

Metric-Aligned Sonnet Generation with LoRA and Self-Critical RL Fine-Tuning

Stanford CS224N Default Project

Timothy Yu*

Department of Computer Science
Stanford University
tyu2105@stanford.edu

Xiang Wan*

Department of Computer Science
Stanford University
oscarwan@stanford.edu

Abstract

Training on token-level cross-entropy but evaluating on sequence-level metrics creates a fundamental mismatch. One that becomes especially visible in creative generation tasks like Shakespeare sonnet continuation, where the target metric is chrF and the target quality is poetry. We address this problem with a two-stage pipeline: LoRA fine-tuning of GPT-2 followed by Self-Critical Sequence Training (SCST) with KL regularization, EMA-based advantage normalization, and an auxiliary cross-entropy term. Our full system improves dev chrF over a strong supervised baseline, and training dynamics confirm that sampled reward tracks held-out evaluation quality. However, these gains do not extend to broader form metrics, and reward continues rising past the point where dev chrF peaks, suggesting that better chrF is not the same as better poetry.

1 Key Information to include

Mentor: Ahmed Ahmed External Collaborators: None Sharing Project: No

2 Introduction

Sequence generation models are typically trained with token-level cross-entropy, yet they are often evaluated with sequence-level metrics that reflect overall output quality rather than next-token accuracy. In the sonnet continuation task, this mismatch is especially pronounced: performance is measured by chrF [1], a character n-gram F-score comparing a generated continuation against the gold poem, while standard cross-entropy training optimizes token prediction under teacher forcing. As a result, the training objective is only indirectly related to the metric used for evaluation.

This gap between training and evaluation is a standard problem in sequence generation. Policy-gradient methods such as REINFORCE [2] and Self-Critical Sequence Training (SCST) [3] address it by treating the evaluation metric as a reward and directly optimizing expected sequence-level performance. However, applying reinforcement learning to language generation is often unstable: policy-gradient updates are high variance, and optimizing reward too aggressively can damage fluency, structure, or generalization. These challenges are especially acute in our low-resource setting, which contains only 154 Shakespeare sonnets.

To address this, we use a lightweight two-stage pipeline designed to align training more closely with chrF while remaining stable and parameter-efficient. In the first stage, we fine-tune GPT-2 [4] on the sonnet corpus using LoRA adapters [5], producing a strong supervised reference policy π_{CE} . In the second stage, we apply SCST to directly optimize chrF, using π_{CE} both as initialization and as a KL anchor to limit policy drift. We further stabilize training with EMA-based advantage normalization and an auxiliary cross-entropy term to preserve fluency during reinforcement-style fine-tuning.

Our experiments show that this metric-aligned approach improves dev chrF over a strong supervised LoRA baseline (43.62 vs. 42.54). The training curves further show that increases in sampled reward generally track improvements in dev chrF, suggesting that SCST is optimizing in the intended direction. At the same time, the gains are not uniform across all quality measures: improvements in chrF do not lead to clear gains in

*Denotes equal contribution.

broader sonnet-form metrics, and extended training can overfit even as reward continues to rise. Overall, our results show that reinforcement-style fine-tuning can improve the target evaluation metric even under strict data and compute limits, while also highlighting the limitations of optimizing chrF alone for poetic generation.

3 Related Work

Large pretrained language models such as GPT-2 [4] have become a standard starting point for text generation. A common approach is to adapt these models with supervised fine-tuning, and recent work has shown that LoRA [5] can do this efficiently with a small number of trainable parameters. In our project, this LoRA fine-tuned GPT-2 model trained with cross-entropy serves as the main baseline. Our work then asks whether this strong supervised policy can be improved by directly optimizing the sequence-level metric used at evaluation time, namely chrF [1].

This idea is related to earlier work on sequence-level training for text generation. REINFORCE [2] provides the basic policy-gradient method for optimizing non-differentiable rewards, and SCST [3] extends it by using the reward of the model’s own test-time decoding output as a baseline, so updates depend on whether a sampled sequence improves on the model’s current behavior. Minimum risk training [6] likewise showed that sequence-level objectives can better match test-time evaluation than token-level training alone. Our method is closest to SCST, but adapts it to sonnet generation by using chrF [1] as the reward together with KL regularization and an auxiliary cross-entropy loss for stability. This also connects our approach to broader work on KL-constrained policy optimization and language model fine-tuning [7, 8, 9, 10], though our focus is a small-scale setting where both poem quality and sonnet form matter.

4 Approach

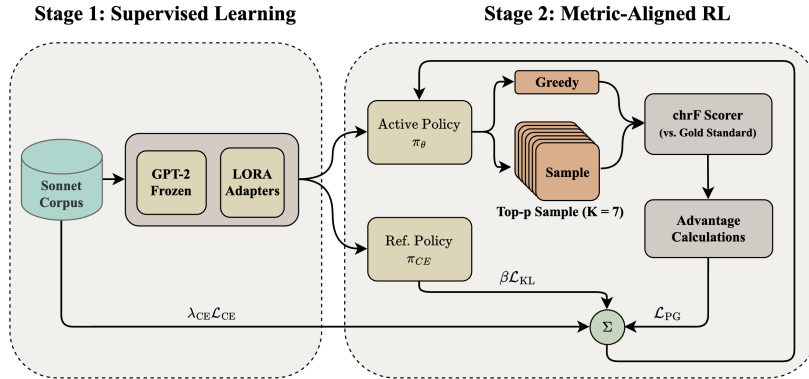


Figure 1: Two-stage fine-tuning pipeline. Stage 1 trains the LoRA-adapted GPT-2 reference policy π_{CE} via cross-entropy. Stage 2 applies our SCST-style policy-gradient fine-tuning, using greedy decoding as a baseline and optimizing a KL-regularized hybrid objective.

We develop a two-stage, metric-aligned fine-tuning pipeline for sonnet continuation that targets chrF [1] while remaining parameter-efficient (Figure 1): (1) learn a supervised reference policy π_{CE} via cross-entropy training and LoRA [?], then (2) directly optimize chrF using policy-gradient fine-tuning with KL regularization and an auxiliary CE loss to stabilize training and preserve fluency [11, 9, 10]. All models are implemented from scratch in PyTorch.

4.1 Baselines

We evaluate three baselines of increasing sophistication. First, we evaluate pretrained GPT-2 (124M) without any fine-tuning [4], which establishes a lower bound on task performance. Second, we fine-tune the full GPT-2 model using teacher-forced cross-entropy on the sonnet corpus, testing whether standard supervised fine-tuning alone is sufficient. Finally, for parameter-efficient training, we fine-tune GPT-2 using LoRA adapters [?]: we freeze the backbone and insert rank $r = 32$, $\alpha = 64$ adapters into the top 6 attention layers

(e.g., W_v, W_o), training only the adapters under the same teacher-forced cross-entropy objective. The best dev chrF checkpoint from this LoRA sweep serves as both our PEFT baseline and the reference policy π_{CE} for KL-regularized RL fine-tuning (Figure 1, Stage 1).

4.2 Metric-Aligned RL Fine-Tuning

Starting from π_{CE} , we apply policy-gradient fine-tuning to directly optimize chrF (Figure 1, Stage 2). We use REINFORCE [2] as our conceptual framework, treating chrF as a sequence-level reward signal, and adopt SCST [3] as a variance-reduction variant that uses the model’s own greedy output as a baseline.

Rollout and reward. For each prompt (the first three lines of a sonnet), we generate one greedy continuation s_{greedy} and $K = 7$ sampled continuations $\{s_k\}$ via nucleus sampling (top- $p = 0.95$) [12]. Each continuation is scored by chrF against the gold continuation, yielding rewards $r_k, r_{\text{greedy}} \in [0, 100]$. We use $K = 7$ due to our compute constraints (154 sonnets, single GPU).

Advantage estimation. We define the raw advantage for each sampled continuation as the difference between its reward and the greedy baseline reward, scaled by $\frac{1}{100}$:

$$A_k = \frac{r_k - r_{\text{greedy}}}{100}.$$

This scaling accounts for the fact that chrF is measured on a $[0, 100]$ scale, so reward differences are naturally expressed in chrF points and can lie in $[-100, 100]$. Dividing by 100 converts the advantage to an approximately unit-scale quantity before adaptive normalization.

This $\frac{1}{100}$ factor is not redundant with the subsequent EMA normalization. In our implementation, the EMA standard deviation is initialized to $\sigma_{\text{EMA}}^{(0)} = 1.0$, so during the first few batches the EMA statistics have not yet adapted to the empirical reward scale. Without the initial $\frac{1}{100}$ scaling, raw reward differences in the tens of chrF points would lead to substantially larger normalized advantages during this warmup period, producing overly large early policy-gradient updates. The scaling therefore acts as an implicit gradient stabilizer in early training, complementing explicit learning-rate warmup.

We further normalize advantages using an EMA of the running mean μ_{EMA} and standard deviation σ_{EMA} :

$$\hat{A}_k = \frac{A_k - \mu_{\text{EMA}}}{\sigma_{\text{EMA}} + \epsilon},$$

where μ_{EMA} and σ_{EMA} are updated after each batch with decay $\gamma = 0.2$, and $\epsilon = 10^{-8}$ prevents division by zero. EMA-based normalization is preferable to per-batch statistics under our small-batch regime, where batch-level estimates of advantage variance are noisy [13]. Updates with $|\hat{A}_k| < \tau = 0.5$ are discarded to suppress near-zero gradient noise.

Training objective. The policy-gradient loss (Figure 1, $\mathcal{L}_{\text{PG}}$) is:

$$\mathcal{L}_{\text{PG}} = -\frac{1}{|\mathcal{A}|} \sum_{k \in \mathcal{A}} \hat{A}_k \log \pi_{\theta}(s_k | p),$$

where $\mathcal{A}$ is the set of retained rollouts after the threshold filter. To mitigate policy collapse and reward over-optimization, we optimize a KL-regularized hybrid objective:

$$\mathcal{L} = \mathcal{L}_{\text{PG}} + \beta \mathcal{L}_{\text{KL}} + \lambda_{\text{CE}} \mathcal{L}_{\text{CE}},$$

where $\mathcal{L}_{\text{KL}} = D_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{CE}})$ penalizes deviation from the supervised reference policy [9, 10], preventing the policy from drifting into degenerate but high-reward outputs, and $\mathcal{L}_{\text{CE}}$ provides auxiliary teacher-forced supervision to preserve language quality [11]. All loss terms are combined at the Σ node shown in Figure 1.

4.3 Original Contributions

While REINFORCE [2] and KL-regularized RL fine-tuning [9, 10] are well-studied for sequence-level objectives, their application to character n-gram metrics like chrF is non-obvious. chrF was designed as an evaluation metric for machine translation [1], not as a reward signal, since it is non-differentiable, operates at the character level rather than the token level, and has no natural decomposition into per-step rewards. Naively treating it as a reward can produce degenerate policies that game the metric by, for instance, repeating high-frequency character sequences without producing coherent text. Our primary contribution is

demonstrating that chrF can be used as a stable training signal with appropriate regularization, and that doing so yields genuine improvements in both metric score and structural quality. To achieve this, we propose: (i) $\frac{1}{100}$ advantage scaling as an implicit gradient stabilizer during EMA warmup, keeping early policy-gradient updates conservative while EMA statistics converge; (ii) EMA-based advantage normalization for stable small-batch training in the low-data regime; and (iii) a KL and auxiliary CE regularized objective that prevents the policy from exploiting chrF in degenerate ways while preserving fluency. We also provide a systematic ablation that isolates the contribution of each stabilization component.

5 Experiments

5.1 Data

We use the Shakespeare sonnet dataset provided with the CS224N default project, consisting of 154 sonnets in `sonnets.txt`. The task is conditional continuation: given the first three lines of a sonnet as a prompt, the model generates the remaining lines. We train on the full `sonnets.txt`, evaluate on prompts from `sonnets_held_out_dev.txt`, and compute dev chrF against gold continuations in `TRUE_sonnets_held_out_dev.txt`. No external data is used. The dataset is notably small, 154 sonnets total, which motivates our parameter-efficient approach and informs several of our stabilization design choices, as discussed in Section 4.

5.2 Evaluation Method

We evaluate all models on prompts from `sonnets_held_out_dev.txt`, computing dev chrF [1] against the gold continuations in `TRUE_sonnets_held_out_dev.txt`. chrF is a character n -gram F-score that measures holistic similarity between a generated sequence and a reference; it is the primary metric for this task and the quantity we directly optimize during RL fine-tuning.

All models are evaluated under a fixed decoding protocol to ensure fair comparison across training methods. At evaluation time we decode greedily, use a fixed random seed (11711), and cap generation at 200 tokens or 11 newline characters, whichever comes first. The newline cap reflects the 14-line structure of Shakespearean sonnets given the 3-line prompt. The 200-token limit prevents a pathological behavior under policy-gradient training where, when a sampled continuation has negative advantage, the model can reduce the magnitude of the loss by generating increasingly long sequences, spreading the negative advantage across more tokens. Applying a fixed cap removes this incentive while still providing sufficient budget to cover a full sonnet given typical line lengths. This protocol is applied identically to all baselines and fine-tuned models, ensuring that differences in reported performance reflect differences in model parameters rather than decoding settings.

In addition to chrF, we report four secondary metrics computed directly from generated outputs to capture dimensions of quality that chrF alone does not reflect:

- *Form compliance*: the fraction of generated poems with exactly 14 non-empty lines. The Shakespearean sonnet is typically defined by its 14-line structure, and a model can achieve reasonable chrF while producing structurally malformed output[14].
- *Line count*: the mean number of non-empty lines per poem, complementing form compliance by revealing systematic over- or under-generation trends rather than just binary pass/fail compliance[15].
- *Characters per line*: the mean character count per non-empty line. Shakespeare’s sonnets average 42.9 characters per line, providing a natural reference for whether generated lines are pathologically short or long[15, 16].
- *Repetition*: the mean fraction of character trigram positions that repeat within the same poem, serving as a proxy for output degeneration. Models that collapse into looping or repetitive text score high here; lower is generally better, with Shakespeare’s sonnets sitting at a baseline of 25.18%[17, 18].

5.3 Experimental Details

Unless otherwise noted, all fine-tuned models are optimized with AdamW on a single NVIDIA A5000 GPU.

Frozen GPT-2. We evaluate the pretrained GPT-2 (124M) model directly without any fine-tuning or parameter changes, using the default model weights from Hugging Face. No hyperparameter tuning was performed; generation uses greedy decoding with the fixed evaluation protocol described in Section 5.2.

CE Full Fine-Tuning. We fine-tune the full GPT-2 model on `sonnets.txt` using teacher-forced cross-entropy with a learning rate of 5×10^{-5} , batch size of 8, and 20 epochs. All model parameters are updated during training. Training takes approximately 10 minutes.

CE + LoRA. We freeze the GPT-2 backbone and train only LoRA adapters with rank $r = 32$, $\alpha = 64$, dropout 0.1, applied to the top 6 attention layers (W_v, W_o). We train for 20 epochs with learning rate 5×10^{-4} and batch size 8, completing in under 15 minutes.

LoRA Sweep. To identify the strongest supervised reference policy π_{CE} , we run a two-stage hyperparameter sweep. In the first stage, we evaluate 36 configurations across rank $r \in \{16, 32, 64\}$, $\alpha \in \{32, 64\}$, dropout $\in \{0.0, 0.1\}$, and top layers $\in \{4, 6, 8\}$ for 5 epochs each, taking approximately 2 hours total. In the second stage, we fully train the top 10 configurations for 20 epochs, taking an additional 2 hours. All sweep runs used batch size 8 and seed 11711. The best checkpoint is saved as π_{CE} and used as the initialization and KL anchor for all RL fine-tuning.

SCST Fine-Tuning. Starting from π_{CE} , we perform a targeted hyperparameter search to identify stable SCST training settings. We began with $K = 4$ rollouts per prompt, a common small-sample setting under limited compute, and found that increasing K improves reward estimation stability. We therefore increase to $K = 7$, which gives slightly better dev performance while keeping training time manageable. We also find that the learning rate and KL coefficient β are the most sensitive hyperparameters, since overly aggressive updates can lead to policy collapse under sequence-level training. We therefore sweep both larger and smaller learning rates as well as several β values.

Our final configuration uses learning rate 4×10^{-5} , $\beta = 0.1$, $\lambda_{\text{CE}} = 0.01$, $K = 7$ rollouts per prompt, advantage threshold $\tau = 0.5$, sampling temperature 1.2, top- $p = 0.95$, EMA decay $\gamma = 0.2$, and a 1-epoch linear warmup. We train for 100 epochs with batch size 16, stopping at approximately epoch 85 where dev chrF begins to plateau, indicating the onset of overfitting to the small training corpus. Only LoRA parameters are updated throughout; the backbone remains frozen. Full SCST training takes approximately 20 hours.

5.4 Results

Paraphrase Detection. While our project primarily focuses on sonnet generation, we also trained the paraphrase detection model required for the CS224N default project. Using standard cross-entropy training, we achieve a final dev accuracy of 0.872 and test accuracy of 0.859. Because this component was not the focus of our work, we do not analyze it further and instead concentrate on the sonnet generation results below.

Table 1: Performance of various methods on the Shakespeare Sonnets Dataset

Method	Dev chrF	Test chrF	Form (%) $\uparrow$	Line Count	Chars/Line	Repetition (%)
<i>Baselines</i>						
Frozen GPT-2	39.544	–	0.0	6.67	104.9	17.22
CE Full Fine-Tuning	42.452	–	8.3	13.00	46.3	22.34
CE LoRA	42.475	–	16.7	13.08	47.8	18.25
Best LoRA Sweep (π_{CE})	42.544	–	8.3	13.50	44.7	18.22
<i>Metric-Aligned Fine-Tuning</i>						
SCST + KL + EMA + CE (Ours)	43.621	43.607	8.3	13.17	48.1	22.36
<i>Ablations</i>						
SCST + KL + EMA	43.336	–	25.0	13.25	48.8	24.40
SCST + KL	42.532	–	8.3	13.42	45.4	21.53
SCST	42.647	–	8.3	10.48	65.4	26.81
<i>Hyperparameter Variants</i>						
Ours with $\beta = 0$	43.283	–	0.0	9.50	78.0	29.07
Ours with $\beta = 1$	42.961	–	16.7	13.17	47.9	24.72
Ours with lr = 8×10^{-5}	43.230	–	25.0	12.58	51.5	23.24
Ours with lr = 2×10^{-5}	43.074	–	41.7	13.33	47.5	22.70

For reference, Shakespeare’s mean characters per line was 42.9, and repetition percentage was 25.18.

The quantitative results for all sonnet generation experiments are shown in Table 1. Our proposed method, *SCST + KL + EMA + CE*, achieves a final test chrF score of **43.607**, as well as the highest dev chrF score of **43.621**, improving upon the strongest supervised baseline π_{CE} (42.544) by approximately **+1.08**

chrF. This improvement demonstrates that directly optimizing the evaluation metric through sequence-level reinforcement learning can provide measurable gains over standard cross-entropy training.

Although our method achieves the best chrF score, it does not dominate every secondary metric. Instead, it remains broadly comparable to the supervised baselines across structural measures. For example, the average line count (13.17) and characters per line (48.1) remain close to the statistics of the best LoRA baseline (13.50 lines and 44.7 characters per line), while the repetition rate (22.36%) is also similar to the cross-entropy baselines. This suggests that metric-aligned fine-tuning improves similarity to the reference continuation without substantially degrading structural properties of the generated poems.

The ablation experiments provide additional insight into the role of the different training components. In particular, removing the KL regularization term ($\beta = 0$) leads to clear structural collapse: the resulting model produces only 9.50 lines on average and 0% form compliance, with extremely long lines (78 characters per line) and high repetition (29.07%). This indicates that KL regularization plays a critical role in preventing the policy from drifting too far from the supervised reference model during sequence-level optimization.

More generally, the KL coefficient β appears to have the strongest influence on the structural metrics. Increasing β changes the balance between optimizing chrF and maintaining the behavior of the original policy, which directly affects line structure and repetition statistics. These results suggest that KL regularization is the primary mechanism controlling policy stability in our training setup.

We also observe that learning rate affects training stability. Larger learning rates slightly improve exploration but can degrade structural metrics, while smaller learning rates lead to more stable but slower improvements in chrF. Overall, the combination of KL regularization, EMA-normalized advantages, and auxiliary cross-entropy provides the most stable training behavior and yields the best overall dev chrF score.

6 Analysis

6.1 Reward and Evaluation Metrics

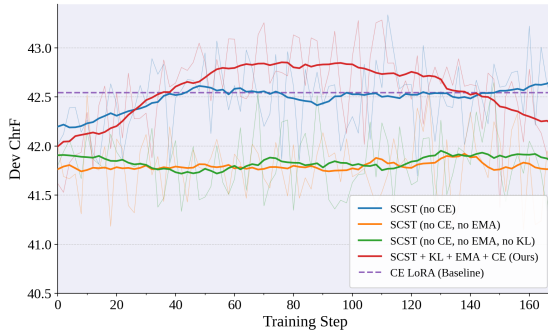


Figure 2: Dev chrF during training for SCST variants

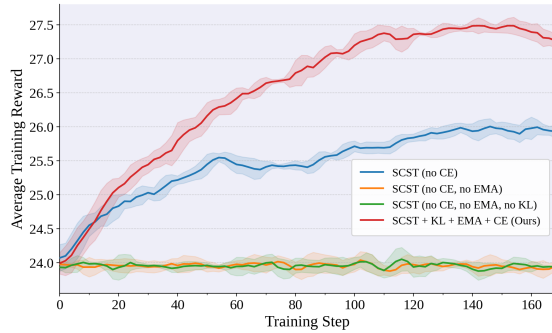


Figure 3: Average sampled reward for SCST variants

Figure 2 and Figure 3 present our ladder ablations across training, tracking dev chrF and average sampled reward respectively. The main observation is that the two metrics move together, since models that accumulate higher reward also tend to score better on dev chrF, giving us confidence that our reward signal is actually tracking translation quality rather than some spurious proxy. Our full method, SCST + KL + EMA + CE, leads on both plots throughout training, though the ablations reveal that the story is not simply “more components = better,” as each one seems to be doing something distinct.

The most dramatic collapse comes from removing EMA. SCST (no CE, no EMA) stays nearly flat in reward across all 170 steps, barely moving from its starting point, suggesting that without EMA to stabilize the reference policy, gradient updates become too noisy to learn from at all. By contrast, SCST (no CE) still climbs steadily and picks up some dev chrF gain, as it just plateaus lower, showing that CE initialization helps but is not strictly necessary. The role of KL is subtler here; the no-KL curve tracks close to the no-EMA curve in both metrics, though Table 1 reveals it is quietly doing important work for structural quality, as form scores drop noticeably when it is removed.

For the full model, dev chrF rises until around step 80 and then gradually declines even as sampled reward continues to climb. We interpret this as the model beginning to overfit to its on-policy training distribution, getting better at optimizing the reward it sees during rollouts, but at the cost of generalization to held-out data. This makes dev chrF a more reliable stopping criterion than reward alone, and it is why we use early stopping rather than training to convergence.

6.2 Optimization and Regularization Dynamics

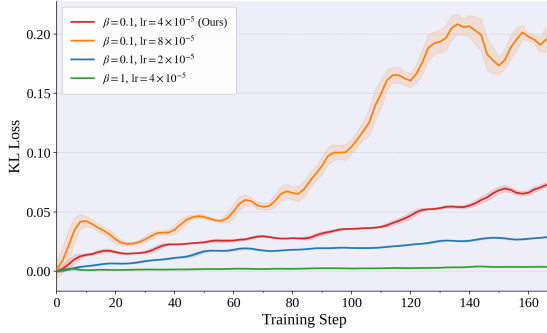


Figure 4: KL Loss during training for ablations

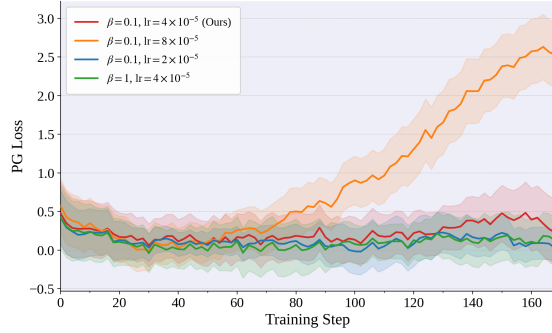


Figure 5: PG Loss during training for ablations

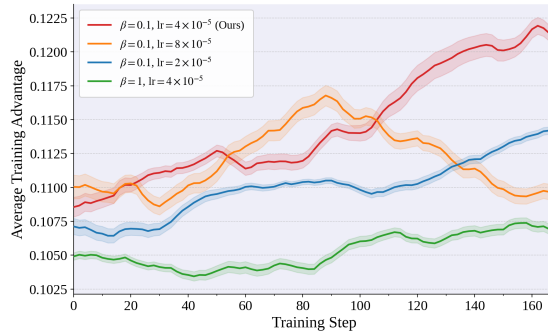


Figure 6: Average advantage during training for ablations

Figures 4, 5, and 6 show why we landed on $\beta = 0.1$ and $\text{lr} = 4 \times 10^{-5}$. We can see that clearest failure mode is the high learning rate, 8×10^{-5} , where KL loss climbs sharply throughout training and PG loss explodes after step 60, while the average advantage spikes with high variance before collapsing, which is a textbook sign of policy divergence. Dropping to 2×10^{-5} fixes the instability but introduces a different problem, where both losses stay flat and the advantage creeps up only slowly, meaning the model is stable but barely learning. Our choice of 4×10^{-5} sits between these, gaining advantage steadily without the late-training blowup.

The β variants tell a complementary story. Increasing it to 1 keeps the policy so tightly anchored to the LoRA initialization that KL loss barely moves and PG loss stays nearly flat, but the advantage never meaningfully grows either. The regularization is doing its job too well, penalizing useful updates alongside harmful ones. At $\beta = 0.1$, we get enough room for the policy to actually improve while still keeping divergence in check. Neither of these ablations reveal a magic number, but together they give us confidence that our final setting is not just an arbitrary choice; it occupies a reasonable middle ground that the alternatives on either side clearly do not.

6.3 Structural Integrity

We now shift from metric-level comparisons to the overall structural integrity of the sonnets produced by different methods. This qualitative view helps contextualize the trends from the previous subsection. We initially expected form compliance to be the most informative, since Shakespearean sonnets are defined by

its 14-line form. However, in the qualitative examples, this binary metric turns out to be less informative than expected, as models with higher form scores can still be structurally weak, while average line count and local coherence provide the clearest picture. In particular, the $\beta = 0$ ablation further supports the earlier finding that KL regularization is important not because it dramatically changes reward or dev chrF, but because it helps preserve poetic form and local coherence. Without the KL term, the policy is freer to chase reward aggressively, but this comes at the cost of structural control such as syntax degrades more quickly, repetitions increase, and the generated sonnet is more likely to have malformed phrasing:

“To dedicate thee both to wonder and thought praise mean what harm thy heart bids mine disagree”
“mine remembrance mine shall help thy friends” (Full sample in Appendix A.4)

However, our comparison with the Best LoRA baseline is more mixed. Although our full method achieves the best dev chrF in Table 1, we do not observe a clear qualitative improvement over LoRA in these examples, and our method may even be worse in local coherence. The LoRA continuation is awkward and semantically imperfect, but lines such as

“Whither most their lover should in evil example reign”
“But thou knowst that, those shadows stray” (Full sample in Appendix A.2)

still read as recognizable poetic lines. In contrast, our SCST sample shows stronger prompt-related affect, but also more severe semantic breakdown:

“my favourite ghost hath, drawing my voice anew”
“Lending me absolute purposive ransom” (Full sample in Appendix A.3)

Taken together, these examples suggest that SCST improves the reward-aligned metric, but does not deliver a clear qualitative gain over the LoRA baseline. In turn, this reinforces our broader conclusion that chrF alone does not fully capture poetic coherence.

7 Conclusion

Starting from a LoRA-tuned GPT-2 baseline, we applied SCST with KL regularization, EMA-based advantage normalization, and an auxiliary cross-entropy objective to directly optimize chrF for sonnet continuation. The full system achieves the best dev chrF in our experiments, and our training curves confirm that sampled reward and dev chrF move together, suggesting that SCST is genuinely optimizing a signal aligned with evaluation. The ablations further show that each stabilization component contributes: EMA and CE sustain learning, while KL is especially important for preserving structural quality.

The primary limitation we found is that optimizing chrF alone only captures part of what makes a good continuation. Our qualitative examples also show that the LoRA baseline sometimes produces smoother, more coherent output despite scoring lower on chrF, and the SCST model does not consistently improve broader structural metrics. The small dataset also makes RL training more sensitive to instability and overfitting, as reflected in the dev chrF decline after step 80 even as reward continues to climb.

The clearest direction for future work is richer reward design. Specifically, we noticed a single reference-based metric encourages character-level overlap with the target but ignores rhyme, meter, coherence, and rhetorical development. We think that incorporating multi-objective rewards that capture these structural and fluency properties would better reflect sonnet quality, and would be a natural next step for applying sequence-level RL to literary generation more seriously.

Team contributions

Timothy implemented the baseline model, full-model fine-tuning, the stabilization components for SCST, refining the method’s performance, baseline evaluations, and SLURM cluster setup. Xiang implemented LoRA fine-tuning, the LoRA hyperparameter sweep, and the core SCST training algorithm. Both collaborated on the ablation studies, shared responsibility for running experiments, and contributed equally to project ideation and report writing.

References

- [1] Maja Popović. chrF: Character n-gram f-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*, 2015.
- [2] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.
- [3] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.
- [5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [6] Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.
- [7] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897, 2015.
- [8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [9] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.
- [10] Tomasz Korbak, Ethan Perez, Christopher L. Buckley, Martin Szummer, Stephen Casper, and Dylan Hadfield-Menell. RL with KL penalties is better viewed as bayesian inference. In *Findings of EMNLP*, 2022.
- [11] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.
- [12] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2020.
- [13] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.
- [14] Melanie Walsh, Anna Preus, and Maria Antoniak. Sonnet or not, bot? poetry evaluation for large models and datasets. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.
- [15] Aitor Ormazabal, Mikel Artetxe, Manex Agirrezabal, Aitor Soroa, and Eneko Agirre. Poelm: A meter- and rhyme-controllable language model for unsupervised poetry generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022.
- [16] Hugo Gonçalves Oliveira, Raquel Hervás, Alberto Díaz, and Pablo Gervás. Multilingual extension and evaluation of a poetry generator. *Natural Language Engineering*, 23(6):929–967, 2017.
- [17] 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.
- [18] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texus: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018.

A Qualitative Examples

Note. All examples in this appendix use the same prompt and gold continuation, drawn from Shakespeare’s Sonnet 132. The prompt (first 3 lines) and gold continuation (remaining 11 lines) are shown once below and held fixed across all subsections. Each subsection shows only the model’s generated output.

Prompt

Thine eyes I love, and they, as pitying me,
Knowing thy heart torments me with disdain,
Have put on black and loving mourners be,

Gold Continuation

Looking with pretty ruth upon my pain.
And truly not the morning sun of heaven
Better becomes the grey cheeks of the east,
Nor that full star that ushers in the even
Doth half that glory to the sober west,
As those two mourning eyes become thy face:
O, let it then as well beseem thy heart
To mourn for me, since mourning doth thee grace,
And suit thy pity like in every part.
Then will I swear beauty herself is black,
And all they foul that thy complexion lack.

A.1 Baseline 0 - Frozen GPT-2

Generated Continuation

Thine eyes I love, and they, as pitying me,
Knowing thy heart torment me with disdain,
Have put on black, and loving mourners be,
For no other ness God this concerning:
High earth forbid what youth died pray thee, no MORE be of alminary fruit rejoice.
Use lambs unto thine name, put on thine horns;
wise eyes behold unclean sins to thy fools,
Stop back from wrath with your those harmful curses,
Thy devils in Paradise want not before, night to dark.
The Hebrews were to worship for a time nothing of this,
But now to all lawd apply, Ciao see,
To pity all within.
God our native

Interpretation. The frozen GPT-2 output quickly breaks away from the sonnet’s theme and structure, drifting into fragmentary religious language and syntactically unstable phrases. Although it preserves a few archaic surface cues, it does not maintain the logic, imagery, or poetic coherence of the gold continuation.

A.2 Baseline 1 - Best LoRA Sweep (π_{CE})

Generated Continuation

Thine eyes I love, and they, as pitying me,
Knowing thy heart torment me with disdain,
Have put on black, and loving mourners be,
Whither most their lover should in evil example reign.
Though my worst vengeance could for thee change,
How are they done whom thou thus affection,
Which makes my blood lack but its opinion?
Mine all-wonderful sympathy is short;
And charity seeking a precious body can not profit
Neither if it retain itself, for thyself is mixed
Seeking something to help his cause.
Whatever mine was, she iniquity and my want
Were sure to employ him only by effecting much.
But thou knowst that, those shadows stray,
When as I have gone away

Interpretation. Compared to the frozen baseline, the LoRA-tuned model produces output that is more poem-like in length and lineation, and it retains more of the prompt’s archaic tone. However, the semantics remain unstable, as several lines are grammatically awkward, the imagery does not develop clearly, and the continuation does not recover the tight argumentative structure of the original sonnet.

A.3 SCST + KL + EMA + CE (Ours)

Generated Continuation

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 avoid jealous lest thy funeral!
Who at my pray’s leave hath all thee worth now
To teach my deceased savage feeling
Now beyond my open pray he covers the sheet.
I find thee grieved by respect with still,
That letting regrets overtake thee,
Though I labor wholly against you alone show,
They for this none can raise nor despise.
They made my credit of conscience used appear,
And now give sightless my anger becomes,
One which my favourite ghost hath, drawing my voice anew,
Lending me absolute purposive ransom
.” Mine beautiful original mark supplies this

Interpretation. This output is more continuous and poetic in texture than the weakest baselines, and it preserves some of the mourning-related mood introduced by the prompt. Still, it remains semantically unstable, with several malformed phrases and an inconsistent line structure that moves away from the sonnet form. Despite this, the sonnet appears to be partially thematic but still noticeably incoherent.

A.4 Ablation - Ours with $\beta = 0$

Generated Continuation

Thine eyes I love, and they, as pitying me,
Knowing thy heart torment me with disdain,
Have put on black, and loving mourners be,
Seeing mine face the sunny eyes, sweet beauty that amstead me,
Their hide for thee's grace so profits the thought is mine,
To dedicate thee both to wonder and thought praise mean what harm thy heart bids mine disagree.
Therefore thy lips compare mine eyes and hope, yours to each weak judge,
Yet may thee thy other mine teach others to say "I love thee with particular beauty that is yours,
Therefore mine eyes thou mightst wit thy hearts none being mine doth I praise thee."
So must you esteem mine: mine remembrance mine shall help thy friends by this name mine quote this verse
both

Interpretation. Removing the β -weighted component leads to highly repetitive and less controlled generation. The output retains some local poetic vocabulary, but it becomes overly tangled syntactically and loses the disciplined progression and rhetorical balance expected from a Shakespearean sonnet.