

Direct Preference Optimization for Improving Sonnet Generation

Stanford CS224N Default Project

Gio Ty

Department of Computer Science
Stanford University
gioty@stanford.edu

Abstract

Large language models trained with maximum likelihood estimation (MLE) generally produce fluent text, but can sometimes struggle to regularly generate stylistically coherent creative writing. This paper investigates whether Direct Preference Optimization (DPO) can improve the quality of Shakespearean sonnets generated. Beginning with the baseline GPT model provided the CS224N default final project, synthetic preference datasets were constructed, and the model was fine-tuned using DPO instead of traditional reinforcement learning from human feedback. Multiple preference construction strategies were explored, including random pairings, log-probability ranking, and repetition-based heuristics. The experiments conducted show that DPO significantly improves generated quality over the baseline model on the CS224N sonnet generation leaderboard. The baseline model achieved a dev score of 35.40, while DPO training improved results to 42.01 using randomly constructed preference pairs. A β hyperparameter sweep was also performed to analyze the sensitivity of DPO training. The results obtained suggest that even simple synthetic preference signals can significantly improve creative text generation when optimized through the DPO objective. Although the paper also discusses other sub-tasks part of the CS224N default final project such as paraphrasing and sentiment classification, the focus is largely on improving sonnet generation through DPO.

1 Key Information to include

- Mentor: Shicheng Liu
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

For neural language models, creative text generation is a challenging task because success can depend on a multitude of factors such as style, structure, and aesthetic quality, not just grammatical correctness. In the complex task of poetry generation, these challenges are increased by strict structural constraints such as rhyme, meter and thematic coherence. Although modern transformer-based language models are capable of producing fluent text, models trained solely using MLE can produce repetitive or inconsistent outputs.

Recent work has indicated that preference based optimization can improve language model outputs through making models more aligned with human feedback. One approach is reinforcement learning from human feedback (RLHF); this entails initially training a reward model, followed by optimizing the language model using reinforcement learning. Although this may seem straightforward,

in practice, RLHF poses several challenges in its implementation and introduces instability in training.

Direct Preference Optimization (DPO) provides a simpler alternative. Instead of learning a reward model, DPO directly optimizes the language model using pairwise preference data, using the observation that language models implicitly represent a reward function through its log-probabilities. This allows for preference optimization using standard gradient-based training without the need for reinforcement learning.

This project investigates whether DPO can improve the downstream task of Shakespearean sonnet generation as outlined by the CS224N default final project. Starting from the provided GPT language model, multiple strategies for generating synthetic preference datasets were explored. The model was then trained using the DPO objective and evaluated using the chrF metric through the dev leaderboard. This paper also lightly touches the other tasks outlined in the default final project, including sentiment classification and paraphrase detection.

Overall, the contributions made from this paper are:

1. Implementing the baseline model for sentiment classification and paraphrase detection
2. Designing several ways to generate synthetic preference pairs
3. Evaluating the impact of preference construction and DPO hyperparameters
4. Demonstrating that DPO significantly improves generation quality over the baseline model

3 Related Work

3.1 Neural Language Models

Modern language generation systems rely heavily on transformer based architectures introduced by Vaswani et al. (2017) Vaswani et al. 2017. Autoregressive transformer models, such as GPT, generate text one token at a time while conditioning on prior context. These models are often trained using MLE on large text datasets. Although MLE training produces strong language modeling performance, it does not always optimize for downstream human preferences such as stylistic quality or coherence.

3.2 Poetry Generation

Prior work on poetry generation shows that good poetry generation depends not just on fluent language modeling, but also on handling poetic constraints such as rhyme and meter, maintaining global coherence across lines, and producing diverse outputs. Recent work emphasizes controllability and better evaluation beyond fluency or similarity to human work (Ghazvininejad et al. 2016) (Lau et al. 2018).

3.3 Reinforcement Learning from Human Feedback

RLHF has become one of the most popular methods for aligning language models with human preferences. The pipeline for this approach has been standardized over the years, generally entailing first pretraining a language model with MLE on a large text corpora, then creating a preference or reward model trained on human data. Finally, a policy model is optimized with reinforcement learning to produce outputs that receive higher rewards under the learned preference model Christiano et al. 2017.

However, RLHF also introduces several challenges. For instance, the reward model must be learned from potentially noisy and finite preference data, creating the risk of reward overoptimization against imperfections in the reward model. Moreover, the learning stage can be unstable and sensitive to optimization details, which requires the careful tuning of hyperparameters. Lastly, RLHF adds significant computational overhead because of the need for a separate reward model and running an RL procedure with the policy. The difficulties discussed here motivated a search for simpler preference optimization models which retain the advantages of RLHF Ghazvininejad et al. 2016.

3.4 Direct Preference Optimization

Introduced by Rafailov et al. (2023), Direct Preference Optimization (DPO) was proposed as a simpler alternative to standard RLHF. The key insight was that under the usual KL-constrained RLHF objective, the optimal policy can be written as a function of an implicit reward. Instead of explicitly fitting a reward model followed by optimizing a policy with reinforcement learning, the policy can be optimized directly on pairwise preference data using a binary classification-style objective. Through this, DPO treats the language model itself as implicitly representing the relevant reward differences between preferred and rejected responses.

The DPO loss for a prompt x , preferred response y_w , rejected response y_l , reference model π_{ref} , and policy π_θ can be expressed as:

$$\mathcal{L}_{DPO} = -\log \sigma \left(\beta \left(\log \pi_\theta(y_w | x) - \log \pi_\theta(y_l | x) - \log \pi_{ref}(y_w | x) + \log \pi_{ref}(y_l | x) \right) \right) \quad (1)$$

where β controls the strength of the preference optimization. Essentially, the objective increases the relative likelihood of preferred responses over rejected ones while regularizing. Because DPO avoids the RL optimization loop and explicit reward-model training, it is generally easier to implement and more stable than standard RLHF pipelines. In the paper cited, Rafailov et al. comment that DPO can match or outperform existing RLHF baselines while being lighter computationally and requiring less tuning (Rafailov et al. 2023).

4 Approach

This project studies three tasks part of the CS224N default final project: sentiment classification, paraphrase detection, and sonnet generation. the first two tasks evaluate the model’s ability to perform understanding, and the last one focuses on creative text generation. The main methodological contribution is to improve sonnet generation using DPO, while the sentiment and paraphrase tasks are baseline evaluations of the model architecture.

4.1 Model Architecture

All tasks used a shared transformer-based model inspired by the GPT architecture. The model consists of a stack of transformer decoder blocks with masked self-attention, allowing it to model text retrogressively. With a sequence of tokens $x_1, x_2, \dots, x_T$, the model predicts the next token using the conditional distribution

$$P(x_t | x_{<t}) = \text{softmax}(h_t W_o) \quad (2)$$

where h_t is the hidden representation outputted by the transformer, and W_o is the output projection matrix. MLE is used to pretrain the model, thereby maximizing the probability of observed sequences in the training data. This model serves as the backbone for all of the tasks outlined.

4.2 Sentiment Classification

Sentiment classification involves predicting the sentiment of a sentence. The training dataset consisted of movie reviews labeled positive or negative, and the problem is then framed as a binary classification task.

Given an input sentence s , the model computes contextualized token embeddings using the transformer encoder; the representation of the final token is then passed through a linear classification layer to produce a sentiment score:

$$P(y|s) = \text{softmax}(W h_s + b) \quad (3)$$

where h_s is the sentence representation and y is the sentiment label. The model is trained using cross-entropy loss.

4.3 Paraphrase Detection

For paraphrase detection, the formulation in the handout was followed. Rather than introducing a separate sentence-pair classification architecture, the task is cast as a cloze-style language modeling problem. Given two sentences s_1 and s_2 , the model predicts whether the second sentence is a paraphrase of the first by generating either "yes" or "no".

4.4 Sonnet Generation

Sonnet generation involves the generation of creative texts in the style of Shakespearean sonnets. The model receives the first few lines of a sonnet as a prompt and must generate the continuation. The baseline model generates text autoregressively; however, models trained only using MLE can produce outputs that are repetitive and lack the stylistic requirements of the task. Therefore, this motivates the use of DPO discussed below.

4.5 Direct Preference Optimization

To improve generation quality, DPO is used which replaces reinforcement learning with a simpler objective that directly optimizes the language model and preference comparison. Equation 1 in section 3.4 outlines the loss function used in this method, and the objective pushes the model towards outputs that are preferred while regularizing against the original distribution.

4.6 Synthetic Preference Construction

One of the main challenges in implementing DPO is obtaining high quality preference data. Human annotation is often used but is expensive, and no labels exist for the dataset used. For this project, synthetic preference pairs were constructed automatically. For each prompt in the training set, two candidate continuations using the baseline language model were produced, and a ranking function was then used to determine which continuation was considered the preferred output. Four strategies for constructing preferences were used:

1. Random Pairing

This is the simplest approach, which assigned preferences randomly between two generated continuations. Although this method does not incorporate any ranking or quality signal, it still provided contrastive training examples. Prior work has observed that even noisy preferences can improve language models when used with DPO.

2. Log-Probability Ranking

This method treats the continuation with higher language model log-probability as the preferred output. Higher probability sequences should correspond to more fluent or likely continuations according to the base model.

3. Repetition-Based Ranking

For this method, a heuristic is introduced which penalizes outputs with repeated words, bigrams, or lines. The continuation with lower repetition is considered the preferred response, and this approach attempts to encourage more diverse and coherent generations.

4. Structural Repetition Ranking

Lastly, the repetition heuristic is extended with structural constraints. In addition to penalizing repeated tokens and phrases, it also penalizes outputs with unusual line lengths or irregular structure

compared to typical Shakespearean sonnets.

Although these heuristics are relatively simple, they provide a way to generate large numbers of preference comparisons without using manual annotation.

5 Experiments

This section evaluates the model across the three tasks outlined above. The sentiment classification and paraphrase detection tasks are carried out as outlined in the default final project handout. For sonnet generation, the given task was used as a baseline, and several DPO preference construction methods are evaluated to see which performed the best.

5.1 Data

The sentiment classification task uses the Stanford Sentiment Treebank (SST) dataset (Socher et al., 2013) which contains movie review sentences labeled as positive or negative, and is formulated as a binary classification task. It also uses the CFIMDB dataset, a larger dataset of 2,434 movie reviews constructed in a similar manner. Unlike the SST dataset, the examples in the CFIMDB set are longer and contain multiple sentences, making it useful for testing sentiment classification beyond shorter reviews.

The paraphrase detection task uses the Quora Question Pairs dataset (Dolan and Beckett, 2005). Each example contains a question pair and a binary label indicating if they are paraphrases of each other or not. The dataset consists of 400,000 examples, with 141,506 allocated to the training set, 20,215 allocated to the dev set, and 40,431 allocated to the test set.

The Sonnet Dataset consists of 154 sonnets written by Shakespeare, each consisting of 14 lines of text following the rhyme scheme ABAB CDCD EFEF GG. 143 sonnets were allocated to the training set, and 12 are allocated to the test set.

5.2 Evaluation method

Sentiment classification and paraphrase detection are evaluated on their accuracy, while sonnet generation was evaluated on its chrF score through the provided dev leaderboard.

5.3 Experimental details

All experiments used the transformer-based language model provided in the starter code. For classification tasks, the pretrained transformer representations were passed through task-specific classification heads; for generation, the model produces text autoregressively. For sentiment classification, training was performed for 3 epochs using the Adam optimizer with a learning rate of 1×10^{-5} and batch size of 16. During training, the entire transformer backbone was fine-tuned rather than frozen, allowing the model to adapt its representations for the task. Paraphrase detection was trained in the same way.

For sonnet generation, the baseline model was fine-tuned using DPO. Preference pairs were constructed automatically using heuristic ranking functions applied to generated candidate continuations. The four synthetic preference construction methods are first evaluated, and for the most successful preference construction method, different β values were tested to see which produced the highest chrF score.

DPO training used the hyperparameters listed below:

- Epochs: 3
- Batch Size: 4
- Learning Rate: 1×10^{-6}

- $\beta = 0.1$

Model checkpoints were saved after each training epoch, and the final checkpoint was used for evaluation, which produced the strongest performance on the development leaderboard during preliminary experiments. For the entire project, an NVIDIA L4 GPU was used to train the models.

5.4 Results

Dataset	Accuracy
SST (Dev)	0.520
SST (Test)	0.552
CFIMDB (Dev)	0.980

Table 1: Sentiment classification results on SST.

Dataset	Accuracy
Dev	0.891
Test	0.874

Table 2: Paraphrase detection results on the Quora Question Pairs dataset.

Method	chrF Score (Dev)
Baseline	35.400
Random	42.012
LogProb	38.187
Repetition	38.027
Repetition + Structure	38.187

Table 3: Sonnet generation results for different preference construction methods under DPO.

β	chrF Score (Dev)
0.01	41.585
0.05	41.510
0.10	42.012
0.50	40.918

Table 4: Beta sweep for DPO using the random preference construction strategy.

On the test leaderboard for sonnet generation, random preference generation was used with $\beta = 0.10$. This achieved a chrF score of 43.242.

6 Analysis

The sentiment classification and paraphrase detection tasks indicate that the baseline transformer architecture is able to learn meaningful semantic representation. The sentiment model achieved strong performance on dev and test sets, demonstrating the pretrained model’s ability to capture useful contextual features for semantic classification.

Similarly, the paraphrase detection model achieves high accuracy on the Quora dataset. Qualitative inspection of the prediction examples suggests that the model performs better when paraphrases show

lexical on syntactic similarity. However, errors occur when paraphrases require deeper semantic reason or external knowledge. For instance, pairs that have similar meaning using different phrasing or speech style were occasionally misclassified due to surface-level similarity being low.

The most significant findings of this paper come from the sonnet generation experiments. Applying DPO consistently improved performance compared to the baseline language model trained with MLE. The baseline model achieved a dev leaderboard score of 35.40, while the DPO models achieved scores of at least 38.027, with the highest (coming from random preference construction) being 42.012. This improvement highlights a key limitation of MLE training; although it nudges the model to mimic the training distribution, it does not explicitly optimize for stylistic qualities that may be expected in creative writing. Preference-based objectives address gap through directly encouraging the model to assign higher probability to preferred outputs compared to weaker alternatives.

A surprising finding of the experiment is that random preference assignment outperformed more sophisticated heuristics. There are several possible explanations for this result. First, the heuristic ranking functions may have introduced unintended biases into the training data. For example, choosing the continuation with the highest language log-probability could bias the model towards overly safe generations, as high-probability texts usually include common phrases rather than text one would look for in stylistic poetry. Heuristics based on repetition and structure rely on imperfect proxies for quality; although repetition is often undesirable, the heuristics used may have penalized valid poetic patterns or may have failed to capture other essential stylistic properties. Random preference assignment, while noisy, avoids these biases. As the DPO objective compares relative probabilities across between pairs, randomly assigned comparisons may still encourage the model to sharpen its probability distribution. The training signal generated here could act as a form of regularization that encourages the model to distinguish between outputs without significantly constraining the generations that it can produce.

Lastly, a qualitative inspection of generated sonnets showed that DPO-trained models produced continuations that were generally more diverse and stylistically coherent, whereas the baseline outputs suffered from repeated phrases, abrupt topic shifts, and inconsistent poetic style. Overall, these results show that preference optimization can significantly improve text generation in creative contexts even if the preference signal is weak or artificially generated.

7 Conclusion

This project mainly investigated the effectiveness of DPO for improving neural sonnet generation as part of the CS224N default final project. The model was first evaluated on sentiment and paraphrase detection tasks, and was able to achieve strong baselines performance and confirming that the GPT transformer architecture was able to learn useful semantic representations. The primary contribution of this paper focused on improving sonnet generation through fine-tuning the sonnet generation model using DPO with automatically constructed preference pairs. DPO improved performance over the baseline MLE model, increasing the dev leaderboard score from 35.40 to 42.01. An interesting finding was that randomly constructed preference pairs produced the best results, suggesting that the contrasting learning signal provided by the DPO objective may be more important than the exact method used to generate preferences. In conclusion, the results obtained demonstrate that preference-based optimization can provide as simple yet effective method for improving generative language models.

References

- Christiano, Paul et al. (2017). “Deep reinforcement learning from human preferences”. In: *arXiv preprint arXiv:1706.03741*. DOI: 10.48550/arXiv.1706.03741. URL: <https://arxiv.org/abs/1706.03741>.
- Ghazvininejad, Marjan et al. (2016). “Generating Topical Poetry”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 1183–1191. DOI: 10.18653/v1/D16-1126. URL: <https://aclanthology.org/D16-1126/>.
- Lau, Jey Han et al. (2018). “Deep-speare: A Joint Neural Model of Poetic Language, Meter and Rhyme”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pp. 1948–1958. DOI: 10.18653/v1/P18-1181. URL: <https://aclanthology.org/P18-1181/>.
- Rafailov, Rafael et al. (2023). “Direct Preference Optimization: Your Language Model is Secretly a Reward Model”. In: *arXiv preprint arXiv:2305.18290*. DOI: 10.48550/arXiv.2305.18290. URL: <https://arxiv.org/abs/2305.18290>.
- Vaswani, Ashish et al. (2017). “Attention Is All You Need”. In: *arXiv preprint arXiv:1706.03762*. DOI: 10.48550/arXiv.1706.03762. URL: <https://arxiv.org/abs/1706.03762>.