Dynamics

Figure 2 shows the training loss and reward margin over 100 epochs for three sonnet DPO configurations. The combined-signal runs ($\beta = 0.05$ and $\beta = 0.1$) converge rapidly (loss $\rightarrow 0$ by epoch

10), while the line-count-only run ($\beta = 0.1$) starts with higher loss and converges more slowly. The reward margin $\mathbb{E}[r_w - r_l]$ grows monotonically for all runs, reaching ~ 17 at epoch 100, confirming that the static preference pairs become increasingly “solved” over training.

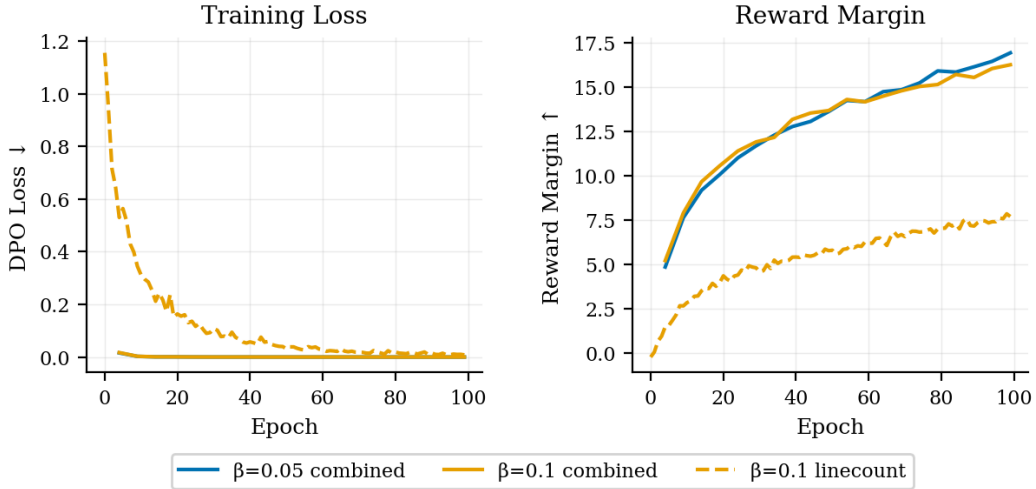


Figure 2: DPO training loss and reward margin over 100 epochs for combined ($\beta=0.05, \beta=0.1$) and line-count-only ($\beta=0.1$) configurations.

A.2 Preference Pair Statistics

The combined preference set (147 pairs) contrasts structurally perfect chosen candidates (mean 11.0 lines, 100% valid line rate) against rejected candidates that are approximately twice the target length (mean 21.8 lines, 75.6% valid line rate). The line-count-only set (113 pairs) provides a complementary signal: chosen candidates have exactly 11 lines but unconstrained word counts (57.5% valid line rate), while rejected candidates are consistently too short (mean 7.5 lines).

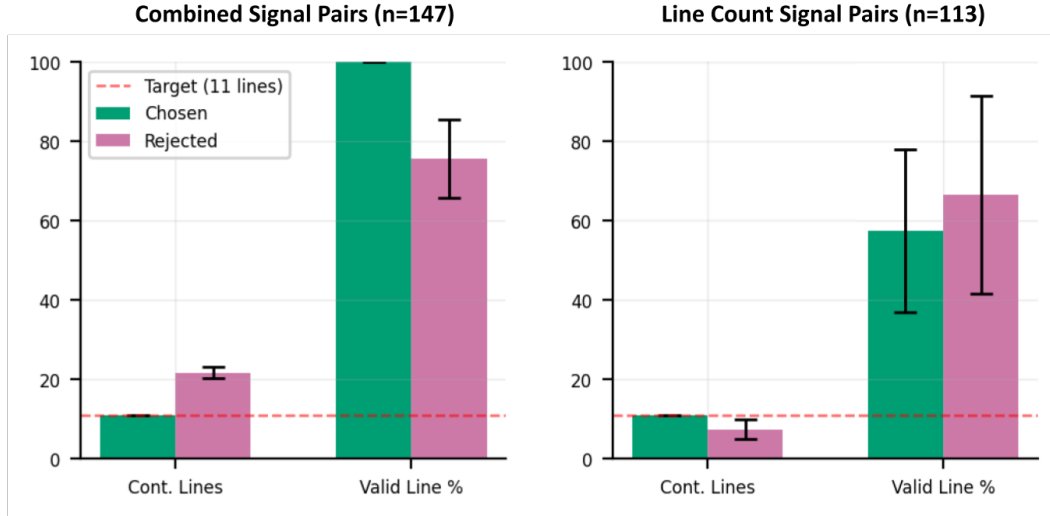


Figure 3: **Structural Signal in Preference Pair Datasets.** Mean structural statistics of the preference datasets used for DPO training with error bars indicating ± 1 std dev. Left: combined dataset (n=147 pairs) where chosen completions satisfy both line-count and word-count constraints. Rejected completions are dramatically longer and exhibit lower word-count validity, providing a multi-dimensional structural signal. Right: line-count-only dataset (n=113 pairs) where chosen completions enforce only the 11-line constraint, providing a simpler signal (continuations that are too short), isolating the effect of enforcing stopping behavior during generation.

A.3 Per-Epoch Evaluation Curves

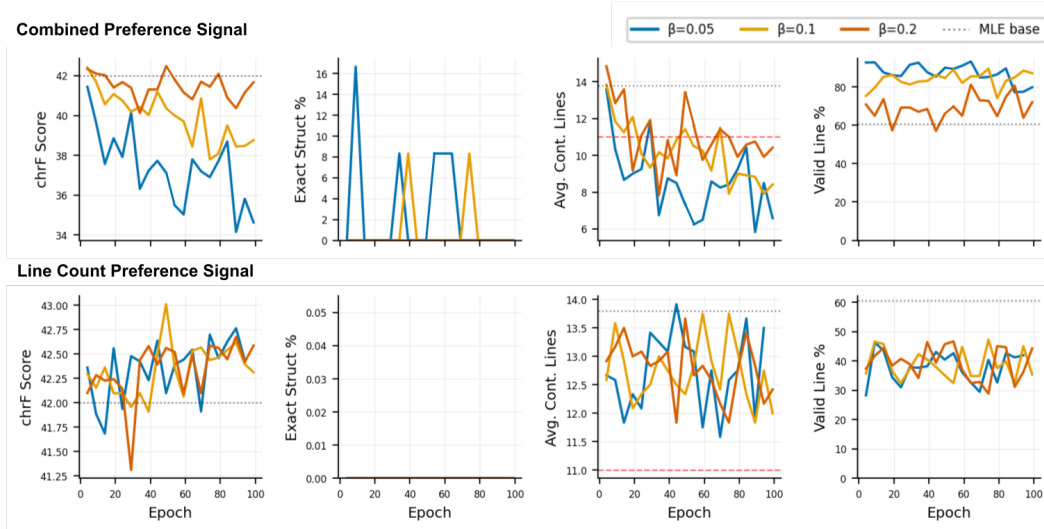


Figure 4: **Training Dynamics of Structural Preference Learning with DPO.** Per-epoch evaluation metrics for DPO models trained with two preference signals: combined structural constraints (line count + word count) and line-count-only constraints, across $\beta \in \{0.05, 0.1, 0.2\}$. Columns show chrF score, exact structural validity rate, average continuation length, and percentage of valid lines. Combined training rapidly improves structural validity and word-count adherence but can degrade language quality when β is too small. In contrast, line-count-only training preserves chrF while slowly shifting continuation length toward the target.