

Robust GPT-2 Fine-Tuning via LoRA and Smoothness-Inducing Regularization

Pooya Nabavi

Department of Computer Science, Stanford University

pnabavi@stanford.edu

Late Days: This submission uses 2 late days

Abstract—We present a complete investigation of parameter-efficient and regularization-based fine-tuning of GPT-2 for three NLP tasks: sentiment analysis (SST-5 and CFIMDB), paraphrase detection (Quora), and sonnet generation. Starting from a fully fine-tuned GPT-2 Small baseline, we implement and evaluate two extensions: Low-Rank Adaptation (LoRA) [1], which restricts weight updates to a low-rank subspace, and SMART [8], which penalizes sensitivity to adversarial input perturbations via Bregman proximal optimization. Our LoRA+SMART classifier achieves 0.518 dev accuracy on SST-5 (surpassing the 0.515 full fine-tuning baseline) and 0.976 on CFIMDB. For sonnet generation, our LoRA model achieves a chrF score of 41.455 on the test set, exceeding the staff baseline of 41.208. For paraphrase detection, LoRA+SMART achieves 0.856 dev and 0.833 test accuracy after 15 epochs, revealing an important dataset-size-dependent tradeoff: LoRA+SMART excels on small datasets where regularization is critical, while full fine-tuning retains an advantage on large-scale tasks.

I. INFORMATION

- **Project type:** Default
- **TA mentor:** Alisa Levin
- **External collaborators:** No
- **External mentor:** No
- **Sharing project:** No

II. INTRODUCTION

Large pre-trained language models such as GPT-2 [2] have demonstrated strong performance across a wide range of NLP tasks

through fine-tuning. However, standard full fine-tuning updates all model parameters, which can be computationally expensive and prone to overfitting, particularly on small datasets. This raises two practical questions: (1) can we match full fine-tuning performance while updating far fewer parameters, and (2) can explicit regularization further improve generalization?

This work investigates these questions in the context of GPT-2 Small (124M parameters) fine-tuned for sentiment analysis, paraphrase detection, and sonnet generation. We implement two complementary techniques. First, Low-Rank Adaptation (LoRA) [1] decomposes weight updates as $\Delta W = BA$ where B and A are low-rank matrices, drastically reducing trainable parameters while introducing no inference overhead. Second, SMART [8] adds a smoothness-inducing adversarial regularization term that penalizes large output changes under small input perturbations, paired with Bregman proximal point optimization to stabilize training.

Our main contributions are: (1) a modular LoRA implementation targeting query and value projections across all GPT-2 layers (rank $r = 8$, replacing 24 linear layers); (2) a SMART regularizer with Bregman proximal optimization integrated into the fine-tuning loop; (3) empirical evaluation across three tasks revealing a dataset-size-dependent tradeoff; and (4) analysis showing LoRA+SMART provides the largest gains where regularization is most

needed.

III. RELATED WORK

Parameter-efficient fine-tuning. The high cost of fine-tuning large language models has motivated a family of parameter-efficient methods. Adapter layers [6] insert small bottleneck modules between transformer layers, training only these while freezing the backbone. Prefix tuning [7] prepends trainable continuous vectors to the input, avoiding any modification of model weights. LoRA [1] takes a different approach: rather than adding new modules, it decomposes weight updates into low-rank products $\Delta W = BA$, achieving strong performance with as few as 0.1% of the original parameters. Unlike adapters, LoRA introduces no additional inference latency since ΔW can be merged into the original weights after training. A theoretical motivation for LoRA comes from the observation that the intrinsic dimensionality of the fine-tuning objective is much lower than the full parameter space [14]—the task-relevant update lives in a low-dimensional subspace, and LoRA explicitly parameterizes this. Our work applies LoRA specifically to the query and value projections of GPT-2’s self-attention layers, following the recommendation of Hu et al. [1] that distributing capacity across more layers at lower rank is preferable to concentrating it in fewer layers at higher rank. An important design choice is the initialization: B is set to zero so that $\Delta W = 0$ at the start of training, ensuring that fine-tuning begins from the pre-trained model behavior.

Regularization for fine-tuning. Overfitting during fine-tuning is a well-documented challenge, particularly on small datasets or when transferring to domains distant from the pre-training distribution. Standard approaches include dropout, weight decay, and early stopping. L^2 -SP [15] regularizes fine-tuned parameters toward their pre-trained values, exploiting the prior that the pre-trained model already encodes useful representations. SMART [8]

proposes a more principled approach based on smoothness-inducing adversarial training: the model is penalized for producing large output changes under small input perturbations in the embedding space. Crucially, these perturbations are found via projected gradient *ascent* (maximizing the divergence), making the regularizer adversarial and thus more effective than isotropic noise. Bregman proximal point optimization [8] further stabilizes parameter updates by penalizing deviation from a snapshot of the model taken at the start of each epoch. This combination has been shown to improve robustness and generalization across multiple GLUE benchmarks [13], and our work represents one of the first applications of SMART to GPT-2 in a multi-task setting.

GPT-2 for classification and generation. While GPT-2 was originally designed for language modeling [2], it has been successfully adapted for classification by extracting the hidden state at the last token position and projecting through a linear head [4]. This is in contrast to BERT-style models [5], which use a dedicated [CLS] token; for GPT-2 the last token is the natural choice given the autoregressive architecture. Large language models have also demonstrated strong in-context learning capabilities [3]; our cloze-style paraphrase formulation draws on this intuition by framing the task as next-token prediction. For generation tasks, nucleus sampling [12] addresses the degeneration problem in neural text generation by truncating the vocabulary to a dynamic top- p probability mass, avoiding low-probability tokens while preserving diversity.

Fine-tuning for creative generation. Fine-tuning language models for constrained creative generation poses unique challenges. The training data is extremely small (typically hundreds of examples), making regularization critical. Prior work on poetry generation has applied RNN-based models [16] and transformer-based models, but systematic evaluation using automatic metrics such as chrF [11] remains rel-

atively rare. Our work contributes an empirical evaluation of LoRA for sonnet generation, showing that parameter-efficient adaptation is particularly well-suited to tiny creative datasets.

IV. APPROACH

A. Model Architecture and Base Implementation

We implement GPT-2 Small (124M parameters, 12 transformer layers, 768-dimensional hidden states, 12 attention heads) [2]. The core components are: (1) causal self-attention with scaled dot-product attention $\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V$ where M enforces the autoregressive property; (2) pre-norm GPT-2 layers with LayerNorm before both attention and feed-forward sub-layers and residual connections after each; (3) token and positional embeddings as a sum of learned embeddings; and (4) AdamW with decoupled weight decay. Pre-trained weights are loaded from HuggingFace’s GPT-2 checkpoint.

B. Task Formulations

Sentiment classification. For SST-5 and CFIMDB, we extract the hidden state at the last non-padding position and project through a task-specific linear head: $\hat{y} = W_c h_{\text{last}} + b_c$, trained with cross-entropy loss. We evaluate last-linear-layer (LLL) and full-model (FM) regimes.

Paraphrase detection. For the Quora dataset, we adopt a cloze-style formulation: Is “{s1}” a paraphrase of “{s2}”? Answer “yes” or “no”. A two-class linear head on the last token’s representation predicts paraphrase (token 8505: “yes”) or non-paraphrase (token 3919: “no”), optimized with cross-entropy.

Sonnet generation. We fine-tune GPT-2 as a language model on 154 Shakespeare sonnets, conditioning generation on the first line. At inference, we use nucleus sampling with temperature $T = 1.3$ and $p = 0.95$.

C. LoRA Implementation

We implement a `LoRALinear` module that wraps an existing `nn.Linear` layer. For a frozen weight matrix $W_0 \in \mathbb{R}^{d \times k}$, the forward pass computes:

$$h = W_0 x + \frac{\alpha}{r} \cdot B A x \quad (1)$$

where $B \in \mathbb{R}^{d \times r}$ is initialized to zero, $A \in \mathbb{R}^{r \times k}$ is initialized with Kaiming uniform noise, $r = 8$, and $\alpha = r$. Initializing $B = 0$ ensures $\Delta W = 0$ at training start. Only A and B receive gradient updates; W_0 is frozen throughout.

We apply LoRA to W_Q and W_V in each of the 12 GPT-2 self-attention layers, replacing 24 linear layers in total. This reduces trainable backbone parameters from 85M to approximately 295K—a 99.7% reduction.

D. SMART Regularization

SMART [8] augments the task loss with a smoothness penalty:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda_s \cdot \frac{1}{n} \sum_{i=1}^n \max_{\|\tilde{x}_i - x_i\| \leq \epsilon} D_{\text{KL}}(f(\tilde{x}_i) \| f(x_i)) \quad (2)$$

The inner maximization is approximated via one step of projected gradient ascent in the embedding space with $\epsilon = 10^{-6}$ and $\lambda_s = 0.02$.

Bregman Proximal Optimization (BPO). At the start of each epoch, we take a snapshot θ_t of the model parameters and augment the objective with $\mathcal{L}_{\text{prox}} = \frac{\mu}{2} \|\theta - \theta_t\|_2^2$ with $\mu = 0.001$, preventing catastrophically large updates. The combined loss is:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_s \cdot \mathcal{L}_{\text{SMART}} + \mu \cdot \mathcal{L}_{\text{prox}} \quad (3)$$

When LoRA is enabled, SMART perturbations operate on the input embeddings while BPO operates on all trainable parameters including the LoRA matrices A and B . This means SMART’s function-space regularization acts orthogonally to LoRA’s parameter-space constraint: LoRA limits which directions the model can adapt, while SMART limits how sensitive the adapted function is.

V. EXPERIMENTS

A. Data

We use the course-provided datasets. **SST-5** [10]: 8,544 / 1,101 / 2,210 train/dev/test, 5-class sentiment. **CFIMDB**: 1,707 / 245 / 488, binary sentiment. **Quora**: 283,003 / 40,429 / 80,858, binary paraphrase detection. **Sonnet generation**: 154 Shakespeare sonnets, evaluated using chrF [11].

B. Experimental Setup

All experiments use GPT-2 Small on a single NVIDIA A100-80GB GPU via Modal cloud compute. Table I summarizes hyperparameters.

TABLE I
HYPERPARAMETER SETTINGS FOR EACH
EXPERIMENTAL CONDITION.

Task	Mode	Epochs	LR	Batch
SST/CFIMDB	LLL	10	10^{-3}	64/8
SST/CFIMDB	FM	10	10^{-5}	64/8
SST/CFIMDB	L+S	30	10^{-4}	64/8
Quora	FM	5	10^{-5}	8
Quora	L+S	15	10^{-4}	64
Sonnet	LoRA	30	10^{-4}	8

C. Results

Sentiment classification. Table II summarizes results. LoRA+SMART achieves 0.518 on SST-5, slightly exceeding full fine-tuning (0.515) despite training only ~ 300 K parameters versus 124M. CFIMDB performance with LoRA+SMART (0.976) is slightly below the full fine-tuning ceiling (0.984).

Paraphrase detection. Table III presents results. Full fine-tuning achieves 0.895 dev accuracy, peaking at epoch 3 and slightly declining at epoch 4 (0.892), indicating overfitting. Our LoRA+SMART model achieves 0.856 dev and 0.833 test accuracy after 15 epochs. While below the full fine-tuning baseline, this represents 95.6% of its dev accuracy using only ~ 295 K trainable parameters. The gap is attributable to

TABLE II
SENTIMENT CLASSIFICATION DEV ACCURACY.

Task	Mode	Epochs	Dev Acc
SST-5	LLL	10	0.453
SST-5	FM	10	0.515
SST-5	L+S	30	0.518
CFIMDB	LLL	10	0.878
CFIMDB	FM	10	0.984
CFIMDB	L+S	30	0.976

TABLE III
PARAPHRASE DETECTION RESULTS. FM =
FULL-MODEL; L+S = LoRA+SMART.

Mode	Epochs	Dev Acc	Test Acc
FM	5	0.895	—
L+S (ours)	15	0.856	0.833
Staff baseline		0.882	0.856

the large dataset (283K examples) where full model capacity provides a tangible advantage.

Sonnet generation. Table IV shows our LoRA model achieves chrF of 41.455 (test) and 41.881 (dev), both exceeding the staff baselines of 41.208 and 41.414 respectively. LoRA’s parameter-efficient adaptation allows the model to specialize to the sonnet domain while retaining broader language modeling capabilities, preventing overfitting on only 154 training examples.

TABLE IV
SONNET GENERATION CHRf SCORES (HIGHER IS
BETTER).

Model	Dev chrF	Test chrF
Staff baseline	41.414	41.208
GPT-2 + LoRA (ours)	41.881	41.455

D. Paraphrase Training Dynamics

Table V presents epoch-by-epoch dynamics for LoRA+SMART on paraphrase detection. Dev accuracy improves steadily from 0.797

to 0.856, with training loss decreasing monotonically throughout all 15 epochs. The absence of overfitting, unlike the full fine-tuning baseline which peaks at epoch 3 demonstrates that SMART’s smoothness constraint effectively prevents the model from memorizing spurious patterns.

TABLE V
LoRA+SMART PARAPHRASE TRAINING DYNAMICS
(LR= 10^{-4} , BATCH 64).

Epoch	Train Loss	Dev Acc
0	0.532	0.797
3	0.386	0.833
6	0.361	0.845
9	0.346	0.851
11	0.338	0.854
13	0.333	0.856
14	0.329	0.856

VI. ANALYSIS

LoRA rank and parameter efficiency. With rank $r = 8$, our LoRA implementation introduces $2 \times 12 \times (768 \times 8 + 8 \times 768) = 294,912$ trainable backbone parameters, a 99.7% reduction from full fine-tuning. Despite this, SST-5 performance with LoRA+SMART (0.518) slightly exceeds full fine-tuning (0.515), consistent with the intrinsic dimensionality hypothesis of Aghajanyan et al. [14]: the task-relevant update lies in a low-dimensional subspace, and LoRA explicitly parameterizes this.

Dataset size and the limits of LoRA. The paraphrase results reveal an important limitation: with 283K training examples, full fine-tuning (0.895 dev) substantially outperforms LoRA+SMART (0.856 dev). We hypothesize that on large datasets, the low-rank constraint becomes a bottleneck—there is sufficient data to justify updating all 85M attention parameters, and the low-rank approximation leaves useful capacity untapped. Table VI summarizes this dataset-size interaction.

Complementarity of LoRA and SMART. LoRA regularizes the parameter space by

TABLE VI
LoRA+SMART VS. FULL FINE-TUNING ACROSS TASKS.

Task	Train Size	FM	L+S	Δ
SST-5	8,544	0.515	0.518	+0.003
CFIMDB	1,707	0.984	0.976	−0.008
Quora	283,003	0.895	0.856	−0.039
Sonnet	154	–	41.455	+0.247 vs. baseline

constraining updates to a low-rank manifold. SMART regularizes the function space by penalizing output sensitivity to input perturbations. These are orthogonal constraints: a low-rank update can still produce a sensitive function, and a smooth function can be achieved through a full-rank update. The paraphrase training dynamics demonstrate a concrete benefit of SMART: unlike full fine-tuning which peaks at epoch 3 and overfits, LoRA+SMART shows no overfitting across 15 epochs, with loss and dev accuracy both improving monotonically.

Training dynamics and BPO. The slower convergence of LoRA+SMART (requiring more epochs than full fine-tuning) reflects the regularized objective. BPO stabilizes training by preventing large steps that would otherwise cause loss spikes when the SMART gradient and task gradient conflict. Without BPO, pilot runs showed periodic instability. With BPO ($\mu = 0.001$), training is stable across all tasks.

Sonnet generation qualitative analysis. Lower temperatures ($T = 1.0$) produced repetitive outputs with phrases like “thy beauty” and “summer’s day” dominating. Higher temperatures ($T \geq 1.5$) introduced incoherent sequences. The value $T = 1.3$ was identified as the empirical sweet spot. The chrF metric’s character-level sensitivity captures partial credit for archaic word forms (“thee”, “thine”, “hath”), making it well-suited for Shakespearean evaluation.

Limitations. We explored only rank $r = 8$; a sweep over $r \in \{4, 8, 16, 32\}$ might reveal

whether higher ranks close the gap with full fine-tuning on Quora. Additional compute for paraphrase training would likely yield further improvement given the still-decreasing loss at epoch 14.

VII. CONCLUSION

We investigated parameter-efficient fine-tuning of GPT-2 via LoRA and SMART across three NLP tasks. Results showed: (1) LoRA+SMART matches or exceeds full fine-tuning on small datasets, 0.518 vs. 0.515 on SST-5, 41.455 vs. 41.208 chrF on sonnet generation-using 99.7% fewer backbone parameters; (2) on large-scale Quora (283K examples), full fine-tuning retains a 3.9-point advantage, revealing that LoRA’s low-rank constraint becomes a bottleneck when data is plentiful; (3) SMART regularization prevents the overfitting observed in full fine-tuning on paraphrase detection, enabling continued improvement over 15 epochs. These results suggest a practical guideline: LoRA+SMART is most beneficial when data is scarce and regularization is critical. Future work should explore higher LoRA ranks for large datasets and per-task tuning of SMART hyperparameters.

REFERENCES

- [1] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *Proc. ICLR*, 2022.
- [2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
- [3] T. Brown *et al.*, “Language models are few-shot learners,” in *NeurIPS*, 2020, pp. 1877–1901.
- [4] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” 2018.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.
- [6] N. Houlsby *et al.*, “Parameter-efficient transfer learning for NLP,” in *Proc. ICML*, 2019, pp. 2790–2799.
- [7] X. L. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” in *Proc. ACL-IJCNLP*, 2021, pp. 4582–4597.
- [8] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, “SMART: Robust and efficient fine-tuning for pre-trained natural language models,” in *Proc. ACL*, 2020, pp. 2177–2190.
- [9] A. Vaswani *et al.*, “Attention is all you need,” in *NeurIPS*, 2017, pp. 5998–6008.
- [10] R. Socher *et al.*, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Proc. EMNLP*, 2013, pp. 1631–1642.
- [11] M. Popović, “chrF: character n-gram F-score for automatic MT evaluation,” in *Proc. WMT*, 2015, pp. 392–395.
- [12] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, “The curious case of neural text degeneration,” in *Proc. ICLR*, 2020.
- [13] A. Wang *et al.*, “GLUE: A multi-task benchmark and analysis platform,” in *Proc. EMNLP Workshop BlackboxNLP*, 2018.
- [14] A. Aghajanyan, S. Gupta, and L. Zettlemoyer, “Intrinsic dimensionality explains the effectiveness of language model fine-tuning,” in *Proc. ACL-IJCNLP*, 2021, pp. 7319–7328.
- [15] X. Li and X. Liang, “Explicit inductive bias for transfer learning with convolutional networks,” in *Proc. ICML*, 2018, pp. 2825–2834.
- [16] X. Zhang and M. Lapata, “Chinese poetry generation with recurrent neural networks,” in *Proc. EMNLP*, 2014, pp. 670–680.