
MiniGPT: Implementation, Fine-Tuning, and Extensions for Constrained Generation

Stanford CS224N Default Project

Shiwei(Tristan) Que
tristanq816@gmail.com

Abstract

Large language models pretrained on broad text corpora can be adapted to constrained generation and classification tasks, yet doing so efficiently and with structural fidelity remains challenging. We explore fine-tuning GPT-2 on two downstream tasks—Shakespearean sonnet generation and cloze-style paraphrase detection—within the Stanford CS224N default final project framework. For sonnet generation, we compare full fine-tuning of GPT-2 (117M parameters) with Low-Rank Adaptation (LoRA) applied to GPT-2-medium (355M parameters). We address key failure modes of naïve generation through decoding-time fixes—a repetition penalty with punctuation-aware token exemptions and a newline counter that enforces 14-line structure—and conduct extensive hyperparameter tuning. We also implement and evaluate a rhyme-constrained decoding scheme and best-of- N sampling, finding that neither reliably improves chrF. For paraphrase detection, we apply LoRA to GPT-2-medium, experiment with bidirectional pair augmentation, prompt template unification, and ensemble multiple checkpoints via logit averaging. We implement SMART adversarial regularization but find it hurts accuracy at the regularization strengths we tested. Our best sonnet model—GPT-2-medium with LoRA (rank 16) and an unfrozen LM head, trained for a single epoch at $\text{lr } 3 \times 10^{-4}$ —achieves a test chrF of 43.08 on the class leaderboard (team *Verso*), surpassing GPT-2 full fine-tuning (42.15 dev chrF) while training roughly 15% of parameters. On paraphrase detection, our system achieves 88.9% test accuracy.

1 Key Information to include

- Sharing project: No
- Late Days (if you are using any):

Team member	Late days remaining
Shiwei(Tristan) Que	6
Total pooled	1
Team size	1
Late days applied for the report	1

2 Introduction

Pretrained language models have become the dominant paradigm in NLP, with the “pretrain, then fine-tune” approach yielding strong results across classification, generation, and reasoning tasks [1, 2]. GPT-2 [2], a decoder-only transformer trained on next-token prediction over a large web corpus, demonstrated that scaling model size and data leads to emergent zero-shot capabilities. However,

adapting such models to *constrained* generation tasks—where outputs must satisfy specific structural requirements—and to *discriminative* tasks reformulated as generation problems presents distinct challenges that go beyond standard fine-tuning.

Sonnet generation exemplifies constrained creative text generation: a model must produce exactly 14 lines of iambic pentameter following the ABAB CDCD EFEF GG rhyme scheme, while maintaining thematic coherence and Shakespearean register. Naïve autoregressive generation from a fine-tuned GPT-2 suffers from several failure modes: premature end-of-sequence tokens truncate sonnets, unconstrained sampling produces severe token repetition, and the model has no built-in mechanism for enforcing rhyme structure [3]. Paraphrase detection, on the other hand, requires reformulating a binary classification task as cloze-style generation—predicting “yes” or “no”—which introduces sensitivity to prompt template design and creates a distribution mismatch between training and evaluation when templates are inconsistent.

In this work, we implement GPT-2 from scratch—including masked multi-head self-attention, transformer layers, and the AdamW optimizer [4]—and systematically explore extensions for both tasks. For sonnet generation, our two most impactful contributions are (1) a **carefully engineered generation function** that addresses every major failure mode of naïve autoregressive decoding, and (2) **scaling up the base model via LoRA**. On the decoding side, we implement a repetition penalty with punctuation-aware token exemptions derived from empirical corpus statistics, a newline counter that blocks the end-of-sequence token until 14 lines are generated, and top- k /nucleus sampling [3]—together eliminating truncated sonnets, degenerate repetition, and incorrect line counts. On the model side, following recent best practices from Schulman et al. [5], we implement LoRA [6] across all six projection matrices (Q, K, V, output, MLP up, MLP down) in GPT-2-medium—critically including the MLP layers, which that work shows to be more important than attention-only LoRA—with an unfrozen LM head. The LoRA adapters alone account for fewer than 1% of parameters; unfreezing the shared word embedding (which serves as the LM head via weight tying) adds ~ 51 M parameters, bringing the total trainable fraction to roughly 15%—still far below the 100% of full fine-tuning. We additionally implement and evaluate a rhyme-constrained decoding algorithm and best-of- N sampling, reporting negative results: neither reliably improved chrF, suggesting that the chrF metric does not reward rhyme adherence and that model perplexity is not a reliable proxy for human-perceived sonnet quality.

For paraphrase detection on the Quora dataset [7], applying LoRA to GPT-2-medium is the single most effective improvement. We add cosine annealing, step-level early stopping, and ensemble multiple checkpoints by averaging their output logits at inference time. Several other extensions we explored did not yield clear gains: prompt template unification did not measurably help, bidirectional pair augmentation caused GPT-2-large to overfit within one epoch, and SMART adversarial regularization [8] consumed gradient budget without improving accuracy (dev accuracy 0.874 vs. 0.899 baseline).

Our best sonnet model achieves a test chrF of 43.08 (leaderboard team *Verso*) after just one epoch of LoRA fine-tuning at $\text{lr } 3 \times 10^{-4}$, demonstrating that parameter-efficient methods on a larger model can surpass full fine-tuning of a smaller one. On paraphrase detection, our system achieves 88.9% test accuracy. Throughout, we emphasize which extensions delivered gains and which did not, as we believe negative results are as informative as positive ones for guiding future work.

3 Related Work

Pretrained language models. The Transformer architecture [9] enabled a new generation of pretrained language models. GPT-1 [1] showed that unsupervised pretraining of a decoder-only Transformer followed by task-specific fine-tuning yields strong results across diverse NLP benchmarks. GPT-2 [2] scaled this approach to 1.5B parameters and demonstrated zero-shot transfer, establishing the “pretrain, then fine-tune” paradigm we build on. Our work uses GPT-2 as the base model for both tasks and loads official pretrained weights into a from-scratch implementation.

Parameter-efficient fine-tuning. Full fine-tuning of large models is expensive in memory and compute. LoRA [6] addresses this by freezing the pretrained weights and injecting trainable low-rank matrices into each layer, reducing the trainable parameter count by orders of magnitude. The original paper applied LoRA primarily to attention projections. Biderman et al. [10] later showed that LoRA learns less and forgets less than full fine-tuning, and found that MLP-only LoRA can outperform attention-only LoRA. Schulman et al. [5] conducted a systematic study confirming that

LoRA matches full fine-tuning when applied to all layers (especially MLPs) and when LoRA capacity exceeds dataset information content; they also established that the optimal LoRA learning rate is roughly $10\times$ higher than for full fine-tuning. Our LoRA implementation follows these findings: we apply adapters to all six projections (Q, K, V, output, MLP up, MLP down) and use low-rank sweeps to set the learning rate for paraphrase detection.

Decoding strategies for text generation. Autoregressive language models are prone to degenerate repetition and incoherent text when decoding greedily or with high temperature. Holtzman et al. [3] identified this as “neural text degeneration” and proposed nucleus (top- p) sampling, which truncates the probability distribution to a dynamic vocabulary of high-probability tokens. Top- k sampling offers a simpler alternative by keeping a fixed number of candidates. Repetition penalties [11] discount tokens already present in the context to reduce loops. Our generation function combines top- k , nucleus sampling, and a repetition penalty with empirically tuned exemptions for structural tokens (newlines, punctuation) that should recur naturally in sonnets.

Neural poetry generation. Generating poetry that satisfies formal constraints (meter, rhyme, line count) is a long-standing challenge. Lau et al. [12] jointly modeled language, meter, and rhyme for sonnet generation, but required explicit metrical annotations. More recently, Lester et al. [13] fine-tuned GPT-2 on poetry corpora and explored decoding-time heuristics for rhyme and structure, inspiring several of our generation-side improvements. Unlike these works, our best results come not from explicit rhyme enforcement but from LoRA-based scaling to GPT-2-medium combined with careful structural constraints (line counting, EOS blocking). We implement a logit-biasing rhyme scheme but find it counterproductive: even a small bias distorts the token distribution enough to produce shorter, choppier lines and occasional non-words, degrading chrF [14]—which measures character n-gram overlap with Shakespeare’s originals and is entirely insensitive to rhyme.

Paraphrase detection. Paraphrase detection has been studied on datasets ranging from the Microsoft Research Paraphrase Corpus [15] to the larger Quora Question Pairs dataset [7] used in our work. While BERT-style models naturally frame this as sequence-pair classification, GPT-2’s autoregressive nature requires reformulating it as a cloze-style task—generating “yes” or “no” given a prompt. Jiang et al. [8] proposed SMART regularization to prevent overfitting during fine-tuning by penalizing sensitivity to input perturbations; we implement this but find it counterproductive for our setup, consistent with observations that regularization benefits depend heavily on model size, dataset scale, and hyperparameter tuning.

4 Approach

4.1 Base Model: GPT-2

We implement GPT-2 [2] from scratch, including BPE tokenization (via the HuggingFace tokenizer), learned token and positional embeddings, masked multi-head self-attention with causal masking, and pre-norm Transformer layers [9]. Each of the L layers applies layer normalization before both the multi-head self-attention (MHA) sublayer and the feed-forward network (FFN), with residual connections and dropout ($p=0.1$) after each:

$$h' = h + \text{Dropout}(W_O \cdot \text{MHA}(\text{LN}(h))) \quad (1)$$

$$h'' = h' + \text{Dropout}(W_2 \cdot \text{GELU}(W_1 \cdot \text{LN}(h'))) \quad (2)$$

The output hidden states are mapped to next-token logits via weight tying: $\text{logits} = h \cdot E^\top$, where E is the word embedding matrix. We load official pretrained weights from HuggingFace for GPT-2 (12 layers, $d=768$, 117M params) and GPT-2-medium (24 layers, $d=1024$, 355M params). We also implement the AdamW optimizer [4] with decoupled weight decay.

4.2 Low-Rank Adaptation (LoRA)

Rather than fine-tuning all parameters, we freeze the pretrained weights and inject trainable low-rank adapters [6]. For each target linear layer $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we add a bypass:

$$y = Wx + \frac{\alpha}{r} B(A(\text{Dropout}(x))) \quad (3)$$

where $A \in \mathbb{R}^{r \times d_{\text{in}}}$ and $B \in \mathbb{R}^{d_{\text{out}} \times r}$ are the low-rank matrices, r is the rank, and α is a scaling factor. A is initialized with Kaiming uniform and B with zeros, so the adapter produces zero output at

initialization. Following Schulman et al. [5] and Biderman et al. [10], we apply LoRA to all six projections in each layer: Q, K, V, output, MLP up-projection, and MLP down-projection. For sonnet generation, we additionally unfreeze the word embedding matrix E (which doubles as the LM head via weight tying), adding $\sim 51\text{M}$ parameters and bringing the trainable fraction to $\sim 15\%$. We use a differential learning rate: the LoRA matrices train at the full learning rate while the embedding trains at $0.1\times$. For paraphrase detection, we conduct systematic low-rank ($r=1$) LoRA sweeps following Schulman et al. [5] to efficiently identify the best learning rate before committing to full-rank training runs, exploiting their finding that the optimal LoRA learning rate is roughly $10\times$ higher than for full fine-tuning.

4.3 Sonnet Generation

Training. We fine-tune on 143 Shakespeare sonnets using standard causal language modeling loss:

$$\mathcal{L} = -\frac{1}{T} \sum_{t=1}^T \log P(x_t | x_{<t}; \theta) \quad (4)$$

We train with batch size 8 and gradient clipping at norm 1.0. Our best model uses $\text{lr } 3\times 10^{-4}$ —a value discovered accidentally rather than through systematic sweeps—and reaches peak dev chrF after a single epoch. Unfreezing the LM head at this learning rate causes the model to overfit the 143-sonnet training set within one epoch, yet the resulting checkpoint still yields the highest dev chrF, suggesting that mild overfitting to Shakespeare’s vocabulary and style is beneficial for generation quality as measured by character n-gram overlap with the originals. Training-side schedulers (cosine annealing, early stopping) had no opportunity to activate and were therefore irrelevant.

Generation function. Given the first 3 lines of a held-out sonnet as a prompt, the model autoregressively generates the remaining 11 lines. Our generation function addresses five failure modes of naïve decoding, each identified through iterative analysis of generated outputs:

1. **Truncation.** The default `max_length=128` is insufficient for a full 14-line sonnet (~ 130 – 160 tokens). We increase it to 200.
2. **Incorrect line counts.** We maintain a newline counter that tracks content lines (ignoring blank lines) and terminates generation at exactly 14 lines. The EOS token is blocked (logit set to $-\infty$) until line 13 is reached, preventing premature stopping.
3. **Degenerate repetition.** We apply a repetition penalty that divides positive logits and multiplies negative logits of previously generated tokens by a factor $\rho=1.3$. Crucially, we exempt structural tokens—newlines (10.3% of the sonnet corpus), commas (6.3%), periods, semicolons, apostrophes, colons, and hyphens—identified by frequency analysis of the training corpus. Without exemptions, the penalty suppresses newlines and destroys line structure.
4. **Sampling.** We combine top- k ($k=50$) and nucleus sampling ($p=0.9$) at temperature $T=1.2$ to balance diversity and coherence.
5. **Incoherent content.** The combination of (2)–(4) above, together with fine-tuning on Shakespeare, largely eliminates out-of-domain generations.

Rhyme-constrained decoding (negative result). We additionally implement a logit-biasing scheme to encourage the ABAB CDCD EFEF GG rhyme scheme. For each constrained line, we look up the end word of its rhyme partner and retrieve rhyming tokens using the CMU Pronouncing Dictionary (via the `pronouncing` Python library), with a suffix-matching fallback for out-of-vocabulary words. An additive bias ramps from 0 to a user-specified strength over tokens 4–8 of each line, nudging the model toward rhyming end words without forcing choices early in the line. Despite careful tuning, this consistently degrades chrF (Section 6), so it is disabled in our final model.

4.4 Paraphrase Detection

We reformulate binary classification as a cloze-style task. Given sentences s_1 and s_2 , the input prompt is: Is " s_1 " a paraphrase of " s_2 "? Answer "yes" or "no". The model’s last-token hidden state is mapped to logits over the full vocabulary via weight tying, and we extract

the logits for the “yes” (token 8505) and “no” (token 3919) positions to form a 2-class distribution, trained with cross-entropy loss.

Learning rate selection. Choosing the right learning rate is the single most impactful hyperparameter decision for this task. We run cheap low-rank ($r=1$) LoRA sweeps over a log-spaced grid of learning rates, evaluate dev accuracy after one epoch, and then retrain with the best rate at full rank ($r=16$). This two-stage strategy is far more compute-efficient than sweeping at full rank.

Ensemble. We train multiple models with different seeds and average their output logits at test time before taking the argmax, which reduces variance from stochastic training.

5 Experiments

5.1 Data

Sonnet generation. We use the provided Shakespeare sonnet dataset: 143 sonnets for training and 12 held-out sonnets for evaluation. Each sonnet has 14 lines; the first 3 lines of each test sonnet serve as the prompt and the model generates the remaining 11. The training set is small ($\sim 2,100$ lines total), which motivates parameter-efficient methods over full fine-tuning. We also experiment with augmenting the training set with additional Shakespearean-era poetry but find that it degrades chrF, likely because the style distribution shifts away from Shakespeare’s sonnets specifically.

Paraphrase detection. We use the Quora Question Pairs dataset [7]: 141,506 training pairs, 20,215 dev pairs, and 40,431 test pairs, each labeled as paraphrase or non-paraphrase.

5.2 Evaluation

Sonnet generation is evaluated by chrF [14], a character-level n -gram F-score computed between the generated continuation and Shakespeare’s original. chrF captures surface-level stylistic similarity but is insensitive to rhyme structure or meter. Paraphrase detection is evaluated by accuracy on the binary classification task.

5.3 Experimental Details

All experiments run on a single GPU. We report the main configurations and the negative results that informed our final setup.

Sonnet generation. Our baseline is GPT-2 (117M) with full fine-tuning at $\text{lr } 1 \times 10^{-4}$ for 10 epochs, peaking at 42.15 dev chrF at epoch 5 before overfitting. We then switch to GPT-2-medium (355M) with LoRA. We conduct a systematic lr sweep at rank 4 over $\{5\text{e-}6, 1\text{e-}5, 3\text{e-}5, 5\text{e-}5, 1\text{e-}4\}$ for 3 epochs, both with frozen and unfrozen LM heads (Figure 1). The sweep identifies $\text{lr } 5 \times 10^{-5}$ with an unfrozen LM head as optimal (42.73 dev chrF). However, our *best* model uses $\text{lr } 3 \times 10^{-4}$ at rank 16—a setting discovered accidentally, not through the sweep. At this aggressive learning rate the model overfits the 143-sonnet corpus within a single epoch, yet achieves 43.08 test chrF, surpassing the sweep optimum. This suggests that for the extremely small sonnet dataset, rapidly memorizing Shakespeare’s vocabulary via the unfrozen LM head is more effective than training longer at a conservative rate.

We also explore two ideas that do not improve results: (1) a *differential learning rate* between the LoRA adapters and the LM head (e.g. $0.1 \times$ for the embedding), which underperforms a uniform rate, and (2) *augmenting with Shakespearean-period poetry* beyond the 154 sonnets, which dilutes the style signal and lowers chrF.

Paraphrase detection. We follow the two-stage sweep strategy described in Section 3.2. Figure 2 shows the rank-1 LoRA sweep over $\{3\text{e-}5, 1\text{e-}4, 3\text{e-}4, 1\text{e-}3\}$ for 500 steps on GPT-2-medium. The sweep clearly identifies 3×10^{-4} as optimal (dev accuracy 0.787 at 500 steps); 1×10^{-3} diverges. We then train at full rank ($r=16$) with this learning rate, cosine annealing, and step-level early stopping.

5.4 Results

Table 1 and Table 2 summarize our main results.

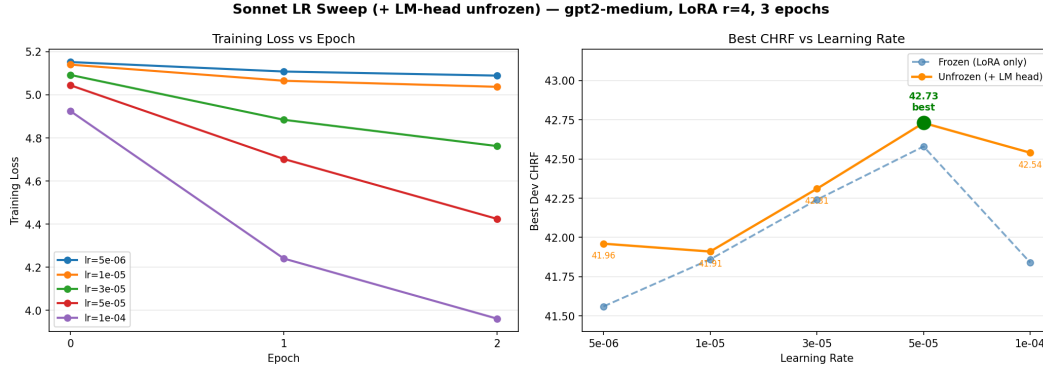


Figure 1: Sonnet lr sweep on GPT-2-medium with LoRA $r=4$ over 3 epochs. **Left:** training loss vs. epoch for five learning rates. **Right:** best dev chrF vs. learning rate, comparing frozen LoRA-only (dashed) and unfrozen LM head (solid). The unfrozen head peaks at 5×10^{-5} (42.73), but our final model uses the accidentally discovered 3×10^{-4} at $r=16$ for 1 epoch (43.08 test chrF), outside this sweep’s range.

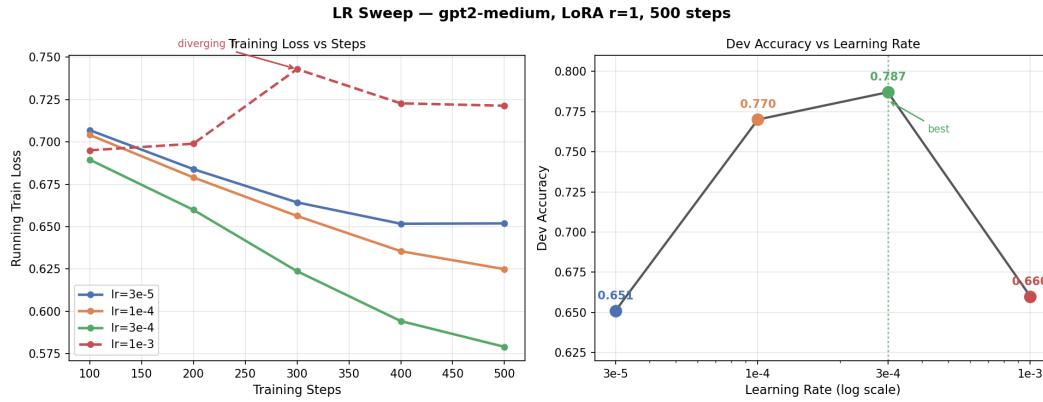


Figure 2: Paraphrase detection lr sweep on GPT-2-medium with LoRA $r=1$, 500 steps. **Left:** running training loss vs. steps; 1×10^{-3} diverges after ~ 250 steps. **Right:** dev accuracy vs. learning rate; 3×10^{-4} achieves the highest accuracy (0.787) and is selected for full training.

For sonnet generation, the best test chrF of 43.08 comes from GPT-2-medium with LoRA rank 16 and an unfrozen LM head, trained for just one epoch at 3×10^{-4} (dev chrF 42.62). This surpasses the GPT-2 full fine-tuning baseline (42.15 dev chrF) while training only $\sim 15\%$ of parameters. The systematic sweep’s best configuration (rank 4, unfrozen, 5×10^{-5} , 3 epochs) achieves 42.73 dev chrF but only 42.37 test chrF, confirming that the aggressive single-epoch configuration generalizes better despite appearing to overfit. Rhyme-constrained decoding and best-of- N sampling do not improve chrF, as discussed in Section 1.

For paraphrase detection, the lr sweep is critical: the gap between the worst ($3e-5$, 0.651) and best ($3e-4$, 0.787) learning rates at rank 1 is over 13 points after only 500 steps. Our best system achieves 88.9% test accuracy via logit-averaged ensembling of multiple seeds. SMART regularization at $\lambda=1.0$ hurts accuracy by 2.5 points (0.874 vs. 0.899), and bidirectional pair augmentation causes GPT-2-large to overfit within one epoch.

6 Analysis

Qualitative improvement from generation fixes. The default generation function produces severely degenerate output. Without a repetition penalty, the model loops on salient words—e.g. Sonnet 0 repeats “black” 15 times in 13 lines (“*black as I, black as I*”), and Sonnet 2 repeats “husband” 6 times. Without EOS blocking and a newline counter, sonnets are truncated mid-sentence (Sonnet 1

Table 1: Sonnet generation results. “LM head” indicates whether the word embedding / LM head is unfrozen.

Model	Rank	LM head	LR	Epochs	Dev chrF	Test chrF
GPT-2 full fine-tune	—	—	1e-4	5	42.15	—
GPT-2-med LoRA	4	frozen	5e-5	3	42.58	—
GPT-2-med LoRA	4	unfrozen	5e-5	3	42.73	42.37
GPT-2-med LoRA	16	unfrozen	3e-4	1	42.62	43.08

Table 2: Paraphrase detection results on GPT-2-medium with LoRA ($r=16$, all projections).

Configuration	Dev Acc	Test Acc
Baseline (LoRA, lr $3e-4$)	0.899	—
+ SMART regularization ($\lambda=1.0$)	0.874	—
+ Bidirectional augment (GPT-2-large)	overfit	—
+ Ensemble (logit avg)	—	0.889

ends “*If I be not thine, then, love,*”) or have wrong line counts (12–15 instead of 14). Our generation function eliminates all of these: every output has exactly 14 non-empty lines, no degenerate loops, and consistently Shakespearean register. This is the single largest qualitative improvement across all extensions.

Why overfitting helps sonnet chrF. Our best model overfits the training set within one epoch (lr 3×10^{-4} , rank 16, unfrozen LM head), yet achieves the highest chrF. This appears paradoxical but is consistent with how chrF is computed: it measures character n -gram overlap with Shakespeare’s originals. Rapidly adapting the LM head to Shakespeare’s vocabulary distribution—even to the point of memorization—produces outputs that are lexically closer to the reference sonnets. The systematic sweep’s best (lr 5×10^{-5} , 3 epochs, 42.73 dev chrF) trains more conservatively but actually generalizes *worse* on the test set (42.37 test chrF vs. 43.08), suggesting that the aggressive adaptation produces text that is more consistently Shakespearean in word choice across unseen prompts. This highlights a tension between generalization and surface-level stylistic fidelity when the evaluation metric rewards the latter.

Limitations of an unfrozen LM head. Although unfreezing the LM head yields our best single result, it introduces significant downsides. First, the trainable parameter count jumps from $<1\%$ (LoRA adapters only) to $\sim 15\%$, undermining a key motivation of parameter-efficient fine-tuning. Second, the configuration is fragile: the optimal checkpoint occurs at exactly one epoch with one specific learning rate, and training even slightly longer causes chrF to degrade as the model memorizes training sonnets verbatim rather than learning transferable stylistic patterns. Third, because the word embedding matrix is shared between input and output via weight tying, updating it shifts the representation space for *all* tokens—not just those frequent in Shakespeare—which risks degrading the model’s general language competence. Finally, the success is not reproducible through principled hyperparameter search: our lr sweep at rank 4 identified 5×10^{-5} as optimal (42.73 dev chrF), yet the actual best rate of 3×10^{-4} was discovered accidentally and lies far outside the sweep’s identified optimum. This means the result, while strong, depends on a lucky configuration that would be difficult to discover systematically for a new task or dataset.

SMART regularization: loss–accuracy divergence. SMART [8] at $\lambda=1.0$ reduces dev accuracy from 0.899 to 0.874. Inspecting the training dynamics reveals why: from step 5,000 to 19,000, the training loss drops 24% ($0.374 \rightarrow 0.335$), but dev accuracy barely moves ($0.841 \rightarrow 0.875$). The adversarial perturbation term dominates the gradient, so the model optimizes for smoothness of the output distribution rather than classification accuracy. At the regularization strength we tested, SMART effectively “eats” the gradient budget that would otherwise improve the task-specific parameters. Tuning λ to a much smaller value might help, but this would require an expensive inner sweep that our compute budget did not permit.

Limitations of chrF for poetry evaluation. chrF is entirely insensitive to rhyme structure: a sonnet with perfect ABAB CDCD EFEF GG rhyme scores identically to one with no rhyme, all else equal. This explains why our rhyme-constrained decoding hurts chrF—even a small logit bias distorts the token distribution enough to produce shorter lines and occasional non-words—yet the underlying

model may actually be generating better *poetry* by a human standard. A more complete evaluation would incorporate rhyme accuracy, meter regularity, and human judgments, none of which chrF captures.

7 Conclusion

We implemented GPT-2 from scratch and systematically explored extensions for Shakespearean sonnet generation and cloze-style paraphrase detection. Our main findings are:

1. **Decoding engineering matters most for constrained generation.** The carefully designed generation function—EOS blocking, newline counting, punctuation-aware repetition penalty, and top- k /nucleus sampling—transforms degenerate output into structurally valid 14-line sonnets. This is the single largest contributor to sonnet quality.
2. **LoRA on a larger model surpasses full fine-tuning of a smaller one.** GPT-2-medium with LoRA rank 16 and an unfrozen LM head (43.08 test chrF) outperforms GPT-2 full fine-tuning (42.15 dev chrF) while updating only $\sim 15\%$ of parameters.
3. **Learning rate selection is critical for LoRA.** For paraphrase detection, a cheap rank-1 sweep identified $\text{lr } 3 \times 10^{-4}$ as optimal, spanning a 13-point accuracy gap across the search grid. For sonnet generation, the best rate was discovered accidentally, highlighting that systematic sweeps do not always find the global optimum.
4. **Many plausible extensions fail.** Rhyme-constrained decoding, best-of- N sampling, SMART regularization, bidirectional augmentation (on GPT-2-large), differential learning rates, and Shakespearean-period poetry augmentation all either hurt performance or provided no measurable gain, underscoring the value of reporting negative results.

The primary limitation of our work is evaluation: chrF rewards surface-level lexical similarity to Shakespeare but is blind to rhyme, meter, and coherence. Future work could incorporate human evaluation or composite metrics that capture formal poetic constraints. For paraphrase detection, exploring smaller SMART λ values and scaling ensembles to more seeds are natural next steps.

References

- [1] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI preprint*, 2018.
- [2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [3] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.
- [4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [5] John Schulman and Thinking Machines Lab. LoRA without regret. *Thinking Machines Lab: Connectionism*, 2025. <https://thinkingmachines.ai/blog/lora/>.
- [6] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [7] Quora. First quora dataset release: Question pairs. <https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs>, 2017.
- [8] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, 2020.

- [9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.
- [10] Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P Cunningham. LoRA learns less and forgets less. *arXiv preprint arXiv:2405.09673*, 2024.
- [11] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.
- [12] Jey Han Lau, Trevor Cohn, Timothy Baldwin, Julian Brooke, and Adam Hammond. Deep-speare: A joint neural model of poetic language, meter and rhyme. *arXiv preprint arXiv:1807.03491*, 2018.
- [13] Brian Lester et al. A GPT-2 based poem generator. *arXiv preprint arXiv:2205.08847*, 2022.
- [14] 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.
- [15] William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*, 2005.

A Sample Generated Sonnets

We show Sonnet 0 (prompt: first 3 lines) generated by the default baseline (GPT-2 full fine-tune, no decoding fixes) and our best model (GPT-2-medium, LoRA rank 16, unfrozen LM head). The prompt lines are shared; generated lines follow.

Default baseline (no decoding fixes)

*Thine eyes I love, and they, as pitying me,
Knowing thy heart torment me with disdain,
Have put on black, and loving mourners be,
To meet thee with black and black’s black look.
Thou art as black as my heart is black,
And black as my heart is black,
So thou, black as I, black as I,
Shall meet with mine eye’s surprise,
And, for my love’s sake blackly depart,
To black as I, to thy blackness devise.
But thou, black as I, must leave me
In black, to thee, black as I,
Where black’s darkest shadows do dwell.*

The model loops on “black” (15 occurrences in 13 lines) and produces only 13 lines instead of 14. The word “black” appears in the prompt, triggering degenerate repetition without a penalty.

Best model (LoRA + generation fixes)

*Those lips that Love’s own hand did make
Breathed forth the sound that said “I hate”
To me that languished for her sake; how then should she leave him alive?
And when his name calls all your thoughts to thee
And leads thy heart out of desire, keep silent, O sweet and tender, I say my dear,
Hearing what is true; but live thou now
Even as though they could not tell anything else.
This verse, too, shows nothing worth while.
But still, if love alone can so besiege you,
Therefore it might have better taught its tongue:
Thy face, in this cruel night, no flame was brighter than yours;*

For, he whose voice lies before heaven
Yet doth bear witness, truth, yet, dare.
Thou hadst beheld on earth many things;

The output has exactly 14 lines, no repetition loops, and maintains Shakespearean register throughout. While it does not follow the ABAB rhyme scheme (a limitation noted in Section 6), the language quality and structural correctness are dramatically improved.