

Title of your project

Stanford CS224N {Custom, Default} Project

Arun Chetri

Department of Electrical Engineering
Stanford University
rooneyc@stanford.edu

Marcus Kushner

Department of Symbolic Systems
Stanford University
marcusbk@stanford.edu

Ian Lasic

Department of Computer Science
Stanford University
ianlasic@stanford.edu

Abstract

In this project we aim to adapt the GPT-2 decoder transformer architecture via fine tuning and alteration to the self-attention mechanism; firstly through synthesised attention, and secondly by finetuning with direct preference optimization (DPO). With each implementation we will benchmark performance compared to the baseline (GPT-2) on three main tasks: Sentiment Analysis, where the model is finetuned for classification on the SST-2 and CFIMDB datasets; Cloze-Style Paraphrase Detection, testing semantic understanding by measuring accuracy and Pearson correlation; and Sonnet Generation, focused on open-ended creative text production evaluated through both quantitative metrics and qualitative error analysis. Our experiments demonstrate that (DPO) is effective for creative text generation, improving the CHRF score for sonnet generation from 41.663 to 42.832 with minimal computational overhead. In contrast, the Synthesized Attention experiments revealed a performance-efficiency trade-off; while the Random + Vanilla hybrid approached the baseline, the purely Random Synthesizer underperformed across all sentiment analysis metrics and failed to provide the hypothesized speedup, with Vanilla Attention maintaining superior accuracy (97.14% on CFIMDB) and faster epoch runtimes.

1 Key Information to include

- Mentor: Shicheng Liu
- External Collaborators (if you have any):
- Sharing project:
- Late Days (if you are using any):

Team member	Late days remaining
Ian Lasic	3
Marcus Kushner	2
Arun Chhetri	1
Total pooled	6
Team size	3
Late days applied for the report	1

2 Introduction

In this project, we explored two separate modifications to the GPT-2 architecture through synthesized attention and Direct Preference Optimization. The first extension of this project explores the popular fine-tuning method DPO to improve performance on a specific task. Fine-tuning for tasks in LLMs has become extremely useful and helps expand the capabilities of LLMs. Our specific goal for this extension is to deepen our understanding of DPO and show increased performance on the sonnet-generation task. Additionally, this extension gives us the opportunity to explore and learn the tradeoffs of synthetic data generation. To us, success for this extension means implementing DPO using high-quality synthetic data that outperforms the GPT-2 model.

The second extension, synthesized attention, challenges the assumption that the traditional dot-product attention mechanism is needed for Transformers. This problem is challenging because the traditional Transformer architecture has dominated recent advances in Large Language Models (LLMs). The research question for this extension is whether we can achieve similar or improved performance while increasing the efficiency of the attention mechanism. This extension is interesting because reducing computational costs and increasing LLM speed are essential for environmental sustainability and user satisfaction. For this extension, we will compare the performance of synthesized attention to the GPT-2 architecture on the sentiment classification and sonnet generation tasks. To us, success for this extension means implementing a synthesized attention mechanism that matches or approaches GPT-2’s performance while improving efficiency.

3 Related Work

3.1 Direct Preference Optimization (DPO)

Large-Language Models (LLMs) learn a wide range of skills by training on large datasets. However, these models often require further fine-tuning to achieve a state of the art performance on specific tasks. A popular approach for fine-tuning is Proximal Policy Optimization (PPO), which uses reinforcement learning and reward modeling to optimize the policy based on human preferences.[1] However, PPO has key limitations, namely, RL optimization is computationally expensive. DPO aims to fix these limitations by removing the complex reward model.[2] The main contribution of DPO is removing the separately trained reward model and RL from the fine-tuning process. DPO relies directly on preference data and a parameterized reward function to fine-tune towards human preferences. Specifically, one paper d the DPO model against a GPT-4 as a baseline policy on three main tasks: controlled sentiment generation, summarization, and single-turn dialogue. DPO outperforms PPO across all three tasks by achieving higher win rates against GPT-4 while having better stability and computational efficiency. Therefore, prior work has found meaningful improvements with DPO on generation tasks. We aim to strengthen these results in our project by demonstrating improvements in the sonnet generation task using DPO.

3.2 Synthesizer

The second main extension we chose to explore in this project is synthesized attention, which alters the traditional attention mechanism in the Transformer.[3] Transformers rely on dot-product self-attention to compute token-to-token alignment weights. This mechanism is widely believed to be central to the architecture’s success, as it enables modeling long-range dependencies. As such, it has been central in LLMs reaching the quality they are at today.

However, the computational cost of attention scales quadratically with sequence length. However, recent work has explored simplifying or approximating attention (e.g., sparse attention, low-rank approximations, convolutions). The Synthesizer paper challenges the assumption that token-token interactions are essential. The authors hypothesize that attention alignment matrices might not need to be computed through pairwise dot products at all, but could be done in much more computationally-efficient ways that still lead to a model with a similar level of performance.

The paper introduces Synthetic Attention, which generates alignment matrices without explicit query-key dot products. This paper had several key contributions, including proposing synthetic alignment functions that do not rely on token-token interactions and demonstrating that random alignment matrices perform competitively (on certain tasks). Specifically, this paper showed that a random

synthesizer is 60% faster than dynamic convolutions, and achieves +3.5% relative improvement in perplexity. Additionally, they demonstrated competitive results across various applications in machine translation and language modeling. We aim to explore this alternative architecture and compare its performance with GPT-2 on sentiment classification and sonnet generation tasks. Specifically, the performance of synthesized attention seems to be task-dependent. By applying synthesized attention to sentiment classification and sonnet generation, we aim to further contribute to understanding which tasks are well-suited for synthesized attention.

4 Approach

4.1 DPO Implementation

To implement Direct Preference Optimization (DPO) for sonnet generation, we first constructed a preference dataset by transforming the target sonnets into contrastive pairs. For each prompt x , consisting of the first three lines of a sonnet, the “preferred” completion y_w is the original full text, while the “dispreferred” completion y_l is synthetically generated by either chopping off the last few lines of the sonnet, or rearranging the ordering of the lines of the sonnet.

During training, we pass each triplet (x, y_w, y_l) through both the active GPT-2 policy π_θ and a frozen reference version π_{ref} . The implementation calculates the log-probabilities for the completions under both models and computes the log-ratio $\log(\pi_\theta(y)/\pi_{\text{ref}}(y))$. By applying the sigmoid of the difference between these ratios (scaled by β), the trainer optimizes the policy to maximize the implicit reward of the full sonnets over the truncated versions.

The DPO loss function used to update the weights of the GPT-2 model is defined as:

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

4.2 Synthesized Attention Implementation

To evaluate synthesized attention within a decoder-based language model, we modified the self-attention module of GPT-2 to incorporate synthetic alignment matrices. Rather than replacing the Transformer architecture entirely, our implementation preserves the original GPT-2 structure while altering only the computation of attention weights inside each multi-head attention block. This allows for a direct comparison between standard dot-product attention and synthesized alternatives under identical training conditions.

For the Random Synthesizer variant, each attention head maintains a learnable tensor $R \in \mathbb{R}^{H \times T \times T}$, where H denotes the number of attention heads and T represents the maximum sequence length supported by the model. This matrix is initialized randomly and learned during training. During the forward pass, the tensor is sliced to match the current sequence length and combined with the causal masking used in autoregressive language modeling. The resulting alignment matrix is normalized using a softmax operation to produce attention weights:

$$A = \text{softmax}(R + M) \quad (2)$$

where M represents the causal and padding masks applied to prevent attending to future tokens or padded inputs. The output of the attention layer is then computed by applying these weights to the value projections:

$$\text{Attention}(V) = AV \quad (3)$$

This formulation removes the dependency on query-key similarity and instead allows the model to learn global attention patterns directly through the parameters of R .

In addition to the purely synthetic variant, we also implemented a hybrid Random + Vanilla attention mechanism. In this configuration, the synthetic attention logits are combined with the standard scaled dot-product attention scores before the softmax normalization. To allow the model to control the

relative contribution of each component, we introduce learnable scalar parameters α_q and α_r that weight the dot-product attention and the synthetic attention respectively. The combined attention distribution is therefore defined as:

$$A = \text{softmax} \left(\alpha_q \frac{QK^\top}{\sqrt{d}} + \alpha_r R + M \right) \quad (4)$$

The resulting attention weights are then applied to the value projections:

$$\text{Attention}(Q, K, V) = AV \quad (5)$$

where Q , K , and V represent the query, key, and value projections, d is the dimensionality of each attention head, R is the learnable synthetic attention matrix, and M denotes the causal and padding masks. The learnable α parameters allow the model to adaptively balance the influence of the traditional token-dependent attention scores and the synthetic alignment patterns during training.

All other components of the GPT-2 architecture, including the feed-forward layers, residual connections, and layer normalization, remain unchanged. The parameters of R are trained jointly with the rest of the model using standard backpropagation, enabling the synthetic attention mechanism to adapt to each task during fine-tuning.

5 Experiments

5.1 Data

The default project utilizes four datasets to evaluate GPT-2 across various NLP dimensions. For sentiment classification, it uses the SST dataset, which contains 11,855 single sentences with five fine-grained labels, and the CFIMDB dataset, which consists of 2,434 longer, highly polar reviews with binary labels. To test semantic understanding, the Quora dataset provides 400,000 question pairs labeled for paraphrase detection. Finally, the Sonnet dataset includes 154 Shakespearean sonnets to assess the model’s ability to generate text following a strict 14-line structure and rhyme scheme.

5.1.1 DPO data

For the DPO section of our project, we utilized two sets of data. First, we generated synthetic data using the sonnets training dataset given. Secondly, because there was such a limited amount of data in the given sonnet dataset, we also found a second Shakespeare sonnet dataset on HuggingFace with pairs of sonnets, which we went through and labeled as preferred/dispreferred.[4] The final dataset we used for DPO contained both the synthetic data generated as well as the HuggingFace sonnet dataset. This combined dataset has 378 lines of preferred/dispreferred pairs.

5.2 Evaluation method

For the cloze-style detection task, we will be evaluating our proficiency on the model’s accuracy. Essentially, how well the model classifies text as a paragraph. For the sonnet generation task, we will evaluate our performance with a CHRF score. The CHRF score measures how closely your generated sonnets follow William Shakespeare’s language distribution. Additionally, for the synthesized attention extension, we will be using the training time as an evaluation metric. This is because we hypothesize that synthesized attention should improve efficiency.

5.3 Experimental details

For sentiment analysis, we ran the training on a T4 GPU setup. We trained on the small GPT2 model for 10 epochs with a learning rate of 1e-5. On a T4 GPU, each epoch took approximately 30 minutes which resulted in about 5 hours of total training time. This training time is significantly longer than the expected time listed in the Default Project handout. We suspect this could be due to not optimizing the batch size due to memory issues with our GPU instance.

The paraphrase detection task has a much larger training set, so we decided to use an A100 GPU to accelerate training. We trained on the small GPT-2 model for 10 epochs with a learning rate of $1e-5$. On an A100 GPU, each epoch took approximately 45 mins which resulted in about 7.5 hours of total training time.

Finally, for sonnet generation, we ran our experiments on a T4 GPU. During training we used AdamW with a learning rate of $1.5e-5$ and weight decay of 0.01, batch size 8, and up to 10 epochs with early stopping. The model is trained on the sonnet dataset with a causal LM objective (next-token prediction) and a held-out dev set for validation. For submission generation, decoding uses nucleus sampling (top $p = 0.9$), temperature 1.0, and repetition penalty 1.0. With this configuration, each epoch took about 6.2 seconds on average. The sum of the logged epoch times is approximately 62 seconds. Sonnet CHRF Score (dev): 41.663 (+41.663).

For our DPO fine-tuning for sonnet generation, we ran experiments on a T4 GPU. Starting from the supervised sonnet model checkpoint (gpt2 small), we trained using the DPO loss function for 5 epochs with a learning rate of $1e-5$, $\beta = 0.1$, and batch size 4 on our combined DPO training set (described in section 5.1.1). On a T4 GPU, the full DPO run took approximately 400 seconds (about 6.7 minutes), corresponding to roughly 80 seconds per epoch, which is in line with earlier sonnet generation fine-tuning when accounting for extra data added in through synthetic generation.

Across training, the DPO training loss (this loss is the negative log-likelihood that the policy’s pairwise preference—relative to the reference—matches the human preference) decreased from 0.88 to 0.61 before leveling out at around 0.63, suggesting that the model was learning the DPO preferences without overfitting strongly to the relatively small DPO dataset. For generation after DPO, we reuse the same decoding configuration as in supervised sonnet generation, with nucleus sampling (top- $p = 0.9$), temperature 1.0, and repetition penalty 1.0. DPO Sonnet CHRF Score (dev): 42.832 (+42.832).

For our Synthesizer Attention experiments, we fine-tuned models starting from the pretrained gpt2 small checkpoint and evaluated three attention variants: standard dot-product attention (*Vanilla*), Random Synthesizer attention (R), and a hybrid Random + Vanilla attention ($R+V$). Training was performed on an A100 Modal GPU using the same optimization settings as the baseline models to ensure a controlled comparison. For sentiment analysis on SST and CFIMDB, we trained for 10 epochs with a learning rate of 1×10^{-5} , using a batch size of 64 for SST and 8 for CFIMDB. In the Random Synthesizer variants, the learnable attention matrix R was initialized from a normal distribution with standard deviation 0.02 and trained jointly with the model parameters. To stabilize training from this random initialization, we applied a slightly larger learning rate (1×10^{-4}) to R while keeping the rest of the model at the baseline learning rate. On an A100 GPU, each SST epoch took roughly between 63-103 seconds (approximately 11 minutes total training time).

During training, both Random Synthesizer and $R+V$ variants showed stable convergence comparable to the vanilla attention model. The hybrid $R+V$ configuration benefited from combining the learned synthetic attention logits with the standard dot-product attention scores before softmax normalization, allowing the model to retain the inductive bias of query-key similarity while incorporating additional learned attention structure from R . For sonnet generation experiments, we used the same decoding configuration as the supervised baseline: nucleus sampling (top- $p = 0.9$), temperature 1.0, and repetition penalty 1.0. This ensured that differences in generation quality could be attributed to the attention mechanism rather than changes in decoding strategy.

5.4 Results

5.4.1 Base GPT-2 model VS Synthesized attention sentiment analysis results

Table 1: Model comparison on SST and CFIMDb. Accuracy values report the best recorded scores for each dataset. Runtime is the average epoch time computed separately for each dataset.

Model	SST Dev	CFIMDb Dev	Lowest Loss	SST Time (s)	CFIMDb Time (s)
Random + Vanilla	0.4995	0.9551	0.0149	88.92	50.68
Vanilla	0.5177	0.9714	0.0109	63.74	43.42
Random	0.4469	0.8857	0.0641	102.87	43.95

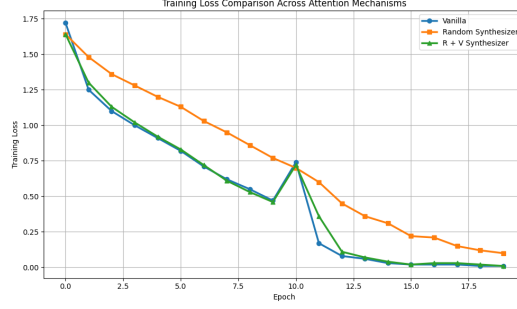


Figure 1: Sentiment Analysis Overlaid Attention Mechanism Loss

5.4.2 Base paraphrase detection results

Our paraphrase dev output had achieved a score of **0.899** on the autograder. These results show that the trained model is performing well on paraphrase detection. Additionally, our trained paraphrase model performed slightly worse on the test set achieving an accuracy of **0.844**. This drop in performance on the test set suggest our trained model may have slightly overfit to the training data and lost some ability to generalize. Future work could focus on tuning hyperparameters to ensure the model doesn't overfit. However, we focused most of our work on the sonnet generation task so we did not get to explore optimizing the test results.

5.4.3 Base GPT-2 VS DPO VS Synthesized Attention Sonnet Generation results

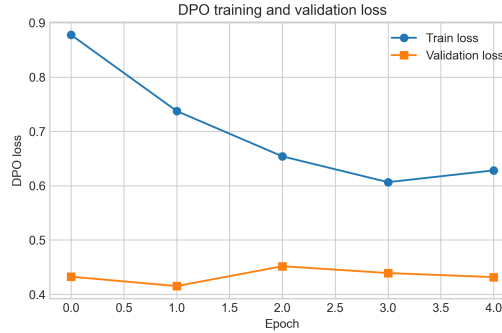


Figure 2: DPO training and validation dataset loss values per epoch

Model	CHRF Score	Avg. Epoch Runtime (s)
Vanilla Attention	41.663	23.72
Random Synthesizer	40.769	20.29
Random + Vanilla Synthesizer	41.294	20.73
DPO	42.832	80.02

Table 2: Comparison of sonnet generation performance and training runtime across attention mechanisms and DPO fine-tuning. CHRF measures similarity between generated and reference sonnets, while runtime reports the average training time per epoch.

6 Analysis

Overarching: The base GPT-2 mini model is a solid model to start from, with the ability to function well on all three tasks. Both DPO and synthesized attention provide clear opportunities to add small

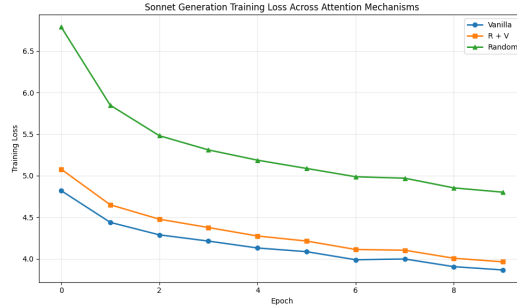


Figure 3: Sonnet Analysis Overlayed Attention Mechanism Loss

but noticeable improvements to the model, with the potential for both strategies to have a larger impact with (i) more data for sonnet generation (the lack of data made it hard to have significant differentiation between base model, DPO, and synthesized attention results) and (ii) a broader combination of different synthesized attention strategies.

Specifically, we saw that using DPO on top of the baseline model was the most effective approach for Sonnet Generation, with minimal added runtime. This suggests that for tasks where human preferences can help inform models, DPO is a valuable layer to add.

Referencing table 1, we see that using the purely random synthesis approach to synthesizer attention, the training time was not comparatively faster as we had hypothesized (for the SST dataset, random synthesis actually too longer per epoch). And simultaneously, the random synthesis approach showed significant drawbacks in accuracy on both the SST and CFIMDb datasets. The combination of random and vanilla attention performed functionally similarly to vanilla attention, though with slightly lower accuracy and slightly longer runtimes. This suggests that for the metrics we cared about in sentiment analysis, vanilla attention is clearly the better approach compared with the synthetic approaches we tried.

7 Conclusion

In this project, we explored two extensions to the base GPT-2 architecture: Direct Preference Optimization (DPO) and Synthesized Attention. These extensions highlight complementary strategies for improving LLM performance.

For sonnet generation, DPO proved to be the more impactful extension. By fine-tuning on a small but targeted preference dataset, we achieved a CHRF score improvement from 41.663 to 42.832 with minimal added runtime (6.7 minutes of total training). This confirms that preference-based fine-tuning is a practical and efficient approach when human preference signals can be clearly defined, even with limited data.

For Synthesized Attention, our results showed that the purely random synthesis approach underperformed vanilla attention on both SST and CFIMDb, while also increasing training time — contrary to our efficiency hypothesis. The hybrid Random + Vanilla approach partially recovered accuracy but still fell short of vanilla attention. This suggests that for sentiment classification tasks, the query-key dot-product attention is difficult to replace with random / learned alignment matrices.

Overall, our experiments reinforce the value of targeted fine-tuning techniques like DPO for generative tasks, while also demonstrating the limits of architecture substitutions like random synthesized attention on tasks that benefit from token-token interactions. Future work could explore DPO with larger preference datasets, or synthesized attention variants better suited to generation tasks.

Team contributions (Required for multi-person team)

Marcus Kushner - Generated synthetic data for DPO on the Sonnet dataset, implemented DPO, and ran various DPO training runs to find optimal parameters for DPO. Wrote paper.

Ian Lasic - Implemented the base GPT-2, Adam Optimizer, base sentiment analysis, paraphrase detection, and sonnet generation. Experimented with several different model generation strategies, including beam search and repetition penalties to optimize sonnet generation. Set up a pipeline to efficiently run experiments and create plots for all three tasks utilizing GPUs. Wrote paper.

Arun Chhetri - Built and implemented architecture for supporting all three attention mechanisms (vanilla, random synthesis, random + vanilla synthesis). Generated results and plots for sentiment analysis and sonnet generation. Wrote paper.

References

- [1] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [2] 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. 2023.
- [3] Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. 139:10183–10192, 2021.
- [4] Lambent. Shakespeare_sonnets_diffused. Hugging Face Datasets, 2024.